Erasmus University Rotterdam

Erasmus School of Economics

Predicting the Next Action of the Customers at Allianz Direct

Wayne Wan Shing Dian (570590)

Supervisor: Prof. Richard Paap

Supervisor Allianz Direct: Garry Martes

August 23, 2021

Abstract

Being in a competitive environment, Allianz Direct is interested to know the next action of their customers on their motor insurance policies. Six models are implemented in this research to find the best model in classifying the next action of the customers. The models are Multinomial Logit (without feature selection), Multinomial Logit (with feature selection), XGBoost, Random Forest, Support Vector Machine and Random Multinomial Logit. The models are evaluated based on their accuracy and F1-score. The machine learning models - XGBoost, Random Forest and Support Vector Machine have shown to outperform the two Multinomial Logit models and Random Multinomial Logit model. Furthermore, XGBoost is the best model with the highest accuracy and F1-score. For this reason, XGBoost is selected by the company to be deployed for real world use.

Contents

1 Introduction

Individuals purchase insurance to ensure they are protected from unexpected risk. In many countries, including the Netherlands, every vehicle owner is even required by law to be insured. In the age of the internet, there exist insurance aggregator websites which allow customers to compare quotation prices from different insurers and this makes it easy for the customers to pick the insurer who provides the best value for money. Typically, customers are offered an annual contract to "lock-in" the price. However, there are often "next action" or changes made to the contracts after their commencement. For instance, since there is no cancellation penalty, the customers are free to cancel their policies or switch to a different insurer at anytime if they find a more suitable product. Further, customers often make changes to their policies such as buying add-ons or cancelling add-ons after learning more about their own needs. All these actions affect the recuperation of the acquisition cost and also affect the process of renewal pricing if there are more than expected cancellations. This in turn will have a direct and significant impact on the revenue of the insurance company.

This research is conducted for Allianz Direct - an auto insurer which provides their insurance products on the internet without the need of any intermediary brokers. Considering that not everybody has the same level of risk, it is important to calculate and charge the customers according to their level of risk to prevent adverse selection. Similar to all other auto insurers, Allianz Direct mainly uses Frequency model, Severity model, Conversion model and Retention model to calculate their insurance tariff. These models are Generalized Linear Models (GLM) which assume non-Gaussian distribution of residuals. The type of distribution depends on the structure of the data. For instance, Dionne and Vanasse (1989) has shown that to estimate the frequency of claims, the Frequency model assumes Poisson error structure and Pinquet (1997) has shown that to estimate the severity of a claim, the Severity model assumes Gamma error structure.

Being in a dynamic and competitive environment, it is of Allianz Direct's best interest to know the next action of their customers and ultimately relay this information into their pricing process. Hence, Allianz Direct is interested to develop a new model which is the Next Action model which predicts the next action of the customers on their motor insurance policies. The Next Action model can be seen as an extension to the Retention model. The latter only predicts whether a policy lapses or not (binary classification), while the former predicts other actions which include lapses (multi-class classification). Allianz Direct then aims to incorporate this new model into their pricing process and monitoring reports. The way to incorporate the model will be the sub-question of this research, with building the new model to classify the next action of the customers as the main focus. The possible next actions of the customers are *Renewal*, *Upscale*, *Downscale*, *Change of Vehicle*, *Cancel at Midterm*, *Cancel at Renewal* and *Others*. These actions are deemed by the pricing team to

be the most interesting and informative that are most likely to have an impact on the tariff.

There are many possible classification models in the literature but not all are suitable for this research. For instance, k-nearest neighbours (Fix, 1985) is a simple and quick non-parametric classification method. However, it was shown in Behera, Kumaravelan, and Kumar (2019) that k-nearest neighbours does not classify well and has a low accuracy in high dimensional settings. Thus, it is not suitable for this research since auto insurance data consists of many variables and levels which may lead to a high dimensional problem. A few models are explored in this research to find the best performing model to solve this multi-class classification problem. The models that will be explored are Multinomial Logit, Random Forest (Breiman, 2001), XGBoost (Chen & Guestrin, 2016), Multi-class Support Vector Machine (Mayoraz & Alpaydin, 1999), and Random Multinomial Logit (Prinzie & Van den Poel, 2008). The Multinomial Logit model - a classical econometric model which can also be seen as a Generalized Linear Model with Poisson error structure (Log link) will act as a benchmark against all the other machine learning models. The models will be assessed by a few popular assessment measures used for multi-class classification.

A few of the popular assessment measures for classifiers will be used to evaluate and find the best model. The assessment measures are Accuracy, Precision, Recall, F1-score and Area under the curve (AUC). Since F1-score is a combination of Precision and Recall, F1-score will be used as a primary assessment measure together with accuracy. AUC will be used as a supplementary assessment measure. Once the best model is determined, an one-dimensional factor analysis will be performed on this model to investigate how the predicted next actions from the model perform against the actual next actions at the levels of a selected independent variable.

The practical application of the Next Action model for Allianz Direct will also be explored in this research. There are currently three ways the Next Action model can be used. First, it will be used to gather a list of policies which are predicted to be downscaled and send the list to the marketing team for further actions. Second, it will be used together with an existing competition monitoring report to focus on policies which are predicted to be cancelled. Lastly, it will be used to generate monitoring reports and ideally incorporate the model into the pricing process in the future.

At the end of this research, the following questions will be answered:

- Which model is the best in predicting a customer's next action for a given policy?
- Do Machine Leaning (black-box) models provide justifiable improvement in accuracy and F1-score over more interpretable classical econometric models (GLM)?
- Does Random Multinomial Logit performs better in terms of accuracy and F1-score

over Multinomial Logit?

- Does Multinomial Logit with feature selection perform better than Multinomial Logit without feature selection in terms of accuracy and F1-score?
- How to incorporate the best model for real world use for Allianz Direct?

The next section of this paper will discuss in detail the data used in this research. Section 3 will discuss the models used and their methodology. Section 4 will discuss the assessment measures used to evaluate the models. Section 5 presents the analysis and results. Section 6 describes how the best performing model will be deployed into practical use for the company. Lastly, Section 7 concludes this research paper.

2 Data

This section discusses the data that is used for this research. The data is provided by the actuarial pricing team at Allianz Direct. The data is at policy level and it consists of policy information of clients who have a contract between 10 September 2019 and 23 March 2021. Further, each policy is tied to a unique vehicle and a customer may have multiple policies. Thus, since the data is at policy level, I am investigating specifically from the perspective what action a customer will make for a particular policy.

Towards the expiration of a contract, Allianz Direct sends a new quotation proposal to the customer; the customer then has the options to cancel the contract, upscale or downscale the contract based on their preference, or accept the new quotation proposal and renew the contract. In addition, upscaling and downscaling a contract may also occur anytime during the tenure of the contract. Further, a customer may also decide to cancel the contract anytime during the tenure of the contract. A customer is also able to make amendments such as changing the vehicle or change of address which can have an effect on the insurance premium. Hence, with these actions in mind combined with feedback from the pricing team, the dependent variable, which is a multi-class categorical variable; is defined with the following possible classification for each policy, *Renewal*, *Upscale*, *Downscale*, *Change of Vehicle*, *Cancel at Midterm*, *Cancel at Renewal* and *Others*. Some of the possible actions in *Others* are change of zipcode, change of mileage, change of main driver and change of number of drivers. These actions have a direct impact on the tariff pricing but they are currently not captured explicitly by the system used by Allianz Direct, thus they are grouped as one action in this research. It was decided to keep this class (*Others*) as one of the possible next action for two reasons. Firstly, from Table 1, it can be seen that it occupies slightly more than 5% of the total frequency. Secondly, Allianz Direct is interested to dive deeper into this particular group in the future for potential insights. Examining Table 1, it is obvious that the distribution of the dependent variable is imbalanced. However, since each of the next actions have at least 5% of the total frequency,

it was decided to use the original data as it is without using any data re-balancing technique.

Furthermore, the data consists of 149,386 policy records with 45 independent variables. The independent variables can be split into 3 categories, individual characteristics, car specifications and address information (zipcode, degree of urbanisation). Individual characteristics consist of variables such as age, claim-free years, coverage type, etc. Car specifications consist of variables such as vehicle age, vehicle horsepower, vehicle price, etc. Address information consists of variables such as zipcode, degree of urbanisation, household income, etc. The full description of the data can be found in Appendix A.

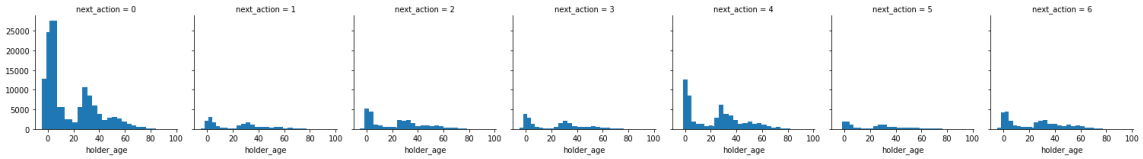
Table 1: Distribution of the dependent variable (next action)

Next_action	Frequency	Percent
Cancel_midterm	55627	37.24%
Cancel_renewal	10315	6.90%
Downscale	15557	10.41%
Others	11210	7.50%
Renewal	32801	21.96%
Upscale	7501	5.02%
Vehicle_change	16375	10.96%

This table shows the frequency and proportion of each class of the dependent variable (next action)

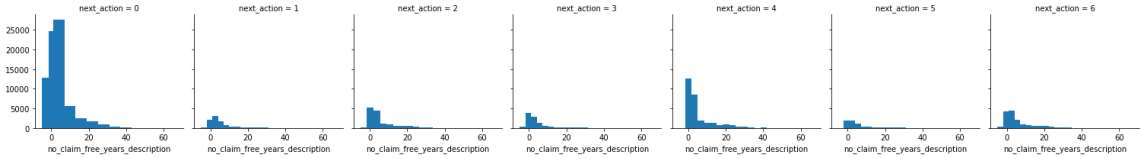
Three histograms are plotted to gain some insights on the correlation between the dependent variable (next action) and some selected independent variables. Age, claim-free years and gender are chosen because these factors are the "usual suspects" in pricing auto insurance. From Figure 1, there is no clear pattern separating the next actions. However, we are able to see that the majority of the policy holders are aged 18 and aged around 30. From Figure 2, the majority of the policies have claim-free years lesser than 5 years. Lastly from Figure 3, there is no clear distinction between the distribution of female and male policy holders. Nonetheless, there are slightly more male than female policy holders.

Figure 1: Correlation between the dependent variable (next action) and age



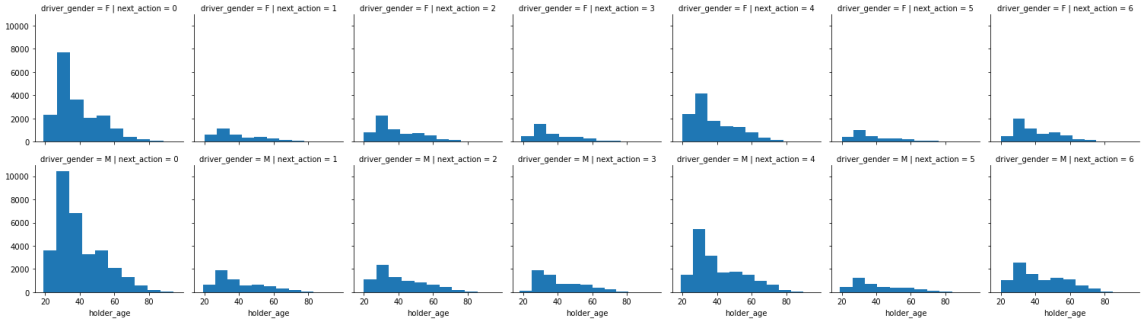
This figure shows the histograms of next action broken down by age. From left to right the next actions are *Cancel at Midterm*, *Cancel at Renewal*, *Downscale*, *Others*, *Renewal*, *Upscale* and *Change of Vehicle* respectively.

Figure 2: Correlation between the dependent variable (next action) and claim-free years



This figure shows the histograms of next action broken down by claim-free years. From left to right the next actions are *Cancel at Midterm*, *Cancel at Renewal*, *Downscale*, *Others*, *Renewal*, *Upscale* and *Change of Vehicle* respectively.

Figure 3: Correlation between the dependent variable (next action) and age + gender



This figure shows the histograms of next action broken down by age and gender. From left to right the next actions are *Cancel at Midterm*, *Cancel at Renewal*, *Downscale*, *Others*, *Renewal*, *Upscale* and *Change of Vehicle* respectively. The histograms on the top are female and the bottom histograms are male.

3 Methodology

This section provides an overview of the methods which will be used in this research. I start by discussing Multinomial Logit. Followed by introducing the concept of decision trees, CART (Breiman, Friedman, Olshen, & Stone, 1984) and Gradient Boosting Machine (Friedman, 2001) which are crucial to understand two of the methods - XGBoost and Random Forest. Further, I will discuss Support Vector Machine and its extension to multiclass classification. Lastly, I will discuss Random Multinomial Logit (Prinzie & Van den Poel, 2008) which is a combination of Multinomial Logit and Random Forest.

3.1 Multinomial Logit

The first model that is considered in this research is a classical econometric model - Multinomial Logit. Multinomial Logit is a generalisation of logistic regression when the dependent variable has more than two possible outcomes. In addition, Multinomial Logit can also be seen as an extension of Generalized Linear Models (GLM) by allowing unordered categorical response. Recall that in this research, there are seven next action classes defined as,

$$y_i = \begin{cases} 1 & \text{if Cancel at midterm,} \\ 2 & \text{if Cancel at renewal,} \\ 3 & \text{if Downscale,} \\ 4 & \text{if Others,} \\ 5 & \text{if Renewal,} \\ 6 & \text{if Upscale,} \\ 7 & \text{if Change of Vehicle} \end{cases} \tag{1}$$

where y_i is the next action label of a policy. Further, with β_k as the set of regression coefficients associated with outcome k, and $x_i^{'}$ as the set of independent variables with $i = 1, \ldots, N$, the probabilities can then be generalised into the following equation for K levels of the dependent variable y,

$$P(y_i = k|x_i) = \frac{e^{x_i^{'}\beta_k}}{\sum_{k=1}^{K} e^{x_i^{'}\beta_k}} \tag{2}$$

For identification purpose, I set $\beta_K = 0$ which results in the following equation,

$$P(y_i = k|x_i) = \frac{e^{x_i^{'}\beta_k}}{1 + \sum_{k=1}^{K-1} e^{x_i^{'}\beta_k}} \tag{3}$$

with $\sum_{k=1}^{K} P(y_i = k|x) = 1$ where one outcome is chosen as a "reference" and the other

$K-1$ outcomes are regressed separately against the "reference" outcome.

The common way to estimate the parameters of a Multinomial Logit model is to use maximum likelihood estimation (MLE). Given $y_1,\ldots,y_N$, the log-likelihood function is,

$$\ell_{MNL}(\boldsymbol{\beta}) = \sum_{i=1}^{N}\sum_{k=1}^{K} I[y_i = k] \ln P[y_i = k|x_i] \tag{4}$$

where $I[.]$ is an indicator function which equals to 1 if the argument is true and 0 otherwise.

Lastly, it is worth noting that the Multinomial Logit model in this research is only intended to be used as a classification method without going into the random utility component of the model.

3.2 Tree Based Methods

3.2.1 Decision Tree & CART

Prior to discussing XGBoost and Random Forest, it is important to first introduce the idea of decision trees and Classification And Regression Trees (CART) (Breiman et al., 1984).

The fundamental idea of a decision tree is to create a model or find a function $\hat{f}(X)$ which uses independent variables X to predict the dependent variable Y. Decision trees consist of nodes and branches. The very first node is called the root node, the intermediary nodes are called internal nodes and leaf nodes are the terminal node on the tree without any further splitting. At each of the leaf nodes is the predicted outcome (next action) and the internal nodes are decision rules based on a certain cost function used to split the tree.

In the process of building a decision tree, two of the most important aspects are which variable should be used to split the tree and when to stop the splitting. There are a few popular existing algorithms used to tackle these aspects. Two of the most used algorithms are CART and C4.5 (Quinlan, 1986). In this research, I will focus on CART which will be used in XGBoost and Random Forest subsequently.

During the splitting procedure, different splits are tried based on different variables by using a cost function and the split is determined by the variable with the lowest cost function. The cost function used in CART is called the Gini index. Gini index informs us what is the chance or probability of misclassifying an observation and thus the lower the value of Gini index, the better the split. The Gini index is defined as,

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{5}$$

where $\hat{p}_{mk}$ is the proportion of class k observations belonging in node m out of the total observations in node m (Hastie, Tibshirani, & Friedman, 2009, p.305). In order to minimize the impurity of the tree, CART chooses the splits with the largest decrease in Gini index.

However, this splitting procedure runs into the risk of over-fitting when there are a large number of independent variables which lead to a large number of possible splits. There are a few stopping criteria which can be used to tackle this problem. One of the criteria is to set the minimum number of training observations required at each leaf (terminal) node. Furthermore, it is also possible to set the maximum depth the tree is able to grow.

Despite these criteria, the resulting tree is likely to still be large and this can be resolved by pruning the tree. Tree pruning removes some sections of the tree which do not provide additional value and thus reduces the size of the tree and avoids the risk of over-fitting. If we denote $C(T)$ as the cost of tree T (training error), $|T|$ as the number of terminal nodes in tree T, then the cost complexity for parameter α is,

$$C_\alpha(T) = C(T) + \alpha|T| \tag{6}$$

where the optimal value for the cost complexity parameter α can be estimated via cross validation.

The downside of a single decision tree is that it is not robust. A tiny change in the data is able to cause a different split in the tree which affects all subsequent splits and thus affecting the whole structure of the tree. A solution to this problem is adopting ensemble methods. Ensemble methods involve building many trees and aggregating the predictions to produce results which are less susceptible to variation in the data and also producing better predictions. The next section discusses Gradient Tree Boosting which is one of the ensemble methods.

3.2.2 Gradient Tree Boosting (XGBoost)

The first ensemble method that I consider is Gradient Tree Boosting which uses gradient boosting. Unlike building a single decision tree as discussed in Section 3.2.1, the trees in Gradient Tree Boosting are built consecutively based upon the previous tree. Further, in gradient boosting such as the one proposed in Friedman (2001), the splits are determined by minimizing a differentiable loss function.

The boosting algorithm that will be used in this research is XGBoost introduced in Chen and Guestrin (2016). Unlike other gradient boosting, XGBoost employs parallel processing which leads to greater speed and performance. XGBoost introduces a regularization term

in the loss function which prevents over-fitting. Further, XGBoost has rules which automatically handle missing values. These are favorable for this research due to multi-class classification along with a relatively huge number of independent variables which may lead to computational issues caused by high dimensionality.

XGBoost has a loss function $\mathcal{L}$ which is a combination of a logarithmic loss function l and a penalty term Ω. The logarithmic loss function l calculates the error between the predicted next actions and the actual next actions. The penalty term Ω prevents the tree from over-fitting by penalizing for complexity. This gives us a *regularized* loss function $\mathcal{L}$,

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k), \quad f_k \in \mathcal{F} \tag{7}$$

where K is the number of trees and $\mathcal{F}$ is the function space of all the trees. For the case of multi-class classification, we have,

$$\begin{aligned} l(y_i, \hat{y}_i) &= \sum_{i=1}^{n} y_i^{'} \log(\hat{y}_i) \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} y_{ik} \log(\hat{y}_{ik}) \end{aligned}$$

where $y_i^{'}$ and $\hat{y}_i$ are vectors of binary indicators and predicted probabilities of each class respectfully. Further, $\hat{y}_{ik}$ is the predicted probability of next action class k and y_{ik} act as an indicator function with $y_{ik} = 1$ if the observation is of class k and 0 otherwise. The penalty term is defined as $\Omega(f_k) = \gamma T + \frac{1}{2}\alpha||\omega||^2$ where T refers to the total number of leaves in a tree and ω refers to the vector of leaf weights. Hence, the parameter γ penalizes according to the number of leaves and parameter α penalizes according to the leaf weights.

The loss function (Equation 7) cannot be optimized using traditional optimization methods and thus, it is optimized iteratively. For the t-th iteration, the loss function can be rewritten as,

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{8}$$

where the algorithm greedily finds a tree f_t which minimizes the loss function. A second order Taylor approximation can then be used to optimize the objective function which gives us the following equation,

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{9}$$

where $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ are the first and second order gradient statistics of $l(y_i, \hat{y}_i^{(t-1)})$. By removing the constant terms, we can simplify equation 9 into,

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \tag{10}$$

By substituting the penalty term Ω into equation 10, and f_t having T leaf nodes, I_j be the set of instances belonging to node j and w_j as the prediction for node j, we get the following equation,

$$\tilde{\mathcal{L}}^{(t)} = \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \alpha) w_j^2] + \gamma T \tag{11}$$

Solving equation 11 with respect to w_j gives us the following optimal weight,

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \alpha} \tag{12}$$

Substituting the optimal weight, w_j^* back into equation 11 gives us the following equation to calculate the corresponding optimal value,

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^{T} \frac{(\sum_{i \in I_j} g_i)^2}{\sum_{i \in I_j} h_i + \alpha} + \gamma T \tag{13}$$

where q is the tree structure (decision rule). As explained in the original paper (Chen & Guestrin, 2016), it is usually impossible to enumerate all the possible structure q. Hence, a greedy algorithm is used instead, starting from a single leaf and adding branches to the tree iteratively. Assuming I_L and I_R are the instance sets of left and right nodes respectively after the split and $I = I_L \cup I_R$, then the split with the best loss reduction is,

$$\mathcal{L}_{split} = \frac{1}{2} [\frac{(\sum_{i \in I_L} g_i)^2}{\sum_{i \in I_L} h_i + \alpha} + \frac{(\sum_{i \in I_R} g_i)^2}{\sum_{i \in I_R} h_i + \alpha} - \frac{(\sum_{i \in I} g_i)^2}{\sum_{i \in I} h_i + \alpha}] - \gamma \tag{14}$$

This loss reduction is then used to evaluate the split candidates, thus defining the tree structure.

3.2.3 Random Forest

The second decision tree ensemble method that I consider in this research is Random Forest (Breiman, 2001). Random forest uses the idea of bootstrap aggregating (bagging) to grow a large collection of de-correlated trees and averaging the predictions or apply majority rule to reduce the variance. Since this research is a classification problem, majority rule will be applied to decide for a predicted class. Each tree in the ensemble "forest" uses a different bootstrapped sample to make predictions. In addition, each tree is built using Classification and Regression Tree (CART) algorithm as discussed in section 3.2.1.

In a single decision tree, each node split is determined by considering all variables. However, in random forest, each node is split by using only a subset of the variables. The random forest algorithm is demonstrated in Algorithm 1 (Hastie et al., 2009, p.588).

Algorithm 1 Random Forest algorithm

1: **for** $b = 1$ `to` B **do**
2: Draw a bootstrap sample of size N from the training data
3: Grow a deep random forest tree T_b with the bootstrapped data by repeating the following steps recursively for each terminal node of the tree, until the minimum node size n_{min} is achieved:
4: Select m variables from p variables
5: Choose the best split point among the m variables
6: Split the nodes into two daughter nodes
7: **end for**
8: Output the ensembles of trees $\{T_b\}_1^B$
9: To make prediction for x_{n+1}
10: Let $\hat{C}_b(x)$ be the prediction of the b-th random forest tree. Then $\hat{C}_{rf}^B(x)$ = majority vote $\{\hat{C}_b(x)\}_1^B$.

It was shown in the original paper (Breiman, 2001) that by fitting an ensemble of trees, the predictions are less volatile and more accurate compared to the prediction by a single CART tree.

3.3 Support Vector Machine

The fourth model that I consider for this research is Support Vector Machine (SVM) introduced by Cortes and Vapnik (1995). SVM is a binary classifier which uses the training data to find an optimal hyperplane which separates the data between two classes.

Given the feature vector $\boldsymbol{x}_i \in \mathbb{R}^p$ and target labels $y_i \in \{-1, 1\}$, SVM aims to find a hyperplane which minimizes mis-classification by maximizing the margin between the hy-

perplane and the nearest data points (support vector) of each class. A hyperplane is defined as $x_i^T\beta + \beta_0 = 0$ where β is a unit vector to the hyperplane and β_0 is an intercept. The margin lines from each class are defined as $x_i^T\beta + \beta_0 - \Delta = 0$ and $x_i^T\beta + \beta_0 + \Delta = 0$. With scaling, Δ can be set to be equal to 1. The objective function can be written as follows,

$$\min \quad \sum_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N}\xi_i \tag{15a}$$

$$\text{subject to} \quad y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \quad \forall i, \tag{15b}$$

$$\xi_i \geq 0, i = 1, \ldots, n \tag{15c}$$

where C is the penalty term and ξ is the proportional amount by which the prediction is on the wrong side of the hyperplane (Hastie et al., 2009, p.418).

However, it is often the case that the data is not perfectly separable by a linear hyperplane. SVM tackles this issue by using a mapping function ϕ. The mapping function ϕ projects the input data onto a feature space of higher dimension by using a kernel function which transforms non-linearly separable data into linearly separable data as shown in Figure 4. In order to do that, we first rewrite equation 15 by using Lagrangian function where we get the following Lagrangian primal problem,

$$L(\beta, \beta_0, \alpha) = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{N}\alpha_i[y_i(x_i^T\beta + \beta_0) - 1] \tag{16}$$

where α_i is the Lagrange multiplier. Equation 16 can be simplified into its Lagrangian dual form,

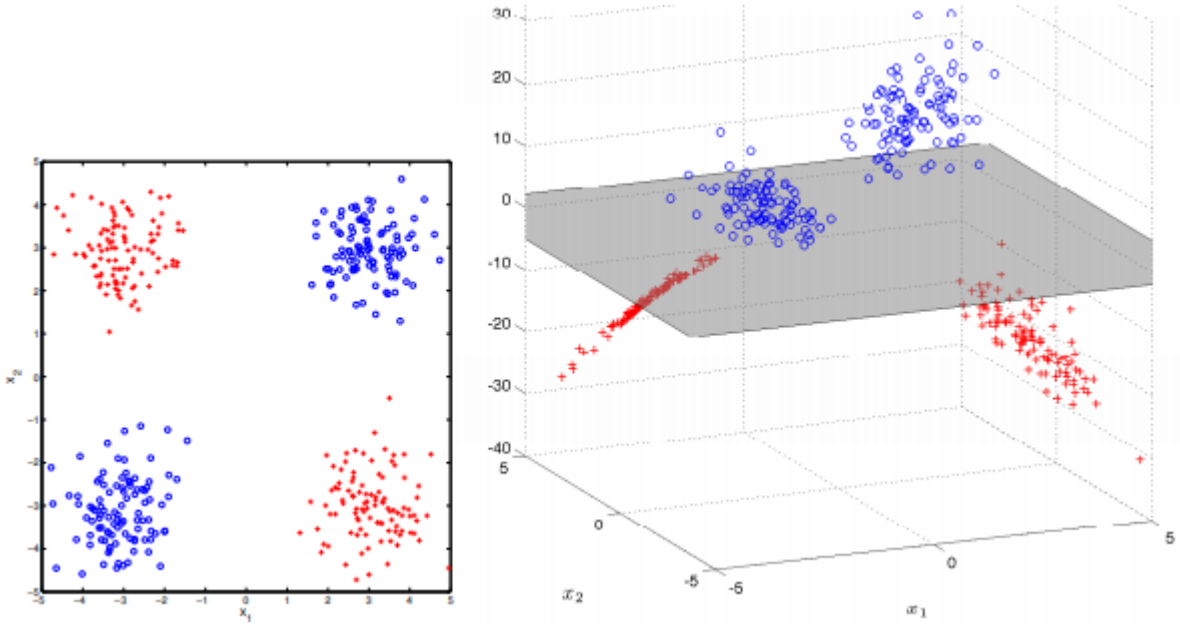
$$L(\beta, \beta_0, \alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j x_i^T x_j \tag{17}$$

To solve equation 17, we only need to solve the inner product of $x_i^T x_j$. The inner products can be replaced by a mapping function $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \boldsymbol{\phi}(x_i)^T\boldsymbol{\phi}(x_i)$. These mapping functions are known as kernel functions.

The kernel function is used to project the data onto a higher dimension feature space. Some of the popular kernels are linear, polynomial, radial basis function (RBF) and sigmoid. Multiple papers such as Yekkehkhany, Safari, Homayouni, and Hasanlou (2014), Hsu, Chang, Lin, et al. (2003) and Guernine and Zeroual (2011) have shown that RBF kernels work well and it is a reasonable first choice for a kernel. Hence, a RBF kernel will be used in this research which has the form of $K(\boldsymbol{x_i}, \boldsymbol{x_j})^{\text{RBF}} = e^{-\gamma\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2}$.

Further, there are two hyperparameters that require tuning which are C and γ. These hyperparameters are shown to be important for the performance of SVMs (Tay & Cao, 2001). C is the penalty term as shown in equation 15a which determines how much training data are allowed to be mis-classified. In general, a large value tries to classify the training data accurately, whereas a low value results in a smoother surface, thus, it is a trade-off. On the other hand, γ determines how far the influence of the training data can reach. where a large value indicates 'close' and a small value indicates 'far'.

Figure 4: Projecting data to a high-dimensional space with a Kernel function (Gretton, 2013)



The left figure shows that there is no possible linear hyperplane to separate the red crosses from the blue circles. The right figure shows that by mapping the data to a higher dimension, we can obtain a linear separable hyperplane (gray plane)

The SVM as discussed above is originally a binary classification technique and it has proven to be very accurate for binary classification (Cortes & Vapnik, 1995). Several methods have been made to extend SVM beyond binary classification. The most popular extensions are the heuristic approaches, one-vs-all (OvA) (Vapnik, 1998) and one-vs-one (OvO) (Krebel, 1999). As demonstrated in Hsu and Lin (2002) and Milgram, Cheriet, and Sabourin (2006), OvO approach is more practical and widely used due to much faster training process. Hence, in this research, I adopt the OvO approach. Assuming there are k

possible classes, then OvO requires to construct $k(k-1)/2$ SVMs for each pair of the classes.

3.4 Random Multinomial Logit

The fifth and last model to be considered in this research is Random Multinomial Logit (Prinzie & Van den Poel, 2008). For the classical Multinomial Logit model introduced in section 3.1, it may run into dimensionality issues when there are many independent variables and many different level of outcomes (Prinzie & Van den Poel, 2008).

To tackle this issue, Random Multinomial Logit adopts the strength and characteristics of Random Forest as discussed in section 3.2.3 and combines with Multinomial Logit models. Each Multinomial Logit b is built from a randomly bootstrapped sample N_b of size N drawn from the original data with replacement. Further, only a subset m out of the total M independent variables are selected to build each Multinomial Logit, which is another core feature of Random Forest. For classification, each Multinomial Logit model votes for its predicted class and then the bagged predictor is obtained by majority voting. The Random Multinomial Logit algorithm is shown in Algorithm 2.

One obstacle faced while implementing this method is that there are no existing software packages available for this model. Hence, I implement the model by programming it myself according to Algorithm 2 which is analogous to the algorithm described in the original paper (Prinzie & Van den Poel, 2008).

Algorithm 2 Random Multinomial Logit (RMNL) algorithm

1: **for** $b = 1$ `to` B **do**
2: Create bootstrap sample N_b of size N by drawing from the training data with replacement
3: Create out-of-bag (oob) sample oob_b by taking samples which were not included in N_b
4: Select m variables from the total independent variables
5: Build a Multinomial model by using bootstrap sample N_b and the selected m variables
6: Evaluate the model with oob_b by computing the Accuracy (Equation 21)
7: **end for**
8: Output the emsembles of Multinomial Logit models $\{MNL_b\}_1^B$
9: To make prediction for x_{n+1}
10: Let $\hat{C}_b(x)$ be the prediction of the b-th Multinomial Logit model. Then $\hat{C}_{rf}^B(x) =$ majority vote $\{\hat{C}_b(x)\}_1^B$.

4 Assessment Measures

In this section, I will first discuss about confusion matrix and followed by the most widely used assessment measures for classifiers - Accuracy, Precision, Recall, F1-score and Area under the curve (AUC) (Kumar, 2019). These assessment measures will be used to evaluate the models mentioned in section 3.

4.1 Confusion Matrix

The assessment measure that will be used for model evaluation rely on a confusion matrix as shown in Table 2.

A confusion matrix consists of the true class and the predicted class. The layout of the matrix table allows a simple yet useful visualisation of the performance of a model. True Positives (TP) indicate where the true class is positive and its predicted class is also positive; True Negatives (TN) indicate where the true class is negative and its predicted class is also negative; False Positives (FP) indicate where the true class is negative but its predicted class is positive; False Negatives (FN) indicate where the true class is positive but its predicted class is negative.

Table 2: Confusion Matrix for binary classification

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Even though the confusion matrix shown in Table 2 is for binary classification, with the same idea and logic, the matrix can be easily expanded for multi-class classification.

4.2 Accuracy, Precision, Recall and F1-score

After establishing the confusion matrix, we can now use it to define the assessment measures. Firstly, Accuracy computes the percentage of of samples correctly identified as either TP or TN out of the total sample. Second, Precision gives the percentage of a class actually belonging to that class. Third, Recall gives the percentage of TP that were correctly classified as positive by the model. Lastly, F1-score represents a comprehensive metric which combines Precision and Recall.

Based on the confusion matrix, the precision, recall and F1 score for each class are deter-

mined as follow where $i = 1, \ldots, K$,

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \tag{18}$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{19}$$

$$F1_i = 2 \cdot (\frac{Precision_i * Recall_i}{Precision_i + Recall_i}) \tag{20}$$

In order to know the overall prediction performance across all classes, we can take the average measurement over all classes. There are three popular averaging methods namely micro-average, macro-average and weighted-average. Since, we have a class-imbalanced dataset (Figure 1) and each class is equally important in this research, weighted-average is selected. The accuracy, weighted average of precision, weighted average of recall and weighted average of F1-score (Behera et al., 2019; Sokolova & Lapalme, 2009) are formulated as,

$$Accuracy = \frac{\frac{\sum_{i=1}^{K} TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{K} \tag{21}$$

$$Precision_{weighted} = \frac{(\sum_{i=1}^{K} N_i * (\frac{TP_i}{TP_i + FP_i})}{N} \tag{22}$$

$$Recall_{weighted} = \frac{(\sum_{i=1}^{K} N_i * (\frac{TP_i}{TP_i + FN_i})}{N} \tag{23}$$

$$F1_{weighted} = 2 \cdot (\frac{Precision_{weighted} * Recall_{weighted}}{Precision_{weighted} + Recall_{weighted}}) \tag{24}$$

Instead of assigning equal weights to each class, we adopt weighted average in this research where N_i denotes the number of samples belonging to class i and N denotes the total sample.

4.3 AUC - ROC Curve

The Receiver Operating Characteristic (ROC) curve is originally an assessment measure for binary classification. The ROC curve is a probability curve by plotting the True Positive Rate (TPR) $\frac{TP}{TP+FN}$ against the False Positive Rate (FPR) $\frac{FP}{TN+FP}$. On the other hand, Area Under the Curve (AUC) utilizes ROC curve to measure the degree of separability. It essentially tells how good a model is at separating between two classes. Hence, an AUC value close to 1 indicates the model has good separability.

To extend AUC and ROC curve to multi-class classification, we can adopt the One vs All technique. In other words, we compute multiple ROC curves by considering one class at a time and group all the other classes as the other label.

4.4 Model Evaluation

To evaluate the models, the primary assessment measures will be the Accuracy and F1-score since F1-score is a comibnation of Precision and Recall. AUC-ROC will be used as a supplementary assessment.

5 Results

In this section I will first discuss hyperparameter tuning of the models in section 5.1. In section 5.2, I will discuss the results of two Multinomial Logit models. One without feature selection and the other with feature selection. In section 5.3, the results from XGBoost will be discussed, followed by the results from Random Forest in section 5.4. Further, the results from Support Vector Machine will be discussed in section 5.5 and followed lastly by the results from Random Multinomial Logit in section 5.6. Upon discussing the results individually, I will discuss the overall results in section 5.7 and select the best performing model based on their accuracy and F1-score. Finally, I will discuss the one-dimensional factor analysis performed on the selected best model in section 5.8.

5.1 Hyperparameters Tuning

Prior to tuning the hyperparameters, the data is split into training and testing sets with the proportion of 70% and 30% respectively. During the split, the data is stratified to the dependent variable to ensure the training and testing sets retain the same proportion as the full data set. The models are then built using the 70% training data set and the 30% testing data set is kept as a hold-out sample to evaluate the models.

Similar to many other machine learning algorithms, XGBoost, Random Forest and Support Vector Machine consist of hyperparameters which require fine-tuning in order to achieve

optimal performance. In general, the hyperparameters optimization methods include grid search, randomized search, Bayesian optimization and heuristic search (Kumar, 2019). In this research, grid search is chosen to optimize over a set of selected hyperparameters combined with K-fold cross validation. A 5-fold cross validation is used in this research which is illustrated in Figure 5. The original 70% training dataset is split into five sub-training datasets with equal size. Four of these sub-training data sets are then used to train the model and the remaining sub-training dataset is used as the validation set. This process is repeated five times until each sub-training is used for training the model and as well as validating the model. Lastly, the average accuracy from the aforementioned procedure is used to determine the optimal hyperparameter values.

The important hyperparameters that are optimized by grid search are shown in Table 3 along with their definitions, search range and optimal values.

Figure 5: Flow chart of 5-fold Cross Validation (Liang et al., 2020)

Table 3: Hyperparameter Optimization Results

Model	Hyperparameters	Definition	Search Range	Optimal Values
XGBoost	n_estimators	Number of trees	[100,500,1000]	500
	learning_rate	Shrinking coefficient of each tree	[0.05,0.01,0.1]	0.1
	gamma	Minimum loss reduction	[0,0.25,0.5,1.0]	1
	reg_lambda	L2 regularization term on weights	[0,1,10]	1
	scale_pos_weight	Control the balance of positive and negative weights	[1,3,5]	1
Random Forest	n_estimators	Number of trees	[100,500,1000]	1000
	max_depth	Maximum depth of a tree	[25,75,100]	75
	min_samples_split	Minimum number of samples needed to split a node	[2,5,10]	5
	min_samples_leaf	Minimum number of samples required to be at a leaf node	[1,2,4]	2
Support Vector Machine	gamma	Kernel coefficient	[0.01,0.1,1.0]	0.01
	C	Regularization parameter	[0.1,1,10]	1

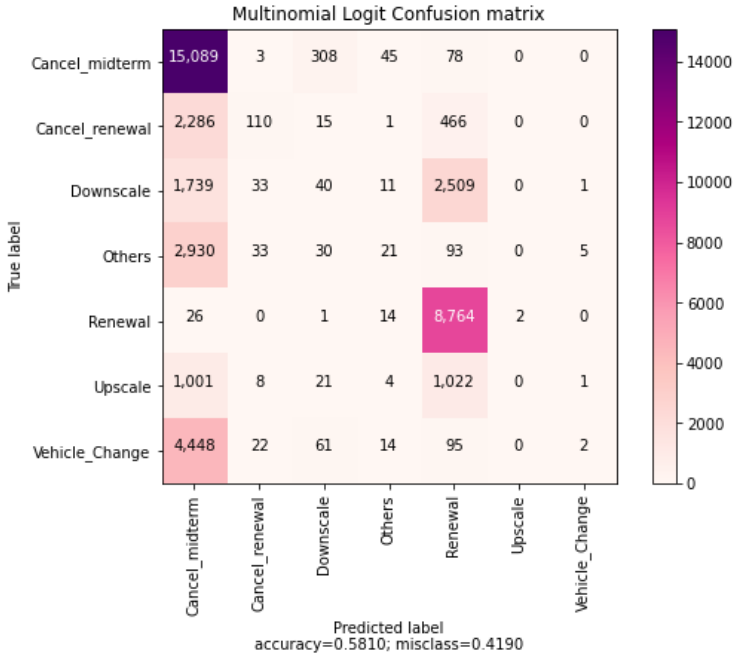
This table shows the hyperparameters that are tuned for XGBoost, Random Forest and Support Vector Machine.

5.2 Multinomial Logit

Two different Multinomial Logit (MNL) models are built. The first model uses the same data and all independent variables as the other machine learning models. The second model uses only the top 10 independent variables based on the variable importance plot of XGBoost (Figure 12). The variable importance plot of XGBoost is chosen because XGBoost is the best performing model based on the accuracy and F1-score.

By looking at Figure 6, it can be seen that the MNL model which utilizes all the independent variables does not perform well. It achieves an accuracy of merely 58.1% and the model only classify relatively accurate for *Cancel_midterm* and *Renewal*, and severely misclassify the other classes. This is further supported by Table 4 which shows that the other classes beside *Cancel_midterm* and *Renewal* have very low Precision, Recall and F1-score. Similarly, Figure 7 also shows that the MNL model is not a good classifier where most of the individual one-versus-all (OVA) AUC are relatively low. This is likely due to MNL is not robust in large feature space and thus suffers from the curse of dimensionality as mentioned in Prinzie and Van den Poel (2008). To improve this, it is necessary to adopt feature selection. Thus, I build a second MNL model which make use of the top 10 features from the Feature Importance plot of XGBoost (Figure 12).

Figure 6: Confusion matrix predicted by the first Multinomial Logit model



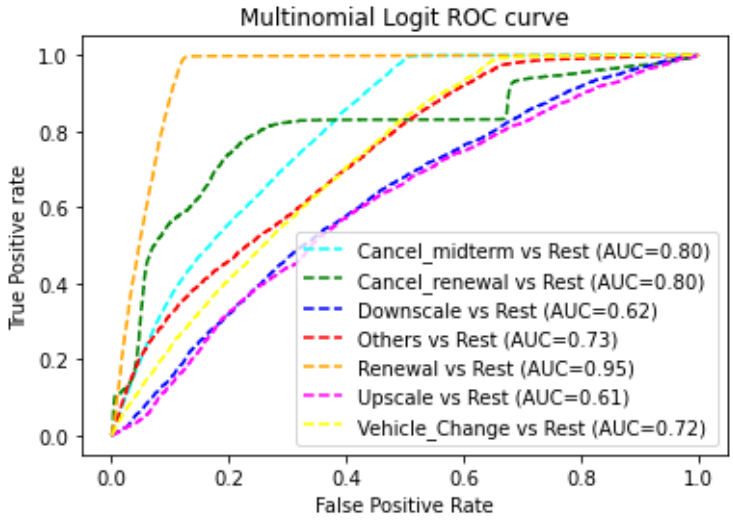
This figure shows the confusion matrix for the first Multinomial Logit model with accuracy of 58.1%.

Table 4: Classification report for the first Multinomial Logit model

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	54.83%	97.20%	70.11%	15523
Cancel_renewal	52.63%	3.82%	7.13%	2878
Downscale	8.40%	0.09%	1.66%	4333
Others	19.09%	0.07%	1.30%	3112
Renewal	67.28%	99.51%	80.28%	8807
Upscale	0.00%	0.00%	0.00%	2057
Vehicle_change	22.22%	0.04%	0.09%	4642

This table shows the Precision, Recall, F1-Score and Support for the first Multinomial Logit model.

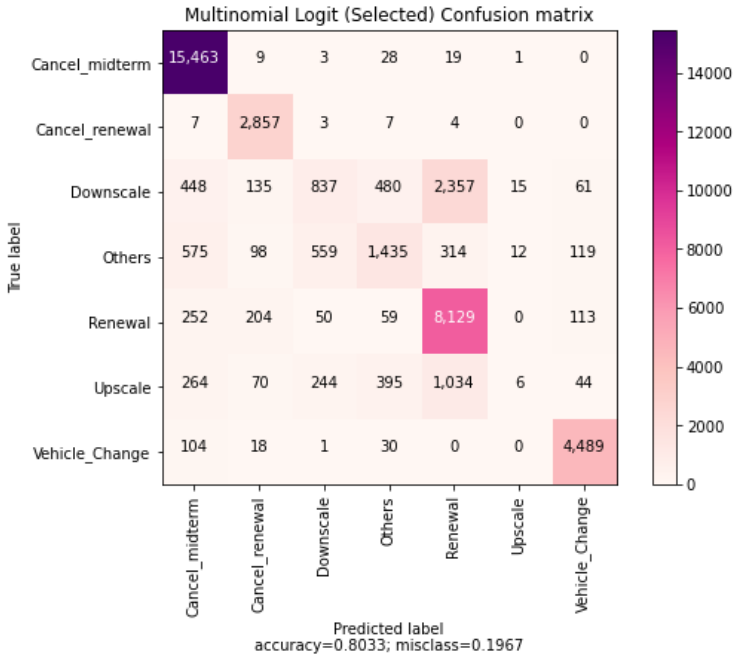
Figure 7: ROC Curves for the first Multinomial Logit model



The one-versus-all ROC curves for all classes from the first Multinomial Logit model

The second MNL model built with selected independent variables improved greatly compared to the first MNL model. By using selected independent variables, I managed to improve the accuracy from 58.1% to 80.3% which is shown in Figure 8. However, the model still do not classify well for some classes, particularly *Cancel_renewal* and *Upscale*. From Table 5, we can see that the Precision, Recall and F1-score improved massively compared to the first MNL model. Most notably with *cancel_renewal* and *vehicle_change*. The ROC curves displayed by Figure 9 shows that the second MNL model is a better classifier compared to the first MNL model with higher individual OVA AUC.

Figure 8: Confusion matrix predicted by the second Multinomial Logit model with selected independent variables



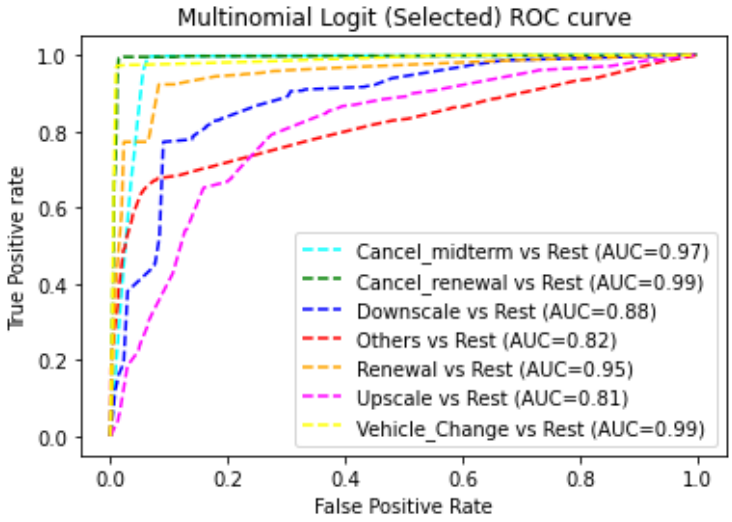
This figure shows the confusion matrix for the second Multinomial Logit model with accuracy of 80.3%

Table 5: Classification report for the second Multinomial Logit with selected independent variables

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	90.36%	99.61%	94.76%	15523
Cancel_renewal	84.25%	99.27%	91.15%	2878
Downscale	49.32%	19.32%	27.76%	4333
Others	58.96%	46.11%	51.75%	3112
Renewal	68.56%	92.30%	78.68%	8807
Upscale	17.65%	0.29%	0.57%	2057
Vehicle_change	93.02%	96.70%	94.82%	4642

This table shows the Precision, Recall, F1-Score and Support for the second Multinomial Logit model.

Figure 9: ROC Curves for the second Multinomial Logit model with selected independent variables



The one-versus-all ROC curves for all classes from the second Multinomial Logit model

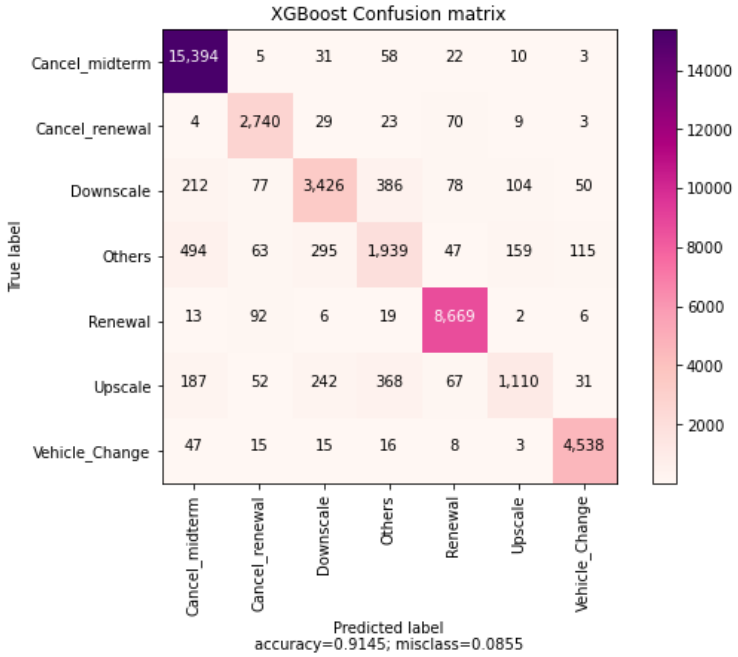
5.3 XGBoost

To prevent over-fitting as well as to speed up the computational time, I only take a subsample of 90% and half of the independent variables from the training data to train each tree. This is done by setting the hyperparameters $subsample = 0.9$ and $colsample\_bytree = 0.5$ respectively. By looking at Figure 10, we can see that XGBoost performs well by attaining an accuracy of 91.5% and classifies every class relatively accurate. This is further supported by Table 6 which shows that all classes are able to achieve Precision, Recall and F1-score of over 90% except for *Others* and *Upscale* which are slightly lower. This is likely due to the data is imbalance and these two classes have lower frequency. Further Figure 11 shows that XGBoost is a very good classifier where most of the individual OVA AUC are close to 1.

The independent variables are seldom equally relevant and often only a handful amongst them have significant influence on the dependent variable (Hastie et al., 2009, p.353). Hence, it is useful to learn which variables are "useful" or "valuable" in predicting the response. A feature importance plot can be constructed for this purpose since the importance can be calculated explicitly for each independent variable and thus allowing them to be ranked and compared with each other. The feature importance is calculated for every decision tree by the amount that each independent variable improves the performance measure - which in this research the Gini index, weighted by the number of observations in

that node (Hastie et al., 2009, p.355). The feature importance of every independent variable is then averaged over the ensemble trees. Figure 12 shows the feature importance plot of XGBoost where only the Top 10 independent variables are shown. These variables are used to build the aforementioned second Multinomial Logit model with selected variables.

Figure 10: Confusion matrix predicted by tuned XGBoost



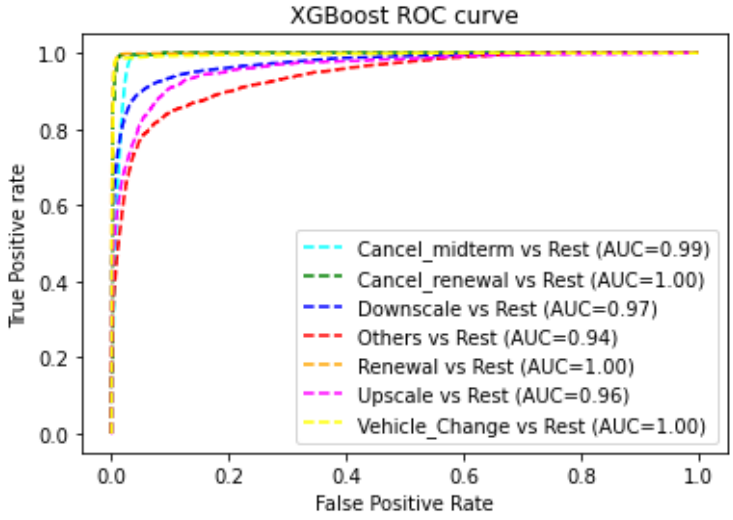
This figure shows the confusion matrix for the XGBoost model with accuracy of 91.5%

Table 6: Classification report for XGBoost

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	94.15%	99.17%	96.59%	15523
Cancel_renewal	90.01%	95.21%	92.54%	2878
Downscale	84.72%	79.07%	81.80%	4333
Others	69.03%	62.31%	65.50%	3112
Renewal	96.74%	98.43%	97.58%	8807
Upscale	79.46%	53.96%	64.27%	2057
Vehicle_change	95.62%	97.76%	96.68%	4642

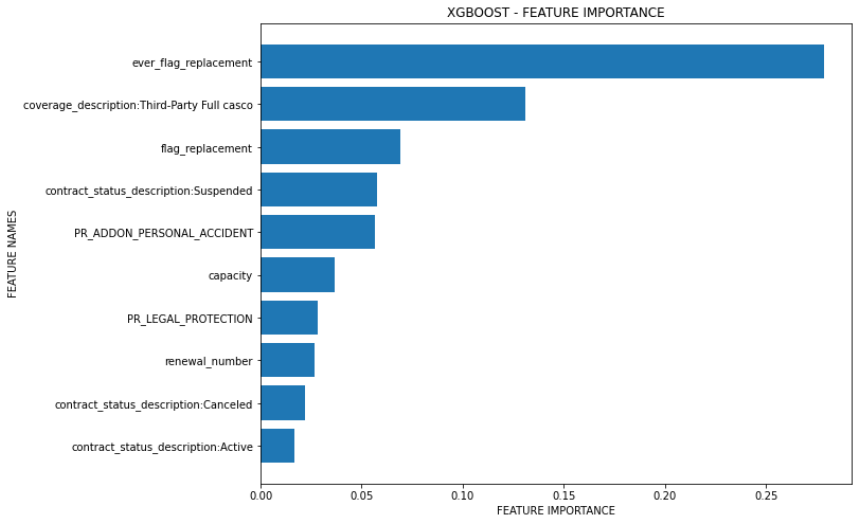
This table shows the Precision, Recall, F1-Score and Support for the XGBoost model.

Figure 11: XGBoost ROC Curve



The one-versus-all ROC curves for all classes from the XGBoost model

Figure 12: XGBoost Feature Importance



The Feature Importance plot for the XGBoost model

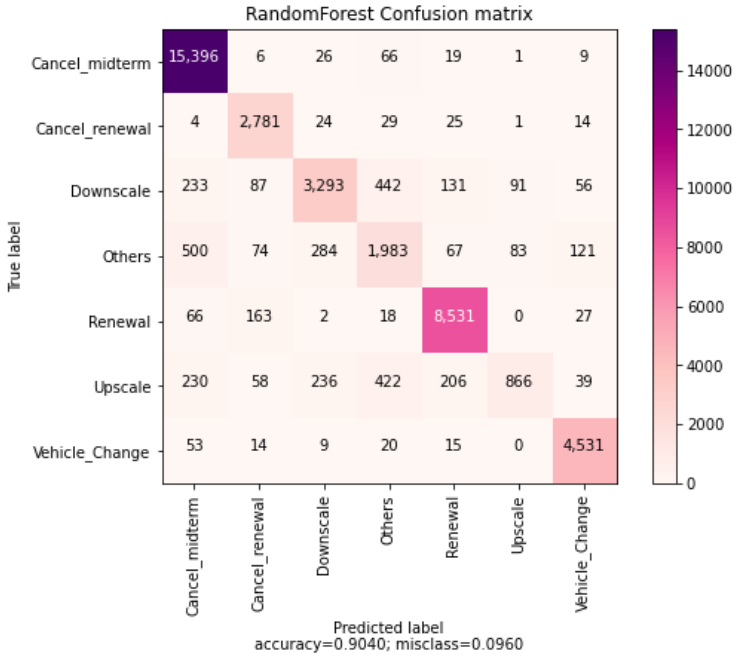
5.4 Random Forest

Random forest is proven to give accurate results even in its default settings (Fernández-Delgado, Cernadas, Barro, & Amorim, 2014). Further, Probst, Wright, and Boulesteix (2019) have shown that random forest is less "tunable" than many other machine learning algorithm such as support vector machines. Nonetheless, it was shown in the same paper that performance gain can still be achieved via hyperparameter tuning compared to the default settings. Although the average performance gain is moderate, it can be an important improvement in this research when wrongly classified instances can lead to inefficiency and higher customer maintenance cost. Ghosh, Fassnacht, Joshi, and Koch (2014), and Kulkarni and Sinha (2012) have shown that the most influential parameter in random forest is the number of features considered at each split ($max\_features$). Subsequently, Ghosh et al. (2014) have shown that the optimal number of features to be considered at each split is the square root of the total features. Even though out-of-bag error estimates of random forest could be used, I opt to use a test set to measure the error for the sake of consistency when it is compared across all machine learning models.

From Figure 13, it can be seen that Random Forest performs well by achieving an accuracy of 90.4% and classifies every class relatively accurate. This is further supported by Table 7 which shows that all classes are able to achieve Precision, Recall and F1-score of over 80% except for *Others* and *Upscale* which are slightly lower. This is likely due to the same problem encountered by XGBoost above where these two classes have lower frequency compared to the other classes. Figure 14 shows that Random Forest is a good classifier where most of the individual OVA AUC are close to 1.

Figure 15 shows the Top 10 independent variables which contributed the most in predicting the next action.

Figure 13: Confusion matrix predicted by Random Forest



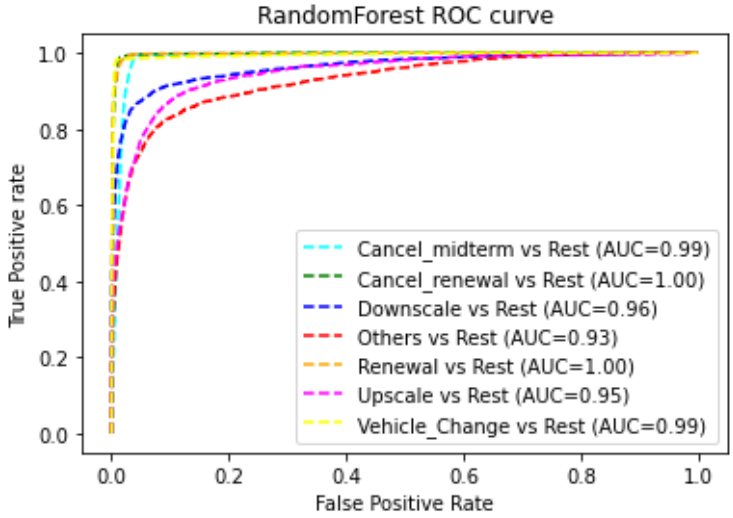
This figure shows the confusion matrix for the Random Forest model with accuracy of 90.4%

Table 7: Classification report for Random Forest

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	93.41%	99.18%	96.21%	15523
Cancel_renewal	87.37%	96.63%	91.77%	2878
Downscale	85.00%	76.00%	80.25%	4333
Others	66.54%	63.72%	65.10%	3112
Renewal	94.85%	96.87%	95.85%	8807
Upscale	83.11%	42.10%	55.89%	2057
Vehicle_change	94.55%	97.61%	96.01%	4642

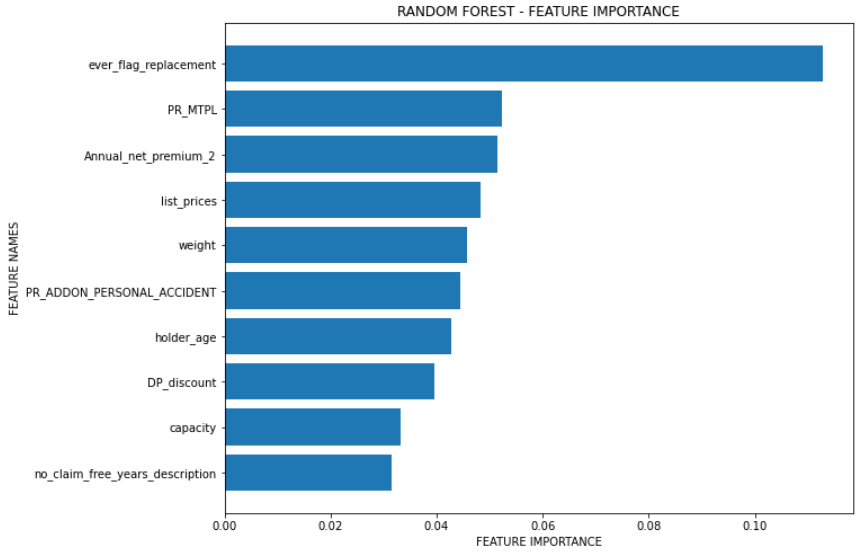
This table shows the Precision, Recall, F1-Score and Support for the Random Forest model

Figure 14: Random Forest ROC Curve



The one-versus-all ROC curves for all classes from the Random Forest model

Figure 15: Random Forest Feature Importance



The Feature Importance plot for the Random Forest model

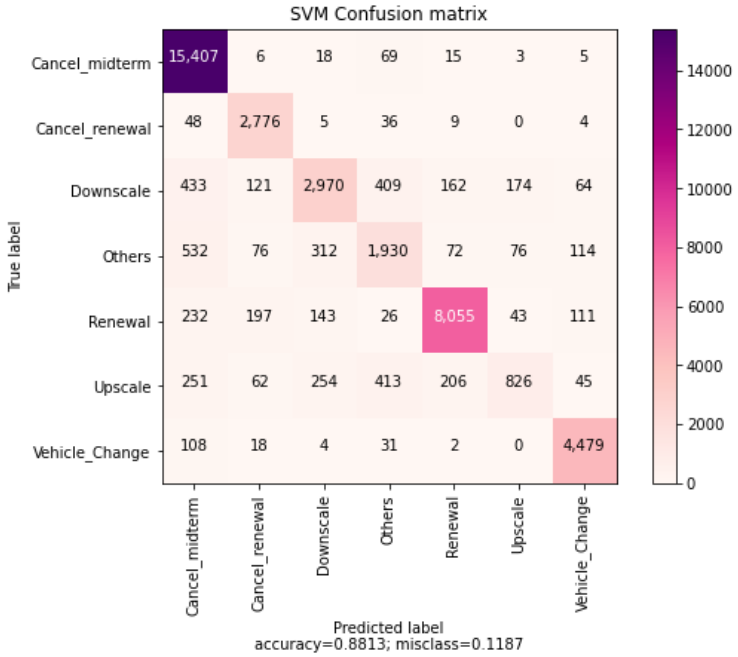
5.5 Support Vector Machine

As mentioned earlier in section 3.3, Support Vector Machine (SVM) tries to maximize the distance between the hyperplane and the support vectors. Thus, the independent variables are first needed to be normalized to prevent any particular one independent variable from dominating over the other.

From Figure 16, it can be seen that Support Vector Machine performs well by achieving an accuracy of 88.1% and classifies every class relatively accurate. This is further supported by Table 8 which shows that all classes are able to achieve Precision, Recall and F1-score of over 80% except for *Others* and *Upscale* which are slightly lower. This is likely due to the same problem encountered by XGBoost and Random Forest above where these two classes have lower frequency compared to the other classes. Figure 17 shows that Support Vector Machine is a good classifier where most of the individual OVA AUC are close to 1.

Unfortunately, unlike XGBoost and Random Forest, since a RBF kernel is used in this research, it is not possible to generate a feature importance plot for this Support Vector Machine model. Recall from section 3.3, a kernel function is used to project the data onto a higher dimensional space. Thus, the separating hyperplane belongs to another "space" which is not directly related to the input space.

Figure 16: Confusion matrix predicted by Support Vector Machine



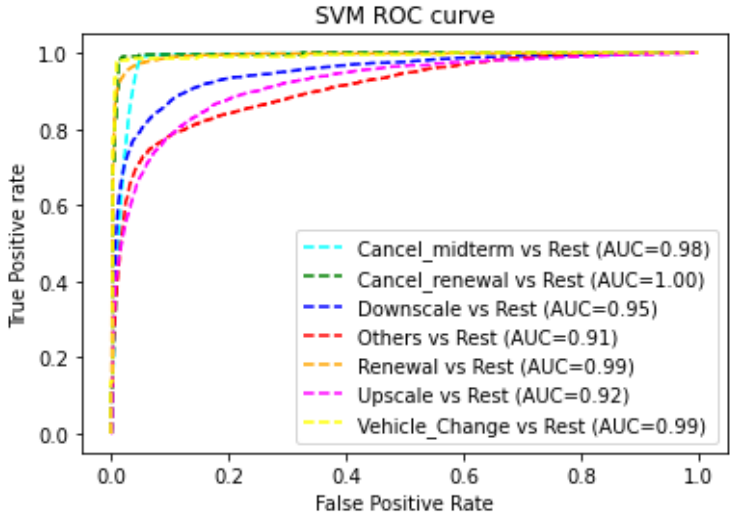
This figure shows the confusion matrix for the Support Vector Machine model with accuracy of 88.1%

Table 8: Classification report for Support Vector Machine

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	90.57%	99.25%	94.71%	15523
Cancel_renewal	85.26%	96.46%	90.51%	2878
Downscale	80.14%	68.54%	73.89%	4333
Others	66.23%	62.02%	64.06%	3112
Renewal	94.53%	91.46%	92.97%	8807
Upscale	73.62%	40.16%	51.97%	2057
Vehicle_change	92.89%	96.49%	94.65%	4642

Classification report for the Support Vector Machine model

Figure 17: Support Vector Machine ROC Curve



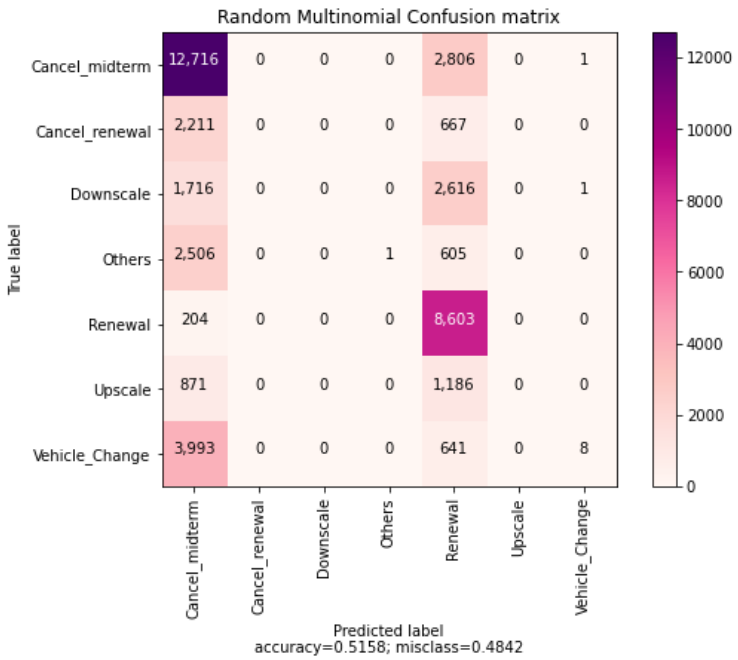
The one-versus-all ROC curves for all classes from the Support Vector Machine model

5.6 Random Multinomial Logit

A Random Multinomial Logit (RMNL) model which uses the properties of Random Forest is built for an attempt to improve the Multinomial Logit models discussed in section 5.2. Referring to algorithm 2, B was set to be 100. In other words, 100 Multinomial Logit models were built using a subset of randomized variables and the final predictions were obtained by majority voting. From Figure 18, it can be seen that RMNL does not perform well and the predictions converged to the two classes with the highest sample frequency. Further supported by Table 9, we can see that the model indeed do not have explanatory power for all the classes besides *Cancel_midterm* and *Renewal*.

A rule adjustment was made to take the class for which the difference between the prediction probability and the sample frequency is the largest. However, this did not solve the convergence issue and the results can be found in Appendix B.1.

Figure 18: Confusion matrix predicted by Random Multinomial Logit



This figure shows the confusion matrix for the Random Multinomial Logit model with accuracy of 51.6%

Table 9: Classification report for Random Multinomial logit

Metrics	Precision	Recall	F1-score	Support
Cancel_midterm	52.51%	81.92%	64.00%	15523
Cancel_renewal	0.00%	0.00%	0.00%	2878
Downscale	0.00%	0.00%	0.00%	4333
Others	100%	0.03%	0.06%	3112
Renewal	50.24%	97.68%	66.35%	8807
Upscale	0.00%	0.00%	0.00%	2057
Vehicle_change	80.00%	0.17%	0.34%	4642

Classification report for the Random Multinomial logit model

5.7 Overall Results

After discussing the models individually, I will discuss the overall results by comparing all the models. In Figure 19, all the models including the assessment measures are shown.

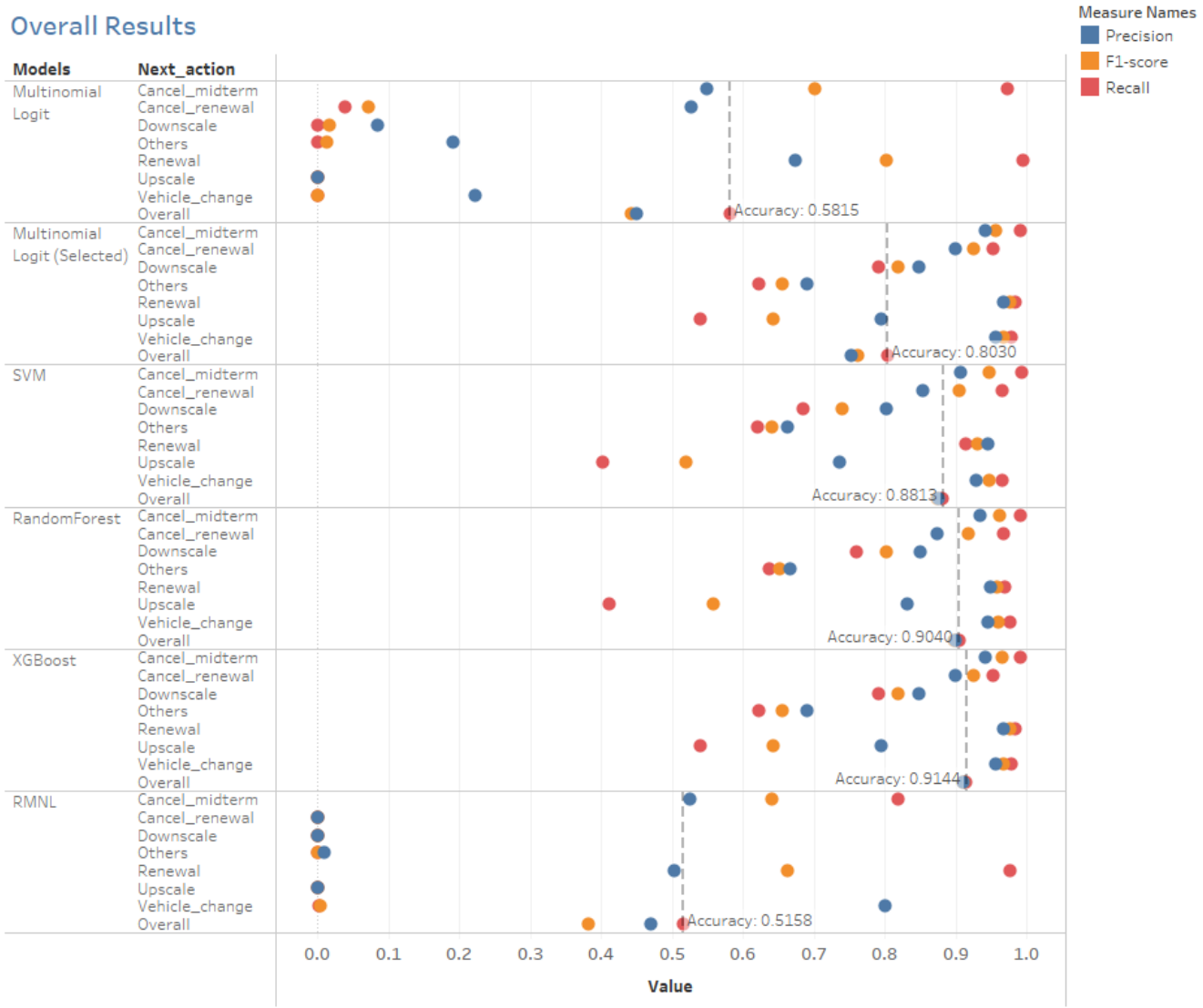
The Precision is indicated by the blue circles; the recall is indicated by the red circles; the F1-score is indicated by the orange circles and the overall accuracy is indicated by the dotted lines.

Figure 19 summarises our findings above that the Random Multinomial Logit (RMNL) is the worst performing model followed by Multinomial Logit model (MNL) with very low Precision, Recall and F1-score. RMNL was intended to improve the MNL model by utilizing some "randomness" but it was shown that RMNL is not suitable for this research and it is likely due to the class frequencies are imbalanced and the variables lacked explanatory power in predicting some of the classes. Furthermore, the only anomaly that can be observed here for MNL is the exceptionally high Recall for *Cancel_midterm* and *Renewal*, shown by the two far right red circles in the first pane. This is due to *Cancel_midterm* and *Renewal* together make up almost 60% of the data and MNL predicts the test data overwhelmingly as either *Cancel_midterm* or *Renewal* and thus, disguising many false positives. Hence, by combining the other assessment measures, RMNL and MNL can be concluded as poor performing models.

Another interesting observation is that by using only the Top-10 features from the Feature Importance plot of XGBoost to build a Multinomial Logit model, I am able to improve the model massively. The assessment measures improved for every class which can be seen in Figure 19 by comparing the colored circles in the first and second panes. We can see that the circles shifted from the left spectrum towards the right which indicate a good improvement. However, the two MNL and RMNL models still do not perform as well as the machine learning models. The machine learning models namely, Support Vector Machine (SVM), Random Forest (RF) and XGBoost have similar performance with XGBoost emerging as the best performing model based on the overall accuracy and F1-score. Notably, all the models predicted badly for *Others*, *Upscale* and also *Downscale* on a lesser extend. This is clearly demonstrated in the same figure where the colored circles for these three classes are more on the left compared to the rest. One possible reason is that these three classes have lower frequencies compared to the other classes.

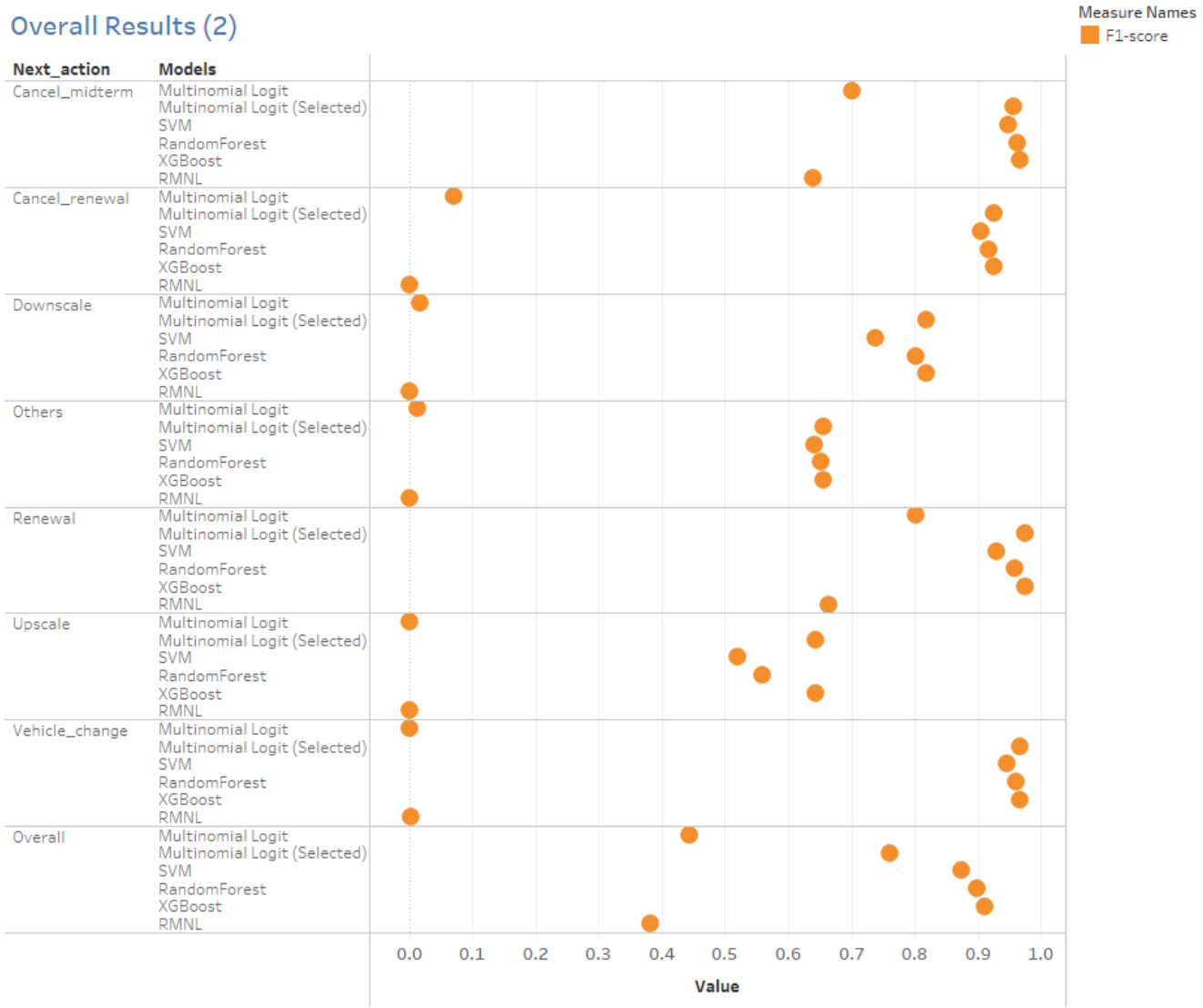
Similarly, the results can be investigated from a different perspective as shown in Figure 20 - the models are now nested within a class. By doing this I am able to tell which model predicts best for a particular class. Further, since F1-score is a weighted average of Precision and Recall, I simplify the figure by using only F1-score to evaluate the models. It is clear that XGBoost outperforms all other models for every class. Finally, looking at the overall results which is in the bottom pane of the same figure, XGBoost demonstrates that it is indeed the best performing model with the highest F1-score.

Figure 19: Overall Results from the models



This figure shows the Recall, Precision and F1-score predicted by every model for each of the class. Next actions are nested within every model. The blue circles denote Precision; red circle denote Recall; orange circles denote F1-score

Figure 20: Overall Results from the models



This figure shows the Recall, Precision and F1-score predicted by every model for each of the class. The models are nested within each next action. The orange circles denote F1-score

5.8 One-dimensional Factor Analysis

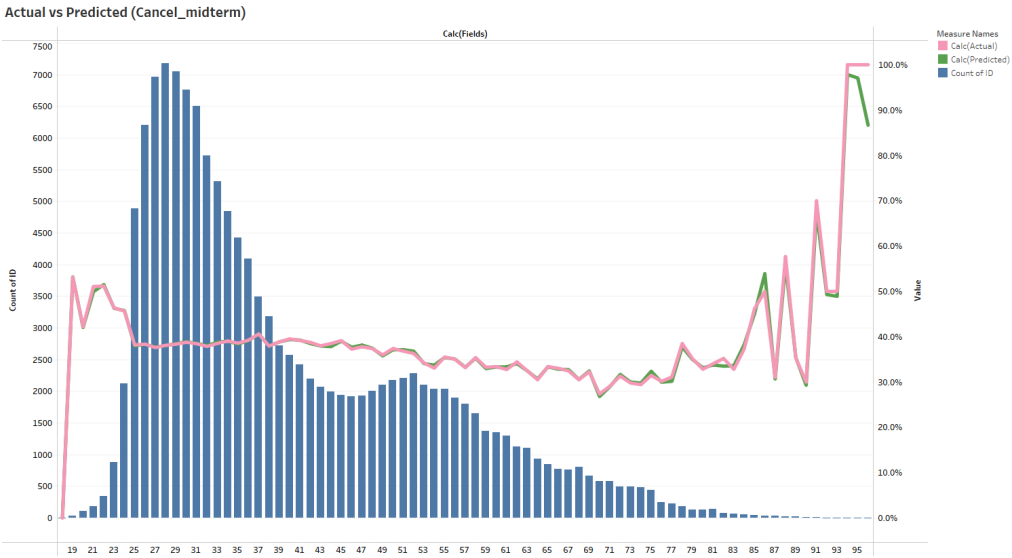
Since XGBoost is the best performing model, I will explore further with XGBoost by investigating from the perspective of one-dimensional factor. In other words, I will look into how the model performs against the actual (Actual vs Predicted) at the levels of the selected independent variable. The one-dimensional factor analysis is a common analysis within the non-life insurance industry where the insurer is keen to know how close (or far) is the prediction compared to the actual value relative to the exposure. By doing this analysis, the insurer is able to tell if the model requires further "tweaking" such as grouping certain homogeneous levels of a factor together.

Two independent variables are selected to be studied separately, one continuous variable - Age (Figures 21 and 22) and another discrete variable - Fuel Type (Figures 23 and 24). Only *Cancel_midterm* and *Upscale* are shown here and the other classes are shown in Appendix B.2 and B.3. In figures 21 and 22, the age distribution is denoted by the blue bars, the pink line denotes the predicted probability from XGBoost and green line denotes the actual probability. In figure 21, the two lines are almost identical which indicate that XGBoost predicted *Cancel_midterm* very well at every age band. Although there are more variations at each age band in figure 22, the green line (predicted) still closely follows the pink line (actual) except at the left spectrum (age 18) where the two lines deviate greatly. However, the exposure for this age band is very low which explains the big variation.

In figures 23 and 24, it can be seen that the exposures are concentrated at Petrol and Diesel. The predicted and actual are very close to each other at these two levels. There are more deviations at the other levels but they are paired with very low exposures. For instance, for fuel type *Others* in figure 23, the actual probability of *Cancel_midterm* is 75% where the predicted probability of *Cancel_midterm* is 65.7% but the exposure is very low with only 4 counts. It is worth noting that since this is a discrete variable, the two lines are only meant for comparison with respect to the exposure without interpreting the order of the variable on the x-axis.

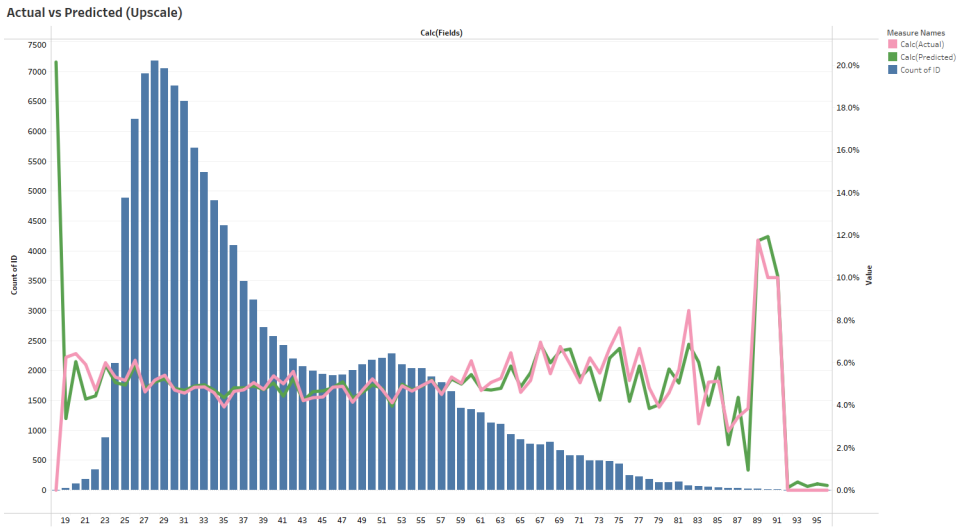
From this analysis, I am able to tell that the prediction from XGBoost is accurate even at the levels of a particular independent variable.

Figure 21: Cancel at Midterm by age



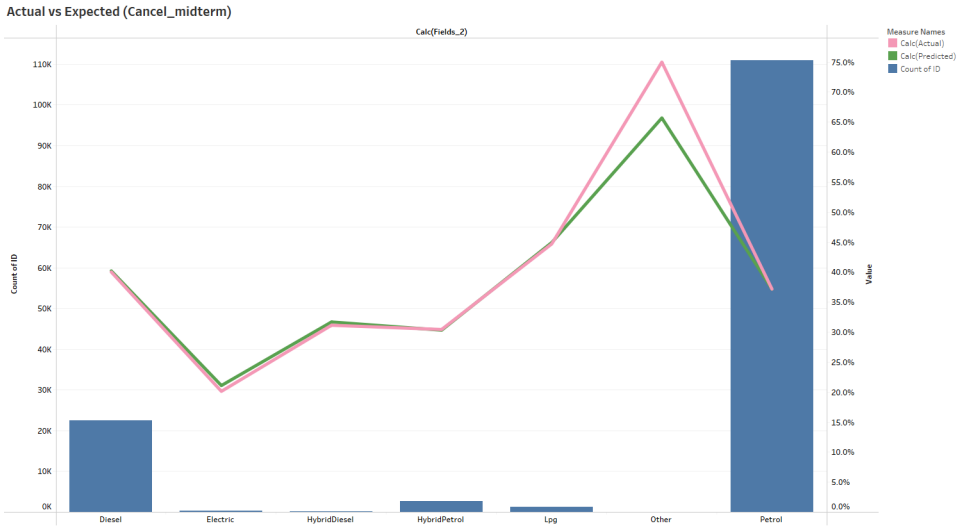
This figure shows the predicted *Cancel_midterm* probability against the actual value at every age band. The actual value is denoted by the pink line; the predicted value is denoted by the green line; the blue bars denote the exposure(count) at every age band.

Figure 22: Upscale by age



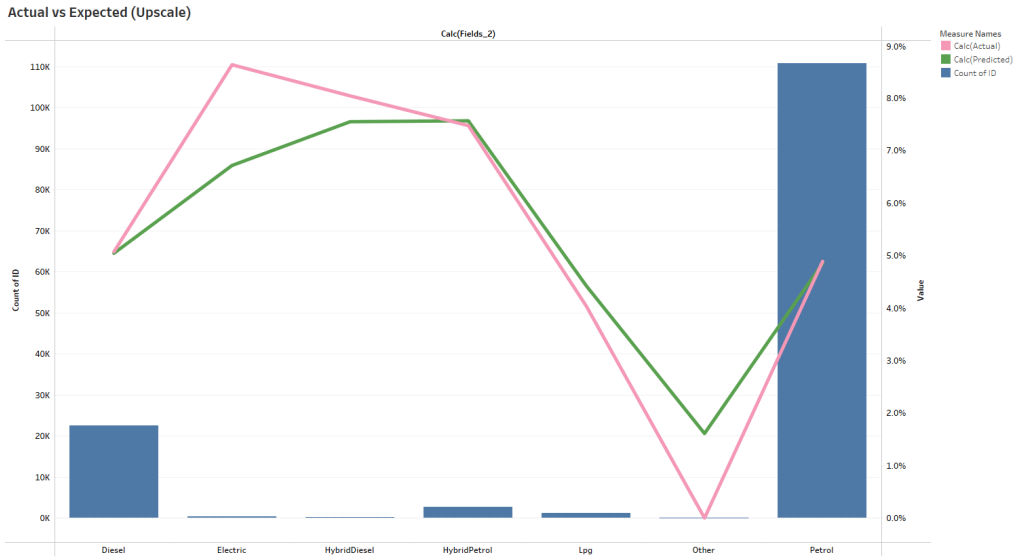
This figure shows the predicted *Upscale* probability against the actual value at every age band. The actual value is denoted by the pink line; the predicted value is denoted by the green line; the blue bars denote the exposure(count) at every age band.

Figure 23: Cancel at Midterm by fuel type



This figure shows the predicted *Cancel_midterm* probability against the actual value at each fuel type. The actual value is denoted by the pink line; the predicted value is denoted by the green line; the blue bars denote the exposure(count) at each fuel type.

Figure 24: Upscale by fuel type

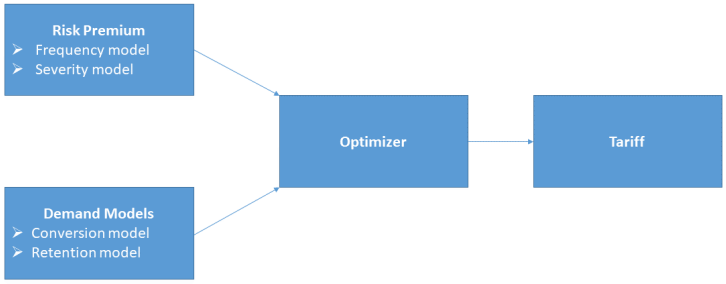


This figure shows the predicted *Upscale* probability against the actual value at each fuel type. The actual value is denoted by the pink line; the predicted value is denoted by the green line; the blue bars denote the exposure(count) at each fuel type.

6 Model Implementation

Non-life insurance companies use many models to compute their insurance tariff which is demonstrated in Figure 25. Besides using these models to compute insurance tariff, the models are used to generate monitoring reports such as model performance reports, sales reports and competitors monitoring reports. The Next Action model has three main purposes. First, it will be used to compile a list of policies which are predicted to downscale for further actions to be taken by the marketing team. Second, it is used along with the existing competition monitoring report to tackle policies which are predicted to cancel. Third, it will be used to generate monitoring reports and ideally be incorporated into the pricing process in the future.

Figure 25: Current Pricing Process Flow Chart



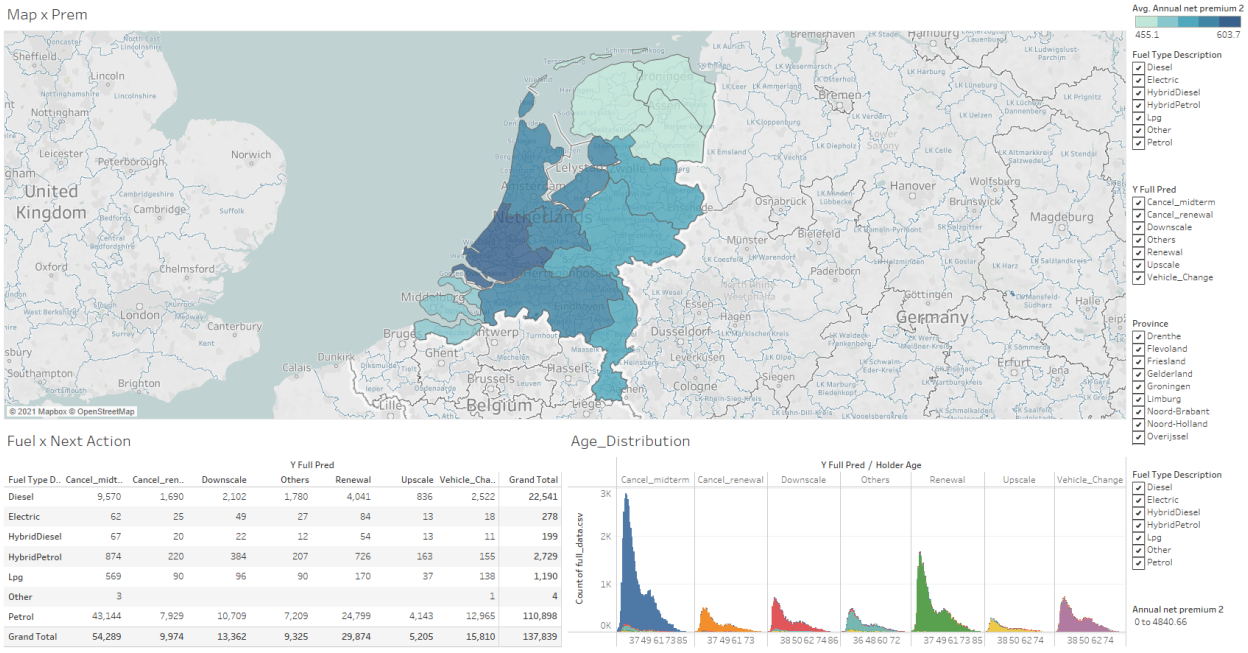
6.1 List for Marketing (Downscale)

Since XGBoost is the best performing model in this research, it is chosen to be implemented for practical use for the company. Besides the next action classification, I output the probability of all of the next actions for every policy so that it is possible to have a probabilistic view. The results are then imported to Tableau - a data visualisation software where I am able to investigate the results in an interactive way. As shown in Figure 26, on the top of the figure, a map is plotted and shaded based on the average premium. The higher the average premium in that province, the darker the color. Further, on the bottom left of the figure, I have the predicted next actions broken down by vehicle fuel type; on the bottom right of the figure, I have the predicted next actions broken by age. These two factors are selected because they have huge influence on the tariff and they are usually monitored by the pricing team. The buttons on the right allow me to determine the criteria.

For instance, the company is interested to know for the customers in Zuid-Holland with premium above 2500, how many will downscale their policies and what are their vehicle fuel type? By setting the criteria with the buttons on the right, I am able to filter out the customers as shown in Figure 27. A list can then be compiled and to be used by the marketing team. For example, the marketing team could send out promotions to the 5 customers who are predicted to downscale their policies to prevent it from happening.

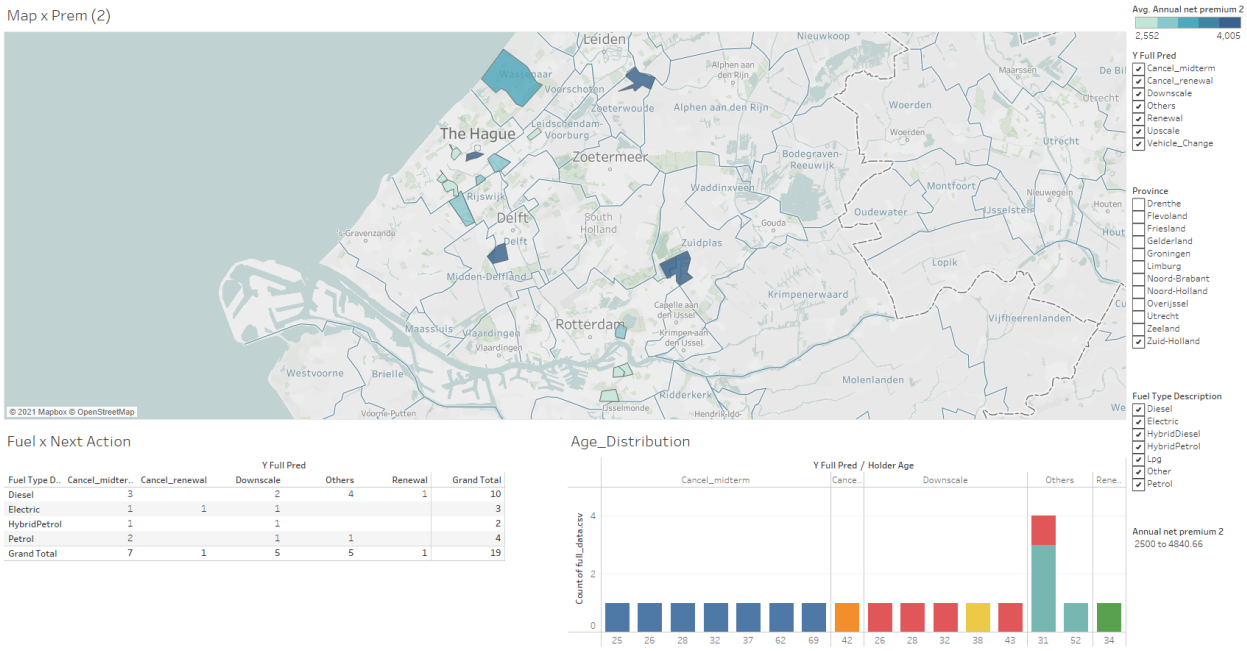
Similar subjective promotions can be given for the other next actions. By doing this, the company is able to better evaluate their customers and potentially reduce customer churning rate, improve retention rate and increase gross premium.

Figure 26: List for Marketing implemented in Tableau



The top of this figure shows a map of Netherlands shaded by level of the average premium. Bottom left of this figure shows the distribution of the next actions broken down by vehicle fuel type. Bottom right of this figure shows the distribution of the next actions broken by age. On the far right of this figure shows the interactive buttons which control the figure.

Figure 27: List for Marketing implemented in Tableau (Filtered by Zuid-Holland)



The top of this figure shows a map of Zuid-Holland which shows this province has a range of premium between 2552 and 4005 euros. Bottom left of this figure shows the distribution of the next actions broken down by vehicle fuel type. Bottom right of this figure shows the distribution of the next actions broken by age. On the far right of this figure shows the interactive buttons which control the figure, which is filtered by selecting only Zuid-Holland

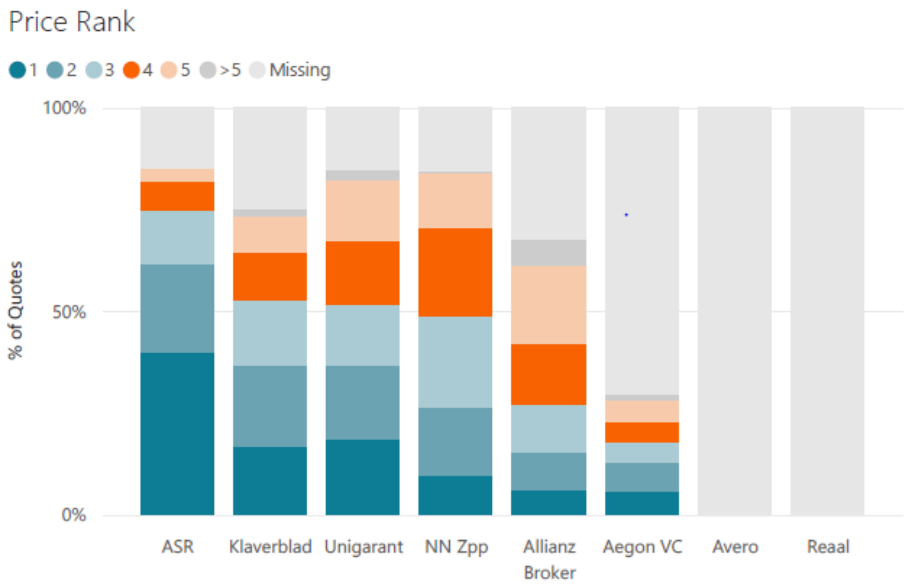
6.2 List for Marketing (Cancellation)

Besides looking at the customers who downscale their insurance policies, another major concern is customers who cancel their policies. One possible and logical reason for customers who cancel is they are able to find better product in the market provided by other insurers. To tackle this issue, the Next Action model can be used along with one of the existing competition monitoring report which keeps track of all premium available in the market and rank them in terms of price. The monitoring report filtered by only the Top-8 insurers is shown in Figure 28.

Using the same interface in Tableau demonstrated in section 6.1, I am able to filter out policies in Zuid-Holland which are predicted to be cancelled, either *Cancel_midterm* or *Cancel_renewal*. As shown in Figure 29, the map at the top of the figure shows the map of Zuid-Holland broken down by zip-code and shaded based on average premium. Hence, the company is able to investigate selected regions (by zip-code) with the highest average

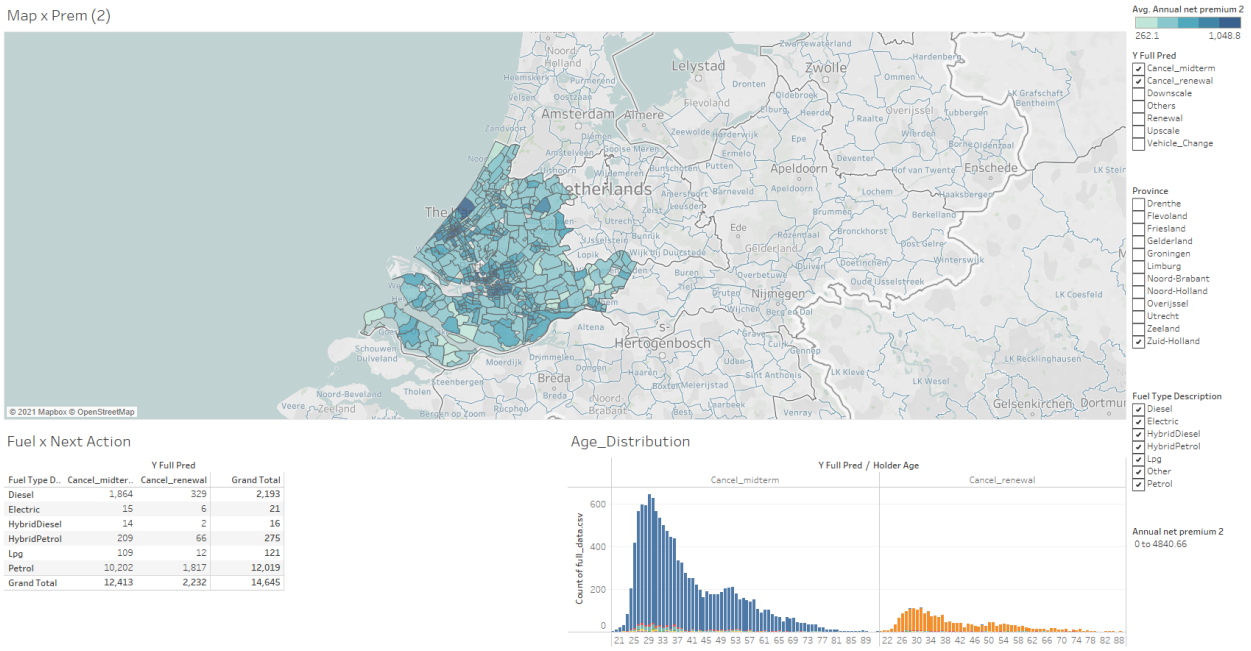
premium and highest cancellation rate. The company is able to use the aforementioned competition monitoring report (Figure 28) to help with the investigation. For instance, the company can look at the price rank of the policies which are predicted to be cancelled. If the to-be-cancelled-policies are consistently ranked lower, the company may consider this as an insight that the customers are switching to the competitors due to price. The company can then fine-tune their price or give the customers discount to prevent the customers from cancelling.

Figure 28: Price Rank Comparison of the Top-8 insurers



This figure shows the price rank of the Top-8 insurers (including Allianz Direct) on an aggregator website. It shows the quoted premius offered to the customers and how do the quotations rank among the other insurers. The ranking is represented by different shades of color.

Figure 29: List for Marketing implemented in Tableau (Filtered by Zuid-Holland and Cancellation)



The top of this figure shows a map of Zuid-Holland shaded by level of the average premium and filtered by policies predicted to be cancelled (*Cancel_midterm* & *Cancel_renewal*). Bottom left of this figure shows the distribution of the next actions broken down by vehicle fuel type. Bottom right of this figure shows the distribution of the next actions broken by age. On the far right of this figure shows the interactive buttons which control the figure.

6.3 Monitoring Report & Pricing Process

Apart from implementing the model in Tableau to have an interactive view, the Next Action model will also be incorporated into *Radar* - a software provided by Willis Towers Watson and to be used by the company. *Radar* will read in required input data every month to generate a next action report using the output of the model. The report consists of the active policies and their predicted next action broken down by variables of interest such as vehicle fuel type, age of policyholder, claim free years and zip-code. The company aims to gain more customer insights with this monitoring report and to relay the information into optimizing the pricing process.

To improve the pricing process, the Next Action model can be treated as an extension to the current Retention model where the latter predicts if a policy lapses or not and the former predicts other actions which include lapses. The Retention model is currently

being used to fine tune the insurance tariff by considering how likely a policy will lapse and subsequently translates that into a multiplicative factor which overlaid over the base insurance tariff. Similarly, the Next Action model can be used to derive a multiplicative factor which consider not only lapses but also other actions.

7 Conclusion

For an insurance company, it is crucial to make use of available data to fine tune their pricing models in order to have an edge over their competitors. For this reason, Allianz Direct is keen to know beforehand what their customers will do and make full use of this information. Hence, Allianz Direct wants to develop a Next Action model which is the focus of this research.

In this research paper, I investigate and implement six different models to predict the next action of the customers at Allianz Direct. The six models are Multinomial Logit without feature selection, Multinomial Logit with feature selection, XGBoost, Random Forest, Support Vector Machine and Random Multinomial Logit. On the other hand, the possible next actions are *Renewal*, *Upscale*, *Downscale*, *Change of Vehicle*, *Cancel at Midterm*, *Cancel at Renewal* and *Others*.

The aim of this research is to answer the questions stated in the introduction. By evaluating the models with the aforementioned assessment measures, it is evident that XGBoost is the answer to the first question where XGBoost outperforms the other models with the highest accuracy and F1-score. By comparing the machine learning (ML) models to the Multinomial Logit (MNL) model, we can see that ML models outperform the MNL without feature selection with significantly higher accuracy and F1-score and thus answering the second research question that it is justified to implement a ML model instead of a classical MNL model. Thirdly, even though by instilling randomness and employing feature selection, Random Multinomial Logit (RMNL) model does not perform better than the two MNL models in terms of accuracy and F1-score. On the other hand, the MNL model which uses feature selection by selecting the Top-10 features from the feature importance plot of the XGBoost model does perform better than the MNL model without feature selection in terms of accuracy and F1-score.

There are currently three ways to make use of the Next Action model which answer the fourth research question laid out in the introduction. The first and second implementation require the output from the XGBoost model to be imported into Tableau - a visualisation software. The results are then explored in an interactive manner where the first usage of the model is to compile a list of customers who are predicted to downscale their policies and filtered by a list of criteria such as zipcode, premium and vehicle fuel type. The second

usage of the model is similar to the first usage but instead focusing on customers who are predicted to cancel their policies and it is used along with an existing competition monitoring report. The downscale list and the cancellation list from the first and second usage of the model respectively, will be sent to the marketing team for further actions such as giving these customer promotions to prevent them from downscaling or cancelling. Lastly, for the third usage of the model, it will be used to generate monthly monitoring reports and ultimately to be incorporated into the pricing process. By implementing the model, the company believes it can boost revenue by giving targeted promotions, improve retention rate by being able to identify customers who are most likely to cancel and also improve the pricing and monitoring process.

Future research could look into comparing the performance between the Next Action model and the Retention model which is currently used by the company. It is also possible to employ the same methodology used in the Next Action model to build a Quotation model so that the tariff pricing process is able to optimize the premium during live customer quotation.

References

Behera, B., Kumaravelan, G., & Kumar, P. (2019). Performance evaluation of deep learning algorithms in biomedical document classification. In *2019 11th international conference on advanced computing (icoac)* (pp. 220–224).

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Classification and regression trees–crc press. *Boca Raton, Florida*.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794).

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Dionne, G., & Vanasse, C. (1989). A generalization of automobile insurance rating models: the negative binomial distribution with a regression component. *ASTIN Bulletin: The Journal of the IAA*, *19*(2), 199–212.

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, *15*(1), 3133–3181.

Fix, E. (1985). *Discriminatory analysis: nonparametric discrimination, consistency properties* (Vol. 1). USAF school of Aviation Medicine.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Ghosh, A., Fassnacht, F. E., Joshi, P. K., & Koch, B. (2014). A framework for mapping tree species combining hyperspectral and lidar data: Role of selected classifiers and sensor across three spatial scales. *International Journal of Applied Earth Observation and Geoinformation*, *26*, 49–63.

Gretton, A. (2013). Introduction to rkhs, and some simple kernel algorithms. *Adv. Top. Mach. Learn. Lecture Conducted from University College London*, *16*, 5–3.

Guernine, T., & Zeroual, K. (2011). New fuzzy multi-class method to train svm classifier. *DBKDA 2011*, 77.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd ed.). Springer series in statistics New York.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al. (2003). *A practical guide to support vector classification.* Taipei.

Hsu, C.-W., & Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, *13*(2), 415–425.

Krebel, U.-G. (1999). Pairwise classification and support vector machines. *Advances in kernel methods: support vector learning*, 255–268.

Kulkarni, V. Y., & Sinha, P. K. (2012). Pruning of random forest classifiers: A survey and

future directions. In *2012 international conference on data science & engineering (icdse)* (pp. 64–68).

Kumar, R. (2019). *Machine learning quick reference: Quick and essential machine learning hacks for training smart data models*. Packt Publishing Ltd.

Liang, W., Luo, S., Zhao, G., & Wu, H. (2020). Predicting hard rock pillar stability using gbdt, xgboost, and lightgbm algorithms. *Mathematics*, *8*(5), 765.

Mayoraz, E., & Alpaydin, E. (1999). Support vector machines for multi-class classification. In *International work-conference on artificial neural networks* (pp. 833–842).

Milgram, J., Cheriet, M., & Sabourin, R. (2006). "one against one" or "one against all": Which one is better for handwriting recognition with svms? In *tenth international workshop on frontiers in handwriting recognition.*

Pinquet, J. (1997). Allowance for cost of claims in bonus-malus systems. *ASTIN Bulletin: The Journal of the IAA*, *27*(1), 33–57.

Prinzie, A., & Van den Poel, D. (2008). Random forests for multiclass classification: Random multinomial logit. *Expert systems with Applications*, *34*(3), 1721–1732.

Probst, P., Wright, M. N., & Boulesteix, A.-L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(3), e1301.

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*(1), 81–106.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, *45*(4), 427–437.

Tay, F. E., & Cao, L. (2001). Application of support vector machines in financial time series forecasting. *omega*, *29*(4), 309–317.

Vapnik, V. (1998). The support vector method of function estimation. In *Nonlinear modeling* (pp. 55–85). Springer.

Yekkehkhany, B., Safari, A., Homayouni, S., & Hasanlou, M. (2014). A comparison study of different kernel functions for svm-based classification of multi-temporal polarimetry sar data. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *40*(2), 281.

A Data

Table 10: Data Dictionary

Variable	**Type**	**Description**
channel_description_abs	categorical	Sales channel
channel_type_sales_abs	categorical	How the contract is established
renewal_number	numeric	Number of previous contracts

(To be continued)

Variable	Type	Description
annual_kilometers_descr_abs	numeric	Kilometers the holder drives yearly
capacity	numeric	Vehicle Fuel capacity
weight	numeric	Vehicle Weight
threshold_description	categorical	Claim threshold description
pr_addon_free_repairshop	numeric	Add-on for repairshop
pr_addon_mobility	numeric	Add-on for mobility
pr_addon_personal_accident	numeric	Add-on for personal accident
pr_new_price_grntee	numeric	Add-on for price guarantee at quote
pr_purch_price_grntee	numeric	Add-on for for price guarantee at purchase
pr_legal_protection	numeric	Add-on for legal protection
pr_fmod	numeric	Add-on for full motor damage
pr_pmod	numeric	Add-on for partial motor damge
pr_mtpl	numeric	Add-on for third-party liability
Annual_net_premium_2	numeric	Annual premium
cd_reason_for_amndmnt	numeric	Reason for amendment
flag_replacement	numeric	Indicator if the vehicle is changed
Prem_diff	numeric	Premium difference with previous month
flag_cov_0	numeric	Indicator for coverage
flag_tpl_0	numeric	Indicator for third-party liability
contract_workflow_descrip	categorical	Contract flow description
contract_status_descrip	categorical	Contract status description
contract_installment_period	categorical	Contract installment type
no_claim_free_years_descrip	numeric	Number of years without a claim
fuel_type_description	categorical	Type of fuel of the vehicle
segment_description	categorical	Type of segment of the vehicle
drive_description	categorical	Type of drive of the vehicle
gear_description	categorical	Vehicle gear type
turbo_description	categorical	Indicator if the vehicle has a turbo engine
driver_gender	numeric	Gender of the driver
coverage_description	categorical	Type of main cover
list_prices	numeric	Price of the vehicle
DP_discount	numeric	Premium discount
RN_Factor	numeric	Renewal factor
holder_age	numeric	Age of the policy holder
ever_flag_replacement	numeric	Indicator for vehicle change
PROVINCIE	categorical	Province in which the policy holder lives
HH_INKOM3	numeric	Household income

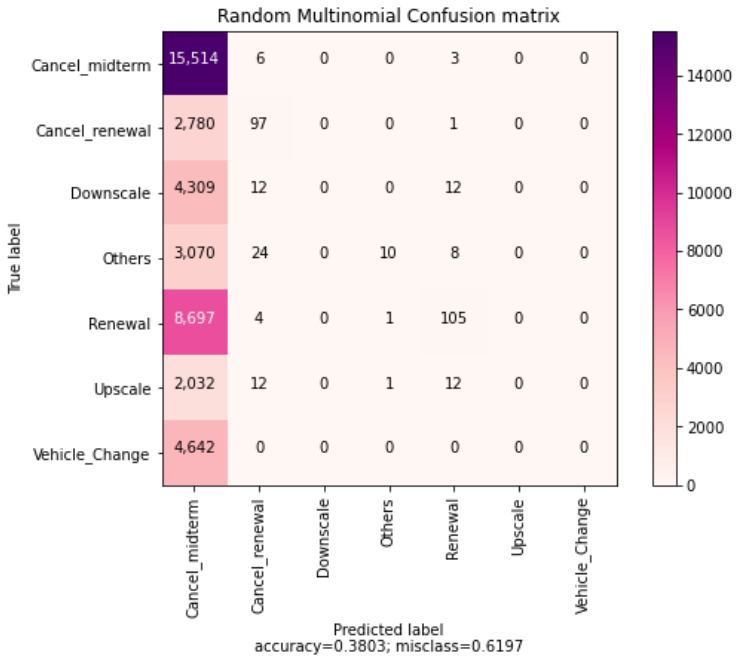
(To be continued)

Variable	Type	Description
HH_CREDCRD	numeric	Household credit card numbers
HA_KREDIET	numeric	Household credit score
HH_PA_AUTO	numeric	Household number of vehicles
price_rank	numeric	Rank of the premium compared to competitor insurers
independent_rank	numeric	Premium rank on the independer site

B Results

B.1 Random Multinomial Logit (Adjusted Rule)

Figure 30: Confusion matrix predicted by Random Multinomial Logit



This figure shows the confusion matrix for the Random Multinomial Logit (adjusted rule) model with accuracy of 38.03%

B.2 One Dimensional Factor (Age)

Figure 31: Cancel at renewal by age

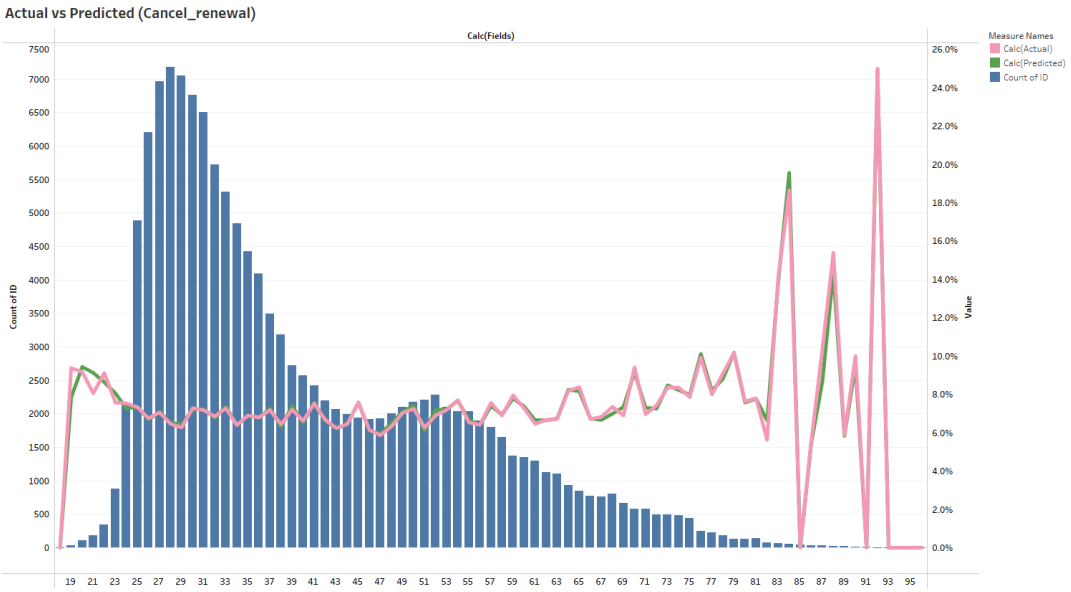


Figure 32: Downscale by age

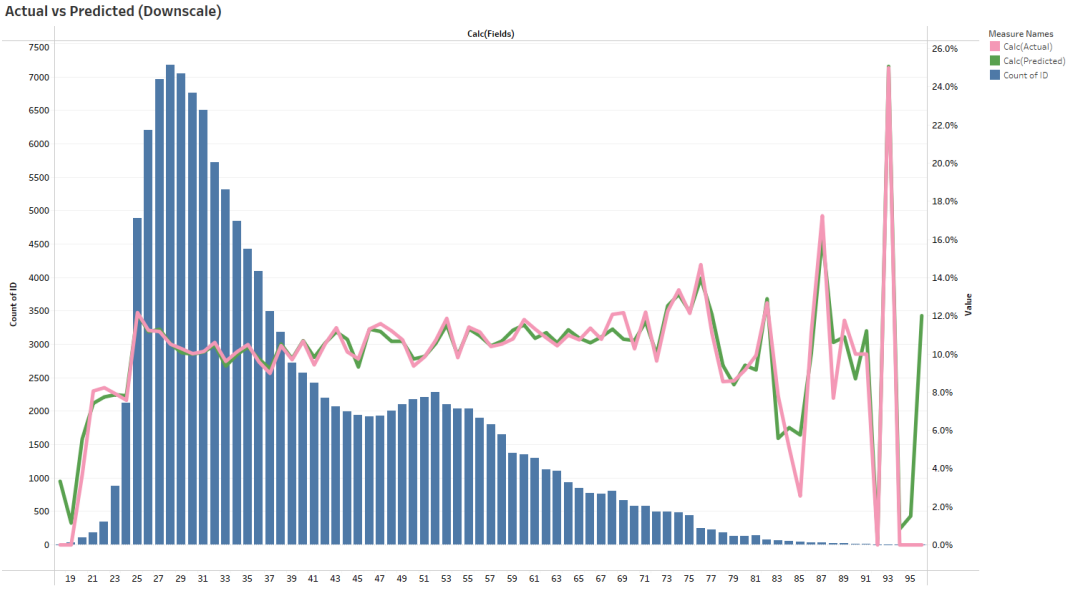


Figure 33: Others by age

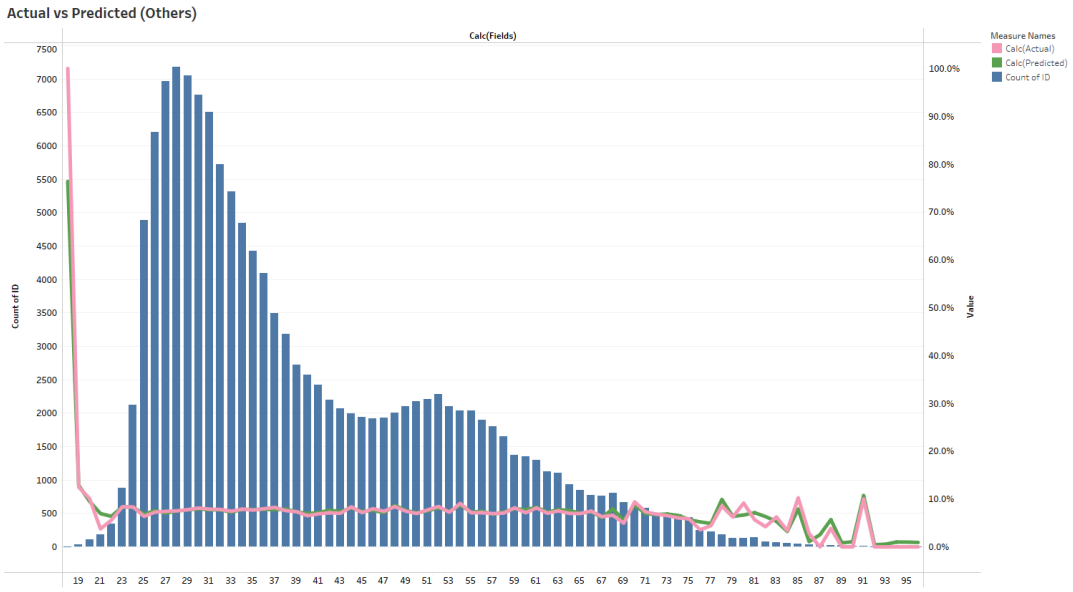
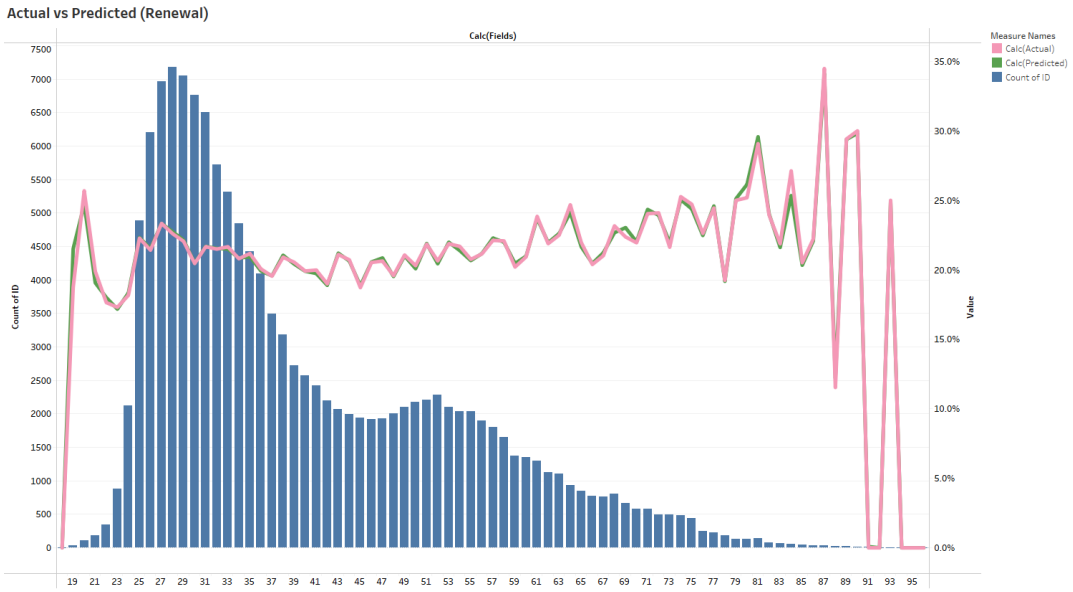


Figure 34: Renewal by age



B.3 One Dimensional Factor (Fuel Type)

Figure 35: Cancel at renewal by fuel type

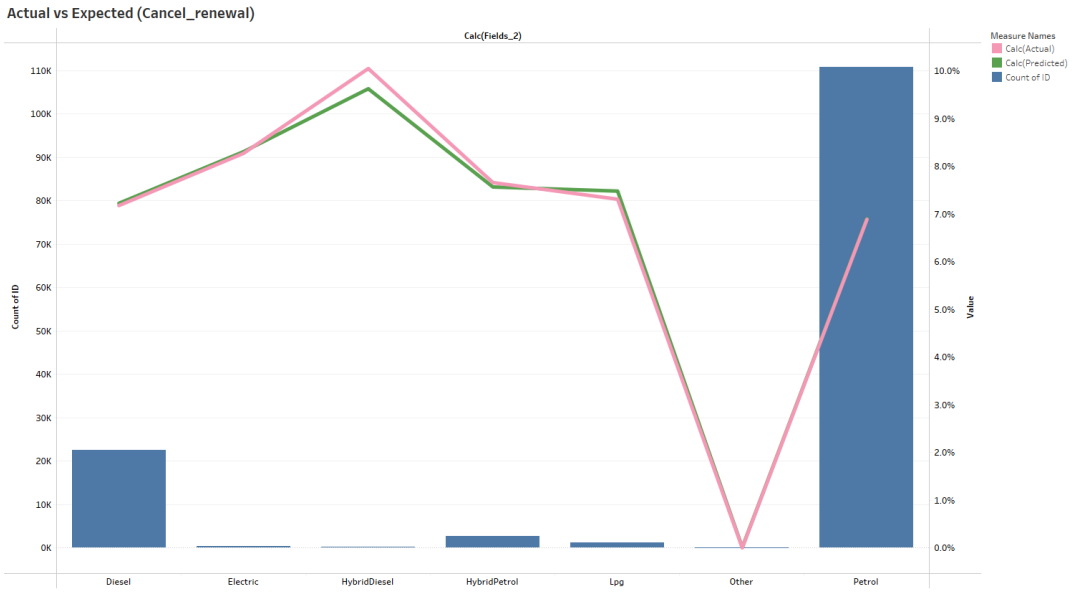


Figure 36: Downscale by fuel type

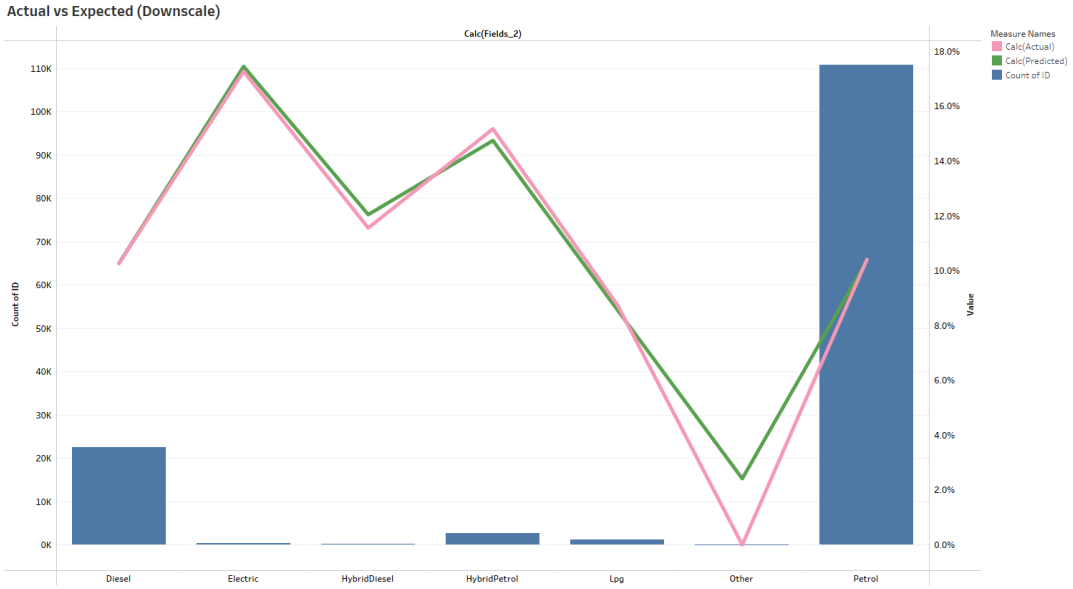


Figure 37: Others by fuel type

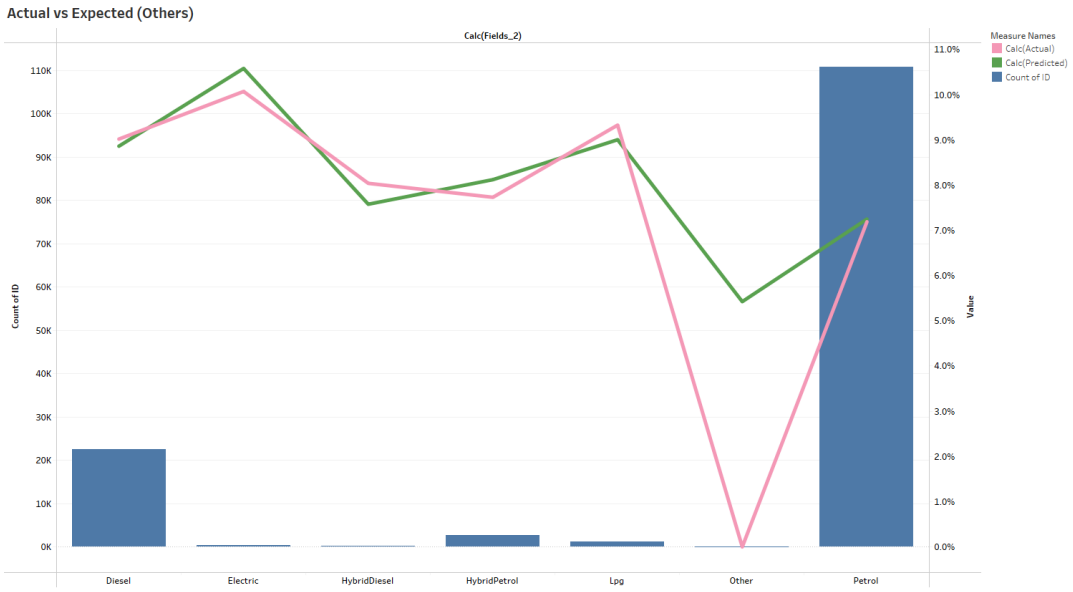


Figure 38: Renewal by fuel type