Behavioural Analytics and Prediction with eCommerce Transactional Data

Paula McMahon

17185602

MSc in Data Science and Analytics

Maynooth University,

Department of Maths and Statistics

August 2019

Supervisor: Dr. Rafael De Andrade Moral

Acknowledgements

I would like to thank the lecturers and staff of the Department of Mathematics and Statistics at Maynooth University for their support over the past two years. Thanks to Dr. Catherine Hurley for accepting my project proposal. In particular, I would like to extend my gratitude to my project supervisor, Dr. Rafael De Andrade Moral, for his enthusiasm, ideas and suggestions during the project. The data for this project comes from Musgrave Retail Group and I would like to sincerely thank Max Darby, a data scientist, for his help and assistance in answering any queries I had with regards to the data, for the duration of the project.

Table of Contents

Acknowledgements 2

Abstract 6

1 Introduction 7

2 Materials and Methods 7

2.1 The Data 7

2.2 Statistical Methods 11

2.2.1 Data manipulation and visualisation 11

2.2.2 The issue of collinearity 11

2.2.3 Creating a new time variable – DaysSinceLastShop 13

2.2.4 Aggregating the data by customer 14

2.2.5 Exploring customer behaviour using higher-order moments 14

2.3 Aggregated Data Methodologies 15

2.3.1 Confusion matrices and test error rates 15

2.3.2 Classification Trees 16

2.3.3 Bagging 17

2.3.4 Random Forests 17

2.3.5 Boosting 17

2.3.6 Support Vector Machines 18

2.3.7 Linear Discriminant Analysis (LDA) 18

2.3.8 Quadratic Discriminant Analysis (QDA) 19

2.3.9 K Nearest Neighbours (KNN) 19

2.3.10 Principal Components Analysis (PCA) 20

2.3.11 Logistic Regression 20

2.4 Full Data Methodologies 21

2.4.1 Generalised Linear Mixed Model (GLMM) analysis 21

2.5 Survey Data Methodologies 23

2.5.1 Sentiment Analysis 23

2.5.2 Word Cloud 23
3 Results 24
3.1 Aggregated Data Results 24
3.1.1 Summary 24
3.1.2 Classification Trees 25
3.1.3 Bagging 28
3.1.4 Random Forests 28
3.1.5 Boosting 29
3.1.6 Support Vector Machines (SVM) 29
3.1.7 Linear Discriminant Analysis (LDA) 29
3.1.8 Quadratic Discriminant Analysis (QDA) 30
3.1.9 K-Nearest Neighbours (KNN) 30
3.1.10 Principal Component Analysis (PCA) 31
3.1.11 Logistic Regression 33
3.2 Full Data Results 36
3.2.1 Patterns of shopping behaviour 36
3.2.2 Current customers – decreasing spending analysis 38
3.2.3 Generalized Linear Mixed Model (GLMM) 40
3.2.4 GLMM assumption violation 47
3.3 Survey Data Results 49
3.3.1 Word Clouds 49
3.3.2 Customer Ratings 51
4 Discussion 52
4.1 What is the “best” model? 52
4.2 April – the most likely month for ‘Churn’ 52
4.3 Churn Threshold Analysis 53
4.4 Future Work 53
5 Conclusions 54

Bibliography 54

Appendices 57

Abstract

In business sectors where there is a lot of consumer competition and choice, knowing customer attrition/churn rates can be very beneficial. With data from online shopping transactions, this project uses Machine Learning (ML) techniques to analyse the features that best predict if a shopper may cease shopping. The data comes from an environment concerned with FMCG (Fast-Moving Consumer Goods). Firstly, the data is explored and then machine learning methods are used on an aggregated, i.e. per customer, basis. Logistic regression is found to be the most successful method. The analysis is then expanded, using the full dataset, to a generalised linear mixed model (GLMM) with random and fixed effects. The approach to obtaining the best performing model is documented. Error rates are provided for this model, for both existing customers, and a new customer dataset, in predicting whether they will become dormant. Word clouds are also generated to express sentiment, positive or negative, emanating from a dataset of customer surveys.

1 Introduction

E-Commerce, also known as electronic commerce or internet commerce, refers to the buying and selling of goods or services using the internet, and the transfer of money and data to execute these transactions (Shopify, 2019). The benefit of applying Machine Learning (ML) techniques in the eCommerce setting are that we can develop models that are transferable and reusable with similar data from the same sector. This allows us to explore the possibilities of prediction by looking at customer behaviour. The objectives of this project are to analyse the behaviour of a set of current and dormant customers by applying a suite of ML techniques to the data. From the model or models deemed most appropriate, we can indicate the most significant features/variables that give the best predictive ability. An additional objective is to extract sentiment from the comments contained in customer surveys or to see if a negative review eventually led to a customer churning.

Some literature was reviewed in the area of customer churn to help formulate a strategy for the project. (Li, 2017) outlines that customer churn occurs when customers or subscribers stop doing business with a company or service, also known as customer attrition. It is also referred to as loss of clients or customers. Li uses Logistic Regression, Decision Tree, and Random Forests to predict customer churn in the telecoms sector. It is an industry in which churn rates are particularly useful, because most customers have multiple options from which to choose within a geographic location. Prior to using the Machine Learning methods, Li outlines the preparatory tasks of data pre-processing and exploratory data analysis.

(Buckinx and Van den Poel, 2004) discuss churn analysis in the retail setting. They say that a non-contractual setting (as opposed to broadband or utility (gas or electricity) suppliers where there is a contract), suffers from the problem that customers are free to continuously change their purchase behaviour without informing the company about it. More specifically, in a grocery retail environment, competition is fierce, and customers have a wide range of alternatives. The authors of the paper develop a model to predict partial defection of customers, focussing on a decreasing purchasing pattern of behaviourally loyal customers.

2 Materials and Methods

2.1 The Data

Musgrave - Ireland’s largest food retailer, have provided the data for this project, under a non-disclosure agreement. The dataset comprises of online SuperValu shopping transactions. The data bank for online, as opposed to in-store data, is more comprehensive and allows for a fuller picture of the customer to emerge (as they must be logged in to use the service). The data was provided in two Excel files – a set for dormant customers and a set for active customers. The dormant data has 5,888 rows for 1000 randomly sampled customers spanning dates from 1st June 2016 to 10th May 2018. The current-shopper data has 25,089 rows for 905 randomly sampled customers spanning dates from 1st June 2017 to 31st May 2018. The data was received in early February 2019. It is backdated by about 6 months to make it less commercially sensitive. Table 1 below shows a description of the data.

Table 1 Dataset variable descriptions, showing type and unit

Variable Name	Type	Unit	Description
OrderRef	Categorical	-	The order reference number for the transaction
sequenceID	Categorical	-	The customer number. The real customer ID's were replaced with sequential numbers for the purposes of anonymisation
StoreID	Categorical	-	The SuperValu store location number
ExpectedGoodsCharge	Numerical	€	The value of the goods ordered when placing the order online
ExpectedFulfillmentCharge	Numerical	€	The delivery charge for the order. Values can range from €0 (suggesting an off-peak delivery time) to €8 (a peak delivery slot)
SlotStartDate	Date	DD/MM/YYYY	The date the online order was placed

Variable Name	Type	Unit	Description
OverallSubstitutionPolicy	Categorical	NONE, ALL, SOME	A customer can choose a substitution policy when shopping online, in the event the chosen products are out of stock
ActualCharge	Numerical	€	The total value of the order after out-of-stock, substitutions and delivery charge are factored in.
TotalOrderLines	Numerical	-	This variable is present in the current shopper dataset but not the dormant one. It was not considered for further analysis as it would contain missing data when the two datasets are merged.
TotalOrderItems	Numerical	-	The total number of items ordered.
TotalPickedLines	Numerical	-	This tells us how many different product lines were picked during the shopping transaction. It may help the retailer to know who chooses a broad range of lines and who shops specific lines only.
TotalItemsApprovedPicks	Numerical	-	This variable is 100% correlated with the aforementioned TotalPickedLines

Variable Name	Type	Unit	Description
TotalQtySubbed	Numerical	-	The number of items substituted
TotalQtyOOS	Numerical	-	The number of items out of stock
TotalQtyOrdered	Numerical	-	The total number of items ordered. This variable is 99% correlated with the aforementioned TotalOrderItems
TotalPickTimeSeconds	Numerical	Seconds	The amount of time taken for the assigned picker to complete the shopping order
AvailabilityPreSubPercentage	Numerical	%	The percentage of the order that was in stock, before any substitutions
AvailabilityPostSubPercentage	Numerical	%	The percentage of the order that was in stock, after any substitutions. These Pre and Post variables are a metric used for stock levels. A large gap between their values would indicate a stock issue.
PercentageOutOfStocks	Numerical	%	The percentage of the total items ordered that were out of stock
PercentageSubstitutions	Numerical	%	The percentage of the total items ordered that were substituted

Variable Name	Type	Unit	Description
ValueOfSubstitutions	Numerical	€	Monetary value of the substituted products
ValueOfOutOfStocks	Numerical	€	Monetary value of the out-of-stock products
RelatedCallsCount	Numerical	-	The number of calls made/received relating to this order.

A third dataset which contains customer surveys was also provided. The survey data has 1,125 rows for 571 customers who were sent surveys about their online shopping experience. The customer ID numbers are the same across the datasets. It is mainly categorical data, but some fields contain typed comments made by the customer. The objective here is to extract some sentiment from this text.

The statistical software R is used for all modelling.

2.2 Statistical Methods

2.2.1 Data manipulation and visualisation

The dormant and current datasets were merged as they share the same variables. For customers 1 – 1000 i.e. the leavers, a variable Churn was added and assigned a value of 1. For customers 1001 – 1905 i.e. those that currently still shop, the Churn variable was added and assigned a value of 0. This variable, Churn, is the response variable that we are interested in predicting. It will be the response variable in all machine learning methods used in the entire project. For all variables in Table 1, R code is used to create boxplots which allows us to see the minimum and maximum values and this could flag anomalies or outliers. Missing data is not an issue in this project.

2.2.2 The issue of collinearity

The `corrplot` function in R (Wei and Simko, 2017) was used to check for issues of multicollinearity. High levels of correlation between independent variables can reduce the precision of the model coefficient estimates. The estimates become very sensitive to small changes in the model. Figure 1 shows the Pearson correlation coefficients between all of the numerical variables.

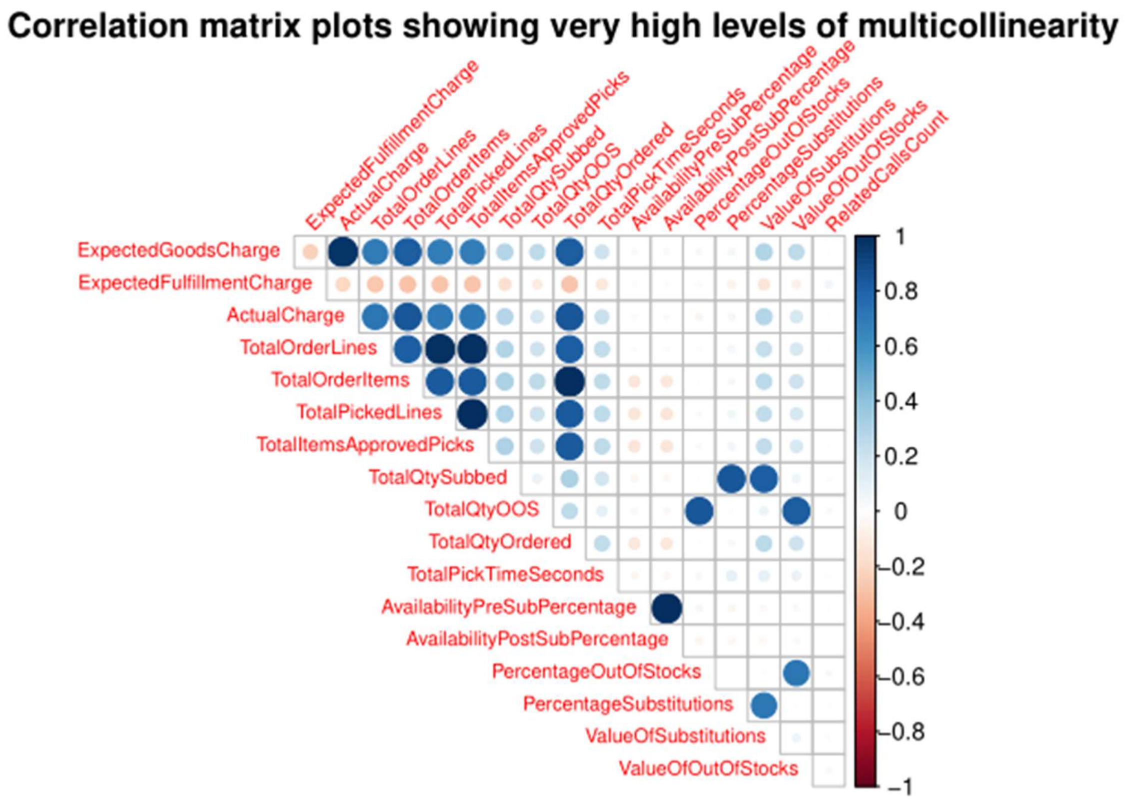


Figure 1 Correlation matrix among numerical variables

Large, dark blue circles indicate very high levels of collinearity (80% - 100%) between two variables. Based on this plot, it was decided to exclude some variables from the analysis:

- OrderRef is a unique reference number for a transaction and is not of any analytic value.
- ExpectedGoodsCharge (the value of the goods at the time the order is placed) is 98% correlated with ActualCharge (the amount charged to the customer after substitutions and delivery charge are factored in). It is more useful to focus on the exact charge to the customer, so ExpectedGoodsCharge was dropped from further analysis.
- TotalOrderLines is 98% correlated with TotalPickedLines and it is an extra variable that features in the current-shopper dataset but not in the dormant. TotalOrderLines was dropped from further analysis as when the datasets become merged, it would create a missing data problem.
- TotalItemsApprovedPicks (how many different product lines were picked during the shopping transaction) is 100% correlated with TotalPickedLines so TotalItemsApprovedPicks was dropped from further analysis. TotalPickedLines is a more intuitive variable name.

- TotalQtyOrdered is 99.9% correlated with TotalOrderItems. There is no obvious reason to drop one over the other, so TotalQtyOrdered was dropped from further analysis.
- AvailabilityPostSubPercentage is 99.9% correlated with AvailabilityPreSubPercentage. These Pre and Post variables are a metric used for stock levels and there is no reason to keep one over the other, so AvailabilityPostSubPercentage was dropped from further analysis.

The correlation matrix plot was regenerated and is shown in Figure 2.

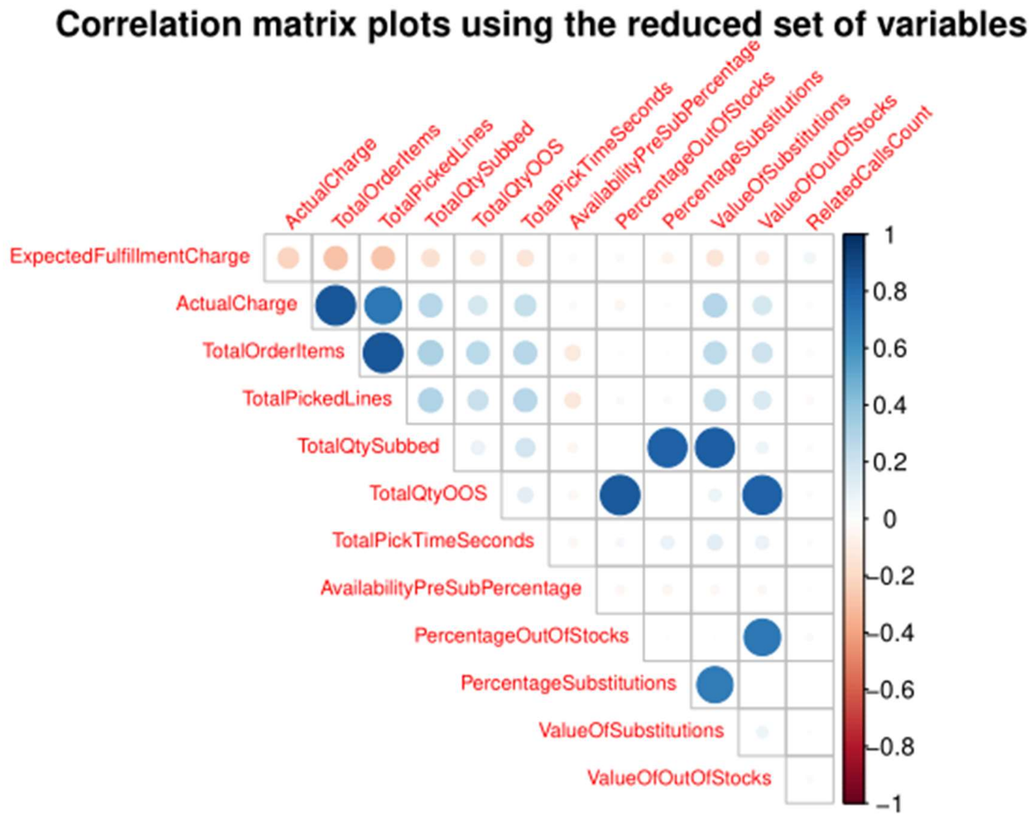


Figure 2 Correlation matrix for the reduced set of variables

Although some correlation figures are in excess of 80%, the plot looks better and a decision was made not to exclude any more variables at this point and the project proceeded using the group of variables in Figure 2.

2.2.3 Creating a new time variable – DaysSinceLastShop

The data has a transaction date variable (SlotStartDate) but this is not of much benefit in its current form. A variable was created, called DaysSinceLastShop, which would count the number of days between a shopping date and some baseline date. The data ends on the 31st May 2018, so the date of 1st June 2018 was used as the baseline. A `for` loop was used in R to parse each row and the `difftime` function (in base R) was used to subtract each SlotStartDate from the base date and calculate the answer in days. Discussion around the use of this variable features in sections 2.2.5, 2.3.11, 3.1.11, 3.2.1 and 3.2.2.

2.2.4 Aggregating the data by customer

The main goal is to analyse the full dataset. We begin, however, by aggregating the full data down to a reduced set using the “customer” variable (sequenceID). Analysis can then commence using a suite of machine learning techniques. Some of these techniques require an assumption of independence between observations and the full data set, with multiple transactions from some customers, does not satisfy this assumption. The aggregation step calculates the mean of all variables and reduces the dataset from 30977 rows to 1905 rows i.e. one observation per customer.

2.2.5 Exploring customer behaviour using higher-order moments

The first four central moments of a distribution are:

- **mean**: the mean is a measure of central tendency of a distribution. It can be estimated by

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

where x_i = the ith observation value and n = the number of items in the sample

- **variance**: the variance is a measure of the width, dispersion or variability of the distribution. The standard deviation is the square root of the variance. The variance can be estimated by the formula below. For very large values of n, the sample variance formula is approximately equal to the population variance formula and so n – 1 could be replaced by n.

$$s^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})}{n - 1}$$

- **skewness:** skewness is a measure of symmetry - positive or negative. If skewness is positive, the mean is bigger than the median and the distribution has a large tail of high values. If skewness is negative, the mean is smaller than the median and the distribution has a large tail of small values. For very large values of n, the sample skewness formula is approximately equal to the population skewness formula.

$$S = \sqrt{n}\, \frac{\sum_{i=1}^{n} (x_i - \bar{x})^3}{\left(\sum_{i=1}^{n} (x_i - \bar{x})^2\right)^{\frac{3}{2}}}$$

- **kurtosis:** kurtosis is a measure of the shape (how peaked the distribution is) i.e. tall or flat. Positive kurtosis indicates a thin pointed distribution. Negative kurtosis indicates a broad flat distribution. For very large values of n, the sample kurtosis formula is approximately to the population kurtosis formula.

$$K = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^4}{s^4}$$

where s is the sample standard deviation.

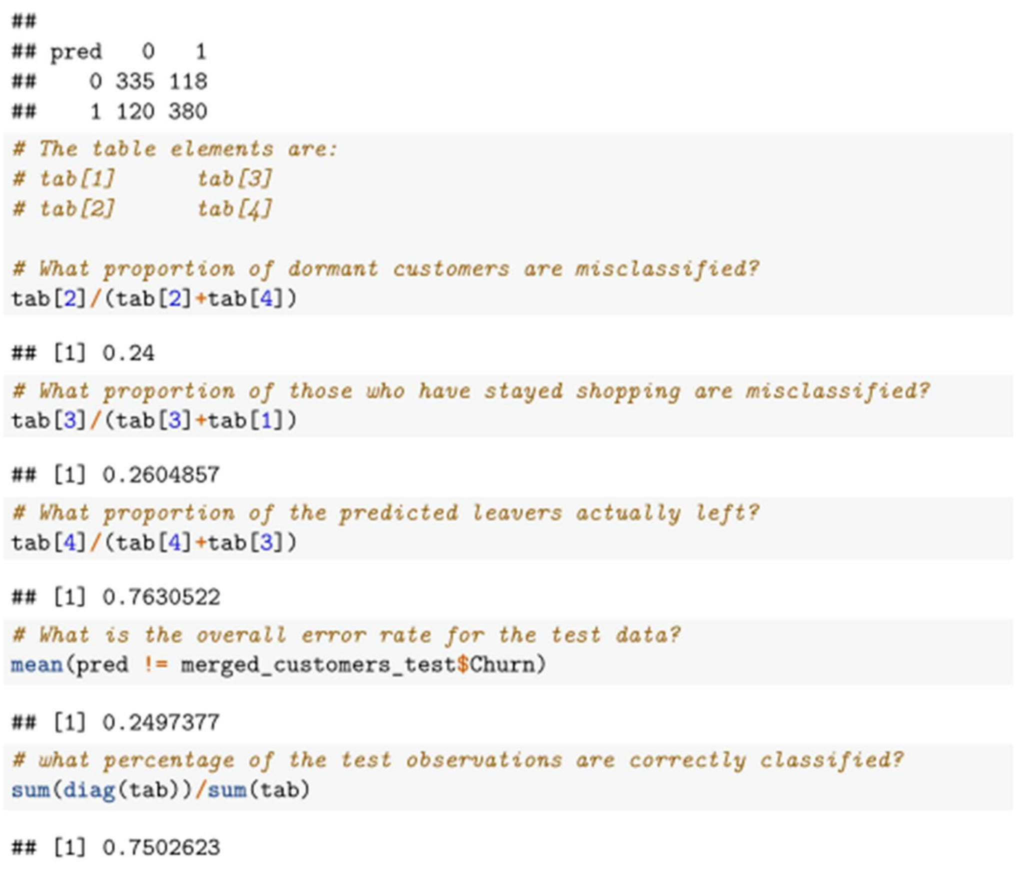
For each customer, it would be useful to calculate these moments for the DaysSinceLastShop variable as they not only show when they shopped but how they shopped e.g. do they start off slow then shop more frequently, do they shop a little then ease off, then perhaps shop some more. However, there was an issue when generating the moments. Out of the 1,905 customers, there are 426 (22%) that have shopped only once. Their standard deviation value will be classified by R as NA (Not Available) as you cannot calculate a standard deviation of one number. Their skewness and kurtosis values will be classified as NaN (Not A Number) e.g. you cannot divide 0 by 0. To use these moments as predictors will mean running a conditional machine learning technique excluding shoppers who shopped just once. The results section 3.1.11 will outline the results of this conditional analysis.

2.3 Aggregated Data Methodologies

For all of the machine learning techniques described in this section, the aggregated data (1905 rows, 1 per customer) was used.

2.3.1 Confusion matrices and test error rates

For each ML method, a test error rate is reported. Misclassification rates can also be determined. The `predict` function in R (R Core Team, 2018) along with the `type='class'` or `type='response'` option creates a vector of predicted classes using the model fit and the test dataset. A 2x2 table or confusion matrix is generated by counting the 1's and 0's in the predicted vector and the test dataset response variable, Churn. This matrix compares the predictions to the true default statuses for the training observations in the data set. Elements on the diagonal of the matrix represent customers whose default statuses were correctly predicted, while off-diagonal elements represent customers that were misclassified (James et al., 2013). The code chunk in Figure 3 below shows an example:



```
##
## pred   0   1
##     0 335 118
##     1 120 380
```

```
# The table elements are:
# tab[1]       tab[3]
# tab[2]       tab[4]

# What proportion of dormant customers are misclassified?
tab[2]/(tab[2]+tab[4])
```

```
## [1] 0.24
```

```
# What proportion of those who have stayed shopping are misclassified?
tab[3]/(tab[3]+tab[1])
```

```
## [1] 0.2604857
```

```
# What proportion of the predicted leavers actually left?
tab[4]/(tab[4]+tab[3])
```

```
## [1] 0.7630522
```

```
# What is the overall error rate for the test data?
mean(pred != merged_customers_test$Churn)
```

```
## [1] 0.2497377
```

```
# what percentage of the test observations are correctly classified?
sum(diag(tab))/sum(tab)
```

```
## [1] 0.7502623
```

Figure 3 R code to obtain a confusion matrix and find error and misclassification rates

Figure 3 shows how you can extract the various matrix elements to work out those who were correctly classified or misclassified. 455 i.e. (335+120) customers were predicted to remain but only 335 actually remained. This means 120 were misclassified. 498 i.e. (118+380) were predicted to leave but only 380 actually left, leaving 118 misclassified. The overall error rate compares the predicted vector of responses, `pred`, with the response vector, Churn, from the test dataset and calculates the average when the comparisons are not equal. For this example, a test error rate of 25% has been reported i.e. 1 in 4 cases will not be correctly classified.

2.3.2 Classification Trees

A classification tree is used to predict a categorical response, in this case, Churn=0 (No) or Churn=1 (Yes). For each observation, a prediction is made that it belongs to the most commonly occurring class of training observations in the region to which it belongs. In interpreting the results of a classification tree, we are often interested not only in the class prediction corresponding to a particular terminal node region, but also in the class proportions among the training observations that fall into that region (James et al., 2013). Classification error rate is used as the criterion for making the binary splits. This error rate is simply the fraction of the training observations in that region that do not belong to the most common class. The `rpart` function (Therneau and Atkinson, 2018) from the library of the same name in R was used to generate the classification tree. The CART (Classification and Regression Trees) algorithm developed by Leo Breiman provides a foundation for algorithms like bagged decision trees, random forests and boosted decision trees. The `rpart` function uses complexity as a way to evaluate the quality of a potential split when growing the tree, or more specifically, it is the amount by which splitting that node would decrease the relative error.

2.3.3 Bagging

Taking repeated samples from the training dataset is called bootstrapping. Bagging is a method where 'B' regression trees are constructed using 'B' bootstrapped training sets and the predictions are averaged i.e. all the trees are combined to create a single predictive model. Bagging is a special case of random forests where all predictors are considered for each split of the tree. This is equivalent to setting the `mtry` argument in function `randomForest` to the number of predictors, p. For the project data, we have p = 14 predictors.

2.3.4 Random Forests

Random forests (Liaw and Wiener, 2002) provide an improvement over bagged trees by way of a random small tweak that decorrelates the trees. Just like in bagging, we build several decision trees on bootstrapped training samples. However, when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. Growing a random forest proceeds in the same way as bagging above, except that we use a smaller value of the `mtry` argument. The data has 14 predictors, so we set mtry = $\sqrt{14} \approx 3$

2.3.5 Boosting

Boosting is similar to bagging except trees are grown sequentially. Each tree is grown using information from previously grown trees. Boosting does not involve bootstrap sampling. Instead, each tree is fit on a modified version of the original dataset. The boosting approach learns slowly. In Boosting (unlike Bagging), the construction of each tree depends strongly on the trees that have already been grown. The `gbm` function in R (Greenwell et al., 2019) is used and takes three arguments:

1. *n.trees* = the number of trees. A value of 6,000 was used but this could be chosen by cross validation.
2. *λ (lambda)* = shrinkage parameter, usually a small positive number, the default is 0.001
3. *interaction.depth* = depth or number of splits in the tree e.g. if d = 1, the trees are stumps. 4 was used to fit the model.

2.3.6 Support Vector Machines

The support vector machine (SVM) methodology, from R library `e1071` (Meyer et al., 2019) is an approach for classification that was developed in the computer science community in the 1990s and that has grown in popularity since then. The support vector machine is a generalisation of a simple and intuitive classifier called the maximal margin classifier. However, this classifier unfortunately cannot be applied to most data sets, since it requires that the classes be separable by a linear boundary. The support vector classifier is an extension of the maximal margin classifier that can be applied in a broader range of cases. The SVM is a further extension of the support vector classifier in order to accommodate non-linear class boundaries. SVMs use a number of parameters. A tuning wrapper function in R, `tune.svm`, chooses some of these values by cross-validation.

1. *kernel*: options are linear, polynomial, radial or sigmoid; This dictates the shape of the classification boundary.
2. *degree*: values 1 to 3; parameter needed when using kernel of type polynomial.
3. *cost*: values .01, .1, 1, 5, 10; cost of constraint violation.

2.3.7 Linear Discriminant Analysis (LDA)

(James et al., 2013) informs us that LDA was developed in the mid-1930s and it was derived to predict qualitative outcomes e.g. “Yes” or “No”, “1” or “0”, so it is a suitable candidate for this shopping transaction data. Let Y equal our response variable, Churn, and let X represent our input variables. The LDA method approximates the Bayes classifier. Bayes classifier is a simple classifier that assigns each observation to the most likely class, given its predictor values. In other words, we should simply assign a test observation with predictor vector x to the class k for which

$$p_k(X) = \Pr(Y = k | X = x)$$

is the largest. This is a conditional probability. It is the probability that Y = k, given the observed predictor vector x. In a two-class problem where there are only two possible response values, (Churn=1 or Churn=0), the Bayes classifier corresponds to predicting the first class if $\Pr(Y = 1 | X = x) >$

0.5, and the second class otherwise. It will always classify an observation to the class for which $p_k(X)$ is largest. In a real-life situation, we are not able to calculate the Bayes classifier, so we approximate it with the LDA classifier. The LDA classifier assumes that the observations in the kth class are drawn from a multivariate Gaussian distribution $N(\mu_k, \Sigma)$, where μ_k is a class-specific mean vector, and Σ is a covariance matrix that is common to all k classes. We must estimate the parameters μ_1, μ_2, π_1, π_2, and Σ. $\hat{\mu}_1$and $\hat{\mu}_2$ are estimated by averaging all of the training observations for class 1 and class 0. $\hat{\pi}_1$ and $\hat{\pi}_2$ are estimated using the proportions of the training observations that belong to each class. The estimate of Σ, more commonly written as $\hat{\sigma}^2$, is calculated as a weighted average of the sample variances for each of the two classes. The function `lda` from the R library `MASS` (Venables and Ripley, 2002) is used to run linear discriminant analysis on the data.

2.3.8 Quadratic Discriminant Analysis (QDA)

According to (James et al., 2013), QDA offers an alternative to LDA (and its covariance matrix that is common to all k classes). QDA assumes each class has its own covariance matrix. That is, it assumes that an observation from the kth class is of the form $X \sim N(\mu_k, \Sigma_k)$, where Σ_k is a covariance matrix for the kth class. This requires estimation of extra parameters. QDA provides more flexible (non-linear) decision boundaries than LDA. This means that LDA may have lower variance but if the assumption that the k classes share a common covariance matrix is not correct, then LDA can suffer from high bias. The function `qda` from the R library `MASS` (Venables and Ripley, 2002) is used to run quadratic discriminant analysis on the data.

2.3.9 K Nearest Neighbours (KNN)

(James et al., 2013) outlines that we would like to predict qualitative responses using the Bayes classifier. However, for real data, we do not know the conditional distribution of Y given X, and so computing the Bayes classifier is impossible. Many approaches attempt to estimate the conditional distribution of Y given X, and then classify a given observation to the class with highest estimated probability. One such method is the K-nearest neighbours (KNN) classifier. Given a positive integer K, and a test observation x_0, the KNN classifier first identifies the K points in the training data that are closest to x_0, represented by N_0. It then estimates the conditional probability for class j as the fraction of points in N_0 whose response values equal j:

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$$

where $I(y_i = j)$ is an indicator variable that equals 1 if $y_i = j$ and 0 if $y_i \neq j$. Finally, KNN applies Bayes rule and classifies the test observation x_0 to the class with the largest probability. The function `knn` from the R library `class` (Venables and Ripley, 2002) is used to run K-Nearest Neighbours analysis on the data.

2.3.10 Principal Components Analysis (PCA)

PCA is an unsupervised learning method where we have p predictors measured on n observations. Even though our data has a response variable, Churn, we will not use this variable in the analysis. (James et al., 2013) advise us that when using PCA, the goal is to discover interesting things about the measurements on X_1, X_2, ... ,X_p. Is there an informative way to visualize the data? Are there subgroups that can be discovered among the variables or the observations? Unsupervised learning is often challenging. The interpretation tends to be more subjective, and there is no simple goal for the analysis, such as the prediction of a response. To run PCA in R, use the `prcomp` function (R Core Team, 2018) and use the option `scale=TRUE`. It is necessary to scale the data before running PCA because it is a variance maximising exercise and if scaling is not done, then variables with larger units will dominate the analysis. PCA calculates a new projection of the data set with the new axes based on the standard deviation of the variables.

2.3.11 Logistic Regression

(James et al., 2013) discuss the fact that if we try to use a linear regression model to predict, in our scenario, the probability of “churn” given p predictors,

$$p(X_p) = \Pr(\text{Churn} = 1 \mid X_p)$$

we may encounter the problem where we get a negative probability of a customer churning or probabilities greater than 1. True probabilities must fall between 0 and 1. To circumvent this problem, we must model $p(X_p)$ using a function that gives outputs between 0 and 1 for all values of X_p. In logistic regression, we use a logistic function.

$$p(X_p) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p}}$$

or

$$\log\left(\frac{p(X_p)}{1-p(X_p)}\right) = \beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_p X_p$$

The model is fit using a maximum likelihood estimation method (MLE). MLE is a method that determines values for the parameters of a model. The parameter values are found such that they maximise the likelihood that the process described by the model, produced the data that were actually observed. The log of the ratio of the probabilities in the equation above is called the logit function and forms the basis for logistic regression.

In the Statistical Methods section 2.2.5, it was asserted that 22% of the dataset or 426 customers shopped only once. For the other 78% i.e. those that shopped more than once, a logistic regression can be run using three moments of the variable DaysSinceLastShop i.e. variance, skewness and kurtosis as predictors. It will be argued in section 3.2.1 that the mean value of DaysSinceLastShop should not be used as a predictor because the dormant data spans a longer time scale than the current data. However, the other moments can be used because they ignore completely when the customers shopped, but they show instead how the shopping was done (a lot at first then less, same intensity all over, small amounts then increasing etc.). So, this is a "conditional" analysis i.e. given that a customer shopped more than once, it's possible to explore the features that best predict that they might leave. All logistic regression models will be fit using the `glm` function (R Core Team, 2018) from base R.

2.4 Full Data Methodologies

2.4.1 Generalised Linear Mixed Model (GLMM) analysis

The data is longitudinal i.e. shopping transactions are made over time by the same customer. (Molenberghs and Verbeke, 2000) tell us that the correlation between the repeated categorical measurements is induced by unobserved random effects. The categorical longitudinal measurements of a subject are correlated because all of them share the same unobserved random effect (with the assumption of conditional independence). Using a mixed model is appropriate for this data and will allow exploration of the changes in the relationship of the response variable to the customers characteristics. In our model, there are two random effects – one for the customer (sequenceID) and one for the store (StoreID). We can declare them as random effects because the customers and stores are considered to be a random sample of all possible customers and stores, i.e. we are interested in the variation between customers or the variation between stores. The random effects are crossed by definition, meaning that a customer can shop online in other stores if they so wish (see Figure 4). However, in the dataset analysed for this project, most customers operated in a nested fashion, always ordering from their local store.

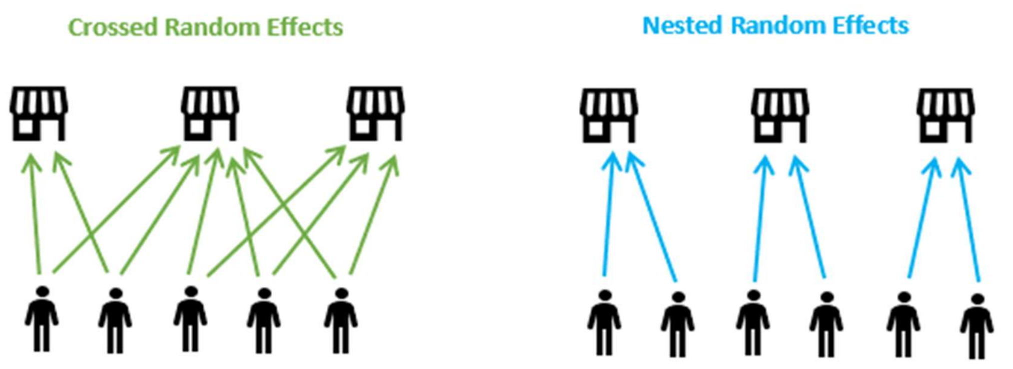


Figure 4 An illustration of customers and stores, showing crossed vs nested random effects

Some literature was reviewed for the purposes of creating the best mixed model for the data. (Cheng et al., 2009) recommend five steps for the model selection process.

1. Specify the maximum model to be considered.
2. Specify a criterion of goodness of fit of a model.
3. Specify a predictor selection strategy.
4. Conduct the analysis.
5. Evaluate the reliability of the model chosen.

(Molenberghs and Verbeke, 2000) recommend that, in terms of model building: start with an elaborate specification of the fixed-effects structure that contains all the variables we wish to study, and potential nonlinear and interactions terms. Then build-up the random-effects structure, starting from random intercepts. In each step, perform likelihood ratio tests to see whether including the additional random effect improves the fit of the model. Having chosen the random-effects structure, we return to the fixed effects and check whether the specification can be simplified.
Bearing that literature in mind, the mixed model equation for the maximal model is given below:

$$\log\left(\frac{\Pr(Churn)}{1-\Pr(Churn)}\right) = \beta_0 + \beta_1\text{*ExpectedFulfillmentCharge} + \beta_2\text{*ActualCharge} + \beta_3\text{*TotalOrderItems} + \beta_4\text{*TotalPickedLines} + \beta_5\text{*TotalQtySubbed} + \beta_6\text{*TotalQtyOOS} + \beta_7\text{*TotalPickTimeSeconds} + \beta_8\text{*AvailabilityPreSubPercentage} + \beta_9\text{*PercentageOutOfStocks} + \beta_{10}\text{*PercentageSubstitutions} + \beta_{11}\text{*ValueOfSubstitutions} + \beta_{12}\text{*ValueOfOutOfStocks} + \beta_{13}\text{*RelatedCallsCount} + \beta_{14}\text{*OverallSubstitutionPolicyNONE} + \beta_{15}\text{*OverallSubstitutionPolicySOME} + \beta_{16}\text{*Month2} + \beta_{17}\text{*Month3} + \beta_{18}\text{*Month4} + \beta_{19}\text{*Month5} + \beta_{20}\text{*Month6} + \beta_{21}\text{*Month7} + \beta_{22}\text{*Month8} + \beta_{23}\text{*Month9} + \beta_{24}\text{*Month10} + \beta_{25}\text{*Month11} + \beta_{26}\text{*Month12} + b1_j + b2_k$$

where:

$b1_j \sim N(0, \sigma^2_{\text{StoreID}})$ and

$b2_k \sim N(0, \sigma^2_{\text{sequenceID}})$

The parameters for the model are $\beta_0, \ldots, \beta_{26}$, $\sigma^2_{\text{StoreID}}$ and $\sigma^2_{\text{sequenceID}}$

All 30977 rows of data are now being used. The training and test data are split 50/50. Note, from the model equation that a new predictor, Month, has been created. The month portion of each transaction has been extracted from the original variable SlotStartDate. The `glmer` function from the R library `lme4` (Bates et al., 2015) is used. The model has convergence issues using the default setting of `nAGQ=1`. This option defines the number of points per axis for evaluating the adaptive Gauss-Hermite approximation to the log-likelihood. Using the default of 1 corresponds to the Laplace approximation. Setting this option to 0 is a further simplification of this approximation. `glmer` documentation tells us that using 0 points yields a faster but less exact form of parameter estimation for GLMMs by optimizing the random effects and the fixed-effects coefficients in the penalized iteratively reweighted least squares step. Using nAGQ=0 solves the non-convergence problem and dramatically reduces the fit time for the model.

2.5 Survey Data Methodologies

2.5.1 Sentiment Analysis

The survey data contains many categorical variables. Customers who complete the survey are asked to give a rating (from 1 to 10) of their online shopping experience covering how easy it is to: search for products, find a suitable delivery slot, find special offers etc. They are also asked if they are happy with the freshness of the produce and with any substitutions made. An analysis of the entire data set is beyond the scope of this thesis. One area of interest is to extract some sentiment from the text field where customers are asked if, based on their online shopping experience, they would recommend SuperValu to a friend or colleague. Customer (sequenceID) numbers are consistent with the other datasets. This will allow us to see if a customer gave a mediocre or scathing review and stayed shopping. Or, if they left, how long after the review, did they leave.

2.5.2 Word Cloud

Two-word clouds were produced in R – one uses all of the survey comments and the other is a subset of the customers that rated their experience 5 or less. Word cloud generation uses functions from the libraries `tm` (Feinerer and Hornik, 2018), `SnowballC` (Bouchet-Valat, 2019), `wordcloud` (Fellows, 2018), `RWeka` (Hornik et al., 2009) and `RColorBrewer` (Neuwirth, 2014). The word clouds will be composed of two-word combinations (when using all survey comments which may elicit more sentiment than using single words alone) and one-or-two-word combinations (when using the subset of comments pertaining to the lower ratings). The steps to generate the word clouds are:

1. Copy the relevant text from the survey excel sheet to a plain text file.
2. The text mining (`tm`) library manages documents using a system called a corpus.
3. Clean the text using functions to convert the text to lower case, remove numbers, remove common English language stop words, remove punctuation and eliminate whitespace.
4. Function `NGramTokenizer` from library `RWeka`, is used to build up a term document matrix of all two-word combinations. This data frame is sorted so the most popular terms will be most visible in the word cloud.
5. Use the wordcloud function to choose colours, scale etc. The cloud was limited to displaying the 120 most popular two-word combinations.

3 Results

3.1 Aggregated Data Results

3.1.1 Summary

Figure 5 below shows a bar chart summarising the results for the aggregated data analysis. The sections that follow show the confusion matrix and overall test error rate achieved using each of the methods.

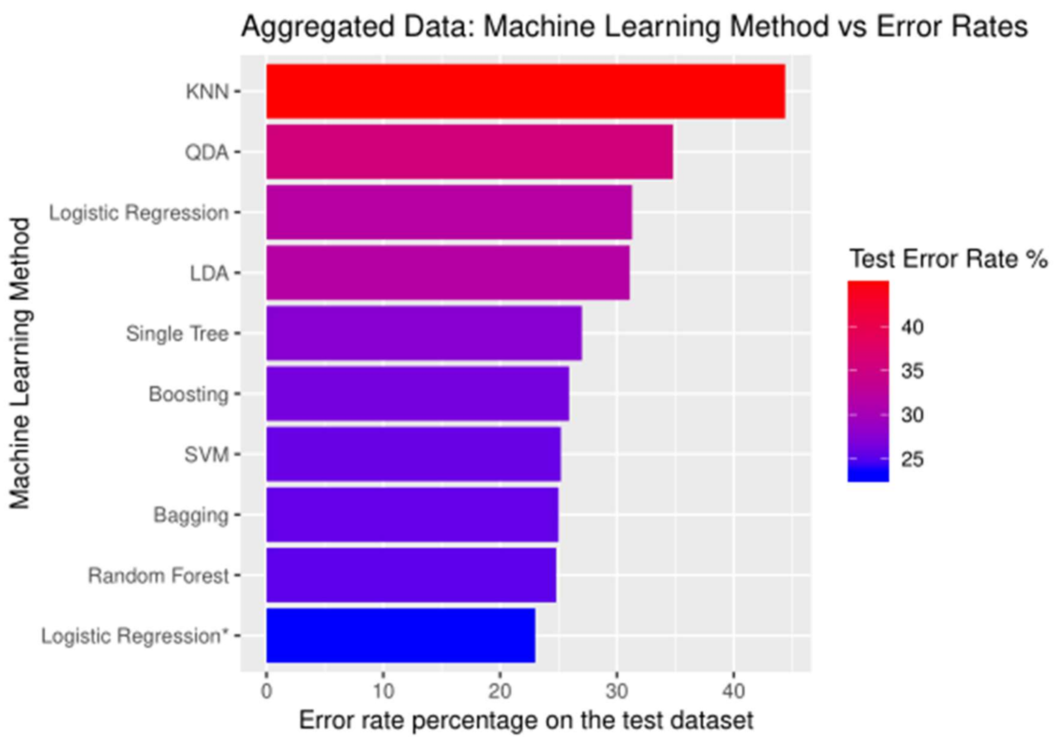


Figure 5 Test error rates for the Machine Learning techniques used on the aggregated data.

Note that "Logistic Regression" refers to a model with one-time shoppers removed and an interaction of the higher order moments (for variable DaysSinceLastShop) included as predictors. See section 3.1.11 for details.*

3.1.2 Classification Trees

The tree generated by `rpart` is shown by Figure 6 below.

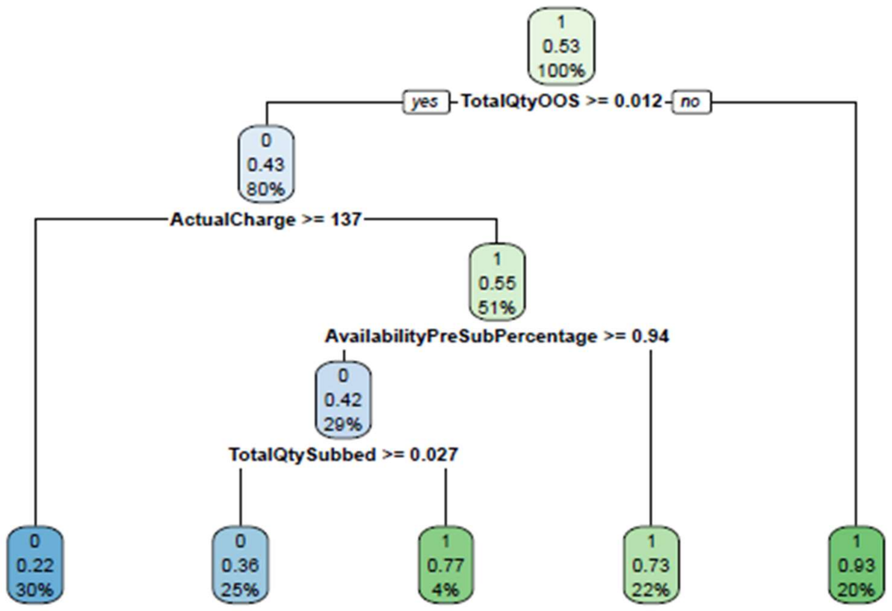


Figure 6 Classification tree generated in R by rpart

Terminal node 1 looks at variable TotalQtyOOS at a particular threshold value (0.012), moving to node 2, or a classification decision of a 1 based on its value. Terminal node 2 looks at variable ActualCharge at a threshold of 137 and moves to node 3 or a classification decision of 0 based on the variable value. Terminal node 3 looks at variable AvailabilityPreSubPercentage at a threshold of 0.94 and moves to terminal node 4 or to a classification decision of a 1 based on its value. Terminal node 4 looks at variable TotalQtySubbed at a threshold of 0.027 and classifies the observation as either 1 or 0 based on its value.

The `varImp` (Kuhn, 2018) function from the R library `caret` informs us that the three variables of most contributory value to the model are: ActualCharge, TotalOrderItems, TotalQtyOOS.

Figure 7 shows a complexity parameter table for the `rpart` fit. This gives a visual representation of the cross-validation results for the fit. The R function `plotcp` was used to generate this figure.

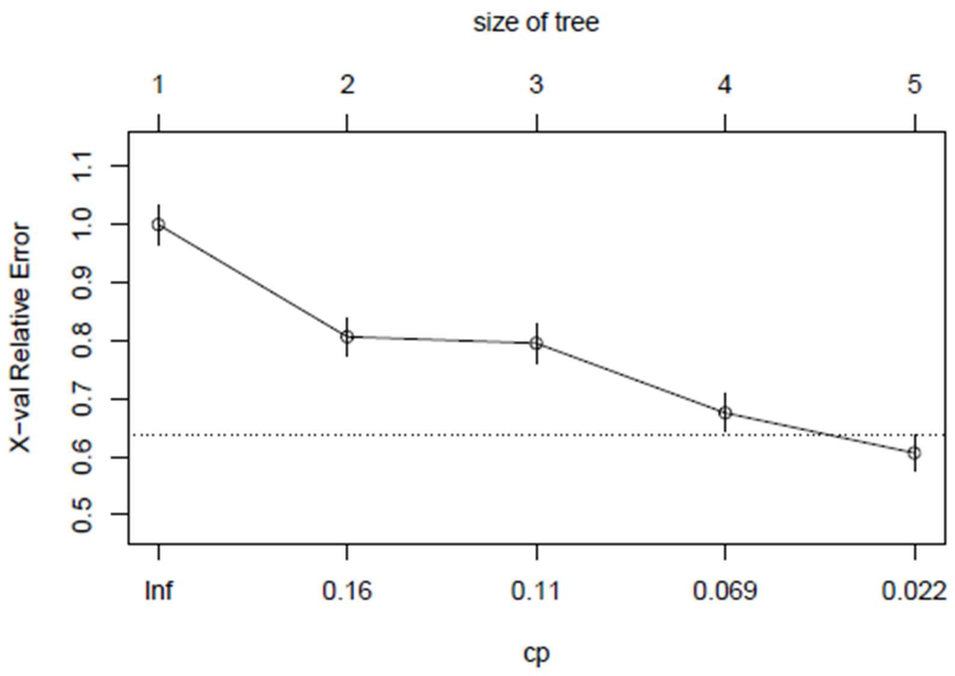


Figure 7 Complexity parameter (CP) table for the rpart fit

The upper horizontal axis is the size of the tree in terms of the number of terminal nodes. The lower horizontal axis is the cost-complexity parameter (cp). The vertical axis is the 10-fold cross-validated relative error. The horizontal dotted line marks the 1 standard error (SE) pruning point**.** The best value of cp should be the smallest one, so that the cross-validated error rate is a minimum. Therefore, we can see from the figure above that five terminal nodes yields the optimum classification tree. No pruning is required in this particular case. If it were needed, the tree can be pruned using the R command: `prune.rpart(fitrpart,0.022)` where the figure in brackets is the lowest value on the cp axis in the complexity parameter table. Table 2 below shows the confusion matrix for this particular machine learning method:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	364	166	530
Actual 1 (leave)	91	332	423
Total	455	498	953

Table 2 The confusion matrix generated for classification trees

The test error rate for a single classification tree on this data is 27%.

3.1.3 Bagging

With this method, variable importance is illustrated by using some R functions. `varImpPlot` (Liaw and Wiener, 2002), applied to the fitted model, shows a dot chart of variable importance as measured by a random forest. The R function `importance` (Liaw and Wiener, 2002), gives a table of two measures. The first measure is computed from permuting OOB (out-of-bag) data. For each tree, the prediction error rate on the out-of-bag portion of the data is recorded. Then the same is done after permuting each predictor variable. The difference between the two are then averaged over all trees and normalized by the standard deviation of the differences. The second measure is the total decrease in node impurities from splitting on the variable, averaged over all trees. For our purposes i.e. classification, the node purity is measured by the Gini index. An index of zero would indicate complete node purity increasing to larger values as the node becomes more impure. Overall, these measures highlight that ActualCharge, ValueOfOutOfStocks, AvailabilityPreSubPercentage and TotalOrderItems are the most significant variables in the bagged tree option.

Table 3 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	335	118	453
Actual 1 (leave)	120	380	500
Total	455	498	953

Table 3 The confusion matrix for a bagged classification tree

The test error rate for a bagged classification tree on this data is 25%.

3.1.4 Random Forests

Table 4 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	336	117	453
Actual 1 (leave)	119	381	500
Total	455	498	953

Table 4 The confusion matrix for Random Forest

The test error rate for a random forest on this data is 24.8%. Comparing all of the tree-based methods, random forest gives the lowest error rate.

3.1.5 Boosting

The model summary (not shown) lists the most important variables. All variables are given a relative influence score, which when combined, add up to 100. ActualCharge is the variable with the largest relative influence (13.4), second is TotalQtyOOS (9.26) and third is TotalOrderItems (9.02).

Table 5 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	337	129	466
Actual 1 (leave)	118	369	487
Total	455	498	953

Table 5 The confusion matrix for boosted trees

The test error rate for using a boosted trees ML method on this data is 25.9%.

3.1.6 Support Vector Machines (SVM)

The best performing SVM as chosen by cross validation uses kernel = radial, degree = 1 and cost = 10.

Table 6 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	316	101	417
Actual 1 (leave)	139	397	536
Total	455	499	953

Table 6 The confusion matrix for Support Vector Machine

The test error rate for using support vector machines on this data is 25.2%.

3.1.7 Linear Discriminant Analysis (LDA)

Table 7 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	281	122	403
Actual 1	174	376	550

(leave)			
Total	455	498	953

Table 7 The confusion matrix for Linear Discriminant Analysis

The test error rate for using linear discriminant analysis on this data is 31.1%.

3.1.8 Quadratic Discriminant Analysis (QDA)

Table 8 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	402	279	681
Actual 1 (leave)	53	219	272
Total	455	498	953

Table 8 The confusion matrix for Quadratic Discriminant Analysis

The test error rate for using quadratic discriminant analysis on this data is 34.8%.

3.1.9 K-Nearest Neighbours (KNN)

A range of K values were used: 5, 30, 40, 50, 100. K = 50 produced the lowest test error.

Table 9 shows the confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	193	162	355
Actual 1 (leave)	261	336	597
Total	454	498	952

Table 9 The confusion matrix for K-nearest neighbours

The test error rate for a k-nearest neighbours analysis on this data is 44.4%. This is a very high error rate. KNN is not a good classification technique for this data. The proportion of customers predicted to stay who actually left is 261/454 = 57.5%. The proportion of customers predicted to leave who actually stayed is 162/498 = 32.5%. It appears that KNN is poorest at predicting the customers who will stay shopping.

3.1.10 Principal Component Analysis (PCA)

Table 10 below shows that 30.48% of the total variability in the data is accounted for by the first component. 49% of the total variability in the data is accounted for by the first two components. 62% of the total variability in the data is accounted for by the first three components. Table 11 shows that PC1 is a measure of the overall charge, the number of items/product lines ordered, the time spent picking the items and the number of items out of stock and items substituted. PC2 is a measure of the items substituted contrasted with the items out of stock.

Variable	PC1	PC2	PC3
Standard Deviation	1.9906	1.5512	1.2979
Proportion of Variance	0.3048	0.1851	0.1296
Cumulative Proportion	0.3048	0.4899	0.6195

Table 10 Describing the variability of the first three components

Variable	PC1	PC2	PC3
ExpectedFulfillmentCharge	0.18307326	0.025457810	-0.01128741
ActualCharge	-0.39509139	-0.025508760	0.27845656
TotalOrderItems	-0.42644654	-0.004815758	0.31878929
TotalPickedLines	-0.39293817	0.016472366	0.32028291
TotalQtySubbed	-0.35752476	0.323259399	-0.29907931
TotalQtyOOS	-0.27444866	-0.459967163	-0.19156184
TotalPickTimeSeconds	-0.30553268	0.046822014	0.13378410
AvailabilityPreSubPercentage	0.02192217	-0.002383474	0.08287021
PercentageOutOfStocks	-0.09126584	-0.496502253	-0.35607579

Variable	PC1	PC2	PC3
PercentageSubstitutions	-0.11731831	0.354997652	-0.51859412
ValueOfSubstitutions	-0.33295237	0.311761734	-0.32812892
ValueOfOutOfStocks	-0.20803090	-0.456324124	-0.24986607
RelatedCallsCount	0.03042904	-0.046662130	-0.03721946

Table 11 Principal Component composition

In addition to running PCA, we can also explore the aggregated data. A useful visual aid to plot different groups of shoppers was generated using the `ggplot2` version of biplot (called `ggbiplot`) for Principal Components. `ggbiplot` (Vu, 2011) allows you to colour different groups of observations. Out of the 1905 customers, R code was used to isolate four specific groups:

- Group 1: customer has shopped once and is now dormant
- Group 2: customer has shopped once and is still a current customer
- Group 3: customer has shopped more than once and is now dormant
- Group 4: customer has shopped more than once and is still a current customer

Table 12 summarises the breakdown of the customers:

Shopped:	Churn = 1 (dormant)	Churn = 0 (current shopper)	Total
1 (once)	412	14	426
>1 (more than once)	588	891	1479
Total	1000	905	1905

Table 12 A summary of the shopping group types

Looking at the numbers, it seems strange that 14 customers only shopped once but have not become dormant. This warranted investigation so the Musgrave contact was asked for some clarification. If a customer has less than 4 shops, Musgrave don't classify them as an established shopper yet. The default is that this type of shopper will go dormant after 70 days. Out of the 14 that have only shopped once, but are still current customers, only 4 of those had last shopping dates longer than 70 days. Musgrave queried these 4 cases and indeed they have shopped again after June 2018 and so it is justified that they are not classified as dormant. The window of time between June 2018 and Feb 2019 is not visible to us.

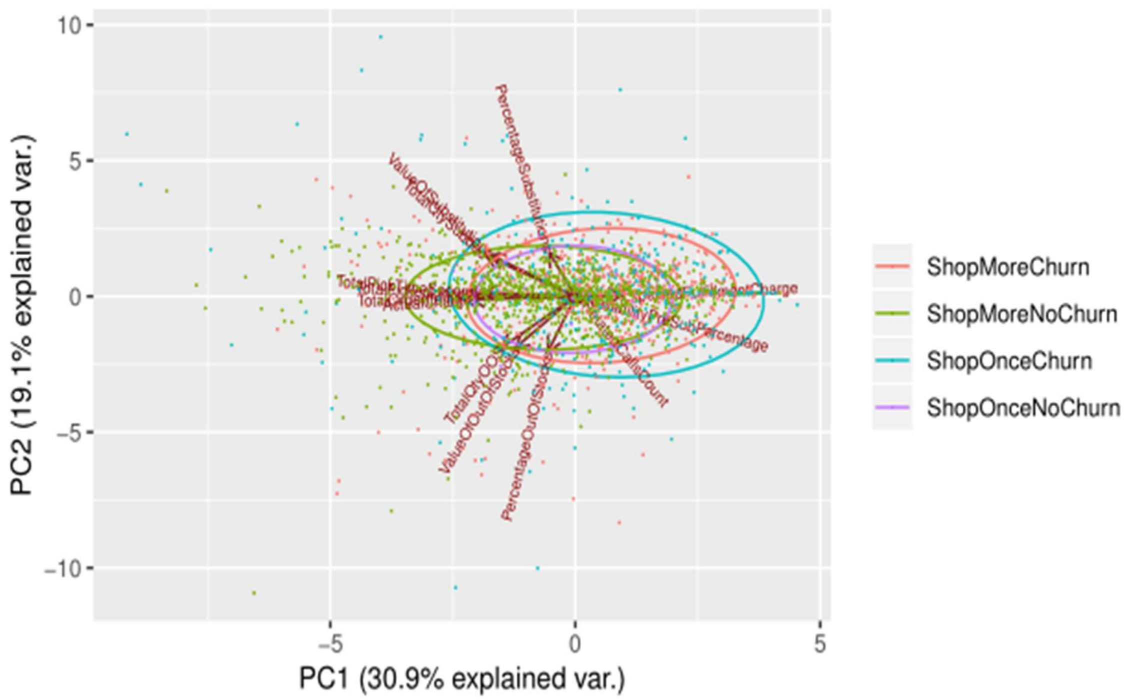


Figure 8 The first two principal components with data points plotted

Looking at the biplot of the first two principal components (Figure 8), the light purple ellipse is where the 14 customers who shopped once, and are not considered as dormant, lie. The centre of the ellipse is at roughly (0,0) meaning they are equally explained by PC1 and PC2. This plot was useful for finding atypical shopping transactions. Some of these were investigated based on their proximity away from the large clusters of data points. In essence, they appeared to be outliers. Examples of these would be a large ExpectedGoodsCharge/ActualCharge ratio, a very large percentage of the order being substituted, a very large percentage of the order being out-of-stock. The Musgrave contact was consulted, and he confirmed that they were legitimate transactions but were mass bulk purchases made on single items - most likely a special offer, and due to stock levels, only part of the order could be fulfilled.

3.1.11 Logistic Regression

Table 13 shows the confusion matrix for the first pass of logistic regression:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	290	165	455
Actual 1 (leave)	133	365	498

Total	423	530	953

Table 13 The confusion matrix for the initial logistic regression run

The test error rate for a logistic regression run on this data is 31.3%.

The model summary (not shown) tell us that variables ActualCharge, ValueOfSubstitutions and RelatedCallsCount are significant in the model as they all have a p-value less than 0.05. A more formal test can be run using the `drop1` function in R. The code to run it is as follows:

```
drop1(fitlr, test="Chisq")
```

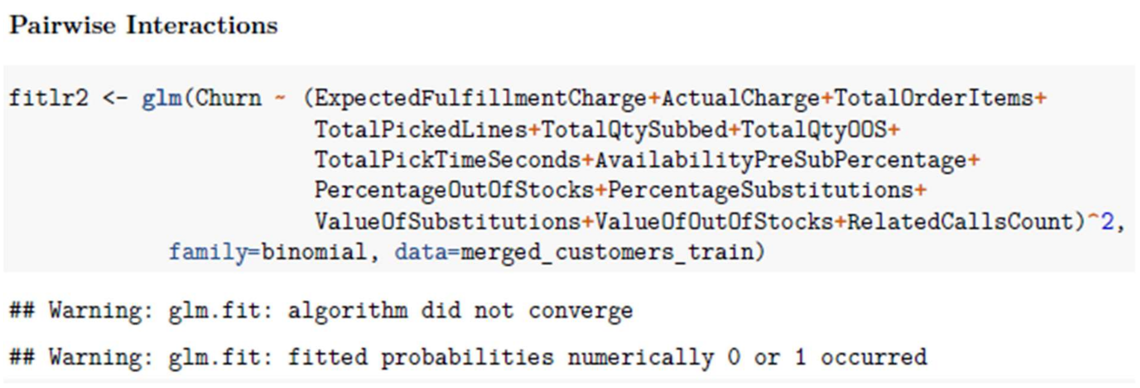
A chi-square test is a statistical method to assess the goodness of fit between a set of observed values and those expected theoretically. The drop1 function takes a fitted model, drops each variable in turn and computes a summary table of the changes in the goodness of fit between the model with and without this variable. This table displays certain criteria to allow the user to determine whether a variable is significant within a model. Table 14 below shows the lines of the summary table pertaining to the variables mentioned above.

Variable	Df	Deviance	AIC	LRT	Pr(>Chisq)
ActualCharge	1	1147.5	1173.5	12.8525	0.000337 ***
ValueOfSubstitutions	1	1141.8	1167.8	7.1423	0.007529 **
RelatedCallsCount	1	1139.2	1165.2	4.4848	0.034 *

Table 14 Results of drop1 analysis: df=degrees of freedom, AIC=Aikake Information Criterion, LRT=Likelihood Ratio Test

We observe from the table above that a larger value of LRT and a lower p-value based on a Chi-square test mean that the variable is significant as a predictor in the model.

Next, we can see if interactions between predictors produce a better error rate. The R code for trying this is shown in the code chunk below.



Pairwise Interactions

```
fitlr2 <- glm(Churn ~ (ExpectedFulfillmentCharge+ActualCharge+TotalOrderItems+
                       TotalPickedLines+TotalQtySubbed+TotalQtyOOS+
                       TotalPickTimeSeconds+AvailabilityPreSubPercentage+
                       PercentageOutOfStocks+PercentageSubstitutions+
                       ValueOfSubstitutions+ValueOfOutOfStocks+RelatedCallsCount)^2,
             family=binomial, data=merged_customers_train)
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

This attempt produced a worse error rate of 39.7% and we can also see warning messages about the non-convergence of the model. It was decided to introduce a condition into the logistic regression environment. Table 15 below shows the confusion matrix conditioning on the fact that one-time shoppers have been excluded. The training and test sets are now smaller due to this.

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	367	75	442
Actual 1 (leave)	144	153	297
Total	511	228	739

Table 15 The confusion matrix for logistic regression with the condition that one-time shoppers have been removed from the analysis.

This model includes the calculations using the DaysSinceLastShop variable related to the higher order moments of standard deviation, skewness and kurtosis. The test error rate for logistic regression, conditioning on the removal of the one-time shoppers whilst adding in the higher order variables related to DaysSinceLastShop is 29.6%. A small improvement. The standard deviation, skewness and kurtosis variables do have significance according to the model summary. Lastly, we can try to add a pairwise interaction term for these significant variables. The code chunk for this is shown below.

```
fitlr5 <- glm(Churn ~ (DaysSinceLastShop_sd+DaysSinceLastShop_skew+
                DaysSinceLastShop_kurt+ActualCharge)^2+
                ExpectedFulfillmentCharge+RelatedCallsCount+
                TotalOrderItems+TotalPickedLines+
                TotalQtySubbed+TotalQtyOOS+TotalPickTimeSeconds+
                AvailabilityPreSubPercentage+PercentageOutOfStocks+
                PercentageSubstitutions+ValueOfSubstitutions+
                ValueOfOutOfStocks,
             family=binomial, data=merged_customers_train)

summary(fitlr5)
```

Table 16 shows the confusion matrix for this version of the model.

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	376	66	442
Actual 1 (leave)	103	194	297
Total	479	260	739

Table 16 The confusion matrix for logistic regression using conditioning and pairwise interactions

The test error rate for a logistic regression (with conditioning) and some 2nd degree interactions is 22.9%. This is the best error rate achieved with the aggregated dataset although it has come at the expense of excluding the data related to one-time shoppers.

3.2 Full Data Results

Prior to presenting the mixed model results, the next two sections outline some noteworthy data exploration results.

3.2.1 Patterns of shopping behaviour

Figure 9 below shows a histogram of the spread of how many times customers shopped.

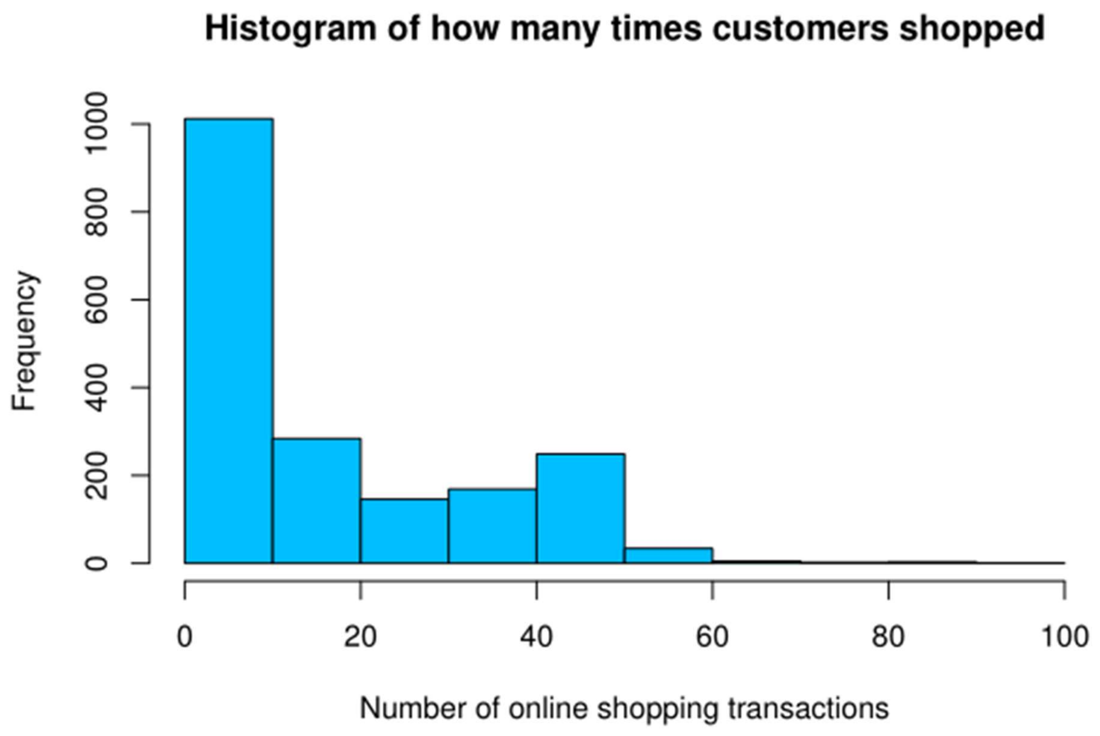


Figure 9 Histogram showing how many times a customer shopped

It is the upper end of this histogram that is most interesting. Some customers have shopped more than 90 times. Let's focus on the patterns of shoppers who have shopped more than 50 times. 64 customers have shopped more than 50 times each. Some of these customers have left (16) and some are still shopping (48). We know this due to their sequenceID. Figure 10 shows two histograms for two dormant customers.

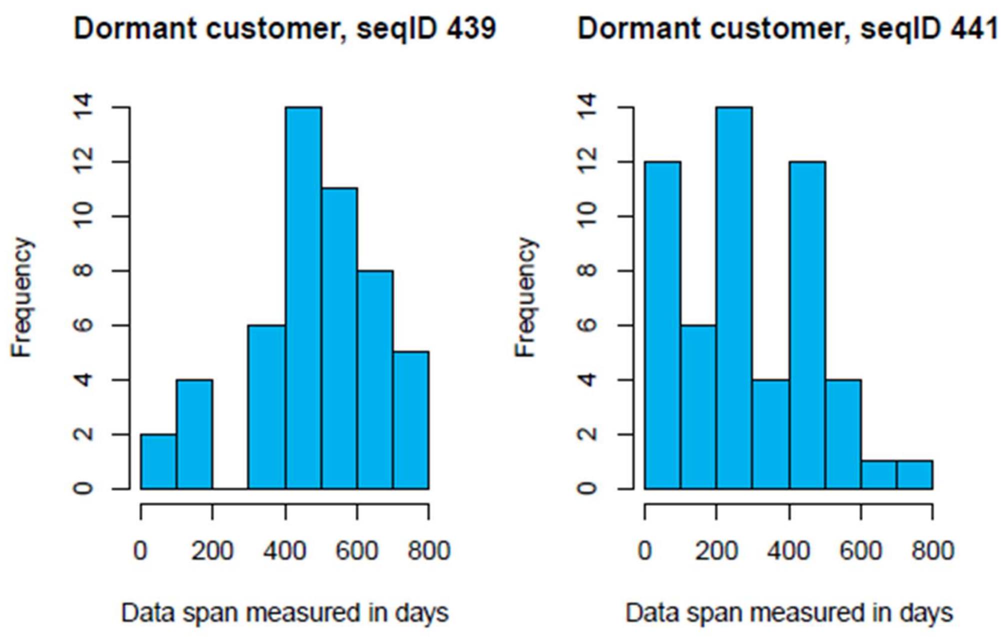


Figure 10 Histograms showing the shopping patterns of two dormant customers

Customer 439 shopped 50 times and the histogram shows that they were a very frequent shopper 400-500 days prior to the base date (1st June 2018). So, they shopped approximately 14 times in the Spring of 2017. They did not shop at all 200-300 days from the cut-off date which would have been roughly late Autumn/early Winter of 2017. Customer 441 shopped 54 times and there is a very seasonal pattern to their shopping habits. They seem to shop more in Spring/Summer and their frequency falls off by about 50%-75% in Autumn/Winter. Note that in both histograms we have shopping data dating as far back as 700-800 days i.e. a 2-year period.

Figure 11 shows some patterns for two current shoppers, and it is clear that they are very different types of shoppers.

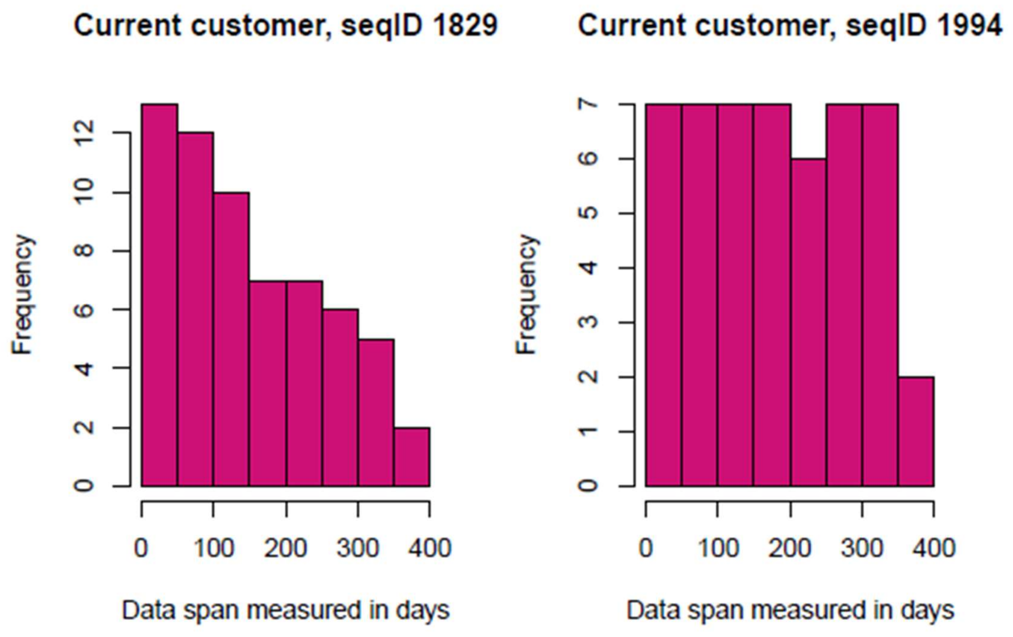


Figure 11 Histograms showing the shopping patterns of two current customers

For customer 1829, there has been a steady increase in their shopping patterns. They made 13 online shopping transactions in the 50 days prior to the base date (1st June 2018). If we look at the bin representing one year before that, they shopped just twice. Customer 1994 is a very regular and predictable shopper. They consistently make 7 shops within a 50-day period i.e. they are a weekly shopper. The histograms shown are only 4 out of 64 histograms generated and many more shopping patterns were seen among them.

The histograms for the dormant shoppers span 0 to 800 days (or roughly 2 years). The histograms for the current shoppers span 0 to 400 days (approximately 1 year). Essentially, this means that time cannot be used as a predictor or it could be construed that if you shop within the period of 400 to 750 days prior to the data cut-off point, you are more likely to become a dormant customer. This is not the case as we do not have any visibility of how the current shoppers performed during this time. Hence, the variable DaysSinceLastShop cannot be used as a predictor.

3.2.2 Current customers – decreasing spending analysis

One of the main project aims is to find a relationship between current and dormant customers and to learn more about what makes a customer stop shopping. However, it is also beneficial to look at the current customers in isolation - to identify variations, or more importantly, decreases in their spending. If they are partially defecting (Buckinx and Van den Poel, 2004) – could they be tempted back by incentivised vouchers or other means? The analysis focussed on established current customers who had shopped 20 times or more in one year. Plots were produced for these customers, plotting the variable ActualCharge on the y-axis and DaysSinceLastShop on the x-axis. Figure 12 below shows the pattern of spending for customer "sequenceID 1996".

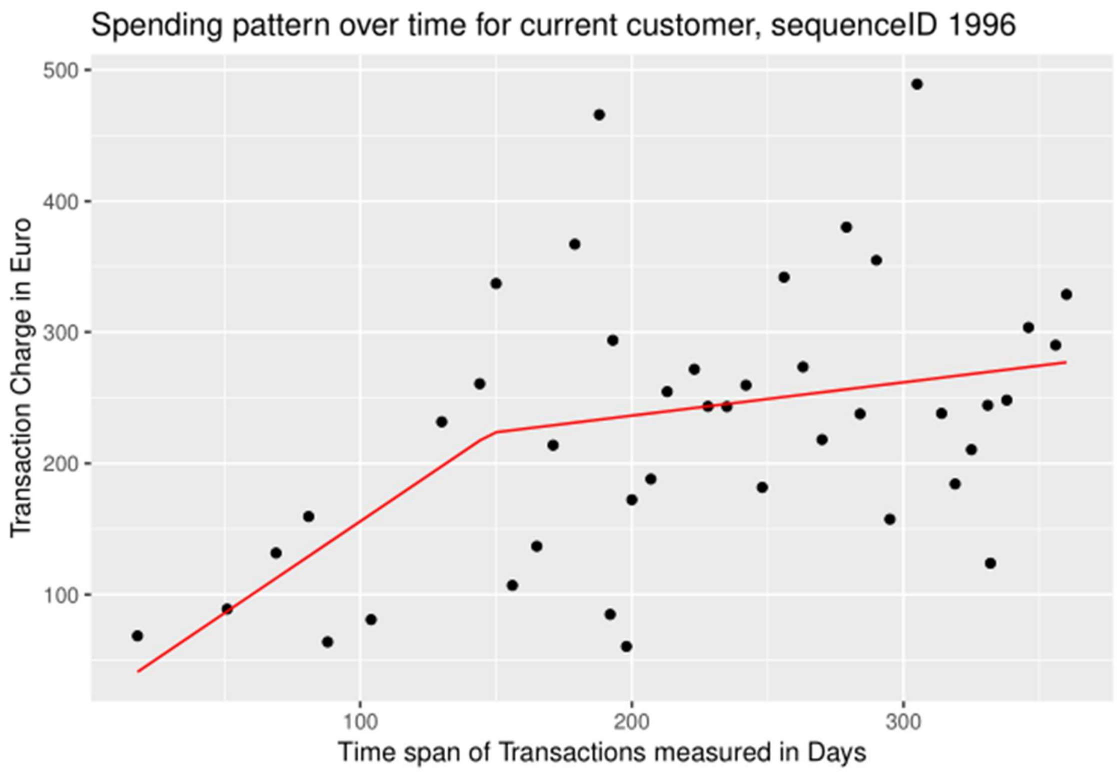


Figure 12 Decreasing spending pattern (amount vs days) for customer 1996

Using June 1st, 2018 as the base date, working backwards in time, we see that this customer shopped 15 times, 200 - 300 days prior the base date. This makes them a weekly shopper. A rough average of what they spent in this time period is €250 per transaction. If we move forward in time to 0 – 100 days prior to the base date, this customer has shopped just 6 times with the average spend of €100. They have now become a bi-weekly shopper and have drastically reduced their spending bill. Furthermore, the superimposed red line shows a piecewise linear regression using one breakpoint. The breakpoint and the regression lines are calculated using the `segmented` function from the R library of the same name (Muggeo, 2008). This customer has started to change their shopping habits approximately 148 days prior to the base date.

Lastly, 566 current customers who had shopped 20 times or more were identified. An analysis of their customer plots tells us that at least 20 of these customers have a decreasing trend in their spending. Table 17 below shows their customer numbers.

1074	1089	1099	1111	1124
1131	1195	1198	1252	1323

1329	1352	1414	1705	1816
1866	1893	1922	1950	1996

Table 17 Specific "sequenceID" customer numbers whose transaction values have decreased over time

Musgrave were consulted and, as yet, these customers have not become dormant, but it is clear they are shopping elsewhere. Corrective action i.e. a free gift or money-off voucher, could be employed here as an incentive to help them remain a loyal shopper and recapture their old spending habits.

3.2.3 Generalized Linear Mixed Model (GLMM)

Table 18 below shows a confusion matrix:

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	11944	583	12527
Actual 1 (leave)	2240	722	2962
Total	14184	1305	15489

Table 18 The confusion matrix for mixed model (glmer) for new customers

The test error rate for mixed model regression (for a new customer) with all predictors is 18.23%.

This error rate is achieved using the option "`re.form=NA`" in the predict function, which means that if our training set already has prior knowledge of a customer and additional transactions from that customer feature in the test set, we do not use the random effect information already gathered for them.

We can also calculate the error rate for an existing customer i.e. the test data contains additional shopping data for customers we already know about. In that case we would use the "`allow.new.levels=TRUE`" in the predict function. The prediction would use the fixed and random effects for customers already known. The confusion matrix for this version of the model is shown in Table 19 below.

	Predicted 0 (stay shopping)	**Predicted 1 (leave)**	Total
Actual 0 (stay shopping)	12522	5	12527
Actual 1 (leave)	155	2807	2962
Total	12677	2812	15489

Table 19 The confusion matrix for mixed model (glmer) for existing customers

The test error rate for mixed model regression (for existing customers) with all predictors is 1%.

From this point onwards, when test error rates are referred to, it is with the "`re.form=NA`" option. It is preferable to look at error rates for a model that will make predictions for new customers (or a new dataset) rather than just the customers we have.

We can analyse the figures for the random effects contained in this model. Table 20 shows the breakdown of the random effects.

Groups	Name	Variance	Std. Dev.
sequenceID	$\sigma^2_{\text{sequenceID}}$	138.722	11.778
StoreID	$\sigma^2_{\text{StoreID}}$	4.315	2.077

Table 20 Breakdown of the random effects

The variance and standard deviation numbers for StoreID are small. Is StoreID significant in the model? To test this, create a model leaving out StoreID as a random effect. The test error rate reduces to 17.8%. We should compare these models using an ANOVA test. The models are:

- fitlrmm1: The model <u>without</u> StoreID included as a random effect (simpler model).
- fitlrmm2: The model <u>with</u> StoreID included as a random effect (more complex model).

Table 21 below shows the results of the ANOVA comparison.

Model	Df	AIC	BIC	logLik	Deviance	Chisq	Chisq Df	Pr(>Chisq)
fitlrmm1	28	3113.3	3327.5	-1528.7	3057.3			
fitlrmm2	29	3111.4	3333.2	-1526.7	3053.4	3.8675	1	0.04923 *

Table 21 ANOVA comparison for simpler vs complex model

The hypotheses are:

$H_0 : \sigma^2_{\text{StoreID}} = 0.$

$H_A : \sigma^2_{\text{StoreID}} > 0.$

The p-value must be halved as we are doing a test in the boundary of the parametric space (σ^2 can't be negative). So, the p-value is 0.0246. This indicates that H_0 should be rejected and that StoreID has significance in the model even though the error rate is slightly higher when it is included.

At this point, work can begin on the removal of some of the non-significant variables. Predictors can be eliminated as long as they do not adversely affect the performance rate of the model. This will be done by a succession of "drop1" analyses. drop1 (Bolker, 2019) computes all the single terms that can be dropped from the model. Those models are fit and a table of the changes in fit are computed. Table 22 below summarises the changes in these models, giving their error rate and AIC. The Akaike Information Criterion (AIC) is an estimator of the relative quality of statistical models for a given set of data. It is calculated by

$$AIC = 2p - 2\log(\hat{L})$$

where p is the number of predictors and $\hat{L}$ is the maximum value of the likelihood function for the model. (Fabozzi et al., 2014) say that AIC is generally considered the first model selection criterion that should be used in practice. Lower values of AIC are preferred.

Model Number	**Variable(s) Excluded based on drop1 results**	**Test Error Rate %**	**AIC**
fitlrmm1	StoreID	17.8%	3113
fitlrmm2	None, full model	18.23%	3111
fitlrmm3	TotalPickTimeSeconds, ValueOfSubstitutions	18.25%	3107
fitlrmm4	PercentageOutOfStocks, PercentageSubstitutions	18.24%	3104
fitlrmm5	ValueOfOutOfStocks	18.24%	3102
fitlrmm6	ExpectedFulfillmentCharge	18.37%	3100
fitlrmm7	TotalOrderItems	18.43%	3099
fitlrmm8	TotalQtyOOS	18.37%	3098
fitlrmm9	RelatedCallsCount	18.41%	3098
fitlrmm10	TotalQtySubbed	18.5%	3098
fitlrmm11	AvailabilityPreSubPercentage	18.35%	3099
fitlrmm12	No exclusions made; interactions considered	18.8%	3158

Model Number	Variable(s) Excluded based on drop1 results	Test Error Rate %	AIC
	between all remaining fixed predictors		
fitlrmm13	Interaction considered ActualCharge*TotalPickedLines	19.5%	3089

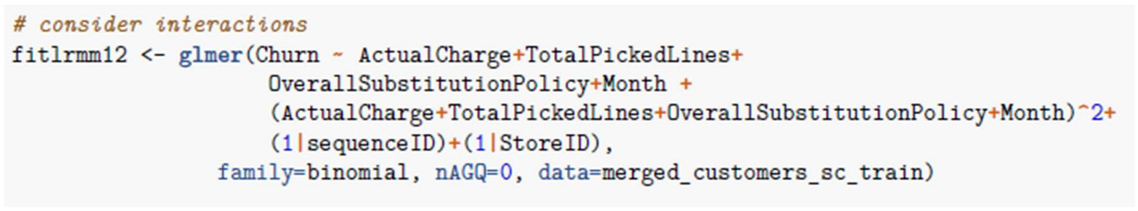
Table 22 Fitted mixed models summary, showing excluded variables for each pass, the test error rate percentage and the AIC value

Table 23 below shows the summary after the drop1 analysis of fitlrmm11. The significance column in the table is a grading scheme for the p-value i.e. the more significant the variable, the higher the number of asterisks. We can see that all remaining predictors are significant in the model. Therefore, no more variables will be removed but interactions should be considered.

Variable	Df	AIC	LRT	Pr(Chi)	Significance
(none)		3098.5			
ActualCharge	1	3107.0	10.540	0.0011769	**
TotalPickedLines	1	3115.2	18.679	1.547e-05	***
OverallSubstitutionPolicy	2	3109.5	15.039	0.0005424	***
Month	11	3170.9	94.426	2.252e-15	***

*Table 23 Results after performing a drop1 analysis of model fitlrmm11. Significance column grading is: '***' = p-value is between 0 and 0.001, '**' = p-value is between 0.001 and 0.01, '*' = p-value is between 0.01 and 0.05, '.' = p-value is between 0.05 and 0.1*

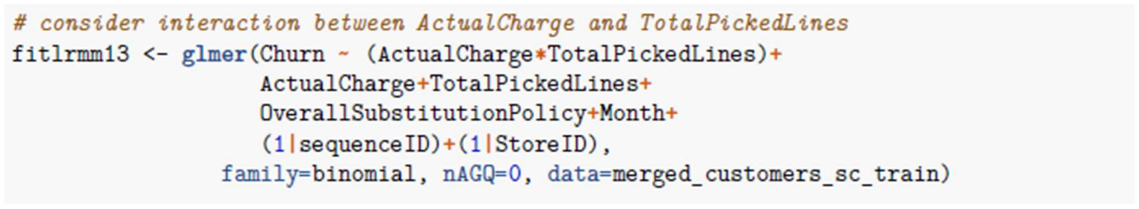
Interactions were considered in model fitlrmm12. All remaining fixed predictors were used in a pairwise interaction. Figure 13 below shows the code chunk used in R to define the model.



```
# consider interactions
fitlrmm12 <- glmer(Churn ~ ActualCharge+TotalPickedLines+
                   OverallSubstitutionPolicy+Month +
                   (ActualCharge+TotalPickedLines+OverallSubstitutionPolicy+Month)^2+
                   (1|sequenceID)+(1|StoreID),
               family=binomial, nAGQ=0, data=merged_customers_sc_train)
```

Figure 13 Model fitlrmm12 is similar to model fitlrmm11 but with pairwise interactions considered

This model has an AIC value of 3158, a test error rate of 18.8% and 63 degrees of freedom so fitlrmm12 is not a suitable model. However, using a drop1 analysis on fitlrmm12 shows that one pairwise interaction is significant – ActualCharge and TotalPickedLines. Therefore, one final model will be considered, fitlrmm13. Figure 14 below shows the code chunk.



```
# consider interaction between ActualCharge and TotalPickedLines
fitlrmm13 <- glmer(Churn ~ (ActualCharge*TotalPickedLines)+
                     ActualCharge+TotalPickedLines+
                     OverallSubstitutionPolicy+Month+
                     (1|sequenceID)+(1|StoreID),
                   family=binomial, nAGQ=0, data=merged_customers_sc_train)
```

Figure 14 Model fitlrmm13 is similar to model fitlrmm11 but with one interaction added, between ActualCharge and TotalPickedLines

This model has an AIC value of 3089 and 19 degrees of freedom. The test error rate has increased slightly to 19.5% (when compared to model fitlrmm11). We can perform an ANOVA test between fitlrmm11 and fitlrmm13 to decide if the interaction is significant. The two model equations are shown below. (Note the factor reference level of predictor Month was redefined so that Month 1 appears in the model and Month 5 has a zero estimate. This will be addressed in section 4.2).

fitlrmm11 model:

$\log\left(\frac{\Pr(Chur\ \ )}{1-\Pr(Churn)}\right) = \beta_0 + \beta_1$*ActualCharge + β_2*TotalPickedLines + β_3*OverallSubstitutionPolicyNONE + β_4*OverallSubstitutionPolicySOME + β_5*Month1 + β_6*Month2 + β_7*Month3 + β_8*Month4 + β_9*Month6 + β_{10}*Month7 + β_{11}*Month8 + β_{12}*Month9 + β_{13}*Month10 + β_{14}*Month11 + β_{15}*Month12 + $b1_j$ + $b2_k$

where:

$b1_j \sim N(0, \sigma^2_{StoreID})$ and

$b2_k \sim N(0, \sigma^2_{sequenceID})$

fitlrmm13 model:

$\log\left(\frac{\Pr(Churn)}{1-\Pr(Churn)}\right) = \beta_0 + \beta_1$*ActualCharge + β_2*TotalPickedLines + β_3*OverallSubstitutionPolicyNONE + β_4*OverallSubstitutionPolicySOME + β_5*Month1 + β_6*Month2 + β_7*Month3 + β_8*Month4 + β_9*Month6 + β_{10}*Month7 + β_{11}*Month8 + β_{12}*Month9 + β_{13}*Month10 + β_{14}*Month11 + β_{15}*Month12 + β_{16}*(ActualCharge*TotalPickedLines) + $b1_j$ + $b2_k$

where:

$b1_j \sim N(0, \sigma^2_{StoreID})$ and

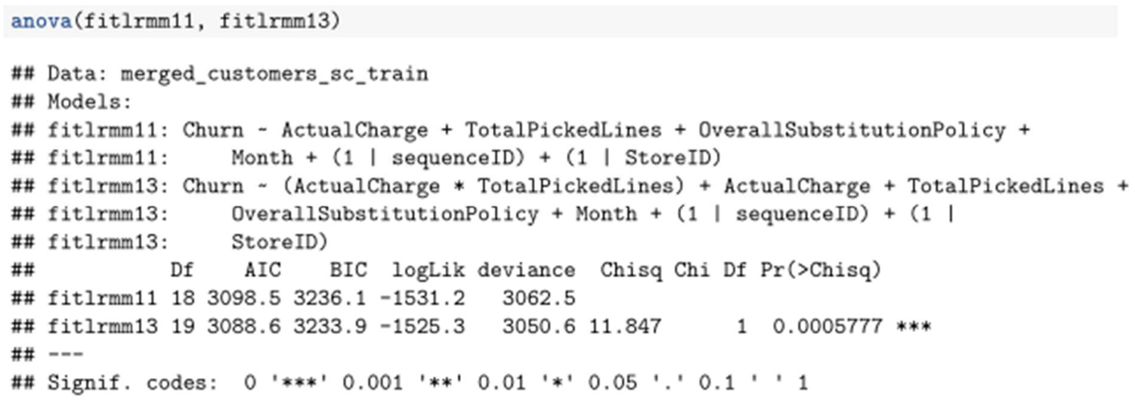
$b2_k \sim N(0, \sigma^2_{sequenceID})$

The hypotheses are:

$H_0 : \beta_{16} = 0$.

$H_A : \beta_{16} > 0$.

Figure 15 below shows the code output from the ANOVA comparison. Recall that we should halve the p-value we see in the summary. Therefore, we should reject H_0 and accept the alternative hypothesis that $\beta_{16} > 0$. We can conclude that the interaction term is significant in the model.



```
anova(fitlrmm11, fitlrmm13)

## Data: merged_customers_sc_train
## Models:
## fitlrmm11: Churn - ActualCharge + TotalPickedLines + OverallSubstitutionPolicy +
## fitlrmm11:     Month + (1 | sequenceID) + (1 | StoreID)
## fitlrmm13: Churn ~ (ActualCharge * TotalPickedLines) + ActualCharge + TotalPickedLines +
## fitlrmm13:     OverallSubstitutionPolicy + Month + (1 | sequenceID) + (1 |
## fitlrmm13:     StoreID)
##           Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## fitlrmm11 18 3098.5 3236.1 -1531.2   3062.5
## fitlrmm13 19 3088.6 3233.9 -1525.3   3050.6 11.847      1  0.0005777 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 15 ANOVA comparison of fitlrmm11 (no interaction term) and fitlrmm13 (interaction considered)

The decision as to whether to choose the fitlrmm11 or fitlrmm13 as the “best” model can be made by summarising over error rate performance, AIC and degrees of freedom. See section 4.1 and Table 24 for the discussion on the final model choice.

We can plot the estimates of the model coefficients. It could provide information as to which month is most likely to lead to customer churn. This plot is shown below in Figure 16 and it shows that month 4 (April) appears to be the most significant month for a customer to become dormant. Error bars are also shown, displaying +/- twice the standard error figure (taken from the model summary). The plot shows that the horizontal zero line passes through the error bars of all months except April meaning that this month is significantly different than the other eleven. May appears to be the least likely month to Churn. From June, all the way through to March, we can see small increments in the model estimate. The Musgrave contact was queried regarding this finding, and he confirmed that April is a significant month for Churn. Discussion section 4.2 delves deeper into the reasons why this is so.

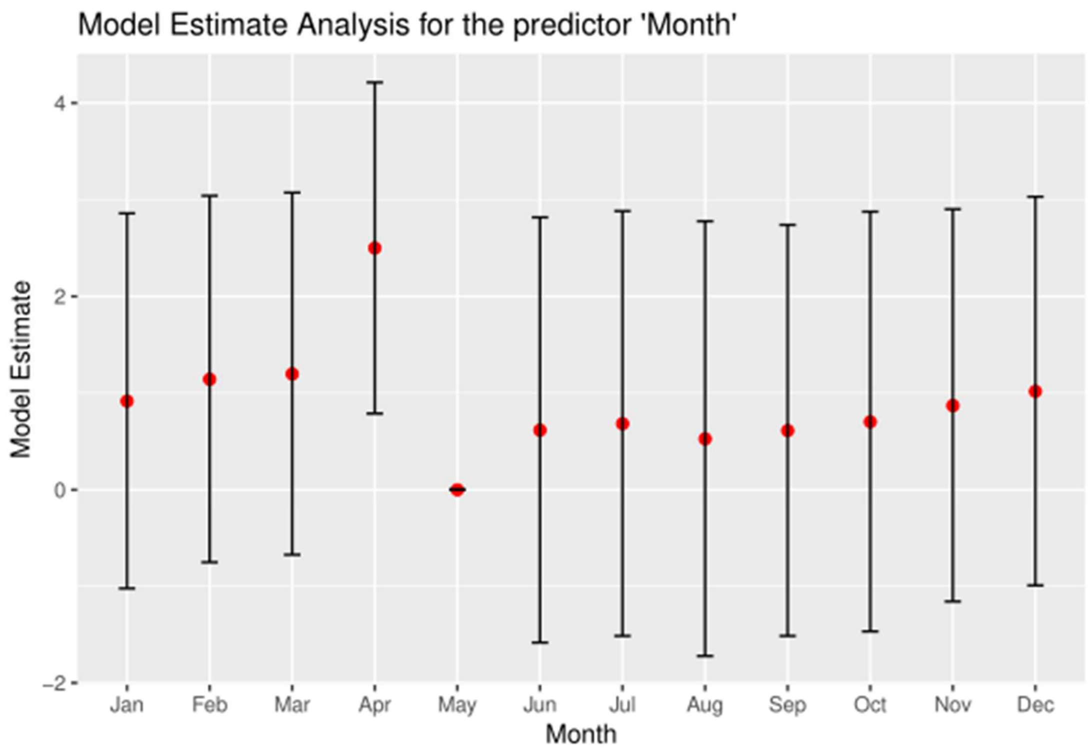


Figure 16 Scatterplot of month versus model estimate (for model fitlrmm11) with +/- 2 x Std Error bars displayed.

Another interesting visual uses `ggplot` (Wickham, 2016) to display ActualCharge against the logit of the Churn variable using the full dataset. We can fit a model, in this case a glm, for each month. Month 4 (April) shows a much steeper negative slope than the other months. This can be interpreted as saying that the customers who spend smaller amounts in April are the most likely to Churn. Those spending larger amounts in April are less likely to Churn. Figure 17 below shows the slopes per month.

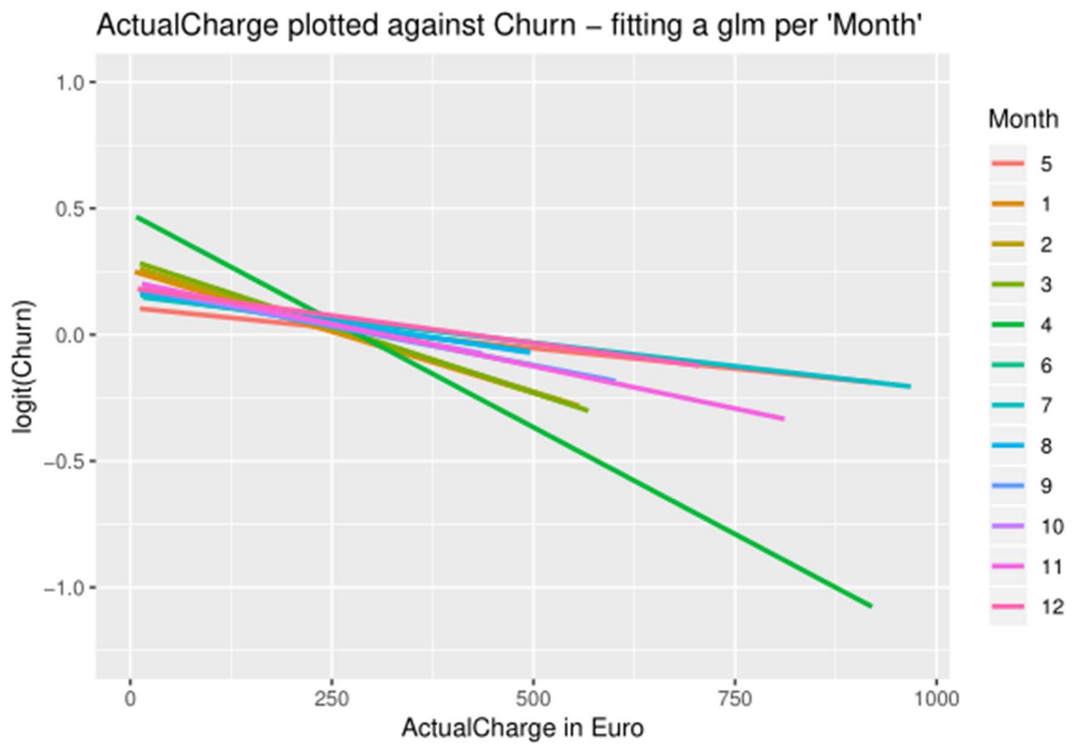


Figure 17 ActualCharge plotted against log(Churn), fitting a glm per Month

Given the results that have been presented, we can do some interpretation on the coefficients and answer the question: how much more likely is a customer to churn in March vs April? From model fitlrmm11 described earlier in this section, β_7 is the coefficient estimate for March and β_8 is the coefficient estimate for April. In the model summary (not shown), $\beta_7 = 1.1983$ and is defined as the log odds of churning in March (on average). $\beta_8 = 2.5005$ and is defined as the log odds of churning in April (on average). It is more intuitive to talk about odds rather than log odd so $\beta_7 = e^{1.1983} = 3.314$ is the odds of churning in March (on average). $\beta_8 = e^{2.5005} = 12.189$ and is the odds of churning in April (on average).

$$\frac{\beta_8}{\beta_7} = \frac{12.189}{3.314} = 3.68$$

Therefore, on average, a customer is 3.68 times more likely to churn in April as opposed to March.

3.2.4 GLMM assumption violation

An assumption of mixed models are that the distributions of the random effects are normal. Using the `ranef` (Bates et al., 2007) function from R library `lme4`, shows that sequenceID has a distribution that is non-normal in shape. It is bimodal. This is a violation of the model assumption. The response variable is binary. Consequently, the two peaks in the distribution indicate that there are two distinct groups – those that have stayed and those that haven't. Figures 18 and 19 below show the distributions for sequenceID and StoreID respectively. The distribution for StoreID is normal. To relax the constraint of the random effects requiring to be normally distributed, another technique called Non-Parametric Maximum Likelihood (NPML) would need to be studied.

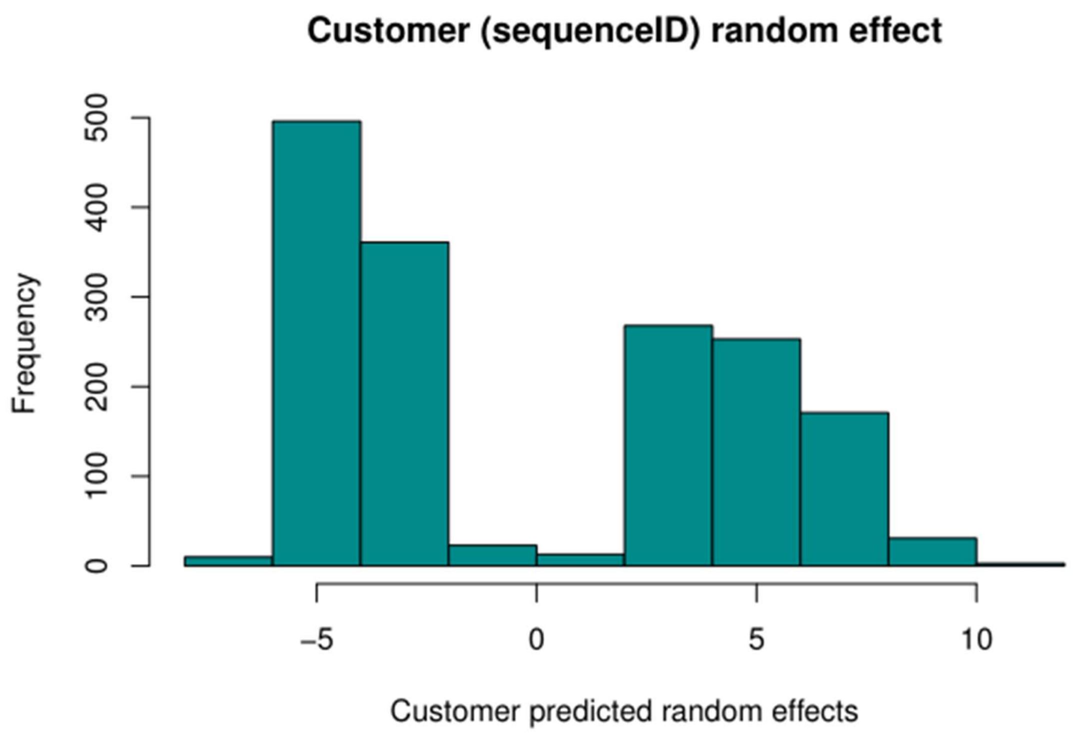


Figure 18 Distribution of the random effect for sequenceID (fitlrmm11) displays a bimodal shape

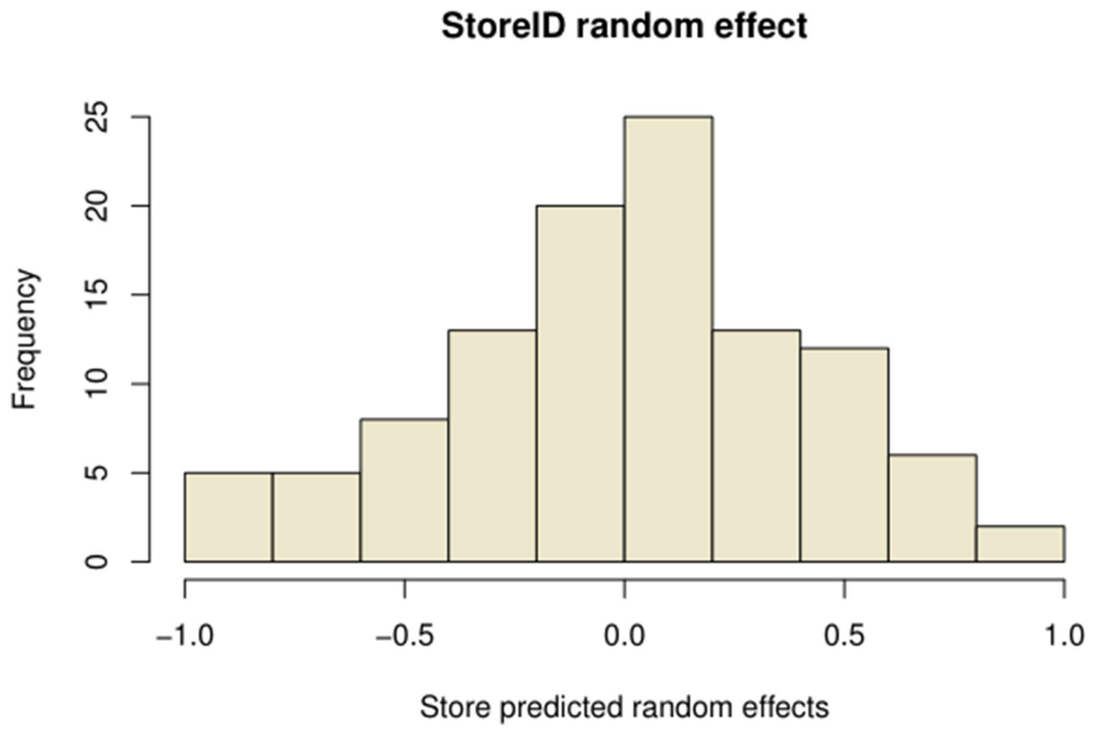


Figure 19 Distribution of the random effect for StoreID (fitlrmm11) displays a normal shape

3.3 Survey Data Results

3.3.1 Word Clouds

Figures 20 and 21 below show both of the word clouds generated.

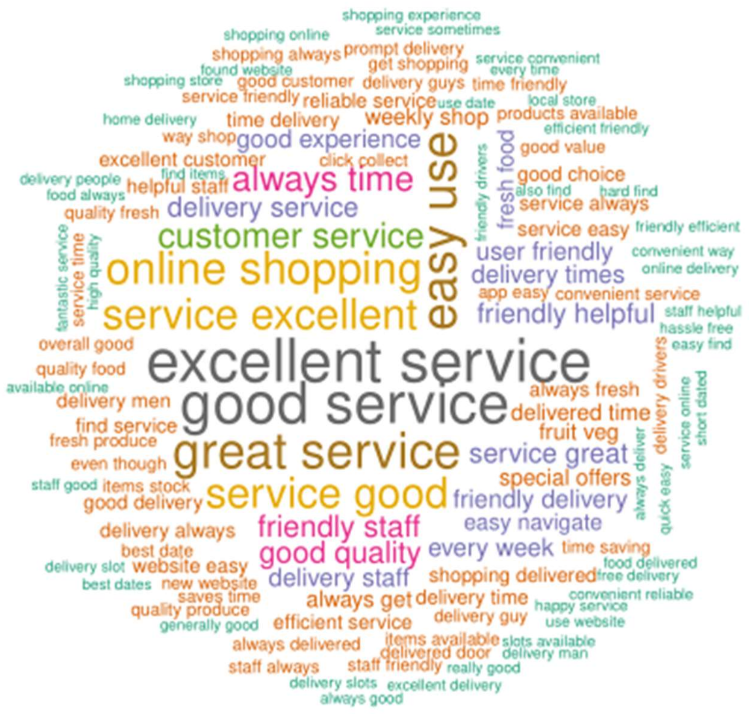


Figure 20 A word cloud showing the most popular two-word combinations using all comments from the survey data

The word cloud in Figure 20 above shows an overwhelmingly positive sentiment coming from the survey. Service - ranging from good to great, to excellent, is the most prominent comment.

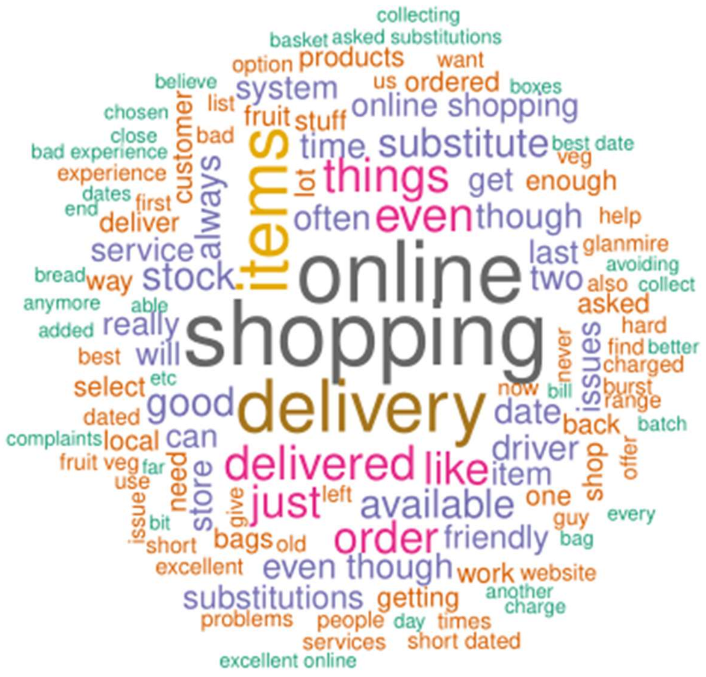


Figure 21 A word cloud showing the most popular one and two-word combinations (using only the comments associated with a rating of 5 or less).

There were 47 ratings of 5 or less in the survey. Figure 21 above shows the word cloud for the comments associated with these particular surveys. There are some negative comments creeping into the extremities of the word cloud. “Bad experience”, “problems”, “bad” and “complaints” all feature, and we do not see words like “excellent”, “great” or “good” featuring as the most popular phrases at the centre of the cloud.

3.3.2 Customer Ratings

A histogram showing the rating given by customers as to whether they would recommend SuperValu based on their experience, is shown in Figure 22 below.

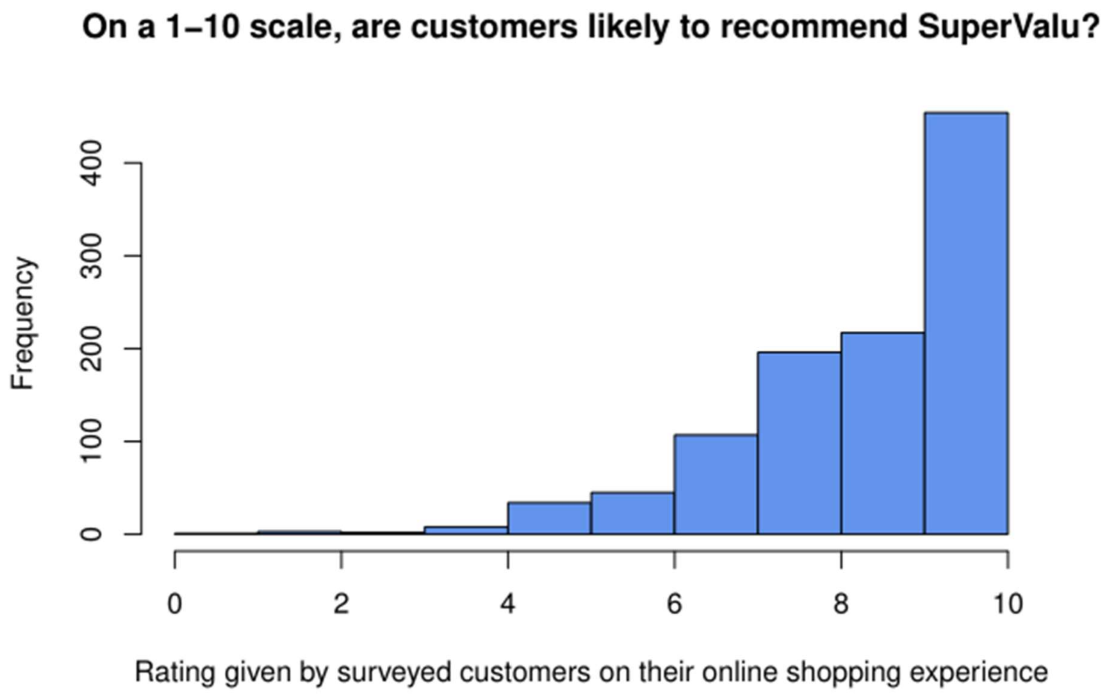


Figure 22 Histogram showing the likelihood of a customer recommending SuperValu, based on their online shopping experience

Out of 1,125 surveys, 867 customers answered an “8, 9 or 10” as to whether they would recommend SuperValu to a friend or colleague. This is a very positive response. From a negative perspective, 11 customers gave a rating of 4 or less. Of these, 3 are now dormant and 8 are still active. What about those dormant customers who gave a poor rating in their survey? Did they stop shopping immediately or continue? Table 23 below shows that 3 dormant customers gave a rating of 4 or less. However, they all went on to shop again subsequent to giving a lacklustre rating, showing that they still have loyalty to the SuperValu brand.

sequenceID	Propensity to recommend to a friend	Transactions made in total	Transactions made after survey
498	3	19	5
572	4	2	1
631	2	4	3

Table 23 Dormant customers who gave poor ratings, showing how many more transactions they made post-survey

4 Discussion

4.1 What is the "best" model?

Table 24 below shows a summary of the two best models:

Model	Test Error Rate %	AIC	Degrees of Freedom
fitlrmm11	18.35%	3099	18
fitlrmm13	19.5%	3089	19

Table 24 Summary of results for the two best performing models

They are similar. fitlrmm11 has the lowest error rate and a smaller number of parameters but the interaction in fitlrmm13 is significant and thus gives a lower AIC value. However, fitlrmm11 was ultimately chosen as the best model - parsimony is always a good strategy to adhere to and one of the main project goals is predictive power in a model, so the slightly lower test error rate is preferable. In summary, the best predictors of "Churn" are the amount charged, the product lines purchased, the month when the shopping is done and the substitution policy choice.

4.2 April – the most likely month for 'Churn'

April would be regarded as a significant month for churning in the grocery retail sector. Easter typically falls in April. Schools are off, so families may be away or out of their regular online shopping routine. Online shopping can be very routine dependant, you need to know where you're going to be (for reserving delivery slots ahead of the time etc) and school holidays disrupt that.

According to the Musgrave contact, the loyalty team issue money back vouchers in the month of April. This may lead to customers going in-store to redeem these vouchers (which the data does not capture) or choosing to wait and spend them online in May which may account for the bounce back effect we see in the model estimates. A lot of churn is typically seen in January too, for the same reason – people are away, eating out, picking up food in convenience stores and there are also limited online shopping slots. The reference level for factor Month was modified so that this 'January' effect could be seen. However, this trend is not seen in the model but this may be because there is only one year of data for current shoppers and two years for dormant shoppers.

4.3 Churn Threshold Analysis

In the course of the project, there was an idea to explore the threshold of a customer moving to dormant status after 70 days of inactivity. Using the mixed model as described in Section 3.2.3, could the threshold be increased or decreased e.g. to 50 days or 90 days and would this have a better or worse effect on the test error rate when using the model? The question of interest is: is 70 days the optimum threshold in this sort of retail environment? Some attempts were made to "move" customers from current to dormant or vice versa depending on these new threshold targets. However, it was recognised that this sort of approach was risky and unsafe for the following reasons. There is a window of time (June 2018 to Feb 2019) with which we have no visibility into, and it is unsafe to guess how customers shopped in this time period. Furthermore, following consultation with Musgrave, some clarification was given on how a customer transitions from current to dormant. The 70 day churn doesn't necessarily apply to all customers. A dynamic model is used that takes into account a customer's shopping frequency when calculating how long they must be inactive for before they are deemed dorment. Plus, there are customers in the dataset who are not yet "established" - customers as described in Section 3.1.10, so it makes little sense in trying to predict their behaviour. Following these exploratory works, it was concluded that this project data needed to be treated as simply a point in time and extrapolation would not be wise.

4.4 Future Work

The work outlined in section 3.2.2 which looked at customers whose spending appeared to be decreasing could benefit from more analysis. It was a manual process to observe the trends in the transactions of the regular customers but a more automated approach which captures these decreasing trends could be valuable. The focus could be on creating a clustering technique to group customers prone to leave and/or return and this could aid Musgrave in allocating targeted incentivised vouchers etc. Research could also be done to see if a Non-Parametric Maximum Likelihood model referenced in section 3.2.4 would be a suitable candidate for this data.

5 Conclusions

Running ten different Machine Learning techniques on the aggregated data was beneficial in acquiring data analysis skills. However, most of the techniques produced very poor error rates. Essentially, the data is being condensed down to a couple of thousand observations. Logistic regression yielded the best result but only under the condition of the removal of the one-time shoppers. This is obviously introducing a bias into the data. The mixed model analysis was more successful and error rates of 1% and 18.35% (for existing and new customers respectively) is a very satisfactory outcome. (The assumption violation must of course be conceded). In effect, this means the model could be re-used on another set of Musgrave customers and be capable of making correct predictions 81% of the time. The literary review provided helpful guidance in formulating an approach for best model selection and also looking for trends of partial defection among customers. The model captured that April was a very strong month for churning, with Musgrave later confirming this phenomenon. It was, indeed, valuable to have a support contact for the project duration to help with queries relating to outlier transactions, verify the status of certain customers outside of the data window and affirm some of the model findings.

The word clouds from the survey data reflect an altogether positive online shopping experience. There are just mere hints of negativity even from those who rated the service poorly. 47 out of 1125 surveys (4.2%) had a rating of 5 or less as to whether they would recommend SuperValu. These surveys appear to be customers conveying honest feedback at a point in time and while they may be disappointed about a minor aspect of their shopping experience, they are still largely happy to remain a customer.

On a final note, the results overall may have had greater appeal to Musgrave if the data had consisted of established customers only i.e. those that had shopped four times or more. 22% of the dataset are one-time shoppers and to build a propensity-to-churn model for loyal customers who shop regularly and spend a consistent amount on their shopping would surely be more advantageous to the retailer.

Bibliography

BATES, D., MACHLER, M., BOLKER, B. & WALKER, S. 2007. Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software,* 67.

BATES, D., MAECHLER, M., BOLKER, B. & WALKER, S. 2015. Fitting Linear Mixed-Effects Models Using lme4. 67**,** 1-48.

BOLKER, B. 2019. *Drop All Possible Single Fixed-Effect Terms From A Mixed Effect Model* [Online]. Available: https://www.rdocumentation.org/packages/lme4/versions/1.1-21/topics/drop1.merMod [Accessed].

BOUCHET-VALAT, M. 2019. SnowballC: Snowball Stemmers Based on the C 'libstemmer' UTF-8 Library. R package version 0.6.0.

BUCKINX, W. & VAN DEN POEL, D. 2004. Customer base analysis: partial defection of behaviorally loyal clients in a non-contractual FMCG retail setting. *European Journal of Operational Research,* 164 (2005)**,** 252-268.

CHENG, J., EDWARDS, L. J., MALDONADO-MOLINA, M. M., KOMRO, K. A. & MULLER, K. E. 2009. Real longitudinal data analysis for real people: Building a good enough model. *Statistics in Medicine,* 29**,** 504-520.

FABOZZI, F., FOCARDI, S., RACHEV, S. & ARSHANAPALLI, B. 2014. *The Basics of Financial Econometrics: Tools, Concepts, and Asset Management Applications* John Wiley and Sons.

FEINERER, I. & HORNIK, K. 2018. *tm: Text Mining Package. R package version 0.7-6* [Online]. Available: https://CRAN.R-project.org/package=tm [Accessed].

FELLOWS, I. 2018. *wordcloud: Word Clouds. R package version 2.6.* [Online]. Available: https://CRAN.R-project.org/package=wordcloud [Accessed].

GREENWELL, B., BOEHMKE, B., CUNNINGHAM, J. & GBM DEVELOPERS. 2019. *gbm: Generalized Boosted Regression Models. R package version 2.1.5.* [Online]. Available: https://CRAN.R-project.org/package=gbm [Accessed].

HORNIK, K., BUCHTA, C. & ZEILEIS, A. 2009. *Open-Source Machine Learning: R Meets Weka.* [Online]. Available: http://doi.org/10.1007/s00180-008-0119-7 [Accessed].

JAMES, G., WITTON, D., HASTIE, T. & TIBSHIRANI, R. 2013. *An Introduction to Statistical Learning,* New York, Springer.

KUHN, M. 2018. *caret: Classification and Regression Training. R package version 6.0-81.* [Online]. Available: https://CRAN.R-project.org/package=caret [Accessed].

LI, S. 2017. *Predict Customer Churn - Logistic Regression, Decision Tree and Random Forest* [Online]. Available: https://datascienceplus.com/predict-customer-churn-logistic-regression-decision-tree-and-random-forest/ [Accessed].

LIAW, A. & WIENER, M. 2002. Classification and Regression by randomForest. *R News,* 2**,** 18-22.

MEYER, D., DIMITRIADOU, E., HORNIK, K., WEINGESSEL, A. & LEISCH, F. 2019. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group* [Online]. Available: https://CRAN.R-project.org/package=e1071 [Accessed].

MOLENBERGHS, G. & VERBEKE, G. 2000. *Linear Mixed Models for Longitudinal Data,* New York, Springer-Verlag.

MUGGEO, V. M. R. 2008. *Segmented Relationships In Regression Models* [Online]. Available: https://www.rdocumentation.org/packages/segmented/versions/1.0-0/topics/segmented [Accessed].

NEUWIRTH, E. 2014. *RColorBrewer: ColorBrewer Palettes. R package version 1.1-2.* [Online]. Available: https://CRAN.R-project.org/package=RColorBrewer [Accessed].

R CORE TEAM. 2018. *R: A language and environment for statistical computing.* [Online]. Available: https://www.R-project.org/. [Accessed].

SHOPIFY. 2019. *What is Ecommerce?* [Online]. Available: https://www.shopify.com/encyclopedia/what-is-ecommerce [Accessed].

THERNEAU, T. & ATKINSON, B. 2018. *rpart: Recursive Partitioning and Regression Trees. R package version 4.1-13.* [Online]. Available: https://CRAN.R-project.org/package=rpart [Accessed].

VENABLES, W. N. & RIPLEY, B. D. 2002. *Modern Applied Statistics with S.,* New York, Springer.

VU, V. Q. 2011. *ggbiplot: A ggplot2 based version of biplot.* [Online]. Available: http://github.com/vqv/ggbiplot [Accessed].

WEI, T. & SIMKO, V. 2017. *corrplot: Visualization of a Correlation Matrix* [Online]. Available: https://github.com/taiyun/corrplot [Accessed].

WICKHAM, H. 2016. *ggplot2: Elegant Graphics for Data Analysis,* New York, Springer-Verlag.

Appendices

Four R Markdown script files are included as appendices:

1. *DataExploration.Rmd*: Exploratory analysis with the current/dormant datasets.
2. *AggregatedRegression.Rmd*: Apply various ML methods.
3. *FullDataMixedModel.Rmd*: Apply a series of mixed models.
4. *SurveyDataAnalysis.Rmd*: Sentiment analysis.