DURHAM UNIVERSITY

MASTERS OF RESEARCH

Statistical modelling of clickstream behaviour to inform real-time advertising decisions

Author:
Ryan JESSOP

Supervisor:
Dr. Jonathan CUMMING

A thesis submitted in fulfillment of the requirements for the degree of Masters of Research

in the

Department of Mathematics

September 4, 2020

Declaration of Authorship

I, Ryan JESSOP, declare that this thesis titled, "Statistical modelling of clickstream behaviour to inform real-time advertising decisions" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

DURHAM UNIVERSITY

Abstract

Department of Mathematics

Masters of Research

Statistical modelling of clickstream behaviour to inform real-time advertising decisions

by Ryan JESSOP

Online user browsing generates vast quantities of typically unexploited data. Investigating this data and uncovering the valuable information it contains can be of substantial value to online businesses, and statistics plays a key role in this process.

The data takes the form of an anonymous digital footprint associated with each unique visitor, resulting in 10^6 unique profiles across 10^7 individual page visits on a daily basis. Exploring, cleaning and transforming data of this scale and high dimensionality (2TB+ of memory) is particularly challenging, and requires cluster computing.

We outline a variable selection method to summarise clickstream behaviour with a single value, and make comparisons to other dimension reduction techniques. We illustrate how to apply generalised linear models and zero-inflated models to predict sponsored search advert clicks based on keywords.

We consider the problem of predicting customer purchases (known as conversions), from the customer's journey or clickstream, which is the sequence of pages seen during a single visit to a website. We consider each page as a discrete state with probabilities of transitions between the pages, providing the basis for a simple Markov model.

Further, Hidden Markov models (HMMs) are applied to relate the observed clickstream to a sequence of hidden states, uncovering meta-states of user activity. We can also apply conventional logistic regression to model conversions in terms of summaries of the profile's browsing behaviour and incorporate both into a set of tools to solve a wide range of conversion types where we can directly compare the predictive capability of each model.

In real-time, predicting profiles that are likely to follow similar behaviour patterns to known conversions, will have a critical impact on targeted advertising. We illustrate these analyses with results from real data collected by an Audience Management Platform (AMP) - Carbon.

Acknowledgements

I would first and foremost like to thank my academic supervisor, Jonathan Cumming, for his guidance throughout my research. His invaluable support has taught me to be analytical and critical in the search for solutions, and fuelled my passion to continue a career in statistics and data science.

I would like to thank Knowledge Transfer Partnerships and Innovate UK for the funding of this research, and sponsoring my Masters by Research. May these grants continue to support innovation and technological projects for many others.

I would like to acknowledge my colleagues from Clicksco and Carbon, for all of the opportunities I was given to conduct my research and develop models for a tech start-up. Specifically, many thanks to my company supervisor, Alistair McLean, and team lead, Ryan Varley, for guiding the direction of my research.

I would like to thank my family for their support in my academic journey, particularly for offering welcome distractions during long weeks of my day job and weekend masters research and writing.

Thank you to my girlfriend, Chloe, for her continuing support in my career and personal development, and for proof reading my work, for which I am very grateful.

A final thanks to Rountons coffee who provided the coffee beans in all the cafes I frequented throughout the write-up.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Sponsored keyword search

The majority of readers will have used a search engine, such as Google, when browsing for a product, brand or company online. When you enter a search term, you will generate a search engine results page (SERP) based on the word or phrase you queried. A search engine will serve an advertisement in response to a search term on the SERP, to promote relevant websites. An illustration of a SERP is provided in Figure 1.1, where the top of the page contains the search term the user entered, and the highlighted area represents where the advertisements are placed. This content is also known as a sponsored listing to differentiate it from the organic, or non-sponsored, listings on the SERP. An advert will typically contain a set of advertiser hyperlinks annotated with tags, titles, and a description.

FIGURE 1.1: An illustration of a search engine results page (SERP).

Typically, the search terms are linked to **keywords** that provide an indication as to what the user wants from their browsing session. Then a pre-determined advert, from a specified advertising campaign, is displayed based on the search terms. Digital marketing specialists are involved in designing and building the advert content and the wider advertising campaign structure. For example, “best running shoes“ is a search term that a user may enter into a search engine such as Google. Advertisers then will have a set of predefined keywords that wish to bid for to display an advert for their product. In this scenario, an advertiser such as Nike, may wish to pay a high price to display an advert on the keyword ‘running shoes’, and direct the user to their website.

To create a sponsored search advertising campaign a team of specialist content providers select terms and search phrases (i.e. keywords) that they believe are likely to be submitted by searchers. These keywords will be applicable to their web content, and will be related to the underlying intent of the searcher [1]. An advertising campaign will have a pre-arranged budget and could run over several weeks. A

campaign will have a collection of keywords, where a keyword can consist of multiple words to make a phrase. These content providers can also tailor the presentation of the search results to target specific search terms, by presenting several variations linked to particular sets of queries within a campaign. Sponsored search can be used by any person or organisation interested in generating user traffic to a particular website for some specific purpose. We could view sponsored search as a version of providing relevant content to a searcher, and not solely as an advertising medium [2].

Displaying an advert to a user is called an **impression**. If the advert appeals to the user, then the advertiser hopes the user will **click** on the advert. In some cases, there is an extra goal added after the user has clicked on the advert, which is named a conversion. An example could be to complete a subscription form for a magazine. In sponsored search advertising, the campaign will specify the price the advertiser is willing to pay to show an impression on the SERP for all specified keywords.

The data in Section 2.2 that we will study is provided by Google Ads (formerly known as Google Adwords), on a number of advertising campaigns that are currently operated by Clicksco. Clicksco is a marketing technology company that uses sponsored search advertising to display adverts for a range of price comparison sites that they own. The search engine will distribute the bidding on keywords over the period of the campaign, and based on the bids will rank the adverts in order on each SERP in real-time. There is a search engine review process to ensure that the provider's content is relevant to the targeted keyword and corresponding contextual description, providing a set of quality metrics for the advert. It is important to note that in sponsored search, advertisers only pay for a click rather than for an impression.

1.2 Targeted intent advertising

When you search online, using a search engine such as Google, you have **intent**, i.e. you have an intention of finding something. On social media websites, such as Facebook, users provide their **interests** through interactions with other users, as opposed to direct search intent. Advertising and marketing teams target groups of individuals, called **audiences**, based on their interests and habits, what they're actively researching or how they have interacted with a product or brand [3].

One major difference in the industry is the assessment of online user interests and user intent. Interest-based targeting is generally used for audiences where the advertiser wants to introduce a product to people who haven't heard of them before [4]. A user's intent with respect to a particular category may be updated over time, and is directly related to making a purchase of a product. As such, based on subsequent user behaviour, a new intent-strength score may be assigned to the user. This value can be used to rank and segment audiences based on the user's classified intent. One example could be that a user's intent-strength score changes according to a function. This function may vary according to one or more variables, including time, product characteristics, user characteristics, and other variables affecting a user's intent with respect to a particular product or category [5]. This score exhibits more than whether or not a user is in-market to make a purchase; it assesses where on that journey the user currently is.

In this thesis, we will use data that is provided by Carbon, a cloud-based Audience Management Platform (AMP). Carbon is a product for companies that own web sites, and want to learn about the intent and interests of users on their sites.

Carbon collects online click data on the partner sites, and we first explore this data in Section 2.3. As a statistician at Carbon, our research aims to develop solutions for online browsing data that can be incorporated into its toolset [6]. One solution that Carbon offers is to build an audience of profiles that an advertising company can decide to target with an advertising campaign. For example, a sports brand may wish to advertise to an audience that are interested in sportswear: Carbon offers a solution that ranks the profiles' intent-based on historical browsing behaviour.

Online behaviour is collected using third-party cookies. A third-party cookie is placed on a user's hard disk by a website from a domain other than the one a user is visiting [7]. Third-party cookies are used to remember something about the user at a later time, i.e. personalised preferences. Advertising companies, such as Carbon, use the cookie to identify a user in targeted advertising. The data is fully anonymised adhering to GDPR regulations; namely the data is unidentifiable and the process is irreversible [8].

1.3 Technologies beyond R

When scoping the research project area, we swiftly became aware that the size of the data would make accessing the problem challenging. To build successful applied statistical models, we would need to expand our knowledge and expertise into new technologies. As a data scientist working at Carbon, collaborating with developers and utilising a range of the cutting-edge technologies for Big Data processing and analysis is essential to finding and creating solutions.

When data is available to store on a hard drive, we can use R to analyse (<1GB). When the data is terabytes we must use Spark for query and analysis work [9]. The majority of the code used in the thesis is written in R, for data exploration, visualisation and building statistical models. To build the data sets for analysis in R, we transform and sample data exploiting cluster-computing using **Apache Spark**. All code relating to this thesis can be found on GitHub [10].

Apache Spark is a popular open source framework that ensures data processing with lightning speed and supports various languages like Scala, Python, Java, and R. Spark is capable of handling several petabytes of data at a time, distributed across a cluster of possibly thousands of cooperating physical or virtual servers. It exhibits very high performance but from the end user's perspective, appears like working on a standalone system. Resilient Distributed Datasets (RDDs) are the building blocks of any Spark application, meaning they are fault tolerant and partitioned [11]. The data can be stored in a format called Apache Parquet, which is efficient as well as performant flat columnar storage compared to row based formats, e.g. CSV [12].

1.4 Other research on clickstream data

A **clickstream** (or click path) is an ordered sequence of page visits (hyperlinks) that a visitor follows on a website - a browsing journey. Clickstream data records the activity on the Internet, on every web page within a web site that is visited, the length of each page visit and the order the pages were visited. Statistical analysis of clickstream data is a good example of data science, which is particularly useful for web activity analytics, advertising and marketing, and improving profit of e-commerce sites. The relationship between the frequency of visits to a web site and purchasing intent in e-commerce online browsing has been well examined [13].

Applying Markov chain type models to clickstream data is well researched in the literature, where clickstreams can be viewed as browsing page type transitions [14]. Many visualisation tools have been created to display fitted Markov chains graphically for users to interact with and understand [15], [16].

Further, the clickstream journeys may be categorised and modelled using a dynamic multinomial probit model of web browsing [17]. A type II tobit model has been fitted to clickstream data to attempt to model the probability of continuing the journey on the site and length of time of viewing each page [18].

More complex statistical modelling has been applied to clickstream data, specifically hidden Markov models (HMMs) [19]. Nested HMMs are used to model the behaviour of returning users within session behaviour and between session behaviour separately [20]. Bayesian HMMs have also been investigated to predict the probability that a browsing session contains an online purchase [21].

1.5 Outline of thesis

In Chapter 1 we introduce the underlying concepts of online advertising and the main types of data that we will be exploring in subsequent chapters.

Chapter 2 discusses the data sets in detail, assessing distributions, associations and how we create the key covariates for modelling. Exploring, cleaning and transforming data of the scale of 10^7 observations on a daily basis and high dimensionality (2TB+ of memory) is particularly challenging. We study real data collected by an Audience Management Platform (AMP) - Carbon.

Chapter 3 aims to solve a problem for intent-based targeted advertising, using a variable selection method. We outline a variable selection method to summarise clickstream behaviour with a single value, and make comparisons to other dimension reduction techniques.

Chapter 4 outlines GLM theory and applies both logistic and Poisson regression to a data set relating to sponsored search from online search terms. This extends to zero inflated models that may provide more accurate modelling of our data set, where our data exhibits an excess number of zeros.

Chapter 5 describes a user's journey or clickstream, as a discrete state with probabilities of transitions between the page visits, providing the basis for a simple Markov model. Further, Hidden Markov models (HMMs) are applied to relate the observed clickstream to a sequence of hidden states, uncovering meta-states of user activity.

Chapter 6 considers the problem of predicting customer purchases (known also as conversions). We discuss the robustness and sensitivity of training HMMs. We apply conventional logistic regression to model conversions in terms of summaries of the profile's browsing behaviour. We combine the modelling approaches of GLMs, Markov chains and HMMs into a set of tools to solve a wide range of conversion types and produce an informative plot to compare and evaluate the predictive capability of each model type. In real-time, predicting profiles that are likely to follow similar behavioural patterns to known conversions, will have a critical impact on targeted advertising. Chapter 7 summarises the findings of this thesis and critically evaluates the real-time prediction models that inform advertising decisions.

Chapter 2

Introduction to Adwords and Clickstream data

2.1 Chapter overview

In this chapter, we will introduce the two data sets that will form the application aspect of this thesis. The first section will analyse a data set relating to the performance of an online advertising campaign. This is a static keyword performance report over a period of time and consists of numerical and categorical variables. We will provide results from the exploratory data analysis, cleaning and preprocessing steps that we will use to prepare the data for modelling. We will look at some association tests and correlation plots to help guide us in later chapters.

The second section discusses a data set relating to online browsing behaviour, which in its original form is big data with many complex data types. Here, we dive into a sample of the data at a specific point in time, which helps to provide a representative overview. We display basic plots of the key variables, and further assessments of any associations between them. We will create and explore additional numerical variables created through aggregations of the data. Further statistical modelling of these variables, after cleaning, enhances our knowledge of the data set.

2.2 Adwords report

2.2.1 The data

The *Adwords Report* dataset is a daily keyword report, provided by Google Ads (formerly known as Google Adwords), on a number of advertising campaigns that are currently operated by Clicksco. A keyword search advertising campaign will have a pre-arranged budget and will run over a number of weeks; there will be marketing and data specialists involved in designing and building the campaign structure and content. A campaign will have a collection of keywords and a pre-determined advert to display following searches for those keywords. Each row in the data set relates to a unique keyword and the performance of that keyword over a specified hour, including the number of clicks the advert receives - the dependent variable. A keyword can consist of multiple words to make a phrase. The content of the advertisements are provided by a company, who provide financial advice on pensions, hence the campaign uses financial keywords. We would like to model the number of clicks for a keyword, as a function of the key variables that we can influence. The model we will develop will aim to influence the bidding mechanism to purchase the opportunity to display an advert on a given keyword.

The data set provided for analysis has 40 variables and approximately 1.8 million rows, including some variables that are redundant which we remove immediately. The input covariates are parameters to the bidding algorithm which affect how the bidding process operates. The output variables provide the key performance indicators for the keyword over the specified hour and cumulatively throughout the day. Further, the quality score variables are provided by Google, which may be relevant to learning about the keyword's performance. We see a visual representation of the main variable groups in Figure 2.1.

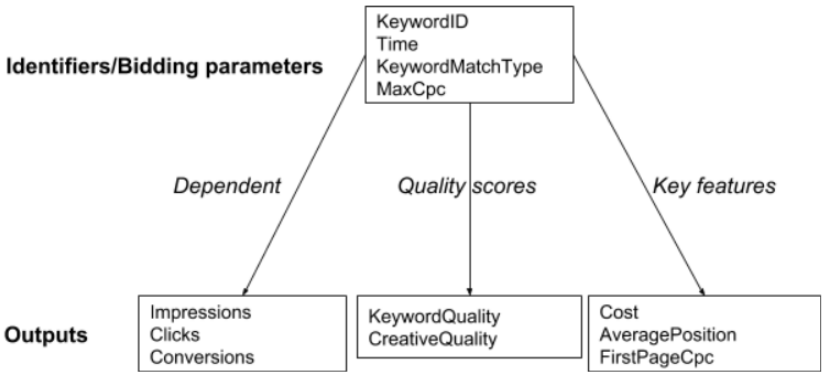


FIGURE 2.1: Separation of the key variables based on their definitions.

There are up to three **identifier** variables: *KeywordID*, *Time* and *CampaignID*. The *Time* (t) provides the timestamp for the hour when the keyword was active. We will concentrate this section on a single advertising *CampaignID*, such that each row only requires the *KeywordID* (k) and t to determine a unique row. An example of a **bidding parameter** covariate is *KeywordMatchType*. A *broad* match means a bid will be placed for the keyword if a subset of the words in the keyword phrase are matched in the user's search term. An *exact* match keyword has to match all of the words in the the search term precisely for a bid to be placed.

For the **dependent** variable group, we are provided with a set of numerical variables which are the performance metrics of the keyword over the specified hour (indicated by the term *Delta*). An **impression** is a term that refers to the point in which an advert is viewed once by a user, or displayed once on a web page. A **click** is defined as a single click on an impression. A **conversion** is when a user has clicked on the advert, and continues to complete a further goal, e.g. fill in contact form, the sale of a product or subscription; in this instance a form is required to be filled in on the website. We expect these variables to be positive integers and are the underlying dependent variables. We will use notation to refer to the key variable's impressions, clicks and conversions as x_{kt}, y_{kt} and z_{kt} respectively. We can derive the cumulative versions of each dependent variable by summing the values for the current day, up to the current hour. For example, the number of cumulative impressions for a keyword at time t is $\sum_{i=1}^{t} x_{ki}$. The cumulative variables are then reset to zero at midnight.

There are a range of other variables provided by Google's internal algorithms, the **quality score** variables, which aim to provide a deeper insight into both the performance of the keyword and advert. *CreativeQuality* is a heuristic score of the similarity between the advert text (the 'creative') and landing web page. A high score for *CreativeQuality* indicates the advert content and the webpage are well matched.

Variable name	Type	Definition	Notation
AveragePosition	Numerical	The average position of the ad in the previous hour	p
Clicks	Numerical	The daily cumulative number of clicks	$\sum_{i=1}^{t} y_{ki}$
ClicksDelta	Numerical	The number of clicks in the previous hour	y_{kt}
Conversions	Numerical	The daily cumulative number of conversions	$\sum_{i=1}^{t} z_{ki}$
ConversionsDelta	Numerical	The number of conversions in the previous hour	z_{kt}
Cost	Numerical	The daily cumulative cost of all ad clicks from the keyword	
CreativeQuality	Categorical	A score for the similarity between the advert content and landing web page	
FirstPageCpc	Numerical	An estimate for the cost to win a bid for the ad to be displayed on the first Google search page	
Impressions	Numerical	The daily cumulative count of the number of ads displayed	$\sum_{i=1}^{t} x_{ki}$
ImpressionsDelta	Numerical	The count of the number of ads displayed in the previous hour	x_{kt}
KeywordID	Integer	A unique identifier for the keyword search term	k
KeywordMatchType	Categorical	The keyword search match type - Exact or Broad (Non-exact)	
KeywordQuality	Categorical	A score for the similarity between the keyword and landing web page	
MaxCpc	Numerical	The maximum cost per click we are willing to pay to bid to win the ad impression	
Time	Timestamp	The end of the hour relating to the performance measures for the previous hour	t

TABLE 2.1: A selection of variables and definitions from *Adwords Report*.

The *KeywordQuality* is the assigned similarity score between the keyword and the landing page. Similarly, a high score for *KeywordQuality* indicates the advert content and keyword are well-suited.

An example of a key feature of the bidding process is the position ranking on the search engine results page (SERP). A numerical variable *AveragePosition* (p) calculates the average position of the advert for any impressions in the previous hour. Predicted and actual costs for the ad clicks are also provided by Google in the keyword reports. An estimate provides the cost if an advert was clicked on the first page of search results - *FirstPageCpc*. The *Cost* of any clicks are provided - note that here costs are only incurred if there is a click on the advert.

The variables we can control from a bidding perspective are the collection of keywords and the variable *MaxCpc*. This is the maximum cost per click for the keyword that we are willing to pay. The value for the *MaxCpc* for which the keyword is deemed **low volume** and **mature** is 2.22, which accounts for around 80% of the original rows in the data set. Further investigation showed that these keywords have zero or very few impressions and almost zero clicks. The value of 2.22 is arbitrary and prevents the keyword from costing money on redundant keywords, as the value is too small to win in the auction. However, 10% of the keywords have their value of *MaxCpc* boosted for approximately 10 days to see if their performance improves.

Due to the low activity, we will remove the instances of these inactive keywords and concentrate the analysis on active keywords.

We did not use all the variables in the original *Adwords Report* data set, as we found it important to concentrate on the initial set of well defined variables in Table 2.1. The choice of variables to focus on was further driven by expert knowledge of potentially effective predictors in modelling clicks [22],[23]. Even so, there are some variables out of our control, for example, Google can update the quality scores at any time. There are basic definitions that Google provide, such as a zero score for *KeywordQuality* means a score is unassigned, so by removing these observations we create a more reliable model using active keywords. A not applicable ('NA') score for *CreativeQuality* translates that Google is yet to assign a score to the keyword and advert text as it is new or is not recently active. For example, there could be a keyword with no impressions and still a quality score if Google has provided an assessment, and a keyword with impressions but no score if Google has deemed there is not enough data available to make a judgement. Hence, this makes them unreliable for use in modelling and the missing ('NA') scores exist in approximately 60% of rows in the original data set. This blog post sets the context and consequences of a good Adwords quality score [24].

There are six campaigns within this dataset, hence we chose a single campaign (using the *CampaignID* identifier) and randomly sampled $100,000$ observations (the total size of the campaign was approximately $250,000$), where the variables contain no missing entries. This allowed us to have a large representative sample for statistical tests and plots.

2.2.2 Exploratory data analysis

The data sample is concentrated to two weeks in August 2017, where the date range of the campaign can be seen in Figure 2.2a, which is a histogram of keyword frequencies over time. The distribution is not uniform across the campaign, highlighting a common problem that we must experiment with high volumes of active keywords and then refine as we learn more about their performance. A variable we can control in the bidding process is the binary variable *KeywordMatchType*, which can either be exact or broad. We have almost a $50/50$ split between the two types (see Figure 2.2b). We can use this variable as a simple categorical predictor in the model. The symmetry here is by design as all keywords are used for both exact and broad match types. Figure 2.2c shows the distribution of impressions with a logarithmic y-axis scale. We see a rapid decay shape, with a high volume of zeros and long tail to the right, further highlighted in Table 2.2 with the maximum value of 626 for *ImpressionsDelta*. These high values are concerning, as the potential incurred cost due to clicks would be significant.

Next we discuss the key variables of clicks and conversions (y_{kt} and z_{kt}), noting the high number of zeros in Figures 2.2d and 2.2e and Table 2.2. A 'real' or 'genuine' zero is generated if $x_{k,t} > 0$, hence we obtain a count distribution for y and z, where the values could be zero. Whereas a 'structural' zero is generated when $x_{kt} = 0$, hence $y_{kt}, z_{kt} = 0$; as it is impossible to see clicks or conversions when there are zero impressions. We plot only the individual distributions containing the 'real' zeros. The distribution does have a tail to the right, however the tail is less dramatic when compared to the impressions distribution in Figure 2.2c. From Figure 2.2e and Table 2.2, we see that $\max z_{kt} = 2$, and approximately only 1% of keywords with at least one impression actually realised a conversion, which is a common problem in online advertising. The average position variable, p, ranges continuously from $1 \leq p \leq 7$,

Variable	Min	LQ	Median	UQ	Max
Time	2017-08-16	2017-08-18	2017-08-20	2017-08-26	2017-10-01
HourOfDay	0	6	12	18	23
ImpressionsDelta	0	0	0	0	626
ClicksDelta	0	0	0	0	16
ConversionsDelta	0	0	0	0	2
AveragePosition	1.0	2.0	2.6	3.6	7.0
MaxCpc	0	0	1	1	13

TABLE 2.2: A table to display numerical summaries of the key variables in the *Adwords Report* data set.

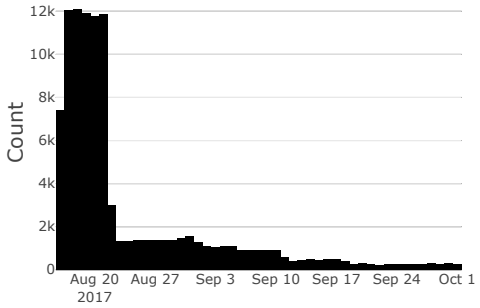
where again we only include keywords that generated at least one impression in Figure 2.2f. A low value ($1 \leq p \leq 2$) relates to the advert shown at the top of the SERP, whereas a high value ($6 \leq p \leq 7$) means the advert is displayed lower down the SERP. We may wish to discretize this variable to the top and bottom of the SERP, as that is where search adverts are placed. We can see from the numerical representation of the distribution in Table 2.2 that the majority of observations lie where $p \leq 3$. We can see the distribution in Figure 2.2f which shows peaks surrounding integers, as a result of computing an average from low or single impressions from keywords. We expect this p to highly correlate with costs and y_{kt}, i.e. a 'higher', more visible position on the SERP incurs a higher cost along with more visibility. Hence, we must be aware of this when using it as a predictor for clicks.

The histograms in Figures 2.2g and 2.2h display the distribution of the variables *CreativeQuality* and *KeywordQuality*. Both categorical variables contain a high proportion of missing values, which have been removed for the plots. The distribution is close to uniform across the creative quality variable, whereas the keyword quality scores have high proportions of low scores, with very few keywords obtaining the highest scores possible. Figure 2.2i shows a logarithmic scale distribution of the variable *MaxCpc*, where the majority of the observations lie at the highest cost.

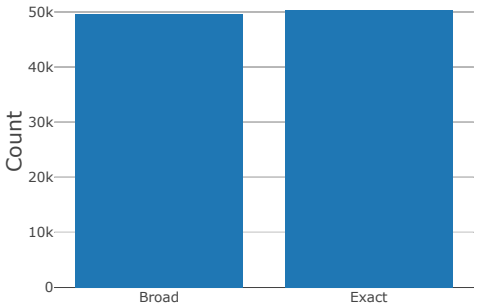
2.2.3 Relationships between variables

It is crucial to understand which variables may impact on the key dependent variables of impressions and clicks, hence we will use them to identify patterns and variable relationships. Continuing with the same sample data as in the previous section, the plots in Figure 2.3 demonstrate that there is potentially a weak linear relationship evident, with a high density of observations close to the origin, and a positive correlation as the number of impressions increase. We know that a successful keyword will achieve a high click-through-rate ($CTR = \frac{y_{kt}}{x_{kt}}$), as x_{kt} increases there are more opportunities the keyword has to achieve a higher value of y_{kt}. To further inspect this relationship, we use other explanatory variables to identify if there are other interactions with *KeywordMatchType*, *CreativeQuality*, *AveragePosition* and *HourOfDay* in Figure 2.3

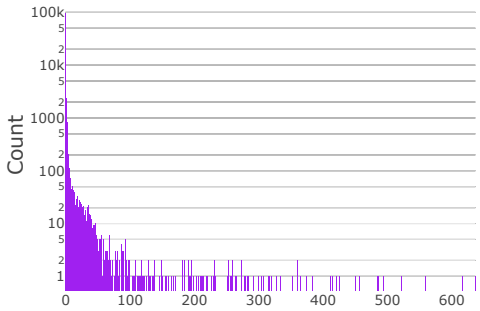
Figure 2.3a displays the two options for the keyword match type binary variable: *broad* or *exact*. From this we learn that the majority of high volume clicks are keywords with a broad match type. We know that 4% of broad match observations have at least one impression and 6% of exact matches have at least one impression. However, the broad match observations make up 81% of the total impressions shown. The broad match type keywords have a higher number of impressions due to a higher chance of matching the search term broadly rather than exactly. We see a



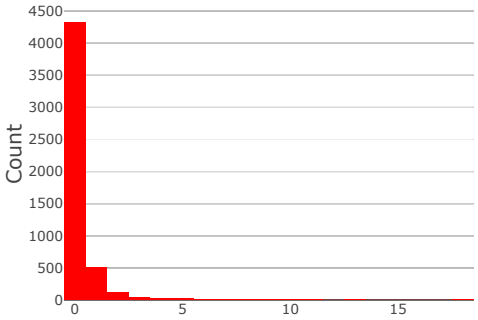
(a) Time, t



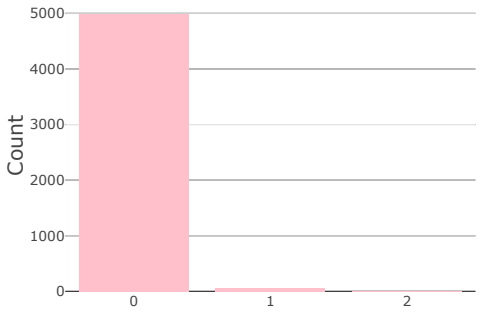
(b) KeywordMatchType



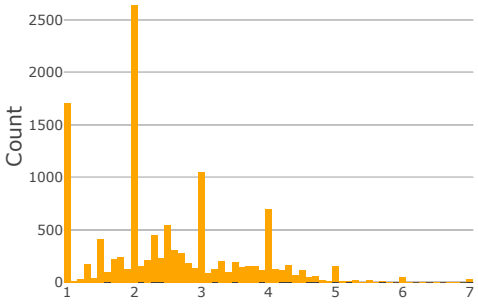
(c) ImpressionsDelta, x_{kt}



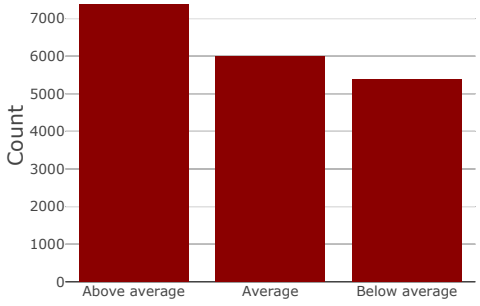
(d) ClicksDelta, y_{kt}



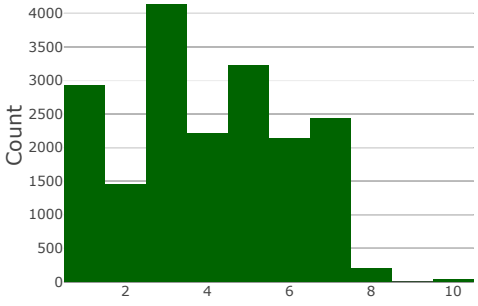
(e) ConversionsDelta, z_{kt}



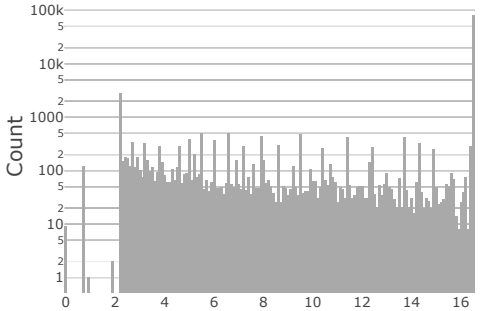
(f) AveragePosition, p



(g) CreativeQuality



(h) KeywordQuality



(i) MaxCpc

FIGURE 2.2: Graphical displays of key variable distributions in the *Adwords Report* data set.

large separation in behaviour in this scatter plot, note that broad match type keywords require a large number of impressions for a small increase in clicks. This is because the viewer may be shown an advert that has no link to their intended search and is ineffective in gaining their attention.

Figure 2.3b, coloured by the categorical creative quality variable, shows a separation of the *Above average* score from the dense region of observations close to the origin. This indicates a higher number of clicks and impressions are associated with a higher score for the advert text. The quality score is the output of a proprietary algorithm used by Google, and so its precise definition is unknown. However, it would be sensible to assume that the number of clicks is taken into account for the score. Keywords with a *Not applicable* or *Below average* score have a lower number of impressions and clicks, i.e. the advert matches the search term poorly, leading to a negative reaction towards the advert by the viewer.

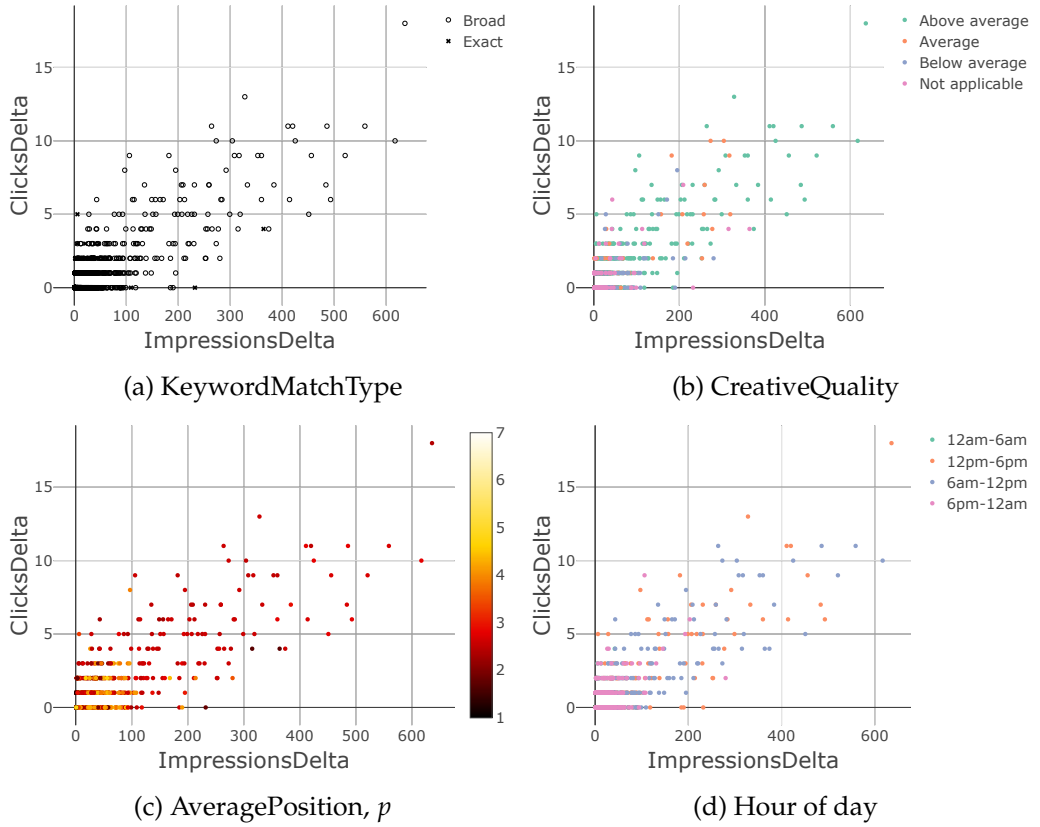
Figure 2.3c overlays the average position variable using a continuous sliding colour scale, separating high and low positions. As the number of impressions increase, we can see that the observations have a higher average position on the SERP. Low positions will have much less visibility, hence a lower chance to receive a click, as the viewer must scroll past multiple options that have a higher position on the SERP.

By splitting the observations based on the hour the data was collected, we obtain another variable which we should explore as this can greatly affect where the user is viewing the advert (i.e. at work or home) and the device type. We have discretized t into four intervals, each 6 hours in length, to look for any patterns with this new variable. The first thing to note in Figure 2.3d is the lack of observations from $12am - 6am$, making up only 4% of the total impression count. As this is a UK based advertising campaign, there are very few impressions for keywords through the night. Keywords active between $6am - 6pm$ make up 80% of the total impression count, whilst observations from $6pm - 12am$, tend to have fewer impressions and are clustered with low volumes of clicks. The morning and afternoon intervals make up 43% and 37% of the total impression count respectively. This makes for a well balanced choice for the time intervals. Also, t is split into equal intervals for simplicity, however alternative discretisations could be explored such as morning, working hours, evening and night.

After viewing the relationships between impressions, clicks, and other categorical and numerical variables, we look for correlations and associations which will inform us when deciding on potentially informative covariates for our model, while avoiding multicollinearity. First, we look for associations and correlations between numerical variables, a combination of positive integers and continuous values.

The pairs plot in Figure 2.4a, shows scatter plots of each combination of numerical variables. The associated plot Figure 2.4b summarises the relationship with the correlation coefficient for each scatter plot in Figure 2.4a. Clearly, the strongest positive correlation is between x_{kt} and y_{kt}, denoted by the largest positive value in the correlation plot. The average ad position is not strongly correlated with any numerical variable, which seems to contradict our expectation. Note that these values are looking for linearity, whereas there could be non-linear relationships apparent.

On the other hand, in Figure 2.4a we display the scatter plots of x_{kt}, y_{kt} vs *MaxCpc*, both showing a weak negative correlation. An intriguing relationship is displayed here, with high values for impression and clicks corresponding to low values for *MaxCpc*. This suggests that the lowest *MaxCpc* bid values may occur in unpopular keyword auctions. The scatter plot of p vs *MaxCpc* appears noisy with no immediate structure, but this relationship has a weak negative correlation of -0.41. This



(a) KeywordMatchType

(b) CreativeQuality

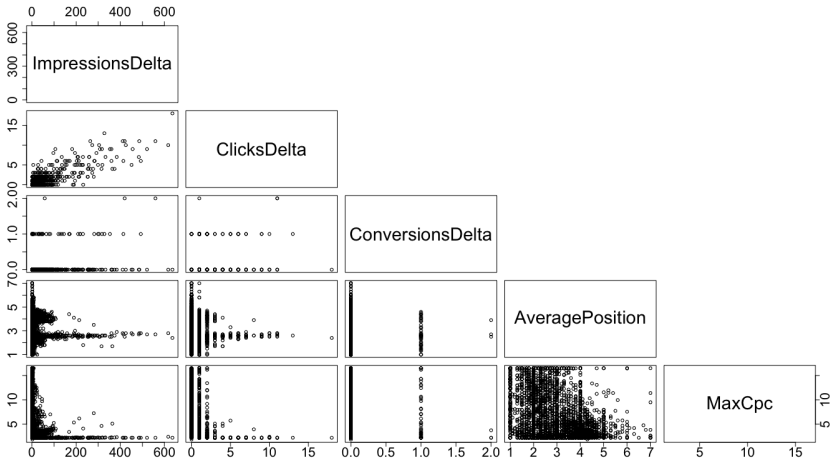
(c) AveragePosition, p

(d) Hour of day

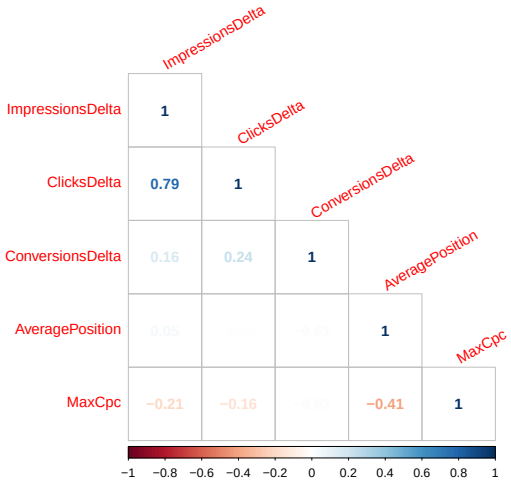
FIGURE 2.3: Four scatter plots to explore Clicks and Impressions against other key variables.

indicates that a high price for *MaxCpc* is associated with a low value for p, which is expected by definition as you pay more for a better ad position.

Next we investigate the keyword quality variable, in particular the relationship with other categorical variables. We build a heat map to represent the count distribution of categorical combinations, only providing information regarding values that have an interpretation, which are displayed in Figure 2.5. If we assume that a good keyword will be associated with the advert content and the landing page after a click, then we would expect a high score for keyword quality to correspond to a high score for creative quality. The darker areas in Figure 2.5a represent a high frequency of observations with the combination indicated in the plot. Overall, our assumption remains true as *Below average* scores correspond to scores of 3 or less; but a large number of observations have an *Above average* score and a score of 3, which is contradictory, hence we should be cautious when using quality scores as covariates in a model. Keyword quality has a high number of scores of 3, emphasised in Figure 2.5b, where both broad and exact match types display high volumes for a score of 3. The exact match type has a wider range of values for keyword quality, compared to broad match type.

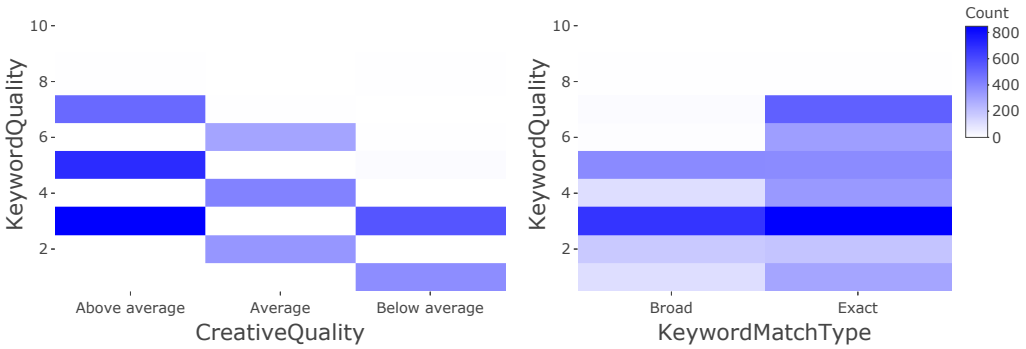


(a) Pairs plot



(b) Correlation plot

FIGURE 2.4: Plots to investigate correlations between the numerical variables.



(a) CreativeQuality *vs* KeywordQuality (b) KeywordMatchType *vs* KeywordQuality

FIGURE 2.5: Heatmaps to display the association between categorical variables.

2.2.4 Discussion

Through exploration of the *Adwords Report* data set we have discovered a range of different types of covariates and explored the performance indicators for keyword bidding advertising. From this analysis, we concentrate on the key dependent variables of clicks and conversions, and model these through a combination of independent variables to describe the advert content, keyword and other factors such as time or position of the advert. We will refer to the variable distributions and numerical summaries throughout other chapters to help determine how we wish to use the predictors in a model. The match type of the keyword clearly has a significant impact on the behaviour of the bidding system, with broad keyword matches providing a higher number of impressions than exact. Whereas, the categorical quality measures may appear unreliable as there are many missing observations.

We have removed many observations due to expert knowledge in identifying mature keywords that are redundant and inactive. We understand that impressions have a crucial impact on clicks, highlighted by a strong positive correlation. However, the unbalanced numbers of observations with and without impressions could give rise bias later when modelling. The impact of relatively few clicks in the sample will lead us to negotiate the modelling process carefully. In recent blog posts, we see click-through-rates in online advertising data are typically low, hence we will need a model that can handle a low number of positive labels and a range of variable types [25], [26].

2.3 Clickstream data

2.3.1 The data

A clickstream (or click path) is an ordered sequence of page visits (hyperlinks) that a visitor follows on a website - a browsing journey. Clickstream data records the activity on the Internet, on every web page within a web site that is visited, the length of each page visit and the order the pages were visited. Figure 2.6 describes an example of a clickstream. After a web browser has been chosen, the user faces a choice - to directly visit a web site or use a search engine. This may depend on whether the user has previously visited the web site, and hence they know the web address; or are searching for a range of options, and we call this the *Referrer*. Internet browsing begins when the user enters the website, an event known as the 'click-on'. Browsing will create a sequence of events corresponding to a path through the web site, where the journey concludes with a 'click-off' event when the user leaves the web site. We will use clickstream data in Markov chain and hidden Markov models in Chapter 5.

Page visit data consists of identifiers, timestamps and web page metadata. Figure 2.7 displays a collection of page visits that belong to the same continuous browsing journey, which we will call the browsing **session**. Multiple browsing sessions are collected over time to create a **profile**. The main variable types at a profile level are identifiers, device information and behaviour labels.

The *Clickstream* data set takes the form of an anonymous digital footprint (a profile) associated with each unique visitor to any of the web pages within the Carbon network. The anonymous page visit data is collected only from sites within the Carbon network, and when a profile visits one of these sites a browser cookie is sent from a website and stored on the profile's device by the web browser while the user

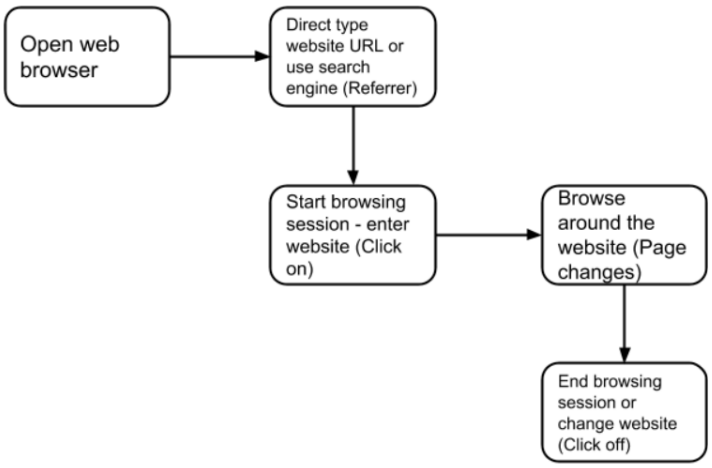


FIGURE 2.6: A diagram to display an example of a clickstream browsing journey.

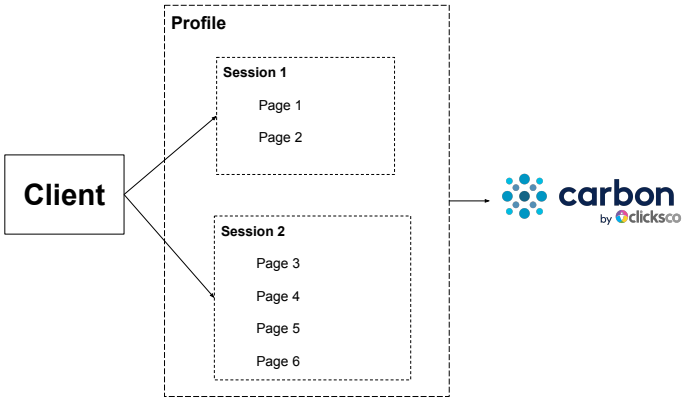


FIGURE 2.7: A diagram to show the distinction between profile, session and page visit (i, j, k) level data.

is browsing. The browser cookie is then used to identify the profile for personalised advertising.

The data set provided comprises 59 variables for each of 10^7 individual page visits associated with 10^6 unique profiles on a **daily** basis. Given the scale of this problem, the process of collecting, storing and managing this data is a non-trivial process. Table 2.3 provides the page visit level features, while Table 2.4 displays the variables at the profile level.

For profiles and sessions (notated by i and j respectively), as represented in Figure 2.7, we use unique identifiers to determine which sessions belong to a profile, and the associated page visit data. These are randomly generated and anonymise the data. We will use these later when predicting behaviour based on the covariates for particular users, browsing sessions (j) and page visits (k). Furthermore, as the data is all collected from the Carbon network, there is an identifier for the web site, *SiteID*, that we can use to investigate if behaviour differs across sites within the network. Time variables are important in clickstream data to determine the order of the page visits in a browsing journey, hence we require the start and finish time of a page visit - denoted by s_{ijk} and e_{ijk} respectively. A binary variable *PageVisitEndType*

describes the method that was used to collect the variable e_{ijk}, either a new page visit occurred or an artificial mechanism determines when the page visit is complete. A page visit may not have a finish time, if for example a browser tab remains open without activity. We will explore the data cleaning steps to retrieve an accurate time later in the chapter.

We obtain metadata that defines the page structure and provides information regarding the content of the web page. The data set provides both the current Uniform Resource Locator (*URL*), known universally as the web address, and if available the previously visited web address - the *Referrer*. In addition, an algorithm uses the web page content to provide a real-time categorisation to assign a label based on the page topic. The labels are provided by the Carbon taxonomy which contains over 4000 nodes in a hierarchical structure, providing a breath of categories for the web page to fall into. Some examples of the most popular Carbon taxonomy nodes are: shopping.couponing, consumer electronics.mobile phones and accessories and interest.news. The categories in the variable *PageVisitCategoryID* can be used to provide analysis on specific areas of interest, which we will exploit later in the modelling stage.

Alongside the clickstream data, we have features that exist at the profile level, for the collection of corresponding page visits. The time that the browser cookie was first placed on the profile's browser is collected in *ProfileCreated*, which helps us know the length of history we have for the profile. For all sessions and page visits, there are device features that stay constant and we will investigate these covariates in detail. A high level overview of the device can be found in the *DeviceType* categories, where devices are grouped by functionality, such as Smartphone, Tablet or Desktop.

We also have more granular device details, in particular the following categorical features: hardware, operating system, screen size and IP address. We will focus on *DeviceHardware* which provides the brand of the device, e.g. Apple, Samsung or Google, and *DeviceOS* gives the operating system, e.g. Windows, macOS or Android. These variables can be used in combination to provide a deeper insight into the profile's browsing behaviour. We expect the device to dictate certain types of behaviour as we use different devices for different purposes depending on our intentions of using the Internet. Further, we obtain an anonymised version of the *DeviceIP* address, which provides an approximate location from which we will investigate at the *Country* level only. An interesting problem known as 'cross-device tracking', arises from the need to connect similar behavioural patterns on a range of devices within the same IP address, but that is outside the scope of the thesis [27]. While it could be possible to connect device profiles to an individual using a statistical model, the most common method would be though directly via shared personal login details. The details would be shared across different devices, such as an individual accessing Facebook on their mobile and desktop. However, this personalised data is not available in this data set and so will not be considered further. However, a possible option for future research could be to build 'household' level data sets which combine all browsing data based on a common IP address.

An algorithm assesses the profile's behaviour over time to check for suspicious activity on the Carbon network. This extreme behaviour will be exhibited in the form of a high number of page visits in a short space of time, which cannot be replicated by a human. This behaviour is attributed to web scraping which is an automated process, usually gathering and copying targeted data into a a database, for later retrieval or analysis, and we represent this using the binary variable *Bot*. The *DeviceIP* can also help as known scrapers can be removed from further analysis. The variable *ProfileDemographics* provides any demographic labels that have been estimated from

Variable name	Type	Definition	Notation
PageVisitEndTime	Timestamp	The time representing the end of a page visit	e_{ijk}
PageVisitEndtimeType	Binary	A label attached to the *PageVisitEndTime* indicating a Real or Artificial page visit end type	
PageVisitID	Numerical	An identification number for the associated page visit	k
PageVisitCategoryID	Integer	A real-time categorisation of the webpage into the Carbon taxonomy	
PageVisitStartTime	Timestamp	The time representing the start of a page visit	s_{ijk}
ProfileID	Mixture	An anonymised unique identifier for the profile	i
Referrer	String	The last web address prior to loading the current page visit	
SessionID	Mixture	A unique identifier for the browsing session - a collection of page visits	j
SiteID	Categorical	A unique identifier for the website	
URL	String	The full web address associated with the page visit	

TABLE 2.3: A selection of variables and definitions at the *page visit* level.

Variable name	Type	Definition
Bot	Binary	A behavioural assessment of whether the profile is a bot
Country	Categorical	The country level location of the IP address
ProfileDemographics	Categorical	Third party information relating to a demographic label attached to the profile
DeviceType	Categorical	The type of device, e.g. Smartphone, Desktop
DeviceHardware	Categorical	The provider of the hardware/make of the device, e.g. Apple, Samsung
DeviceIP	String	An anonymised Internet Protocol (IP) address of the device
ProfileID	Mixture	An anonymised unique identifier for the profile
ProfileCreated	Timestamp	The time to represent when the *ProfileID* was first created

TABLE 2.4: A selection of variables and definitions from *profile* level data.

Carbon's algorithms based on browsing behaviour. Examples of the information this variable may provide are predictions of the profile's gender, age or profession.

2.3.2 Exploratory data analysis

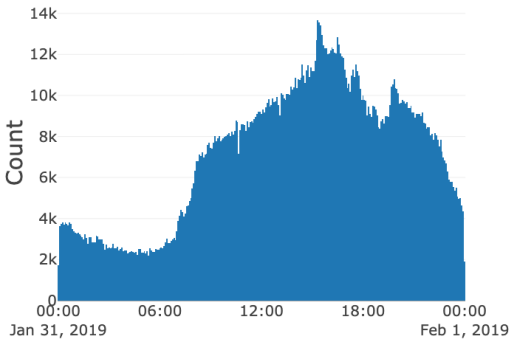
Due to the vast amount of *Clickstream* data we have available, we sample to bring the population to a more manageable size. We choose at random, approximately 3% of the total number of profiles that visit the Carbon network on the 31st January 2019. The resulting data set contains 287736 unique profiles, 392532 browsing sessions, and 2064446 individual page visits ranging across 6856 different web sites. Approximately 2% of the profiles are labelled as bots due to the known nature of their IP address, which we will exclude from any further analysis as their data does not resemble human behaviour.

Firstly, we will inspect individual variable distributions. In Figure 2.8a we display start time, s_{ijk}, across the entire day; note that we collect the timestamps in Coordinated Universal Time (UTC). As the data set ranges across a number of countries, as shown in Figure 2.8b, we must convert the timestamps before drawing conclusions from the distribution in Figure 2.8a. In the data set, there are IP addresses associated with over 200 different countries, as a result in Figure 2.8b we only display countries with over 2000 page visits. The United Kingdom and United States are the most populated categories, as the web sites that create the data target audiences in the UK and US.

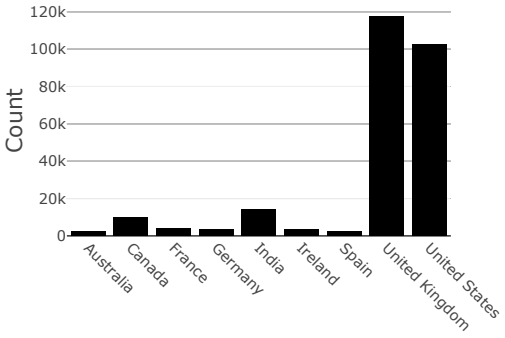
Discussing covariates at the page visit level data, we explore the binary variable for the page end time type, seen in Figure 2.8c, this highlights that 70% of recorded page visits have a *Real* end time, while the remaining page visits have an estimate for the finish time - labelled as *Artificial*. This label is created by a 'heartbeat', which checks for user activity of the web page. The heartbeat mechanism occurs when we cannot record when a page visit has officially ended, hence we must make an estimate. We can identify a known time when the user was still active on a web page using checkpoints (heartbeats) and estimate a real or artificial end time. We will explore the effect of the heartbeat in more detail later.

The distribution for the most populated page content category IDs is Figure 2.8d, where each category has over 20000 visits. The highest volume categories in this sample are varieties of clothing, shopping and interest. Only the main *Referrer* values are shown in Figure 2.8e, with the Internet giants Facebook and Google accounting for approximately 50% of the values, with other news sites' home pages attracting a modest proportion of the variable.

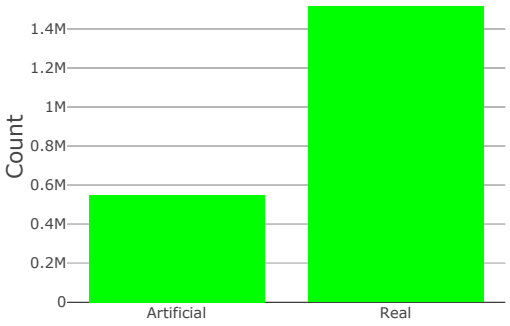
Switching to data aggregated at the profile level, all possible categories for the device types are shown in the plot in Figure 2.8f on a logarithmic scale. The largest categories by a substantial amount are desktop and smart phone, which combined make up almost 90% of the variable. Apple-based hardware leads the hardware variable in Figure 2.8g, closely followed by Samsung and other top brands from around the world, filtered with brands with more than 1000 unique profiles. We are observing the data unfiltered and raw, but expert knowledge has informed us that the number of profiles are inflated due the difficulty in tracking Apple and Mac devices. Each profile can be associated with multiple predicted demographic labels. The count for each individual demographic label is displayed in Figure 2.8h, which shows that the highest volume of labels are related to gender, followed by age, social grade and marital status. Not all profiles have a demographic label and this inconsistency in coverage must be taken into consideration if using this as a covariate in any modelling.



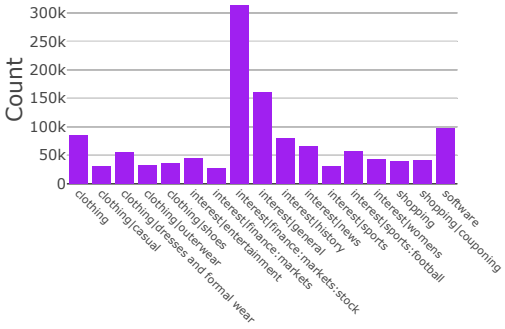
(a) PageVisitStartTime, s_{ijk}



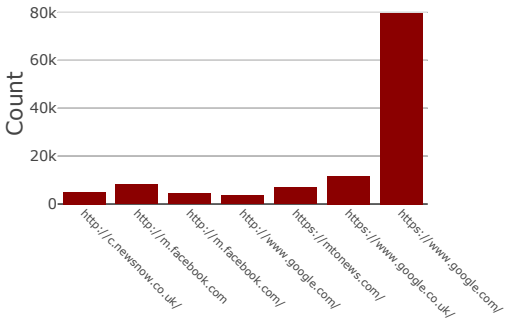
(b) Country



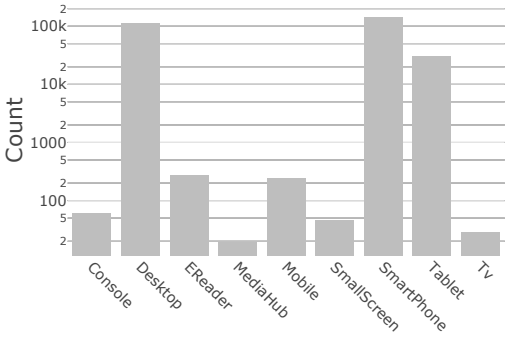
(c) PageVisitEndtimeType, e_{ijk}



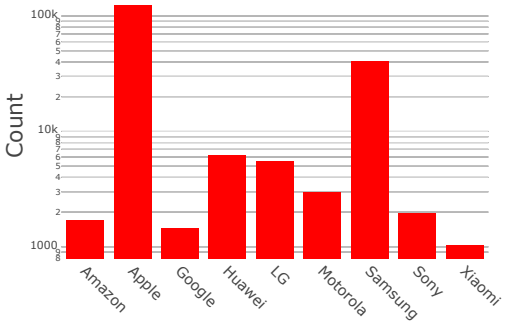
(d) PageVisitCategoryID



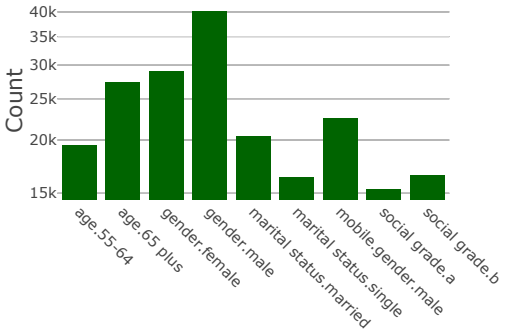
(e) Referrer



(f) DeviceType



(g) DeviceHardware



(h) ProfileDemographics

FIGURE 2.8: Graphical displays of key variable distributions in the *Clickstream* data set.

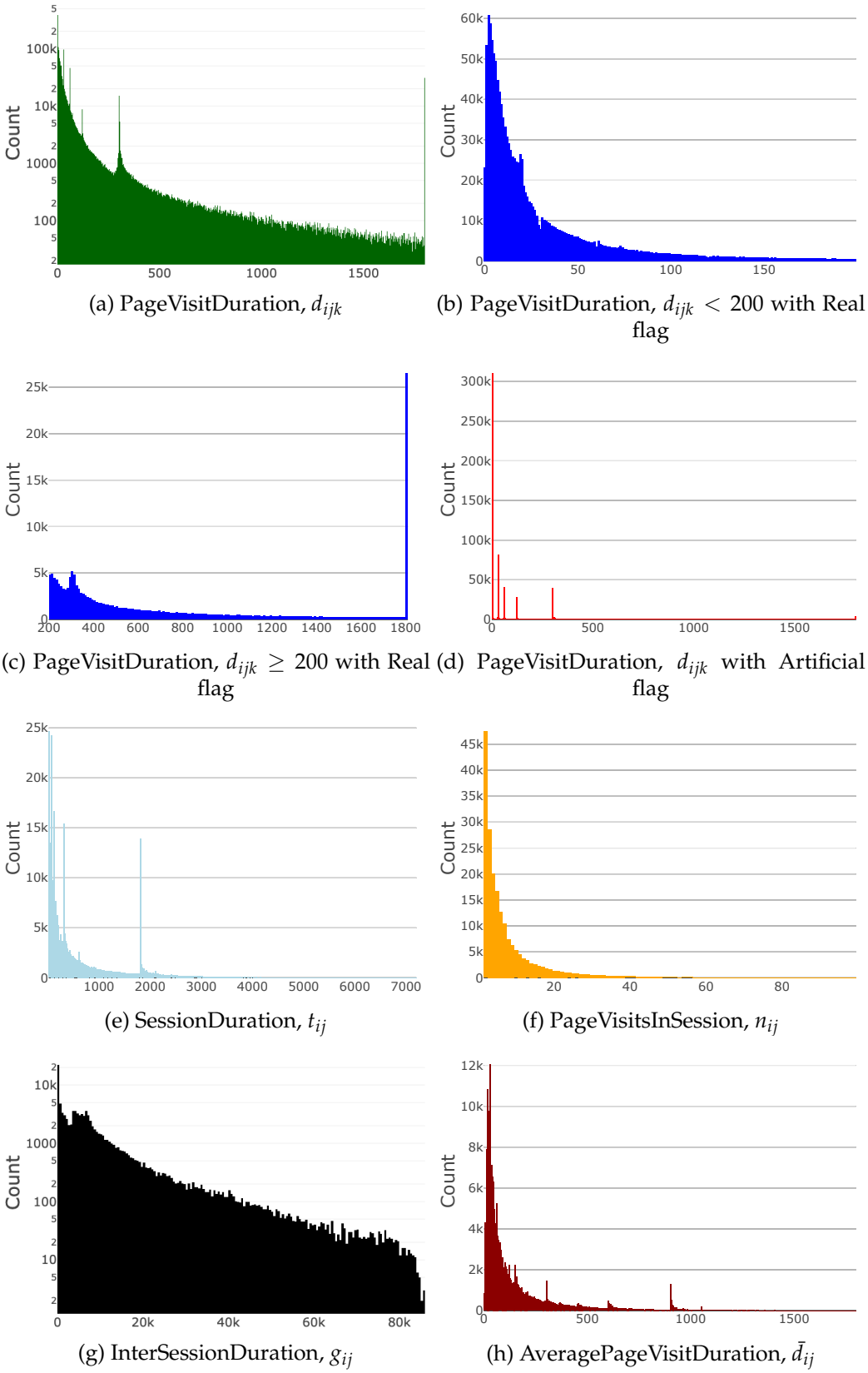
To further explore the *Clickstream* data set, we can create numerical variables from differences between time-based covariates at the page visit level. The definitions of each of the new variables can be seen in Table 2.5. They are a set of numerical variables that summarise the behaviour of browsing journeys.

Missing and extreme values have been removed from the plots in Figure 2.9, and these numerical variables add to the richness of the data and usability for modelling. We expect to see decaying distribution curves for each count variable. We also expect that the artificial value for page end time type may dictate a structure to the length of page visits, as this may inflate specific time durations. Furthermore, there is an upper limit of page visit length at 30 minutes (1800 seconds), so we expect this have a large effect on long browsing sessions.

Prior to our research, we expected the length of time spent on web pages to be a useful covariate to describe the behaviour of a browsing session. We calculate *PageVisitDuration*, d_{ijk}, by finding the difference between the start and end time of the page visit in seconds, i.e. $e_{ijk} - s_{ijk}$. In Figure 2.9a, we can see clear spikes at regular intervals in the distribution. Before we can begin any analysis, we require clean data, that is free from outliers and other anomalies. This improves the reliability and value of our data and creates robustness of our statistical models. The heartbeat mechanism checks for activity at intervals of 30, 60, 120 and 300 seconds from the start of the page visit. If the heartbeat returns as active, then we can say that the page visit length is at least this value. Noting the logarithmic scale, we can see the rough outline of an underlying decaying distribution, but with spikes at the heartbeat checkpoints. This is emphasised by the numerical summary of the distribution in Table 2.6, where the upper quartile value is exactly 60 seconds. Splitting the original distribution based on the page end time type label, the spikes in the original distribution in Figure 2.9a, are entirely removed to leave a smooth decay curve in Figure 2.9b. We have set the maximum page visit length to be 1800 seconds, and we note the spike here as a result of this upper bound, seen in Figure 2.9c. Figure 2.9d displays small regions around each of the heartbeat intervals. The shapes that surround each of the heartbeat values can be explained due to slight time delays in the collection of the data.

Variable name	**Type**	**Definition**	**Notation**
AveragePageVisitDuration	Numerical	The average value of *PageVisitDuration* within a *SessionID*	$\bar{d}_{ij}$
InterVisitDuration	Numerical	The length of time between consecutive *SessionID*'s for the same *ProfileID*	$g_{ij} = \lvert e_{i,j-1,N} - s_{ij1} \rvert$ where $N = n_{i,j-1}$
PageVisitDuration	Numerical	The length of time between the *PageVisitStartTime* and *PageVisitEndTime*	$d_{ijk} = e_{ijk} - s_{ijk}$
PageVisitsInSession	Integer	The number of page visits with a *SessionID*	n_{ij}
SessionDuration	Numerical	The length of time between the first *PageVisitStartTime* and final *PageVisitEndTime* within a *SessionID*	$t_{ij} = e_{ijn} - s_{ij1}$

TABLE 2.5: A selection of created numerical variables and definitions from *Clickstream* level data.



(a) PageVisitDuration, d_{ijk}

(b) PageVisitDuration, $d_{ijk} < 200$ with Real flag

(c) PageVisitDuration, $d_{ijk} \geq 200$ with Real flag

(d) PageVisitDuration, d_{ijk} with Artificial flag

(e) SessionDuration, t_{ij}

(f) PageVisitsInSession, n_{ij}

(g) InterSessionDuration, g_{ij}

(h) AveragePageVisitDuration, $\bar{d}_{ij}$

FIGURE 2.9: Graphical displays of numerical variables created from *Clickstream* data.

Variable	Min	LQ	Median	UQ	Max
PageVisitDuration	0	4	18	60	1800
SessionDuration	30	94	301	934	7197
AveragePageVisitDuration	0.35	30.00	65.54	172.00	1799.50
InterSessionDuration	0	922	5334	11249	85861
PageVisitsInSession	2	3	5	9	99

TABLE 2.6: A table to display numerical summaries of the key variables in the *Clickstream* data set.

We calculate the total of all values for d_{ijk} within the same session, j, and calculate *SessionDuration* t_{ij} by $e_{ijn} - s_{ij1}$ where n is the number of page visits k in the session j. In Figure 2.9e, we recognise the the spikes at each of the same intervals for the distribution of t_{ijk}. We see a longer tail to the right of the distribution, as we are finding the summation of values in 2.9a. The maximum page visit length is 1800 seconds, and the tall peak at this value suggests there a high number of sessions with only a single page visit of the maximum length.

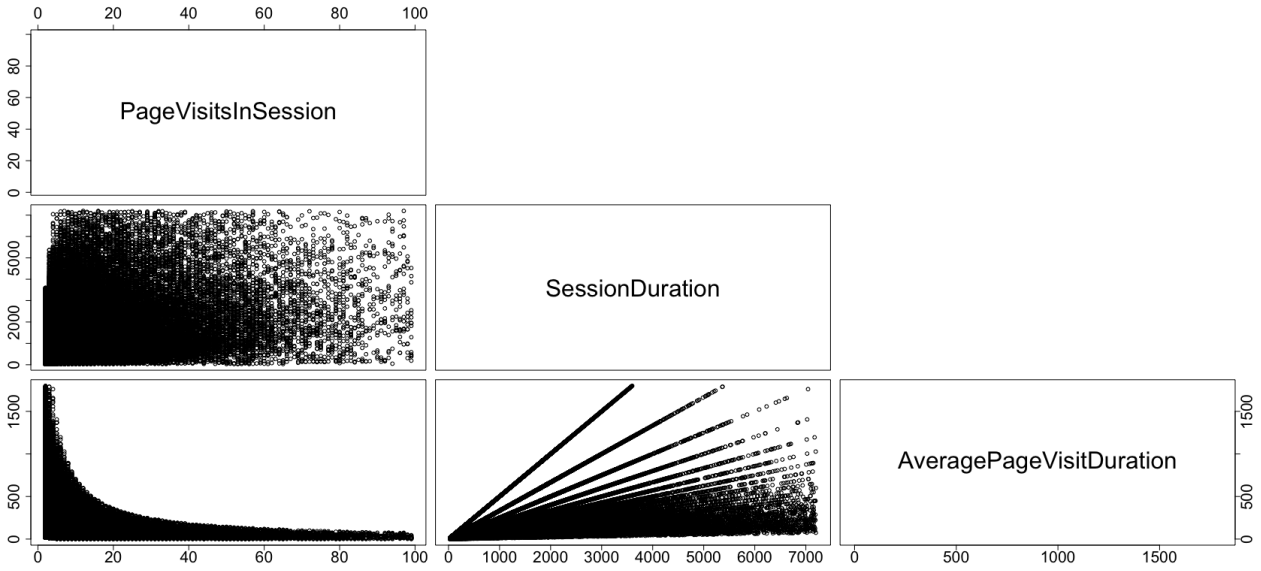
Another variable we can create is the number of page visits in a session denoted by n_{ijk}. We count the number of distinct page visits k within the session j, $\sum_{k \in j} 1$. We see a smooth decay curve in Figure 2.9f, with a limit for the maximum value of $n_{ijk} = 100$, as emphasised in Table 2.6. After this point, we will remove observations with a value for $t_{ij} < 30$ seconds and $n_{ij} = 1$. This is known in the industry as the **bounce filter**. This benchmark attempts to remove browsing visits that appear to be accidental or not worthy of counting as a genuine visit.

We calculate the time difference between browsing sessions as *InterSessionDuration*, g_{ij}, i.e the difference between the end of the $(j-1)^{st}$ session and start of the j^{th}. Small values for g_{ij} are created by multiple sessions in quick succession. Whereas, a device may be used before and after a working day, such that we see the browsing sessions further apart and hence larger values for g_{ij}. In Figure 2.9g and Table 2.6, we see a heavy tailed distribution on a much larger scale than Figure 2.9e, with the peak under 60 seconds, and a maximum value of just under 24 hours. We will explore this variable over multiple days in later chapters. We found observations where $g_{ij} < 0$, where two browsing sessions occur simultaneously, either by a data recording error or multiple browser tabs, so the time elapsed between the sessions should be zero.

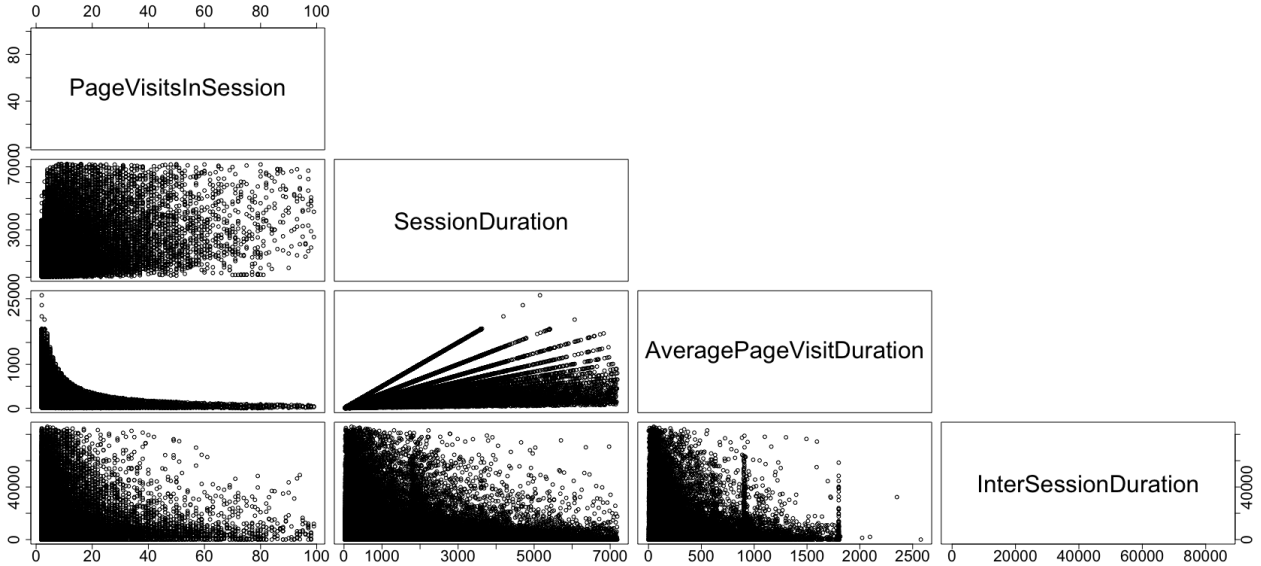
We find the average of all values of d_{ijk} within a session j, which we call *AveragePageVisitDuration*, $\bar{d}_{ij}$. The distribution can be seen in Figure 2.9h, with more regular peaks but with a much smoother overall shape, due to dividing the session duration variable by the number of page visits.

2.3.3 Relationships between variables

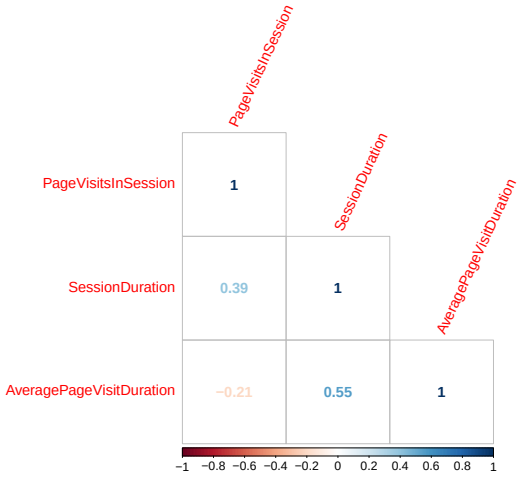
We aim to investigate the relationship between the numerical summaries of browsing sessions that we have created from the *Clickstream* data set. Figure 2.10 presents a pairs scatter plot and correlation plot of the variables n_{ij}, t_{ij} and $\bar{d}_{ij}$. There is no clearly visible pattern between n_{ij} and t_{ij}, indicating a need to consider both variables when building models. We note that $\bar{d}_{ij} = \frac{t_{ij}}{n_{ij}}$. The strict linear patterns for the plot in Figure 2.10a, are created by high volumes of small integer values for n_{ij}, which is reflected in the highest correlation of 0.55 in Figure 2.10c.



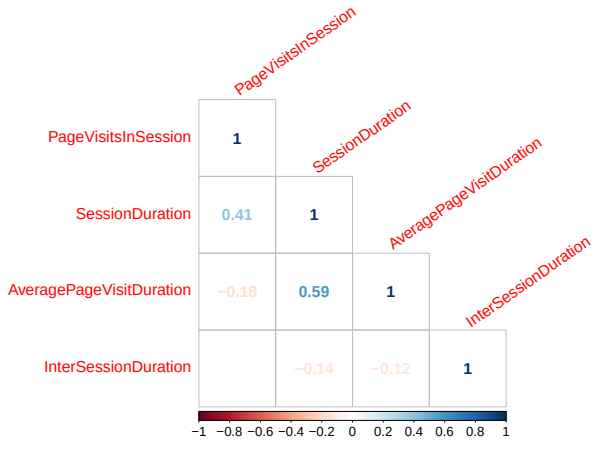
(a) Pairs plot of numerical summaries



(b) A pairs plot of numerical summaries for returning visitors



(c) Correlation plot of numerical summaries

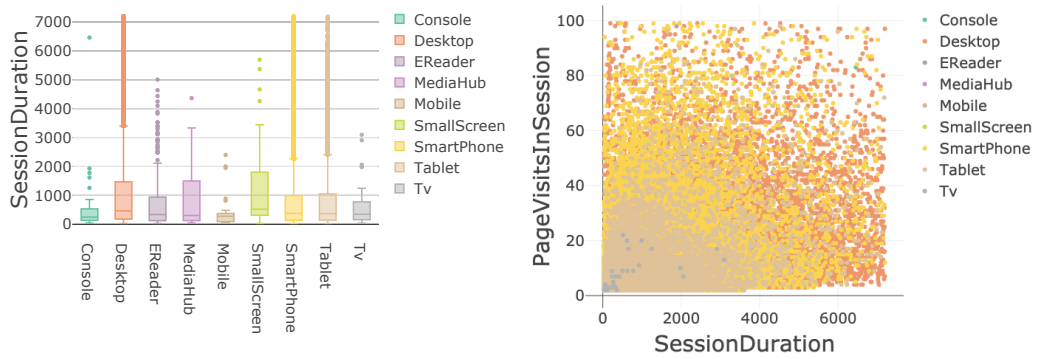


(d) A correlation plot of numerical summaries for returning visitors

FIGURE 2.10: Numerical summaries to describe browsing sessions.

By assessing browsing journeys from **returning visitors**, we can calculate g_{ij}, and investigate variable relationships for these returning visitors. In Figures 2.10b and 2.10d, the results are similar for the three variables n_{ij}, t_{ij} and $\bar{d}_{ij}$, suggesting that the relationships for these variables for returning visitors still hold. The addition of the inter session times presents no strong correlations to other 'within' session variables, which suggests we can consider this variable to indicate a new aspect of the browsing behaviour of the visitor. We would expect an engaged profile to display a short time between browsing sessions, i.e. a low value of g_{ij}, whereas a longer time suggests a lack of interest in the web site.

Exploring the relationship between types of devices and browsing session level numerical variables could lead to further insights. Figure 2.11a presents a boxplot of device type against t_{ij}, which suggests there might be a difference between devices for this numerical summary. Figure 2.11a shows all device types have skewed distributions with long right tails, emphasised for the most popular devices: desktop, smart phone and tablet. The median and upper quartile values suggest that desktop devices have longer values for t_{ij} out of the most popular device types. This is emphasised by the scatter plot in Figure 2.11b where observations from desktop lie in the upper regions of both t_{ij} and n_{ij}, with smart phone observations similarly scattered but with overall lower values, whereas the tablet observation are located much closer to the origin. This shows a clear difference in behaviour for the largest device types for browsing sessions, which corresponds to an intuitive view of using each of these devices, shorter visits of hand-held devices and longer visits on larger home-based devices.



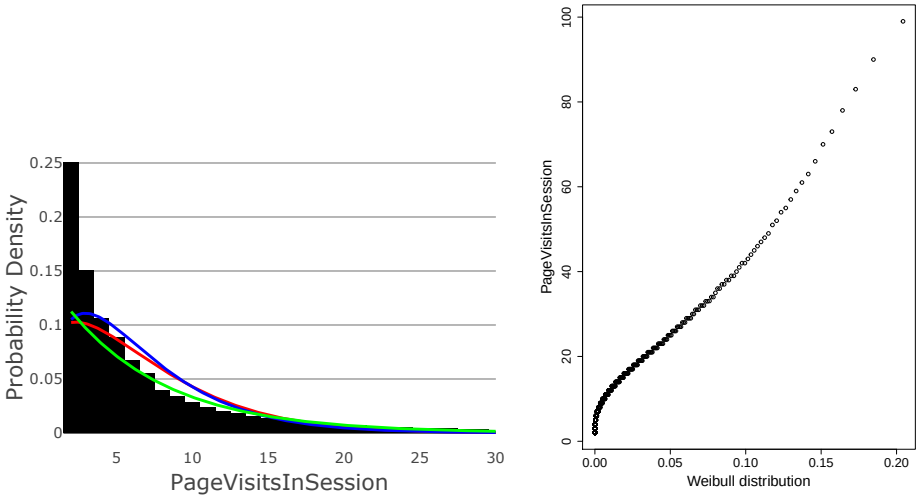
(a) Boxplot of *DeviceType* vs *SessionDuration* (b) Scatter plot of *SessionDuration* vs *PageVisitDuration*. Colour by *DeviceType*

FIGURE 2.11: Relationship between *DeviceType* and numerical summaries of browsing sessions.

The number of page visits in a session, n_{ij}, is a covariate that provided a smooth distribution that matched our expectations, with a shape that can be described in the form of a Weibull distribution. The choice of distribution was influenced by work in [21]. We remove browsing sessions with only a single page visit and display the distribution in Figure 2.12a. We can find the probability density distribution and estimate the parameters that would provide a Weibull curve to fit the variable distribution. A Weibull distribution is defined with two parameters, the scale parameter λ and the shape parameter α. The probability density function of a Weibull random variable x is:

$$f(x|\alpha,\lambda) = \begin{cases} \frac{\alpha}{\lambda}\left(\frac{x}{\lambda}\right)^{\alpha-1}\exp^{-\left(\frac{x}{\lambda}\right)^{\alpha}} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

We model the distribution for n_{ij} finding the maximum likelihood estimate for the parameters as Weibull$(1.0, 8.6)$, which is visualised in Figure 2.12a. We can use this to find the probability of each value for each observation n_{ij}, which can be used to indicate long or short browsing sessions based on high or low probabilities. We can see the heavy tail reflected in the quantile-quantile plot in Figure 2.12b, for which we used the R package *fitdistplus* [28]. The Weibull distribution curve does not fit the data well and the quantile plot is curved indicating a departure from this distribution. We experimented with other distributions, such as Exponential and Gamma, but failed to find a better fit – likely due to the strong skewness in the count data. Note that we have removed single page visits sessions from this analysis.



(a) Weibull: Red, Gamma: Blue, Exponential: Green.

(b) Quantile-quantile plot for *PageVisitsInSession* and Weibull(1.0, 8.6).

FIGURE 2.12: A comparison of the variable *PageVisitsinSession* and a fitted Weibull distribution.

2.3.4 Discussion

This introduction to the *Clickstream* data set and browsing journeys has provided a platform to model this special type of sequential data in a variety of ways. We have learnt that we are dealing with a significant amount of data and we must adopt methods for the Big Data environment. Ultimately, our goal will be to build a set of numerical and categorical variables which will be important components in a model to predict a general event based on historical browsing behaviour. Examples of such events include clicking on an advert, purchasing a product, or meeting some other success criteria.

In this chapter, we have introduced potential covariates that describe the browsing behaviour data. There are a number of variables that relate to page visit level and at the profile level (i.e. device information). The raw variables in the original *Clickstream* data set are mainly categorical or time based, hence we create numerical

summaries to develop extra covariates for exploration and modelling. These numerical variables provided intriguing results and did not fully meet our expectations.

We found a heartbeat mechanism that has a significant impact on the distribution of the page visit lengths, which could have an effect when using the time duration variables in a statistical model. Scatter plots of the session level variables help us understand that the correlations may not be strong enough to have a significant impact of multicollinearity on our models. Finally, we described the distribution of the number of page visits in a session using a Weibull distribution.

Chapter 3

Statistical dimension reduction of browsing behaviour

3.1 Chapter overview

In this chapter, we will focus on dimension reduction to provide a single score to assess browsing behaviour. In section 3.2, we explore appropriate features, and outline our variable selection process to determine how to create an effective score. We will use statistical techniques to highlight where we can improve on existing methods that assess behaviour. In section 3.5.2, we aim to produce a statistically informed way to reduce the number of dimensions, and explain the benefits of our method over the current solution. We compare our novel method to principal component analysis (PCA) and other dimension reduction techniques. In section 3.6.2, we explore an application of the algorithm to label a variety of types of behaviour, and compare to an existing rule-based classifier. We aim to determine more accurate behavioural traits and increase the number of labels.

The one-dimensional summary used in the industry for these purposes is called **intent score**, which is designed to provide a level of intent to purchase based on a set of behaviours. One example of the application of intent score is in targeted advertising, where a numerical representation of the level of activity can be used to rank profiles' behaviour and decide which profile to display an advert to. With a naive approach, the profile with the highest intent is the most likely to engage with the advert and click, driving the success of an advertising campaign. A further application of intent score, and other statistical summaries of historical data, is to drive a labelling system that describes and interprets a profile's browsing behaviour.

3.2 The aim

In this section, we will present the process to combine the key attributes of browsing behaviour, and create a one-dimensional summary while maintaining the variance within the data.

We are motivated by an existing algorithm, developed by Carbon, that creates summary variables of browsing behaviour attributes and a rule-based classifier for activity labels. Upon closer inspection, we believe that the algorithm could be enhanced with further statistical rigour and additional variables to describe different attributes of user behaviour. The algorithm contains variables with **too few levels** with **arbitrary thresholds** to discretise the continuous variables. The structure of the existing algorithm is outlined in Figure 3.2a and the aim is to determine a score for each of the three attributes. The rules and interpretations of the scores are provided in Figure 3.1.

Recency

- A score of 5 if: $r \geq (t - 7)$.
- A score of 4 if: $r \geq (t - 30)$.
- A score of 3 if: $r \geq (t - 90)$.
- A score of 2 if: $r \geq (t - 180)$.
- A score of 1 if: $r \geq (t - 365)$.

where t is the current date, r is the last date the interest has been visited.

Variability

- A score of 5 is given if the interest is visited daily.
- A score of 4 is given if the interest has been visited in all 7 day partitions.
- A score of 3 is given if the interest has been visited in all 30 day partitions.
- A score of 2 is given if the interest has been visited in the last year.
- A score of 1 is given to all.

Frequency

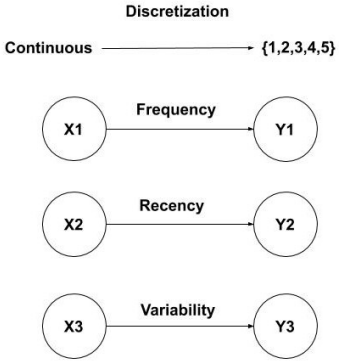
$$\left\lceil \frac{5f}{\max(f)} \right\rceil$$

where f is the total number of days the interest is visited in the last 60 days.

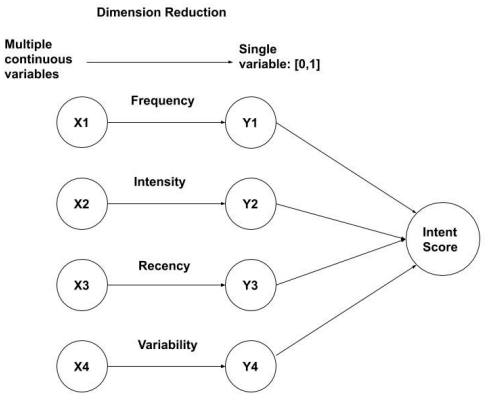
FIGURE 3.1: The rules for the existing algorithm

A limitation of the existing approach is that many aspects are arbitrary and it is not driven by the data as the discretisation rules are fixed, hence it cannot adapt to changes in the data. Furthermore, it is difficult to interpret what the resulting levels of the attribute represent and the proportion of data classified into each level.

Our goal is to develop a similar procedure that is instead informed by the statistical behaviour of the data rather than prescribed arbitrary values, and we can add subjective and expert insights. Techniques such as PCA, and summaries of the empirical distribution (quantiles) will be particularly helpful in the analysis. We will use Figure 3.2b to help explain the concept of the variable exploration and the overarching framework of the dimension reduction algorithm. We focus on reducing multiple continuous statistical variables, grouped into attributes, for example *Frequency*, to a single representation of each attribute ranging between zero and one. We can begin with simple models with 1 or 2 contributing variables per attribute, stretching to a complex process of a linear combination of statistical variables. This structure allows the underlying variables to change based on new information or a change in approach, without redesigning the entire algorithm. We will then create the intent score value by combining all our chosen behavioural attributes. Further, we aim to include the attribute of *Intensity* to add a new aspect of browsing behaviour.



(a) The structure of the existing algorithm.



(b) The structure of the new algorithm we propose.

FIGURE 3.2: Visual representations of the algorithms we discuss in Section 3.2.

3.3 Building statistical variables

The current algorithm aggregates the *Clickstream* data to a **daily** level, i.e. we aggregate the page visit data into 24-hour periods and create numerical summaries of the behaviour for each day. Each row in this transformed data set contains the profile, page visit category, date (d) and the number of pages visited - *Hits* (h). An example of the data format is presented in Table 3.1a. A profile will exist over multiple rows if it has a rich history of behaviour, with multiple values for the page visit category over a multitude of dates. To restrict the date range, we only consider the most recent 60 days of browsing behaviour. From this dataset, we can create appropriate numerical features to describe the activity of a profile on a number of different categories. The data is of the order of approximately 10^8 observations, hence we use Spark to enable us to transform and query the data set.

This section sets out a large number of numerical variables from the aggregated data set in Table 3.1a. Before exploring the data set, we must form an idea of the types of variables that may be useful in describing the activity level in a varied and insightful way. There are a number of key areas that statistical variables can define, which relate to specific attributes of behaviour. For each attribute, we will explore how we can transform the data into statistical summaries, and use them to inform us about levels of engagement for each unique pair of profile and page visit category.

ProfileID	PageVisitCategoryID	Date (d)	Hits (h)
abc	100	2018-01-20	6
abc	100	2018-01-25	4
abc	100	2018-01-26	12
abc	100	2018-01-29	6
abc	100	2018-02-08	2
abc	100	2018-02-10	20

(A) Page visit data aggregated to the daily level.

Attribute	Variable	Notation	Value
Frequency	DayCount	$f = \sum d$	6
	WeekdayCount	$f_d = \sum_{d=Weekday} d$	4
	WeekendCount	$f_e = \sum_{d=Weekend} d$	2
Intensity	AverageHits	$\bar{h}$	8.33
	MaximumHits	$\max(h)$	20
	MedianHits	$\tilde{h}$	6
	MinimumHits	$\min(h)$	2
	RangeHits	$\max(h) - \min(h)$	18
Recency	AverageDate	$\bar{d}$	2018-01-30
	MaxHitCountDate	$d : h = \max(h)$	2018-02-10
	MedianDate	$\tilde{d}$	2018-01-27
	OldestDate	$\min(d)$	2018-01-20
	RecentDate	$\max(d)$	2018-02-10
Variability	AverageInterVisit	$\bar{g}$	4.16
	DateRange	$\max(d) - \min(d)$	25
	MaximumInterVisit	$\max(g)$	10
	MedianInterVisit	$\tilde{g}$	3.5
	MinimumInterVisit	$\min(g)$	1
	RangeInterVisit	$\max(g) - \min(g)$	9

(B) Transformed data in Table 3.1a to statistical variables.

TABLE 3.1: An example to display how to create the statistical variables.

Frequency - summaries of f. How often did the profile return to the category? We can use information across days to build the frequency attribute. We count the total number of days that activity has occurred in the last 60 days to create f where:

$$f = \sum_{i=1}^{60} V_i \text{ where } V_i = \begin{cases} 1 \text{ if visited on day } i \\ 0 \text{ otherwise} \end{cases}$$

Further, we can find the total number of days of activity on weekdays and weekends. We can establish if the date lies on a weekday (Monday to Friday) or weekend (Saturday or Sunday), to calculate the variables, f_d and f_e, respectively. This could be used to create a ratio, for example $\frac{f_d}{f_e}$, which may be useful for more complex models. The current algorithm uses f to calculate the frequency attribute, which clearly tells us useful information about the level of intent that a profile has for a particular page visit category. For our toy example in Table 3.1a, we can see the frequency variables in Table 3.1b, where we have 6 days of visits and we have double the number of

weekday visits f_d compared to weekend visits f_e.

Intensity - summaries of h. What was the depth of the browsing behaviour? By depth, we mean a measure of the frequency of within-day browsing behaviour. Using the *Hits* count variable would be an addition to the current algorithm, providing information about the depth of the visits. Numerical summaries of h (for each profile and page visit category) can be used to build data for the intensity attribute. Examples of the summaries are the mean, median, maximum, minimum and range, with the calculations displayed in Table 3.1b. This level of detail should provide an extra layer that does not exist in the existing algorithm. In our example in Table 3.1a and 3.1b, we have a variation in the values for h, where $\max(h) = 20$ and $\min(h) = 2$. Also the mean is larger than the median, $\bar{h} = 8.33 > 6 = \text{median}(h) = \tilde{h}$, indicating a right skew due to larger extreme values.

Recency - summaries of d. Was the browsing behaviour observed recently? Another measure of engagement can be derived from the recency of the browsing activity, the most recent date, $\max(d)$, will be useful and is used in the existing algorithm. However, we can describe recency, through a range of date-based variables to provide more granularity and describe the location of the activity in the past 60 days. We can find the mean, median, maximum and minimum of the values for d. We will explore to see if any of these variables gain us a deeper insight into the recency attribute. We can also find the value for d corresponding to $\max(h)$, i.e. the peak of interest from the user, which could indicate a more substantial meaning of recency. From Table 3.1b, we can see that the mean, $\bar{d}$, is larger than the median, $\tilde{d}$, emphasising the skew in the behaviour towards the most recent date.

Variability - summaries of g. Is there a regularity to the browsing behaviour? The date provides us with key information about the regularity or irregularity of page visits, and statistical variables are calculated using the intervals between visits. Thus, we obtain variables measured in days, relating to the spread of activity. We measure the number of days between each date of activity for a profile and page visit category, generating a list of the 'inter-visits' g, i.e. $g_i = d_{i+1} - d_i$ for $i \in [1, f)$. We find the average, median, maximum, minimum and range of the values in this new list g_i. Further, we find the overall date range, using the formula $\max(g) - \min(g)$. The example in Table 3.1b shows a lot of variation in the values obtained from the list g_i, specifically the $\text{mean}(g) = \bar{g} > \tilde{g} = \text{median}(g)$, indicating the distribution of g is skewed by larger values, namely $\max(g) = 10$.

3.4 Exploratory data analysis

We explore a real data sample of approximately 30,000 rows, and 19 statistical variables across 4 attributes, outlined in Table 3.1b, to investigate the distributions and relationships between key variables. Note that, the sample contains only repeat visitors because we want multiple visits to create values for the inter-visits. We explore the individual distributions with histograms in Figure 3.3, and using the quantiles in Table 3.2. The correlation plot in Figure 3.4 and pairs plots in Figure 3.5 will help us to visually analyse the relationships between the variables, and look for any obvious patterns and correlations, aiding the process of variable selection. We expect these plots will reinforce that each variable only represents it's respective attribute.

We would see this by only having strong correlations between the variables within each attribute and weaker correlations for variables relating to different attributes.

The count variables for the frequency attribute f, f_d and f_e display long right tails and steep decay curve distribution shapes. In Figure 3.3a and Table 3.2, f is dominated by the value 2, hence we need to consider how to treat this when we discretize. Figure 3.4 shows strong positive correlations between the frequency statistical variables, and only weak correlations with some variability variables, which are driven by the majority observations where $f = 2$ and there is a single value in the list g. This supports our decision to group these variables in the frequency attribute, and represent that aspect of behaviour using a combination of these variables.

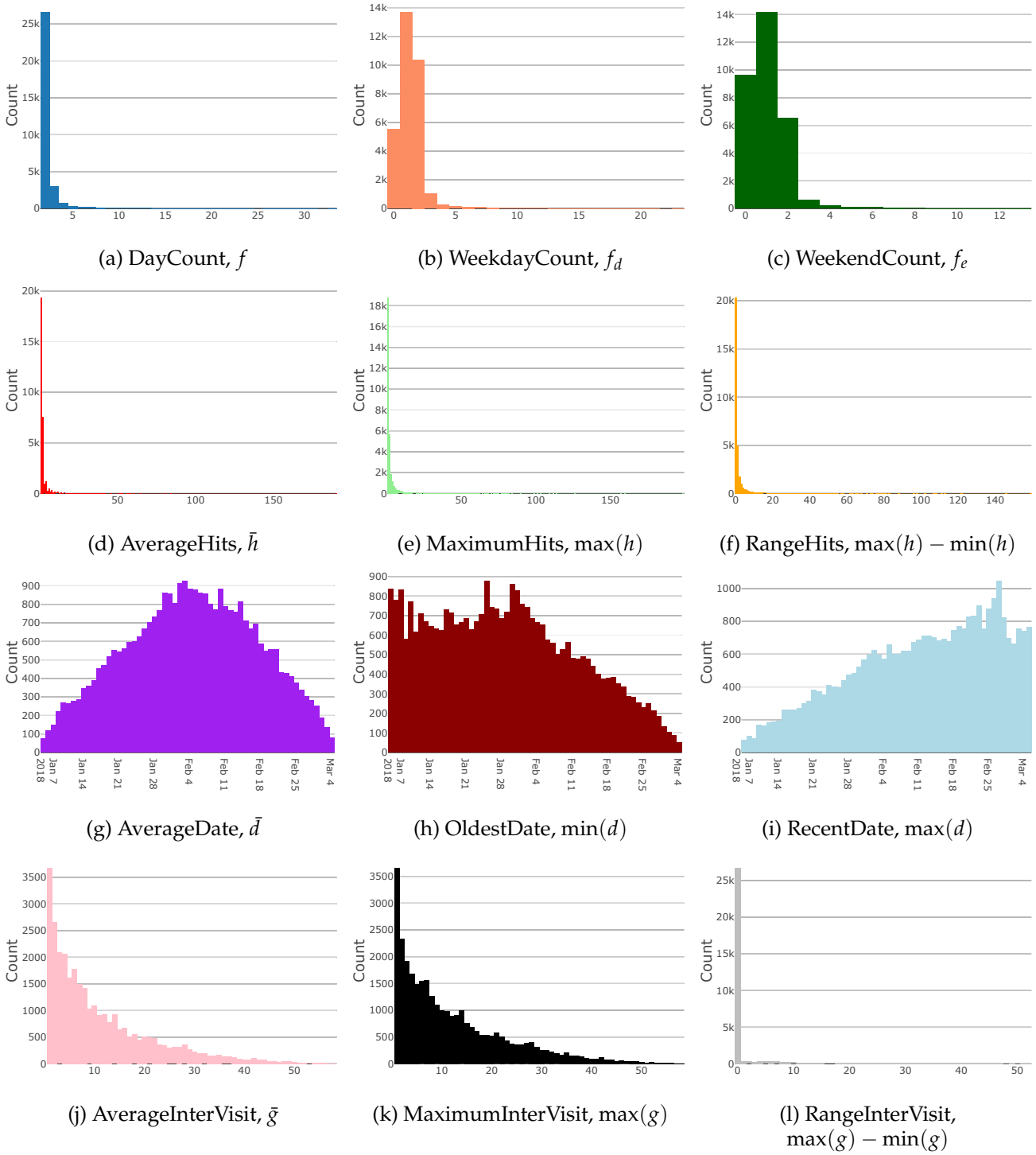
The individual distribution plots chosen for the intensity attribute are the average, maximum and range of values for h. The distributions show right-skew towards low values of hits with extreme values in long right tails in Figures 3.3d-3.3f, which are likely to correspond directly to bot activity. Figure 3.5b displays a sparse intensity pairs plot due to the existence of extreme values, with positive linear patterns visible.

Variable	Symbols	Min	LQ	Median	UQ	Max
DayCount	f	2	2	2	2	33
WeekdayCount	f_d	0	1	1	2	23
WeekendCount	f_e	0	0	1	1	13
AverageHits	$\bar{h}$	1	1	1	2	190
MaximumHits	$\max(h)$	1	1	1	2	199
MedianHits	$\tilde{h}$	1	1	1	2	190
MinimumHits	$\min(h)$	1	1	1	1	181
RangeHits	$\max(h) - \min(h)$	0	0	0	1	159
AverageDate	$\bar{d}$	06/01 12:00	26/01 00:00	04/02 16:00	14/02 12:00	04/03 12:00
MaxHitCountDate	$d : h = \max(h)$	06/01	23/01	04/02	17/02	05/03
MedianDate	$\tilde{d}$	06/01 12:00	26/01 00:00	05/02 00:00	14/02 12:00	04/03 12:00
OldestDate	$\min(d)$	06/01	17/01	28/01	08/02	04/03
RecentDate	$\max(d)$	07/01	02/01	13/02	24/02	05/03
AverageInterVisit	$\bar{g}$	1	3	8	16	58
DateRange	$\max(d) - \min(d)$	1	4	10	21	58
MaximumInterVisit	$\max(g)$	1	3	9	18	58
MedianInterVisit	$\tilde{g}$	1	3	8	16	58
MinimumInterVisit	$\min(g)$	1	2	7	15	58
RangeInterVisit	$\max(g) - \min(g)$	0	0	0	0	52

TABLE 3.2: A table to display numerical summaries of the statistical variables.

There are strong correlations between the intensity variables and no interactions with other attributes in Figure 3.4. Hence, we can represent the intensity attribute exclusively with a combination of these variables.

For the recency attribute variables, we display the distributions for $\bar{d}$, $\min(d)$ and $\max(d)$ in Figures 3.3g-3.3i. The variables are skewed in the directions we would naturally expect them to be, i.e. the $\min(d)$ to the left (dates further in the past) and $\max(d)$ to the right (more recent dates). Figure 3.5c, the recency pairs plot shows clear structural patterns, which can be explained as we are using summary statistics



(a) DayCount, f

(b) WeekdayCount, f_d

(c) WeekendCount, f_e

(d) AverageHits, $\bar{h}$

(e) MaximumHits, $\max(h)$

(f) RangeHits, $\max(h) - \min(h)$

(g) AverageDate, $\bar{d}$

(h) OldestDate, $\min(d)$

(i) RecentDate, $\max(d)$

(j) AverageInterVisit, $\bar{g}$

(k) MaximumInterVisit, $\max(g)$

(l) RangeInterVisit, $\max(g) - \min(g)$

FIGURE 3.3: Graphical displays of key statistical variable's distributions.

that bound each other, i.e. the maximum value will always be higher or equal to than the minimum value. This is reflected in strong correlations between the variables, in Figure 3.4. This suggests we may choose a selection of these variables to describe the recency of visits, and there are no obvious linear correlations with other attributes.

Using the values g_i, we display the distributions of $\bar{g}$ in Figure 3.3j, which is a smooth decay curve and similar to $\max(g)$ in Figure 3.3k. The quantiles values for $\max(g)$ are greater or equal to the $\bar{g}$ values in Figure 3.2. The range of the inter-visit

distribution in Figure 3.3l is dominated by zeros, where there is only a single value in g, due to the high count of rows where $f = 2$, and $\max(g) = \min(g)$. Figure 3.4 shows strong correlations between most of the variables in the variability attribute, except the range which is mainly zeros. The variability pairs plot in Figure 3.5d shows a linear diagonal that corresponds to the observations where there is a single inter visit. A combination of variables could lead to the best way to describe the variability of the browsing activity.

There are strong correlations between variables within each attribute and this strengthens our motivation to split the statistical variables by attribute. We have shown this through analysis of the correlation matrix structure, which has showed that we can keep the attributes separate and focus on describing each of them using the statistical variables as we set out in Figure 3.2b. There is a clear group structure within each attribute, and low correlation between the four attributes. We have discovered a new attribute, *intensity*, that provides new information about browsing behaviour attributes that wasn't included in the existing algorithm. We will use this to create a more interpretable and informative intent score, reduce the multicollinearity in later modelling, and enhance the quality of the activity labels.

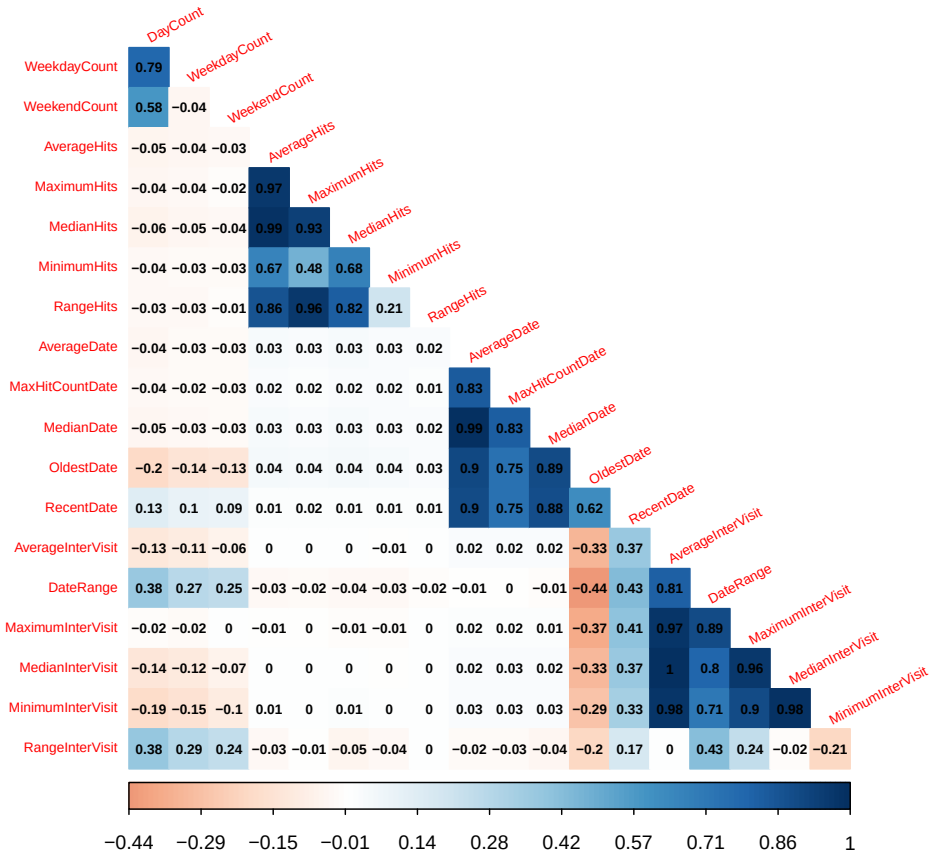
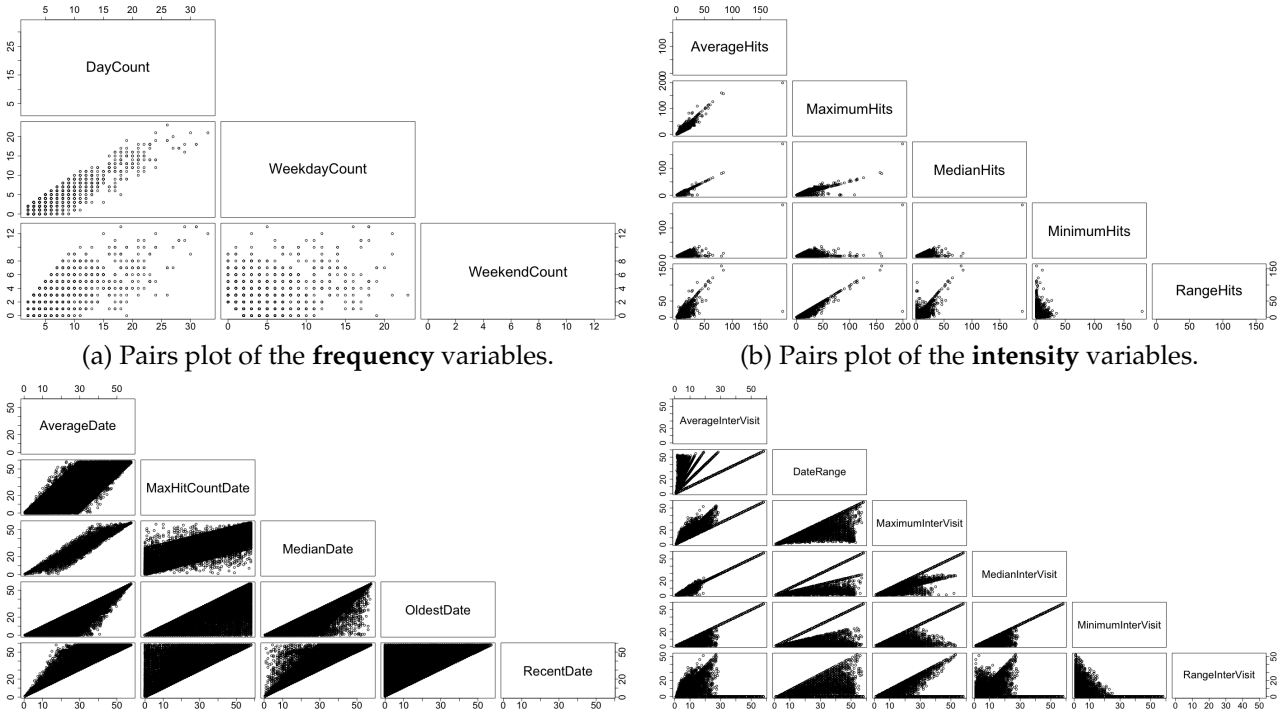


FIGURE 3.4: Correlation plot of all statistical variables.



(a) Pairs plot of the **frequency** variables.

(b) Pairs plot of the **intensity** variables.

(c) Pairs plot of the **recency** variables.

(d) Pairs plot of the **variability** variables.

FIGURE 3.5: A collection of pairs plots split by the attributes.

3.5 Dimension reduction

In this section, we will refer back to the aim set out in Section 3.2, and discuss how we will perform dimension reduction, building an algorithm to transform the statistical variables within each attribute to an intent score. We will provide the steps and reasoning behind decisions; we aim to select variables in a way that we can explain each attribute in a succinct and interpretable way, whilst keeping the variability within the data.

Based on the data exploration in Section 3.4, we decide to make a manual selection of variables, from each of the attributes, to provide an intent score with interpretability, as opposed to a dimension reduction procedure.

3.5.1 Informed variable selection by hand

We chose the variables highlighted in Figure 3.6a, and we will explain the reasoning behind each selection.

The frequency attribute has 3 candidate variables and we can select a subset to reduce the number of dimensions, as in Figure 3.2b. The variable f counts the number of days a profile hits a page visit category on a daily level over 60 days of historical data. In future work, we could look at using f_d or f_e in this attribute, but we conclude that it does not provide extra information to the interpretation of level of intent as we decide that the day of the week currently does not affect the overall frequency attribute. Further, both are strongly positively correlated with f, as seen in Figure 3.4, and by selecting multiple correlated variables we would introduce redundancy. This selection provides a strongly interpretable variable that we can directly link to a level of browsing interest.

After inspecting the pairs plot in Figure 3.5b and the separation from other variables in the correlation plot, Figure 3.4, we can choose the variables for the intensity attribute. We propose to only take a single variable, due to the strong correlations within the attribute, and we choose the average value $\bar{h}$. There are extreme values within the hit count, as displayed in Figures 3.3d and 3.3e, which are likely to be bot activity. Thus, taking an average rather than taking the maximum will be more robust to the extremes. However, the average will still be more sensitive to outliers compared to the median, but the median is too computationally intensive in practice using Spark. To calculate the median value we require the position of every value, which is unfeasible in a distributed data structure at this volume. One option is to use an approximation function, but as the average can be calculated in parallel we prefer this statistic. Further, using the average will combine and assess the entire history of the profile, rather than a single day of behaviour (i.e. the maximum or minimum), which we expect to build a better representation of the profile's intent.

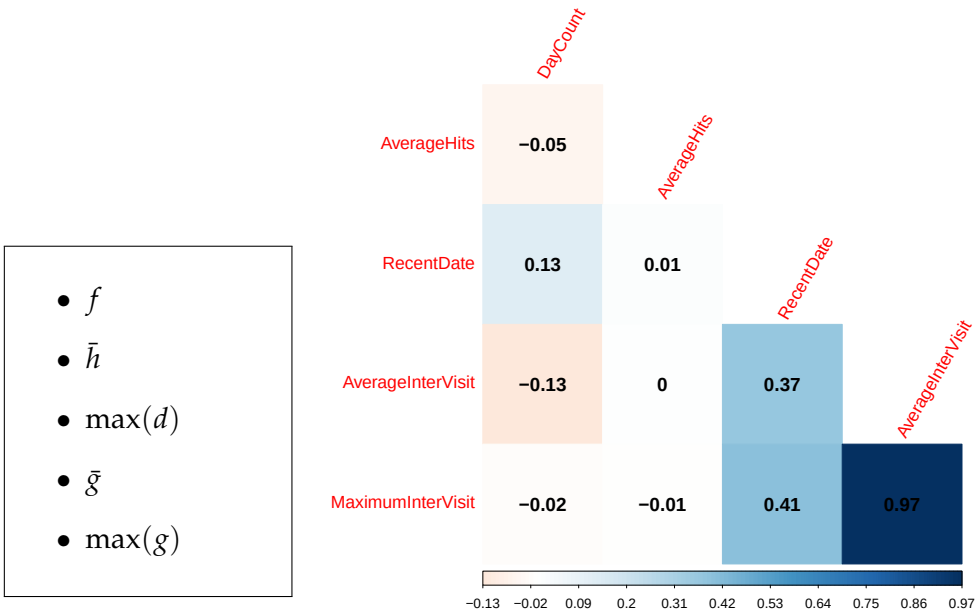
We impose a similar approach for the recency attribute of choosing a single variable, as the correlation matrix in Figure 3.4 displays that the recency variables have weak correlations with other attributes and strong correlations within the attribute. Table 3.2 informs us that the quantile values are well spread which will help to keep variability in the attribute. However, the key motivation is to assess the intent of the behaviour and for this we require an assessment the last seen visit, i.e. $\max(d)$. This best describes the recency of the browsing behaviour and is the most natural interpretation of the recency attribute.

For the variability attribute, we propose to use a combination of the average and maximum summary statistics - $\bar{g}$ and $\max(g)$. The average and maximum values help us identify a measure of regularity between the visits, which we see as a major aspect of browsing intent. The motivation for this lies in the interpretability we desire for this attribute, as we can see from Figure 3.4 that the correlation is high between $\bar{g}$ and $\max(g)$. This correlation is driven by the high number of single values for g, for which clearly the average and maximum are equal. Using the average and maximum together can help identify a regular visitor, i.e. the values for both variables are similar. Additionally for a regular visitor we would set a threshold for frequency; for example more than four visits. Whereas with irregular visiting, we expect the maximum to be much larger than the average. We want large gaps between returning visits to be assigned a low score for variability as the profile intent is low and irregular. We aim to see lots of variation in this attribute when we combine these values, and we have a more complex approach to variability than the existing algorithm. We will describe later how to reduce the dimensions of the variability attribute. In future work, we may wish to combine these into a single variable to represent the difference between the largest value and the average, i.e. $\max(g) - \bar{g}$, however keeping the interpretation of each variable simple is crucial.

Figure 3.6b shows the correlation matrix for the selected variables we have discussed. There are weak correlations across the attributes, emphasising our reasoning to select from each of them, and a strong correlation between the inter-visit variables.

3.5.2 Comparison to PCA and other methods

We created 19 statistical variables across 4 attributes, and we have provided a variable selection process to reduce each attribute to a single summary variable. We aimed to capture variation within the data, while keeping the interpretation of the score to describe intent. We can compare our approach with principal component



(a) Our variable selection by hand.

(b) The correlation matrix.

FIGURE 3.6: Our variables we have selected.

analysis (PCA), and other methods to determine if the variables we selected are appropriate and valid.

Our variable selection

Firstly, we will assess our choice of variables from our data, $X = (x_1, X_{-1})$. We present an expression to quantify the proportion of variability left over in our data, X_{-1}, after accounting for a linear dependency on our chosen variable x_1:

$$V(X_{-1}|x_1) = \frac{\text{tr}(Var[X_{-1}] - Cov[X_{-1}, x_1]Var[x_1]^{-1}Cov[x_1, X_{-1}])}{\text{tr}(Var[X_{-1}])} \tag{3.1}$$

As a result, we can calculate how much variation can be attributed to our chosen variable or set of variables.

In Table 3.3, we provide the proportion of variance explained using our variable selection method. Most of the figures in the table highlight that our manual selection is effective as the proportions are high. For each attribute of frequency, intensity and variability we have selected variables that explain over 50% of the variation within each attribute. The weakest selection appears to be for the frequency attribute, but we note that our variable selection has led to an interpretable score which assesses browsing intent and engagement which also separates the data well. Also, we have only accounted for a linear dependency in Eq 3.1; non-linear dependencies have not been investigated. An alternative for measuring associations is the maximal information coefficient, which we do not consider in this thesis [29].

	Our method		Highest loading PC1		Random Forest	
Attribute	Variable	%	Variable	%	Variable	%
Frequency	f	52	f	52	f	52
Intensity	$\bar{h}$	85	$\max(h)$	88	$\bar{h}$	85
Recency	$\max(d)$	61	$d : h = \max(h)$	62	$\bar{d}$	81
Variability	$\bar{g}, \max(g)$	94	$\max(d) - \min(d)$	63	$\bar{g}$	83

TABLE 3.3: A table to show the proportion of variance that is retained within each attribute in the variable selection process in Section 3.5 compared to other methods.

PCA

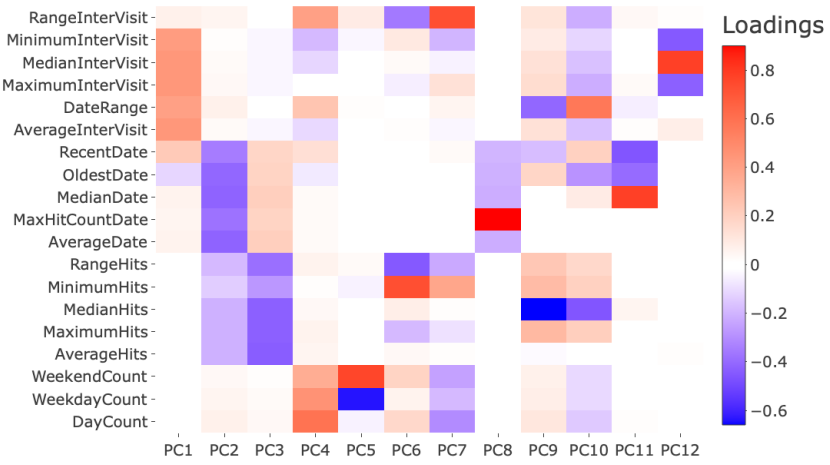
Principal component analysis (PCA) is a popular dimension reduction technique that seeks a linear combination of variables such that the maximum variance is extracted from the variables [30]. Using PCA we will compare our variable selection against the principal components from the transformation. The principal components are linear combinations of the original variables weighted by their contribution to explaining the variance in a particular orthogonal dimension. If we have p predictors: $X_1, ..., X_p$, then the first principal component can be written as:

$$Z_1 = \phi_1^1 X_1 + \phi_2^1 X_2 + ... + \phi_p^1 X_p$$

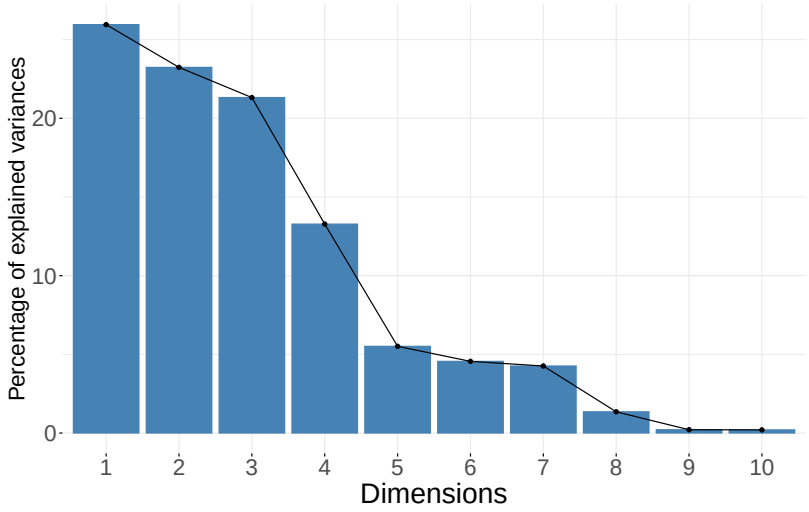
where Z_1 is a linear combination of original predictor variables which captures the maximum variance in the data set and ϕ^1 is the vector comprised of the weighting for each predictor. The coefficients of ϕ^1 are the loadings, and correspond to the coefficients of the first eigenvector of $Var[X]$. We remove this variance and seek a second linear combination which explains the maximum proportion of the remaining variance, and so on. The result is orthogonal (uncorrelated) factors.

First, we apply PCA to all statistical variables and display the component loadings in Figure 3.7a and a scree plot in Figure 3.7b. The heatmap displays the first 12 principal components and shows the structure for each component loading, where the darker regions highlight higher contributing values to the component. The first component represents variability, the second is recency, the third is intensity and the fourth is mainly frequency. The attributes are approximately orthogonal components of variation. This outcome reflects our choice of attributes and emphasises the weak correlations between attributes. The scree plot shows that each of the first components explain over 20% of the variance, and the fourth contributes over 10%, which together describe over 80% of the variation.

We want to compare PCA to our manual variable section and to do this we perform PCA separately on each attribute. We can inspect the first two principal components for each individual attributes in Table 3.4 and use them to determine the combination of variables to compare to our selection. We also show the cumulative proportion of variance that is explained by each principal component, which we will compare to other dimension reduction methods. Firstly, for the frequency attribute the first principal component chooses f as the largest contributor and accounts for 76% of the variation. This is higher than the variance explained (52%) for the single variable choice of f from our variable selection. Next, the first principal components for both intensity and recency attributes approximately find an 'average' of all variables and both explain over 85% of the variation in the data. Lastly, the first component (87% of variance explained) for the variability attribute is an 'average' of



(a) A heatmap to visualise the component loadings from PCA.



(b) A scree plot from PCA.

FIGURE 3.7: The results from PCA on all statistical variables.

all variables except for $\max(g) - \min(g)$, which is contained in the second component. This seems sensible as we know that the range of the inter visits is not a useful summary of the variability behaviour from our exploration analysis.

We will use an extension of PCA to select a single variable, by choosing the maximum contribution to the first principal component loading [31]. We will apply this to the within attribute PCA results in Figure 3.4, and display the results in Figure 3.3. The frequency attribute is the only component that agrees and chooses f, whereas the other attributes are not in agreement with our variable selection. The variance proportions that are explained by these variables are not dissimilar to our selection method. Specifically, the intensity attribute highest loading from PC1 is $\max(h)$, which from Table 3.3 explains 88%, compared to $\bar{h}$ (86%) from our selection. In fact, the correlation is high between these variables at 0.97, from Figure 3.4. The largest component loading for the recency attribute is the date with the maximum number of hits. This variable will be sensitive to extreme values of hits, and might not give an indication of true recency of browsing behaviour. Similarly, comparing the two selections from each method shows another high correlation of 0.75. Although, this PCA extension provided different selections, the variables are highly correlated and

Attribute	Variable	PC1	PC2
Frequency	f	0.7934	-0.1929
	f_d	0.5637	0.5906
	f_e	0.2296	-0.7835
	Cumulative % of variance explained	0.7629	1.0000
Intensity	$\bar{h}$	-0.3612	-0.3477
	$\max(h)$	-0.6511	-0.0136
	$\tilde{h}$	-0.3472	-0.4145
	$\min(h)$	-0.0878	-0.6014
	$\max(h) - \min(h)$	-0.5633	0.5878
	Cumulative % of variance explained	0.9262	0.9953
Recency	$\bar{d}$	-0.4395	0.0059
	$d : h = \max(h)$	-0.4776	-0.0232
	$\tilde{d}$	-0.4427	0.0151
	$\min(d)$	-0.4385	0.7073
	$\max(d)$	-0.4365	-0.7064
	Cumulative % of variance explained	0.8585	0.9372
Variability	$\bar{g}$	-0.4415	0.1911
	$\max(d) - \min(d)$	-0.4680	-0.6493
	$\max(g)$	-0.4542	-0.1095
	$\tilde{g}$	-0.4409	0.2179
	$\min(g)$	-0.4299	0.4334
	$\max(g) - \min(g)$	-0.0243	-0.5428
	Cumulative % of variance explained	0.8785	0.9825

TABLE 3.4: A table to display the first two principal components for each attribute.

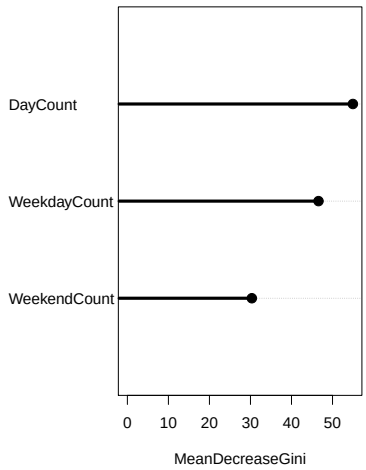
our variables have stronger interpretability.

Random forests

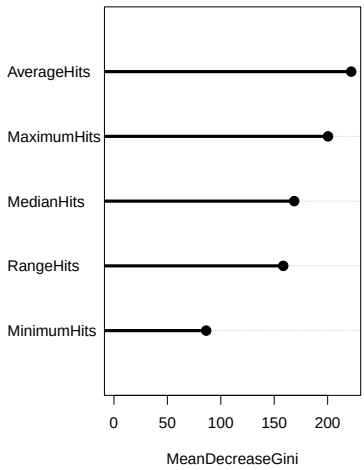
Random forests are a popular machine learning algorithm for classification which consists of a number of decision trees [32]. Every node in the decision trees is a condition on a single feature, designed to split the data set into two so that similar response values end up in the same set. The measure on which the (locally) optimal condition is chosen is called impurity. The tree-based approach used by random forests ranks possible splits by how well they improve the purity of the node, using the mean decrease in impurity over all trees (called Gini impurity). The Gini impurity of a node is the probability that a randomly chosen sample in a node would be incorrectly labelled if it was labelled by the distribution of samples in the node. Nodes with the greatest decrease in impurity happen at the start of the trees, while nodes with the least decrease in impurity occur at the end of trees. Thus, by pruning trees below a particular node, we can create a subset of the most important features.

We can inspect each feature importance plot in Figure 3.8 and use them to determine the single most important variable according to the random forest algorithm, which we compare in Table 3.3. Based on the figures, we can hand pick the top-most features to reduce the dimensionality of our data set. For both the frequency and intensity attribute, this method agrees with our variable selection as the most important with the maximum values in Figures 3.8a and 3.8b. The recency attribute finds that $\bar{d}$ is the most important feature, which we do not consider in our selection

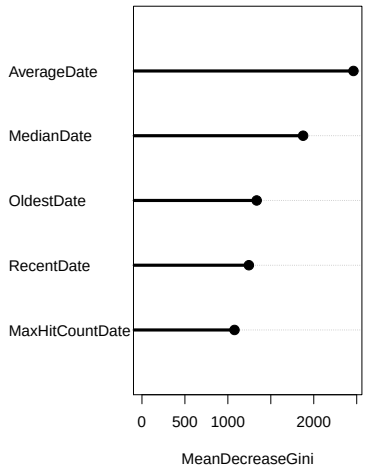
process and explains more variance in the data 82% versus 61% from Table 3.3. We made a manual choice that the intent of a profile is directly attributed the most recent visit, therefore explaining the maximum variance was not a priority. Finally, the variability attribute finds the top 2 variables are the ones we chose in our method, which is satisfying.



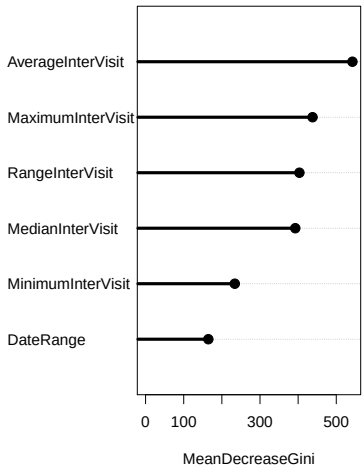
(a) Frequency attribute.



(b) Intensity attribute.



(c) Recency attribute.



(d) Variability attribute.

FIGURE 3.8: Importance plot from Random forest algorithm applied to each attribute variable subset.

3.6 Discretisation and labelling

3.6.1 Discretisation

Discretisation is the process of transferring continuous numerical variables into discrete valued variables. For the consistency of implementation and compatibility with existing software systems, we require discrete attributes and activity labels that can be interpreted by a non-statistican and provide interpretability to the existing algorithm. In statistical terms, this requires us to sacrifice some of the richness of the information in these continuous quantities, however it is a requirement of the procedure. Producing a one-dimensional intent score is a research objective, a feature of the *Carbon* platform as an essential component to describing website behaviour. The score would be used to segment the website audience, such that differing behaviour can be targeted via bespoke methods. The current algorithm is poor in the assignment of discrete levels for continuous variables; there are too few discrete levels with arbitrary thresholds and uninterpretable attribute values. This will lead to an ineffective one-dimensional summary, which lacks interpretability due to the lack of a statistical approach. We propose to use a quantile-based approach, finding discrete regions and proportions of the population they aim to represent. The large volume of data allows us to assume the quantiles are representative. The quantiles are driven by the data, hence the discretisation will be sensitive to the input values, but it should be transferable and easy to manage.

A key feature of quantiles is that we can use them to describe continuous variables on a common scale $(0-1)$, using the empirical probabilities.

$$q = F^{-1}(x) \; s.t. \; F(x) = P(X \leq x) = p$$

The inverse distribution function for continuous variables $F^{-1}(x)$ is the inverse of the cumulative distribution function (CDF). The CDF gives you probabilities of a random variable X being less than or equal to some value x. The inverse CDF, gives a value for x such that $F(x) = P(X \leq x) = p$. Where p is where random draws would fall $p \times 100$ percent of the time. The inverse of the CDF tells you what value x would make $F(x)$ return a particular probability p.

All attributes, i.e. the Y nodes in Figure 3.2b, will be discretised such that we can create interpretable levels and are comparable on a common scale. For Y nodes that have multiple X covariates, we will combine the values in some way to assign the attribute level. We want to keep the consistency of implementation as the existing algorithm, but consider more attributes. Further, we this method will equally weight each of the attributes in the final score.

We will map the continuous variables of f and $\bar{h}$ to discrete levels using the equivalent quantile probability. We display the discrete levels and quantile probability values we assign in Figures 3.9a and 3.9b. The actual discrete scores in Figure 3.9 are the midpoints between the two empirical probabilities at each level, providing scores that are not arbitrary, can be compared relatively against each other and retain interpretability. Further, this approach allows us to vary the number of discrete levels within each attribute, while obtaining scores that are still comparable between attributes on the scale between 0 and 1. Adding the intensity attribute should add a deeper level of granularity into the activity labels. For both intensity and frequency, we have right-skew distributions for the variables we have chosen, hence we use high quantile probabilities to expose the variation in the data when mapping to the common scale.

$$f_{score} = \begin{cases} 0.4 & q \in (0, 0.8] \\ 0.875 & q \in (0.8, 0.95] \\ 0.975 & q \in (0.95, 1] \end{cases}$$

(a) Frequency.

$$\bar{h}_{score} = \begin{cases} 0.4 & q \in (0, 0.8] \\ 0.875 & q \in (0.8, 0.95] \\ 0.975 & q \in (0.95, 1] \end{cases}$$

(b) Intensity.

$$\max(g)_{score} = \begin{cases} 0.3 & q \in (0, 0.6] \\ 0.75 & q \in (0.6, 0.9] \\ 0.95 & q \in (0.9, 1] \end{cases}$$

(c) Variability - Maximum.

$$\bar{g}_{score} = \begin{cases} 0.3 & q \in (0, 0.6] \\ 0.75 & q \in (0.6, 0.9] \\ 0.95 & q \in (0.9, 1] \end{cases}$$

(d) Variability - Average.

$$\max(d)_{score} = \begin{cases} 0.15 & q \in (0, 0.3] \\ 0.475 & q \in (0.3, 0.65] \\ 0.75 & q \in (0.65, 0.85] \\ 0.9125 & q \in (0.85, 0.975] \\ 0.9875 & q \in (0.975, 1] \end{cases}$$

(e) Recency.

FIGURE 3.9: The quantile mapping rules for our variable selection process.

The most recent date variable, $\max(d)$, is well spread across the range of dates, as shown in Figure 3.3i, and the quantile mapping method is displayed in Figure 3.9e. For variability, we selected two variables that we discretise independently, as shown in Figures 3.9c and 3.9d, then average the two scores to find a single value based on both variables $\bar{g}$ and $\max(g)$, using the equation below.

$$g_{score} = \frac{\bar{g}_{score} + \max(g)_{score}}{2}$$

Using the methods explained above, we reduce the number of dimensions to the four attributes. To provide a one-dimension intent score, we could decide on a prior set of weightings to determine which attributes should contribute with more or less significance to the final score. In practice, we can experiment with the weighting to obtain a variety of intent scores for a variety of problems. The final values can be normalised to any range that is needed for analysis.

3.6.2 Application of activity labels

In this section, we will look to improve the existing rule-based classifier using the attribute scores discussed in Section 3.5. Particular combinations of discretised attribute variables are associated with labels to describe similar types of behaviour patterns and increase the knowledge about each profile. For example, we can choose a set of levels: a high level of recency, intensity and frequency and a low level of variability. This label would define a subset of profiles with high engagement relative to the population, containing potentially more profitable profiles we can target with advertising, all based on a selection of levels for each attribute.

We will map the existing version of the activity labels onto key statistical variables we describe in previous sections and visualise the label structure. The existing classifier uses the variables f, r and v, as described in Figure 3.1. We focus on three activity labels:

- *Active*: A frequent pattern of behaviour
- *HyperIntent*: A significant amount of recent activity
- *Regular*: A pattern of repetitive periodic behaviour

The values that correspond to each label are outlined in Table 3.5. The proportions of profiles assigned existing labels are displayed in Table 3.6, and we immediately see that they are low in volume, due to strict rules, i.e. 0.01% for *HyperIntent* is not useful. We know from inspecting the algorithm, that the thresholds are arbitrary and there is little structure to allow us a clear interpretation of the labels. Also, there is a large number of other variables that we will use to describe activity levels, hence a novel approach will only add to the information we can gather from a profile.

Labels	Active	HyperIntent	Regular
Existing	$f \geq 3$ $r \geq 3$	$f \geq 4$ $r = 5$	$f \geq 3$ $v \geq 3$
New	$f_{score} \geq 0.875$ $\max(d)_{score} \geq 0.75$	$f_{score} \geq 0.875$ $\bar{h}_{score} \geq 0.875$ $\max(d)_{score} \geq 0.9125$	$f_{score} \geq 0.875$ $\max(d)_{score} \geq 0.475$ $g_{score} \leq 0.4$

TABLE 3.5: The activity labelling rules for the existing and new algorithm.

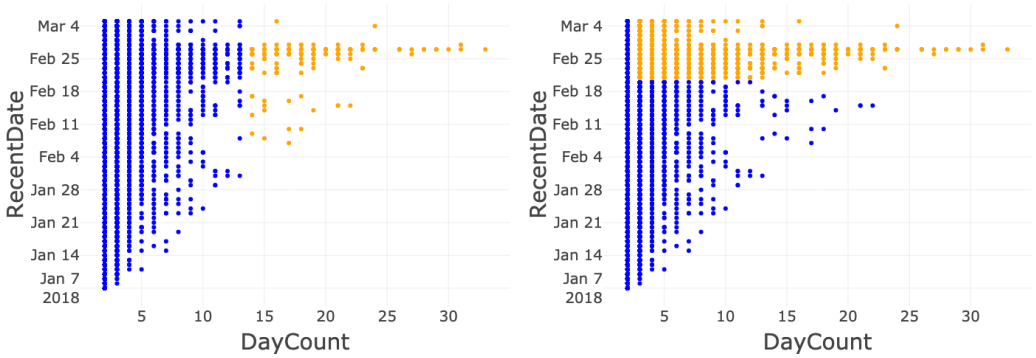
Label	Existing Count	New Count
None	30956 (98.3%)	28022 (89%)
Active	161 (0.5%)	2305 (7.3%)
HyperIntent	4 (0.01%)	128 (0.4%)
Regular	525 (1.7%)	2618 (8.3%)

TABLE 3.6: A table to display the proportions of existing *vs* new labels in the data set.

First, in Figure 3.10 we label the *Active* users, which describe profiles that are actively engaged in browsing a category. The existing method uses the total number of days visited, $f \geq 3$, which leads to a high number of visits required as seen in Figure 3.10a. The volume of this label is approximately 0.5%, from Table 3.6, and we can use quantiles to help boost the volume using a more sensible approach to defining the number of visits required for an *Active* label.

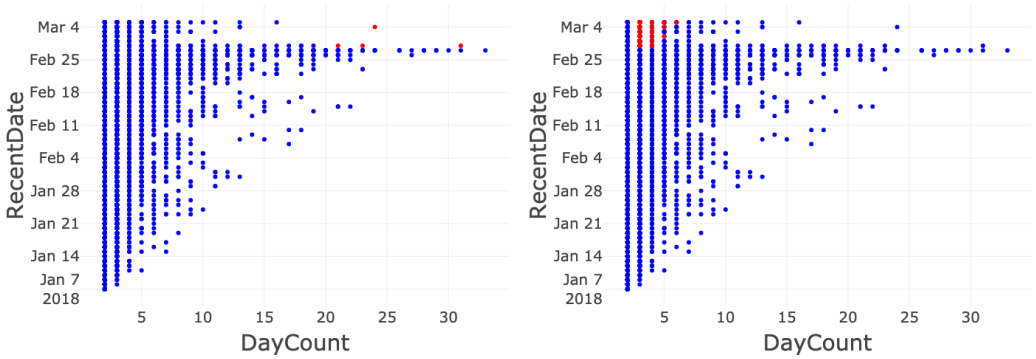
We will use a less strict threshold, of $f_{score} \geq 0.875$, to increase the number of labels to approximately 7%, while introducing a stronger emphasis on the recency attribute as seen in Figure 3.10b.

In Table 3.6, we see that only 4 observations the sample (0.01%) have been labelled as *HyperIntent*. The identified behaviour is both highly recent ($r = 5$) and highly frequent ($f \geq 4$), as seen in the existing rules in Figure 3.5. Figure 3.11a shows this combination of behavioural traits, with the red observations in the top right corner. The motivation for this label is to find the most engaged behaviour, such that targeted advertising in theory would be the most effective. We can attempt to extend the reach of this label to other profiles, while softening the rules on frequency and recency. Further, we will add the intensity attribute to this label to assess the frequency of browsing behaviour within each day, where $\bar{h}_{score} \geq 0.875$. It is clear from Figure 3.11b, there is a dense cluster of observations with the new label



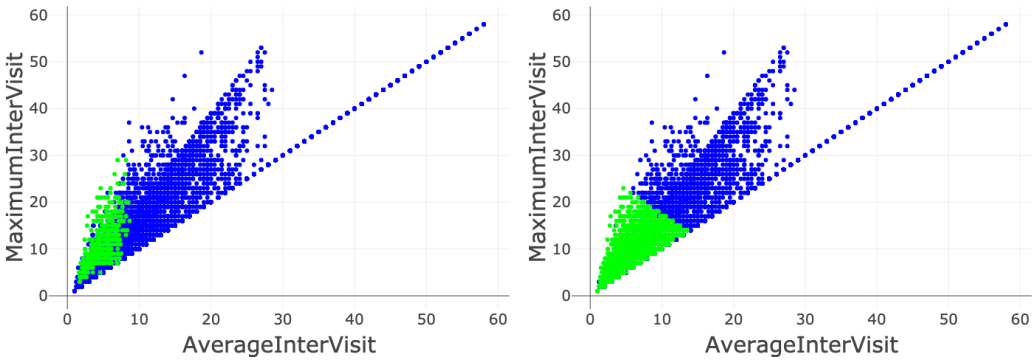
(a) Existing (b) New

FIGURE 3.10: Plots to compare the methods to label *Active* users.



(a) Existing (b) New

FIGURE 3.11: Plots to compare the methods to label *HyperIntent* users.



(a) Existing (b) New

FIGURE 3.12: Plots to compare the methods to label *Regular* users.

in red, and we have increased the proportion of labels from 0.01% to 0.4%. Interestingly, there is no overlap between the two versions of this Hyperintent label, and we believe the intensity attribute brings important additional information, causing the definition to change and overlap is not necessary.

The most common existing label is *Regular*, with the highest proportion in Table 3.6 of 1.7%, which describes behaviour that has occurred throughout the whole date range with regularity. We define regularity as visiting in at least each distinct set of 20 days over the last 60 days, hence the activity must have regularity across the entire 60 days of data. Alternatively, we may want to determine regularity over just the range of data available for that profile. This is where variability that is not currently used in the existing labels can help refine the *Regular* label. This created a better defined structure for the new label using the rules in Figure 3.5, and can be seen with a dense cluster by the origin in Figure 3.12b. The main difference is the regularity of visits does not have to be consistent throughout the entire 60 day history, but only over the the period of browsing behaviour we have observed. This is informed through the variability attribute, and creates a more rigid label structure with a higher count of over 8%.

3.7 Discussion

This chapter has set out a methodology to transform, explore and assess a large amount of online browsing data in an informed and structured process. We discovered a wide variety of statistical variables that describe a number of behavioural attributes, which we condensed to provide a one-dimensional summary. The application of this score will impact in targeted online advertising, as well as forming a new covariate to describe recent behaviour in a formal modelling process we will set out in the next chapter.

We introduced more statistical rigour to the existing algorithm, while softening some of the strict rules in place for labelling. We aggregated the original data to create new data sources, from which we added a new behavioural trait for the depth of visit to get a new perspective on the browsing behaviour intent. This will help with the interpretability of the dimension reduction algorithm, as it add further insight into the user's journey.

The next step with the intent score algorithm is to broaden the frequency attribute to look at behaviour within each category, which differs to the approach of comparison with the entire population in this chapter. We believe this add true variation within each page visit category, where the behaviour may vary significantly.

Chapter 4

Generalised Linear Models

4.1 Chapter overview

In this chapter, we will discuss formal modelling techniques which we will apply to the data we described in Section 2.2. Namely, we introduce the generalised linear model (GLM), review key elements of GLM theory and introduce zero-inflated models, motivated by the response variable in our application. This type of model is used to split the data based on *structural* and *real* zeros.

In the *Adword* data set, a keyword can only obtain clicks if it has impressions for the specified hour, i.e. the advert is actively shown to a user. We have discovered from earlier exploratory analysis that a number of observations have zero impressions, and hence zero clicks. A *structural* zero is an observation that has the combination of zero impressions and necessarily zero clicks, whereas a *real* zero has one or more impressions and zero clicks. This led us to consider zero-inflated models, variants of GLMs, that may provide more accurate modelling of our data set, where our data exhibits an excess number of zeros.

We will compare the standard GLM and zero-inflated modelling approaches, and provide results when applied to our data set. We will use a range of types of covariates, and interpret the results from the models we create. We will explore how to assess each model type, and provide model diagnostics as well as statistical evaluation measures. We explore the Poisson distribution for the response count variable.

It is important to note why we need a model with a 'good' fit. The structure of the model describes the patterns of association and interaction. The size of the model parameters determine the strength and importance of the effects. Inferences about the parameters evaluate which explanatory variables affect the response variable y, while controlling effects of possible confounding variables.

4.2 Generalised Linear Models

The main components of a GLM are [33]:

1. A response data vector,
$$\mathbf{y} = (y_1, ..., y_n),$$
assumes a probability distribution that belongs to the **exponential dispersion family** (EDF),
$$P(y|\theta, \phi) = \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y, \phi)\right].$$

2. A set of fixed explanatory variables, $\boldsymbol{X}$, and estimated coefficients, $\boldsymbol{\beta}$, form a **linear predictor**,
$$\eta = \boldsymbol{X}\boldsymbol{\beta}. \tag{4.1}$$

3. A **link function**, g, specifies a function of the expected value of Y, that is used to model the data, by relating the explanatory variables to the response distribution,
$$E[y|\beta, x] = \mu = h(\eta) = h(\boldsymbol{X\beta}) \text{ where } g \equiv h^{-1}.$$

Let's motivate each component of a GLM.

4.2.1 Exponential dispersion families

We wish to model a functional relationship between two variables, where the value of one variable, y, depends on another, x, through the value of a function $f(x)$. In general, knowing x will not make us certain about the value of the response variable y. To model the uncertainty, we take a general probability distribution on y,

$$P(y|f, x, k),$$

where the function $f(x)$ is a 'parameter' for the distribution. One way of expressing this relationship is via the expectation of y,

$$E[y|f, x, k] = f(x),$$

which can be a suitable constraint for the model.

We need to restrict the model space for tractability, so we let the output space (response) be $\mathcal{Y} \subseteq \mathbb{R}$ and let the probability density function (pdf) be an EDF. We will explain the key properties of an EDF and how we use them to define a component of a GLM. Advances in statistical theory and computer software allow us to use methods that can handle EDF distributions.

We will assume that the observations, $\mathbf{y}$, have probability distribution with parameter θ. The distribution belongs to the exponential dispersion family if it can be written in the form:

$$P(y|\theta, \phi) = \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y, \phi)\right] \tag{4.2}$$

where $\theta \in \mathbb{R}$ is the **natural** parameter, $\phi \in \mathbb{R}_{\geq 0}$ is the **dispersion** parameter, $b : \mathbb{R} \to \mathbb{R}$ and $c : \mathbb{R} \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ are both functions to be determined.

Many well-known distributions belong to the exponential dispersion family including the normal, binomial, Poisson, exponential, gamma and inverse Gaussian. We will concentrate on the Bernoulli and Poisson distributions. Table 4.1 displays the specific components to verify that they belong to the EDF. To progress with forming the basis of GLMs, we derive the key results for EDFs - the expectation and variance.

Lemma 4.2.1. *If Y has a probability distribution function in the form of Eq 4.2, then for all θ, ϕ,*

$$E[Y] = b'(\theta) \tag{4.3}$$

$$Var[Y] = \phi b''(\theta) \tag{4.4}$$

Proof. We know that,

$$\int_y P(y|\theta, \phi)\, dy = 1,$$

Component	Bernoulli	Poisson
$P(y)$	$P(y\|\pi) = \pi^y(1-\pi)^{1-y}$	$P(y\|\lambda) = \frac{\lambda^y \exp(-\lambda)}{y!}$
θ	$\log(\frac{\pi}{1-\pi})$	$\log(\lambda)$
ϕ	1	1
$b(\theta)$	$-\log(1-\pi)$	λ
$c(y,\phi)$	0	$-\log(y!)$
E[Y]	π	λ
Var[Y]	$\pi(1-\pi)$	λ
Natural response	$\mu = \frac{\exp(\eta)}{1+\exp(\eta)}$	$\mu = \exp(\eta)$
Natural link	$\eta = \log\left(\frac{\pi}{1-\pi}\right) = \text{logit}(\pi)$	$\eta = \log(\mu)$

TABLE 4.1: Bernoulli and Poisson distributions as members of the exponential dispersion family.

so we begin by integrating Eq 4.2 over all possible values for y,

$$\begin{aligned} 1 &= \int_y \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy \\ &= \exp\left[\frac{-b(\theta)}{\phi}\right] \int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \end{aligned}$$

hence,

$$\exp\left[\frac{b(\theta)}{\phi}\right] = \int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy$$

taking the natural logarithm gives us,

$$\frac{b(\theta)}{\phi} = \log\left\{\int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy\right\}. \tag{4.5}$$

Eq 4.5 determines b in terms of ϕ, θ, c. By differentiating Eq 4.5 w.r.t. θ, and using the Leibniz integral rule to pull the derivative into the integral (as the space of y is not a function of θ) below we see that

$$\begin{aligned} \frac{d}{d\theta}\left[\frac{b(\theta)}{\phi}\right] &= \frac{1}{\int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right]} \times \frac{d}{d\theta}\left\{\int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy\right\} \\ \frac{b'(\theta)}{\phi} &= \frac{1}{\int_y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right]} \int_y \frac{y}{\phi} \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \\ &= \frac{1}{\exp\left[\frac{b(\theta)}{\phi}\right]} \int_y \frac{y}{\phi} \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \\ &= \exp\left[\frac{-b(\theta)}{\phi}\right] \int_y \frac{y}{\phi} \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy. \end{aligned}$$

Rearranging gives us,

$$\begin{aligned}\frac{b'(\theta)}{\phi} &= \int_y \frac{y}{\phi} \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy \\ &= \frac{1}{\phi}\int_y yP(y|\theta,\phi)\, dy \\ &= \frac{1}{\phi}E[y|\theta,\phi]\end{aligned}$$

hence,

$$E[y|\theta,\phi] = b'(\theta). \tag{4.6}$$

Differentiating Eq 4.6 w.r.t. θ and again using the Leibniz rule below gives us,

$$\begin{aligned}\frac{d}{d\theta}\left[b'(\theta)\right] &= \frac{d}{d\theta}\left\{\int_y y \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy\right\} \\ b''(\theta) &= \frac{d}{d\theta}\left\{\exp\left[\frac{-b(\theta)}{\phi}\right]\right\} \times \int_y y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \\ &\quad + \exp\left[\frac{-b(\theta)}{\phi}\right] \times \frac{d}{d\theta}\left\{\int_y y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy\right\} \\ &= \frac{-b'(\theta)}{\phi} \times \exp\left[\frac{-b(\theta)}{\phi}\right] \times \int_y y \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \\ &\quad + \exp\left[\frac{-b(\theta)}{\phi}\right] \times \int_y \frac{y^2}{\phi} \exp\left[\frac{y\theta}{\phi} + c(y,\phi)\right] dy \\ &= \frac{-b'(\theta)}{\phi} \int_y y \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy \\ &\quad + \frac{1}{\phi}\int_y y^2 \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy \\ &= \frac{-b'(\theta)}{\phi} \times \left[b'(\theta)\right] + \frac{1}{\phi}\int_y y^2 \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy \\ &= \frac{1}{\phi}\left\{-\left[b'(\theta)\right]^2 + \int_y y^2 \exp\left[\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right] dy\right\} \\ &= \frac{1}{\phi}\left\{-E[Y|\theta,\phi]^2 + E[Y^2|\theta,\phi]\right\} \\ &= \frac{1}{\phi}Var[Y|\theta,\phi]\end{aligned}$$

□

These are useful properties as knowing the general form of the expectation and variance of any exponential dispersion family can be used to determine features that you may be looking for in your data. In specific situations, we can use these equations to find the asymptotic properties of the maximum likelihood estimate.

4.2.2 Linear predictor and the link function

Another restriction we make is that instead of modelling the expected value directly, the relationship between the response and explanatory variables, i.e. Eq 4.1, need not be of the simple linear form. We will introduce an injective differentiable response

function h, relating $\hat{y} = E(y) = \mu$ to the linear component, $\eta = \boldsymbol{X\beta}$, such that

$$\mu = E[y|\beta, x] = h(\eta) = h(\boldsymbol{X\beta}).$$

We can also write that,

$$g(\mu) = \boldsymbol{X\beta},$$

where $g \equiv h^{-1}$ is the link function. Some examples are the identity, log, logit, probit. When the link function makes the linear predictor η the same as the canonical parameter θ, we say that we have a canonical (natural) link.

We will show how to find the natural link. We know from EDFs, Eq 4.6,

$$\mu = E[Y|\theta, \phi] = b'(\theta) \tag{4.7}$$

and we know by definition of the GLM,

$$\mu = E[Y|\beta, x] = h(\boldsymbol{X\beta}). \tag{4.8}$$

By combining Eq 4.7 and 4.8,

$$h(\boldsymbol{X\beta}) = b'(\theta),$$

and simplifying our choice for h, i.e. $h = b'$,

$$\begin{aligned} h(\boldsymbol{X\beta}) &= h(\theta) \\ \theta &= \boldsymbol{X\beta}, \end{aligned}$$

hence,

$$g = (b')^{-1}.$$

We display the natural link functions for the two distributions, Bernoulli and Poisson, in Table 4.1. In practice, we require a considerable amount of computation involving numerical optimisation of non-linear functions. Procedures to do these calculations are now included in many statistical programs.

4.2.3 Maximum likelihood estimation

To fit a GLM to a data set, we need a method to estimate the parameters β. We will outline the approach to finding the maximum likelihood estimate for both the logistic and Poisson regression models.

We begin by finding the log-likelihood function for a logistic regression model, by stating the Bernoulli distribution from Table 4.1 for a single observation y_i:

$$P(y_i|\pi_i) = \pi_i^{y_i}(1-\pi_i)^{1-y_i}.$$

We can write the likelihood function, l, given our data, $\mathbf{y} = (y_1, ..., y_n)$ as [34]

$$l(\boldsymbol{\pi}|\mathbf{y}, \mathbf{x}) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{1-y_i} \tag{4.9}$$

$$= \prod_{i=1}^{n} \left(\frac{\pi_i}{1-\pi_i}\right)^{y_i} (1-\pi_i). \tag{4.10}$$

From Table 4.1, we have the natural link for the Bernoulli distribution, and we can rearrange as:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \eta = x_i\beta \tag{4.11}$$

$$1-\pi_i = \frac{1}{1+\exp(x_i\beta)}. \tag{4.12}$$

Substituting Eq 4.11 and 4.12 into 4.10,

$$l(\beta) = \prod_{i=1}^{n} [\exp(x_i\beta)]^{y_i}\left[\frac{1}{1+\exp(x_i\beta)}\right]$$

hence the log-likelihood, $L = \log(l)$, for the logistic regression model is

$$\begin{aligned} L(\beta) &= \sum_{i=1}^{n} \log\left\{[\exp(x_i\beta)]^{y_i}\left[\frac{1}{1+\exp(x_i\beta)}\right]\right\} \\ &= \sum_{i=1}^{n} y_i x_i\beta - \sum_{i=1}^{n} \log\left[1+\exp(x_i\beta)\right]. \end{aligned} \tag{4.13}$$

To find the likelihood function for Poisson regression, we begin by stating the Poisson distribution from Table 4.1 for a single observation y_i [35],

$$P(y_i|\lambda) = \frac{\lambda_i^{y_i}\exp(-\lambda_i)}{y_i!}.$$

Thus, we can write the likelihood, l, given our data as,

$$\begin{aligned} l(\boldsymbol{\lambda}|\mathbf{y},\mathbf{x}) &= \prod_{i=1}^{n}\left[\frac{\lambda_i^{y_i}\exp(-\lambda_i)}{y_i!}\right] \\ &= \left(\prod_{i=1}^{n}\frac{\lambda_i^{y_i}}{y_i!}\right) \times \exp\left(-\sum_{i=1}^{n}\lambda_i\right). \end{aligned}$$

Using the natural link for the Poisson distribution from Table 4.1, $\lambda_i = \mu_i = \exp(x_i\beta)$, the log-likelihood, L, is

$$\begin{aligned} L(\boldsymbol{\lambda}) &= \sum_{i=1}^{n}\log\left(\frac{\lambda_i^{y_i}}{y_i!}\right) - \sum_{i=1}^{n}\lambda_i \\ &= \sum_{i=1}^{n} y_i\log(\lambda_i) - \sum_{i=1}^{n}\lambda_i - \sum_{i=1}^{n}\log(y_i!) \\ L(\beta) &= \sum_{i=1}^{n} y_i x_i\beta - \sum_{i=1}^{n}\exp(x_i\beta) - \sum_{i=1}^{n}\log(y_i!). \end{aligned} \tag{4.14}$$

To maximise the log-likelihood functions, Eq 4.13 and 4.14, we set the partial derivative with respect to β equal to zero, i.e. $\frac{\partial L}{\partial \beta} = 0$. We call this the score equation. For p components of β, there will be p such equations to solve. There is no analytic solution, hence we must rely on numerical methods. There is an algorithm to help us solve this set of equations called 'iteratively re-weighted least squares' (IRLS), which is based on the Newton-Raphson algorithm [36]. The key steps are listed below:

1. Guess the estimates for $\hat{\beta}^{(0)}$
2. For the log-likelihood function, L, evaluate the vector of partial derivatives $L'\left(\hat{\beta}^{(0)}\right)$, equate the result to zero and solve.
3. Update the values of β using the information from the partial derivatives to find $\hat{\beta}^{(1)}$
4. Iterate steps 2-3 until the difference between consecutive estimates, i.e. $|\hat{\beta}^{(i)} - \hat{\beta}^{(i-1)}| < \epsilon$, where ϵ is some threshold close to zero.

Further, we can use the second partial derivatives, with respect to β, to obtain Fisher's information and properties such as the expectation and asymptotic distributions. In our analysis, we will not implement these methods directly and we will use standard methods from R [37].

4.2.4 Coefficient estimates and prediction

Given a fitted GLM, such that we have the estimated coefficients $\hat{\boldsymbol{\beta}}$, then for a new predictor variable $\boldsymbol{x_0}$, we can compute the predicted value of the corresponding response y_0 using the model as:

$$E[y_0] = h(\hat{\eta}_0) = h(\boldsymbol{x_0^T}\hat{\boldsymbol{\beta}}). \tag{4.15}$$

The coefficient estimates, $\hat{\boldsymbol{\beta}}$, inform us about the relationship between the independent variables and the dependent variable.

For a logistic regression model, the dependent variable is on the logit scale, i.e. we predict the log odds using Eq 4.15. Because these coefficients are in log-odds units, they are often difficult to interpret, so one way to interpret the estimated coefficients is to find the odds ratio, by the transform, $e^{\hat{\beta}_i}$. These estimates quantify the amount of increase in the predicted odds ratio that would be predicted by a 1 unit increase in the covariate, when maintaining all other covariates constant. Given the different scales and types of our covariates, these estimates are not comparable across variables. We use the the estimates to gain insight into how the model would behave for a change to each covariate independently. Note, for the independent variables which are not significant, the coefficients are not significantly different from 0, which should be taken into account when interpreting the coefficients. For this we inspect the p-values testing whether the coefficients are statistically significant.

4.2.5 Model diagnostics

In GLMs, we can assess the 'fit' of the model using residuals, by examining the adequacy of the model and investigating potential outliers. Residuals are a measure of the distance of the observations from the fitted regression line. We interpret the residuals as the quantity of the response predictions that are not explained by the model. An ideal residual in linear regression, would look like an i.i.d. sample that exhibits homogeneity, normality, and independence. For GLMs, we will look at two types of residuals: raw and Pearson. The Pearson residuals account for heteroscedasticity by standardising the residuals.

1. Raw residual, for an observation y_i and model prediction $\hat{\mu}_i$:

$$r_i = y_i - \hat{\mu}_i$$

2. Pearson residual, for an observation y_i and model variance function $V(\hat{\mu}_i)$:

$$r_i^p = \frac{y_i - \hat{\mu}_i}{\sqrt{V(\hat{\mu}_i)}},$$

4.3 Zero-inflated models

We can assume that the number of clicks a keyword receives every hour could be modelled with a Poisson distribution. We know from Section 2.2.2 that there are a high number of observations in the *Adwords* data set that have zero clicks - over 99.3%. We will explore traditional Poisson regression models, however we may expect them to perform poorly in this application due to the considerable number of zeros.

We will explore the zero-inflated Poisson model (ZIP), which can be used in applications where the count data exhibits over-dispersion and excess zeros. ZIP theory is provided and applied to the context of manufacturing defect data [38]. This example assumes that the zero-inflated distribution is appropriate, where the population considered consists of two sub-populations based on two states: one state where the possibility of a defect is extremely rare and another where defects are possible but unlikely. Other areas of application for ZIPs include road safety, species abundance and sexual behaviour [39], [40], [41].

As discussed previously, if we wish to predict clicks, our data exhibits structural zeros. We display a summary of the joint distribution for impressions and clicks in Table 4.2, where 94.9% of our observations are structural zeros. A structural zero occurs when it is inevitable there will be zero clicks as are there are zero impressions, whereas real zeros occur by chance. The counts for clicks lie in the set, $y_{kt} \in \mathbb{Z}_{\geq 0}$.

We have a high number of structural zeros in our data set, and one modelling approach would be to regress on the set of observations where $x_{kt} \geq 1$. However, the aim of our study was to attribute a prediction to every keyword. The subjective generation of keywords means that measuring the potential performance of keywords even with very few impressions, could be paramount to a successful advertising campaign.

Similarly to GLMs, ZIP regression can be performed on both numeric and categorical variables. Let's introduce the mixture distribution and regression models for ZIP.

		Clicks		
		$y_{kt} = 0$	$y_{kt} \geq 1$	Total
Impressions	$x_{kt} = 0$	230555 (94.9%)	0	230555
	$x_{kt} \geq 1$	10610 (4.4%)	1857 (0.7%)	12467
	Total	241165	1857	243022

TABLE 4.2: A table displaying the joint distribution of impressions and clicks.

4.3.1 Zero-inflated Poisson model

Suppose that for each observation y_i, there are two possible cases. In the first case, the count is zero with probability p_i, i.e. a binary distribution that generates the structural zeros. However, in the second case, the counts (including zeros), are generated by a Poisson model with mean λ_i which occurs with probability $1 - p_i$. We

want a non-constant probability p_i and the mean λ_i to differ for each response y_i in the zero-inflation and Poisson count model. We have an independent response variable, $Y = (y_1, y_2, \ldots, y_n)$, where λ_i and p_i change with the value of i, so formally:

$$\begin{aligned} y_i &= 0 && \text{with probability } p_i, \\ y_i &\sim \text{Poisson}(\lambda_i) && \text{with probability } 1 - p_i. \end{aligned}$$

The assumed distribution combines the Poisson distribution and the logit link function. The probability distribution function of the ZIP model for a random variable, Y, can be written as:

$$P(Y = y_i | x_i, p_i) = \begin{cases} p_i + (1 - p_i) \exp(-\lambda_i) & y_i = 0 \\ (1 - p_i) \frac{\exp(-\lambda_i)\lambda_i^{y_i}}{y_i!} & y_i \geq 1 \end{cases}$$

where the outcome variable y_j has any non-negative integer value, λ_i is the expected Poisson count and p_i is the probability of a structural zero for the i_{th} observation. We can express both the mean and variance of the ZIP model as:

$$\begin{aligned} E[Y_i] &= (1 - p_i)\lambda_i \\ Var[Y_i] &= \lambda_i(1 - p_i)(1 + p_i\lambda_i). \end{aligned}$$

The ZIP model has two components, one that models the count distribution when impressions are larger than zero as a function of covariates x_i. The other component models the behaviour of p as a function of covariates, z_i. The two link functions for a ZIP are [42]:

$$\log(\lambda_i) = x_i^T \beta \tag{4.16}$$

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1 - p_i}\right) = z_i^T \gamma \tag{4.17}$$

where β and γ represent two coefficient vectors of covariates x_i and z_i, where $z_i \subseteq x_i$. We use the log link of the mean λ_i for the Poisson model, and the logit link function of p_i to model the structural zeros. We will investigate if p_i varies with i, however in our application we may see that p is constant because of the impression and click relationship. The ZIP model should enable us to better understand the effect of covariates by distinguishing the effects of each specific covariate on structural zeros and on the count response.

4.3.2 Maximum likelihood estimation

The standard technique to estimates the parameter for a ZIP model is based on the likelihood, similar to Section 4.2.3. We can express the joint likelihood, l, as [42],

$$\begin{aligned} l(\mathbf{p}, \boldsymbol{\lambda} | \mathbf{y}, \mathbf{x}) &= \prod_{i=1}^{n} P(y_i | x_i, p_i) \\ &= \prod_{y_i = 0} \left[p_i + (1 - p_i) \exp(-\lambda_i)\right] \times \prod_{y_i \geq 1} \left[(1 - p_i) \frac{\exp(-\lambda_i)\, \lambda_i^{y_i}}{y_i!}\right]. \end{aligned}$$

We can rearrange Eq 4.16 and 4.17 to the form of:

$$\lambda_i = \exp\left(x_i^T \beta\right) \tag{4.18}$$

$$\frac{p_i}{1-p_i} = \exp\left(z_i^T \gamma\right). \tag{4.19}$$

Further rearrangement of Eq 4.19 gives us,

$$p_i = \frac{\exp\left(z_i^T \gamma\right)}{1+\exp\left(z_i^T \gamma\right)} \tag{4.20}$$

$$1-p_i = \frac{1}{1+\exp\left(z_i^T \gamma\right)}. \tag{4.21}$$

Hence the log-likelihood function, denoted by $L = \log(l)$, is

$$\begin{aligned}
L(\mathbf{p}, \boldsymbol{\lambda}|\mathbf{y}, \mathbf{x}) &= \sum_{y_i=0} \log\left[p_i + (1-p_i)\exp\left(-\lambda_i\right)\right] + \sum_{y_i \geq 1} \log\left[(1-p_i)\exp\left(-\lambda_i\right)\frac{\lambda_i^{y_i}}{y_i!}\right] \\
&= \sum_{y_i=0} \log\left[p_i + (1-p_i)\exp\left(-\lambda_i\right)\right] + \sum_{y_i \geq 1} \log(1-p_i) \\
&\quad + \sum_{y_i \geq 1} \log\left[\exp\left(-\lambda_i\right)\right] + \sum_{y_i \geq 1} \log\left(\lambda_i^{y_i}\right) - \sum_{y_i \geq 1} \log\left(y_i!\right) \\
&= \sum_{y_i=0} \log\left[p_i + (1-p_i)\exp\left(-\lambda_i\right)\right] + \sum_{y_i \geq 1} \log(1-p_i) \\
&\quad - \sum_{y_i \geq 1} \lambda_i + \sum_{y_i \geq 1} y_i \log\left(\lambda_i\right) - \sum_{y_i \geq 1} \log\left(y_i!\right).
\end{aligned}$$

Substituting in Eq 4.18,

$$\begin{aligned}
L(\mathbf{p}, \boldsymbol{\lambda}|\mathbf{y}, \mathbf{x}) &= \sum_{y_i=0} \log\left\{p_i + (1-p_i)\exp\left[-\exp\left(x_i^T \beta\right)\right]\right\} + \sum_{y_i \geq 1} \log(1-p_i) \\
&\quad - \sum_{y_i \geq 1} \exp\left(x_i^T \beta\right) + \sum_{y_i \geq 1} y_i \log\left[\exp\left(x_i^T \beta\right)\right] - \sum_{y_i \geq 1} \log\left(y_i!\right) \\
&= \sum_{y_i=0} \log\left\{p_i + (1-p_i)\exp\left[-\exp\left(x_i^T \beta\right)\right]\right\} + \sum_{y_i \geq 1} \log(1-p_i) \\
&\quad - \sum_{y_i \geq 1} \exp\left(x_i^T \beta\right) + \sum_{y_i \geq 1} y_i x_i^T \beta - \sum_{y_i \geq 1} \log\left(y_i!\right).
\end{aligned}$$

Substituting in Eq 4.20 and 4.21,

$$\begin{aligned}
L(\beta,\gamma) &= \sum_{y_i=0} \log\left\{\left[\frac{\exp\left(z_i^T\gamma\right)}{1+\exp\left(z_i^T\gamma\right)}\right] + \left[\frac{1}{1+\exp\left(z_i^T\gamma\right)}\right]\exp\left[-\exp\left(x_i^T\beta\right)\right]\right\} \\
&\quad + \sum_{y_i\geq 1}\log\left[\frac{1}{1+\exp\left(z_i^T\gamma\right)}\right] + \sum_{y_i\geq 1}\left[y_i x_i^T\beta - \exp\left(x_i^T\beta\right)\right] - \sum_{y_i\geq 1}\log\left(y_i!\right) \\
&= \sum_{y_i=0}\log\left\{\exp\left(z_i^T\gamma\right)+\exp\left[-\exp\left(x_i^T\beta\right)\right]\right\} - \sum_{y_i=0}\log\left[1+\exp\left(z_i^T\gamma\right)\right] \\
&\quad - \sum_{y_i\geq 1}\log\left[1+\exp\left(z_i^T\gamma\right)\right] + \sum_{y_i\geq 1}\left[y_i x_i^T\beta - \exp\left(x_i^T\beta\right)\right] - \sum_{y_i\geq 1}\log\left(y_i!\right) \\
&= \sum_{y_i=0}\log\left\{\exp\left(z_i^T\gamma\right)+\exp\left[-\exp\left(x_i^T\beta\right)\right]\right\} - \sum_{y_i\geq 0}\log\left[1+\exp\left(z_i^T\gamma\right)\right] \\
&\quad + \sum_{y_i\geq 1}\left[y_i x_i^T\beta - \exp\left(x_i^T\beta\right)\right] - \sum_{y_i\geq 1}\log\left(y_i!\right).
\end{aligned}$$

Let

$$I\left(y_i=0\right) = \begin{cases} 1 & y_i = 0 \\ 0 & \text{otherwise} \end{cases}$$

be an indicator function to determine observations that take the value 0. Then we can write the joint log-likelihood function for the ZIP regression model as:

$$\begin{aligned}
L(\beta,\gamma) &= \sum_{i=1}^{n} I\left(y_i=0\right)\log\left\{\exp\left(z_i^T\gamma\right)+\exp\left[-\exp\left(x_i^T\beta\right)\right]\right\} \\
&\quad - \sum_{i=1}^{n}\log\left[1+\exp\left(z_i^T\gamma\right)\right] \\
&\quad + \sum_{i=1}^{n}\left[1-I\left(y_i=0\right)\right]\left[y_i x_i^T\beta - \exp\left(x_i^T\beta\right)\right].
\end{aligned}$$

Since the term involving $y_i!$ does not affect parameter estimation, we remove it from the equation above. We can find the derivatives of L, but the solutions are not explicit, hence we must use a numerical method to solve them. In practice, this will be the EM algorithm which is well known, and we will not discuss in this thesis [43].

4.3.3 Prediction

Given we have parameter estimates, $(\hat{\beta},\hat{\gamma})$, we can use the ZIP model to predict the response variable. The expectation of y_i given x_i is [42],

$$E[y_i|x_i] = (1-\hat{p}_i)e^{x_i\hat{\beta}}$$

where $\hat{p}_i$ is the predicted probability of observation i being a structural zero, and p_i is estimated by the following,

$$\hat{p}_i = \frac{e^{z_i\hat{\gamma}}}{1+e^{z_i\hat{\gamma}}}.$$

The coefficients are interpreted in a similar way to standard GLM theory, as in the case of a ZIP model, the two components are a simple logistic regression and Poisson count model.

4.4 Application to Adwords data set

The application of the GLM theory we will use is the data set that we explored in Section 2.2.1, which consists of the performance of keywords in online advertising. We have a number of suitable covariates, and finding a model with a 'good fit', would influence new keywords and bidding strategies to improve the results of the campaign. First, we randomly split the full data set into a training and test set, using the ratio 60 : 40. We do this so that we can train our models, and then use unobserved test data to check for model overfitting.

We will model the dependent variable that counts the number of clicks for a keyword, based on a number of key covariates. We select the variables in Table 4.3, where the target variable y_{kt}, *ClicksDelta* (a count outcome), or $\tilde{y}_{kt}$, *Clicks* (a binary outcome), depending on the model type. The explanatory variables that we consider are driven from the data exploration, from which we know that to obtain a click for a keyword, impressions of the advert must be served to users, x_{kt}. We also know that there are two main types of keyword, m_{kt}, an exact or broad match. We manually set the maximum price we will pay for a click, c_{kt}, so we will determine if this affects the performance. The performance metrics for a keyword are obtained over time, t, and we will use the hour to look for time-based effects on clicks. We have discretised the time variable into 6-hour intervals, h_{kt}, to represent different parts of the day as shown in Table 4.3. While finer discretisations would preserve more information from the original variable, a model that updates every hour would not be practical to implement. As a compromise, discretising into 6 or 12 hours windows could provide insights into the morning, afternoon, evening and night time behaviours. For the majority of observations, we do not have a value for the creative quality score, q_{kt}, however we know from the data exploration that a high score correlates with high keyword performance. A covariate that we could explore in future research is a transformation of the count of impressions, x_{kt}, to a binary variable of zero impressions or at least one impression. However, in this application the advertiser pays per click, emphasising the importance of the impression count, hence predicting a count over a binary outcome would be more useful in practice.

We use the GLM theory we set out in this chapter, and we want to understand which variables are strong predictors for the outcome we wish to model. Specifically, we will fit a logistic and a Poisson regression model, followed by a zero-inflated Poisson model. We use the same covariates for each of the standard GLM models, and for the ZIP model we must add the extra zero-inflated covariates.

Variable	**Type**	**Notation**
ClicksDelta	Numerical	$y_{kt} \in \mathbb{Z}_{\geq 0}$
Clicks	Binary	$\tilde{y}_{kt} = \begin{cases} 0 & y_i = 0 \\ 1 & y_i \geq 1 \end{cases}$
CreativeQuality	Categorical	$q_{kt} = \{\text{Above average, Average, Below average, Not Applicable}\}$
ImpressionsDelta	Numerical	$x_{kt} \in \mathbb{Z}_{\geq 0}$
KeywordMatchType	Binary	$m_{kt} = \{\text{Exact, Broad}\}$
MaxCpc	Continuous	$c_{kt} = [0, 15]$
HourOfDay	Categorical	$h_{kt} = \{\text{12am-6am, 6am-12pm, 12pm-6pm, 6pm-12am}\}$

TABLE 4.3: The variables we selected for modelling in this section.

4.4.1 Logistic regression

We present the formulas to indicate which covariates we use to fit the logistic regression model to our training data set [37]. Eq 4.22 and 4.23 display the covariate names and equivalent notation. We wish to model the dependent variable of at least one click or no clicks for a keyword, $\tilde{y}_{kt}$. We assume all variables are independent for this simple model, and do not fit any interactions between covariates. We will keep the covariate structure consistent for all types of models we explore in later sections.

$$\begin{aligned}\text{logit}(\text{Clicks}) &= \alpha + \beta_1 \text{ImpressionDelta} + \beta_2 \text{KeywordMatchType} \\ &\quad + \beta_3 \text{MaxCpc} + \beta_4 \text{CreativeQuality} + \beta_5 \text{HourOfDay} \qquad (4.22)\\ \text{logit}(\tilde{y}_{kt}) &= \alpha + \beta_1 x_{kt} + \beta_2 m_{kt} + \beta_3 c_{kt} + \beta_4 q_{kt} + \beta_5 h_{kt} \qquad (4.23)\end{aligned}$$

The baseline level for each factor variable is shown in Table 4.4. We present a summary of the fitted model in Table 4.5, the coefficient estimates and standard errors, along with the p-values to assess whether the covariates are significant. Hence, we can see that all variables, except when h_{kt} is either '12pm-6pm' or '6pm-12am', are significant at the 5% level in this model. In future models, we could split h_{kt} into a binary variable representing '12pm-12am' and '12am-12pm', such that all covariates may be significant in the model. The final row provides the log-likelihood, -4056.13, for the fitted logistic regression model, which we will use to compare in Section 4.4.4.

Symbol	Covariate
m_{kt}	KeywordMatchType - Broad
q_{kt}	CreativeQuality - Average
h_{kt}	HourOfDay - 6am-12pm

TABLE 4.4: The baseline levels for each factor covariate.

Symbol	Covariate	β estimate	SE	p-value	Odds ratio
α	Intercept	-2.1941	0.1149	2×10^{-16}	0.1115
x_{kt}	ImpressionDelta	0.0681	0.0025	2×10^{-16}	1.0704
m_{kt}	KeywordMatchType - Exact	-0.2498	0.0765	0.00109	0.7789
c_{kt}	MaxCpc	-0.1677	0.0079	2×10^{-16}	0.8456
q_{kt}	CreativeQuality - Above average	0.1918	0.0826	0.0202	1.2114
	CreativeQuality - Below average	-0.7337	0.1129	8.13×10^{-10}	0.4801
	CreativeQuality - No score	-2.4667	0.1377	2×10^{-16}	0.0849
h_{kt}	HourOfDay - 12am-6am	-1.4095	0.1402	2×10^{-16}	0.2443
	HourOfDay - 12pm-6pm	0.0723	0.0868	0.4048	1.0750
	HourOfDay - 6pm-12am	0.0332	0.0863	0.7006	1.0337
	Log-likelihood	-4056.13			

TABLE 4.5: The parameter estimates for the fitted logistic regression model.

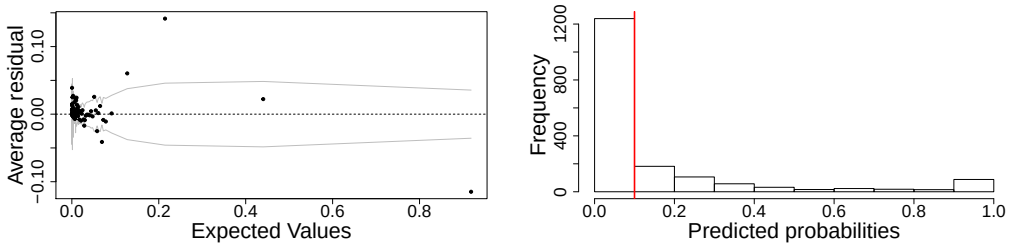
The odds ratio column in Table 4.5 provides the exponential of the coefficient estimates, and we use this to interpret each covariate. For every unit count of an impression for a keyword, there is an increase of 1.07, on the likelihood of a click, which is sensible as there is then more chance to obtain at least one click. The exact match type reduces the chance of click by 0.78, as opposed to the reference category

of broad match type. One reason for this could be that a broad match type may cause more impressions (as it doesn't have to be an exact match to the search term), in turn more chance for a click. Alternatively, the keywords are generated subjectively, hence they might not match the viewer's opinion on the advert, i.e. the exactness of the match might not be accurate based on subjectivity.

For a unit increase of c_{kt}, the odds of a click decrease by 0.85, leading us to consider reducing the maximum cost we set. This may at first appear counter-intuitive, however we know from the distribution in Figure 2.2i that c_{kt} has a high count of values at the maximum. The response to c_{kt} does not appear to be linear, as below a threshold we would not win any bids to show an impression at all, hence we should be careful with the interpretation. Further analysis would be required to assess the impact of cost. The covariate *CreativeQuality*, q_{kt}, shows an odds increase of 1.21 for above average scores, a decrease of 0.48 for below average scores, and a large decrease for a no score of 0.08. The creative quality score appears to be a good indicator of keyword performance, as better scores intuitively generate more clicks. There is a large decrease of odds of 0.23 for a click in the hours of 12am-6am, this is most likely due to a lack of online traffic through the night.

Starting with model diagnostics, Figure 4.1a shows a scatter plot of binned residuals from the training observations versus the fitted values from the model. The residuals have been grouped and aggregated to find the average residual over an interval of similar predicted values from the model. The grey lines represent ± 2 standard error (SE) bands, which we would expect to contain about 95% of the observations [44]. This model does appear reasonable (though it is tricky to tell with the bunching on the left hand side of the plot), the majority of the values fall within SE bands. The bunching occurs due to the majority of observations with low probabilities of obtaining a click. There is an extreme value in the bottom right of the plot, suggesting predictions are unstable at the higher probabilities. We applied the fitted model to the test data set and Figure 4.1b shows a histogram of the predicted probabilities. The probabilities are heavily skewed towards zero, which is intrinsic in the data set as the labels are unbalanced and skewed towards no clicks, so we must proceed carefully when using these predictions.

We discretise the predictions such that we can obtain a confusion matrix, with key statistical metrics which provide an interpretable summary of the predictive capability of the model. Using Figure 4.1b, we choose a fixed threshold of 0.1 to determine the binary outcome, i.e. click or no click. Further tuning could be done in future work to find an optimal threshold if a binary outcome was required in practice. As the proportion of click versus no click in our data is 0.7%, a 10% chance of a click seems reasonable and note that cost is only incurred upon a click, so we can relax this threshold. Table 4.6 displays the confusion matrix for the click predictions. Note that the performance here is determined by the arbitrary threshold of 0.1 chosen above, and we see that our model under predicts but with a high number of true negatives. In future work we could explore calibrating this threshold to mitigate this under-prediction. If we require a binary outcome, however we have chosen a value for illustrative purposes and to make direct comparisons between different types of models. From these values, we can easily find the accuracy (99.2%), recall (44.9%) and precision (54.6%). We prioritise obtaining a high value for precision, whereas we are not concerned about recall until budget constraints are introduced.



(a) Binned residuals versus fitted values. (b) Histogram of predicted probabilities.

FIGURE 4.1: Plots relating to the fitted logistic regression model.

		Reference y_{kt}		
		0	1	Total
Predicted $\hat{y}_{kt}$	0	59996	200	60196
	1	296	241	537
	Total	60292	441	60733

TABLE 4.6: The confusion matrix for the click predictions from the logistic regression model on the test set.

4.4.2 Poisson regression

The Poisson regression model fitted to our training data set is given below in Eq 4.24 and 4.25, which display the covariate names and equivalent notation. We have modelled the dependent variable of the number of clicks for a keyword, y_{kt}. We fit the same covariates as the logistic regression model, where we assume all variables are independent in a simple model, and do not fit any interactions between covariates [37].

$$\begin{aligned}\log(\text{ClicksDelta}) &= \alpha + \beta_1 \text{ImpressionDelta} + \beta_2 \text{KeywordMatchType} \\ &\quad + \beta_3 \text{MaxCpc} + \beta_4 \text{CreativeQuality} + \beta_5 \text{HourOfDay} \qquad (4.24)\\ \log(y_{kt}) &= \alpha + \beta_1 x_{kt} + \beta_2 m_{kt} + \beta_3 c_{kt} + \beta_4 q_{kt} + \beta_5 h_{kt} \qquad (4.25)\end{aligned}$$

The Poisson regression coefficients for each of the variables are given along with the standard errors, and p-values for the coefficients in Table 4.7. We interpret the factor on the expected count outcome, like in a standard linear regression model. Similar to the logistic model, we see that all variables except h_{kt} = '12pm-6pm' are significant at the 5% level in this model. Similarly, this covariate value was not significant in the logistic regression model, where both models had the same baseline of '6am-12pm'. In future work, this is a further suggestion that combining these values might lead to a better model fit. The final row provides the log-likelihood, -6441.489, for the fitted Poisson regression model.

For every unit count of an impression for a keyword, there is a smaller increase of 1.01 on the click count compared to the logistic model. The exact match type, similarly reduces the click count by a factor of 0.27, as opposed to the reference category of broad match type. For a unit increase of c_{kt}, the click count decreases by a factor of 0.77, leading us to consider reducing the maximum cost we set. Similarly to the logistic regression model, below a threshold we would not win any bids to show an impression at all, hence we should be careful with the interpretation and further work is required to assess maximum ad cost. The variable *CreativeQuality*,

q_{kt}, shows an high increase of 1.56 for above average scores, a decrease of 0.41 for below average scores, and a large decrease for a no score of 0.11. In agreement with the logistic model, the creative quality score is a good indicator of keyword performance. The categorical variable *HourOfDay*, h_{kt}, shows a negative impact on the expected click count for early morning and late evening, whereas the afternoon times are not significant compared to the reference of morning time. This suggests that outside of working hours, keyword adverts are performing worse, which could be due to less online traffic.

Symbol	Covariate	β estimate	SE	p-value	Count factor
α	Intercept	-0.2353	0.0635	0.0002	0.7903
x_{kt}	ImpressionDelta	0.0058	0.0001	2×10^{-16}	1.0058
m_{kt}	KeywordMatchType - Exact	-1.3000	0.0480	2×10^{-16}	0.2725
c_{kt}	MaxCpc	-0.2612	0.0070	2×10^{-16}	0.7701
q_{kt}	CreativeQuality - Above average	0.4474	0.0546	2×10^{-16}	1.5643
	CreativeQuality - Below average	-0.8914	0.0811	2×10^{-16}	0.4101
	CreativeQuality - No score	-2.2953	0.0990	2×10^{-16}	0.1007
h_{kt}	HourOfDay - 12am-6am	-2.1878	0.1141	2×10^{-16}	0.1122
	HourOfDay - 12pm-6pm	-0.0286	0.0458	0.5322	0.9718
	HourOfDay - 6pm-12am	-0.40490	0.0556	3.2×10^{-13}	0.6670
	Log-likelihood	-6441.489			

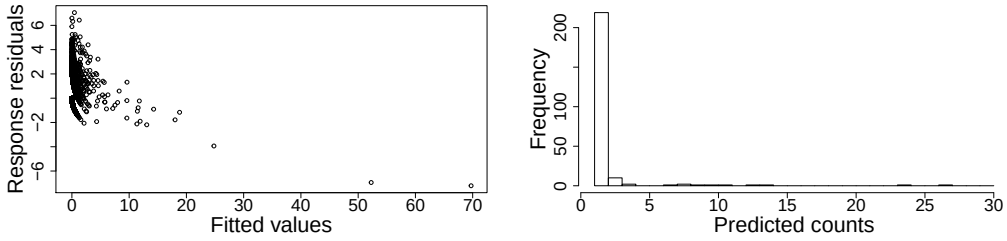
TABLE 4.7: The parameter estimates for the fitted Poisson regression model.

The Poisson model response predictions are calculated using the *predict* function in R, then rounded the nearest integer [45]. Figure 4.2a shows a scatter plot of the response (raw) residuals from the training observations versus the fitted values from the model. A good fitted model should have a spread that increases with fitted values, which on the whole we do observe, however the plot is skewed with extreme values in the bottom right. A histogram of the predicted counts from the fitted Poisson model on the test data set is shown in Figure 4.2b. There are very few counts above zero, as we expect from earlier modelling, but we do have a wide range of predicted counts up to 27 clicks for an observation.

Tables 4.8 and 4.9 display two versions of the same confusion matrix for the click count predictions on the test observations. Our Poisson model, similarly to the logistic model, under-predicts the click counts. As seen in the confusion matrix where the top right of the matrix contains larger values than the bottom left. Here, the over-prediction of zeros could be due to an excess of zeros in the data set, leading us to consider and explore the zero-inflated model structure.

		Reference y_{kt}		
		0	≥ 1	Total
Predicted $\hat{y}_{kt}$	0	60162	330	60492
	≥ 1	130	111	241
	Total	60292	441	60733

TABLE 4.8: The confusion matrix for the click predictions from the Poisson regression on the test set.



(a) A residual plot of the response values. (b) Histogram of predicted counts.

FIGURE 4.2: Plots relating to the fitted Poisson regression model.

		Reference y_{kt}				
		$=0$	$=1$	$=2$	≥ 3	Total
Predicted $\hat{y}_{kt}$	$=0$	60162	277	40	13	60492
	$=1$	129	38	15	19	201
	$=2$	1	1	1	15	18
	≥ 3	0	0	1	21	22
	Total	60292	316	57	68	60733

TABLE 4.9: The confusion matrix for the click predictions from the Poisson regression on the test set.

4.4.3 Zero-inflated Poisson model

The fitted zero-inflated Poisson regression model to our training data set is given in Eq 4.26 and 4.27; note the separation of the logistic (zero-inflated) and Poisson (count) components of the ZIP model [46]. We use only a single covariate of impressions, x_{kt} for a keyword to determine the structural (inflated) zeros. The odds of zero clicks is a function of the number of impressions. For the count model, we wish to model the dependent variable of the number of clicks for a keyword, y_{kt}. We fit the same covariates as in previous sections for the count model component, where we assume all variables are independent in a simple model, and do not fit any interactions between covariates.

$$\text{logit}(p_i) = \alpha + \beta_1 x_{kt} \tag{4.26}$$

$$\log(\lambda_i) = \alpha + \beta_1 x_{kt} + \beta_2 m_{kt} + \beta_3 c_{kt} + \beta_4 q_{kt} + \beta_5 h_{kt} \tag{4.27}$$

For both components of the ZIP model, the coefficients for each of the covariates are provided along with the standard errors, and p-values for the coefficients in Table 4.10. We see that the majority of the covariates, except some categories of q_{kt} and h_{kt}, are significant at the 5% level in this model. The final row provides the log-likelihood, -4705, for the fitted ZIP model.

The zero-inflated aspect of the model shows that the intercept and impression covariate are not significant in predicting the structural zeros. The calculations in Table 4.11 show that all the zero-impressions are treated as zero-inflated clicks, plus 69% of the single-impression data. That does seem relatively sensible in the context. We propose that the model for p_i is more complex than a purely linear relationship on impressions. We found that p_i is estimated to be large for small impressions, but then decays rapidly. This agrees with the estimated coefficients of a big intercept and big negative slope.

On the basis of the estimates for p_i, the zero-inflation is primarily separating

out the no-impression data as a structural zero. A potential reason for the insignificance of the coefficients in that part of the model is that the relationship is not well expressed linearly in terms of impressions, and perhaps a logical factor like "Impressions=0" might give a cleaner fit.

Symbol	Covariate	β estimate	SE	p-value	
	Zero-inflated model				**Odds ratio**
α	Intercept	17.59	51.36	0.732	4.3×10^{7}
x_{kt}	ImpressionDelta	-16.79	51.36	0.744	5×10^{-8}
	Count model				**Count factor**
α	Intercept	-0.2578	0.0653	7.9×10^{-5}	0.7727
x_{kt}	ImpressionDelta	0.0054	0.0001	2×10^{-16}	1.0054
m_{kt}	KeywordMatchType - Exact	-0.7677	0.0491	2×10^{-16}	0.4641
c_{kt}	MaxCpc	-0.0946	0.0073	2×10^{-16}	0.9097
q_{kt}	CreativeQuality - Above average	0.1679	0.0551	0.0023	1.1828
	CreativeQuality - Below average	-0.9372	0.0813	2×10^{-16}	0.3917
	CreativeQuality - No score	-0.0218	0.0946	0.8177	0.9784
h_{kt}	HourOfDay - 12am-6am	-1.3451	0.1149	2×10^{-16}	0.2605
	HourOfDay - 12pm-6pm	0.0012	0.0461	0.9789	1.0012
	HourOfDay - 6pm-12am	-0.0492	0.0561	0.3803	0.9520
	Log-likelihood	-4705			

TABLE 4.10: The parameter estimates for the fitted zero-inflated Poisson model.

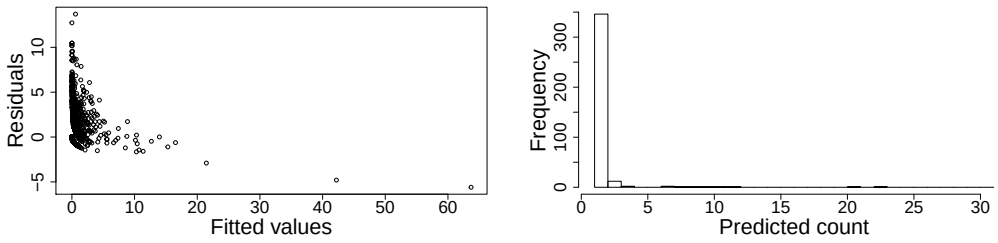
x_{kt}	p_i
0	1
1	0.69
2	1.1×10^{-7}
3	5.8×10^{-15}

TABLE 4.11: A table of output probabilities from the logistic component of the ZIP model.

In the Poisson count model, for every unit count of an impression for a keyword, there is an increase of 1.01, on the expected count rate, which is a small increase but intuitively justified for the count model. The exact match type, reduces the click count by a factor of 0.46, as opposed to the reference category of broad match type. For a unit increase of c_{kt}, the click count decreases by the factor of 0.91, leading us to consider reducing the maximum cost we set. The variable *CreativeQuality*, q_{kt}, shows an increase of 1.18 for above average scores and a decrease of 0.39 for below average scores, whereas the no score category is not significant in the model. The categorical variable *HourOfDay*, h_{kt}, shows a decrease by the factor of 0.26 on clicks for early morning, whereas other times are not significant compared to the reference of morning time.

Figure 4.3a shows a scatter plot of the response residuals from the training data set observations versus the fitted values. A good fitting model should have a spread that increases with fitted values, which on the whole we do observe, however the plot is skewed with extreme values in the bottom right. Figure 4.3b shows a histogram of the predicted counts, rounded to the nearest integer, using the fitted Poisson model on the test data set observations. There are very few counts above zero, as

we expect from earlier modelling, but we do have a wide range of predicted counts up to 117 clicks for an observation. Table 4.12 displays a variation on a confusion matrix for the click count predictions on the test observations. All 57721 structural zeros were predicted a zero count by the ZIP model. Our ZIP model under predicts the counts as reflected that the top right of the matrix contains larger values than the bottom left. Here, we again observe an over-prediction of zeros, even with the zero-inflated model. If we directly compare to Table 4.9, the ZIP appears to present slightly better results with less over-prediction of zeros than the simple Poisson regression model.



(a) A residual plot of the response values.

(b) Histogram of predicted counts.

FIGURE 4.3: Plots relating to the fitted zero-inflated Poisson model.

		Reference y_{kt}				
		$= 0$	$= 1$	$= 2$	≥ 3	Total
Predicted $\hat{y}_{kt}$	$= 0$	60095	233	26	9	60363
	$= 1$	196	81	28	19	324
	$= 2$	1	2	2	17	22
	≥ 3	0	0	1	23	24
	Total	60292	316	57	68	60733

TABLE 4.12: The confusion matrix for the click predictions from the zero-inflated Poisson model on the test set.

4.4.4 Model comparison

To compare the fitted models we require measures to assess the fit of the model to our data. The two criterion we will use are the Akaike Information Criterion (AIC) and the Bayes Information Criterion (BIC), which are defined as:

- AIC $= 2p - 2l$
- BIC $= p\log(n) - 2l$

where p is the number of parameters in the model, l is the likelihood of the data and n is the number of observations in the data. We want to minimise the AIC or BIC values, i.e. we want to maximise the log-likelihood and minimise the number of parameters used in the model. AIC has a mid sized penalty for large values of p. Whereas, BIC has a severe penalty for larger models, when n is large. Table 4.13 displays the AIC and BIC values for our fitted models. While it is generally not meaningful to compare likelihoods from such different and non-nested models, we give the values to illustrate the general results. For both criterion, the logistic model has the lowest AIC value of 8132, followed by the ZIP model with 9434, and the Poisson model with the highest AIC value 12903.

We also provide the proportion of correctly identified zeros (true positives) and non-zeros (true negatives) from our model predictions. All models score highly, with over 99.5%, at identifying zeros correctly, whereas they differ greatly on identifying correct non-zeros. The ZIP model performs stronger than the simple Poisson model, 39% versus 25%, but the logistic model has the best true-negative rate with 55%. We must note that the predictions for the logistic model are determined by the arbitrary threshold that we set to determine a click or not click. Hence the proportions would differ with different choices for the threshold and to note the Poisson model predictions are rounded to the nearest integer.

Model	AIC	BIC	Correct zeros	Correct non-zeros
Logistic	8132.3	8233.39	99.51%	54.65%
Poisson	12902.98	13004.11	99.78%	25.17%
ZIP	9433.82	9555.18	99.67%	39.23%

TABLE 4.13: A table of model comparison measures for logistic, Poisson and ZIP.

4.5 Discussion

We discussed the theory of GLMs, focusing on the family of distributions EDFs that we can use to change the model based on our application. We explored properties of EDFs, in particular the Bernoulli and Poisson distributions, as we use logistic and Poisson regression for our data set. We set out how to inspect the output of GLMs and looked at model diagnostics, i.e. the residuals, and we will explore model selection methods in Chapter 6.

We discussed zero-inflated models and why they would be appropriate in our specific application. We used the structural zeros in the data to aim for a better fitting model. We outlined how to form a zero-inflated Poisson model and the key differences from GLMs.

We applied the GLM theory to the Adwords data set by fitting the models to a binary and count outcome variable of clicks. Overall, we fitted 3 models a logistic

regression, Poisson regression and zero-inflated Poisson model. We use a consistent set of covariates, interpret the coefficient estimates, assess the model fit through residuals and prediction using a confusion matrix.

The models lead to similar conclusions in terms of coefficient interpretation and under prediction of clicks, which is most likely due to the high number of zeros in the model. There seems to be extreme values that provide skewed residual plots and prediction distributions. To improve these models we hoped the zero-inflation component could help, but the model still under predicts, so we feel that improvements could be made in our variable selection.

Chapter 5

Markov models for clickstream analysis

5.1 Chapter overview

This chapter will discuss a set of tools we can use to analyse clickstream data. First, in Section 5.2 we will motivate the discrete-time Markov chain and(describe the key properties. Next, we extend this to a two-step Markov chain and then explore hidden Markov models (HMMs). We pose a set of problems to be solved using HMMs, hence we explore the theory for the forward (5.3.1), Viterbi path (5.3.2) and Viterbi training (5.3.3) algorithms for HMMs. In Section 5.4 we will evaluate the models on a variant of the clickstream data described in Chapter 2.

A single visit to a website (or 'session') can be represented as a sequence of web pages visited within that site. Statistical analysis of this data is a good example of data science, which is particularly useful for web activity analytics, advertising and marketing, and improving profit of e-commerce sites. Applying Markov chain type models to clickstream data is well researched in the literature, where clickstreams can be viewed as browsing page type transitions and visualisation tools exist to display the results [15], [17]. Further, complex modelling of similar clickstream data using hidden Markov models can be found in the literature, where nested and Bayesian HMMs have been investigated to find the probability that a browsing session contains an online purchase [20], [21].

5.2 Markov chains

We can consider the clickstream for a single session as a sequence of categorical variables $(X_1, ..., X_t)$, where X_t is a random observation at time t. The simplest model for a session could then be a Markov chain, where at any time t the user randomly selects a new page, X_{t+1}, and moves to that page according to a probability distribution over the available web pages which depends only on the current page, X_t. We can see an example of this state journey path in Figure 5.1. The Markov chain allows us to summarise all of the movement through the website via the matrix of transition probabilities $P[X_{t+1}, X_t]$, which provides a simple mechanism for prediction as well as an effective way to map the traffic through the site.

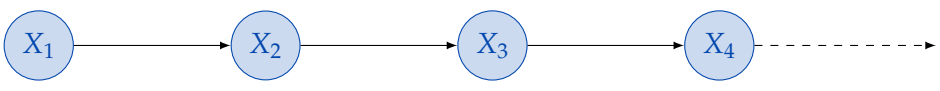


FIGURE 5.1: A Markov chain.

For a random variable $\boldsymbol{X}$, we define a homogeneous Markov chain in discrete-time, where the actual observations we denote as $x_1, ..., x_t$, and the possible set of observations as the set: $\mathcal{X} = \{o_1, ..., o_M\}$ [47]:

$$P(X_{t+1} = x_{t+1} | X_t = x_t, X_{t-1} = x_{t-1}, ..., X_1 = x_1) = P(X_{t+1} = x_{t+1} | X_t = x_t).$$

We specify this by the probability $P(X_{t+1} = o_j | X_t = o_i) = p_{ij}$ by the estimate $\hat{p}_{ij}$ where,

$$\hat{p}_{ij} = \frac{\#\ o_j's \text{ following } o_i's}{\#\ \text{all transitions}}.$$

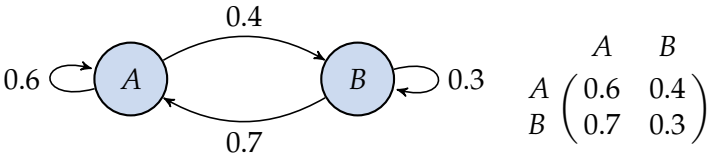
The values of p_{ij} are the transition probabilities for the Markov chain and only depend on the states o_i, o_j and not on the time t or the previous states $o_{i-1}, ..., o_1$. The key components for a Markov chain are summarised with the corresponding notation in Table 5.1. The initial distribution for the Markov chain is determined by

$$\pi(o_j) = P(X_1 = o_j)$$

which we estimate by,

$$\hat{\pi}(o_j) = \frac{\#\ o_j's \text{ at step } 1}{\#\ \text{all transitions at step } 1}.$$

Further properties of a Markov chain can be found in the literature [48]. We display the dynamics of a Markov chain using a graph with nodes representing the categorical states of the Markov chain and edges representing transitions. A simple example can be seen in Figure 5.2a. There are two states $\mathcal{X} = \{A, B\}$ and the transition matrix representing the Markov chain is displayed in Figure 5.2b.



(a) As a graph.

$$\begin{array}{c} \\ A \\ B \end{array} \begin{array}{c} \begin{array}{cc} A & B \end{array} \\ \begin{pmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{pmatrix} \end{array}$$

(b) As a transition matrix.

FIGURE 5.2: Example of a Markov chain.

Name	Symbol
Number of possible states	M
Random observation at time t	X_t
Observed sequence	$x_1, ..., x_t$
The set of possible states	$\mathcal{X} = \{o_1, ..., o_M\}$
Transition probability from state i to j	$\hat{p}_{ij}$
Transition probability from states i, j to k	$\hat{p}_{ijk}$

TABLE 5.1: Key concepts and notation to describe a Markov chain.

However, the main disadvantage and chief limitation of the Markov chain is its lack of memory (the next page depends only on the current one), which makes it computationally quick and easy but makes it impossible to use information previous to the current page. As a result we introduce the two-step Markov chain, where we

add an extra state into the memory of the model. Formally,

$$P(X_{t+1} = x_{t+1}|X_t = x_t, X_{t-1} = x_{t-1}, ..., X_1 = x_1) = P(X_{t+1} = x_{t+1}|X_t = x_t, X_{t-1} = x_{t-1}).$$

We estimate the probability $P(X_{t+1} = o_k|X_t = o_j, X_{t-1} = o_i) = p_{ijk}$ by,

$$\hat{p}_{ijk} = \frac{\#\ o_k's \text{ following } (o_i, o_j) \text{ pairs}}{\#\ \text{all transitions}}.$$

The initial distribution for the two-step chain is the joint distribution,

$$\pi(o_k, o_j) = P(X_2 = o_j, X_1 = o_i),$$

which we estimate by,

$$\hat{\pi}(o_k, o_j) = \frac{\#(o_j, o_k) \text{ pairs at steps 1,2}}{\#\ \text{all steps 1,2 transitions}}.$$

5.3 Hidden Markov models

So far we have considered Markov models where each state corresponds to an observable event, i.e. a page type. The concept of Markov models can be extended to unobserved states that are probabilistic. This forms the basis of a hidden Markov model (HMM) where hidden states are observed through a set of stochastic processes that produce the sequence of observations. This model can be applied to time series data and is used in a wide range of applications including: speech recognition systems, pattern recognition and molecular biology [49], [50], [51]. The motivation for our application is to discover the hidden states in the clickstream data and hope we can interpret them as meta-states or behavioural modes.

We will describe a discrete-space and discrete-time hidden Markov model, as this framework fits the clickstream data, and we know that this application has been studied before [52],[53]. We have a discrete random observed state X_t at time t, and we assume that the observation was generated by some process whose random state Y_t is hidden. A second assumption is that the hidden states follow a Markov process, as in Section 5.2, where Y_{t+1} only depends on the current hidden state Y_t. We will define the main elements of an HMM, which are summarised in Table 5.2:

1. An actual hidden state sequence is denoted by $\mathbf{Y} = (y_1, ..., y_t)$. The set of possible hidden states are given by $\mathcal{Y} = \{q_1, ..., q_N\}$, where the number of possible **hidden states** in the model is denoted by N. Hence, the hidden state at time t is y_t.

2. An actual observation sequence is denoted by $\mathbf{X} = (x_1, ..., x_t)$. The possible individual states are represented by $\mathcal{X} = \{o_1, ..., o_M\}$, where the number of distinct **observational states** is denoted by M. The notation for possible observations follow directly from Markov chain theory in Section 5.2.

3. The **hidden state transition** probability distribution for Y_t to Y_{t+1} is given by

$$a(y_{t+1}|y_t) = P(Y_{t+1} = y_{t+1}|Y_t = y_t),$$

which reduces to transition probabilities between the N hidden states in $A = (a_{ij})$ an $N \times N$ matrix,

$$a_{ij} = P(Y_{t+1} = q_j | Y_t = q_i), \qquad 1 \leq i, j \leq N.$$

4. The **observation probability (emission) distribution** for X_t in hidden state Y_t is given by

$$b(x_t | y_t) = P(X_t = x_t | Y_t = y_t),$$

which reduces to emission probabilities to the observed states from a hidden state in $B = (b_{ik})$ an $N \times M$ matrix,

$$b_{ik} = P(X_t = o_k | Y_t = q_i), \qquad 1 \leq i \leq N, 1 \leq k \leq M.$$

5. The **initial state distribution**, $\pi = (\pi_i)$, is defined as

$$\pi_i = P(Y_1 = q_i), \qquad 1 \leq i \leq N.$$

We can specify the joint distribution of the hidden state and observation sequence as:

$$P(Y_{1:T}, X_{1:T}) = P(Y_1)P(X_1|Y_1)\prod_{t=2}^{T} P(Y_t|Y_{t-1})P(X_t|Y_t) \tag{5.1}$$

where $1 : T$ denotes the sequences of X, Y from time 1 to T. We have dropped the dimensions of N, M and simplified the notation such that $P(X_t = x_t | Y_t = y_t)$ is replaced by $P(X_t|Y_t)$. We can visualise this as a set of dependencies in a network in Figure 5.3. To summarise, we can denote an HMM by $\lambda = (A, B, \pi)$, given the two model parameters N and M.

Name	Symbol
Number of possible hidden states	N
Set of all possible hidden states	$\mathcal{Y} = \{q_1, ..., q_N\}$
Actual hidden state sequence	$y_1, ..., y_t$
Random hidden state at time t	Y_t
State transition matrix	$A = (a_{ij})$
Observation emission matrix	$B = (b_{ik})$
Initial state distribution	$\pi = (\pi_i)$
HMM	$\lambda = (A, B, \pi)$

TABLE 5.2: A table to describe the elements of a HMM.

Toy example

We present a simple example of an HMM; suppose we wish to determine the daily temperature based on weather patterns. Simplifying the problem, we consider only 2 possible temperatures 'hot' or 'cold' (H and C respectively) and define $N = 2$ and $\mathcal{Y} = \{H, C\}$. We do not have direct knowledge or measurements of the daily temperature, hence we will refer to them as 'hidden' (our $\mathcal{Y}$ states), however we can describe the relationship by quantifying the chance of transitioning between the two states. By treating the transitions as a first order Markov chain, where the next state only depends on the previous state we summarise the relationship with the

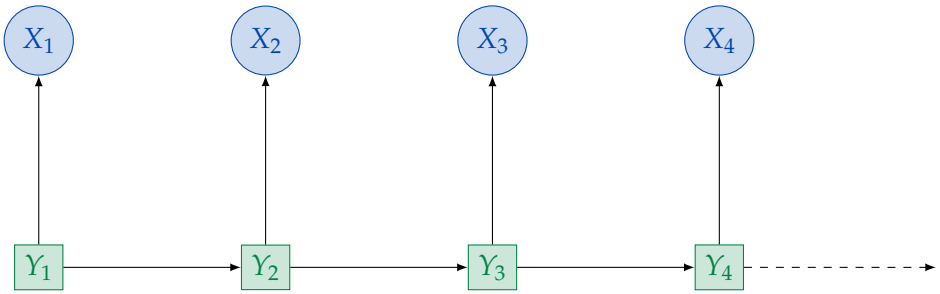


FIGURE 5.3: A network specifying the conditional independence relations for a HMM.

following matrix:

$$A = \begin{array}{c} \\ H \\ C \end{array} \begin{array}{c} \begin{array}{cc} H & C \end{array} \\ \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix} \end{array}$$

We know that the weather pattern correlates with the temperature, and we have observable evidence (our $\mathcal{X}$ states) of 3 distinct types: R (rainy), W (windy) and S (sunny). Hence, we have $M = 3$ and $\mathcal{X} = \{R, W, S\}$. We provide the probabilistic relationship between temperature and the weather:

$$B = \begin{array}{c} \\ H \\ C \end{array} \begin{array}{c} \begin{array}{ccc} R & W & S \end{array} \\ \begin{pmatrix} 0.1 & 0.4 & 0.5 \\ 0.7 & 0.2 & 0.1 \end{pmatrix} \end{array}$$

Further, suppose there is a distribution for the first hidden state:

$$\pi = \begin{array}{c} \begin{array}{cc} H & C \end{array} \\ \begin{pmatrix} 0.6 & 0.4 \end{pmatrix} \end{array}$$

We now have all the elements to define a HMM. We use this example to motivate 3 key problems, which must be solved to apply an HMM to real applications.

1. For the HMM defined above, can we 'score' a sequence of observed weather patterns? Given an HMM, λ, and a sequence of observations $\mathbf{X}$, what is the likelihood of the data given the model $P(\mathbf{X}|\lambda)$.

2. What is the most likely temperature sequence, given a set of observed weather patterns? Given an HMM, λ, and a set of observations $\mathbf{X}$, find the optimal hidden state sequence $\mathbf{Y}$.

3. If we have a history of weather pattern observations and know the number of temperature possibilities, what is the best fitting HMM to describe the physical process? Given a set of observations $\mathbf{X}$, the dimensions N and M, find the model λ that maximises the probability of $\mathbf{X}$.

We will investigate each problem in the following sections and refer back to this simple HMM to exemplify the solutions.

5.3.1 Forward algorithm (Problem 1)

We start with an example. Given the simple weather-temperature HMM, what is the probability of the weather sequence SSR? For a simple Markov chain, we would

simply multiply each transition probability together. The difference for HMMs is that we do not know what the hidden state sequence is. Let's start by assuming we know the hidden state sequence, i.e. HHC. We can now easily compute the likelihood for the sequence, $\mathbf{X} = (x_1, x_2, x_3) = (S, S, R)$, given the Markov assumptions, as

$$P(SSR|HHC) = P(S|H) \times P(S|H) \times P(R|C).$$

However, we do not know the sequence of hidden temperature states, hence we will need to sum over all possible hidden states. The joint probability will include the weighting of the probability of the hidden state sequence. For our particular case we obtain:

$$\begin{aligned} P(SSR, HHC) &= P(SSR|HHC) \times P(HHC) \\ &= P(S|H) \times P(S|H) \times P(R|C) \\ &\quad \times P(H) \times P(H|H) \times P(C|H). \end{aligned}$$

To compute the full likelihood for an observation sequence, we would have to apply the partition theorem over the possible hidden sequences:

$$\begin{aligned} P(SSR) = &P(SSR|HHH)P(HHH) \\ &+ P(SSR|CHH)P(CHH) \\ &+ P(SSR|CCH)P(CCH) + \dots \end{aligned}$$

For this example, we would need to sum over eight possible hidden sequence states, which is tractable here only because T, N, M are small. More formally, we have a HMM, λ, and an observation sequence $\mathbf{X} = (x_1, ..., x_T)$; we want to find $P(\mathbf{X}|\lambda)$. Let $\mathbf{Y} = (y_1, ..., y_T)$ be a sequence of hidden states. Then by the definition of B and by Eq. 5.1 we have

$$P(\mathbf{X}|\mathbf{Y}, \lambda) = b(x_1|y_1)b(x_2|y_2) \dots b(x_T|y_T)$$

using the definition of the initial state π and A we know that

$$P(\mathbf{Y}|\lambda) = \pi(y_1)a(y_2|y_1)a(y_3|y_2) \dots a(y_T|y_{T-1}).$$

With the example above, we have motivated that,

$$\begin{aligned} P(\mathbf{X}|\lambda) &= \sum_{\mathbf{Y}} P(\mathbf{X}|\mathbf{Y}, \lambda) \times P(\mathbf{Y}|\lambda) \\ &= \sum \pi(y_1)b(x_1|y_1) + a(y_2|y_1)b(x_2|y_2) + \dots + a(y_T|y_{T-1})b(x_T|y_T). \end{aligned}$$

In generality, for an HMM with N possible hidden states and an observation sequence of length T, there are N^T possible hidden sequences. This can be a large number for even modest values of T and direct computation of this value rapidly becomes infeasible.

To find $P(\mathbf{X}|\lambda)$, we use the **forward algorithm**, for $t = 1, ..., T$ and $i = 1, ..., N$ [52]. We define,

$$\alpha(x_t, y_i) = P(X_1 = x_1, ..., X_t = x_t, Y_t = y_i|\lambda). \tag{5.2}$$

Then $\alpha(x_t, y_i)$ is the likelihood of the observation sequence up to x_t, where the hidden state is y_i at time t. We can compute this value recursively using the following algorithm:

1. Let $\alpha(x_1, y_j) = \pi(y_j)b(x_1|y_j)$ for $1 \leq j \leq N$
2. For $1 \leq t \leq T$ and $1 \leq j \leq N$, compute iteratively

$$\alpha(x_t, y_j) = \left[\sum_{i=1}^{N} \alpha(x_{t-1}, y_i)a(y_j|y_i)\right] b(x_t|y_j)$$

3. From 5.2 we know that

$$P(\mathbf{X}|\lambda) = \sum_{i=1}^{N} \alpha(x_T, y_i)$$

We set out the pseudocode for the forward algorithm in Algorithm 1. We store the values for step (2) in a matrix form, such that we can then sum over the hidden states. We will use this later to determine the likelihood of a sequence, given an HMM model.

Algorithm 1 Forward algorithm

```
function FORWARD(N, T)
Set Forward matrix of size (N, T)
    for n ∈ {1, ..., N} do
        Forward[n, 1] = π_n × b_{n,1}
    for t ∈ {2, ..., T} do
        for n ∈ {1, ..., N} do
            Forward[n, t] = Σ_{n'=1}^{N} Forward[n', t − 1] × a_{n,n'} × b_{n,t}
    ForwardProb = Σ_{n=1}^{N} Forward[n, T]
    return ForwardProb
```

In our example we introduced in Section 5.3, let's use the forward algorithm to find the likelihood for our observation sequence, i.e. what is the probability of $P(SSR|\lambda)$?

1. We first find the likelihood of observing the first state, S, over all possible hidden states, H and C:

$$\alpha(x_1, q_1) = \alpha(S, H) = \pi(H)b(S|H) = 0.6 \times 0.5 = 0.3$$
$$\alpha(x_1, q_2) = \alpha(S, C) = \pi(C)b(S|C) = 0.4 \times 0.1 = 0.04$$

2. We now recursively calculate the likelihood for the sequence at steps 2 and 3, over the possible hidden states:

$$\begin{aligned}
\alpha(x_2, q_1) = \alpha(S, H) &= [\alpha(S,H)a(H|H) + \alpha(S,C)a(H|C)]\, b(S|H) \\
&= (0.3 \times 0.7 + 0.04 \times 0.4) \times 0.5 = 0.113 \\
\alpha(x_2, q_2) = \alpha(S, C) &= [\alpha(S,H)a(C|H) + \alpha(S,C)a(C|C)]\, b(S|C) \\
&= (0.3 \times 0.3 + 0.04 \times 0.6) \times 0.1 = 0.0114 \\
\alpha(x_3, q_1) = \alpha(R, H) &= [\alpha(S,H)a(H|H) + \alpha(S,C)a(H|C)]\, b(R|H) \\
&= (0.113 \times 0.7 + 0.0114 \times 0.4) \times 0.1 = 0.008366 \\
\alpha(x_3, q_2) = \alpha(R, C) &= [\alpha(S,H)a(C|H) + \alpha(S,C)a(C|C)]\, b(R|C) \\
&= (0.113 \times 0.3 + 0.0114 \times 0.6) \times 0.7 = 0.028518
\end{aligned}$$

3. Finding the total likelihood of the sequence given the HMM, we sum the probabilities at the final step of the forward algorithm over the possible hidden states:

$$\begin{aligned}
P(SSR|\lambda) &= \sum_{i=1}^{2} \alpha(x_3, q_i) \\
&= \alpha(x_3, q_1) + \alpha(x_3, q_2) \\
&= \alpha(R, H) + \alpha(R, C) \\
&= 0.008366 + 0.028518 = 0.036884
\end{aligned}$$

So the likelihood of the observation, $\mathbf{X} = (S, S, R)$, given the HMM is $P(SSR|\lambda) = 0.0369$ (4 d.p.).

5.3.2 Viterbi path algorithm (Problem 2)

Returning to the temperature-weather HMM, we pose the question, given a sequence of weather observations SSR, what is the most likely hidden temperature sequence? One thought process to solve this might be to compute the likelihood for all possible hidden temperature state sequences, given the weather observations. However, this would be too computationally expensive as the number of possibilities increases exponentially. Alternatively, a dynamic programming (i.e. Viterbi) approach attempts to find the highest overall scoring path. We explore the Viterbi algorithm [54], [55].

To find the most likely hidden state sequence, $\mathbf{Y} = (y_1, ..., y_T)$, for a given observation sequence $\mathbf{X} = (x_1, ..., x_T)$, we define a probability value, δ, for $t < T$. The value δ specifies the maximum probability of observing the state, x_t over all possible hidden state sequences at time t:

$$\begin{aligned}
\delta(x_t, y_j) &= \max_{Y} P(Y_1 = y_1, ..., Y_t = y_j, X_1 = x_1, ..., X_t = x_t|\lambda) \\
&= \max_{i} \delta(x_{t-1}, y_i) a(y_j|y_i) b(x_t|y_j).
\end{aligned}$$

The Viterbi algorithm computes these values recursively such that $\delta(x_t, y_j)$ is the most likely hidden state sequence for the observed sequence at time t, ending at hidden state y_j. We then find the most probable overall path, by taking the maximum over all possible previous steps of the algorithm.

There are different interpretations of 'most likely', one idea could be to maximise the expected number of correct hidden states. These two approaches will not necessarily agree on the 'most likely' hidden state sequence, given an observation sequence and HMM.

Formally, we define the **Viterbi algorithm** as follows:

1. For $1 \leq j \leq N$, find the starting emission probabilities over all possible hidden states:

$$\delta(x_1, y_j) = \pi(y_j)b(x_1|y_j)$$
$$\psi(x_1, y_j) = 0$$

2. For $1 \leq j \leq N, 2 \leq t \leq T$, recursively calculate the maximum value of the likelihood of observing the state, x_t, over all hidden possible hidden states at time t:

$$\delta(x_t, y_j) = \max_i \delta(x_{t-1}, y_i)a(y_j|y_i)b(x_t|y_j)$$
$$\psi(x_t, y_j) = \arg\max_i \delta(x_{t-1}, y_i)a(y_j|y_i)b(x_t|y_j)$$

3. Find the final probabilities for the path,

$$p^* = \max_i \delta(x_T, y_i)$$
$$q_T^* = \arg\max_i \delta(x_T, y_i)$$

4. Backtrace for $t = T-1, T-2, \ldots, 1$, to find the most likely sequence of hidden states, maximising the probability at each step:

$$q_t^* = \psi(x_{t+1}, y_{q_{t+1}^*}),$$

to create the full hidden state sequence,

$$\mathbf{Q} = (q_1^*, ..., q_T^*).$$

Algorithm 2 shows pseudocode for the Viterbi algorithm. Note that the Viterbi algorithm is very similar to the forward algorithm, except that it takes the maximum over the previous path probabilities, whereas the forward algorithm takes the summation. The extra component that the forward algorithm doesn't include is the 'lookback'. The reason for this is, while the forward algorithm needs to produce only the observation sequence likelihood, the Viterbi path algorithm must produce a probability and also the most likely hidden state sequence. We compute this best state sequence by keeping track of the path of hidden states that led to each state, and then at the end backtracing the best path to the beginning.

For the toy example in Section 5.3, let's use the Viterbi algorithm to find the 'most likely' hidden state sequence for the observed sequence, i.e. what is the Viterbi path of $\boldsymbol{X} = (S, S, R)$?

1. We calculate the probabilities that state S was emitted by states H and C,

$$\delta(x_1, q_1) = \delta(S, H) = \pi(H)b(S|H) = 0.6 \times 0.5 = 0.3$$
$$\delta(x_1, q_2) = \delta(S, C) = \pi(C)b(S|C) = 0.4 \times 0.1 = 0.04$$

Algorithm 2 Viterbi algorithm

```
function VITERBI(N, T)
Set Viterbi matrix of size (N, T)
    for n ∈ {1, ..., N} do
        Viterbi[n, 1] = π_n × b_{1,n}
        Lookback[n, 1] = 0
    for t ∈ {2, ..., T} do
        for n ∈ {1, ..., N} do
            Viterbi[n, t] = max_{n'} Viterbi[n', t − 1] × a_{n,n'} × b_{n,t}
            Lookback[n, t] = arg max_{n'} Viterbi[n', t − 1] × a_{n,n'} × b_{n,t}
    ViterbiPathProb = max_n Viterbi[n, T]
    ViterbiPathEnd = arg max_n Viterbi[n, T]
    ViterbiPath = Backtrace from ViterbiPathEnd using Lookback
    return ViterbiPath, ViterbiPathProb
```

2. We iteratively compute the δ probabilities as follows:

$$\begin{aligned}
\delta(x_2, q_1) = \delta(S, H) &= \max\{\delta(S,H)a(H|H), \delta(S,C)a(H|C)\} \times b(S|H) \\
&= \max(0.3 \times 0.7, 0.04 \times 0.4) \times 0.5 = 0.105 \\
\delta(x_2, q_2) = \delta(S, C) &= \max\{\delta(S,H)a(C|H), \delta(S,C)a(C|C)\} \times b(S|C) \\
&= \max(0.3 \times 0.3, 0.04 \times 0.6) \times 0.1 = 0.009 \\
\delta(x_3, q_1) = \delta(R, H) &= \max\{\delta(S,H)a(H|H), \delta(S,C)a(H|C)\} \times b(R|H) \\
&= \max(0.105 \times 0.7, 0.009 \times 0.4) \times 0.1 = 0.00735 \\
\delta(x_3, q_2) = \delta(R, C) &= \max\{\delta(S,H)a(C|H), \delta(S,C)a(C|C)\} \times b(R|C) \\
&= \max(0.105 \times 0.3, 0.009 \times 0.6) \times 0.7 = 0.02205
\end{aligned}$$

3. The Viterbi matrix can be constructed using the δ values we have calculated at each step of the observed sequence over all possible hidden states:

	S	S	R
H	0.3	0.105	0.00735
C	0.04	0.009	0.02205

We find the path which corresponds to the highest probability, which ends in hidden state C with the value $0.02205 > 0.00735$, which is the probability of the most likely path. We then backtrace, from right to left in the table above, finding the highest probability in each column, which in this case is HHC.

5.3.3 Viterbi training (Problem 3)

Given a history of weather pattern observations and the number of possible temperature states, how do we estimate the parameters of an HMM with the 'best fit' to our data. More formally, given a set of observations $\mathbf{X}$, the dimensions N and M, find the model λ that maximises the probability of $\mathbf{X}$. The training method does not know the hidden state sequence that corresponds to the observations, and there is no closed form expression to determine them. Therefore it is a tricky problem to estimate the matrices A and B, unlike with a simple Markov chain.

The Baum-Welch algorithm is a special case of the EM algorithm [56]. It considers paths for the training sequences using the current values of A and B. It can be shown

that the log likelihood of the model will increase through iterations and converge to a local maximum [57]. In practice, we note that the Baum-Welch algorithm failed on the larger data sizes we used to train an HMM.

Alternatively the **Viterbi training** method is proposed as an approach to speed up the learning process [54]. The basic principle is to replace the computationally costly expectation E-step of the EM algorithm by an appropriate maximisation step with fewer and simple computations. Hence, the maximum of the likelihood can be approximated by maximising the probability of the best fitting hidden state sequence for the training observations, given the parameters of the HMM. The most likely path for the training sequence is derived using the Viterbi path algorithm (Section 5.3.2), which estimates the parameters for the HMM.

1. Initialisation: Randomly choose the initial parameters

$$\pi^{(1)}, A^{(1)}, B^{(1)}$$

2. Iteration: We derive the most likely path of hidden states using Viterbi algorithm, given the model λ and training observations $\mathbf{X} = (x_1, ..., x_T)$:

$$\mathbf{Q} = (q_1^*, ..., q_T^*)$$

   Calculate $a(q_j^*|q_i^*)$ and $b(x_t|q_i^*)$, using proportions from the observations and hidden state sequences. Update to find:

$$\pi^{(2)}, A^{(2)}, B^{(2)}$$

   Repeat this step.

3. Termination: Stop when the model parameters converge, i.e. the values $|A^{(t)} - A^{(t-1)}| < \epsilon$, $|B^{(t)} - B^{(t-1)}| < \epsilon$, where ϵ is a value close to zero.

5.4 Application to Clickstream data set

We will apply the techniques in this chapter to a subset of the clickstream data. We focus on page visits on a single e-commerce site. The dataset we will use is from 20th April 2019. There are 12492 clickstream sequences, where each clickstream is at least length 3, up to a maximum 50 page visits. We transform the data into sequences of clickstreams corresponding to each profile. We encode the clickstream path such that each page visit is represented by a category. The 14 categories represent the context of the page visit, i.e. the **page type**, and the definitions of the page types are displayed in Table 5.3.

We can compactly represent paths using the first initial of our categories as an abbreviation. For example, the string "HPDE" would denote a user who starts at a home page, moves to a overview list for a type of product, and concludes their session at a product page after considering an individual item.

To illustrate our data we list the browsing journeys of four selected users in in Table 5.4. We introduce artificial states to represent time-based features in the data. We add an artificial *Exit* state (E) which tells us there is no further page visit data for the user. The *Return* state (R) represents when a user has returned to the site in a later browsing session. The site has a popular range of products which are listed in *Product* (P) and displayed individually in more detail in page type *Product Detail* (D).

Page Type	Symbol	Definition
Article List	A	A list of articles titles with short descriptions and images
Shopping Basket	B	A list of the items that a visitor has added to be purchased at a later time
PrimaryTextContent (Article)	C	A piece of content writing
Product Detail (Single Product)	D	A detailed description of a product available to purchase
Exit	E	An artificial state to identity when a user has left without returning to the site
Home	H	The main entry point to the site
Site Page	I	Pages to help guide the visitor to information about the company
Login	L	A page to enter credentials to access the visitors profile
Product List	P	A list of names, images and brief descriptions of products
Return	R	An artificial state to determine when a visitor has left and returned to the site
Search Results	S	The resulting page after a search term has been queried
Profile	U	Any page that is specialised for the individual visitor with personal information or settings for the site
Checkout	X	A page to complete the payment for items in the basket
Conversion Success	Y	A page that confirms success after completing the checkout successfully

TABLE 5.3: A table of page types and their definitions.

There is a small blog type section with articles on the products they offer, identifiable with the page types: *Article List* (A) and *PrimaryTextContent* (C).

Notice the first three users do not make a purchase in Figure 5.4, while the final user reaches the *Checkout* page (X). A focus of this application is the conversion or purchase event on the e-commerce site. There are multiple steps in the purchasing process, firstly the user is viewing the *Shopping Basket* (B). Then the user will checkout and enter the checkout page (X), with areas such as the delivery and payment details. Finally, a successful purchase or conversion success (Y) is the main target in the conversion. This is so we can compare the browsing behaviour between successful orders and checkout abandoners.

We expect that most of the time users will click on hypertext links on the web page, however some pages may be selected through the browser's interface, i.e. using back and forward buttons, bookmarks, or history. We first clean the data, removing multiple instances of same url, i.e. when the page has been refreshed, as there is no real change of state. It is also possible that two pages that are not linked

ProfileID	Clickstream
1	PDPDDE
2	DPPDDDDDDPDPPE
3	HPPDPDPDPDDPDBBE
4	HPDBXE

TABLE 5.4: A sample of the sequential clickstream data for the Markov models.

together will appear as consecutive viewings in our data set because a user could have multiple windows (web browsers) open in different areas of the web site. As we see a continuous stream of page views, we cannot determine if there are multiple windows open. This is important when analysing page type transitions, as it is possible that the user could navigate to any page type through the browser interface or multiple windows.

The occurrences of each state in the data set can be seen in the final column of Table 5.5, and an added grouping for page types that are closely related. As the site is focused on selling their products, we see that the product pages are the most popular. The range of starting states can be seen in Table 5.5, and we would expect a high number of journeys to begin on the *Home* page (H). However, this is not the case as the data tell us that over 85% of clickstreams begin entering directly to product type pages. A possible reason for this is direct links from advertisements, targeted email campaigns or from other price comparison sites. We note that all the final states are the *Exit* state (E) which is the absorbing state.

Grouping	State	Starting Probability	End Probability	Number of occurrences
Artificial	E	0	1	4749
	R	0	0	998
Content	A	0.0010	0	127
	C	0.0135	0	557
Home	H	0.0913	0	862
Products	D	0.5233	0	9800
	P	0.3317	0	6323
	S	0	0	66
Profile	L	0	0	65
	U	0.0120	0	572
Purchasing	B	0.0045	0	830
	X	0.0015	0	634
	Y	0.0040	0	235
Site Information	I	0.0171	0	620

TABLE 5.5: A table to display the clickstream sequence state probabilities and occurrences in the data set.

5.4.1 First order Markov chain

We fit the clickstream sequence data to a first order Markov chain, where each state in the network represents a page type from Table 5.3. We use the R package, *clickstreams*, to manipulate the sequence data and calculate the transition matrix with probabilities $\hat{p}_{ij}$ [58].

Figure 5.4a is a heat map to display the transition matrix. Red regions of the heat map indicate higher transition probabilities, whereas yellow areas represent regions of low probabilities. The transition probabilities are the conditional probabilities, i.e. $P(\text{To}|\text{From})$. Figure 5.4b is a network graph visualisation of the same transition matrix in Figure 5.4a, which is generated using the package *igraph* [59]. The colours in the network represent the groupings of the page types in the first column of Table 5.5. Using the transition matrix, the opacity of the lines are proportional to the value of the transition probability between the two states in the direction associated with the arrowhead. The node sizes are proportional to the number of occurrences of each of the state in the data from Table 5.5.

In e-commerce, shopping online tends to follow a pattern where the customer moves from a list of products to individual product views, P to D, and we see a high transition probability of 0.51 as shown in Figure 5.4. Users who use the *Login* page (L) are likely to enter the *profile* pages (U) once they have successfully logged in, and we see a high probability for L to U of 0.44. Note that customers with a profile on this website are also likely to stay around the profile pages (U to U has probability 0.5), which contains pages concerning past orders and delivery details.

From Figure 5.4, we can see that by focusing on the purchasing states X and Y, we know from the website structure that you can access the *Checkout* (X) state from any other page type, due to a basket that acts as a pop up. However, you can only access the payment success from the checkout page, i.e. Y from X. The transition from B to X is high at 0.44, which is the shopping basket to checkout page. Another high transition probability of 0.56 is the successful conversion state, Y, to E, when a payment is complete and the user has left the site.

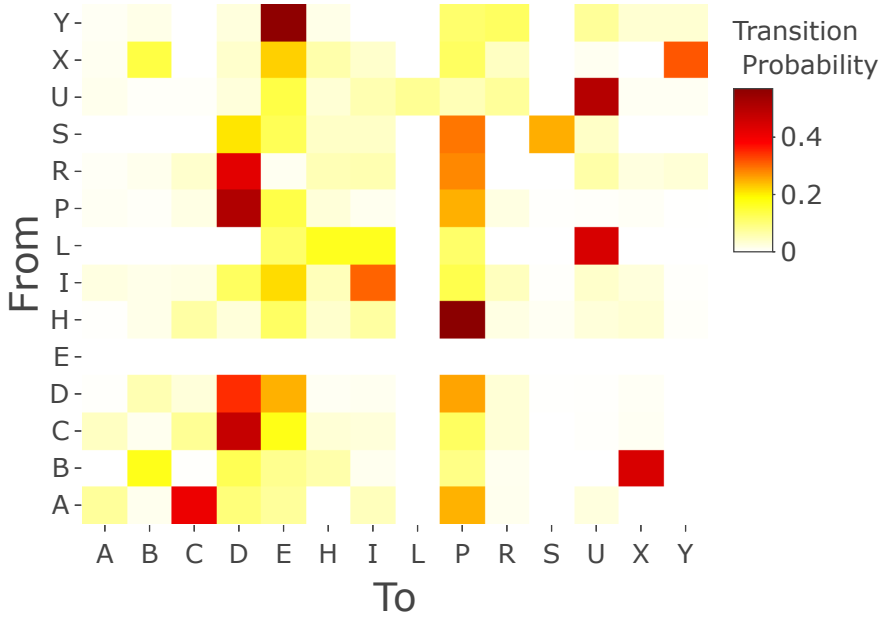
Returning users appear to follow the same pattern as first time visitors, which we can see by comparing the transitions from the *Return* state R to other states, against the starting state probabilities. A transition probability of 0.03 shows that some returning visitors move directly to the Y state, this could be explained by possibly browsers left open from previous purchases.

The *Search results* pages, S, appear to be an infrequently used part of the site, with the most likely transition of 0.25 to remain on the search page, possibly identifying an issue where users cannot find the item they are searching for using keyword search terms. We see a high probability of A to C (0.41), from a list of blogs and articles to a single article page. A further observation is transition from article page C to single product page D is 0.47, where a user reading an article has led to viewing a product. The website may want to know this transition probability, and use it as a measure of how successful the article is in promoting a specific product.

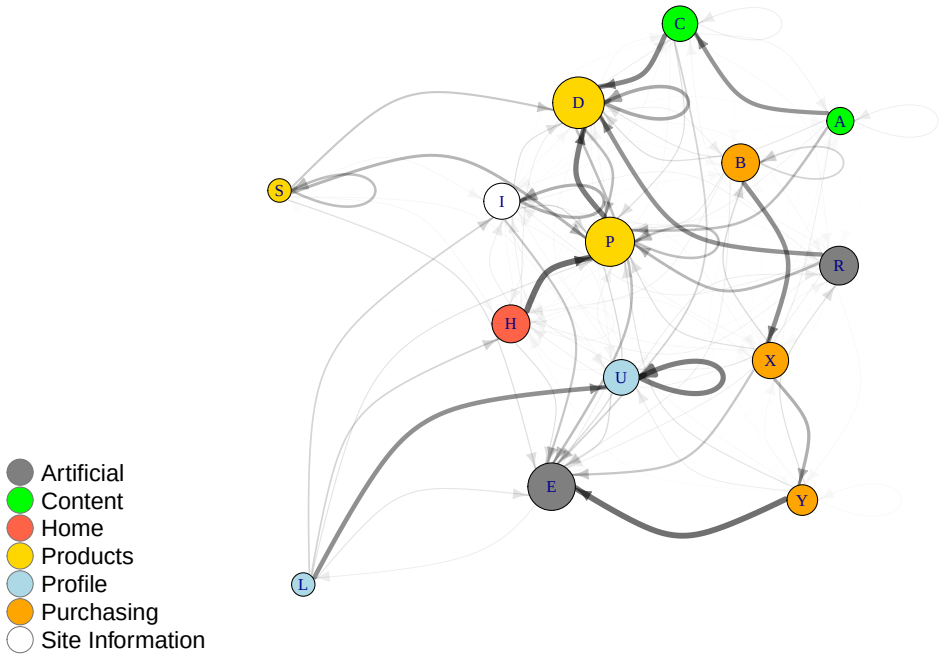
5.4.2 Two-step Markov chain

We aim to uncover some of unseen patterns with the extra memory state in the two-step Markov chain, in particular around the purchasing states. We fit the clickstream data to a two-step Markov chain, by finding all possible two-state memory values and the proportion of transitions to other states. As we are adding more history to the model, the dimensions of the transition matrix increase. We can see this clearly in Figure 5.5a, which is a complex heat map in comparison to the first order, where the number of rows has increased from the number of states S to S^2.

We transform the matrix of the joint probabilities, $P[A, B, C]$, where A is the next state; to conditional probabilities, $P[A|B, C]$, on the two previously observed states: B, C. This is now comparable to the first order Markov chain transition matrix and row stochastic. The darker red areas in Figure 5.5a indicate high probabilities,



(a) A heatmap visualisation of the first order Markov model.



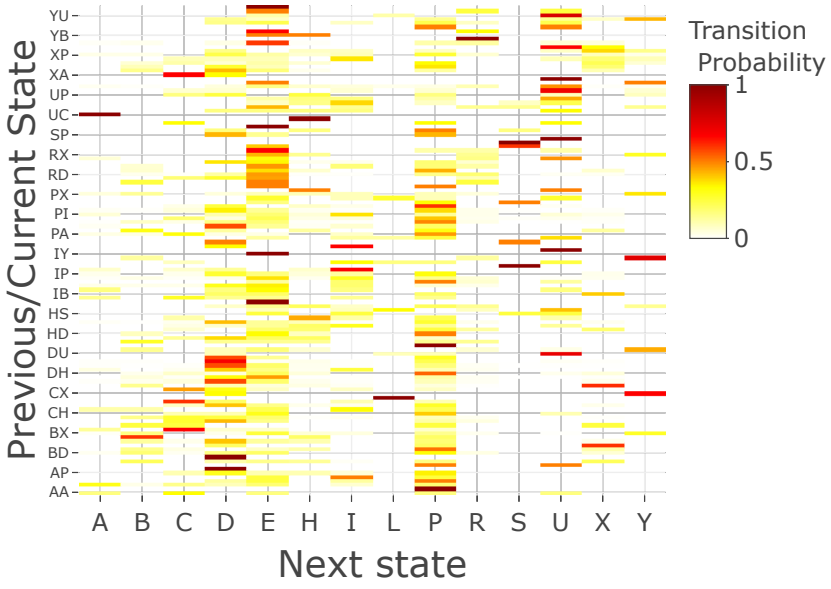
(b) A network visualisation of the first order Markov model.

FIGURE 5.4: Visualisations corresponding to the first order Markov model fitted to our sample data.

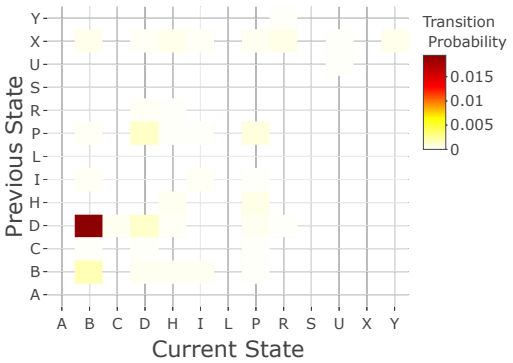
whereas lighter yellow areas represent low transition probabilities. These plots are useful for an overview of data set, but are hard to navigate, hence we will focus on a subset of this matrix and condition on the target purchasing states of X and Y.

From Figure 5.5b, it is clear that transitions from a range of page types to the checkout state (X) are seen in the data. This is most likely to be a result of the basket pop up on the site. The natural purchasing journey begins at a product, which the user then decides to buy, adds the item to the shopping basket, and then reaches the checkout to make the purchase. This is highlighted with the most common two-state transition is DB to X with joint probability of 0.02. The checkout state, followed by another page type and returning to the checkout can be seen in Figure 5.5b. This could be explained by possibly going back a page and then forward again for a last minute check on the product or another page.

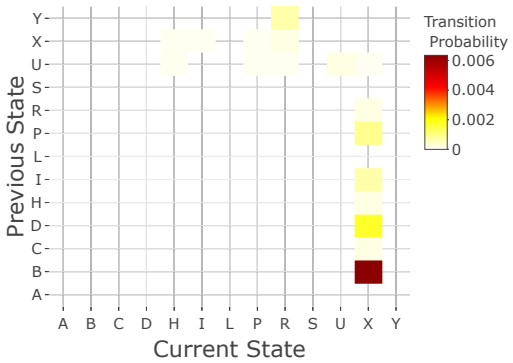
Switching the focus to conditioning on the target state Y, a successful conversion, in Figure 5.5c we see a much clearer structure. We note that B to X to Y is the most likely transition at (0.006), and the most sensible steps determined by the web site structure and user experience. Other transitions are likely to be a result of unclean data, where the browsing sessions have overlapped due to multiple tabs or the data has missing values.



(a) The result applied to the clickstream data.



(b) Transition to X state.



(c) Transition to Y state.

FIGURE 5.5: Heat map visualisations of the two-step Markov models.

5.4.3 Hidden Markov model

In the previous section we have analysed the observed clickstream sequence states using a simple Markov chain approach, now we will extend this to hidden Markov models (HMMs). The motivation to do this is to find a more sophisticated approach where we now have an additional "hidden" variable, which we wish to try and interpret as the background mode (or meta-state) of the user's activity (e.g. "browsing", "updating preferences", "shopping", "just here to pay my bills"). While we can use the simple Markov model to find likely paths through a website, when browsing to buy a product online a purely random behaviour does not seem appropriate. Certain paths could be attributed to a 'purchasing' behaviour, and other characteristic behaviours could be identified. We continue to assess multiple sessions for the same profile and if returning visitors' behaviour is modelled differently using the hidden states. In an HMM, it is this background "hidden state" which evolves like a Markov chain, and now the types of pages visited depend only on the value of this background state (e.g. visiting the Basket will be more likely under a "shopping" state than a "browsing" state).

Our data consists of approximately 25000 page type observations across approximately 5000 profile browsing sequences. We will use the HMM package in R for the initialisation of the HMMs, and we generate random uniformly distributed values, $\mathbb{U}(0,1)$, transformed to be row stochastic for the initial distribution, transition and emission matrices [60]. Specifically, from the package we use the *viterbiTraining* algorithm to train the model, as explained in Section 5.3.3. Further, we must explicitly set the highly influential number of hidden states before we train the HMM.

We do not know the exact number of hidden states that we are searching to find in the data. We have 13 distinct observable states, thus a sensible maximum would be 13, as more would give rise to degenerate solutions. Our desire is to obtain a range of hidden states that could be interpreted as the meta-state of the browsing journey. Some real examples of these states could be: purchasing, product browsing, updating profile preferences, and a general online browsing behaviour that contains all observed states that could not be defined well in a more specific hidden state. In this section, we will focus on an HMM with 5 hidden states. In the next chapter we will explore models using a range of numbers of hidden states.

We display the initial, transition and emission matrices in Tables 5.6 for the 5-state HMM. We will explore these numerical values, to see if the HMM exhibits any interesting features, in particular relating to the purchasing observable states.

The prior distribution of the hidden states in Table 5.6a is set in the training and we make it uniform, as we don't have have any prior information to do otherwise. The hidden states transition matrix in Table 5.6b describes the Markov chain process between the hidden states. The largest transition probabilities lie on the diagonal, meaning most likely transition is to remain in that state. Table 5.6c shows the emission probability matrix which is visualised as a heat map in Figure 5.6. If the observed states for a given hidden state have a non-zero probability, then they can occur in that state. We can use these to infer an interpretation of the hidden states.

Focusing on the purchasing state, state D contains non-zero probabilities for both the purchasing X and Y observed states, and state E contains the checkout state (X). As a result, we could interpret that state D could be the 'purchasing' state, and state E as the 'pre-purchasing' or 'abandoners' meta-state, where the majority of the purchasing journey (up to X) can be observed except the conversion confirmation (Y). The hidden state with the fewest observable states is C, where the profile states L and U are prominent, leading us to interpret this as the 'profile preferences' viewing

Hidden States	Priors
A	0.2
B	0.2
C	0.2
D	0.2
E	0.2

(A) The prior probabilities of the hidden states for the 5-state HMM.

	To				
From	**A**	**B**	**C**	**D**	**E**
A	0.7412	0.1765	0.0824	0	0
B	0	0.9685	0	0.0274	0.0040
C	0.2692	0	0.7308	0	0
D	0.0057	0.3541	0	0.6402	0
E	0.1300	0	0	0.0500	0.8200

(B) The transition matrix for the hidden states in the 5 state HMM.

	Hidden states				
Observed states	**A**	**B**	**C**	**D**	**E**
A	0.0941	0.0034	0	0	0
B	0	0.0040	0.0385	0.2861	0.27
C	0	0.0214	0	0	0.04
D	0	0.4286	0	0	0.19
E	0.1176	0.2103	0	0.0142	0
H	0.1294	0.0274	0.0385	0.0028	0.23
I	0.5412	0.0112	0	0.0057	0
L	0	0	0.1154	0.0170	0
P	0.04714	0.2512	0	0.0085	0.15
R	0.0706	0.0389	0	0.0793	0
S	0	0.0036	0	0	0
U	0	0	0.8077	0.1926	0
X	0	0	0	0.2946	0.12
Y	0	0	0	0.0992	0

(C) The emission matrix for the 5 state HMM.

TABLE 5.6: The numerical values for the 5 state HMM.

state. The most general state is B, which absorbs the most amount of page types without the conversion target states, so we could say this is the general 'browsing/shopping' behaviour state. It is worth noting that from Table 5.6b only B and E states can transition into the purchasing state D, however the probabilities are small with 0.0274 and 0.004 respectively.

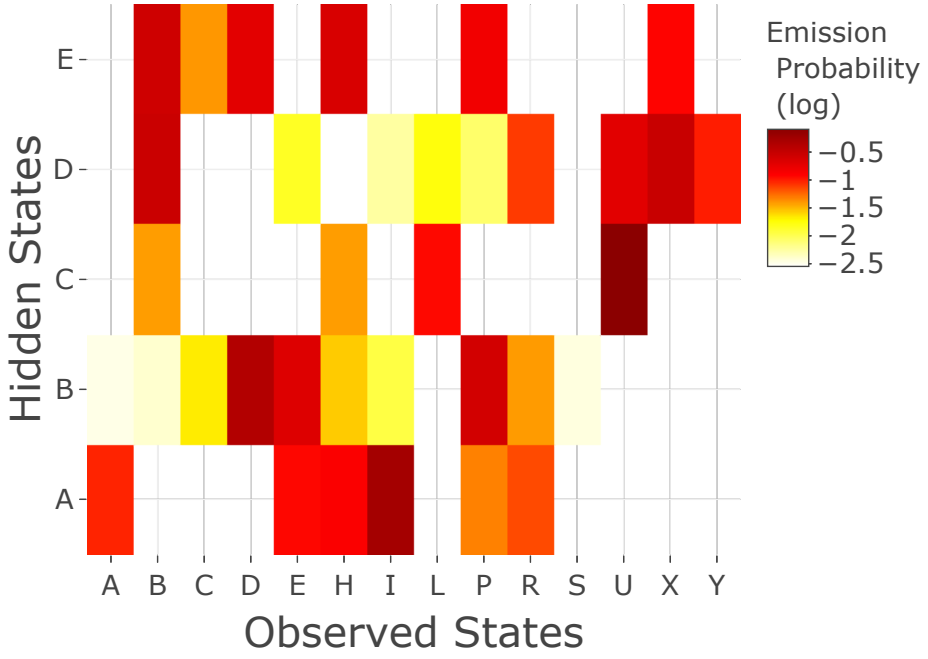


FIGURE 5.6: A heat map to visualise the emission matrix for a 5-state HMM.

We can use the Viterbi algorithm, outlined in Section 5.3.2, to predict the most likely hidden state at each time in the observed sequence. This will help us understand the proportion of time that our observed sequence exists in each state, and we display the results in Table 5.7 for the 5-state model. The majority of the clickstreams (87.4%) are spent in hidden state B, which we labelled a general 'browsing' state. State D is our next immediate state of interest, where over 7% of the data is predicted to be in this hidden state. Hidden states A and E are less frequently visited, and we note that A includes a high emission to observed state I, which we know from Figure 5.5 is not a frequently observed state, so we wouldn't expect this hidden state to be seen often. Hidden state E appears to be a pre-purchase state, and could be abandoners or focused visits without a final decision on the purchase. Hidden state C is only seen in less that 1% of the clickstream, which is labelled as the 'profile' browsing state.

Hidden States	%
A	1.96
B	87.4
C	0.783
D	7.64
E	2.19

TABLE 5.7: The most likely hidden state prediction for all clickstreams for the 5-state model.

5.5 Markov model comparison

Analysing clickstream sequences as states in a first order Markov chain is a good place to start. It provides a simple structure such that we can quickly learn about the data. In practice, we have used this model to learn about the movements of users around the entire structure of the website. Further, we use the transition matrix, to find potential inaccuracies in the data and poor website design. This research led to the wider team searching for new methods to collect this data, namely a new 'custom' event type was created, which is able to more accurately detect purchase and other specified browsing events. The sales team used the Markov visualisation in Figure 5.4b to provide feedback to the website designers on weaker (less sticky) areas of the website. In this example, the search box layout was updated and the results page simplified.

The second order Markov chain does not provide a nice visual summary, see Figure 5.4b versus 5.5a, however it does provide an extra layer of memory. We motivated why this could help us in our application with the 3 purchasing states. We aim to compare the predictive capability from both models for purchases made on the site. More data cleaning steps were added as a direct result of the second order Markov chain, bringing clarity on the multiple browser windows or the lag effect when collecting the data.

Markov chains provide simple tools to analyse our data, but the large number of observable states and messy data causes difficulty in interpreting the fitted models. We could add more states of memory to the Markov chain, however we do not see any need for this in our application, and interpreting the results would be even more challenging. The hidden Markov model uncovers a layer beyond the observed data and web page transitions, through learning about similar journeys, over more than two steps in a sequence. We learn more about the user than before, as a result of interpreting the latent variable. The complexity of HMMs adds difficulty in stabilising the model fit. The estimated parameters of an HMM alone do not inform us about how 'good' the model fit is, labelling the meta-states (if they even have an interpretation) can be masked without careful inspection. The HMM doesn't respect the web page structure, as the simple Markov chain does, but we hope that the underlying type of behaviour will provide extra information we can use. In practice, for predicting a purchase, we will explore possible ways to do this in Chapter 6. Further, we will later discuss some model comparison techniques, as there are a number of areas of sensitivity when training HMMs, unlike first or second order Markov chain models.

5.6 Discussion

This chapter has described a range of Markov models that can help us to analyse clickstream data. We began with a first order Markov model, which allows us to see a transition matrix of how users are moving around the website. We inspected the three main purchasing states in greater detail, and we required an extra state of memory to discover more about them so a two-step Markov chain model was fitted.

We outlined the motivation for fitting clickstream data to a hidden Markov model, to find meta-states in the user behaviour. We outlined the forward and Viterbi path algorithms, which we used to find the likelihood of an observable sequence and optimum hidden state sequence respectively. We described the Viterbi training algorithm and fitted a 5-state HMM, and possibly found some of the behavioural modes

we were searching for. The next steps with these Markov tools is to make comparisons between them and the GLMs we encountered in the previous chapter.

Chapter 6

Exploration and comparison of models for clickstream data

6.1 Chapter overview

This chapter will discuss a range of models that we have explored throughout previous chapters to analyse clickstream data. First, we will outline GLM variable selection criterion based techniques, applied to a set of summary variables of clickstream data. Next, we discuss the robustness and sensitivity of training HMMs on the same clickstream data set. In Section 6.4, we present a simulation study of the trained HMMs, to justify the validity of the model. We combine the two modelling approaches, GLMs versus Markov, and produce an informative plot to compare and evaluate the predictive capability of each model type.

Using a criterion-based model selection for GLMs and subjective interpretation-based comparison for HMMs to find the best models, we build a prediction visualisation inspired from the literature, where we wish to predict a conversion based on the customer journey [17]. The ability to accurately predict a successful conversion in the customer click journey provides a useful insight for online advertisers. An action could be taken to attract the customer's attention, by showing an advert or voucher with discount, if we can identify that engagement with the product is falling and more persuasion is needed for the customer to purchase.

6.2 Predicting a conversion using GLMs

In Chapter 4, we described GLM theory and in Section 4.4.1 we fitted a logistic regression model to the Adwords data set. We examined the output probabilities and some model diagnostic and evaluation techniques. Likewise, we can use a logistic regression model to predict the probability of a conversion based on web browsing data. First, let's describe the covariates we can create from clickstream data.

6.2.1 Clickstream data as sessions

To build a logistic regression model from the clickstream data we described in Section 5.4, we must aggregate data to create numerical and categorical covariates, which provide statistical summaries of clickstream journeys. Our first step is similar to Section 2.3.2 where we aggregated single page visits into session-level statistical summaries. This type of data is lost when we condense the observations to categorical page type sequences. The aim of the logistic regression model is to predict, based on a set of covariates, the probability of a conversion occurring in the current

session. We use the page type Y to identify a successful conversion, which in turn produces a binary label of a conversion within a session or not.

Table 6.1 shows a summary of the covariates that we will use to fit the logistic regression model. Each row will represent a session, j, for a profile, i. The independent variable is a binary outcome, i.e. does a conversion occur in the current session, denoted by y_{ij}.

Some of the dependent numerical summary variables for the current session are the same as in Section 2.3.2, namely g_{ij}, t_{ij}, n_{ij}. Furthermore, we previously encountered the device type, m_{ij}, and the time of the session, which is provided through a combination of the variables - h_{ij} and w_{ij}. We have included intent score, q_{ij}, created using the algorithm described in Chapter 3, which assesses the profile's browsing behaviour over the previous 60 days. We include the duration of the previous session, $t_{i,j-1}$ for a more accurate measure of the most recent browsing activity.

We choose a subset of the numerical covariates above, and calculate the the cumulative values of recent behaviour over the previous 7 days, as a summary of activity over this period. We wish to see if recent historical information is useful in predicting the outcome of a conversion. In particular, we find the totals for the number of previous page visits (N_{ij}), sessions (J_{ij}), length of time (T_{ij}), and conversions (Y_{ij}). Intuitively, if we can summarise recent behaviour through these features, then we would expect that browsing a product over a longer period of time shows more intent. This is a similar approach to the frequency and intensity attributes of behaviour in Chapter 3.

Now we have a set of covariates we can use to fit our model, and these features are not involved in any of the Markov models that we will discuss later, hence they should provide a different account of the data. We show the correlation matrix between the numerical variables we have calculated in Figure 6.1. As expected, the variables based on historical behaviour have strong correlations, the largest 0.81, between the total sessions and total page visits. It is also worth noting that intent score (a one-dimensional summary of behaviour) has medium sized correlations, between 0.34 and 0.53, with other historical summaries which is encouraging. However, we need a methodology to decide which covariates we should include in our model, and for this we will explore a criterion based approach. We randomly split the full data set into a training and test set, using the ratio 75 : 25. We do this so that we can train our models, and then use unobserved test data to check for model overfitting.

6.2.2 Criteria-based model selection

In Section 4.4.4, we introduced a criterion, AIC, which can be expressed as follows:

$$\text{AIC} = 2p - 2l$$

where p is the number of parameters in the model, l is the likelihood of the data and n is the number of observations in the data. We can use these model assessment criteria to outline a methodology that can perform variable selection for our logistic regression model. If there are p potential predictors, then there are 2^p possible models. We could choose to fit all these models, using the training data, and choose the best one according to a criterion, i.e. AIC. A reminder that we prefer models with lower values of AIC, i.e. we want to maximise the log-likelihood and minimise the number of parameters used in the model.

Variable name	Type	Definition	Notation
Conversion	Binary	Has a conversion/purchase occurred in the current session?	y_{ij}
DayOfWeek	Categorical	Is the current session on a weekday or weekend?	w_{ij}
Device	Categorical	The type of device of the profile, e.g. Smartphone, Desktop	m_{ij}
HourOfDay	Categorical	The hour of the visit split as in Table 4.3	h_{ij}
IntentScore	Numerical	The score provided by the algorithm in Chapter 3	q_{ij}
InterSessionDuration	Numerical	The length of time since the previous session for the same profile	g_{ij}
PageVisitsInSession	Integer	The number of page visits in a session	n_{ij}
PreviousSessionDuration	Numerical	The length of the previous session	$t_{i,j-1}$
SessionDuration	Numerical	The length of time of the current session	t_{ij}
TotalConversions	Numerical	The total number of previous conversions prior to the current session	$Y_{ij} = \sum_{k=1}^{j-1} y_{ik}$
TotalDuration	Numerical	The sum of all previous page visits from sessions before the current session	$T_{ij} = \sum_{k=1}^{j-1} t_{ik}$
TotalPageVisits	Numerical	The total number previous page visits in all session before the current session	$N_{ij} = \sum_{k=1}^{j-1} n_{ik}$
TotalSessions	Numerical	The total number of previous sessions before the current session	$J_{ij} = \sum_{k=1}^{j-1} 1$

TABLE 6.1: A table of the statistical variables we can generate from the clickstream data, aggregated to the session level.

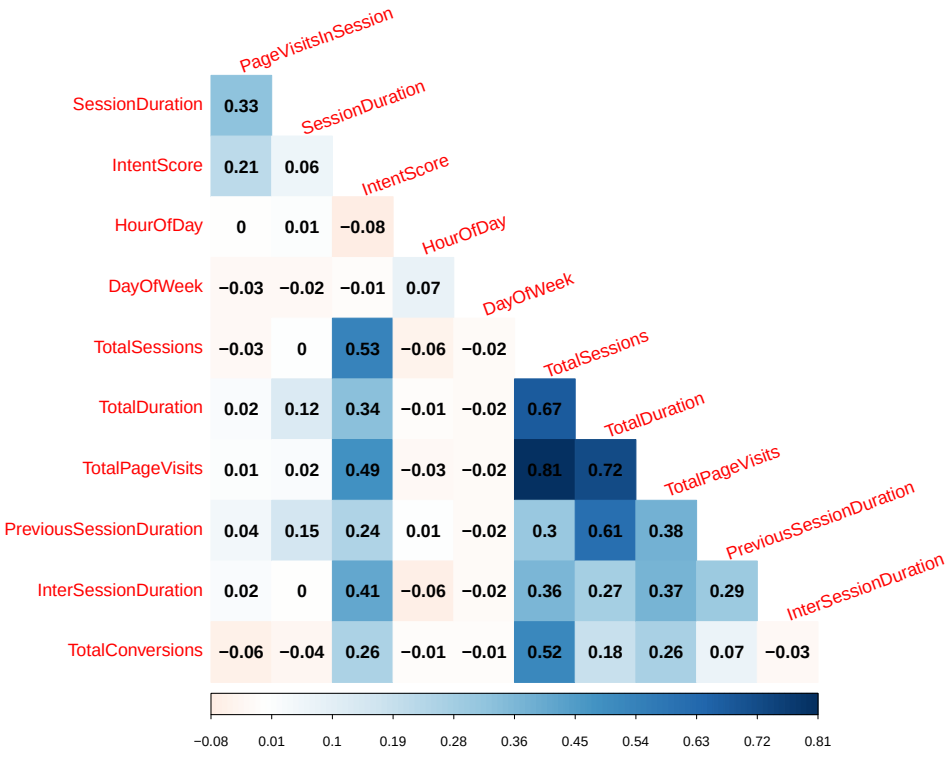


FIGURE 6.1: Correlation matrix between continuous numerical variables.

Stepwise variable selection does not evaluate the AIC for all possible models, but instead uses a search method that compares models sequentially. We describe the main steps for the **forward** stepwise model selection [61]:

1. Start with a model with no covariates, the intercept model.
2. Test the addition of each covariate to the model from a pre-defined set. Add the covariate to the model which provides the minimum AIC value.
3. Repeat the step (2) until all remaining covariates increase the AIC value for the model.

The algorithm can be reversed such that it begins with the full (saturated) model, containing all possible covariates, and attempts to remove a variable at each step until the termination step is reached. This method is called **backward** stepwise model selection. Further, combining the two approaches and allowing the stepwise method to move 'forwards' and 'backwards', we have a method for the inclusion and exclusion of variables.

The full logistic regression model, Eq 6.1, contains all the covariates in our data set. Note, we have removed the variable, TotalDuration T_{ij}, due to the high correlation shown in Figure 6.1 to avoid possible issues arising from multicollinearity. On the other hand the intercept model, Eq 6.2, contains with no covariates from our data. We decided not to include any higher order terms or interactions in our logistic regression models to maintain the interpretabilty of covariates, and in future work we may wish to introduce these concepts.

$$\begin{aligned}\text{logit}(y_{ij}) \sim \alpha &+ \beta_1 w_{ij} + \beta_2 m_{ij} + \beta_3 h_{ij} + \beta_4 q_{ij} + \beta_5 g_{ij} + \beta_6 n_{ij} \\ &+ \beta_7 t_{i,j-1} + \beta_8 t_{ij} + \beta_9 Y_{ij} + \beta_{10} T_{ij} + \beta_{11} N_{ij} + \beta_{12} J_{ij} \qquad (6.1)\\ \text{logit}(y_{ij}) \sim \alpha & \qquad (6.2)\end{aligned}$$

The forward selection method begins with the intercept model, Eq 6.2, and the results of the stepwise procedure are displayed in Table 6.2 [62]. The intercept model has an AIC value of 1955, and the first variable to add is the TotalConversions, Y_{ij}, reducing the AIC to 1746, which suggests this is a useful predictor which is intuitive. If the customer has purchased recently, they might be more likely to have a loyalty to the product and brand, and so will return. The next covariate to be added is the number of page visits in the current session, n_{ij}, and the AIC reduces to 1573. These two covariates provide the largest decreases in AIC, whereas the consequent variable additions do not cause the same proportion of reduction in AIC.

Step	Model	AIC
0	Intercept model (6.2)	1955.4
1	Add Y_{ij}	1746.91
2	Add n_{ij}	1572.99
3	Add q_{ij}	1535.86
4	Add g_{ij}	1534.45
5	Add N_{ij}	1533.12
6	Add J_{ij}	1518.88
7	Add t_{ij}	1517.39
8	Add m_{ij} (6.4)	1516.69

TABLE 6.2: A summary of the forward stepwise AIC model selection.

Step	Model	AIC
0	Full model (6.1)	1521.69
1	Remove h_{ij}	1519.69
2	Remove $t_{i,j-1}$	1517.77
3	Remove w_{ij} (6.4)	1516.69

TABLE 6.3: A summary of the backward stepwise AIC model selection.

Similarly, the backward direction algorithm in Table 6.3, provides the same result. This method removes three covariates, which reduces the AIC from 1521, the full model, to 1517. The best set of covariates that both selection methods reach is presented in Eq 6.3 and 6.4. It is satisfying that the result is the same for both forward and backward selection, and we will continue to check the validity of this set of covariates in the next section.

$$\begin{aligned}\text{logit}(\text{Conversion}) \sim \alpha &+ \beta_1 \text{TotalConversions} + \beta_2 \text{PageVisitsInSession} \\ &+ \beta_3 \text{IntentScore} + \beta_4 \text{InterSessionDuration} + \beta_5 \text{TotalPageVisits} \\ &+ \beta_6 \text{TotalSessions} + \beta_7 \text{SessionDuration} + \beta_8 \text{Device} \qquad (6.3)\\ \text{logit}(y_{ij}) \sim \alpha &+ \beta_1 Y_{ij} + \beta_2 n_{ij} + \beta_3 q_{ij} + \beta_4 g_{ij} + \beta_5 N_{ij} + \beta_6 J_{ij} \\ &+ \beta_7 t_{ij} + \beta_8 m_{ij} \qquad (6.4)\end{aligned}$$

Variable selection algorithms aim to construct a model that predicts well or explains the relationships in the data. Automatic variable selections are not guaranteed to be consistent with these goals. Criterion-based stepwise methods involve a wide search and compare models, but we should only use these methods as a guide [63]. For example, our final model includes both the total page visits and sessions, N_{ij} and J_{ij}, with the highest correlation value of 0.81 in the correlation matrix of all variables. Hence, we should consider the final parameter estimates of the model to see if they seem sensible, check residuals and model diagnostics, as well as exploratory data analysis to check for multicollinearity.

6.2.3 Model evaluation and diagnostics

We explore the covariate selection that both directions of the stepwise method produced, and we have presented the formulas in Eq 6.3 and 6.4. A summary of the model fit is given in Table 6.4, where we display the coefficient estimates and standard errors, along with the p-values to assess whether the covariate is significant in the fitted model. Hence, we can see that all variables, except g_{ij} and when m_{ij} = *SmartPhone* (where the baseline category is *Desktop*), are significant in this model at a p-value of 0.1. The final row provides the log-likelihood, -747, for the fitted logistic regression model.

The odds ratio column provides the exponential of the coefficient estimates, and we use this to interpret each covariate. For every unit count of the total number of conversions, Y_{ij}, there is the large increase of 1.37, on the likelihood of a conversion, which shows historical purchasing behaviour is a good predictor in the model. The intent score, q_{ij}, covariate has the largest value of the odds ratio, at 1.76, suggesting that it is a good indicator of conversions. The score meets the expectation that higher scores should represent more intent to purchase.

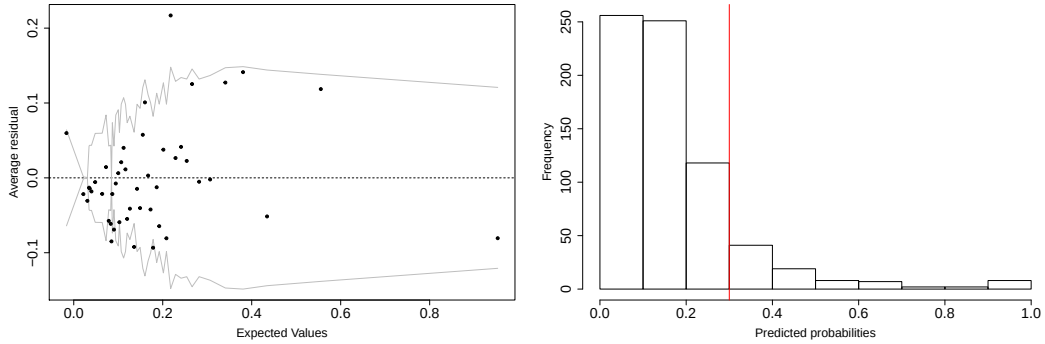
The reference category for m_{ij} is desktop, and the tablet category reduces the odds of a conversion slightly by 0.93. This agrees with the expectation that higher valued items are purchased on a desktop as opposed to a handheld device. Other covariates have very small effects on the odds of a conversion, possibly suggesting they don't have a major effect.

Symbols	Coefficients	Estimates	SE	p-value	Odds ratio
α	Intercept	-0.2036	0.0587	0.0005	0.8158
Y_{ij}	TotalConversions	0.3176	0.0323	2×10^{-16}	1.3738
n_{ij}	PageVisitsInSession	0.0164	0.0014	2×10^{-16}	1.0165
q_{ij}	IntentScore	0.5639	0.1281	1.13×10^{-5}	1.7576
g_{ij}	InterSessionDuration	2.989×10^{-7}	1.941×10^{-7}	0.1236	1.0001
N_{ij}	TotalPageVisits	-4.823×10^{-3}	1.163×10^{-3}	3.49×10^{-5}	0.9952
J_{ij}	TotalSessions	0.0323	8.180×10^{-3}	8.15×-5	1.0328
j_{ij}	SessionDuration	-2.202×10^{-5}	1.118×10^{-5}	0.0491	0.9999
m_{ij}	Device - SmartPhone	-7.750×10^{-3}	0.0162	0.6320	0.9923
	Device - Tablet	-0.0705	0.0327	0.0312	0.9320
	Log-likelihood	-747.3426			

TABLE 6.4: The parameter estimates for the fitted logistic regression model.

Inspecting model diagnostics, Figure 6.2a shows a plot of binned residuals from the training data set observations versus the fitted values from the model. The residuals have been grouped and aggregated to find the average residual over an interval

of similar predicted values from the model. The grey lines represent ±2 standard error (SE) bands, which we would expect to contain about 95% of the observations [44]. This model looks reasonable, in that the majority of the fitted values seem to fall within the standard error bands.



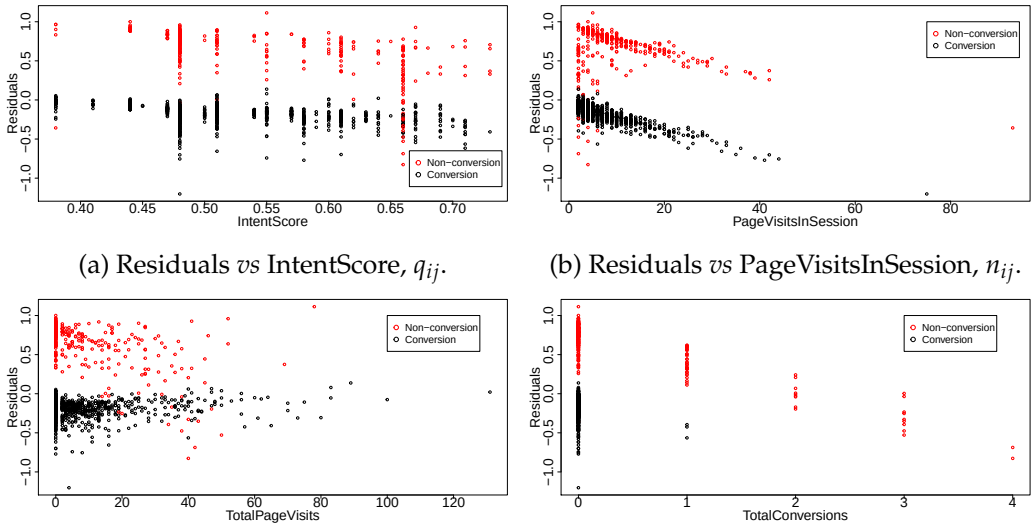
(a) A plot of binned residuals versus fitted values.

(b) Histogram of predicted probabilities for conversions.

FIGURE 6.2: Plots relating to the fitted logistic regression model.

A further method for checking the residuals is to plot them against explanatory variables, where we want to observe no pattern, because the residuals and covariates should not be correlated. Overall, the plots in Figure 6.3 suggest there are two distinct groupings of the residual values. These two groupings correspond to observations that did convert and those that did not, as displayed by the colouring in Figure 6.3. This could be interpreted that we under-predict conversions, due to the majority resulting in a negative residual and over-predict for non-conversions highlighted by the positive residuals. Figure 6.3a shows a consistent spread of residuals as intent score changes, which is encouraging. Figure 6.3b and Figure 6.3c show patterns that may suggest the residuals are heteroscedastic, so we may need to revisit these covariates. The spread of residuals decreases as Y_{ij} increases in Figure 6.3d, however this could be due to very few observations with a non-zero value for Y_{ij} (2.8%).

Using the predicted probability distribution in Figure 6.2b, we choose a fixed threshold of 0.3 to determine a binary outcome, i.e. a conversion or not. We discretise the predictions such that we can obtain a confusion matrix, with key statistical metrics which provide an interpretable summary of the predictive capability of the model. We decided on the threshold of 0.3 as we found this value led to the best balance of evaluation metrics; further tuning of this parameter would be done in future work. Table 6.5 displays the confusion matrix for the resulting conversion predictions. From these values, we obtain the accuracy (84.7%), recall (59.8%) and precision (42.3%). Our model under-predicts with this threshold, so we may want to revisit this choice when we directly compare to Markov models later in this chapter. This result is similar to other GLMs we studied in Chapter 4, which may be a feature of our data sets with low volumes of positive samples.



(a) Residuals *vs* IntentScore, q_{ij}.

(b) Residuals *vs* PageVisitsInSession, n_{ij}.

(c) Residuals *vs* TotalPageVisits, N_{ij}.

(d) Residuals *vs* TotalConversions, Y_{ij}.

FIGURE 6.3: Plots of the residuals from the fitted model against key covariates.

		Reference Y_{ij}		
		0	1	Total
Predicted $\hat{Y}_{ij}$	0	551	74	625
	1	35	52	87
	Total	586	126	712

TABLE 6.5: The confusion matrix for the conversion predictions from the logistic regression model on the test set.

6.3 HMM robustness of model fitting

We aim to outline an approach to explore and assess the training of HMMs through a range of areas of comparison. This is a particularly challenging task, with no selection method or algorithmic approach to finding the optimal model, as in the context of GLMs. One approach that we could investigate in the future, is to apply a probabilistic distance metric for measuring the dissimilarity between pairs of HMMs, though this is beyond the scope of this thesis [64]. A mixture of numerical summaries and subjective opinions on HMM parameters, could help us summarise the robustness of training HMMs on clickstream data. Thus, to evaluate HMMs we consider the behaviour of estimated model outputs, $\hat{A}$ and $\hat{B}$, alongside the predictive performance on unseen data.

In the following sections, we explore areas of sensitivity in the training of HMMs, namely:

1. In advance, we must specify the **number of hidden states**. There is no physical representation of the system we are attempting to model, hence we do not know the correct number of hidden states to search for. Does changing the number of states input affect the model fit, can we find a maximum and minimum? How many states until the interpretability of the model vanishes? How many states do we need to create enough separation, such that we can interpret distinct states as behavioural meta-states?

2. We are not limited by the **size of training data**, so we ask the question: does the size of data have an impact on the fitted model? We assume a larger data set will increase the training time of the model, but how much can we improve the model fit? Is there a limit to the input data size for the algorithm implementation we are applying?

3. We must specify the **initial values** of the HMM parameters before training. Do probabilities of 0 or 1, in any of the initial matrices affect the model fit? Do larger values on the diagonal of the transition matrix affect the model fit?

6.3.1 Choosing the number of hidden states

Our goal is to infer possible behavioural characteristics from the meta-states, and some possible examples of hidden behavioural meta-states could be:

1. Purchasing
2. Product browsing
3. Updating profile preferences
4. General online browsing

Thus, from an interpretation perspective we could motivate that there are at least 3 or possibly 4 distinct behaviours that we would hope to detect. For various different numbers of hidden states, we will explore the fitted transition and emission matrices, and additionally the numerical measure of the proportion of occurrences of each hidden state and log-likelihood of a sample of training data using the Forward algorithm from Section 5.3.1. The hidden state transition matrices in Figure 6.5 describe the Markov chain process between the hidden states. Figure 6.4 shows the emission probability matrices visualised as a heat map. If the observed states for a given hidden state have a non-zero probability, then they can occur in that state. We can use these to infer an interpretation of the hidden states, and we delve deeper into the particular states of interest using the Viterbi path predictions in Table 6.6.

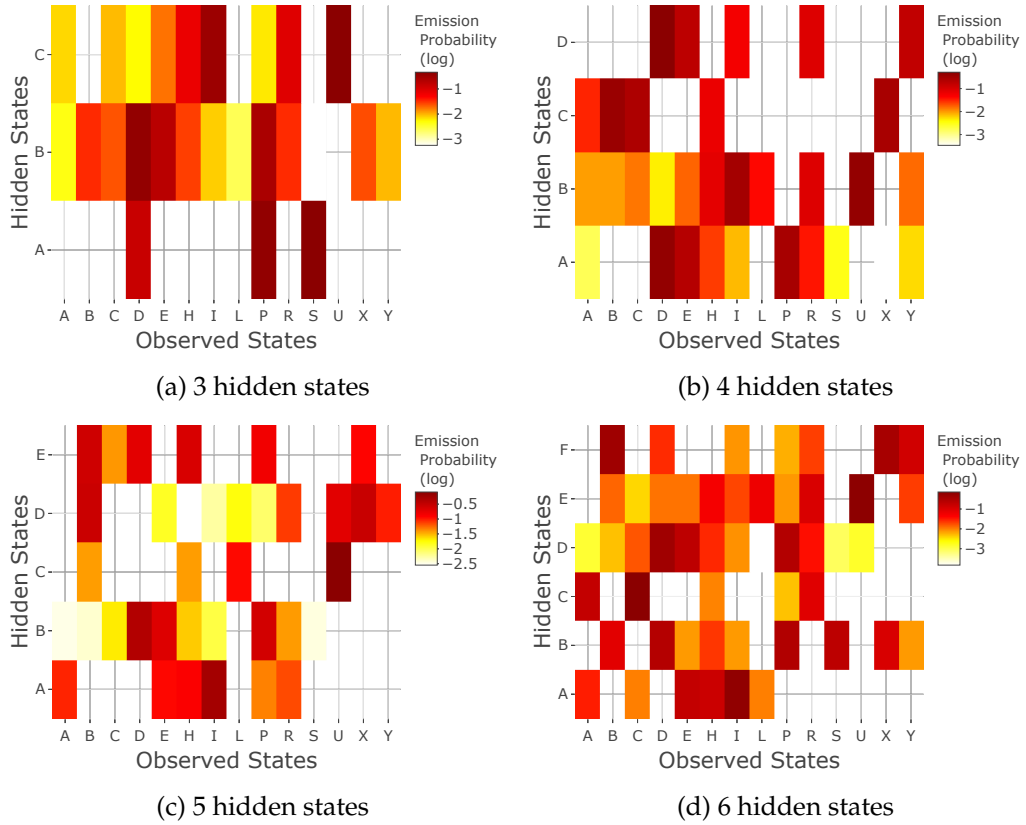
As previously mentioned in Section 5.4.3, a sensible maximum for the number of hidden states would be 13, the number of distinct observable states, as more could give rise to degenerate solutions. A sensible minimum would be 2 states, and currently we have nothing to suggest we should increase this lower bound.

Starting with the 3 hidden state model in the emission matrix in Figure 6.4a, we expect this to be too few hidden states to separate types of clickstream behaviour. We would observe this by finding 'noisy' hidden states, which contain emissions to many observable states and there is little distinction between the hidden state structures. For example, we can see this in hidden state B in the emission matrix, which contains the majority of observable states, and Table 6.6 predicts almost 95% of the data exists in state B. Further, from Figure 6.5a the most likely transition from hidden state B, is to state C with a very low probability of 0.014. Hence, the hidden states lack interpretation and we should investigate a larger number of hidden states to attempt to find a clearer structure in the emission matrix. We conclude that the 3-state model has insufficient flexibility for the hidden states to specialise, we have fewer 'average' and general behaviour states instead.

In Figure 6.4b, the 4-state model shows a little more sparsity in the emission matrix indicating less generic behaviour. Further, the Viterbi path predictions in Table 6.6 show a better separation in the data, compared to the 3-state HMM. Despite this,

in the emission matrix we observe that 3 out of 4 hidden states contain the Y conversion state, and combined these hidden states provide 90% of the Viterbi predictions. Hence, we suggest that this HMM does not distinguish the 'purchase' meta-state that we might like to discover from our HMM. The likelihood of the clickstream sequence has decreased from -45596 to -66139, possibly due to the increased complexity of the model by adding the extra meta-state.

The 5-state HMM in Figure 6.4c displays the matrix we described in Section 5.4.3. We found this HMM provided an interpretable set of meta-state behaviours. Focusing on the purchasing state, state D contains non-zero probabilities for both the purchasing X and Y observed states, and state E contains only the X checkout state without Y. As a result, we could interpret that state D could be the 'purchasing' state, and state E as the 'pre-purchasing' or 'abandoners' meta-state, where the majority of the purchasing journey (up to X) can be observed except the conversion confirmation (Y). Although, this doesn't preclude the transition from hidden state D to E, to access the observed state Y. The hidden state with the fewest observable states is C, where the profile states L and U are prominent, leading us to interpret this as the 'profile preferences' viewing state. The most general state is B, which includes the most amount of page types without the conversion target states, so we could say this is the general 'browsing/shopping' behaviour state. It is worth noting that from Figure 6.5c only B and E states can transition into the purchasing state D, however the probabilities are small: 0.0274 and 0.004 respectively.



(a) 3 hidden states

(b) 4 hidden states

(c) 5 hidden states

(d) 6 hidden states

FIGURE 6.4: Heat maps to represent the **emission** probabilities for a variety of trained HMMs.

The majority of the clickstreams (87.4%) are spent in hidden state B, from Table 6.6 which we labelled a general 'browsing' state. State D is our next immediate state of interest, where over 7% of the data is predicted to be in this hidden state.

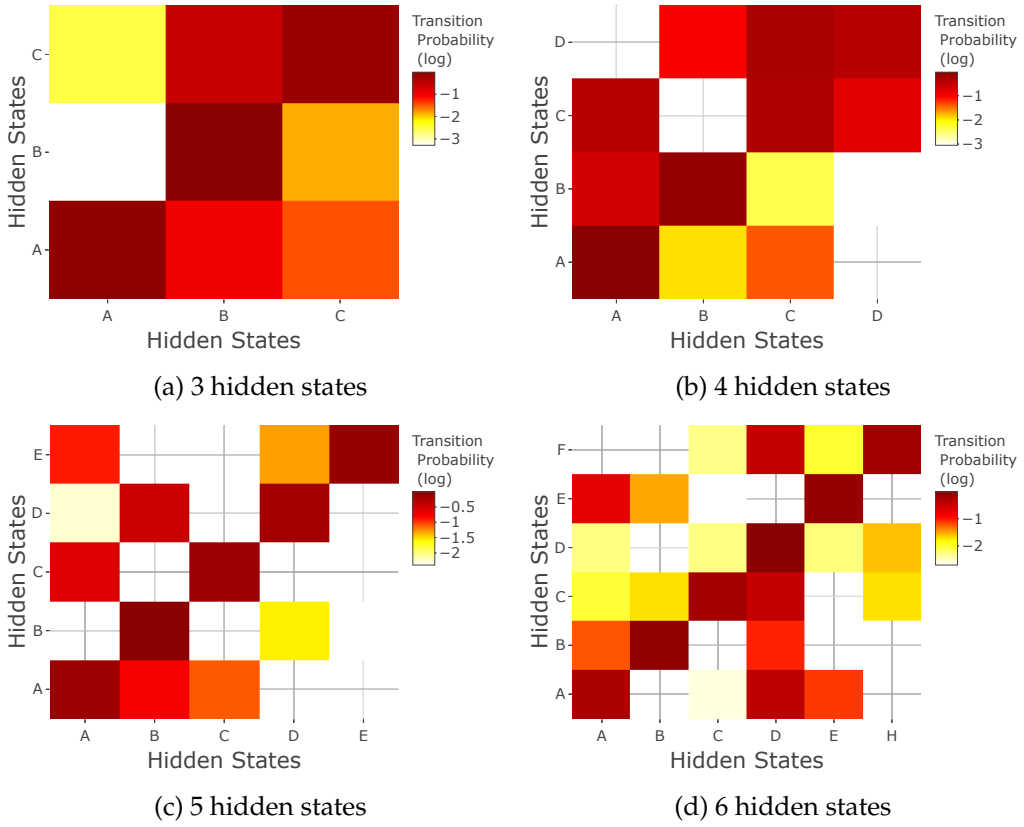


FIGURE 6.5: Heat maps to represent the hidden state **transition** probabilities for a variety of trained HMMs.

Hidden states A and E are less frequently visited, and we note that A includes a high emission to observed state I, which we know from Table 5.5 is not a frequently observed state, so we wouldn't expect this hidden state to be seen often. Hidden state E appears to be a pre-purchase state, and could be abandoners or focused visits without a final decision on the purchase. Hidden state C is only seen in less that 1% of the clickstream, which is labelled as the 'profile' browsing state. Compared to the 4-state HMM, the 5-state HMM has an increased log-likelihood for the given sequence to -65964, which is encouraging as our model is it a better fit even with an extra latent variable.

Adding an extra meta-state, the 6-state HMM emission matrix in Figure 6.4d, is dense with some distinction between the hidden states. There are 3 states; B, E and F that include the conversion states containing slightly different compositions of other observed states. The majority of the states in the clickstream sequences (over 85%) appear in hidden state D, and the transition probabilities to escape this state are small. Overall, this many hidden states makes interpretability challenging as a lot of the states are overlap. The 6-state HMM has the lowest likelihood given the same clickstream sequence, most likely due to the extra hidden state.

	% of hidden state occurrences			
Hidden States	**3 states**	**4 states**	**5 states**	**6 states**
A	0.378	84.2	1.96	2.52
B	94.9	5.23	87.4	0.639
C	4.76	8.36	0.783	1.16
D		2.18	7.64	86.5
E			2.19	2.98
F				6.16
Log-likelihood	-45596.45	-66139.56	-65964.36	-86612.17

TABLE 6.6: The most likely hidden state prediction for clickstreams for a variety of trained HMMs.

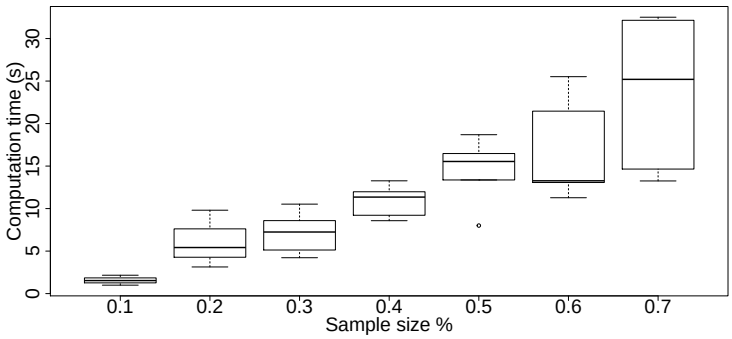
6.3.2 Size of training data

We will experiment with the sample size of the clickstream data we use to train the HMM. We expect that the trade-off will be an increase in computation time, as there is more data to estimate the parameters. Note that we are using the Viterbi training algorithm set out in Section 5.3.3, and we found that when we increased the size close to the maximum of our data set, we encountered errors caused by the computation time.

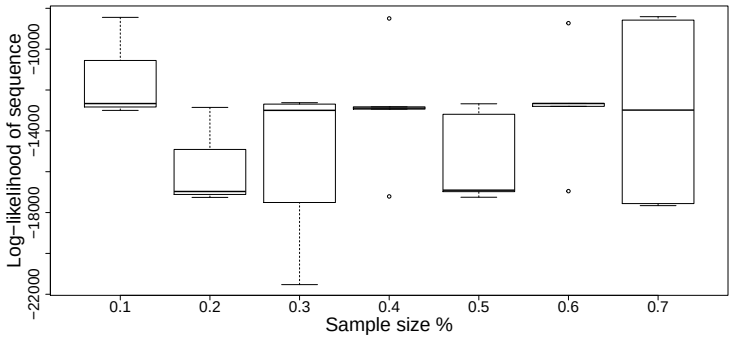
We would expect the likelihood of a given clickstream sequence to change with sample size, where we use the Forward algorithm from Section 5.3.1. By increasing the size of the training data, we would expect the HMM to be a better fit to our data. Thus given a constant sequence across two models, we expect the likelihood of the sequence to be higher for the model with more training data. However, with more training data we may observe more intricacies in the data, causing a more complex model which includes rarer observed states and transitions, which may alternatively decrease the likelihood of a given sequence.

Firstly, we explore how the computation time differs by varying the data input size, our full data set consists of 25000 page types across 5000 browsing sequences. Figure 6.6a shows the proportion of the data set used for training against computation times. We repeated the training to obtain a distribution of computation timings, and in general it increases approximately linearly across our results, as the proportion increases the computation time also increases. The model training failed on some attempts due to the data size, and the number of samples here is a maximum of 5. This is a small sample due to the computational complexity involved and the high frequency of failed training attempts. In practice, we will focus on the predictive capability of the model as opposed to the training time, as this is not a priority.

Figure 6.6b displays the proportion of the data set used for training against the likelihood of the same observed sequence. There is no clear pattern in the plot, suggesting that other factors may be influencing the values. We did not expect to see anything clear here, as more data could mean a more complex model that includes more rare observed states and transitions, hence we can conclude that we should use close to the maximum input data size as possible.



(a) Training size *vs* computation time.



(b) Training size *vs* sequence log-likelihood.

FIGURE 6.6: Plots to compare HMM training with different input data sizes.

6.3.3 Initial parameter estimates

Before training an HMM, the elements we must specify are the initial state distribution, the hidden state transition matrix and the emission matrix. We want to test the sensitivity of the training of HMMs, by changing the initial parameter estimates for the transition and emission matrices. The variations we will briefly consider in this section are: what happens when we use fitted HMM parameters as input values (recursive) and is the result sensitive to extreme starting probabilities of 0 or 1?

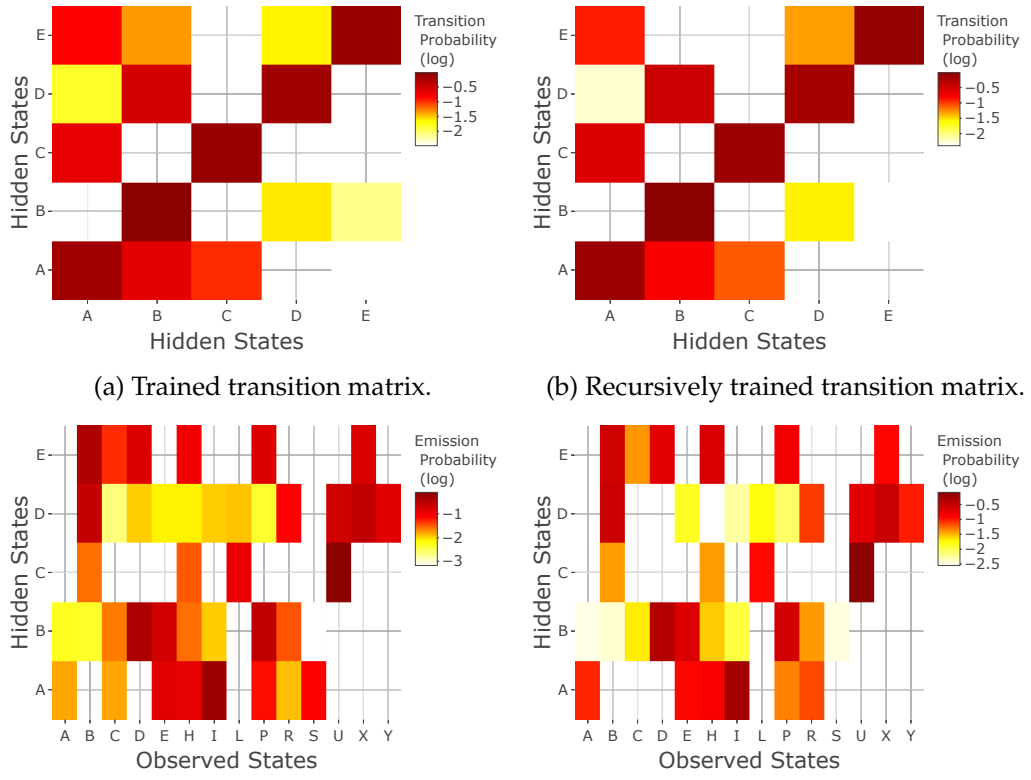
The initial estimates we use throughout training HMMs are random uniform values between 0 and 1 (exclusive), with larger values along the diagonal of the transition matrix. These matrices are scaled such that they define a matrix of the appropriate form. We decided to set these values so as to not induce bias in the training of the HMMs. The larger values on the diagonal of the transition represent a higher transition probability to remain in the current state, this is a typical feature of Markov chains. Assessing the performance of a variety of input values on a large scale is challenging and time consuming, so we present a few examples to illustrate the key takeaways.

We explore the recursion idea first and begin by training a 5-state HMM using the standard method we described above for the initial parameter estimates. We will use half of our data set to train the HMM, resulting in the transition and emission matrices in Figures 6.7a and 6.7c. The recursion idea is to use this information to inform the initial values to recursively train an HMM on the other half of our training data. The 'second' training of the 5-state HMM arrives at the matrices in Figures 6.7b

and 6.7d, which generally show a lot of similarity to the matrices in Figures 6.7a and 6.7c. We also used the full dataset to train a 5-state HMM, and the results can be seen in Figure 6.8. This HMM contrasts from the recursively trained HMM, where the combination of observed states in each hidden state is different and therefore interpretation of each label changes. Hence, from this example we suggest that the recursive method can stabilise the training of an HMM, but with the caveat that it is still dependent on the first training.

Another option we consider are extreme values in the initial parameter estimates, i.e. 0's and 1's. We added 0's to the starting probabilities and the initial estimates are shown in Figures 6.9a and 6.9c. In total, we set zero's for approximately 35% and 40% of the initial transition (non-diagonal) matrix and the emission matrix respectively. We trained an HMM using these inputs, which resulted in the updated parameters in Figures 6.9b and 6.9d. We found that including 1's caused the Viterbi training algorithm to fail, which is sensible as training data that opposes this in the transition or emission matrix will cause an error. On the whole, we can conclude that the initial parameters do not dictate the form of the trained parameters.

In conclusion, the initial estimates can affect the model, and we assume a likely scenario is that the position of the 'conversion' hidden states may shuffle. The meta-states may need to be re-interpreted as they may not be in the same position with different training. Recursive training could help to stabilise and provide a more accurate model, and we should avoid adding 0's and 1's in the initialisation for good practice.



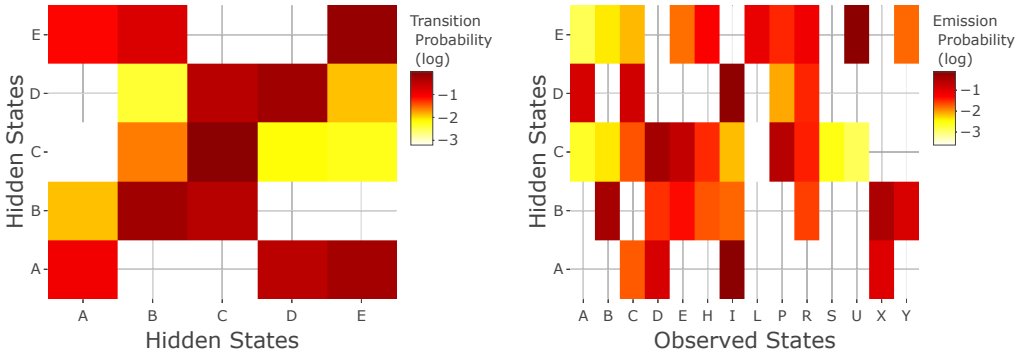
(a) Trained transition matrix.

(b) Recursively trained transition matrix.

(c) Trained emission matrix.

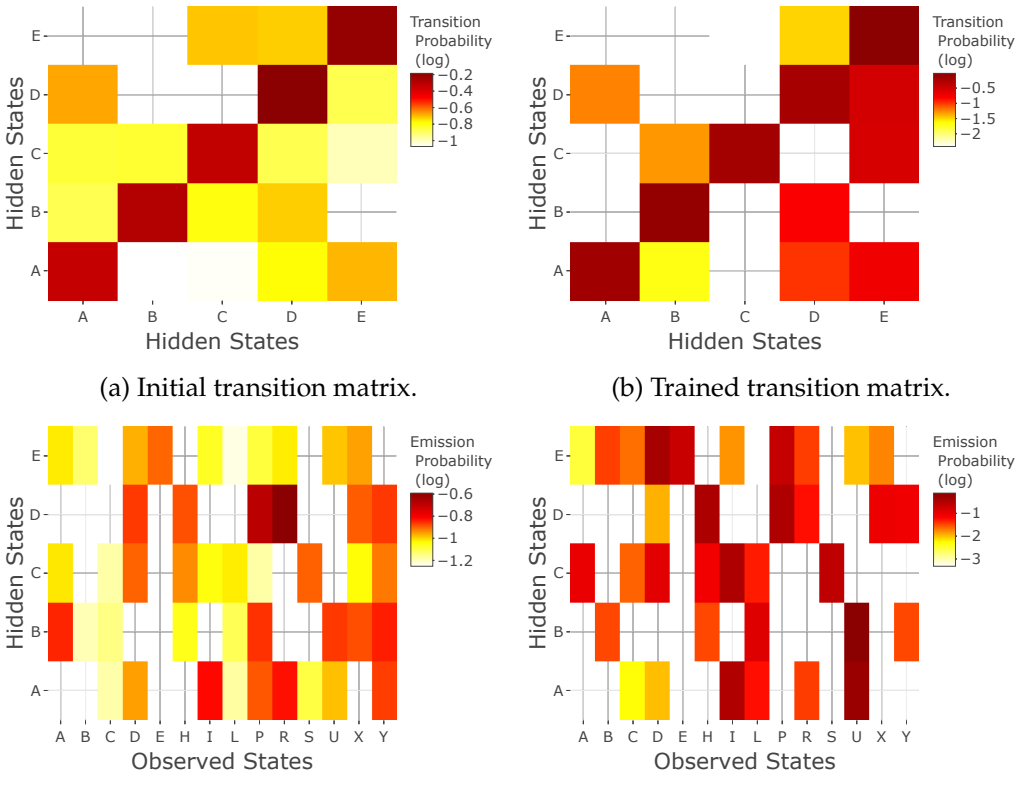
(d) Recursively trained emission matrix.

FIGURE 6.7: A comparison of training using initial parameters by recursion.



(a) Transition matrix. (b) Emission matrix.

FIGURE 6.8: A training of a 5-state HMM using the full dataset.



(a) Initial transition matrix. (b) Trained transition matrix.

(c) Initial emission matrix. (d) Trained emission matrix.

FIGURE 6.9: A comparison between the parameter estimates with added zero values before and after training.

6.4 Simulation study of generated sequences

Finally, we hope to explore simulated data from a fitted HMM to make comparisons with real data to assess the plausibility of the type of data it can generate. Our expectation of a good model is that for a large data size that it will be reasonably similar when compared against what we actually observed. We will calculate the transition matrix for the simulated clickstream data and compare against the real data, which we expect may not display some of the intricacies of real user journeys.

Our method to simulate the data, begins by using the fitted HMM state transition matrix to generate a sequence of 10,000 hidden states. Consequently, we use this to generate a sequence of observed states that are simulated from the emission matrix of the fitted HMM [65].

First, we directly compare the proportions of all possible observable states in our real data set against the simulated data, where the HMMs looks plausible in terms of these summaries. The results are displayed in Table 6.7, which on the surface shows a lot of similarity between the observed and simulated data, overall the simulated figures are almost indistinguishable from the real data. However, this only means the proportions of the states are similar, it does not provide any information regarding the transition patterns between the states.

		Simulation %		
State	**Actual %**	**4 states**	**5 states**	**6 states**
A	0.48	0.44	0.44	0.45
B	3.14	3.27	2.61	3.71
C	2.11	2.24	1.86	2.43
D	37.07	37.15	38.56	37.04
E	17.96	18.35	18.91	18.01
H	3.26	3.26	3.11	2.66
I	2.35	1.97	2.16	1.99
L	0.25	0.20	0.14	0.22
P	23.92	23.96	23.27	23.56
R	3.77	3.58	4.05	3.78
S	0.25	0.32	0.25	0.28
U	2.16	1.94	1.70	2.21
X	2.40	2.46	2.33	2.73
Y	0.89	0.86	0.61	0.93

TABLE 6.7: A comparison of the proportions of observed state occurrences between the real and simulated data.

We can calculate the transition matrix, by fitting a first order Markov chain, from the simulated clickstreams, such that we explore a different aspect of the data. We re-show the first order Markov chain output from Section 5.4.1 in Figures 6.11a and 6.11b. We compare the simulated data transition matrix for the 5-state HMM simulation in Figure 6.11c, and the network visualisation is shown in Figure 6.11d. The occurrences of each state are proportional to the node sizes and the arrow widths are proportional to the transition probabilities in the network visualisations.

The simulated data contains a column of high probabilities that is not fully replicated in the real data set, this is the page type D column in Figure 6.11c. These transitions have been over represented in the simulation, which could be because they are the highest occurring states in the data. This column represents transitions from

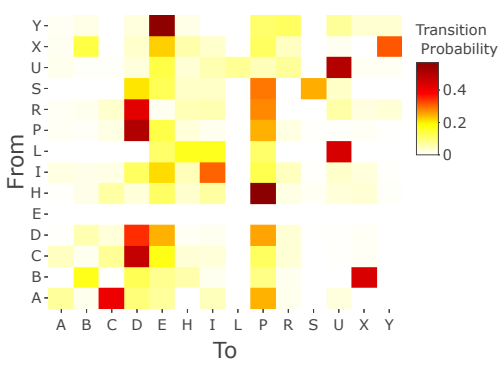
all states to the product detail page, D. There are fewer large transition probabilities in the simulation, with a reduction in some of the rarer transitions. For example, the transition probability from pages A to C in the real clickstream data is 0.41, and this transition does not exist in the 5-state HMM simulation. This highlights a concern that the model does not have data regarding this transition, from a list of articles to a single article, hence it does not appear to be good model.

Further, the purchasing states of the basket and checkout pages (B,X,Y) do not show the same intuitive structure that we see in the real data. The transition from basket B to checkout X, is much smaller (0.44 reduced to 0.16) and similarly state X to the checkout success Y (0.31 reduced to 0.06). Also, the conclusion of a purchasing journey Y to E, has the largest reduction of 0.56 to 0.06. The occurrences of these states are small proportions of the whole data set, and this coupled with the low chance of completing a full purchase in this order, causes the simulation to resort back to the more popular states. The HMM doesn't accurately capture detailed structure, as a disadvantage of the model is that it only creates dependencies between observed pages dictated by the hidden state structure of the emission matrix. From this simulation, we see the constraints placed by the model choice, where we lose some of the page visit transitions that occur in real sequences.

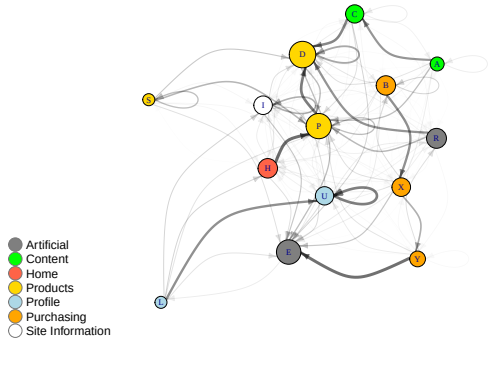
We explore a selection of the simulated clickstreams from the 5-state HMM simulation, to see if they match our expectation of real sequences, and look for unlikely transitions. We display a sample of clickstreams in Figure 6.10. Overall, the lengths of the clickstreams seem reasonable and the dominance of states P and D are also seen in the real clickstreams. In the final clickstream example, the journey begins with the checkout state X which is not sensible and would seem erroneous if identified in the original data. We conclude that the 5-state HMM simulated clickstreams are on the whole reasonable, however there are a lot of intricacies that are missed by the model.

- PPDDE
- PDPIDDPPDDDPDDE
- DDDDDDPPE
- BUBE
- PDIDDDDDRDE
- DDPDDPDDIIHIE
- DDHDPPDHE
- DDDDDDPPDPDDDDPE
- DPE
- XPHDIHPDDPPPDE

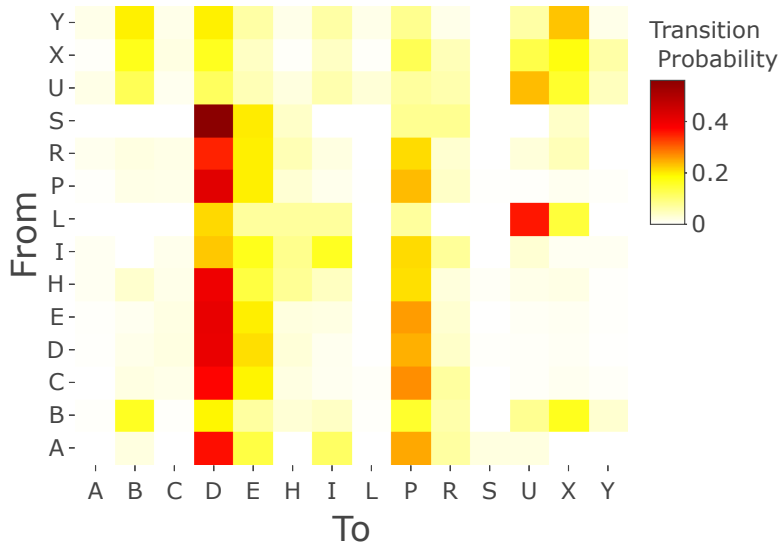
FIGURE 6.10: A list of clickstreams from the 5-state HMM simulation.



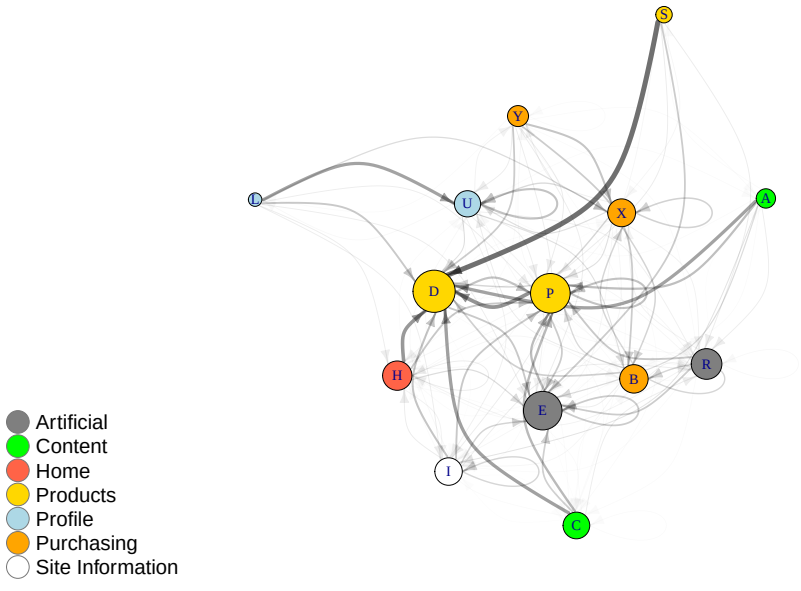
(a) Actual observations in a transition matrix.



(b) Actual observations in a transition network.



(c) Simulated observations in a transition matrix.



(d) Simulated observations in a transition network.

FIGURE 6.11: A comparison between actual observations we explored in Section 5.4.1 and the simulated data.

6.5 Model comparison by prediction

After exploring the training sensitivity of HMMs and comparing simulated sequences to real observations, we will compare the predictive capability of HMMs regarding conversions. If the HMM's hidden states are assigned specific behaviours, this could provide clearer and simple information that could be actioned in real-time. For example, a high probability of checkout could display a discount or offer to incentivise the purchase a product, or personalise the page content based on the behavioural state. We want these models to be influential in real-time decision processes, thus we place a high importance on predictions in our model selection. The models that we will compare in the section are:

- First order Markov chain model (5.4.1)
- Two step Markov chain model (5.4.2)
- Logistic regression (6.2.3)
- 4-state HMM (6.3.1)
- 5-state HMM (5.4.3)
- 6-state HMM (6.3.1)

We will include the first order and two-step Markov chain models that we explored in Section 5.4.1 and 5.4.2. Further, we will use the logistic regression model from Section 6.2.3 for predictions using numerical and categorical summaries of clickstream data.

We will explore the probability that the next move the user will make is to an observed state of interest and also the probability of transitioning to a hidden state of interest. In particular, we will focus on the successful conversion observed state, Y. We will identify which hidden states can be labelled as purchasing, and make comparisons across the HMMs we trained in Section 6.3.1. We expect that HMMs with a distinct, non-overlapping hidden state structure, which includes a version of the purchasing state will perform well in this comparison method.

We assess the potential for practical use and implementation of HMM in terms of what the next page visit will be. Exploring all future paths of all possible lengths is a computationally demanding calculation. As, ultimately, any practical use of these ideas will be required in the first 30ms of the page visit, we restricted our focus to a short term calculation.

6.5.1 Transition to observed state

In this section, we will use each model to predict the probability that the next page visit is precisely a successful conversion, i.e. the next state we observe is page type Y which we denote as P_Y. We use a sequence from the training data, and also a devised sequence to compare the performance of the models. Note the different uses of clickstream data in the logistic regression model and Markov models, however they are still predicting the same outcome and can be compared directly on the same scale.

We aim to build a prediction graphic that in a single plot can summarise the model predictions, an example of this plot is Figure 6.12. The basic features of the plot are the x-axis provides the clickstream sequence, the page types (or states): $X = (x_1, ..., x_T)$. The y-axis displays the predicted probability of transitioning to the conversion state, P_Y, and each line on the plot corresponds to a different model. Each page type can produce a prediction of the likelihood of a next transition to the observable state Y. The first order and two-step Markov chains provide transition probabilities for P_Y as:

$$P(X_{t+1} = Y|X_t = x_t) \text{ and } P(X_{t+1} = Y|X_t = x_t, X_{t-1} = x_{t-1})$$

respectively, and these are the values that we plot. The logistic regression model also provides a prediction from Eq 6.4, using historical numerical summaries and current session information. For all hidden state possibilities, $\mathcal{Y} = \{y_1, ..., y_N\}$, we predict the likelihood of observing the page type Y using the emission matrix,

$$P_Y = \sum_{i=1}^{N} P(X_{t+1} = Y, X_t = x_t, ..., X_1 = x_1, Y_t = y_i|\lambda).$$

The HMMs use the Viterbi path algorithm, Section 5.3.2, to find the optimal hidden state sequence of current journey. We can quantify the differences and similarities between the model predictions, as well as uncovering qualitatively the hidden state interpretations and discuss how we could use the predictions in a 'real-time' environment. We would expect to see an increased probability for P_Y mainly around the other purchasing states of B and X.

First, let's inspect Figure 6.12, which uses a short devised sequence to display how the plot can provide an insight into how we can assess and compare model performance. The clickstream sequence, HPDCDBXY, is created in such a way that it begins with generic browsing behaviour of products and articles, and subsequently enters in the purchasing phase of the journey. We focus on the predictions at the pre-conversion states (B and X) in the clickstream sequence, as these are the preceding states to the successful purchase, Y.

The simplest model, the first order Markov chain, which only takes into account the current state, has a peak in P_Y at the B and X states. The corresponding values for P_Y are 0.03 and 0.06 which are small transition probabilities, despite this the model highlights the power of this graphic. The two-step Markov chain displays the highest peak where $P_Y = 0.27$, corresponding to observing the pattern of B then X. The two-step model gives a better prediction in this scenario, as the final page visit is a conversion, most likely as a result of the extra state in memory. We created data for the logistic regression model, with observations at the beginning and end of the sequence. This is more of an illustration of how it could be used, however with real data we can directly compare the predictions of this model alongside the Markov models.

We display the predictions from 3 variations of HMMs, including the most likely hidden state labels at each observed state in Figure 6.12. The 'best' model prediction is the highest probability for P_Y at the page (X) before a conversion page (Y), where our models are asked the question: will the user convert from the basket/checkout? The 4-state HMM has the earliest peak at the fifth page visit, it transitions to the 'conversion' state (D), displaying the earliest indication of movement towards a purchase for this sequence. The 5-state HMM provides a shallow increase of likelihood in the purchasing phase of the sequence, and at the checkout page predicts that $P_Y = 0.03$. This is not a strong prediction, and is lower than either of the Markov chain models, but it outperforms the 4-state HMM. Our interpretation of this plot is that the best HMM is the 6-state HMM, where the hidden state transitions from browsing to 'conversion' (D to F) at the basket page and P_Y is the highest at 0.13.

We now use a real clickstream sequence from the training data, DBXPPDPDRD, and the plot is shown in Figure 6.13. This sequence begins with a visit to products, then moves to the basket and checkout states, and subsequently we observe continued site engagement with a returning visit to a product. We would expect the models to provide an increase in P_Y around the B and X pages.

We first observe that P_Y for the 4-state HMM is never larger than 0.006, indicating this model expects a very small chance that this clickstream will convert. A slight improvement in terms of displaying some interesting activity is the 5-state HMM, which peaks at $P_Z = 0.03$. The 6-state HMM and first order Markov model provide almost identical predictions throughout the clickstream sequence. The 6-state HMM moves into the 'conversion' state (F), as in the previous example, when entering the basket and checkout pages. The two-step Markov chain similarly observes the pair of B, X, and provides the highest likelihood P_Y and converting into a purchase. The logistic regression model displays a wider trend of intent to purchase, using the numerical summaries.

6.5.2 Transition to hidden state

Now we consider a variation on the plot in Section 6.5.1, where we aim to focus solely on the comparison between the HMMs. The key difference is that rather than predicting the transition to an observed state, we consider the transition probabilities to hidden states as the goal. We first label a 'conversion' hidden state based on the existence of the observed conversion page Y in the emission matrix. Note here that there may be multiple hidden states with the 'conversion' definition, which we define as $Z = \{z_1, ..., z_L\}$ where $Y \in z_l$. We aim to predict transitions into the meta-states of interest, where the exact observed state may not be relevant, but the overall behavioural meta-state journey is more useful. We define the probability of the next transition is to a 'conversion' labelled hidden state as P_Z.

The x-axis provides the clickstream sequence and the page types (or states): $X = (x_1, ..., x_T)$. The y-axis displays the predicted probability of transitioning to any 'conversion' hidden state,

$$P_Z = \sum_{i=1}^{N} \sum_{l=1}^{L} P(Y_{t+1} = z_l, X_t = x_t, ..., X_1 = x_1, Y_t = y_i | \lambda).$$

Each line on the plot corresponds to a different HMM with a variety of numbers of hidden states for comparison. An example using the devised clickstream sequence, from Section 6.5.1, is displayed in Figure 6.14. To support Figure 6.14, for the same set of HMMs and sequence, we display the full probability matrix from the forward

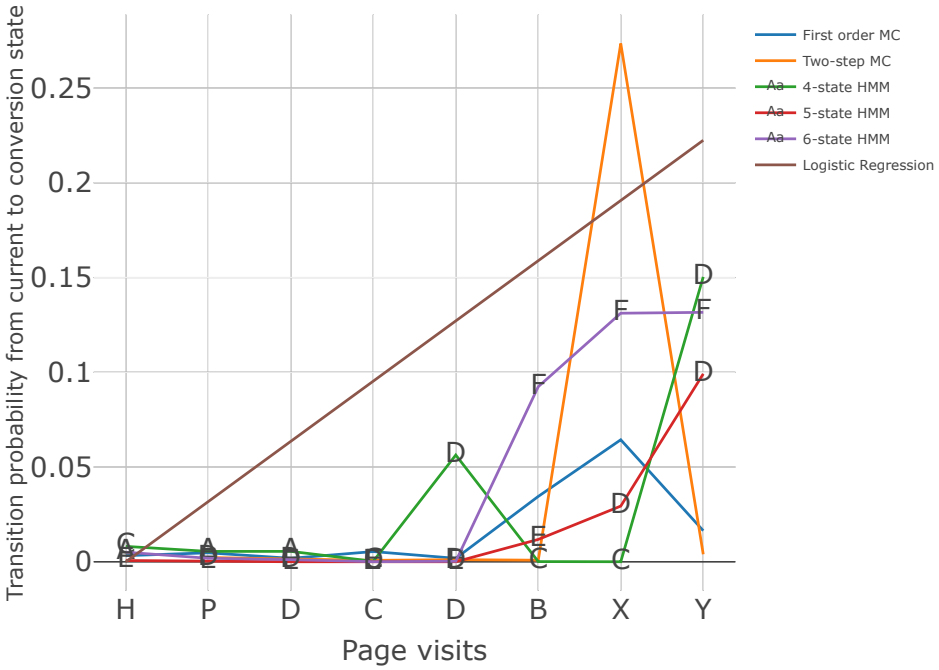


FIGURE 6.12: A graphic to display the models predictions applied to a devised clickstream sequence.

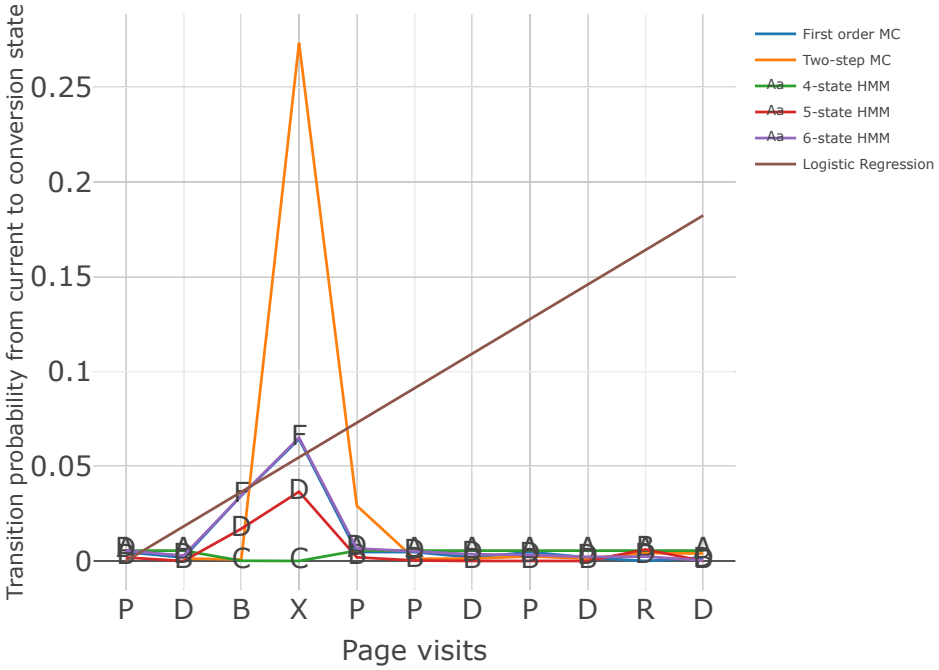


FIGURE 6.13: A graphic to display the models predictions applied to a clickstream sequence from the training data.

algorithm (Section 5.3.1), in the form of a heat map. An example in Figure 6.15a provides the probability distribution for the likelihood of each hidden state at each step in the sequence. Large forward probabilities provide high confidence that the user is in a given hidden state. We expect to see an increase in P_Z when the sequence is in the purchasing states of B and X, and a high probability of remaining in an appropriate hidden state when we observe those states. Note that a probability of $P_Z = 1$ denotes that all possible transitions from the current hidden state are to a 'conversion' hidden state.

To learn how to interpret this type of plot we inspect Figure 6.14, and immediately we note the change in scale of the y-axis compared to the similar plots in Section 6.5.1. At the first observed state, we encounter an issue with the 4-state HMM, where P_Z is very high with only one observed state which provides an unrealistic and misleading prediction. Further into the sequence, P_Z for the 4-state HMM jumps suddenly between a 0 to 1. These predictions do not seem sensible or useful, as it does not produce a stable long term meta-state that we could use to inform a real-time advertising strategy. This could be a result of the separation between Basket/Checkout (B,X) and Conversion (Y) states in the emission matrix in Figure 6.4a. Figure 6.15a emphasises these unstable hidden states, only when the sequence reaches the state Y, does a transition to hidden state E occur, and this HMM does not identify a useful purchasing meta-state.

The predictions from the 5-state and 6-state HMM overall follow a similar pattern. In particular, we focus on the basket page visit to determine if the visit will transition into the purchase phase or whether the user leaves the site or continues to browse. From Figure 6.15b, we see the 5-state HMM moves between 3 hidden states, B to E then to D (the 'conversion' state) – here the basket and checkout pages are not seen as part of the same hidden state. The 6-state HMM predicts $P_Z = 1$, which seems unrealistically high and could be a result of overfitting - too many hidden states and separation of page types. However in Figure 6.15c, we see the final 3 states are in the 'conversion' state (F), emphasising that this could be a valid interpretation.

We plot the real sequence in Figure 6.16, which appears to separate the behaviour of the three HMMs. The 4-state HMM, again provides many predictions where $P_Z = 1$, it is counter-intuitive as in the purchasing observed states the probability reduces close to 0. This is due to the emission matrix in Figure 6.4b separating the purchasing pages across multiple hidden states. Figure 6.17a emphasises that the behaviour is labelled as general browsing, and only when we observe state Y does the HMM identify a change in this behaviour. This HMM does not provide the outcome that we wanted and would provide false information to a real-time advertising system.

The predictions from the 5-state and 6-state HMM vary in Figure 6.16, overall the 6-state HMM predicts the highest value for P_Z during the basket and checkout pages. At the early stages of the clickstream journey, the 6-state HMM predicts there is a high chance of the user converting, however we don't observe an actual conversion, so the model could be misleading. In contrast, the 5-state HMM in Figure 6.17b, moves between 3 states, B to E then returning to B, nether are interpreted as 'conversion' states and the prediction probability of a conversion stays relatively low throughout. In Figure 6.17c, there is some overlap in the forward probability predictions of the most likely hidden states. On reflection, the 6-state HMM in general displays an intuitive interpretation of the clickstream journey. The sequence begins with high intent to purchase and then further browsing reduces the chance of being in a 'conversion' meta-state, and is more likely to be in a browsing meta-state.

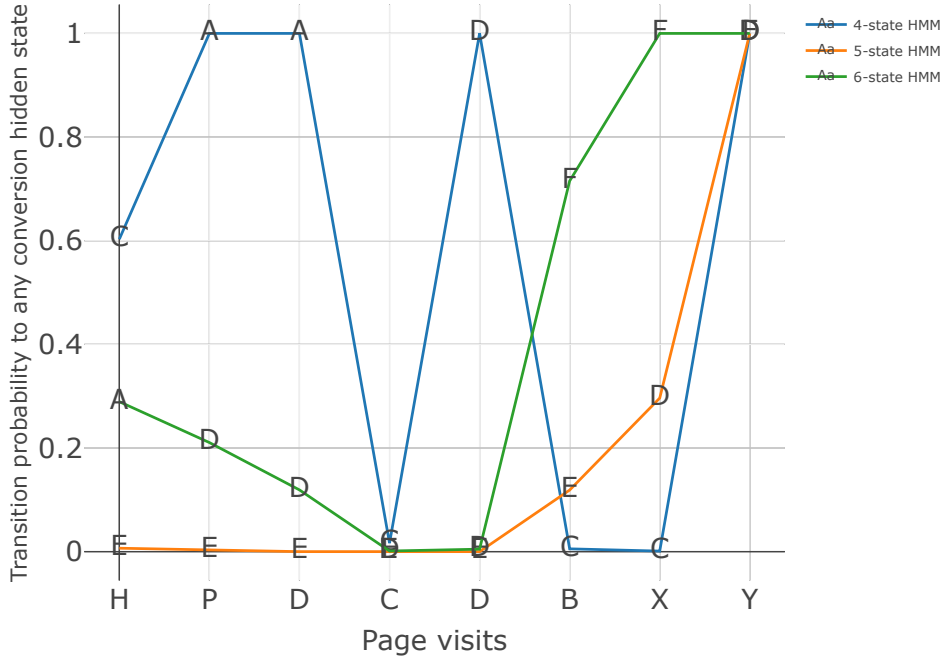
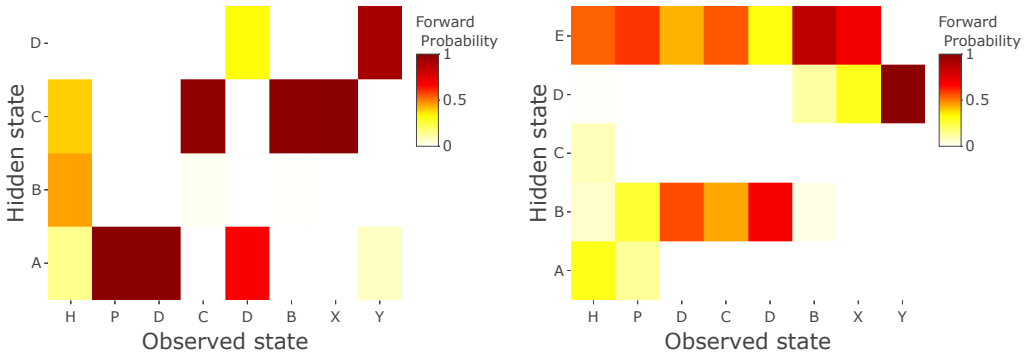
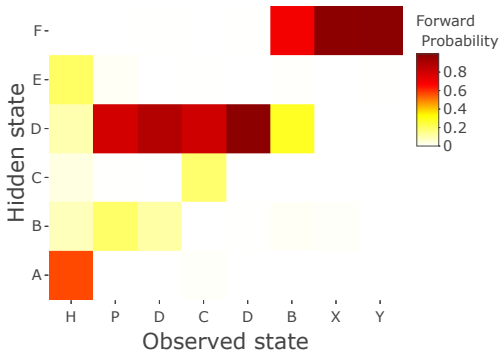


FIGURE 6.14: A graphic to display the model's predictions applied to a devised clickstream sequence.



(a) 4-state HMM.

(b) 5-state HMM.



(c) 6-state HMM.

FIGURE 6.15: The forward probabilities for the hidden states using a variety of HMMs and applied to the devised clickstream sequence.

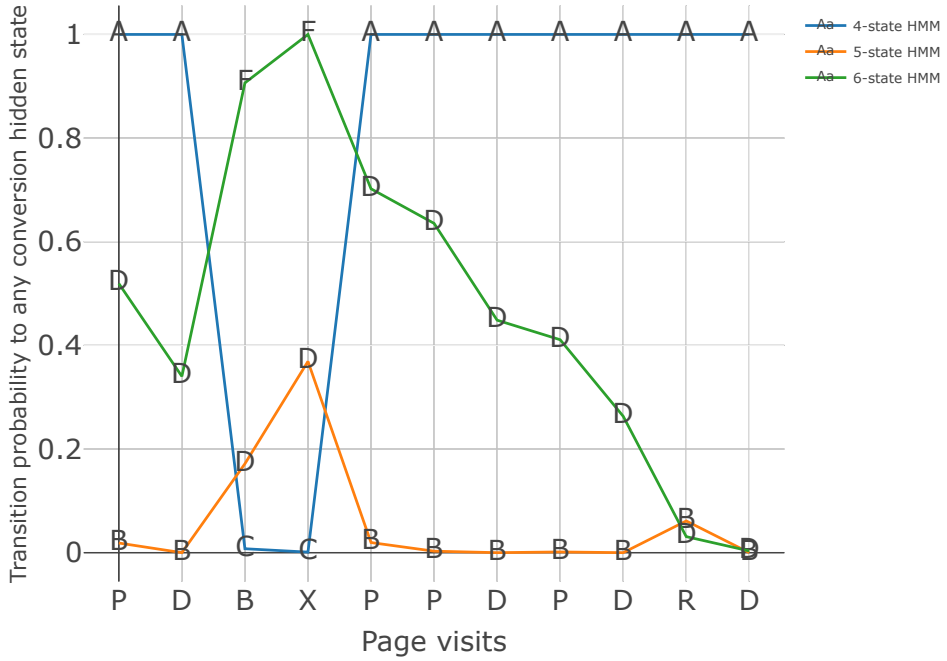
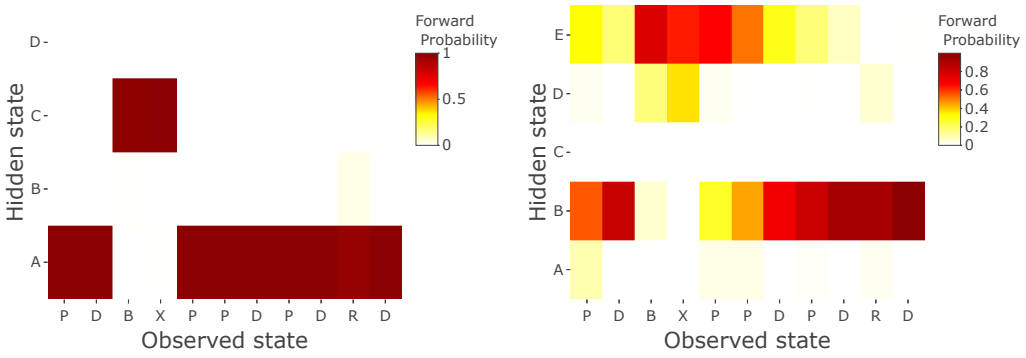
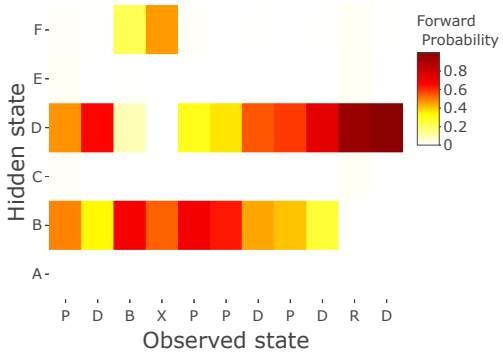


FIGURE 6.16: A graphic to display the model's predictions applied to a clickstream sequence from the training data.



(a) 4-state HMM.

(b) 5-state HMM.



(c) 6-state HMM.

FIGURE 6.17: The forward probabilities for the hidden states using a variety of HMMs and applied to a clickstream sequence from the training data.

6.6 Discussion

This chapter has discussed a range of models that we have explored in this thesis, and applies them to predicting online purchases on an e-commerce web site. We described the numerical summaries of browsing journeys and key covariates that we include in a logistic regression model. We outlined the forward and backward stepwise model selection methods, and apply them. We found that they agreed on the same set of covariates, and we evaluated this model using the residuals and predictions. We found that the model under predicted conversions, but also that the intent score (from Chapter 3) and the number of previous conversions positively influenced the prediction of a conversion.

We described the problems we faced when fitting an HMM to our data, and explored how each of them affected our results. We concluded that the optimal number of hidden states is directly linked to the interpretation of the hidden states. We aim to use as much training data as possible, because we are not time dependent in our scenario, however too much data will break the algorithm implementation we are using. Our brief exploration of the initial parameter estimates concluded that they do not hugely affect the training of an HMM, but crucially they may affect the interpretation of the hidden states. Shuffling the position of the states affects any programmatic/automated exploitation of the state information, as without interpretation providing the context then it's impossible to find the purchasing meta-state. We simulated sequences from an HMM to look for obscure examples of clickstreams, and make quantitative comparisons to real data. Using the observed state frequencies we found that the individual marginal probabilities seemed plausible, however fitting a Markov chain showed that the simulation's transition probabilities did not capture realistic sequence structures. A natural evolution would be to use a mixture of the Markov models, where we respect the page structure but are capable of shifting the transition probabilities based on the background behaviour.

Finally, we created a graphical display to compare the predictive capabilities of all of the models in this chapter. We discovered that the 6-state HMM provided the most interpretable and realistic set of predictions, this was a useful insight into how we could use the meta-states in a real-time prediction environment.

We examined and assessed the potential for practical use and implementation of HMM in terms of what the next page visit will be. This is a deliberately greedy calculation, and in general it would be better to have a longer term view by possibly extending the prediction for the next 2 or 3 steps. Other models with better memory would certainly help here, but fall outside the scope of our work.

Given further time, assessing the predictive capability over multiple sequences could be a useful step to add to the model selection process. A form of accuracy or precision metric for predicting a conversion would be more robust compared to inspecting only single sequences - this is left for future exploration.

Chapter 7

Conclusion

7.1 Thesis summary

Chapter 2 introduced the keyword performance data set, discovered a range of different types of covariates and explored the performance indicators for keyword bidding advertising. We modelled the dependent variable of clicks through a combination of independent variables such as the advert content, keyword and other factors such as time or position of the advert. The match type of the keyword has a significant impact as broad keyword matches provide a higher number of impressions than exact. The categorical quality measures proved unreliable as there are a high number of missing values. We understand that impressions have a crucial impact on clicks, highlighted by a strong positive correlation, however there are many observations with very few impressions leading to a strong bias in the data.

We introduced clickstream data and how browsing journeys can be modelled as a special type of sequential data in a variety of ways. We have learnt that we are dealing with a significant amount of data and we must adopt methods for the Big Data environment. There are a number of variables that relate to the page visit and the profile level (i.e. device information). We calculated numerical summaries to create covariates for exploration. These numerical variables provided intriguing results yet did not fully meet our expectations. We found a heartbeat mechanism that has a significant impact on the distribution of the page visit lengths, which could have an effect when using the time duration variables in a statistical model. Finally, we described the distribution of the number of page visits in a session using a Weibull distribution.

Chapter 3 described a variety of statistical variables that describe a number of behavioural attributes, which we condensed to provide a one-dimensional summary. The application of this score is in targeted advertising and can also be used as a covariate when modelling conversions. We introduced more statistical rigour to the existing algorithm, outlined in Figure 3.1, and we added a new behavioural trait for the depth of visit to get a new perspective on the browsing behaviour intent. This adds to the interpretability of the dimension reduction algorithm, as it adds further insight into the user's journey. We also created a greater number of behavioural labels than the existing algorithm.

Chapter 4 discussed the theory of GLMs, focusing on logistic and Poisson regression. We discussed zero-inflated models and why they would be appropriate in our specific application. We used the structural zeros in the data to aim for a better fitting model. Overall, we fitted 3 models: logistic regression, Poisson regression and zero-inflated Poisson model. We chose a consistent set of covariates, interpreted the coefficient estimates, and assessed the model fit through residuals and a confusion matrix. The models led to similar conclusions in terms of coefficient interpretation

and an under prediction of clicks, which is most likely due to the high number of zeros in the model. There are extreme values that provide skewed residual plots and prediction distributions. To improve these models we hoped the zero-inflation component could help, but the model still under predicts, so we feel that improvements could be made in our variable selection.

Chapter 5 described a range of Markov models that can help us to analyse clickstream data. We began with a first order Markov model, which provides a transition matrix quantifying how users moving around a website. We inspected the three main purchasing states in greater detail, and added an extra state of memory, so we fitted a two-step Markov chain model. We outlined the motivation for fitting a hidden Markov model to clickstream data, to find meta-states of user behaviour. We outlined the forward and Viterbi path algorithms, which we used to find the likelihood of an observable sequence and optimum hidden state sequence respectively. We used the Viterbi training algorithm to fit a 5-state HMM, and possibly unearthed some of the behavioural modes we were searching for.

Chapter 6 outlined the forward and backward stepwise model selection methods, and these methods agreed on a set of covariates for a logistic regression model to predict conversions. The intent score (from Chapter 3) and the number of previous conversions are highly influential covariates in our model. We concluded that the optimal number of hidden states is directly linked to the interpretation of the hidden states. We must be careful with the data size and initial parameter estimates when fitting HMMs, to make the output interpretable and useful. A graphical display of predictions led us to discover that the 6-state HMM provided the most interpretable and realistic set of predictions. This shows how we could use the meta-states in a real-time prediction environment.

7.2 Discussion of the implementation of our models

The algorithm outlined in Chapter 3 has been implemented in Spark, such that we can calculate intent scores for 10^8 profiles and a total of 10^9 rows. The intent scores are updated daily, and determine the stage of the purchasing journey that each user is on for a category of online intent. Marketing specialists are able to use the scores to target the relevant audience for an advertising campaign, and do so with greater accuracy than before. The behavioural attributes are easily explainable to non-specialists. A further extension to the algorithm is to experiment with weightings for each attribute's contribution to the final intent score.

The fitted zero inflated Poisson model in Chapter 4 is useful to indicate important variables in this challenging data set. These key covariates are well known by marketing experts, and don't come as a surprise to them. There are further complexities with bid prices, campaign budgets and time frames that we did not explore in this study. The general concept of a zero inflated model fits the underlying nature of the data accurately in theory, however there could be too many other factors we don't have available in our data set to form a full advertising strategy. Using these models in future campaigns may not be sensible as the keywords for each campaign are picked subjectively, hence the model may need to be updated regularly based on new data. Multiple campaigns would be required to assess if this could be a general model for sponsored search.

Influencing and informing real-time decisions is an exciting and interesting statistical problem, and the commercial impact can be huge. Building a complex statistical model is not enough to solve this problem alone; it must be explainable and

interpretable. Categorising the page visits across a variety of e-commerce sites into page types may pose a challenge, and this is a requirement to define the states in a Markov chain. The Markov chain visualisations provide the sales team with a tool to represent the conversion journey of users on a website. Note that we only explored data relating to purchasing a product on an e-commerce web site. However the concept extends to alternative 'conversions', such as subscriptions or any goal-based metric that can be tracked.

The HMMs in production provide real-time predictions based on a pre-trained model. A drawback is that there must be statistical expertise to interpret the hidden states, and how the predictions of transitioning into these states should be used. For a pre-trained model, this can be done in advance and labelled with clarity and confidence. If the model requires updating, i.e. the HMM parameters are not providing a good fit to new clickstream sequences, then an expert is required. All Markov chain models are fitted to data from a single source web site; this means that the model may encounter issues of out-of-sample predictions, for example on a different e-commerce site. The labelling of the pages may be slightly different, and the purchasing journey may be adjusted depending on the web site structure.

7.3 Future research

Our variable selection process focuses on interpretability, as opposed to reducing the dimensions such that we find the maximal variance. If the use case arises, a combination of a dimension reduction method, such as PCA, in combination with an interpretable algorithm could provide scores with better separation. Also, the score may appear skewed if the underlying type of e-commerce site requires more visits or a longer period of time to make a decision. For example, when purchasing a new item of clothing, the decision process may not take longer than say one week, hence recency may be the most relevant attribute. On the other hand, to purchase a car, the journey may take longer due to the larger amount of money involved. This currently is not considered in our variable selection method, as no distinction is made between the category of the visit. In future work, we could explore if the period of activity is correlated with the category, and consequently the algorithm can adapt to provide accurate scores for the purchasing journey on a wider basis.

Our model provided a baseline prediction for keyword performance; however we recommend that an alternative approach may be necessary to achieve more accurate outcomes. One approach could be to build a model that measures the similarity between the keywords and advert description. For example, creating numerical summaries of the keyword and advertising text using tf-idf would provide a metric that we don't have to rely on Google for [66]. Sponsored search modelling has been explored using a Bayesian approach; in which a probit regression model and prior beliefs on the input covariates predict click through rates [67]. This structure may provide an ability to add expert information into the model, which our model lacks.

We found it challenging to find an appropriate method to compare hidden Markov models; as previously mentioned there is a distance measure that we could explore in future work [64]. We also want to impose an initial structure to aid the training of an HMM, such that the hidden states are stable and to persuade some separation in the behaviour of each state. Hence a Bayesian approach, by determining the structure of the HMM as a Bayesian network could offer a solution [53]. As previously stated using Bayesian inference from a mixture of HMMs could also produce more stable results, while maintaining interpretability [21]. A different approach would be

to apply a hidden semi-Markov model (HSMM) where the underlying observed data exhibits a semi-Markov structure [68]. This structure dictates that the probability of a change in the hidden state is dependent on the amount of time that has elapsed since entry into the current state. The data for the time spent in each observed state is available, and the semi-Markov structure seems sensible in our application, where the longer a user browses a product the more intent they show to buy it. However, we note that statistical inference for HSMMs is more difficult than for HMMs, since algorithms such as the Baum-Welch algorithm are not directly applicable, and must be adapted.

Bibliography

[1] Daniel C Fain and Jan O Pedersen. "Sponsored search: A brief history". In: *Bulletin of the american Society for Information Science and technology* 32.2 (2006), pp. 12–13.

[2] Bernard J Jansen and Tracy Mullen. "Sponsored search: an overview of the concept, history, and technology". In: *International Journal of Electronic Business* 6.2 (2008), pp. 114–131.

[3] Google. *About audience targeting*. 2019. URL: https://support.google.com/google-ads/answer/2497941?hl=en-GB.

[4] Aske Christiansen. *The Ultimate Guide to Facebook Ads Interest Targeting Research (Advanced Methods Exposed)*. 2017. URL: https://adespresso.com/blog/guide-facebook-ads-interest-targeting-research-easy-advanced-methods-exposed/.

[5] Dou Shen et al. *Action-aware intent-based behavior targeting*. US Patent App. 13/017,843. 2012.

[6] Carbon. *Carbon DMP*. 2019. URL: https://carbondmp.com/.

[7] Margaret Rouse. *Definition: Third party cookie*. URL: https://whatis.techtarget.com/definition/third-party-cookie.

[8] EU. *EU data protection rules*. 2018. URL: https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en.

[9] Bernard Marr. *What Is Spark In Big Data?* 2019. URL: https://www.bernardmarr.com/default.asp?contentID=1079.

[10] Ryan Jessop. *GitHub Repository*. URL: https://github.com/ryanjessop.

[11] Carol McDonald. *Spark 101: What Is It, What It Does, and Why It Matters*. 2018. URL: https://mapr.com/blog/spark-101-what-it-what-it-does-and-why-it-matters/.

[12] Apache. *Apache Parquet*. URL: https://parquet.apache.org/.

[13] Wendy W Moe and Peter S Fader. "Capturing evolving visit behavior in clickstream data". In: *Journal of Interactive Marketing* 18.1 (2004), pp. 5–19.

[14] Alice Marques and Orlando Belo. "Discovering Student Web Usage Profiles Using Markov Chains." In: *Electronic Journal of e-Learning* 9.1 (2011), pp. 63–74.

[15] Shirbi Ish-Shalom and Samuel Hansen. *Visualizing clickstream data as discrete-time markov chains*.

[16] Juhnyoung Lee et al. "Visualization and analysis of clickstream data of online stores for understanding web merchandising". In: *Data Mining and Knowledge Discovery* 5.1-2 (2001), pp. 59–84.

[17] Alan L Montgomery et al. "Modeling online browsing and path analysis using clickstream data". In: *Marketing science* 23.4 (2004), pp. 579–595.

[18] Randolph E Bucklin and Catarina Sismeiro. "A model of web site browsing behavior estimated on clickstream data". In: *Journal of marketing research* 40.3 (2003), pp. 249–267.

[19] Courosh Mehanian et al. *Clickstream Purchase Prediction Using Hidden Markov Models*. US Patent App. 14/580,220. 2015.

[20] Steven L Scott and Il-Horn Hann. "A nested hidden markov model for Internet browsing behavior". In: *Marshall School of Business* (2006).

[21] Mohammadamin Jamalzadeh. "Analysis of clickstream data". PhD thesis. Durham University, 2011.

[22] Thore Graepel et al. "Web-Scale Bayesian Click-Through Rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine". In: (2010).

[23] Matthew Richardson, Ewa Dominowska, and Robert Ragno. "Predicting clicks: estimating the click-through rate for new ads". In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 521–530.

[24] Brad Smith. *All About AdWords Quality Score—And How to Improve It Fast*. 2018. URL: https://blog.hootsuite.com/adwords-quality-score/.

[25] Mark Irvine. *Google Ads Benchmarks for YOUR Industry [Updated!]* 2019. URL: https://www.wordstream.com/blog/ws/2016/02/29/google-adwords-industry-benchmarks.

[26] Christine Laubenstein. *What's a Good Click-Through Rate (CTR) for PPC?* 2018. URL: https://www.wordstream.com/blog/ws/2010/04/26/good-click-through-rate.

[27] Justin Brookman et al. "Cross-device tracking: Measurement and disclosures". In: *Proceedings on Privacy Enhancing Technologies* 2017.2 (2017), pp. 133–148.

[28] Marie Laure Delignette-Muller. *fitdistrplus: An R Package for Fitting Distributions*. 2014. URL: https://cran.r-project.org/web/packages/fitdistrplus/vignettes/paper2JSS.pdf.

[29] David N Reshef et al. "Detecting novel associations in large data sets". In: *science* 334.6062 (2011), pp. 1518–1524.

[30] Ian Jolliffe. "Principal Component Analysis". In: *International Encyclopedia of Statistical Science*. Ed. by Miodrag Lovric. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1094–1096. ISBN: 978-3-642-04898-2. DOI: 10.1007/978-3-642-04898-2_455. URL: https://doi.org/10.1007/978-3-642-04898-2_455.

[31] Ian T Jolliffe. "Discarding variables in a principal component analysis. I: Artificial data". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 21.2 (1972), pp. 160–173.

[32] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[33] J. A. Nelder and R. W. M. Wedderburn. "Generalized Linear Models". In: *Journal of the Royal Statistical Society. Series A (General)* 135.3 (1972), p. 370. DOI: 10.2307/2344614.

[34] Federico Vegetti. *Intro to GLM – Day 2: GLM and Maximum Likelihood*. URL: https://federicovegetti.github.io/teaching/ecpr_glm/slides/GLM_slides_Day2.pdf.

[35] Richard Lockhart. *Theory of Generalized Linear Models*. URL: http://people.stat.sfu.ca/~lockhart/richard/350/08_2/lectures/GLMTheory/web.pdf.

[36] Heather Turner. *Introduction to Generalized Linear Models*. URL: http://statmath.wu.ac.at/courses/heather_turner/glmCourse_001.pdf.

[37] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2018. URL: https://www.R-project.org/.

[38] Diane Lambert. "Zero-inflated Poisson regression, with an application to defects in manufacturing". In: *Technometrics* 34.1 (1992), pp. 1–14.

[39] Shaw-Pin Miaou. "The relationship between truck accidents and geometric design of road sections: Poisson versus negative binomial regressions". In: *Accident Analysis & Prevention* 26.4 (1994), pp. 471–482.

[40] Martin Ridout, Clarice GB Demétrio, and John Hinde. "Models for count data with many zeros". In: *Proceedings of the XIXth international biometric conference*. Vol. 19. 19. International Biometric Society Invited Papers. Cape Town, South Africa. 1998, pp. 179–192.

[41] David C Heilbron. "Zero-altered and other regression models for count data with added zeros". In: *Biometrical Journal* 36.5 (1994), pp. 531–547.

[42] Taghi M Khoshgoftaar, Kehan Gao, and Robert M Szabo. "An application of zero-inflated poisson regression for software fault prediction". In: *Proceedings 12th International Symposium on Software Reliability Engineering*. IEEE. 2001, pp. 66–73.

[43] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[44] Andrew Gelman and Yu-Sung Su. *arm: Data Analysis Using Regression and Multilevel/Hierarchical Models*. R package version 1.10-1. 2018. URL: https://CRAN.R-project.org/package=arm.

[45] John Fox and Sanford Weisberg. *An R Companion to Applied Regression*. Third. Thousand Oaks CA: Sage, 2019. URL: https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

[46] Achim Zeileis, Christian Kleiber, and Simon Jackman. "Regression Models for Count Data in R". In: *Journal of Statistical Software* 27.8 (2008). URL: http://www.jstatsoft.org/v27/i08/.

[47] James R Norris. *Markov chains*. 2. Cambridge university press, 1998.

[48] Anders Tolver. "An introduction to Markov chains". In: *University of Copenhagen* (2016).

[49] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[50] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[51] Ewan Birney. "Hidden Markov models in biological sequence analysis". In: *IBM Journal of Research and Development* 45.3.4 (2001), pp. 449–454.

[52] Mark Stamp. "A revealing introduction to hidden Markov models". In: *Department of Computer Science San Jose State University* (2004), pp. 26–56.

[53] Zoubin Ghahramani. "An introduction to hidden Markov models and Bayesian networks". In: *Hidden Markov models: applications in computer vision*. World Scientific, 2001, pp. 9–41.

[54] Richard Durbin et al. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[55] Mark Borodovsky and Svetlana Ekisheva. *Problems and solutions in biological sequence analysis*. Cambridge University Press, 2006.

[56] Jeff A Bilmes et al. "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models". In: *International Computer Science Institute* 4.510 (1998), p. 126.

[57] Przemyslaw Dymarski. *Hidden Markov Models: Theory and Applications*. BoD–Books on Demand, 2011.

[58] Michael Scholz. "R Package clickstream: Analyzing Clickstream Data with Markov Chains". In: *Journal of Statistical Software* 74.4 (2016), pp. 1–17. DOI: 10.18637/jss.v074.i04.

[59] Gabor Csardi and Tamas Nepusz. "The igraph software package for complex network research". In: *InterJournal* Complex Systems (2006), p. 1695. URL: http://igraph.org.

[60] Scientific Software Development Dr. Lin Himmelmann and www.linhi.com. *HMM: HMM - Hidden Markov Models*. R package version 1.0. 2010. URL: https://CRAN.R-project.org/package=HMM.

[61] URL: http://www.biostat.jhsph.edu/~iruczins/teaching/jf/ch10.pdf.

[62] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Fourth. ISBN 0-387-95457-0. New York: Springer, 2002. URL: http://www.stats.ox.ac.uk/pub/MASS4.

[63] URL: https://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/26/lecture-26.pdf.

[64] B-H Juang and Lawrence R Rabiner. "A probabilistic distance measure for hidden Markov models". In: *AT&T technical journal* 64.2 (1985), pp. 391–408.

[65] Giorgio Alfredo Spedicato. "Discrete Time Markov Chains with R". In: *The R Journal* (July 2017). R package version 0.6.9.7. URL: https://journal.r-project.org/archive/2017/RJ-2017-036/index.html.

[66] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc., 1986. ISBN: 0070544840.

[67] Thore Graepel et al. "Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine". In: Omnipress. 2010.

[68] Leonard E Baum and Ted Petrie. "Statistical inference for probabilistic functions of finite state Markov chains". In: *The annals of mathematical statistics* 37.6 (1966), pp. 1554–1563.