Hybrid Model for Claim Frequency and Claim Severity

Abhirupa Sen

A Thesis
in
The Department
of
Mathematics and Statistics

Presented in Partial Fulfillment of the Requirements
for the Degree of
Master of Science (Mathematics and Statistics) at
Concordia University
Montréal, Québec, Canada

September 2024

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Abhirupa Sen**
Entitled: **Hybrid Model for Claim Frequency and Claim Severity**

and submitted in partial fulfillment of the requirements for the degree of

Master of Science (Mathematics and Statistics)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Examiner
Dr. Arusharka Sen

_____ Examiner
Dr. Frédéric Godin

_____ Thesis Supervisor
Prof. Jose Garrido

Approved by _____
Dr. Marco Bertola, Chair of Department

_____ 2024

_____
Dr. Pascale Sicotte, Dean of Faculty

Abstract

Hybrid Model for Claim Frequency and Claim Severity

Abhirupa Sen

Rate making in insurance refers to the pricing of insurance premiums through calculations, by actuaries, and adjustments in various factors. Fair pricing of insurance products is of utmost importance for insurance companies to be able to face market competition and stay in business. Therefore poor rate making, which could be the result of poor prediction of risks, would be dangerous for insurers.

Insurance data is characterized by an imbalance between the number of policyholders that claim and those that do not. The majority of premium payers do not incur accidents and thus do not claim their losses, resulting in a large number of “zero–claims". However, it is very important for the company to identify the customers who are more likely in future to file a claim, because every claim incurs a cost to the enterprise. Chapter 1 proposes a sampling technique devised to improve the identification of the possible future losses by better tracking of the non–zero claims.

Generalized Linear Models have long been used by actuaries to accomplish the rate making task. The method is parametric and is based on certain assumptions about the distribution of the data. Insurance data with its probability mass at zero, do not fall exactly into the framework of GLMs. However, in the recent years, various new machine learning algorithms have provided improvement by being more effective predictors than GLMs. What these algorithms lack is interpretability. Chapter 2 uses simple algorithms like regression trees, in combination with GLMs, to create a pre–processed GLM that is more effective than a standalone GLM.

The endeavour of improving the classical GLM continues in Chapter 3. Here the another combination of trees and regularized GLMNet is used to produce results with more predictive capability that any one of these algorithms as stand-alone. The new results are interpretable as well as improved.

Contents

List of Figures

List of Tables

Chapter 1

Improving Minority Class Identification in Imbalanced Datasets

1.1 Abstract

Imbalance between classes is a characteristic feature of insurance classification data. Accidents are not typical, and nor are the claims. As a result, the number of claims filed in a year is primarily concentrated at zero. Due to the presence of deductibles in the policy clauses, many minor accidents go unreported, and companies remain unaware of insureds with the potential for sizeable future claims. Failing to identify risky policyholders is equivalent to losing revenue and incurring future loss. This chapter suggests and shows how re-balancing an imbalanced data set multiple times, and aggregating the predictions made by models built on each balanced data set, drastically improves the prediction probabilities of a potential claim.

1.2 Introduction

Classification for imbalanced classes is challenging, primarily because classification algorithms are built on the assumption that data is balanced between the classes. Severely skewed datasets lead to poor performance of the models. When a dataset becomes complex, the already present imbalance compounds the difficulty performing an efficient classification of the data. Insurance claim counts are discrete quantitative values that can be regressed, but the number of claims submitted by policyholders is seldom more than 4. In such cases, the problem can be seen as a 5-class classification problem with a high concentration in one class; for instance 90% of the claim counts being 0s. Claims' existence and non-existence can itself be considered a binary classification problem. Often known as claim propensity, it is a binary response which is set to 1 when claim count is one or more. Effective modelling of claim propensity can be used in lapse/renewal studies.

Claim counts or claim propensities do not obey normal distribution. The response

variables here are neither continuous nor symmetric but rather binary or discrete, and positively skewed. In insurance applications the variance of these types of responses are seen to be dependent on the mean value. These typical characteristics of insurance data are often modeled using a family of distributions known as the exponential dispersion (ED) family. Claim counts are modeled by non-negative integer valued random variables, also known as counting random variables. Whether a given policy is prone to have future claims is called the claim propensity. The response variable Y is either 1 or 0 depending on whether there will be at least one claim filed in the future or not. Claim propensity can be modelled using a Bernoulli distribution. The expected number of claims in a portfolio, in that case, follows a binomial distribution (sum of Bernoulli random variables). Both these two distributions belong to the ED family.

Modelling imbalanced data using classical methods is bound to perform poorly unless some modification is used. Additionally, the standard metrics fail to give a fair picture of how good a model is. Classifying all samples as 0s in a dataset with 97% 0s fetches a great accuracy score even when it fails to identify a single 1. However, the unidentified 1s could be potential frauds or accident claims, which can cause significant losses to companies if allowed to go astray. The probability of non-zero claims is unknown and is estimated using the proportion of claims in the data. It can be shown that when the probability of a claim or no claim are equal then the response becomes bell shaped like a normal distribution.

When applied to imbalanced data, any classical model does an excellent job identifying the majority class (0 claim count for insurance data). However, classical models are never good in predicting the minority class (1 or more claim counts). The accuracy of predicting the minority class is hardly 50% on average, which means that the model is missing out on potential losses irrespective of whether the class balance is 7:3 or 9:1. So, the first modification must be to strike a balance in the data. Re-sampling techniques are used to make data balanced. Re-sampling could mean adding more samples of the minority class (over-sampling), removing samples from the majority class(under-sampling), or combining both. One such popular over-sampling technique is the Synthetic Minority Oversampling Technique (SMOTE), introduced by Chawla et al. (2002). Instead of random oversampling from the minority class, SMOTE creates synthetic samples using interpolation. By considering k-nearest neighbors in the feature space of the minority class, a sample is created along the line segments joining any or all the k nearest minority class neighbors. Even if the method is popular SMOTE has the potential of performing poorly on high dimensional data. Under performance of the k-NN method is discussed elaborately in Hastie et al. (2009). The data sets used in the insurance industry are frequently high dimensional. Creating artificial sample points from the minority class using the k nearest neighbour method comes with the danger of creating records not so similar to the minority class in reality. It is worth mentioning the works of Stocksieker on data augmentation on imbalanced regression, for instance Stocksieker et al. (2023a). Another of his works Stocksieker et al. (2023b) throws light on the importance of oversampling in dealing with data imbalance.

Ensemble methods are potent because they combine multiple base learners, for example classification trees, to create a single model. A single learner doing an excellent job on a training data set could fail on new data points because it minimizes the variance in the training data and fits into its noise. This is overfitting the data. However, when many weak learners, whose results have higher bias, are combined, it produces a model of higher efficiency not only on the training but also on new data points.

Ensemble methods are either parallel learners, iterative, or hybrid. One prevalent parallel ensemble method, bagging (bootstrap aggregation), improves the model's generalization ability by creating random subsamples from the original dataset with replacement and then combining the predictions generated by base learners trained on each bootstrap subsample. Boosting is the most common iterative ensemble method. The first applicable boosting algorithm, Adaboost, developed by Freund et al. (1996), gives higher weights to samples misclassified, thus forcing the future base learners to focus more on learning the samples the previous learners failed. In recent years, tree-based ensemble methods have achieved great success in predictive performance. See Guelman (2012), Guelman et al. (2012), Olbricht (2012), Guelman et al. (2015), Wüthrich (2018), Yang et al. (2018), Lopez et al. (2019), Hu et al. (2022), Henckaerts et al. (2021), and the references therein, for additional examples of actuarial applications.

In this chapter, a combination of resampling and ensemble techniques is explored. Unlike SMOTE, where new minority class samples are synthesized, all the minority class samples are used repeatedly in constructing subsamples. The popular argument against undersampling of the majority class is the loss of information. The subsampling method ensures that every majority data point is included in at least one of the subsamples. An ensemble of learners trained on balanced subsamples is created. Each balanced sample has an equal proportion of minority and majority classes. Since the minority sample size is small, they are recycled in every subsample where an equal number of majority class members get added. Hence, every majority data point is used for training learners. All the learners train on balanced data and have a say on the final prediction of a data point. A new data is classified by each learner, and the class with a majority vote wins.

The rest of the chapter is divided as follows. Section 1.3 reviews the binomial distribution as part of exponential dispersion family and establishes how the mean and variability of this distribution can be made to work in favour of the minority class. Section 1.4 reviews the class proportion and its effect using the Bayesian approach. This section also sees how the remedial step suggested by us, removes the bias. Section 1.5 dicussed bias and variance with respect to the CART algorithm and how bagging as a method of aggregation, creates an optimal classifier. Finally it shows mathematically and illustrates with a toy example how the suggested method of subsampling tweaks the posterior Bayesian probability in favour of the minority class.

1.3 Bernoulli and Binomial Distributions as Members of Exponential Dispersion Family

We start this section by writing down the definition of the exponential dispersion family.

Definition 1 *Consider a response variable Y defined in the domain S, a subset of the real line. The distribution of Y is said to belong to the Exponential Dispersion (ED) family if $p_Y(y)$, the probability mass function (when Y is discrete) or $f_Y(y)$, the probability density function (when Y is continuous) of Y is of the form*

$$p_Y(y) = \exp\left\{\left(\frac{y\theta - a(\theta)}{\frac{\phi}{\nu}}\right)\right\} c\left(y, \frac{\phi}{\nu}\right), y \in S, \tag{1}$$

where:
θ = real-valued location parameter, called the canonical parameter,
ϕ = positive scale parameter, called the dispersion parameter,
ν = known positive constant, called the weight,
$a(.)$ = monotonic convex function of θ,
$c(.)$ = positive normalizing function.

The normal distribution belongs to the ED family. If a random variable $Y \sim N(\mu, \sigma^2)$ then the p.d.f of Y can be written as

$$f_Y(y) = \begin{cases} Y\ s\frac{1}{\sigma\sqrt{2\pi}} exp\left(\frac{-(y-\mu)^2}{2\sigma^2}\right), & -\infty < y < \infty, \\ 0, & \text{otherwise,} \end{cases}$$

where $f_Y(y)$ can be expanded in the following steps

$$f_Y(y) = \frac{1}{\sigma\sqrt{2\pi}} exp\left(\frac{-y^2 + 2y\mu - \mu^2}{2\sigma^2}\right),$$

$$= exp\left(\frac{y\mu - \frac{\mu^2}{2}}{\sigma^2}\right) \frac{exp(-\frac{y^2}{2\sigma^2})}{\sigma\sqrt{2\pi}},$$

where:
$\theta = \mu$
$a(\theta) = \frac{\mu^2}{2}$
$\phi = \sigma^2, \nu = 1$
$c(y, \frac{\phi}{\nu}) = \frac{exp(-\frac{y^2}{2\sigma^2})}{\sigma\sqrt{2\pi}}$

1.3.1 Bernoulli Trial and Bernoulli Distribution

Many counting variables, especially those that result in only 0 or 1 can be said to be the outcome of Bernoulli trials. The indicator function $I(.)$ is defined such that for any event

A,

$$I(A) = \begin{cases} 1 & \text{if A is true} \\ 0 & \text{otherwise} \end{cases}$$

The claim propensity is defined using the identity function such that $Y = I(N \geq 1)$, where N is the number of claims filed by a policyholder. The variable Y is thus dichotomous or $Y \in [0,1]$. Such a variable is said to be Bernoulli distributed. The probability mass function of the Bernoulli distribution is given by

$$p_Y(y) = \begin{cases} q^y(1-q)^{1-y} & \text{if } y \in \{0,1\} \\ 0 & \text{otherwise} \end{cases}$$

The mean and variance of $Y \sim Ber(q)$ are given by

$$E[Y] = q$$

$$Var[Y] = q(1-q)$$

It can be shown that the Bernoulli distribution belongs to the ED family by rewriting its pmf:

$$p_Y(y) = q^y(1-q)^{1-y}$$

$$= exp\left(y \ln(\frac{q}{1-q}) + \ln(1-q)\right)$$

Thus

$$\theta = \ln\left(\frac{q}{1-q}\right) => q = \frac{\exp\{\theta\}}{1+\exp\{\theta\}}$$

$$a(\theta) = -\ln(1-q) = \ln(1+\exp\{\theta\})$$

$$\phi = 1, \nu = 1, c(y,\phi) = 1$$

1.3.2 Binomial Distribution

The Binomial Distribution corresponds to the total number of successes in a sequence of independent Bernoulli trials. So the number of policy holders who are likely to file claims in a portfolio of m policies can be thought of as a Binomial random variable. Denoting Y as the number of policyholders in the portfolio who could file claims in future, we can see that $Y \in \{0,1,2,...m\}$. The probability mass function of Y can be written as

$$p_Y(y) = \begin{cases} \binom{m}{y} q^y(1-q)^{1-y} & \text{if } y \in \{0,1,2,...,m\} \\ 0 & \text{otherwise} \end{cases}$$

It can be shown that the binomial distribution belongs to the ED family by rewriting its pmf:

$$p_Y(y) = \binom{m}{y} q^y(1-q)^{1-y}$$

$$= exp\left(y\ln(\frac{q}{1-q}) + m\ln(1-q)\right)\binom{m}{y}$$

Thus

$$\theta = \ln(\frac{q}{1-q}) => q = \frac{\exp\{\theta\}}{1+\exp\{\theta\}}$$

$$a(\theta) = -m\ln(1-q) = m\ln(1+\exp\{\theta\})$$

$$\phi = 1, v = 1, c(y,\phi) = \binom{m}{y}$$

The mean and variance of $Y \sim Bin(m,q)$ are given by

$$E[Y] = mq$$

$$Var[Y] = mq(1-q)$$

It is important to note that the variance of a binomially distributed random variable is a function of its mean and is lower than the mean. Also, when q is closer to 0 the expected values of Y moves towards 0 with low variance. However, when q is closer to 0.5, the expected value of the variable is almost half of the portfolio and the variance is high. In fact, the variance of the binomial distribution is highest when $q = 0.5$. The third central moment of, measuring the skewness, is given by $mq(1-q)(1-2q)$. Hence the binomial distribution is symmetric when $q = \frac{1}{2}$. Figure 1.1 illustrates this property. It can be seen that the last curve (in red broken lines), when $q = 0.5$, is symmetric and has the highest spread due to highest variance. For all values of $q < 0.5$, the curves are negatively skewed with lesser spread owing to their lower variance.

1.4 Bayes Classifier and Classification

This section studies what effect class proportions have on the posterior probability of prediction. Assume that d, the dimension of X, is 1. Suppose X represents the age of the policyholder. Also assume that the sample proportion of 1 or more claims be 0.2 and let there be no claims for the rest of the sample points. We want to predict the probability $p(y_i = 1|X)$. Suppose X can be split into disjoint sub spaces X_A and $X_{A'}$ ($X = X_A \cup X_{A'}$), then the posterior probability of having a claim from a driver belonging to the age group X_A is

$$p(y_i = 1|X \in X_A) = \frac{p(y_i = 1)p(X \in X_A)|y_i = 1)}{p(X \in X_A)} = \frac{p\left((X \in X_A)\bigcap(y_i = 1)\right)}{p(X \in X_A)}$$

The prior $p(y_i = 1)$ is estimated from the data as $\hat{p}(y_i = 1) = 0.2$

$$\hat{p}(X \in X_A) = \hat{p}(y_i = 1)p(X \in X_A)|y_i = 1) + \hat{p}(y_i = 0)p(X \in X_A)|y_i = 0)$$

$$= p\left((X \in X_A)\bigcap(y_i = 1)\right) + p\left((X \in X_A)\bigcap(y_i = 0)\right)$$

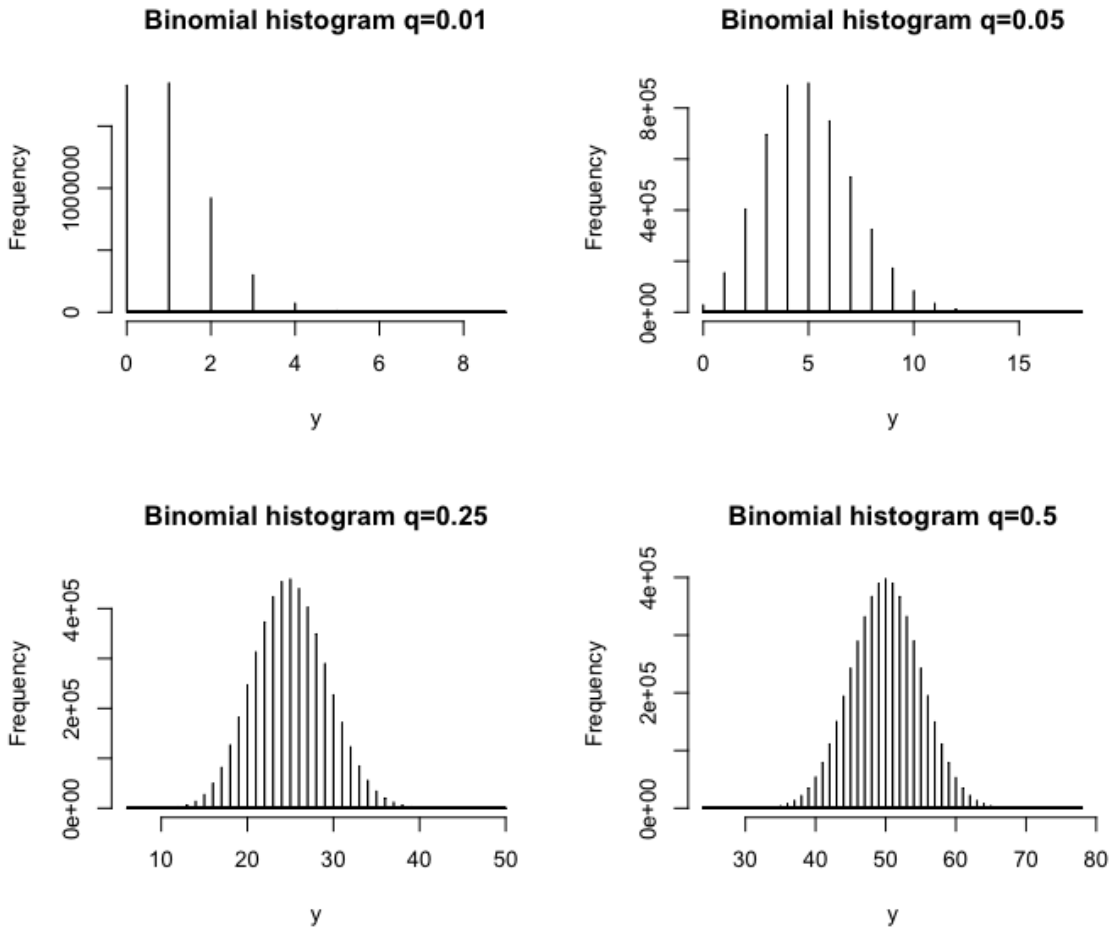


Figure 1.1: Probability mass functions of the binomial distribution (m,q) for $q = 0.01, 0.05, 0.25, 0.5$

Thus the the posterior probability of having a claim in the subspace X_A is

$$p(y_i = 1 | X \in X_A) = \frac{p\left((X \in X_A) \bigcap (y_i = 1)\right)}{p\left((X \in X_A) \bigcap (y_i = 1)\right) + p\left((X \in X_A) \bigcap (y_i = 0)\right)} \tag{2}$$

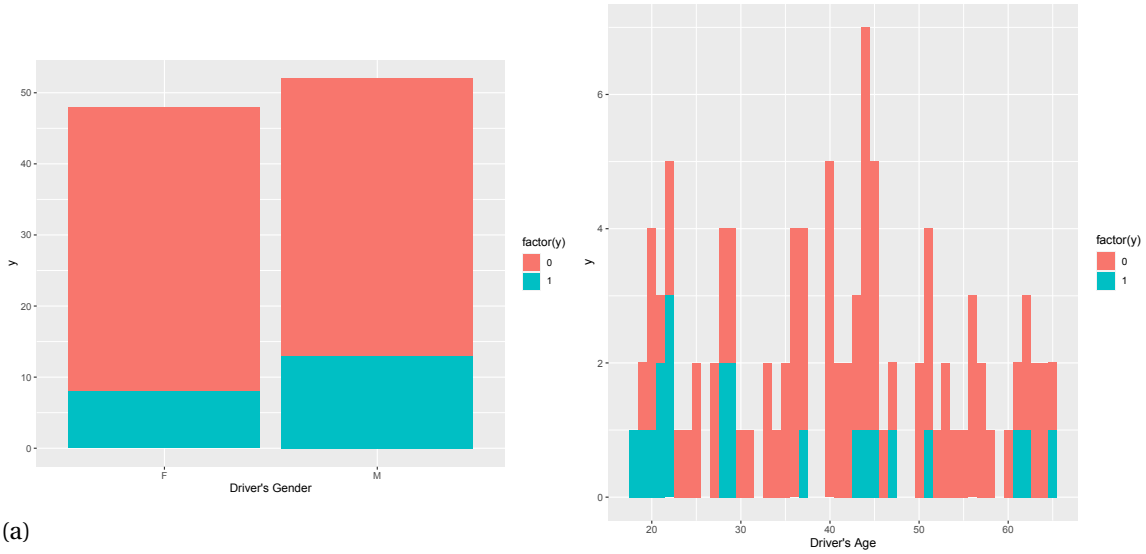
Equation (2) is a measure of the probability of getting the minority class in a given predictor subspace. The class distribution of any given predictor subspace can be one of the following.

1. The subspace has only data points from one of the two classes.
2. There are points from both the classes in X_A

While the first case does not pose much problem but it is the second scenario that creates difficulty in identifying the class especially when the dataset is imbalanced. In case of such predictor spaces, there will be few where the non-zero claim points have clear majority. In most of the subspaces where data from both the class are present, the zero claim points will have overwhelming majority. The posterior probability as obtained in Equation (2) will predict the policy holders as non risky and will mask the risk that come with the attribute in that space. The Bayes posterior probabilities are the ones that any classification algorithm strives to attain. The closer the predicted probabilities are to the actual Bayesian posterior probabilities, better is the model's depiction of the underlying model probabilities. In this paper the suggested subsampling method ends up altering this posterior probabilities of the predictor sub-spaces with overlapping classes. This is done by adding more minority points than there actually is in the predictor sub-spaces. The alteration is more in favour of the minority class (the claimants) so that the risky attributes as well as the risks are recognizable.

Here a toy data with 100 sample points and 21% claims is presented to illustrate this idea. There are two predictor variables age and gender. The age of the drivers are from 18-65 years. There are 21 points with non-zero claims and 79 with no claims. Figure 1.2 shows the class overlaps in the predictor space. Figure 1.2(a) shows the overlap with driver's gender using a barplot and (b) shows the same against driver's age . For a given age/gender the orange portions represent the number of zero claim points and the teal parts give the number of non-zero claims among drivers. Both the genders include close to 20% claimants. Policy holders with claims are present in higher proportion in the age groups 18-22 than the ones with no claims. For the ages 28 and 29 there are equal proportions of the two class. For the rest of the age groups, the non-zero claims, if present are much lesser in proportion. While Equation (2) will predict the age group 18-22 as risky, it will fail to identify most of the risky age groups above 30 years. For 28-29 years the Bayes posterior is equally like to predict any one of the two class.

A classification tree is applied to this data. The tree is fully grown. Figure 1.3 shows the tree. The terminal nodes give the homogeneous predictor subspaces produced by the tree. Figure1.4(a) is the scatter plot of the original data and (b) is the same for the predicted class as obtained by the tree. In both the plots blue circles are zero claims while red triangles are non-zero claims. Thus a blue triangle and a red circle in plot (b) are



(a)

(b)

Figure 1.2: (a) Barplot showing overlap of two classes with driver's gender (a) Barplot showing overlap of the two classes at different age. The zero claim class is in orange and the non-zero claims in teal.

the misclassified data points. As expected in the previous paragraph most of the claim points above the age of 30 have been missed. The zero claims, have been misclassified in the age group 18-22. The age group 28-29 is part of the age group 23-29 and all the claim points are missed. The age group 61-65 has 27% minority data belonging to both the genders. Even by splitting the data into 3 subgroups by age and gender the claimants could not be fully classified.

The tree in Figure 1.3 is impractical because it allows nodes with only one point. In real life the data sets are many fold larger. Growing trees with one point in the terminal nodes is not feasible when the data has hundreds of thousands of points and the number of predictors runs in hundreds. Such a tree has been used here to illustrate how even a full grown tree fails to identify the minority class.

	True (No Claim)	True (Claims)
Predicted (No Claim)	74	12
Predicted (Claims)	5	9

Table 1.1: Confusion Matrix for Toy Data

Table 1.1 is the confusion matrix for the classified toy data. The overall accuracy of the classification is 83%. However only 43% of the risky policy holders could be identified by the model. The minority class constitutes 80% of the 15% of misclassified points. The majority class is favoured by the classification algorithms. Even with the underestimation of the minority class the overall accuracy of prediction of the model is 83% and

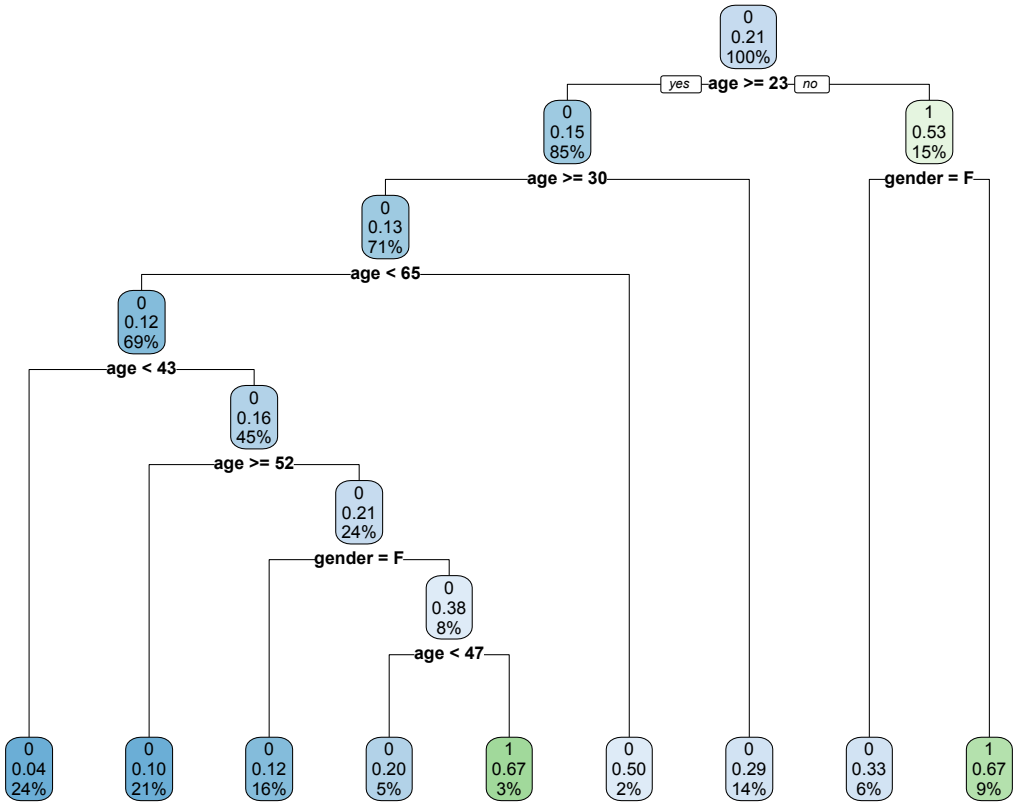
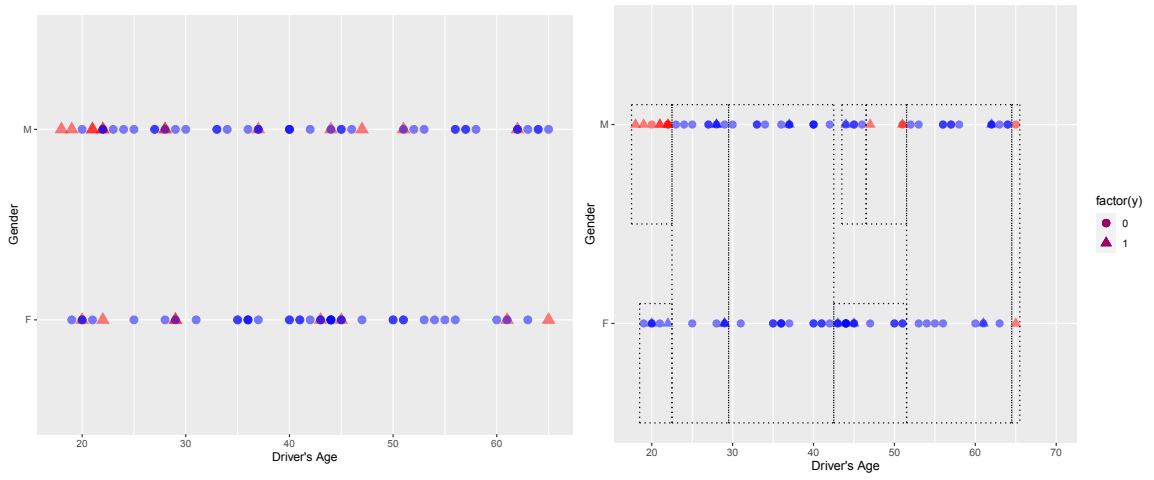


Figure 1.3: The fully grown tree built on the toy data

almost 94% of the non risky policy holders have been identified. This phenomenon is valid and understandable from the perspective of estimation of the population probability of success, but it leads to poor identification of a risky class. Since the proportion of risky customers is usually less than 10% of the data, the bias in their identification affects this small share. It can be overlooked unless the cost of the error is high. For insurance companies it boils down to loss in revenue. The remedy suggested here is to reduce the minority bias. This improvement comes with a cost of diminished accuracy in identifying the non risky customers. Since they hold the major share in data percentage, any small decrease in the accuracy of their identification also affects the overall precision of the model. However, the decrease is marginal. The mathematical explanation of the remedial step is explained in the following sections.



(a) (b)

Figure 1.4: (a) The actual scatter plot of the data against the two predictors age and gender of the drivers. Red triangles are points with claims and the blue circles are without any claims (b) The predictions in 9 terminal nodes. The zero claim predictions are given in blue circles and non-zero claims in red triangles. A red circle or a blue triangle is a misclassified point.

1.4.1 Balancing Method for Imbalanced Data

Hopefully it is now clearer how the the posterior probabilities are tilted towards the majority class in most of the predictor space due to the small proportion of one of the class data. In reality the sample proportions, for insurance datasets, are far from equal. As a remedy, the sample is broken into multiple subsamples, each with equal proportion of the two classes, and a classification model is built on each subsample. The size of each subsample is twice the size of minority class in the original data. Equal number of points from both class are drawn with replacement. Tree based classification methods are used to build the models with each subsample. At every split the binary tree method considers only one attribute in the d dimensional space; the one that best splits the data into two homogeneous groups. After looking into all possible split points within all the d attributes, the attribute that has a split point producing maximum reduction in the heterogeneity is selected.

The improvement of prediction accuracy in the minority class is illustrated using the toy data introduced in Section 1.6. The dimension of the predictor X is 2 ($d = 2$). Let X_d where ($d = 1,2$) has a subspace X_{pB} such that $p[(X \in X_{pB}) \cap (y = 1)] < 0.5$. This condition can be broken into two sub conditions.

1. $p[(X \in X_{pB}) \cap (y = 1)] = 0$: There is no minority sample points in this predictor subspace. For all the subsamples, data from this subspace are always in class 0. A split by any predictor into $\{X_{pB}, X_{pB'}\}$, will have all points in subgroup corresponding to X_{pB} classified as 0s.

2. $0 < p[(X \in X_{pB}) \cap (y = 1)] < 0.5$: In the predictor subspace X_{pB} there are few minority points which are more likely present in all subsamples. Whether a split into $\{X_{pB}, X_{pB'}\}$ will classify the class corresponding to X_{pB} to 0 or 1 depends on the proportion of majority class from this subspace included in X_{pB}. Similar is the argument for $X_{pB'}\}$. It will be illustrated that every minority point has higher chance of being selected in the subsamples than a majority point. This condition will tilt the posterior probability $p[(y = 1)|(X \in X_pB)]$ towards the minority class making its identification easier. The method contributes to some decrease in the accuracy of majority class prediction. One way to mitigate the error is by reducing the classification bias as much as possible. This is equivalent to growing the individual trees fully. Apart from large trees further tweaking of proportions helps in improving majority class error. It has been discussed in Section 1.7.2.

Relation Between Homogeneity and Variance in CART

In the Classification and Regression Trees (CART) algorithm, when a node is split into two, a natural goodness of a the split criterion could be the one that maximizes the reduction in the nodes misclassification error. This criterion, unfortunately, has a serious drawback as discussed in Breiman's Classification and Regression Trees Breiman (2017). The construction of a binary tree revolves around 3 elements:

1. The selection of splits.
2. The decision when to declare a node terminal or when to split it further.
3. The assignment of each terminal node to a class.

Some terminology pertaining to the CART is introduced here. Nodes are denoted as $t_1, t_2, t_3, ..., t_m$, where t_1 is the root node which is split into t_2 and t_3. In the next split t_3 gets split into t_4 and t_5 while t_2 gets split into t_6 and t_7 and so on. Suppose the variable j denote the number of classes in the data. If there are 6 classes, j = 1,..,6. For binary classification j = 1,2.

1. Define node proportion $p(j|t)$, to be the proportion of the cases in t to belong to the class j. Hence $\Sigma_j p(j|t) = 1$.
2. Define a measure $i(t)$ of the impurity of node t as a non-negative function ϕ of $p(j|t)$ such that ϕ is maximum when the node has all the classes in equal proportion and minimum when all the cases in the node belong to any one class. For example when j = 3,
   $\phi(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}) = maximum$
   $\phi(1,0,0) = \phi(0,1,0) = \phi(0,0,1) = 0$

   For any node t, suppose there is a candidate split s of the node which divides it into descendants t_L and t_R, such that the proportion of cases of t to go to t_L is p_L

and the proportion to go to t_R is p_R. Then the goodness of the split is the decrease in the impurity

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R)$$

3. Define a candidate set S of binary splits s at each node. Generally, the set S of splits can be conceived of as a set of questions, where each question is of the form: Is $x \in A? A \subset \mathbf{X}$.
Then the associated split s sends all the cases in t that answer "yes" to t_L and all cases with answer "no" to t_R. Thus $t_L = t \cap A$ and $t_R = t \cap A^c$. To grow a full tree, at the root node t_1, a search is made through all the candidate splits over all $x \in \mathbf{X}$ to find that split s^* which gave the largest decrease in impurity; i.e,

$$\Delta i(s^*, t_1) = max_{s \in S} \Delta i(s, t_1)$$

Then t_1 is split into t_2 and t_3 using the split s^* and the same search procedure is repeated for the best $s \in S$ for t_2 and t_3 separately. The tree is grown in this way until the terminal conditions are met.
The class of the cases in a terminal node is determined by the plurality rule. If

$$p(j_0|t) = max_j p(j|t)$$

then t is designated as a class j_0 terminal node.

4. Define resubstitution estimate $r(t)$ of the probability of misclassification, given that a case falls into node t, is

$$r(t) = 1 - max_j p(j|t)$$

If the classification is binary then

$$r(t) = min_j p(j|t)$$

Denote the resubstitution estimate for the misclassification rate at a node t as

$$R(t) = r(t)p(t) \tag{3}$$

Let T be the final tree and $\tilde{T}$ be the set of all the terminal nodes of T. Given any terminal node t, the resubstitution estimate for the overall misclassification rate for the tree T is given by

$$R(T) = \Sigma_{t \in \tilde{T}} R(t) = \Sigma_{t \in \tilde{T}} r(t)p(t) = \Sigma_{t \in \tilde{T}} \Sigma_{j \neq max_j} p(j|t)p(t) = \Sigma_{t \in \tilde{T}} \Sigma_{j \neq max_j} p(j \cap t) \tag{4}$$

In other words, for every $t \in \tilde{T}$ the probability of misclassification is the probability of the case to be in node t and to belong to the class that is not assigned to the node.

Definition 2 *An impurity function is a function ϕ defined on the set of all J-tuples of numbers $(p_1, p_2, ..., p_j)$ satisfying $p_j \geq 0, j = 1, 2, ...J, \Sigma_j p_j = 1$ with the properties*

(i) ϕ is maximum only at the point $(\frac{1}{J}, \frac{1}{J}, ..., \frac{1}{J})$,

(ii) ϕ achieves its minimum only at points (1,0,..., 0), (0,1, ...,0), ..., (0,...,0,1),

(iii) ϕ is a symmetric function of $p_1, p_2, ...p_j$.

If reduction in the nodes misclassification error rate is considered to determine splits then the best split of t would maximize

$$r(t) - p_L r(t_L) - p_R r(t_R)$$

or, equivalently, would maximize

$$R(t) - R(t_L) - R(t_R)$$

The corresponding node impurity function is

$$\phi(p_1, p_2, ..., p_j) = 1 - max_j p_j$$

This function has all the desirable properties listed in Definition 2. Even then, selecting splits that maximize the reduction in $R(T)$ is not desirable. The following discussion explains why it is so.

For any split of a node t into t_L and t_R,

$$R(t) \geq R(t_L) + R(t_R)$$

The equality holds only when the predicted class in all the three nodes t, t_L, t_R denoted by $j(t), j(t_L), j(t_R)$ respectively are the same i.e.

$$j(t) = j(t_L) = j(t_R)$$

In any binary class problem where one class is present in a larger number in node t (the zero claim class in our case for example), it is possible that every best split generated produces nodes t_L and t_R, where both have the same class as majority as in t. Then $R(t) - R(t_L) - R(t_R) = 0$ for all splits.

There is also a second defect. Consider an example from Breiman (2017) Classification and Regression Trees as illustrated in Figure 1.5. Here the root node of the tree has equal proportion of both classes. The first split leads to a tree with 200 out of 800 cases misclassified. $R(T) = \frac{200}{800} = 0.25$. The second split also misclassifies 200 cases and has $R(T) = 0.25$. Even though both the splits are given equal ratings by the $R(T)$ criterion, the second tree is probably preferable in terms of future growth of the tree. For the first split $r(t_L) = r(t_R) = 0.25$. Both these nodes will need more splitting to get a tree with lower value of $R(T)$. In the tree from the second split $r(t_L) = 0.33$ and $r(t_R) = 0$. Thus $r(t_R)$ is a terminal node with perfect classification accuracy. It accounts for $\frac{1}{4}^{th}$ of the

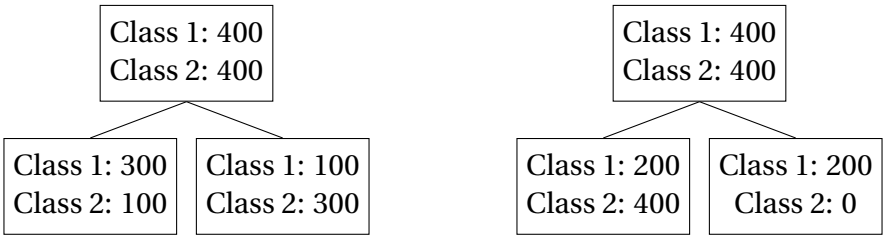


Figure 1.5: Two possible splits starting at the same node

total cases. Though the second tree is more preferable, the $R(T)$ minimization criterion cannot differentiate between the two.

In the binary classification problem, where p_1 and p_2 are the two class probabilities, the node impurity function corresponding to the node misclassification rate is

$$\phi(p_1, p_2) = 1 - max(p_1, p_2) = min(p_1, p_2) = min(p_1, 1 - p_1).$$

Figure 1.6 shows the graph for the function. Though this function satisfies all the conditions of Definition 2 it does not sufficiently reward purer nodes.

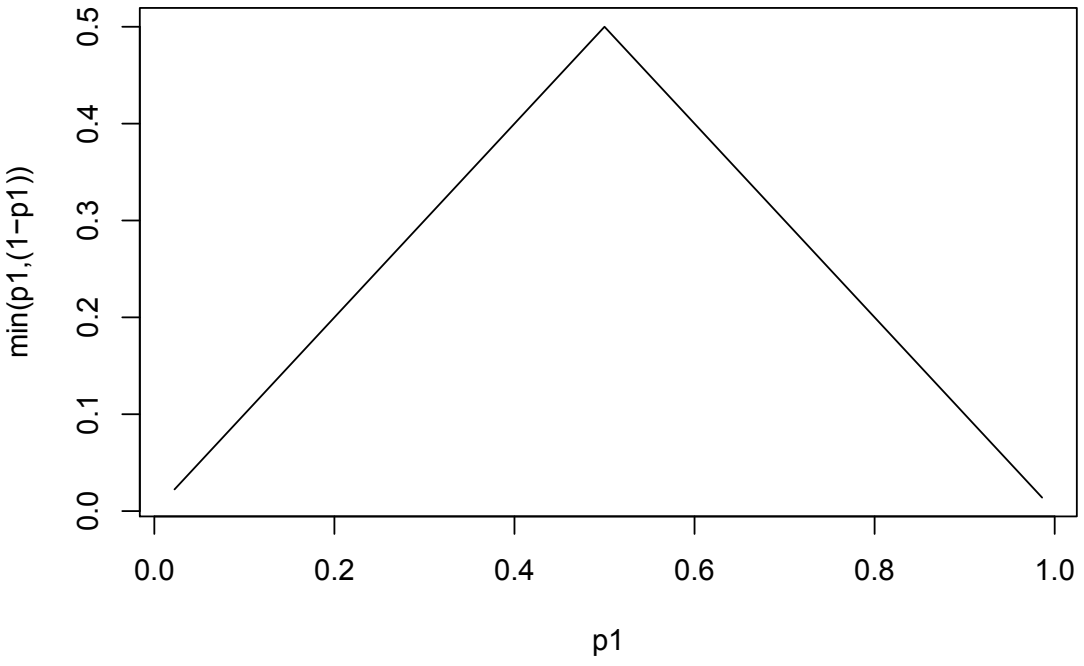


Figure 1.6: The impurity function corresponding to minimization of R(T) criterion

Suppose $p_1 > 0.5$; then $\phi(p_1) = 1 - p_1$ decreases linearly in p_1. To have a criterion that would select the second split in Figure 1.5, it is necessary that the impurity function corresponding to the criterion decreases faster than linearly as p_1 decreases. This can be constructed by imposing the condition that if $p_1'' > p_1'$ then $\phi(p_1'')$ is less than the corresponding point on the tangent line at p_1'.

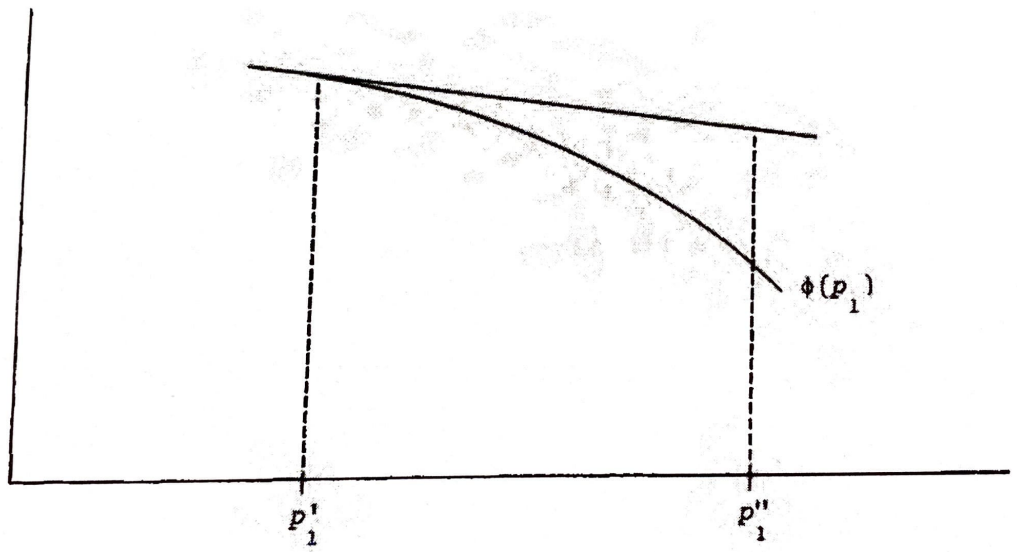


Figure 1.7: The concave node impurity function. The graph is from Breiman's Classification and Regression Trees Breiman (2017)

If ϕ has a continuous second derivative on [0,1] then the strict concavity would translate into $\phi''(p_1) < 0$, $0 < p_1 < 1$. Let $\phi(x) = a + bx + cx^2$. The function must satisfy all the conditions of Definition 2. Hence $\phi(0) = \phi(1)$ which gives $a = 0$ and $b + c = 0$. So $\phi(x) = b(x - x^2)$. The condition $\phi(p_1) = \phi(1 - p_1)$ implies that $b > 1$. Without loss of generality we can assume $b = 1$, giving $\phi(x) = x(1 - x)$. Thus for a node t, the impurity function $i(t)$ can be

$$i(t) = \phi(p_1) = p_1(1 - p_1)$$

Replacing p_1 by q, where q is the probability of non-zero claims we see that $i(t) = q(1 - q)$, is also the variance of the node t, which when minimized, minimizes the node impurity.

1.5 Bias and Variance in Classification

Unlike regression, the response variable of a classification problem is not numerical and continuous. Bias and variance of a classifier as explained in Breiman (1996b) and Tibshirani (1996) is used in this section to argue in favour of the suggested subsampling and

aggregation methods. The bias and variance are obtained from decomposing the prediction error in classification problems. Let the learning sample used to train a model be denoted by $\mathscr{L} = \{(y_i, x_i), i = 1,2,...n\}$ where y_i are categorical outputs and x_is are multidimensional input vectors. Some function is applied to these input vectors to construct predictors of future y values. The response variable $Y \in \{1,2,...J\}$ represents class labels. In binary classification $Y \in \{0,1\}$. Given $\mathscr{L}$ some method $C(x,\mathscr{L})$ is constructed to predict ys, the observed values of Y. The letter $\mathscr{L}$ in the function is to show that the function is built using the xs in the learning set $\mathscr{L}$. Assume that the training set consists of iid samples from the unknown probability distribution of (Y,X). The misclassification error of this sample is

$$PE(C(,\mathscr{L})) = E_{X,Y}[C(X,\mathscr{L}) \neq Y] = P_{X,Y}(C(X,\mathscr{L}) \neq Y)$$

and $PE(C)$ is the expected value of $PE(C(,\mathscr{L})$ over $\mathscr{L}$.

$$PE(C) = E[P_{X,Y}(C(X,\mathscr{L}) \neq Y)] \tag{5}$$

Denote:

$$P(j|X) = P(Y = j|X = x)$$

$$P(dx) = P(X \in dx)$$

Let C^* denote the Bayes classifier. Minimum misclassification error is given by the Bayes classifier.

$$C^*(x) = argmax_j P(j|x)$$

with the misclassification error rate

$$PE(C^*) = P(C^*(x) \neq Y) = 1 - \int max_j(P(j|x))P(dx)$$

Now, define the probability that the classifier C predicts a class j conditioned on the predictor as

$$Q(j|x) = P_{\mathscr{L}}(C(x,\mathscr{L}) = j)$$

and define an aggregated classifier as:

$$C_A(x) = argmax_j Q(j|x)$$

This is aggregation by voting. Thus if there are many independent replicas of the training sample $\mathscr{L}_1, \mathscr{L}_2, ...\mathscr{L}_B$, and classifiers constructed on them are $C(x,\mathscr{L}_1)$, $C(x,\mathscr{L}_2)$, ... $C(x,\mathscr{L}_B)$ then at each x the class output of $C_A(x)$ is the most popular class output of the above B classifiers. The definition of an unbiased classifier follows.

Definition 3 *$C(x,\mathscr{L})$ is unbiased at x if*

$$C_A(x) = C^*(x)$$

This means that if $C(x, \mathscr{L})$ is unbiased at x then over replications of $\mathscr{L}$, $C(x, \mathscr{L})$ will pick up the right class more often than any other class. It is not necessary that a classifier that is unbiased is the most accurate classifier. Going back to our toy example where $P(0|x) = 0.9$ and $P(1|x) = 0.1$, if we have C such that $Q(0|x) = 0.6$ and $Q(1|x) = 0.4$ then C is an unbiased classifier at x even though the probability of a correct classification of C at x is 0.9x0.6 + 0.1x0.6 = 0.58.

If C is unbiased at x then $C_A(x)$ is optimal. Let U be the set of all x where C is unbiased. The complement of U is the bias set denoted by U'.

Definition 4 *The bias of a classifier C is*

$$Bias(C) = P_{X,Y}(C^*(X) = Y, X \in U') - E_{\mathscr{L}}[P_{X,Y}(C(X, \mathscr{L}) = Y, X \in U')]$$

and its variance is

$$Var(C) = P_{X,Y}(C^*(X) = Y, X \in U) - E_{\mathscr{L}}[P_{X,Y}(C(X, \mathscr{L}) = Y, X \in U)]$$

Thus

$$\begin{aligned} Bias(C) + Var(C) &= P_{X,Y}\left(C^*(X) = Y, X \in U'\right) + P_{X,Y}\left(C^*(X) = Y, X \in U\right) \\ -E_{\mathscr{L}}\left[P_{X,Y}(C(X, \mathscr{L}) = Y, X \in U')\right] &- E_{\mathscr{L}}\left[P_{X,Y}(C(X, \mathscr{L}) = Y, X \in U)\right] \\ &= P_{X,Y}\left(C^*(X) = Y\right) - E_{\mathscr{L}}\left[P_{X,Y}(C(X, \mathscr{L}) = Y)\right] \\ &= 1 - P_{X,Y}\left(C^*(X) \neq Y\right) - E_{\mathscr{L}}\left[P_{X,Y}(C(X, \mathscr{L}) = Y)\right] \\ &= E_{\mathscr{L}}\left[1 - P_{X,Y}(C(X, \mathscr{L}) = Y)\right] - PE\left(C^*\right) \\ &= E_{\mathscr{L}}\left[P_{X,Y}(C(X, \mathscr{L}) \neq Y\right] - PE\left(C^*\right) \end{aligned}$$

Therefore using Equation (5)

$$PE(C) = Bias(C) + Var(C) + PE(C^*) \tag{6}$$

Here, it is to be noted that replacing C with C_A reduces the variance to zero. However there is no guarantee that it will reduce the bias. Recollect that the bias within each classifier has been reduced to its maximum by using large unpruned trees. Certain properties of the bias and variance become obvious from the above

(a) Bias and variance are always positive. $Q(j|X)$ is a probability and

$$Var(C) = \int_U \left[max_j P(j|X) - \Sigma_j Q(j|x)P(j|x)\right] P(dx)$$

Similarly

$$Bias(C) = \int_{U'} \left[max_j P(j|X) - \Sigma_j Q(j|x)P(j|x)\right] P(dx)$$

(b) The variance of C_A is zero. This is obvious from Definition 3. $C_A = C^*$ in U.

(c) If C is deterministic, i.e, if C does not depend on $\mathscr{L}$, then variance in zero.

(d) The bias of C^* is zero.

1.5.1 Bagging as the Method of Aggregation

Bagging (Bootstrap Aggregation) is the method of creating multiple versions of predictors over independent and identically distributed samples and creating an aggregated predictor by averaging the results of the multiple versions created. Instead of creating one predictor with the learning sample, it is perturbed in such a way that the results obtained from each perturbation is significantly different from the one predictor. The learning set $\mathscr{L}$ described in the previous section can be used to create multiple bootstrap sets $\mathscr{L}_1, \mathscr{L}_2 ... \mathscr{L}_B$. The number B is large and subjective. Usually each $\mathscr{L}_b$ is of the same size as $\mathscr{L}$. Perturbation is introduced by drawing same number of sample points with replacement from $\mathscr{L}$ to construct each $\mathscr{L}_b$. This makes the sample points iid but introduces more bias within each $\mathscr{L}_b$. Having created repeated bootstrap samples, a predictor is trained on each of them as $C_b = C(, \mathscr{L}_b)$.

If y is numerical the final aggregate predictor is an average of all the predictors. In classification the ys are class labels where the final aggregate predictor is the class with the maximum vote. Bagging works when each classifier is unstable and largely different from the single classifier. Instability comes when the results change largely with small change in data. If each of the C_bs are not very different from C then not much improvement can be expected from the procedure. Unstable classification and regression methods are methods like neural networks, classification and regression trees while methods like k-nearest neighbours, linear discriminant analysis etc., are stable methods.

In Breiman (1996a) on bagging predictors, the overall probability of correct classification is given as

$$r = \int [Q(j|x)P(j|x)]P(dx)$$

where $\Sigma_j Q(j|x)P(j|x)$ is the probability that a classifier predicts the class correctly at x. It is worth noting here that

$$\Sigma_j Q(j|x)P(j|x) \leq P(j|x)$$

The equality holds only if

$$Q(j|x) = \begin{cases} Y\ s1 & \text{if } P(j|x) = max_i P(i|x) \\ 0 & \text{otherwise} \end{cases}$$

The Bayes predictor $C^*(x)$ leads to the above expression for $Q(j|x)$ and gives the highest attainable classification rate as

$$r^* = \int max_j P(j|x)P(dx)$$

The region of predictor space where the classifier C is unbiased is also known as order correct. Thus for all the xs C is order-correct or unbiased if

$$argmax_j Q(j|x) = argmax_j P(j|X)$$

This means that if class j is the most frequently predicted class at x and C also predicts the same class at x, then C is order correct. Note that C need not be the most accurate classifier as explained in the example with Definition 3. When we combine multiple

classifiers $C_1, C_2, ...C_B$ with bagging, the probability of correct classification by the aggregate classifier C_A is

$$\Sigma_j \mathbb{1}(argmax_i Q(i|x) = j)P(j|x) \tag{7}$$

If C is order correct at x then, Equation (7) equals $max_j P(j|x)$. U' is the set of all xs where C is not order correct, the probability of correct classification of C_A is given by

$$r_A = \int_{x\in U} max_j P(j|x)P(dx) + \int_{x\in U'} [\Sigma_j \mathbb{1}(C_A(x) = j)P(j|x)]P(dx)$$

Even if a classifier C_b is order correct at x, its correct classification rate can be far from optimal. But the aggregate classifier C_A is optimal. If a classifier is good in the sense that it is order correct/unbiased for most of the xs then aggregation can transform it into a nearly optimal one.

How Does the Suggested Subsampling Decrease Minority Misclassification

This section clarifies how the method of subsampling suggested in this proposal primarily improves misclassification error of the minority class. It is worth noting at this point that each subsample $\mathscr{L}_b$ is created from $\mathscr{L}$ in such a way that the Bayes classifier gets tweaked so that the posterior probabilities favour the minority class for those xs where both the class points are present. Let this tampered Bayes classifier be called C_s^*. Also, the C_{sb} be the classifier built with the subsample $\mathscr{L}_b$. For a sample size n of $\mathscr{L}$ the probability that a sample point is included in a sample $\mathscr{L}_b$ is

$$1 - \left(1 - \frac{1}{n}\right)^n \to 1 - e^{-1},$$

when n is large. Thus, asymptotically, at least 63% of the sample points are included in each bootstrap sample.

Since method suggested in this proposal is not to improve the overall accuracy of prediction but to improve the minority class prediction accuracy, it comes with a little compromise in the prediction accuracy of the majority class as well as the overall accuracy. Each sample $\mathscr{L}_b$ is not of size n but $2nq'$ where q' is the proportion of the minority class in $\mathscr{L}$. The probability that a minority class point is included in the sample is $1 - \left(1 - \frac{1}{nq'}\right)^{nq'} \to 1 - e^{-1} = 0.63$ whenever $nq' \geq 100$. However, it is not the same for the majority class. The probability for a majority class point to be included in a sample is

$$1 - \left(1 - \frac{1}{n(1-q')}\right)^{nq'} = 1 - \left(1 - \frac{1}{n(1-q')}\right)^{n(1-q')\frac{q'}{(1-q')}} \tag{8}$$

When n is large so that $n(1-q') > 100$ then $\left(1 - \frac{1}{n(1-q')}\right)^{n(1-q')} \to e^{-1}$. Therefore, the probability of a majority class to be included in a subsample moves close to $1-(e^{-1})^{\frac{q'}{(1-q')}}$.

n	Probability of inclusion with q'=0.05	Probability of inclusion with q'=0.1	Probability of inclusion with q'=0.2
100	0.051535	0.105717	0.222425
1000	0.051297	0.105216	0.221321
10000	0.051273	0.105166	0.221211
100000	0.051270	0.105161	0.221200

Table 1.2: Table showing the probability of inclusion in a sample for a majority sample point. The minority proportions q' at 5%, 10%, and 20%. Larger the n, the values move towards the limit $1-(e^{-1})^{\frac{q'}{(1-q')}}$.

Table 1.2 gives the probability of inclusion for each majority class sample point in a sub sample. The probability decreases with the increase in imbalance. This lower inclusion of the majority class contributes to lower posterior probabilities for the majority class, that favours sieving the minorities out.

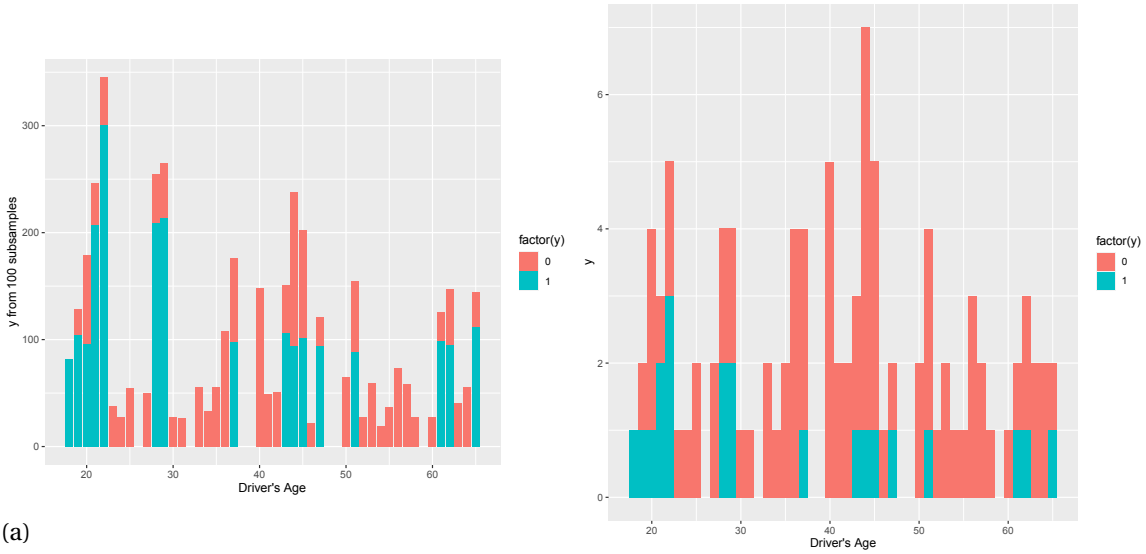
It is obvious from the above argument that in the suggested subsampling each minority data point has higher probability of included in a subsample than a point from the majority class. From the fourth column of Table 1.2 we can see that given a x while a claim data has 63% chance, a zero-claim point has 22% chance of being included in any subsample. If X_A is a predictor subspace where data points from both classes are present in equal numbers, then for every subsample the probability of having at least one point with zero-claims is much lower than the same for a point with claims. This happens consistently on all the subsamples drawn with replacement from the initial sample. The posterior probability $P_s(y=1|X_A) > P_s(y=0|X_A)$ and hence $C_s^*(X_A)$ predicts $Y=1$ whenever $x \in X_A$. The suffix s has been used to differentiate the altered posterior probabilities from subsampling. At all the predictor space where the a classification method is order correct (or unbiased), the aggregate classifier C_{sA} built using all the subsamples will be optimal and equal to C_s^* in predictive accuracy. Policy holders with x attributes, which have both the classes in equal proportions in the main sample, will be now classified as risky. It goes without saying that the same will hold wherever the minority class is present in higher proportions.

From Equation (8) the probability that a given point from the majority class is not included in a subsample is $e^{-\frac{q'}{(1-q')}}$. Let there be m points with no claims and one point with more than zero claim in the space X_A. In a given subsample, the probability that none of the zero-claim points are included is $e^{-\frac{mq'}{(1-q')}}$. The probability that the single claimant data point is part of the sample is $1-e^{-1}$ Equating this with the probability that at least one point from the majority class is included in the sample we get

$$1-e^{-1} = 1-e^{-\frac{mq'}{(1-q')}},$$

when $m = \frac{1-q'}{q'}$. Therefore, the posterior probability $P_s(y=0|X_A) \geq P_s(y=1|x_A)$ when

$m \geq \frac{1-q'}{q'}$. In other words, until the number of zero-claim points is at least m times the number of claimants, the posterior probability and the Bayes classifier will predict the policy holders from that space as risky. This phenomenon can be illustrated using the toy data of 100 points introduced in Section 1.6. Recollect the data has approximately 21% of the points belonging to the minority class. Thus $m = \frac{.79}{.21} = 3.76$. 100 bootstrap samples, each of size 42 with 21 points drawn with replacement from each class, were collected from the main sample. These samples were combined into one and a barplot of the two classes in each age group is shown in Figure 1.8(a). Barplot of the actual sample proportions by age is shown in Figure 1.8(b).



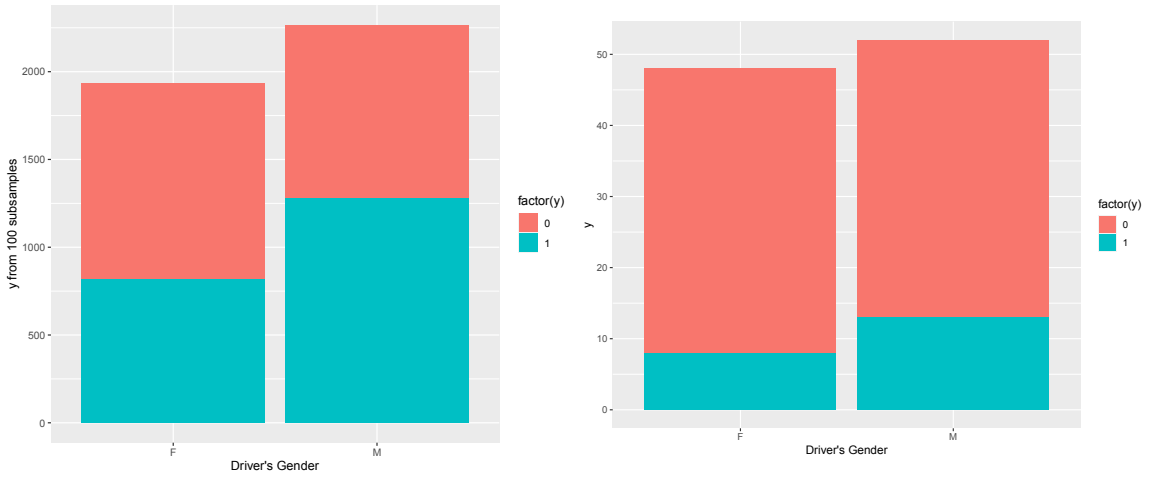
(a)

(b)

Figure 1.8: (a) Barplot showing the proportion of two classes at different age with 100 subsample points combined (b) Barplot showing the proportion of two classes at different age in the original sample

Table A.1 in the Appendix A, shows the proportion of the two class in the original sample for different ages. There are certain age groups where there is no overlap of the data points like 18, 23-27, 30-36, 40-42, 46, 50, 52-60, and 63-64 years. For all other ages data from both classes are present. For ages 44 and 45 the number of majority points is at least 3.76 times the number of minority points. From Figure 1.8(a) we can observe that age 45 has equal number of points from both class because the number of zero-claims is 4 times that of non-zero claims. At 44 years the majority class is 7 times more than the minority class in number. The combined sample has above 60% points from the majority class and around 39% from the minority class at age 44. The age groups with no minority class have subsample points only from the majority class. Likewise, only minority points are sampled for age groups with no majority points which is age 18. Most of the age groups with overlap of both the class (except 44 and 45 years of age) fail to exceed the m value and have lesser points from the majority class than the

minority. Hence, the posterior probability for $y = 1$ is higher for all these age group and consequently classified as risky by the Bayes classifier C_s^*. The aggregate classifier C_{sA} will be equal to C_s^* for all the x where C_{sb}s are unbiased. The fourth column of Table A.1 gives the class prediction by C_{sA}. The fifth column is the prediction by the Bayes Classifier C_s^*. The mismatches in these two columns make the corresponding xs fall in U', the region of bias in the predictor space.



(a) (b)

Figure 1.9: (a) Barplot showing the proportion of two classes for the combined samples for the two genders (b) Barplot showing the proportion of two classes for the two genders in the original sample

In Figure 1.9(a) we see the barplot of the 100 combined samples split by the gender of the drivers. In the original sample 8 of the 48 female drivers have claims making the non-risky class 5 times the risky class. On the other hand, there are 13 risky drivers out of 52 male drivers making the ratio of the majority class to minority 3:1 for male drivers. Likewise the posterior probability would predict male drivers as risky and the females as not-risky.

Table 1.3. shows confusion matrices obtained by the aggregate classifier C_{sA} of two sets of trees. In Table 1.3(a) the minority class identification accuracy has gone up to 81% from 38% as in Table1.1. However, this comes with the cost of increased false positives resulting in a decrease of the overall accuracy. There is further improvement in the minority class identification with the second aggregation as can be seen in Table 1.3(b). This improvement is attributed to bigger trees used in the aggregate. However, the false positives have increased while the true negatives have suffered more. In the following section it has been shown that the compromise in the accuracy of the majority class can be curtailed while preserving the improvement in the minority class identification.

[a]

	True (No Claim)	True (Claims)
Predicted (No Claim)	57	4
Predicted (Claims)	22	17

[b]

	True (No Claim)	True (Claims)
Predicted (No Claim)	55	3
Predicted (Claims)	24	18

Table 1.3: Confusion Matrices for Toy Data predictions obtained from the aggregate classifier using (a) classification tree that allows split when there are at least 5 points in a node (b) classification tree that allows split when there are at least 3 points in a node.

1.5.2 How Subsampling with Different Proportions can Curtail the Majority Class Bias

Recall once more that when the initial sample size is n and we select a sample of same size with replacement from it, every point in the sample has a probability of $1-e^{-1}$ to be selected if n is large. Suppose the sample has nq points with at least one claim and $n(1-q)$ points with no claims. Also suppose $q << 0.5$. Instead of drawing equal proportions of both the classes in each subsample, each subsample is now of size n with np members drawn from the claim points and $n(1-p)$ points from the zero-claim points. $0 < p < 1$ and it is necessary that $p > q$. Thus the probability that any given non-zero claim point will be included in a sample is

$$1-\left(1-\frac{1}{nq}\right)^{np}$$

$$=1-\left(1-\frac{1}{nq}\right)^{nq\frac{p}{q}} \to 1-e^{-\frac{p}{q}},$$

when nq is large. Since $p > q$, $\frac{p}{q} > 1$ and thus probability is more than $1-e^{-1}$. Considering the toy data from the previous section again where $q \approx 0.2$, if each subsample of size 100 includes 40% data from the minority class then each minority point has 86% chance of being included in a subsample. This would tilt the posterior probabilities more towards the minority class but also decrease the majority bias. The explanation follows.

Consider the predictor subspace X_A again, where there are m zero-claim points and only 1 point with some claim. Probability that none of these m points are selected in the sample is

$$\left(1-\frac{m}{n(1-q)}\right)^{n(1-p)}$$

$$=\left(1-\frac{m}{n(1-q)}\right)^{n(1-q)\frac{1-p}{1-q}} \to (e^{-m})^{\frac{1-p}{1-q}},$$

when n is large enough. Thus the probability of at least one point from the majority class

to be selected in any subsample is

$$1 - e^{-\frac{m(1-p)}{(1-q)}}$$

And points from both the class become equally probable in a sub sample if

$$\frac{p}{q} = \frac{m(1-p)}{1-q}$$

or,

$$m = \frac{p(1-q)}{q(1-p)}$$

By altering the proportion p the majority bias can be controlled. Going back to the toy data example, when $p = 0.4$, $m \approx 2.5$. Thus for all predictor spaces where number of majority class points is at least 2.5 times the number of minority points, the posterior probability will be the same for both the class. Recall that m was equal to 3.76 when equal proportions of both the class were included in every subsample. Higher proportion of majority points was necessary for the class to be as likely as the minority class.

The toy data of 100 points is used again to illustrate the results. This time each subsample has 100 points with 40% of them being drawn from the minority class and 60% from the majority class. The $m \approx 2.5$. Thus any age group that has at least 2.5 times more of zero-claim points than claim points will have a chance of the Bayes classifier favouring the non-risky class. Figure 1.10 compares the combined points of 100 samples from subsampling with $p = 0.4$ with that of subsamples where equal proportion of both class ($p = 0.5$) were sampled. Note that in 1.10(a) where $p = 0.4$ the share of the orange part is increased over all the age group where both the class are present. The ages 37 and 51 has exactly 3 times more majority points than minority and note how the share of the majority class is more than the minority class in the bar plot (a) while it is less than 50% for bar plot (b).

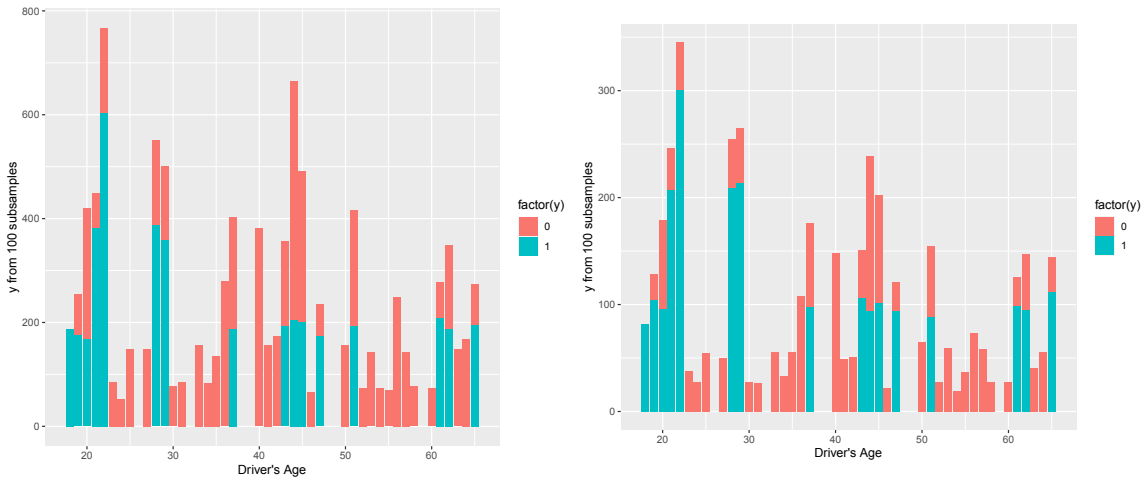
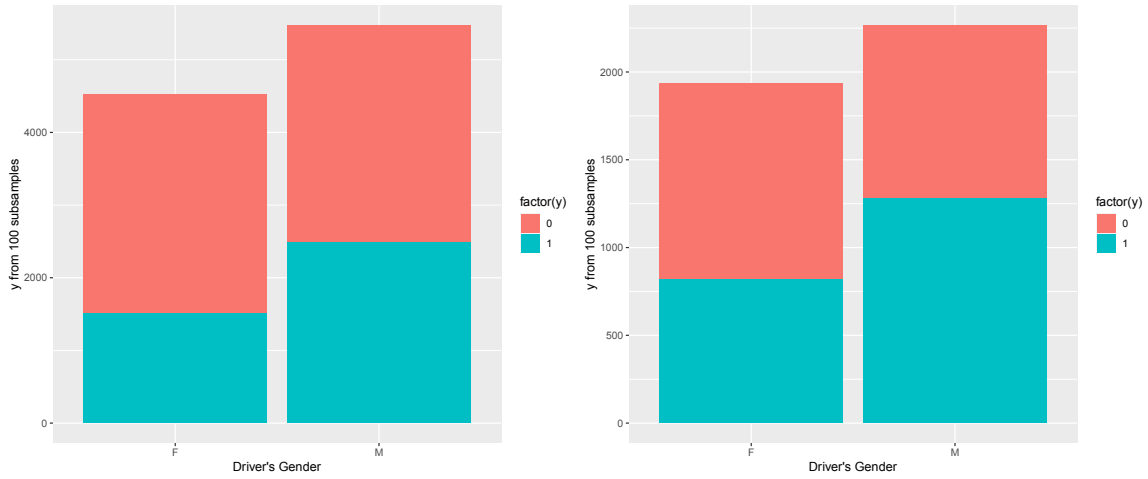


Figure 1.10: (a) Barplot showing the class proportions at different ages with 100 subsample points combined when $p = 0.4$ (b) Same when $p = 0.5$

Similarly Figure 1.11(a) shows the proportion of the two classes for female and male drivers in 100 subsamples when $p = 0.4$. Compare it with the proportions in Figure 1.11(b) where $p = 0.5$. The increase in the share of majority class is clearly visible is the plot of (a).



(a) (b)

Figure 1.11: (a) Barplot showing the class proportions for the combined samples for the two genders when $p = 0.4$ (b) Same when $p = 0.5$

Table 1.4 is the confusion matrix obtained from the aggregate classifier C_{sA} when $p = 0.4$. The improvement in the majority class accuracy in table (a) can be observed by comparing with Table 1.3(a). While maintaining the improved prediction accuracy for the minority class, the majority class prediction could be improved by 9%. In Table1.4(b) which is from larger trees the improvement in the majority class prediction is accompanied by further improvement in the minority class prediction. The optimal p value would have to be decided depending on how much the cost is of missing a risky driver. Multiple p values has to be considered to find the optimal choice for a given data set.

[a]

	True (No claim)	True (Claims)
Predicted (No claim)	64	4
Predicted (Claims)	15	17

[b]

	True (No claim)	True (Claims)
Predicted (No claim)	62	1
Predicted (Claims)	17	20

Table 1.4: Confusion Matrices for Toy Data predictions obtained from the aggregate classifier when $p = 0.4$. Table(a) is aggregated from classification trees that allow a split when there are at least 5 points in a node. Table (b) is the confusion matrix from aggregate of classification trees that splits when there are at least 3 points in a node.

Chapter 2

Preprocessing Dataset with Regression Tree to Improve GLM Output

2.1 Abstract

Generalized Linear Models are a traditional way to model claim numbers and claim sizes for insurance contracts. Insurance datasets are notoriously imbalanced with large proportions of policyholders with zero claims leading to lower prediction accuracy of results obtained from traditional models. Tree based methods recursively divide the data into binary groups, that are more homogeneous, within groups. However, the GLM method is preferred by actuaries over more accurate machine learning methods because of its higher interpretability. In this chapter a hybrid model is described where variables and interactions between them are retrieved from the splits of the tree built on the data. A GLM model built using only these variables selected from trees are not only more accurate than a GLM without such preprocessing but also attains better prediction accuracy compared to the optimal tree. The results are empirically illustrated on a public dataset.

2.2 Introduction

The major challenge posed by insurance datasets is the striking imbalance owing to the fact that most of the policyholders do not file any claims over the observed period of time. Claim occurrences are rare events. In addition, sometimes when claims occur the customers prefer to not report them if they are small, or smaller than the policy deductible amount. This reluctance in reporting is to avoid an increase in future premiums that might result from a claim report. Thus it is apparent that the distribution of the claims has an inordinate mass at zero. Getting reasonably good prediction accuracy with these highly imbalanced datasets is an uphill task because most of the statistical methods work best under the tacit assumption of more or less balanced data. While a traditional Generalized Linear Model or a more contemporary Regression tree model work as best they can, a collaboration between the two models seems to achieve better

results.

Generalized Linear Models have been a popular tool in actuarial data analysis, especially to obtain the expected frequency and severity of claims. The GLM method was introduced by Nelder and Wedderburn (1972). GLMs extend the traditional assumption of a linear relation between the claim frequencies and the independent variables to non-linearity, by using probability distributions from the exponential family for the response variable. While Poisson is a popular choice for claim frequency, the gamma distribution is favoured when it comes to modelling claim size. McCullagh and Nelder (1989) illustrate the application of GLMs in predicting claim severity in motor insurance. There exists extensive work illustrating the effectiveness of GLM on insurance tariffs. Renshaw (1994) show how GLM can be used to analyze the claims frequency and claims severity based on individual data at the insured level. Brockman and Wright (1992) use GLM software to statistically model the claims frequency and severity in premium pricing for motor liability insurance. Haberman and Renshaw (1996) present a comprehensive overview of the application of GLM for various actuarial problems such as: survival models, multiple condition models, claims distribution models, insurance premium pricing and claims reserves in non-life insurance.

There are numerous works and papers also on the application of machine learning methods on insurance data. Some of the authors have illustrated the superior predictive power of these methods over GLMs. Noll et al. (2020) is a case study where different machine learning models like regression tree, boosting, and neural network models are benchmarked against GLM to illustrate their superiority over traditional GLM methods. However, the interpretability of the results obtained from these models is a challenge that needs to be overcome to make them acceptable to the regulators and other stake holders. On the one hand the advanced machine learning methods are capable of overcoming the shortcomings of traditional GLMs, and on the other hand GLMs are more interpretable and presentable. A combination of machine learning methods and GLMs to build a hybrid model that has the advantage of both, could help reach a useful compromise. This chapter combines the excellent variable selection capabilities of a regression tree to selecting the GLM variables and interactions terms. The results are comparable to many advanced machine learning methods.

Denuit et al. (2019) presents the insurance modelling ideas in a modern context. Apart from statistical theory, their first book provides an insight into GLM construction. In their second book (Denuit et al., 2020, Chap 4), the authors explain how to build regression trees using a Poisson deviance as the loss function. Using the same dataset as used by Noll et al. (2020) in their case study, a regression tree and a GLM model are fitted. This first GLM and its results are used as a benchmark. This is followed by identifying the terminal nodes of the tree to which each record belongs and adding this information in the predictor space. Thereafter, a second GLM with better predictive power is built. This method and its results are explained in the subsequent sections.

2.3 Generalized Linear Model for Claim Frequency

The theory covered in this section is from (Denuit et al., 2019, Chap 4). Motor insurance pricing has two components, the number of claims (frequency) and the size of the claims (severity). For each insurance contract the number of claims submitted by the policyholder and the respective costs of these claims are recorded. The Poisson distribution is a popular choice for modelling claim frequency. Since GLMs assume that the response to follow a distribution from the exponential dispersion (ED) family, it has a score that is a linear combination of the available features, and a function that links the mean response with this score. A GLM consists of three components:

- The distribution of the response from the ED family,
- the linear score, and
- the link function.

Let, $Y_1, Y_2, \ldots, Y_n$ be responses measured on n individuals. Each of these response has characteristics that can be summarized into a $d+1$ dimensional vector $\mathbf{x_i} = \{1, x_{i1}, x_{i2}, ...x_{id}\}$. The initial 1 forces an intercept term in the score. The features are typically the measurable characteristics of the policyholders like, their age, the age of their vehicle, the vehicle brand, etc.. The response (also called target) variable Y is then explained by the vector of $\mathbf{x}$ values

$$\mu(x) = E[Y|X = \mathbf{x}]$$

is a regression function. In rate making, an actuary is interested in predicting the loss of a homogeneous group of individuals rather than the loss incurred by individual policyholders, hence reducing dimensionality. What is available are the random observations $(y_i, \mathbf{x_i})$. Instead, it is assumed that the Y_is are independent and follow a distribution from the ED family, given the information available in the $\mathbf{x_i}$s. The canonical parameter of this ED distribution is obtained as a function of the $\mathbf{x_i}$s.

$$\theta = \theta(\mathbf{x_i}),$$

while the dispersion parameter ϕ remains the same for all predictor variables. The quantity of interest is the policyholder's mean response μ_i given as

$$\mu_i = \mu(\mathbf{x_i}) = a'(\theta_i) = a'(\theta(\mathbf{x_i}))$$

2.3.1 Linear Score

The GLM score for response Y_i is given by

$$score_i = \mathbf{x_i}^T \beta = \beta_0 + \Sigma_{j=1}^{d} \beta_{ij} x_{ij}, \quad i = 1, 2, \ldots, n$$

where $\beta = (\beta_0, \beta_1, \ldots, \beta_d)$ is the vector of dimension $(d+1)$ of unknown regression coefficients to be estimated from the data.

The quantity β_0 is the same for all the scores. This is the intercept and it corresponds to the score when the x_{ij}s are equal to 0 for all $j = 1, 2, \ldots, p$. Then each β_j quantifies the impact on the score for an increase by one unit in the corresponding feature x_{ij}. The coefficient β_0 explains the first entry of 1 in the vector $\mathbf{x_i}$.

2.3.2 Design Matrix

Now is the time to introduce the design matrix X obtained by binding all the $\mathbf{x_i}$, row-wise. Precisely the i^{th} row of X is denoted as $\mathbf{x}^T$. Thus

$$X = \begin{pmatrix} \mathbf{x_1}^T \\ \mathbf{x_2}^T \\ \vdots \\ \mathbf{x_n}^T \end{pmatrix} = \begin{pmatrix} 1, & x_{11} & \cdots & x_{1p} \\ 1, & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1, & x_{n1} & \cdots & x_{np} \end{pmatrix}$$

The vector of the scores $s_1, s_2, \ldots, s_n$ for all the n responses is obtained as

$$\mathbf{s} = (s_1, s_2, \ldots, s_n)^T = X\beta$$

At this point it is in the best interest of the chapter to show the design matrix when there are two explanatory categorical attributes. The reason will become clear with further reading of the chapter.

Suppose there are two attributes, the gender (male and female) and the coverage extent (with three levels) of the policies. The three types of coverages are the compulsory third part liability abbreviated as TPL, limited damage and comprehensive. Both the features are categorical and the dataset, by counts, is specified in Table 2.1.

	TPL only	Limited Damage	Comprehensive
Males	1683	3403	626
	(10,000)	(30,000)	(5,000)
Females	873	2423	766
	(6,000)	(24,000)	(7,000)

Table 2.1: Observed number of claims together with corresponding exposures-to-risk (in policy years) appearing between brackets. Motor insurance, hypothetical data

The two features presented in Table 2.1 are categorical. They are now coded with binary feature variables. Let x_{i1} represent the gender variable. From the table we see that the total exposure for male policyholders is 45,000 against 37,000 for the females. Thus the male class is taken as the base class and x_i1 is defined as follows

$$x_{i1} = \begin{cases} 1, & \text{if the policyholder i is a woman,} \\ 0, & \text{otherwise,} \end{cases}$$

The attribute coverage extent has three levels and hence it can be represented using two dummy binary variables x_{i2} and x_{i3}. Table 2.1 shows that the maximum exposure

is for the category of limited damage and hence it should be the base class. Hence the definition of x_{i2} and x_{i3} is as follows

$$x_{i2} = \begin{cases} 1, & \text{if the policyholder i has TPL,} \\ 0, & \text{otherwise,} \end{cases}$$

,and

$$x_{i3} = \begin{cases} 1, & \text{if the policyholder i has comprehensive,} \\ 0, & \text{otherwise,} \end{cases}$$

Therefore the base class is a male policyholder with limited damage coverage. The design matrix for this dataset in Table 2.1 is as follows

$$X = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

The corresponding vector of observations is

$$Y = \begin{pmatrix} 1,683 \\ 3,403 \\ 626 \\ 873 \\ 2,423 \\ 766 \end{pmatrix}$$

A similar structure will be used for the design matrix by converting even the continuous variables into categorical ones, depending on the number of splits made in the tree. This will become clearer with further explanation of the methodology.

2.3.3 Link Function and Log Link

Multiple linear regression equates the expected response to the linear score. Instead of that, GLM maps the mean response to a one-to-one, continuous and differentiable function of the linear score. This function is called the link function. Thus for the response Y_i, for $i = 1, 2, \ldots, n$, if the link function is denoted by g, then

$$g(\mu_i) = score_i,$$

where g is monotone and invertible,

$$g(\mu_i) = score_i \iff \mu_i = g^{-1}(score_i).$$

This is a conditional model where the Y_i's follow a distribution from the ED family. When the distribution of the Y_i's representing the claim counts is Poisson, it means that $Y_i \sim Poi(\exp\{(\beta^{\mathbf{T}}\mathbf{x_i})\}$ given $\mathbf{x_i}$, conditionally. The mean of the distribution is $\exp\{(\beta^{\mathbf{T}}\mathbf{x_i})\}$. This is the log-link function

$$\ln \mu_i = score_i \iff \mu_i = \exp\{score_i\}.$$

The log-link function has a multiplicative structure.

$$\mu_i = \exp\left\{\left(\beta_0 + \Sigma_{j=1}^{d}\beta_j x_{ij}\right)\right\} = \exp\{\beta_0\}\left(\Pi_{j=1}^{d}\exp\{\beta_j x_{ij}\}\right).$$

When the explanatory variables are categorical each of the policy holders are represented by d components of x_{ij} which are either 0 or 1. When all of the x_{ij} are 0's then

$$\begin{aligned} \exp\{\beta_0\} &= \text{the expected response for the base class} \\ \exp\{\beta_j\} &= \text{relative effect of the feature j} \end{aligned}$$

Going back to the data in Table 2.1 the expected claim frequency for a male policyholder with limited damage coverage, is $\exp\{(\beta_0)\}$, that is the base class. For female policyholders with similar coverage the expected claim frequency is

$$\exp\{\beta_0\}\exp\{\beta_1\}$$

If x_{ij} has an increasing effect on the base class then $\beta_j > 0 \implies \exp\{\beta_j\} > 1$, and vice versa.

2.3.4 Interactions

Interactions arise when a particular value of a risk factor is reliant on the other. As an example it is often observed in motor insurance claims that on an average young, female drivers submit lesser number of claims than young, male drivers. This difference often disappears at older ages. Hence it is natural to assume that the effect of age is dependent on gender, but GLM models do not account for interactions automatically. For real data sets with large number of predictor variables it is very difficult to identify interactions. The one mentioned between age and gender is a common feature that has been observed over years. It is not possible to identify an interaction in a new dataset automatically. Tree based methods are better at identifying interactions automatically, a feature that is used in this chapter. Before discussing trees, we review first how to deal with interactions in GLMs.

Interactions are included in GLMs by adding to the linear term a product of independent variables. Consider Taylor's expansion of the mean function on the score scale:

$$g(E[Y|\mathbf{X} = \mathbf{x}]) \approx \beta_0 + \Sigma_{j=1}^{d}\beta_j x_{ij}.$$

Since the $\hat{\beta}$ estimates are obtained by maximizing the likelihood function it must be noted here that

$$\frac{\partial^2 g(E[Y|\mathbf{X} = \mathbf{x}])}{\partial x_j \partial x_k} = 0.$$

Thus

$$\beta_0 = g(E[Y|\mathbf{X} = \mathbf{0}])$$

and

$$\beta_j = \frac{\partial g(E[Y|\mathbf{X} = \mathbf{x}])}{\partial x_j}\Big|_{\mathbf{x}=0}.$$

The second derivative is always zero when the score is only a linear combination of features. However, if the score contains terms that are products of features x_j and x_k, for $j \neq k$, then the second derivatives will no longer be 0. A more accurate approximation of the regression function with interaction terms is given by

$$g(E[Y|\mathbf{X} = \mathbf{x}]) \approx \beta_0 + \Sigma_{j=1}^{d} \beta_j x_{ij} + \Sigma_{j=1}^{d} \Sigma_{k \geq j+1} \gamma_{jk} x_{ij} x_{ik},$$

where

$$\gamma_{jk} = \frac{\partial^2 g(E[Y|\mathbf{X} = \mathbf{x}])}{\partial x_j \partial x_k}\Big|_{\mathbf{x}=0}.$$

It is to be noted that this two–way interaction term can exist only if the main terms are also included in the model. For example, if a dataset has two continuous features then a score with the main and interaction terms is

$$score_i = \beta_0 + \beta_1 xi1 + \beta_2 x_{i2} + \beta_3 x_{i1} x_{i2}.$$

Here the third coefficient is for the interaction of the two features x_{i1} and x_{i2}. So every unit increase of the feature x_{i1}, increases the score by $(\beta_1 + \beta_3 x_{i2})$. Similarly, for every unit increase in the feature x_{i2}, the score is increased by $(\beta_2 + \beta_3 x_{i1})$. If we tweak the example by making one of the variables x_{i2} (say) a binary, categorical variable then the score is given by

$$score_i = \begin{cases} \beta_0 + \beta_1 x_{i1}, & if\, x_{i2} = 0, \\ \beta_0 + \beta_2 + (\beta_1 + \beta_3) x_{i1}, & if\, x_{i2} = 1. \end{cases}$$

Recall the example introduced with the data in Table 2.1. This example is a simple illustration of what can be done by converting all the variables in binary categorical variables. Suppose that a two–way interaction term is introduced between the two existing features, gender and the coverage extent of the policyholders:

$$\begin{aligned} score_i &= \beta_0 + \beta_1 xi1 + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i1} x_{i2} + \beta_5 x_{i1} x_{i3} \\ &= \begin{cases} \beta_0, & \text{if policyholder is male, coverage for limited damage,} \\ \beta_0 + \beta_2, & \text{if policyholder is male, TPL coverage,} \\ \beta_0 + \beta_3, & \text{if policyholder is male, comprehensive coverage,} \\ \beta_0 + \beta_1, & \text{if policyholder is female, coverage for limited damage,} \\ \beta_0 + \beta_1 + \beta_2 + \beta_4, & \text{if policyholder is female, TPL coverage,} \\ \beta_0 + \beta_1 + \beta_3 + \beta_5, & \text{if policyholder is female, comprehensive coverage.} \end{cases} \end{aligned}$$

The next subsection discusses claim exposure in the context of claim frequency and the log–link function.

2.3.5 Canonical Links and Poisson Distribution

Recall in Equation (1) the real-valued location parameter θ, the canonical parameter bears the relation $\mu_i = a'(\theta)$, where a is a continuous and monotone function of θ and $\mu_i = E[Y_i|\mathbf{X} = \mathbf{x_i}]$. In the previous section we have seen that $g(E[Y_i|\mathbf{X} = \mathbf{x}]) = score_i$. A link function in which the score is same as the canonical parameter, is said to be a canonical link function. Under the assumption that the Y_i's conditionally follow a Poisson distribution with mean μ_i,

$$f_Y(y_i) = \frac{\exp(-\mu_i)\mu_i^{y_i}}{y_i!} = \exp\big(y_i \ln \mu_i - \mu_i\big)\exp\big(-\ln(y_i!)\big), \tag{9}$$

with

$$\begin{aligned} \theta_i &= \ln \mu_i, \\ a(\theta_i) &= \mu_i = \exp(\theta_i). \end{aligned}$$

Thus, $a'(\theta_i) = \exp(\theta_i) = \mu_i$, $\phi = 1$, and $\nu_i = 1$.

Equation (9) shows that the Poisson distribution follows the ED form in Equation (1). Moreover, for the log–link, the Poisson GLM is defined as

$$\ln E(Y_i|X = x_i) = score_i = \theta_i$$

Thus the log–link function is a canonical link function.

2.3.6 Likelihood Equation and Fisher's Information with Canonical Link

Having specified the GLM specified in terms of its link function and the ED family of distributions, the β's are estimated by the method of maximum likelihood. This approach seeks to fit the GLM that has the highest probability to produce the actual data. Although the dispersion parameter ϕ could also be estimated by maximum likelihood, a moment estimator is generally used instead, after the β's are estimated. Thus, the likelihood function will be studied in the context of just the β estimation.

The likelihood function which is essentially the joint probability of the observations in the dataset is transformed by the logarithm so that the product is replaced with the sum of the probability densities and that eases the mathematical operations: the $\beta_0, \beta_1, \ldots, \beta_d$ that maximize this sum are estimated thereafter. For the ED family the log–likelihood of the $Y_1, Y_2, \ldots, Y_n$ is given by

$$\ln L(\beta) = \Sigma_{i=1}^n \ln f_{\theta_i}(y_i) = \Sigma_{i=1}^n \left(\frac{y_i\theta_i - a(\theta_i)}{\frac{\phi}{\nu_i}} \right) + c(y_i, \frac{\phi}{\nu_i}),$$

where θ_i, in general, is a function of the linear score s_i. The maximum likelihood estimate for β is obtained by differentiating $\ln L$ with respect to β_j and equating it to 0:

$$\begin{aligned}\frac{\partial \ln L(\beta)}{\partial \beta_j} &= \frac{\partial}{\partial \beta_j}\left(\Sigma_{i=1}^{n}\frac{y_i\theta_i - a(\theta_i)}{\frac{\phi}{\nu_i}} + c(y_i, \frac{\phi}{\nu_i})\right) \\ &= \frac{\partial}{\partial \theta_i}\left(\Sigma_{i=1}^{n}\frac{y_i\theta_i - a(\theta_i)}{\frac{\phi}{\nu_i}} + c(y_i, \frac{\phi}{\nu_i})\right)\frac{\partial \theta_i}{\partial \mu_i}\frac{\partial \mu_i}{\partial \beta_j} \\ &= \Sigma_{i=1}^{n}\left(\frac{\nu_i(y_i - a'(\theta_i))}{\phi}\right)\frac{\partial \theta_i}{\partial \mu_i}\frac{\partial \mu_i}{\partial \beta_j},\end{aligned}$$

where,

$$\frac{\partial \mu_i}{\partial \theta_i} = \frac{\partial a'(\theta_i)}{\partial \theta_i} = a''(\theta_i) = \frac{\nu_i Var(Y_i)}{\phi},$$

as $\mu_i = a'(\theta_i)$, and

$$\frac{\partial \mu_i}{\partial \beta_j} = \frac{\partial \mu_i}{\partial s_i}\frac{\partial s_i}{\partial \beta_j} = \frac{\partial \mu_i}{\partial s_i}x_{ij}.$$

Thus the equation to solve is

$$\Sigma_{i=1}^{n}\left(\frac{(y_i - \mu_i)}{Var(Y_i)}\right)\frac{\partial \mu_i}{\partial s_i}x_{ij} = 0. \tag{10}$$

With canonical the link function, the canonical parameter is equal to the score: $\theta_i = s_i$. Thus

$$\frac{\partial \mu_i}{\partial \theta_i} = \frac{\partial \mu_i}{\partial s_i} = \frac{\partial a'(\theta_i)}{\partial \theta_i} = a''(\theta_i) = \frac{\nu_i Var(Y_i)}{\phi}.$$

This reduces Equation (10) to

$$\Sigma_{i=1}^{n}\left(\frac{(y_i - \mu_i)}{\frac{\phi}{\nu_i}}\right)x_{ij} = 0.$$

Taking $\nu_i = 1$, for all i, the solution ensures that the fitted mean

$$\hat{\mu_i} = \mu_i(\hat{\beta}) = g^{-1}(\mathbf{x_i}^T\hat{\beta})$$

fulfill the condition

$$\Sigma_{i=1}^{n}(y_i - \hat{\mu_i})x_{ij} = 0 \iff \Sigma_{i=1}^{n}y_i x_{ij} = \Sigma_{i=1}^{n}\hat{\mu_i}x_{ij}, \quad j = 0, 1, 2 \ldots d.$$

Without the generalization of setting the weights to 1, if W is a diagonal matrix with the i^{th} diagonal element = ν_i, then the above result can be written as

$$X^T W(\mathbf{y} - \mu(\hat{\beta})) = 0,$$

where

$$W = \begin{pmatrix} \nu_1 & 0 & \cdots & 0 \\ 0 & \nu_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \nu_n \end{pmatrix}$$

Now is the time to see Fisher's information for the likelihood estimates of GLMs with canonical link functions.

$$\ln L(\beta) = \frac{1}{\phi}\Sigma_{i=1}^{n} \nu_i (y_i s_i - a(s_i)) + \text{constant w.r.t. } \beta$$

So

$$\begin{aligned} \frac{\partial^2 \ln L(\beta)}{\partial \beta_j \partial \beta_k} &= -\Sigma_{i=1}^{n} \frac{\nu_i x_{ij}}{\phi} \frac{\partial \mu_i}{\partial \beta_k} \\ &= -\Sigma_{i=1}^{n} \frac{\nu_i x_{ij}}{\phi} \frac{\partial \mu_i}{\partial s_i} x_{ik} \\ &= -\Sigma_{i=1}^{n} \frac{\nu_i x_{ij} x_{i_k}}{\phi} a''(\theta_i). \end{aligned}$$

That makes the $(j,k)^{th}$ entry for the Fisher's information matrix equal to:

$$H_{jk}(\beta) = -\Sigma_{i=1}^{n} \frac{x_{ij} x_{i_k}}{Var(Y_i)} (a''(\theta_i))^2,$$

which is independent of the responses. Hence

$$\frac{\partial^2 \ln L(\beta)}{\partial \beta_j \partial \beta_k} = E\left[\frac{\partial^2 \ln L(\beta)}{\partial \beta_j \partial \beta_k}\right],$$

showing that for GLMs with canonical link functions, the expected information matrices coincide with the observed ones.

2.3.7 Individual and Grouped Data

This section is about grouping data into mutually exclusive sub spaces. The example is taken from (Denuit et al., 2019, Chap 4). Table 2.2 is from a portfolio that has been grouped into 4 classes, according to the two binary features encoding the information available about each policyholder (gender and annual distance traveled being more or less than 20,000 km per year).

The response Y_i is the number of claims reported by policyholder i. Here $Y_1, \ldots, Y_n$ are assumed to be independent and Poisson distributed, with means $\mu_1, \ldots, \mu_n$, respectively. The expected number of claims for policy i in the portfolio is expressed as

$$\mu_i = e_i \exp\big(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}\big),$$

	Annual distance traveled (< 20,000 km)	Annual distance traveled (≥ 20,000 km)
Male	143 (2,000)	1,967(18,000)
Female	278 (6,000)	354 (4,000)

Table 2.2: Number of claims and corresponding risk exposures (in policy-years, within parentheses) for an hypothetical motor insurance portfolio observed during one calendar year

where e_i is the risk exposure for policyholder i, x_{i1} records the annual mileage, and x_{i2} records gender:

$$x_{i1} = \begin{cases} 1 & \text{if policyholder i drives less than 20,000 km a year,} \\ 0 & \text{otherwise,} \end{cases}$$

and

$$x_{i2} = \begin{cases} 1 & \text{if policyholder i is female,} \\ 0 & \text{otherwise.} \end{cases}$$

From the variable definition and the data in Table 2.2 we see that the reference class, $x_{i1} = 0$ and $x_{i2} = 0$, is the most populous (male drivers who drive more that 20,000 km annually).

Table 2.2 summarizes the experience of the whole portfolio by grouping all the data into only four risk classes, each grouping drivers with similar attributes (that is, aggregating all policies that are identical with respect to the two features x_{i1} and x_{i2}). The corresponding exposures are placed with brackets in each cell. This section shows that such a grouping is allowed when working with Poisson distributed responses (and more generally, with any ED distribution), without changing parameter the estimation.

In the frequency model, y_i denotes the observed number of claims for policyholder i with an exposure of e_i. The corresponding likelihood function is

$$\mathscr{L}_{ind}(\beta_0, \beta_1, \beta_2) = \Pi_{i=1}^{n} \exp\big(-\mu_i\big) \frac{\mu_i^{y_i}}{y_i!}.$$

The subscript *ind* indicates that it is the likelihood of individual observations. This product can be subdivided into 4 categories matching each cell of Table 2.2 as follows:

$$\begin{aligned}
\mathscr{L}_{ind}(\beta_0,\beta_1,\beta_2) &= \Pi^n_{i|x_{i1}=x_{i2}=0}\exp(-\mu_i)\frac{\mu_i^{y_i}}{y_i!} \times \Pi^n_{i|x_{i1}=0,x_{i2}=1}\exp(-\mu_i)\frac{\mu_i^{y_i}}{y_i!} \\
&\quad \times \Pi^n_{i|x_{i1}=1,x_{i2}=0}\exp(-\mu_i)\frac{\mu_i^{y_i}}{y_i!} \times \Pi^n_{i|x_{i1}=x_{i2}=1}\exp(-\mu_i)\frac{\mu_i^{y_i}}{y_i!} \\
&\propto \exp\left\{-\exp\left(\beta_0\Sigma_{i|x_{i1}=x_{i2}=0}e_i\right)\right\}\left[\exp\left(\beta_0\right)\right]^{\Sigma_{i|x_{i1}=x_{i2}=0}k_i} \\
&\quad \times \exp\left\{-\exp\left(\beta_0+\beta_2\right)\Sigma_{i|x_{i1}=0,x_{i2}=1}e_i)\right\}\left[\exp\left(\beta_0+\beta_2\right)\right]^{\Sigma_{i|x_{i1}=0x_{i2}=1}k_i} \\
&\quad \times \exp\left\{-\exp\left(\beta_0+\beta_1\right)\Sigma_{i|x_{i1}=1,x_{i2}=0}e_i)\right\}\left[\exp\left(\beta_0+\beta_1\right)\right]^{\Sigma_{i|x_{i1}=1x_{i2}=0}k_i} \\
&\quad \times \exp\left\{-\exp\left(\beta_0+\beta_1+\beta_2\right)\Sigma_{i|x_{i1}=x_{i2}=1}e_i)\right\}\left[\exp\left(\beta_0+\beta_1+\beta_2\right)\right]^{\Sigma_{i|x_{i1}=x_{i2}=1}k_i},
\end{aligned}$$

where k_i are the counts for each class. The $\propto$ indicates that the two expressions are proportional, upto a constant factor that is independent of the β_j's that is to be estimated.

This shows that the individual likelihood is proportional to the likelihood based on the data aggregated into 4 risk classes as it involves the total exposure and the total claim numbers for each risk class as in Table 2.2.

Now, shifting to aggregate counts for the 4 classes we have:

$$\begin{aligned}
Y_{00} &= \Sigma_{i|x_{i1}=x_{i2}=0}Y_i, \\
Y_{01} &= \Sigma_{i|x_{i1}=0,x_{i2}=1}Y_i, \\
Y_{10} &= \Sigma_{i|x_{i1}=1,x_{i2}=0}Y_i, \\
Y_{11} &= \Sigma_{i|x_{i1}=x_{i2}=1}Y_i.
\end{aligned}$$

These produce a likelihood function that is proportional to the one above, so the maximum likelihood procedure yields exactly the same estimates. In fact, the Poisson distribution properties ensure that the four aggregate variables $Y_{00}, Y_{01}, Y_{10}, Y_{11}$ are Poisson distributed also:

$$\begin{aligned}
Y_{00} &\sim Poi(\lambda_{00}), \text{ with } \lambda_{00} = e_{00}\exp\left(\beta_0\right), \text{ where } e_{00} = \Sigma_{i|x_{i1}=x_{i2}=0}e_i, \\
Y_{01} &\sim Poi(\lambda_{01}), \text{ with } \lambda_{01} = e_{01}\exp\left(\beta_0+\beta_2\right), \text{ where } e_{01} = \Sigma_{i|x_{i1}=0,x_{i2}=1}e_i, \\
Y_{10} &\sim Poi(\lambda_{10}), \text{ with } \lambda_{10} = e_{10}\exp\left(\beta_0+\beta_1\right), \text{ where } e_{10} = \Sigma_{i|x_{i1}=1,x_{i2}=0}e_i, \\
Y_{11} &\sim Poi(\lambda_{11}), \text{ with } \lambda_{11} = e_{11}\exp\left(\beta_0+\beta_1+\beta_2\right), \text{ where } e_{11} = \Sigma_{i|x_{i1}=x_{i2}=1}e_i.
\end{aligned}$$

Thus the likelihood function $\mathscr{L}_{group}(\beta_0,\beta_1,\beta_2)$ associated with this grouped data is the

product of 4 Poisson probabilities:

$$\begin{aligned}\mathscr{L}_{group}(\beta_0,\beta_1,\beta_2) = \exp\{-e_{00}\exp(\beta_0)\}\frac{\left[e_{00}\exp(\beta_0)\right]^{y_{00}}}{y_{00}!}\\ \exp\{-e_{01}\exp(\beta_0+\beta_2)\}\frac{\left[e_{01}\exp(\beta_0+\beta_2)\right]^{y_{01}}}{y_{01}!}\\ \exp\{-e_{10}\exp(\beta_0+\beta_1)\}\frac{\left[e_{10}\exp(\beta_0+\beta_1)\right]^{y_{10}}}{y_{10}!}\\ \exp\{-e_{11}\exp(\beta_0+\beta_1+\beta_2)\}\frac{\left[e_{11}\exp(\beta_0+\beta_1+\beta_2)\right]^{y_{11}}}{y_{11}!}.\end{aligned}$$

It is clear that

$$\mathscr{L}_{ind}(\beta_0,\beta_1,\beta_2) \propto \mathscr{L}_{group}(\beta_0,\beta_1,\beta_2),$$

so that one can work with aggregated data and still get the same estimation of the parameters as from individual observations. Aggregating a whole portfolio into homogeneous classes does not result in any loss of information. However, this property is only valid because of the sufficiency properties under the ED assumption for the responses.

Real insurance datasets are far more complicated than the one in this illustrative example, with the number of attribute variables in the hundreds. Getting the proper attributes that account for best homogeneous aggregation of the data can be tedious and slow. This is where the tree method comes to help. In the next section the regression tree algorithm is discussed with respect to the Poisson distribution.

2.4 Regression Trees for Claim Frequency

Classification and regression trees are at the core of various advanced machine learning algorithms like bagging, random forests, and XGBoost. While Chapter 1 deals with classification trees this chapter focuses on regression trees. The main concepts and ideas of a regression tree can be found in Breiman (2017). While Denuit et al. (2020) presents the regression tree concepts with respect to insurance data.

As mentioned in (Denuit et al., 2020, Chap 3) a regression tree recursively partitions the feature space X into smaller, disjoint sub spaces $\{X_t\}_{t\in T}$, where T is a group of indices. On each subset X_t, the predictions $\hat{c}_t$ of the response is constant. The resulting prediction of $\hat{\mu}(x)$ is given as

$$\hat{\mu}(x) = \Sigma_{t\in T}\hat{c}_t\mathbf{1}[x\in X_t].$$

The tree method is a powerful alternative to the parametric techniques like regression, primarily because of the following advantages:

(i) The recursive partitioning of the predictor space is not based on any assumption of the existence of a parametric relationship with the response. It rather does a better job in identifying the non–linear and the interactive effect of the features on the response.

(ii) The tree building method does not require any knowledge on the hierarchical importance of the features in terms of their predicting capabilities on the response. In fact, the method is built to look for the feature that best splits the response in terms of minimizing the loss.

(iii) The tree method is highly interpretable. A tree can be easily visualized and interpreted by a graphical model.

The CART (Classification and Regression Tree) algorithm (see Breiman, 2017) is formed of three steps, briefly discussed here:

1. **Growing the Tree**: First a large tree is grown. The algorithm starts from a first node t_0, also called the root node, that includes the entire data set with the feature space X. In the first step this feature space is split into two disjoint sub spaces X_{t_1} and X_{t_2} such that $X = X_{t_1} U X_{t_2}$. These X_{t_1} and X_{t_2} correspond to the nodes t_1 and t_2, respectively. The binary splits are usually of the form $X_{ij} <= k$ versus $X_{ij} > k$ where k is a constant for continuous or ordinal variables X_{ij}. If on the other hand, X_{ij} is a categorical feature then the split is $X_{ij} \in S$, where S is a subset of the values that X_{ij} may take. The criterion behind is the reduction of the heterogeneity of the node t_0 so that the nodes t_1 are t_2 are more homogeneous, in terms of the X_{ij} they contain, as measured by the loss function that is used to build the trees. In the ED family setting a natural candidate for the loss is the deviance. It is denoted by $D_{X_t}(\hat{c}_t)$ and the Poisson Deviance is given as

$$D_y(\hat{\mu}) = 2\Sigma_{i=1}^{n}\left(y_i \ln\frac{y_i}{\mu_i} - (y_i - \hat{\mu}_i)\right)$$

, where $ylny = 0$ if $y = 0$ Here we build a Poisson Regression tree and hence consider the Poisson Deviance as the loss function. The optimal split s_t is the one that solves

$$\min_{s\in S_t}\left\{D_{X_{t_L^{(s)}}}\left(\hat{c}_{t_L^{(s)}}\right) + D_{X_{t_R^{(s)}}}\left(\hat{c}_{t_R^{(s)}}\right)\right\},$$

where $t_L^{(s)}$ and $t_R^{(s)}$ are the left and right children of the parent node t, resulting from the split s and $\hat{c}_{t_L^{(s)}}$ and $\hat{c}_{t_R^{(s)}}$ are the corresponding predictions. The optimal split s_t then leads to the children nodes $t_L^{(s_t)}$ and $t_R^{(s_t)}$, also denoted as t_L and t_R. It is to be noted that minimizing the sum of the two deviances of the two children nodes is equivalent to maximizing the overall decrease of the deviance at node t, namely

$$\max_{s\in S_t}\left\{D_{X_t}(\hat{c}_t) - \left(D_{X_{t_L^{(s)}}}\left(\hat{c}_{t_L^{(s)}}\right) + D_{X_{t_R^{(s)}}}\left(\hat{c}_{t_R^{(s)}}\right)\right)\right\}.$$

The tree starts from the whole feature space X and is grown by iteratively dividing the subsets of X into smaller subsets. Every node t is split using the optimal split s_t that locally maximizes the decrease in deviance. The strategy is greedy so that splits could be looked into deeper down the tree. Hence the tree keeps splitting

until some pre–specified criteria, like the minimum number of observations in the terminal nodes and/or the pre–specified depth of the tree is reached. Let the tree thus generated be named T.

2. **Tree Pruning:** This step trims the large tree T produced in the previous step to generate a sequence of nested subtrees. The creation of subtrees relies on a cost–complexity pruning algorithm. If $|T|$ denotes the number of terminal nodes of a tree T, and τ_T the set of indices for the terminal nodes of T, then the cost complexity measure of a the tree T is given by

$$R_\alpha(T) = D\left((\hat{c}_t)_{t\in\tau_{(T)}}\right) + \alpha|T|,$$

where the parameter α is a positive real number. The number of terminal nodes $|T|$ is called the complexity of the tree T. Thus the cost–complexity measure $R_\alpha(T)$ is a combination of the deviance $D((\hat{c}_t)_{t\in\tau_{(T)}})$ and a penalty for the complexity of the tree $\alpha|T|$.

When we increase by one the number of terminal nodes of a tree T, by splitting one of its terminal nodes t into two children nodes t_L and t_R, then the deviance of the resulting tree $T^{'}$ is smaller than the deviance of the original tree T, that is

$$D((\hat{c}_t)_{t\in\tau_{(T^{'})}}) \leq D((\hat{c}_t)_{t\in\tau_{(T)}}).$$

Since the deviance always favours the more complex tree $T^{'}$ over T, the introduction of a penalty for the tree complexity may make the original tree T preferable over the more complex tree $T^{'}$. The cost–complexity measure of $T^{'}$ can be written as

$$\begin{aligned} R_\alpha(T^{'}) &= D((\hat{c}_t)_{t\in\tau_{(T^{'})}}) + \alpha|T^{'}| \\ &= D((\hat{c}_t)_{t\in\tau_{(T^{'})}}) + \alpha(|T|+1) \\ &= R_\alpha(T) + D((\hat{c}_t)_{t\in\tau_{(T^{'})}}) - D((\hat{c}_t)_{t\in\tau_{(T)}}) + \alpha. \end{aligned}$$

Thus $R_\alpha(T^{'}) \geq R_\alpha(T)$ if and only if

$$\alpha \geq D((\hat{c}_t)_{t\in\tau_{(T)}}) - D((\hat{c}_t)_{t\in\tau_{(T^{'})}}).$$

That is, $R_\alpha(T^{'}) \geq R_\alpha(T)$ if and only if the deviance reduction that is obtained by producing the tree $T^{'}$ is smaller than the increase in the penalty for having one more terminal node. When the value of α satisfies this condition, the more complex tree $T^{'}$ is preferred.

If $\alpha = 0$, the cost-complexity measure is same as the deviance. The largest tree would minimize $R_\alpha(T)$. As α increases the penalty for growing a large tree increases thus resulting in fewer terminal nodes. At $\alpha = 0$ let T_{init} be the initial tree. Suppose we find any pair of terminal nodes with a common parent node t such

that the branch $T_{init}^{(t)}$ can be pruned without increasing the cost-complexity measure. These nodes are pruned and this is continued until such a pair can no longer be found in order to obtain a subtree of T_{init} with the same cost-complexity of T_{init}. Define $\alpha_0 = 0$ and the resulting subtree of T_{init} as T_{α_0}. When α increases, it becomes optimal to prune the branch $T_{\alpha_0}^{(t)}$ rooted at node t of T_{α_0} such that the smaller tree $T_{\alpha_0} - T_{\alpha_0}^{(t)}$ becomes better that T_{α_0}. When α is high enough to have

$$R_\alpha(T_{\alpha_0}) \geq R_\alpha(T_{\alpha_0} - T_{\alpha_0}^{(t)})$$

The deviance of T_{α_0} can be written as

$$\begin{aligned} D\left((\hat{c}_s)_{s\in\tau_{(T_{\alpha_0})}}\right) &= \Sigma_{s\in\tau_{(T_{\alpha_0})}} D_{X_s}(\hat{c}_s) \\ &= \Sigma_{s\in\tau_{(T_{\alpha_0}-T_{\alpha_0}^{(t)})}} D_{X_s}(\hat{c}_s) + \Sigma_{s\in\tau_{(T_{\alpha_0}^{(t)})}} D_{X_s}(\hat{c}_s) - D\left((\hat{c})_{s\in t]}\right) \\ &= D\left((\hat{c})_{s\in\tau_{(T_{\alpha_0}-T_{\alpha_0}^{(t)})}}\right) + D\left((\hat{c})_{s\in\tau_{(T_{\alpha_0}^{(t)})}}\right) + D\left((\hat{c})_{s\in t}\right). \end{aligned}$$

Furthermore,

$$|T_{\alpha_0}| = |T_{\alpha_0} - T_{\alpha_0}^{(t)}| + |T_{\alpha_0}^{(t)}| - 1.$$

Thus the cost–complexity measure $R_\alpha(T_{\alpha_0})$ can be rewritten as

$$R_\alpha(T_{\alpha_0}) = R_\alpha(T_{\alpha_0} - T_{\alpha_0}^{(t)}) + D\left((\hat{c})_{s\in\tau_{(T_{\alpha_0}^{(t)})}}\right) + \alpha|T_{\alpha_0}^{(t)}| - D\left((\hat{c})_{s\in t}\right) - \alpha. \tag{11}$$

Therefore, $R_\alpha(T_{\alpha_0}) \geq R_\alpha\left(T_{\alpha_0} - T_{\alpha_0}^{(t)}\right)$ if and only if

$$D\left((\hat{c})_{s\in\tau_{(T_{\alpha_0}^{(t)})}}\right) + \alpha|R_\alpha(T_{\alpha_0})| \geq D\left((\hat{c})_{s\in t}\right) - \alpha. \tag{12}$$

Hence, it becomes optimal to cut the branch $T_{\alpha_0}^{(t)}$ once the cost–complexity measure $R_\alpha(T_{\alpha_0}^{(t)})$ becomes higher than the cost complexity measure $R_\alpha(t)$ of the node rooted at t. This requires α to satisfy the condition

$$\alpha \geq \frac{D\left((\hat{c})_{s\in t}\right) - D\left((\hat{c})_{s\in\tau_{(T_{\alpha_0}^{(t)})}}\right)}{|T_{\alpha_0}^{(t)}| - 1}. \tag{13}$$

The right hand side of (13) is henceforth denoted by $\alpha_1^{(t)}$. Let the set of non-terminal nodes of a tree T be denoted by $\tilde{\tau}_{(T)}$. For each non-terminal node t of T_{α_0}, $\alpha_1^{(t)}$ is calculated.

$$\alpha_1 = min_{t\in\tilde{\tau}_{(T_{\alpha_0})}} \alpha_1^{(t)}.$$

Cutting branches of T_{α_0} is not optimal as long as $\alpha < \alpha_1$. Once the parameter α reaches the value α_1, it becomes preferable to prune T_{α_0} at its weakest links. The

resulting tree thus obtained is T_{α_1}. The same process is now repeated on T_{α_1}. For all non terminal nodes of T_{α_1} it is preferable to cut the branch $T^{(t)}_{\alpha_0}$ when

$$\alpha \geq \frac{D\left((\hat{c})_{s\in t}\right) - D\left((\hat{c})_{s\in \tau_{(T^{(t)}_{\alpha_1})}}\right)}{|T^{(t)}_{\alpha_1}| - 1}. \tag{14}$$

The right hand side of (14) is denoted by $\alpha^{(t)}_2$ and define

$$\alpha_2 = min_{t\in \tilde{\tau}_{(T_{\alpha_1})}} \alpha^{(t)}_2.$$

The non-terminal nodes t of T_{α_1} for which $\alpha^{(t)}_2 = \alpha_2$ are the weakest links of T_{α_1}. It is now preferable to cut these nodes once α reaches the value α_2 to get the next tree T_{α_2}. Thus the pruning process yields a sequence of subtrees $T_{\alpha_0}, T_{\alpha_1}, T_{\alpha_2}, \ldots, T_{\alpha_k}$.

3. **Selecting the best pruned tree:** Once the sequence of trees $T_{\alpha_0}, T_{\alpha_1}, T_{\alpha_2}, \ldots, T_{\alpha_k} = \{t_0\}$ has been built the next step is to select the best pruned tree. The selection is usually achieved by cross–validation. In K-fold cross–validation, the training set D is partitioned into K subsets $D_1, D_2, \ldots, D_K$ of roughly equal size I_j. The j^{th} for $j = 1, 2, \ldots, K$, training set contains all observations in $D \backslash D_j$, $j = 1, 2, \ldots, K$. For each training set the sequence of smallest minimizing subtrees $T^{(j)}_{\alpha_0}, T^{(j)}_{\alpha_1}, T^{(j)}_{\alpha_2}, \ldots, T^{(j)}_{\alpha_k}$ are built starting from a sufficiently large tree $T^{(j)}_{init}$. Since the observations of the set D_j are not used to build the trees, this set plays the role of a validation set for these trees. If $\hat{\mu}_{T^{(j)}(\alpha_k)}$ denotes the model produced by the tree $T^{(j)}(\alpha_k)$ then

$$\hat{\mu}_{T^{(j)}(\alpha_k)}(x) = \Sigma_{t\in \tau_{(T^{(j)}(\alpha_k))}} \hat{c}_t I[x \in X_t].$$

Hence an estimate of the generalization error of $\hat{\mu}_{T^{(j)}(\alpha_k)}$ on D_j is given by

$$\hat{Err}^{val}(\hat{\mu}_{T^{(j)}(\alpha_k)}) = \frac{1}{|I_j|} \Sigma_{i\in I_j} L\left(y_i, \hat{\mu}_{T^{(j)}(\alpha_k)}(x_i)\right).$$

where L is the loss function measured in terms of the deviance $D_y(\hat{\mu}_{T_{(\alpha_k)}}(x))$ So, the K-fold cross–validation estimate of the generalization error for the regularization parameter α_k is given by

$$\begin{aligned} \hat{Err}^{CV}(\alpha_k) &= \Sigma^K_{j=1} \frac{|I_j|}{|I|} \hat{Err}^{val}(\hat{\mu}_{T^{(j)}(\alpha_k)}) \\ &= \frac{1}{|I|} \Sigma^K_{j=1} \Sigma_{i\in I_j} L\left(y_i, \hat{\mu}_{T^{(j)}(\alpha_k)}(x_i)\right). \end{aligned}$$

The right sized tree T_{prune} is selected as the tree $T_{\alpha_{k^*}}$ of the sequence $T_{\alpha_0}, T_{\alpha_1}, T_{\alpha_2}, \ldots, T_{\alpha_k}$ such that

$$\hat{Err}^{CV}(\alpha_{k^*}) = min_{k\in\{0,1,\ldots K\}} \hat{Err}^{CV}(\alpha_k).$$

2.5 Improvement in GLM Accuracy Using a Regression Tree as Preprocessor

Generalized Linear Models are traditionally a popular method used by actuaries to model the claim counts (frequency), size of the claims (severity), and the loss cost.

Most of the insurance portfolios are typically imbalanced, with a large proportion of zero claims and can be best modelled by a Poisson distribution when it comes to claim counts. However, the traditional GLM method, though a favourite for its high interpretability, usually falls short in handling the skewness exhibited by the data. Also, it does not have built-in processes for the selection of the important features, their interactions, nor to correct for non-linearities.

The binary tree method as explained briefly in the previous section is a powerful machine learning algorithm for decision making across many disciplines. The tree method is better equipped in capturing variables that are more accountable for optimally splitting the data into mutually exclusive, but homogeneous groups. It creates a hierarchical structure that can be visualised and interpreted easily. Under the tree each data record falls through a unique path to land into a terminal node that aggregates similar records amongst themselves. However, a tree predicts the outcome of each terminal node by a constant. Even though a tree identifies the locality of homogeneous groups, these localities, once-identified can be handed over to the GLM method to devise a hybrid model that combines the benefits of both.

In this section an algorithm is devised that uses a regression tree to improve the predictive power of a GLM. Since the model uses inputs from a tree to build the GLM, it is a hybrid model. As the first step of the hybrid model a regression tree is developed by recursively splitting the predictor space of the insurance data. The response variable for the analysis is the claim frequency of insureds. A full tree is first grown and then pruned to the optimal level.

Due to the very large number of zero claims compared to positive claims, the distribution of the number of claims is best fitted using the Poisson distribution. Once the optimal tree is obtained, a categorical variable is added to the predictor space. This categorical variable has as many levels as there are terminal nodes. In other words, this categorical variable identifies the terminal node of the tree to which a record belongs. The base class for this variable corresponds to the node with the highest exposure. The addition of this categorical variable with the variables identified by the tree as the best candidates for its splits are used to create a GLM model. This hybrid model produces an improvement when bench marked against a traditional GLM that is built with all the available variables in the predictor space. The method has been tried on a public French motor insurance data available as a CAS dataset. From the analysis in the following section it can be seen that each bin, the newly added categorical variable under which all the records within a homogeneous terminal node of the tree reside, turns out to be a significant variable.

Further improvement is gained when interactions between the bin variable and regular variables are added to the GLM. The overall method is schematized in Algorithm 1,

followed by the analysis of the empirical data and the improvements obtained.

Algorithm 1 Implementation of the hybrid tree–based GLM

Input:Training dataset train, Test dataset test.
Output: Tree structure, node information, GLM model, fitted dataset.

Grow a large tree on the training dataset using binary splitting.
Prune the tree using cost–complexity.
Obtain the exposure for each terminal node.
Obtain the response for each terminal node.
$K \leftarrow$ number of terminal nodes.
$bin \leftarrow$ list of zeros of length equal to the training data.
$bin_{test} \leftarrow$ list of zeros of length equal to the test data.
Predict training set using the tree and obtain the node information for each record.
Predict the test set using the tree and obtain the node information for each test record.
for $k \in K$ **do**
  if $exposure \neq max(exposure)$ **then**
    $bin \leftarrow k$
    $bin_{test} \leftarrow k$
  else
    continue
  end if
end for
$train[Bin] \leftarrow bin$
$test[Bin] \leftarrow bin_{test}$
$var_{imp} \leftarrow$ non-zero variables of importance from tree.
formula $\leftarrow y \sim Bin+$
for $var \in var_{imp}$ **do**
  if var is last var_{imp} **then**
    formula = formula + var + var*Bin
  else
    formula = formula + var + var*Bin +
  end if
end for
GLM_{model} = GLM(formula, data = train , model = Poisson)
  return $GLM_{model}, tree_{model}, GLM_{fit}$ on train, GLM_{fit} on test

2.6 Data and Results

The dataset *freMTPL2freq* used in this analysis is available is the R package CASdatasets. The dataset has 678,013 records where third party liabilities were collected over a period of one year. The dataset has twelve columns including one column for the policy ID that

is a unique identification number for each policy. The other variables are as follows

1. ClaimNb : Number of claims submitted during the exposure period.
2. Exposure : The exposure period.
3. Area : The area where the vehicle is registered.
4. VehPower : The power of the vehicle. This is an ordered categorical type variable.
5. VehAge: The age of the vehicle in years.
6. DrivAge : The age of the driver in years. The minimum being 18 years in France.
7. BonusMalus : Bonus/malus between 50 and 350: < 100 means bonus, >100 means Malus in France.
8. VehBrand : The vehicle brand. This is a categorical variable.
9. VehGas : The vehicle gas type. This variable is a binary categorical variable with two types, Regular and Diesel.
10. Density : Density of population in terms of the number of people living in per km^2 area of the city where the driver of the vehicle lives.
11. Region : The policy region in France (based on standard French classification). This variable is categorical.

The dataset has approximately 5% claims putting a large mass on zero claims like any other insurance data. The dataset has a few records with exposures more than a year which were corrected to be 1 year. Also, it was noted that 9 policies have more than 4 claims with the largest number of claims being 16. This was corrected by making the maximum number of claims to be 4.

After the initial rectifications, the dataset is divided into the training set (learn) which contains 80% of the dataset and the remaining 20% is kept for validation (test) of the models trained. A basic GLM model is first built on the training data set. The output of the R function summary() when run on this model and all the subsequent models are in Appendix B. This GLM is used as a benchmark to gauge the relative performance of the GLMs preprocessed using tree structure. From the summary of this benchmarking GLM it is visible that most of the regions are not statistically significant along with several areas. The variable Density is not significant along with most of the vehicle brands.

Fig 2.1 is the optimally pruned tree of maximum depth 3 that splits the data into six homogeneous leaves. We call each of these leaves a bin and create a new categorical variable Bin in the training as well as the validation dataset. The bin with the highest exposure forms the base bin (Bin 0) here. It is the leaf of the tree corresponding to the vehicles more than a year in age and with drivers with BonusMalus less than 58. The next GLM model is created including the additional and categorical variable Bin in the data.

The summary of the model is included in Appendix B. Here from one bin to the other the intercepts vary while the slopes of the covariates remain constant. The following shows the relationship between the expected response and the intercept of the base bin when all other variables are set to zero:

$$E(y|\mathbf{X}=0) = \exp\{intercept\}.$$

This base value changes from the base bin to any other bin (bin 4 say) as follows

$$E(y|\mathbf{X}=0) = \exp\{intercept + \beta_{Bin4}\} = \exp\{intercept\}\exp\{\beta_{Bin4}\}.$$

Keeping everything else 0, only for Vehicle Brand 12 the expected response in the base bin is

$$\exp\{intercept\} + \beta_{VehBrand12}$$

and the same for the bin 4 is

$$\begin{aligned}\exp\{intercept + \beta_{Bin4} + \beta_{VehBrand12}\} &= \exp\{intercept\} \\ &\quad \exp\{\beta_{Bin4}\} \times \exp\{\beta_{VehBrand12}\}\end{aligned}$$

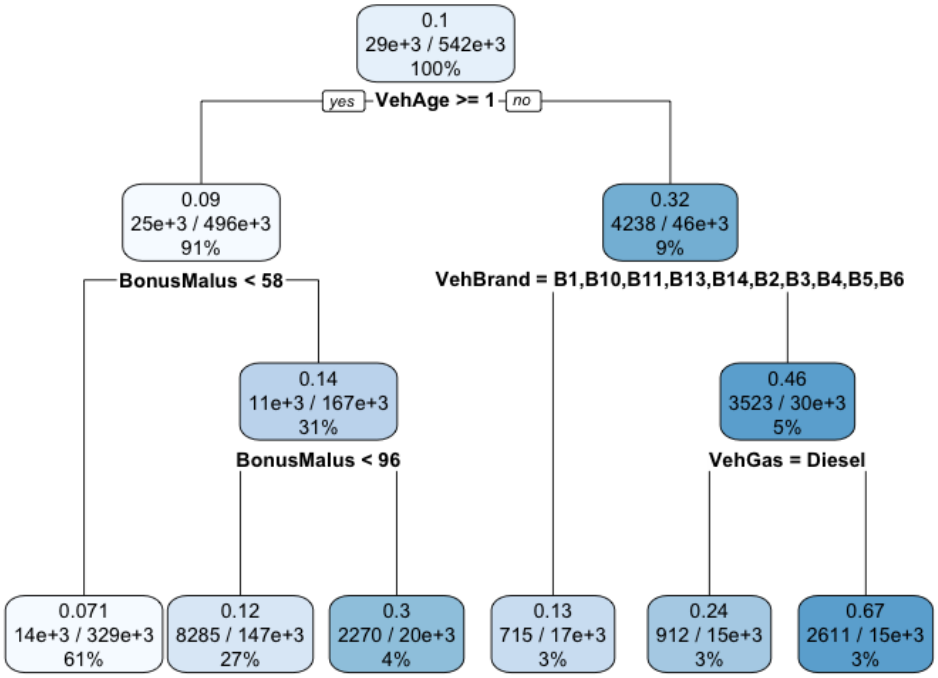


Figure 2.1: The optimal tree with 6 leaves from the training data

From the summary table in Appendix B it is clear that all Bin variables are statistically significant. This means that the intercepts for every group/bin are significant. Considering Akaike's information criterion (AIC), that includes a penalty term for over–fitting,

the model with the smallest AIC value should be preferred. Here, based on the AIC we should prefer slightly the GLM model with the bin variables included.

Figure 2.2 shows that the average predictions of the GLM with the Bin variable are closer to the actual averages in Deciles 3 to 9. The average predictions of this model are more underestimated in Decile 1 and 2 and overestimated in Decile 10. The earned exposures are also plotted as a measure of volume for each decile, a good fit being more important where the weight is larger.

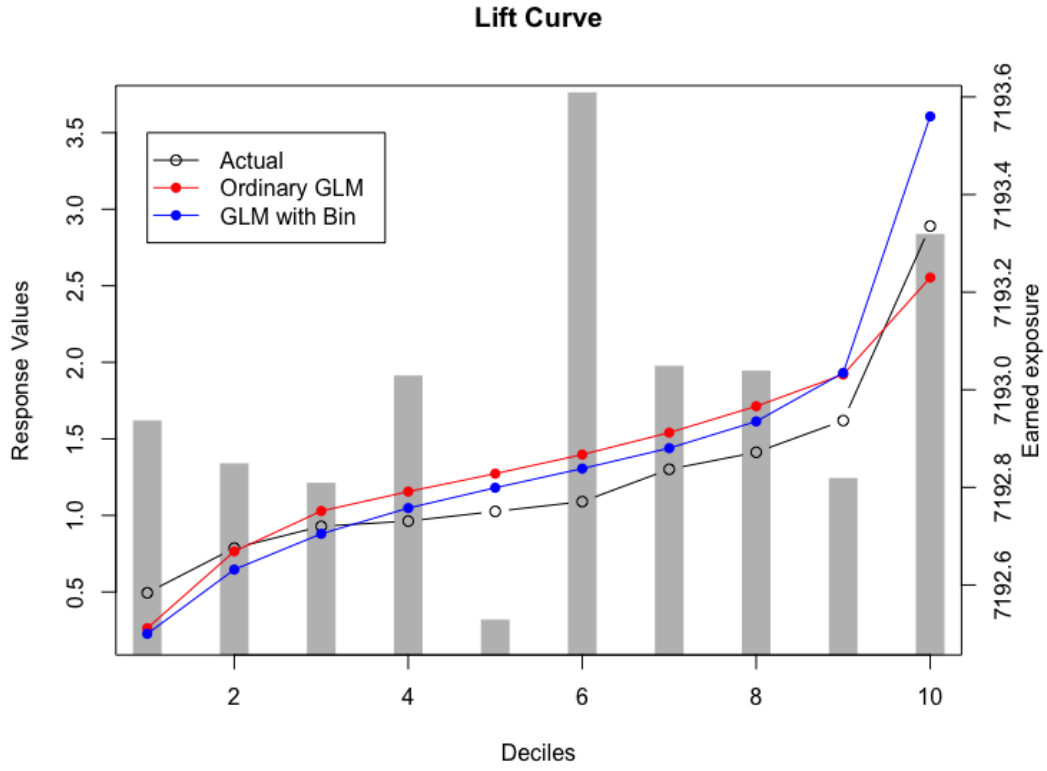


Figure 2.2: Lift curve comparing the predictability of the GLM with Bin to the benchmark GLM

Finally, let us go one step further to include interaction terms between the Bin and other variables. This generates different intercepts for each bin as well as different slopes for each variable within each bin. It is equivalent to having different GLMs within each tree leaf/bin. Hence, keeping every other variable at 0, the relation between the expected frequency and the Bonus Malus in the base bin is

$$\exp\{intercept\}\exp\{\beta_{BonusMalus}\},$$

while the same for Bin 4 (say) is given by

$$\exp\{intercept\}\exp\{\beta_{Bin4}\}\exp\{\beta_{BonusMalus}\}\exp\{\beta_{BonusMalus:Bin4}\}.$$

From the Appendix B summary it appears that not all variables are equally significant within each bin. For example the region named Centre is significant in Bins 1 and 5, not significant for Bin 4, while it is not present in Bin 0 (base bin), 3 and 6, which means there is no data from this region in these bins due to the tree splits.

Figure 2.3 compares the average predictions given by the benchmarking GLM to the GLM that includes the Bin variable and its interaction with other covariates. In the latter the average predictions is closer to the actual predictions in Deciles 3 to 9.

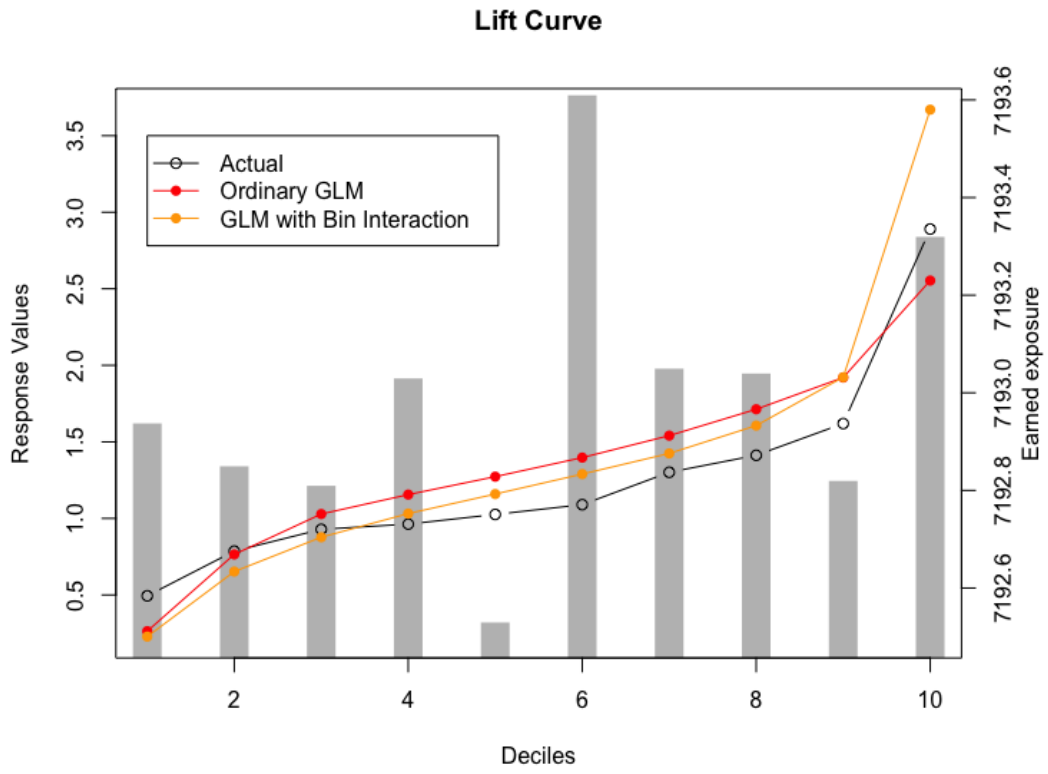


Figure 2.3: Lift curve comparing the predictability of the GLM with Bin and its interaction with other covariates to the benchmark GLM

The double lift curve in Figure 2.4 shows that the average predictions of the GLM with the Bin variable interacting with the other covariates is closest to the actual response for all deciles.

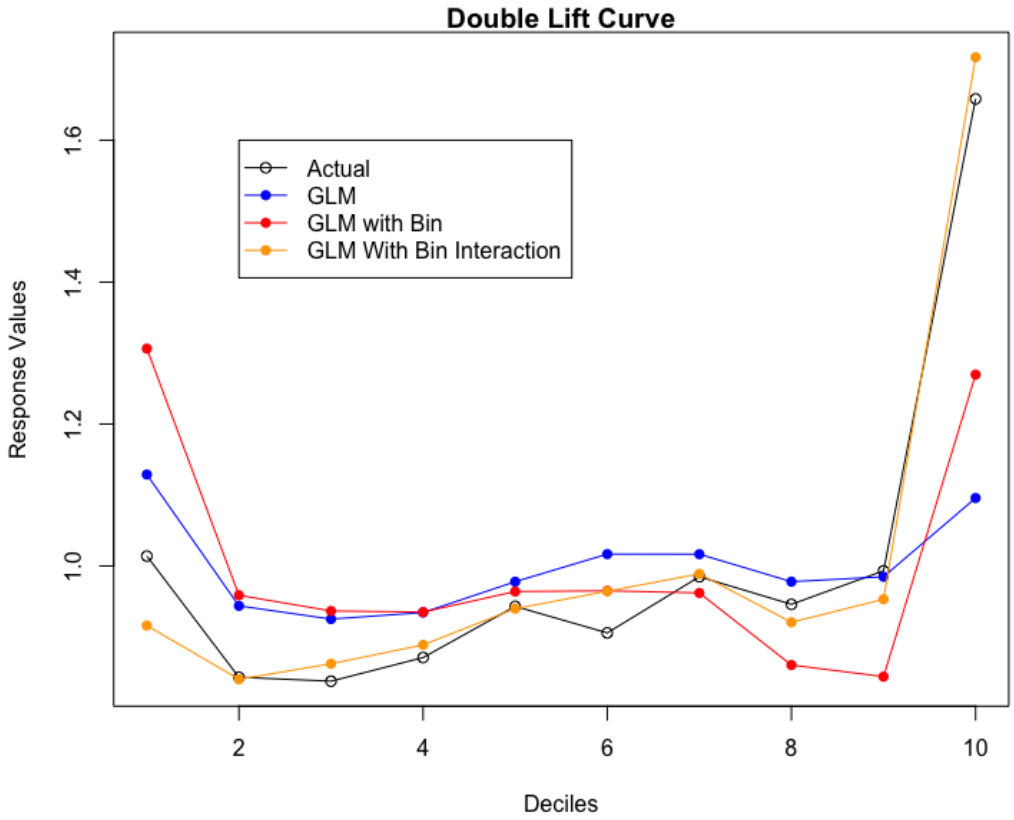


Figure 2.4: Double lift curve comparing the predictability of the GLM with Bin variable, to GLM with Bin and its interaction with other covariates and to the benchmark GLM

The summary in Appendix B shows the statistical significance of the Bin variables and the presence of quite a few significant interactions between Bin and other variables. Out of the 3 GLMs the AIC score is lowest for this model.

Lastly Table 2.3 compares the prediction capability of each model over the training as well as the validation set. It is clear that the GLM model with Bin and variable interaction with Bin performs best.

	Percentage mean Poisson deviance from training set	Percentage mean Poisson deviance from validation set
Bench Marking GLM	31.973	31.932
GLM with Bin	30.938	31.064
GLM with Bin Interactions	30.724	30.935

Table 2.3: Comparison of predictive performance of GLM with tree Bins and GLM with tree bins interacting with variables with respect to the benchmark GLM

Chapter 3

Hybrid Combination of Gradual Minority Oversampling and GLMNet

Abstract

Tree-based methods have gained popularity to model non-linear relationships in data. Classification and regression trees are non parametric methods and divide the predictor space into more homogeneous subspaces. Given that insurance claims are rare events in portfolios of insurance policies, the prediction of claim proneness, and in turn claim frequency, is like searching for a few needles in a haystack. Here the haystack is the large number of policies that did not file a claim, which dominate the portfolio dataset. No single model is perfect at sieving out the policies that might claim, and thus, a combination of methods is suggested here, to improve performance in claim frequency modelling. This chapter mixes the knowledge obtained about the impact of oversampling the minority class (group of policyholders that filed a claim), studied in Chapter 1, to divide the predictor space into bins that contain claim propensity in different proportions. Then, in a second step, these bins are used as predictor inputs into a GLMNet model to improve the claim frequency predictions in a regression setting.

3.1 Introduction and Motivation

Building good regression models for insurance claims prediction is a challenge. One of these challenges with insurance datasets is the presence of a high proportion of policies that have not filed a claim (called policies with zero claims), as compared to policies with claims. This creates extreme skewness in the data. In the context of regression, the covariates (predictors or independent variables) throw light on the heterogeneity of the individual policies.

Insurance claims are often studied in two parts, the first being the claim frequency model, and the second part being the average severity of claims per policy. The product of these two components is known as the loss cost and the product of their expected

values is the pure premium.

In modelling the claim frequency of each policyholder it can be analyzed as a Poisson process, where the claims within a given time period (a year in most cases) occur at a given rate. It is the differences in this rate, from on policyholder profile to another, that needs to be predicted by an actuarial model. However, when we look into the predictor space as a whole, the percentage of policyholders with claims is usually less than 10% of the total portfolio. This unbalanced distribution of risky clients makes difficult the task of identifying their higher premium requirement.

Several methods of the two part framework of insurance models are discussed in Klugman et al. (2012). Employing regular classification trees on a dataset with less than 10% claims usually requires a large tree, to identify the very few profiles with non-zero claims (minority class). Note that for a dataset with only 3% claims, if a model predicts every policy holder as non-risky, then the model accuracy rate would be of 97%, which is excellent. Thus the usual predictive accuracy criteria to assess models is not sufficient to measure the actual performance of a classification model when there is such imbalance in the data.

The problem of binary classification of insurance portfolios, along the zero claim to non-zero claims divide, is discussed in Chapter 1, where we propose a method that improves the identification of non-zero claims profiles. There we note the considerable loss in accuracy to predict zero claims. Nevertheless, it is crucial to identify correctly potential claimants, as there is a real cost in classifying non-risky clients as risky, leading to the loss of customers in a competitive market. The result in Chapter 1 shows that one can control the misclassification of zero claim profiles by adjusting the proportion of oversampled minority class data points. That same idea is used in this chapter, but in a regression context, to identify subsets in the predictor space with higher or lower proportions of claims. To identify the policyholders with a risky profile the predictor space is binned into corresponding "more risky" and "less risky" bins.

The bins are created using the prediction results of classification trees applied to the predictor space. Clearly, these bins are dependent on the prediction variables and may introduce collinearity in the regression model. This is one of the reasons why we choose to use GLMNet over regular GLMs in the second phase. GLMnet is a package that fits generalized linear models with a penalized likelihood. The regularization path is computed for a lasso or elastic net penalty at grid values (on the log scale), the regularization parameter is called lambda (λ). The model applies a regularization penalty to the regression coefficients to reduce overfitting. In more practical terms, in the presence of dependent covariates the model does an automatic features selection with the least significant covariates having their coefficients reduced to 0. The *R* library GLMNet package implements the method and includes a functionality for log–Poisson models used by actuaries for frequency (claim count) models.

Apart from combining bins to prediction variables we also consider a reduced model with bin variables only. The motivation is that the bins identify subsets of the overall predictor space that have been constructed by optimally splitting the sample space using classification trees. For example, it is possible that some of the bins might include only

one of the levels of a character variable and there it does not make much sense to include the variable itself, as the bin already reflects the split generated by the tree at the previous step.

The rest of this chapter is organized as follows. Section 3.2 provides the methodology and the technical details of the two-step hybrid method. In Section 3.3 we investigate various models that can be used for modelling claim frequency. Section 3.4 contains the results of the hybrid method and their visualization on a CAS dataset. It also gives a comparison of these results with those obtained from methods discussed in the previous section. Finally Section 3.5 gives the conclusions.

3.2 Methodology

The hybrid method can be divided broadly into 3 steps:

1. Start with the original proportion of the minority and the majority classes, and build a classification tree. Depending on the proportion of observations in the minority class, and the complexity parameter selected there are 2 possibilities :

   (i) None of the non-zero claims get identified by the tree. At this stage it is possible to decrease the complexity parameter which leads to building a bigger tree and identifying some of the non-zero claims. However, the bigger the tree, the higher the risk of overfitting. The complexity parameter has to be tuned to get the best result.

   (ii) Some of the non-zero claims do get identified. It is not identifying individual policy records correctly, but rather one or more nodes of the tree identify a predictor subspace as being risky. This subspace would also contain zero claims but in a lesser percentage than the non-zero claims. The subspace now gets identified as the first bin and all the policies belonging to this subspace would have the corresponding bin classification value (risky or non-risky).

2. Progressively, the number of minority class data points is doubled, then tripled and so on until their sampled number becomes equal to the number in the majority class. It is to be noted that the sample points from the majority class is kept fixed through out. A classification tree is built at every step while adding more minority sample points and the corresponding prediction of the tree is noted. The predictions that change from zero in the previous tree to non-zero (claims) in the current tree are taken into a new bin.

3. Having divided the data into discrete bins a matrix of the bins is produced by hot encoding and is used as the predictor for GLMNet model. The first bin would have the highest proportion of claims compared to the following bins. Bin 0 is the default bin. Records in this bin have never been categorized as non-zero claims. Within the dataset it is the subspace with lowest proportion of claims, if any.

3.2.1 How Does this Method Help

Let us consider a predictor subspace $\mathscr{S}$ that is d dimensional. Let $Y \in 0,1$ be the response variable where $Y = 1$ implies that there is at least a claim (also referred to above as risky or non-zero). Let q be the proportion of records with at least a claim in this space and also let n be the total number of records. Thus empirically,

$$P(Y = 1|X \in \mathscr{S}) = \frac{nq}{n} = q$$

and

$$P(Y = 0|X \in \mathscr{S}) = \frac{n(1-q)}{n} = (1-q).$$

At this point when we add nq sample points from the minority data into the predictor space then,

$$P(Y = 1|X\ in \mathscr{S}) = \frac{2nq}{n+nq}$$

and

$$P(Y = 0|X \in \mathscr{S}) = \frac{n(1-q)}{n+nq}.$$

At this point if $P(Y = 1|X \in \mathscr{S}) > P(Y = 0|\mathscr{S})$, only then will an algorithm predict points belonging to $\mathscr{S}$ as risky. And for that it is necessary that

$$2nq > n(1-q) \quad \Leftrightarrow \quad 3nq > n \quad \Leftrightarrow \quad q > \frac{1}{3}.$$

Thus a predictor subspace which has more than one-third of its records being risky will be identified. Denote this as being the second stage in the subsampling process, that is $k = 2$.

Then, at any given k^{th} stage of the subsampling process, the minority points are increased k-folds. If records belonging to the predictor space $\mathscr{S}$ are classified as being risky, then at this stage it means that

$$P(Y = 1|\mathscr{S}) > P(Y = 0|\mathscr{S}) \quad \Leftrightarrow \quad knq > n(1-q) \quad \Leftrightarrow \quad (k+1)nq > n \quad \Leftrightarrow \quad q > \frac{1}{k+1}.$$

For example, consider a data subspace $\mathscr{S}_{\mathscr{A}}$ that is predicted risky ($\hat{y} = 1$) in the 4^{th} iteration. This means that the classification tree requires 4 times more minority sample points than there actually is, to be able to see the presence of minority data points in this subspace. Note that it was predicted non-risky ($\hat{y} = 0$) in the 3^{rd} iteration, when the subsample contained 3 times more minority sample points than the original number. It can be seen from the result above that for $k = 4$, the fraction of minority data points in $\mathscr{S}_{\mathscr{A}}$ is at least $1/(4+1)$, that is 20%. Thus, the larger k, the smaller the actual proportion of minority data points in $\mathscr{S}_{\mathscr{A}}$.

All the data points whose predictions change from 0 to 1 at the k^{th} stage are put in the same bin. Bin 0 is the one with the lowest proportion of risky records and Bin 1 is the bin with highest proportion of the same. The the other bins contain a proportion

of risky data points that is in between those in Bin 0 and Bin 1. Then the bins are used as categorical variables to form the one-hot encoded matrix used as predictors of the GLMNet model. Algorithm 2 describes the process.

Algorithm 2 Binning the data and building GLMNet with the bins

Input:Training dataset train, Test dataset test.
Output: GLMNet model.

$K \leftarrow \lfloor \frac{number\ of\ majority\ points}{number\ of\ minority\ points} \rfloor$.
$n_q \leftarrow minority\ data\ size$
$n_p \leftarrow majority\ datasize$
$pred_{mat}[,1] \leftarrow predict(tree, train)$
$test_{mat}[,1] \leftarrow predict(tree, test)$
for $k \in 2:K$ **do**
  $min_{sample} \leftarrow sample(k * n_q, minority\ data)$
  $maj_{sample} \leftarrow sample(n_p, majority\ data)$
  $sub_{sample} \leftarrow min_{sample} + maj_{sample}$
  $tree \leftarrow rpart(sub_{sample}, method = class)$
  $pred_{mat}[,k] \leftarrow predict(tree, train)$
  $test_{mat}[,k] \leftarrow predict(tree, test)$
end for
$bin_{cnt} = 0$
$bin = [0, size = train]$
$bin_{test} = [0, size = test]$
if $(sum(pred_{mat}[,k] == 1)! = 0)$ **then**
  $bin_{cnt} = bin_{cnt} + 1$
  $bin[pred_{mat}[,k] == 1] = bin_{cnt}$
  $bin_{test}[test_{mat}[,k] == 1] = bin_{cnt}$
end if
for $k \in 2:K$ **do**
  if $(sum(pred_{mat}[,k] == 1\ \&\ pred_{mat}[,k-1] == 0)! = 0)$ **then**
    $bin_{cnt} = bin_{cnt} + 1$
    $bin[pred_{mat}[,k] == 1\ \&\ pred_mat[,k-1] == 0] = bin_{cnt}$
    $bin_{test}[test_{mat}[,k] == 1\ \&\ test_{mat}[,k-1] == 0] = bin_{cnt}$
  end if
end for
$train[Bin] \leftarrow bin$
$test[Bin] \leftarrow bin_{test}$
$GLMNet_{model} = glmnet(formula, data = train[Bin], model = Poisson)$
  return $GLMNet_{model}$

3.2.2 GLMNet

As mentioned earlier we build a GLMnet with predictor variables and the bins, as well as a GLMNet with only the bins. Let $\beta_0, \beta_1, \ldots, \beta_p$ be the coefficients corresponding to the p predictor variables $X_{i1}, X_{i2}, \ldots, X_{i,p}$. The Poisson distribution belongs to the exponential dispersion family, as the Poisson probability mass function can be written as equation (1). The Poisson link function is the log. Let us look at the Poisson probability mass function (p.m.f.) $f(y_i)$, where $i = 1, 2, \ldots, N$ is the record index:

$$f(y_i) = \frac{e^{-\lambda}\lambda^{y_i}}{y_i!} = e^{(y_i \ln\lambda - \lambda)}\frac{1}{y_i!}.$$

The likelihood of this dataset is given by

$$\Pi_{i=1}^{N} f(y_i) = \Pi_{i=1}^{N} e^{(y_i \ln\lambda - \lambda)} \Pi_{i=1}^{N} \frac{1}{y_i!}.$$

The Log-likelihood of the dataset is thus

$$\Sigma_{i=1}^{N} log(f(y_i)) = \Sigma_{i=1}^{N}(y_i \ln\lambda - \lambda) + log(\Pi_{i=1}^{N}\frac{1}{y_i!}).$$

As mentioned earlier, we need the prediction for $\lambda = E(Y|\mathscr{X}, \beta)$, modeled using the link function

$$\log\big(E(y_i|\mathscr{X}, \beta)\big) = \beta_0 + \Sigma_{j=1}^{p}\beta_j x_{ij}.$$

The log likelihood for dataset $\{x_i, y_i\}$ is given by

$$l(\beta|\mathscr{X}, Y) = \Sigma_{i=1}^{N}\left(y_i(\beta_0 + \Sigma_{j=1}^{p}\beta_j x_{ij}) - e^{\beta_0 + \Sigma_{j=1}^{p}\beta_j x_{ij}}\right). \tag{15}$$

The penalized log-likelihood that is optimized is

$$\min_{\beta_0,\beta}\left\{-\frac{1}{N}l(\beta|\mathscr{X}, Y) + \Sigma_{i=1}^{N}\lambda\left[(1-\alpha)\frac{1}{2}\Sigma_{j=1}^{p}\beta_j^2 + \alpha\Sigma_{j=1}^{p}|\beta_j|\right]\right\}. \tag{16}$$

Unlike the *glm* function in *R*, GLMNet itself does not accept *data.frame* objects as an input and requires a model matrix. After the creation of the bins, the X_{ij} value in equation (15) is replaced by bin_i. The bin variable is a character variable and the bin variable is zero for the base class for which is Bin 0. Recall, that the bin is zero for records belonging to the subspace which has the least percentage of claims. For K bins the matrix produces K columns with 1 at the column corresponding to the bin and 0 elsewhere. Thus the intercept, β_0 corresponds to the base bin=0. The Poisson predictions are multiplicative. Thus the prediction corresponding to the k^{th} bin is

$$exposure * \left(\exp\{\beta_0\}\exp\{\beta_k\}\right),$$

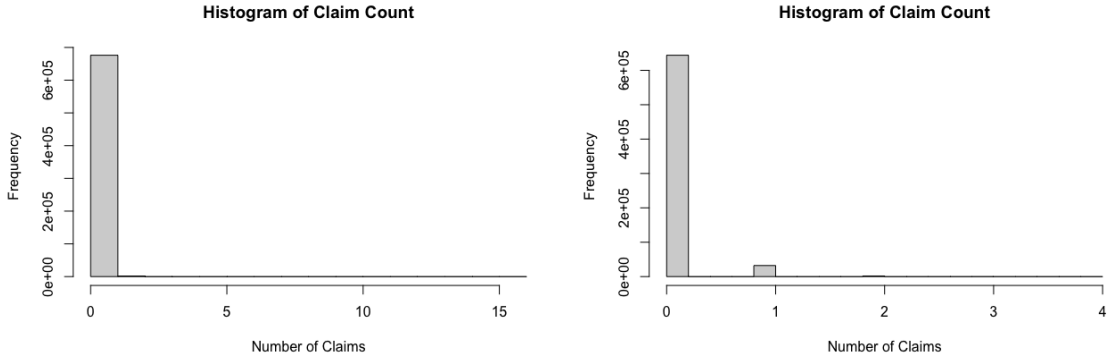
where exposure is the fraction of the time period the policy was valid for. Thus a policy valid for 3 months has $exposure = 0.25$ while the one that is valid for a year has $exposure = 1$.

3.3 Dataset, Models and Results

The dataset *freMTPL2freq* used in this analysis is available is the *R* package CASdatasets. It contains 678,013 records, where third party liability claims were collected over a period of one year. The dataset has twelve columns, including one column for the policy ID that is a unique identification number for each policy. The other variables are as follows:

1. ClaimNb : Number of claims submitted during the exposure period.
2. Exposure : The exposure period.
3. Area : The area where the vehicle is registered.
4. VehPower : The power of the vehicle. This is an ordered categorical variable.
5. VehAge: The age of the vehicle in years.
6. DrivAge : The age of the driver in years. The minimum being 18 years in France.
7. BonusMalus : Bonus/malus between 50 and 350: < 100 means bonus, > 100 means Malus in France.
8. VehBrand : The vehicle brand. This is a categorical variable.
9. VehGas : The vehicle gas type. This variable is a binary categorical variable with two types, Regular and Diesel.
10. Density : Density of population in terms of the number of people living per km^2 in the area of the city where the driver of the vehicle lives.
11. Region : The policy region in France (based on standard French classification). This variable is categorical.

The dataset has approximately 5% of the records being non–zero claims, meaning that the number of policies with zero claims is strongly dominant, as is common with insurance data. The dataset had a few records with exposures larger than a year, which was corrected to be 1 year. Fig 3.1(a) shows the histogram of the number of claims in the data. Table 3.1 shows that only 9 policies have more than 4 claims, with the highest number being 16. The percentage of exposure contribution for policies with more than 4 claims is 0.0009%. As this is marginal, to simplify the model the data was corrected to set the maximum number of claims to be 4. Fig 3.1(b) is the histogram on the dataset with claim count capped at 4.



(a) (b)

Figure 3.1: (a) Histogram of the original number of claims in the data (b) Histogram of the number of claims with maximum number of claims capped at 4

number of claims	0	1	2	3	4	5	6	8	9	11	16
number of policies	643953	32178	1784	82	7	2	1	1	1	3	1
total exposure	336616	20671	1153	53	3	1	0.3	0.4	0.1	1.1	0.3

Table 3.1: Split of the portfolio with respect to number of claims and exposure

3.3.1 Models like GBM and XGBoost

This section reviews two models also popular to describe insurance claim frequency. The tuning of the hyper parameters, specific to the dataset to obtain the best possible result, is discussed here as well as the results obtained.

Gradient Boosting Machine (GBM)

Gradient Boosting machines or GBMs are a family of powerful supervised machine learning techniques that are customizable to the particular need of the application. These models are non parametric ensemble methods that rely on combining a number of weak learners to obtain a strong prediction. The principle idea behind this algorithm is a learning procedure that results from consecutive error fitting. The choice of the loss function is up to the researcher, in our case we opted for the Poisson deviance.

The *R* library *gbm* is used to implement gradient boosting and for testing. The primary parameters of interest in this function are:

1. Formula : It specifies the model parameters and the dependent variable

2. Distribution : It specifies the distribution followed by the response variable and thus determines the loss function that will be used when fitting the model.

3. Number of trees : Also the number of iterations in the additive ensembling technique

4. Depth of a tree : This parameter specifies how deep each tree of the ensemble can be grown. The deeper a tree the stronger is it as a learner.

5. Minimum number of observations in the terminal node of the tree : This is the minimum number of observations (in integer form) allowed in a terminal node of the trees.

6. Shrinkage : Also known as the learning rate or step size. The smaller this rate is the larger the number of trees required.

7. Cross Validation folds : Number of cross-validation folds to perform. If larger than 1, then *gbm*, in addition to the usual fit, will perform a cross-validation, and calculate an estimate of the generalization error (returned in *cv.error*).

A hyper parameter grid is created for the *gbm* function to find out the best choice for the particular dataset. The first hyper parameter grid was created for the shrinkage parameter (or the learning rate). An optimal value of 0.3 was obtained. Having fixed the shrinkage to 0.3, the next grid was created for the number of trees, the depth of each tree and the minimum number of observations in the terminal node of the trees. The aim was to find the optimal number of trees that would produce the best Poisson deviance. This is because GLMNet will optimize the error rate for the Poisson distribution with a Poisson deviance loss function.

The optimal hyper parameters obtained in the second hyper parameter tuning round are as follows:

- Number of trees = 50,
- Depth of each tree = 7,
- Minimum number of observations in the terminal node of the tree = 15,
- Shrinkage (learning rate) = 0.3.

Figure 3.2 gives an idea of the training and test for the Poisson deviance against the number of iteration/trees. The corresponding model performance on the test data set is disclosed in the result section when all the model performances are compared.

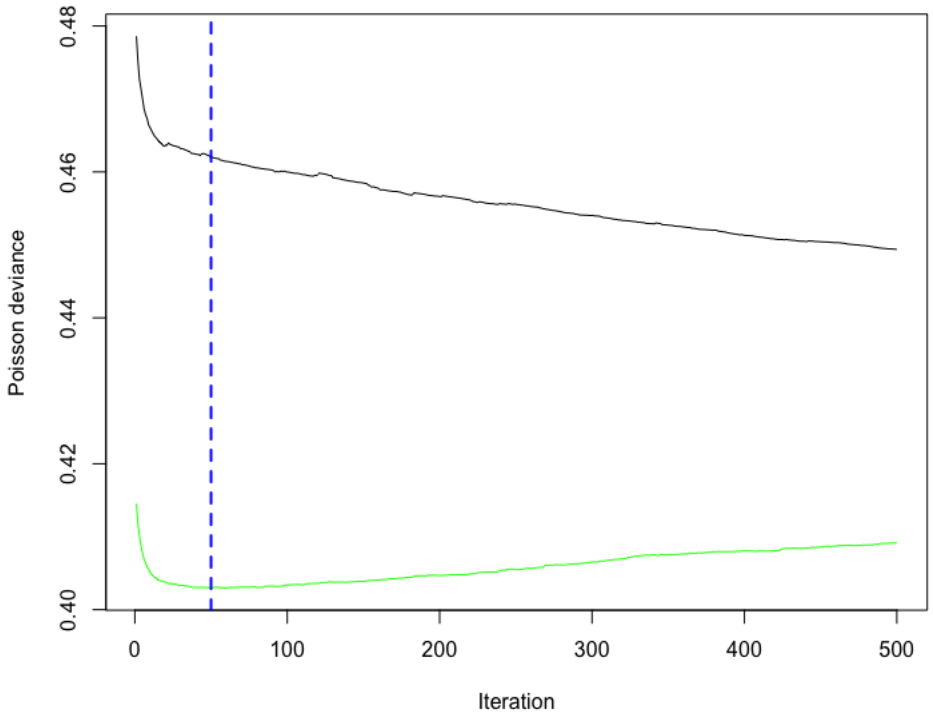


Figure 3.2: GBM training and testing Poisson deviance plot vs the number of iterations/trees

3.3.2 Extreme Gradient Boost (XGBoost)

Extreme Gradient Boosting (XGBoost) is another variant of boosting algorithm that uses regularization to manage overfit. XGBoost is a process that ideally requires less processing time, at the same time as it exhibits a satisfactory accuracy for various classification and regression cases. The algorithm can efficiently handle missing data and does not require prior imputations.

The *R* library *xgboost* was used to implement the XGBoost on our illustrative data set. Like GBM, several choices of loss function are available in *xgboost* and the one pertaining to our claim count application is Poisson. The parameters of interest in this function are of two types. The general parameters and the boosting parameters. The general parameters can be listed as follows:

1. booster : This parameter specifies the learner type. It is possible to use a regression learner in *xgboost*. However the default is *gbtree* for gradient boosting trees.

2. silent : The parameter specifies if any running message will be printed. The default value is 0. Once set to 1, there will be no message printed while xgboost runs.

3. nthread : This is used for parallel processing, and the number of cores in the system should be entered. If no value is entered the algorithm is intelligent enough to detect the cores and run them all.

The boosting parameters are the ones that need to be tuned for optimal performance of the algorithm. For our application, the boosting parameters interest are as follows:

1. eta : This the learning rate or shrinkage in GBM. Typically the value is tuned somewhere between 0.01 to 0.2.
2. min_child_weight : This is the minimum sum of weights of the observations required in a child node of the trees.
3. max_depth : This is the maximum depth to which the trees can be grown.
4. max_leaf_nodes: The maximum number of terminal nodes or leaves in a tree.
5. gamma : Specifies the minimum loss reduction required to make a tree split. This parameter is tuned depending on the loss function.
6. subsample : Denotes the fraction of observations to be sampled when building each tree. Mainly useful for large datasets. We set this parameter to 1.
7. colsample_bytree : Denotes the fraction of columns/variables to be sampled when building each tree. Mainly useful for datasets with a large number of variates. We set it to 1.
8. nrounds[default=100] : It controls the maximum number of iterations (steps) required for gradient descent to converge.This parameter is tuned using cross validation.
9. lambda : L2 regularization term on weights (analogous to ridge regression).
10. alpha : L1 regularization term on weight (analogous to Lasso regression).

We start with the hyper parameter tuning of the XGBoost model by first training it with the default parameters. The in-built cross validation function *xgb.cv* calculates the number of iterations (nrounds). The best iteration is obtained as 89.

Thereafter a grid search is made to find the best eta/learning rate, the maximum depth of trees, and the min_child_weight. The learning rates considered are 0.05, 0.1, 0.2, 0.5 and 1. The maximum depths of trees considered are 2, 4, 6 and 10. The min_child_weight that were considered in the grid are 6, 8, 10 and 10.

Having optimized the parameters in the first grid search, the regularization parameters are tuned in the second round grid search. The parameters, gamma, lambda and alpha are then tuned.

The optimal hyper parameters obtained after all the tuning rounds are as follows:

- Number of rounds = 89,

- eta = 0.05,
- max_depth = 6,
- min_child_weight = 6,
- gamma = 0.2,
- alpha= 0.01,
- lambda = 0.1.

The optimal XGBoost model performance on the test data is disclosed in the result section along with the performance of the optimized GBM and the hybrid model.

3.3.3 Hybrid Model Results and Discussion

As argued, the few initial trees only have a single leaf node. Unless the proportion of non-zero claims data goes up to a substantial percentage of the full dataset, *rpart* fails to detect their presence. Figure 3.3 shows the common result obtained by the first three classification trees.

Figure 3.3: Classification tree for k= 1, 2, 3 when CP = 0.001

For this dataset the tree starts splitting when $k = 4$; Figure 3.4 shows the splits of the fourth classification tree built during the process.

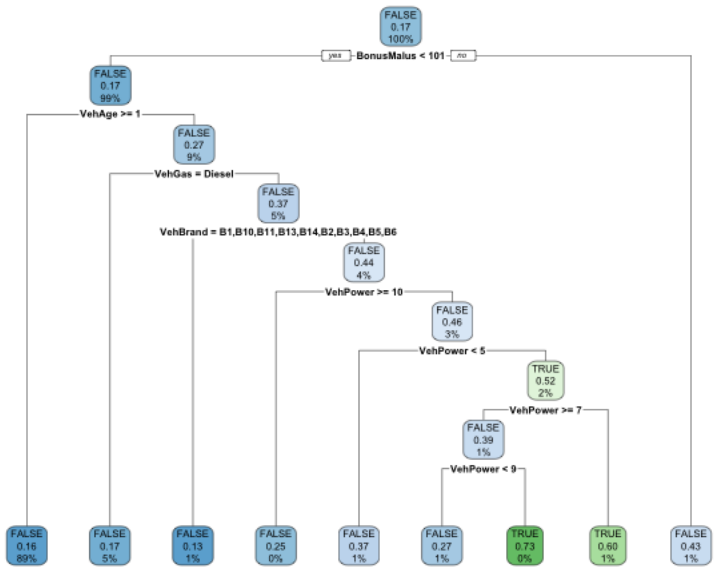


Figure 3.4: Classification tree for k=4 and CP=0.001

Figure 3.5 shows the splits of the tree for $k = 19$.

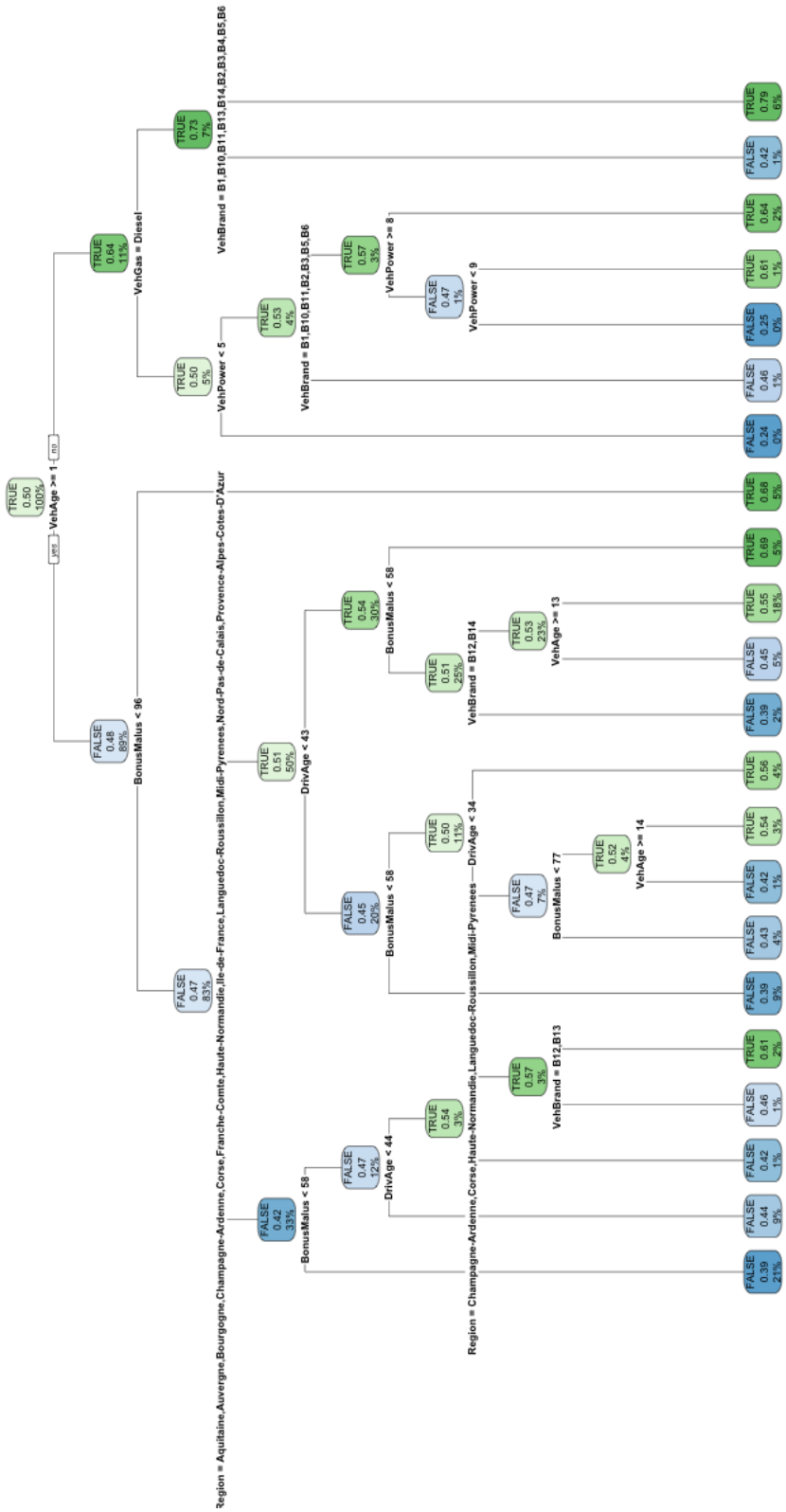


Figure 3.5: Classification tree for k=19 and CP=0.001

Table 3.2 gives an idea of how the bins identify the subspaces that have higher proportions of claim records. The data tells us how the bin corresponding to $k = 4$ identifies that predictor space where there is almost 29% claim records as compared to the overall 5% claim records in the dataset.

K value	Percentage of Claim Records
0	3.49
4	28.78
6	21.68
9	17.30
19	8.07

Table 3.2: Percentage of claim records for different k values

We differentiate the records using the bin variables. There are subspaces within the predictor space where there is 28% claim records when the overall claim percentage is only 5%. Figure 3.6 shows the β coefficient values for different values of the regularization parameter lambda in equation (15). This plot includes the predictor variables X_j along with the bins.

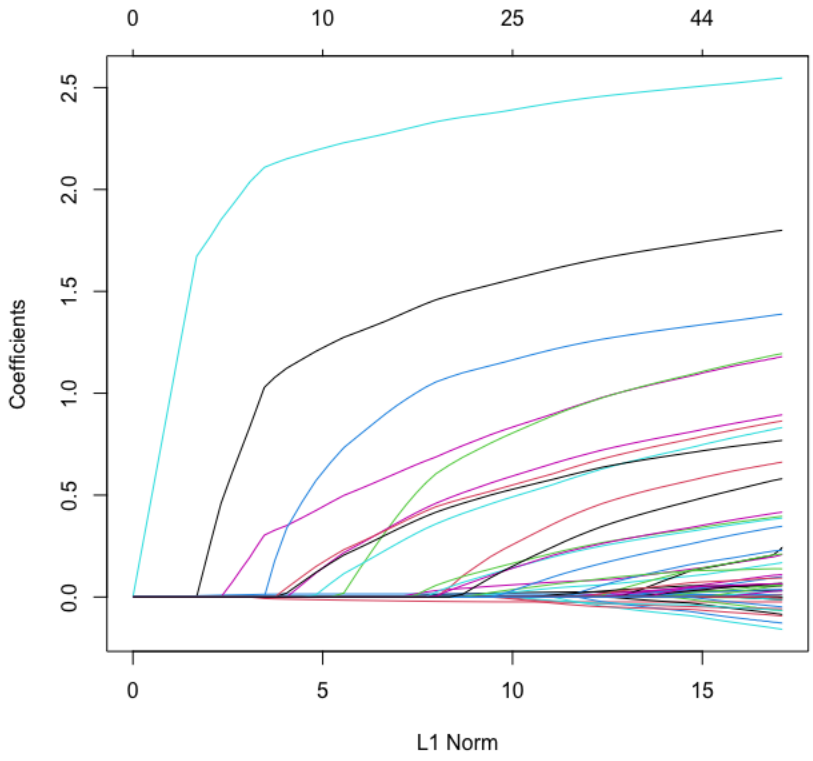


Figure 3.6: GLMNet coefficients vs the regularization parameter lambda

Figure 3.7 shows how the Poisson deviance changes with the regularization parameter lambda. The final GLMNet model is built with the best lambda which the one that minimizes the Poisson Deviance.

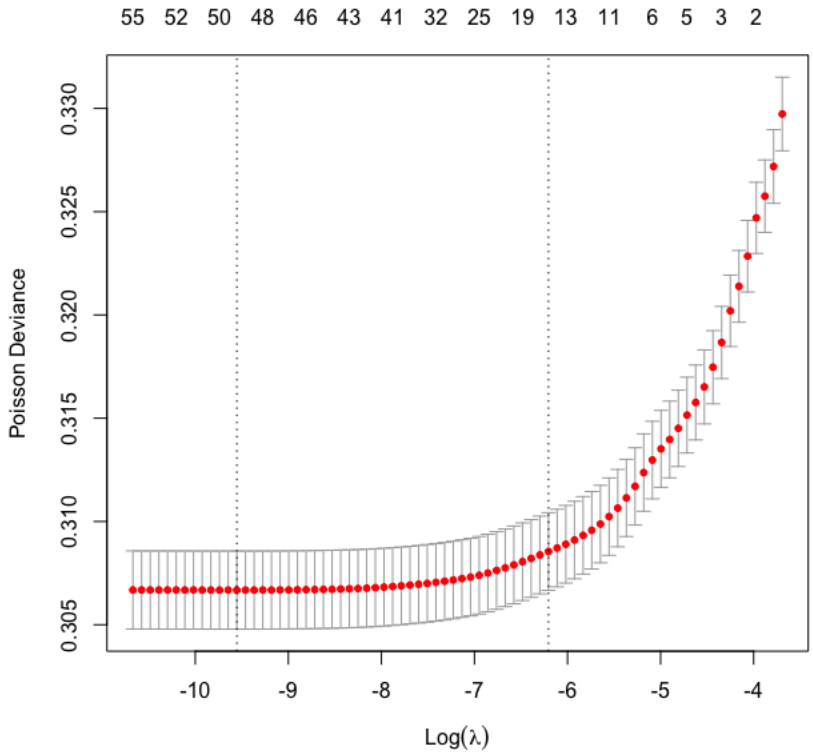


Figure 3.7: Poisson deviance against the log values of the regularization parameter lambda

Figure 3.8 shows that the most important variables are the intercept and the coefficients for the bins that have higher percentage of minority/non-zero claims data. The number of bins formed is not equal to the number of trees built, which is 19 for this data set. Recall that the trees did not split in the first few iterations. There are 16 bins and the intercept accounts for the bin with least percentage of claims. Bin 1, which is for the second most important variable, is the subspace with approximately 29% of non-zero claims.

The next model considered excludes the predictors and considers only the bin variables built with the trees.

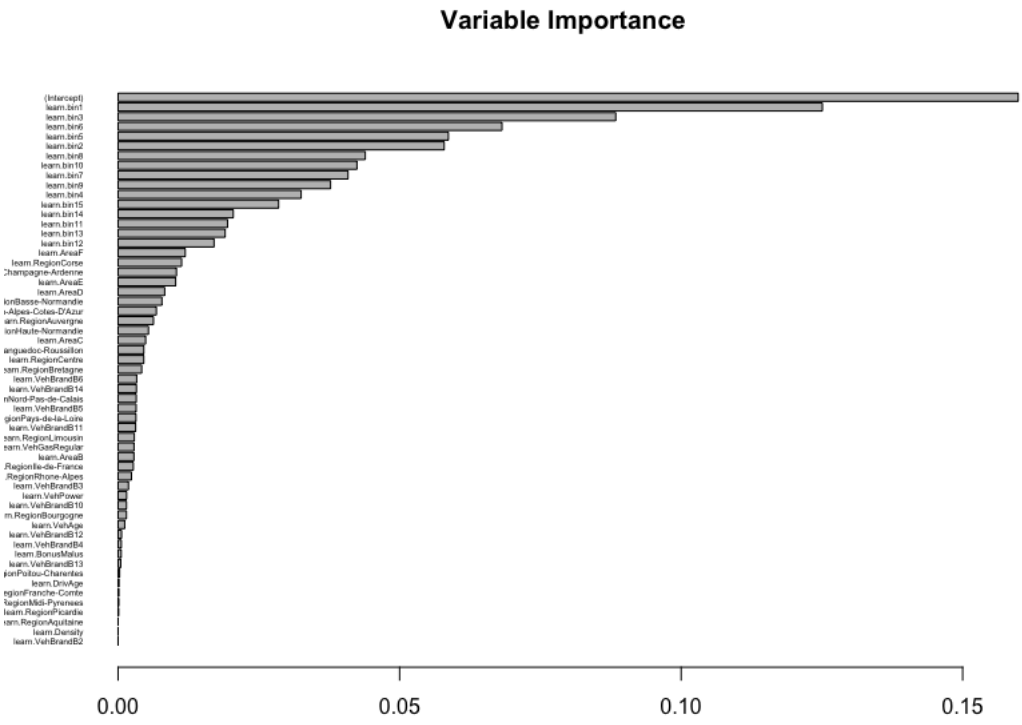


Figure 3.8: Horizontal barplots for variable importance

Figure 3.9 shows the variation in the coefficients for the bins by changing the value of the regularization parameter lambda.

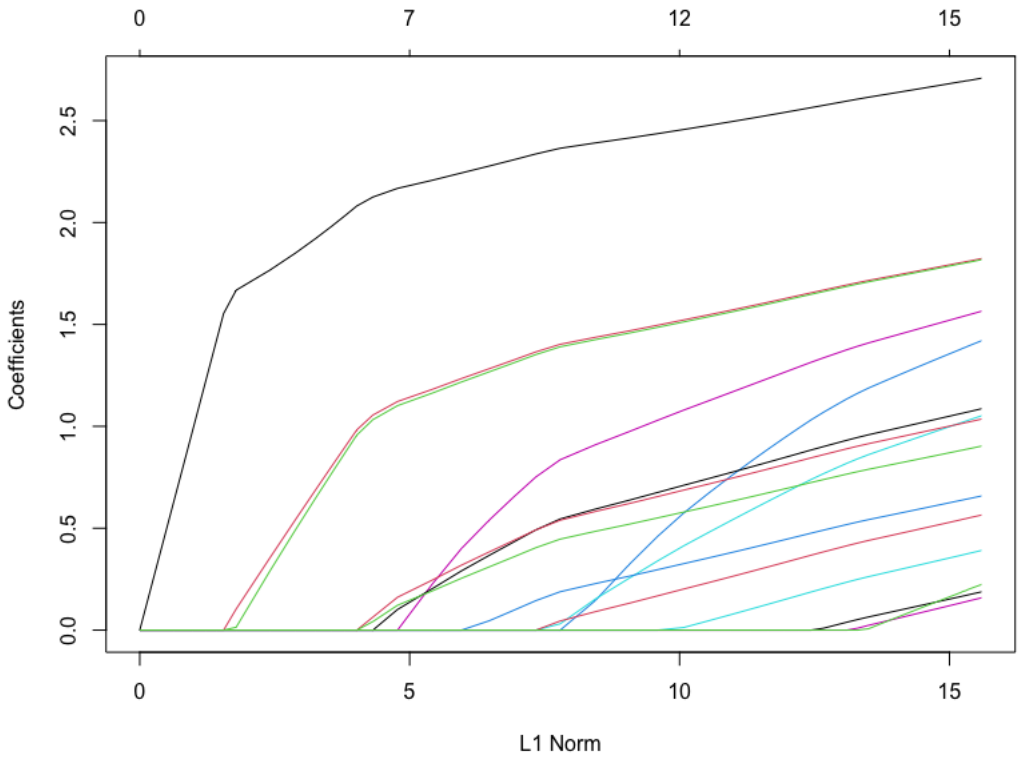


Figure 3.9: GLMNet coefficients for bins only vs the regularization parameter lambda

Figure 3.10 shows how the Poisson deviance changes with the regularization parameter lambda. We select the best lambda to be the one that gives the minimum Poisson Deviance.

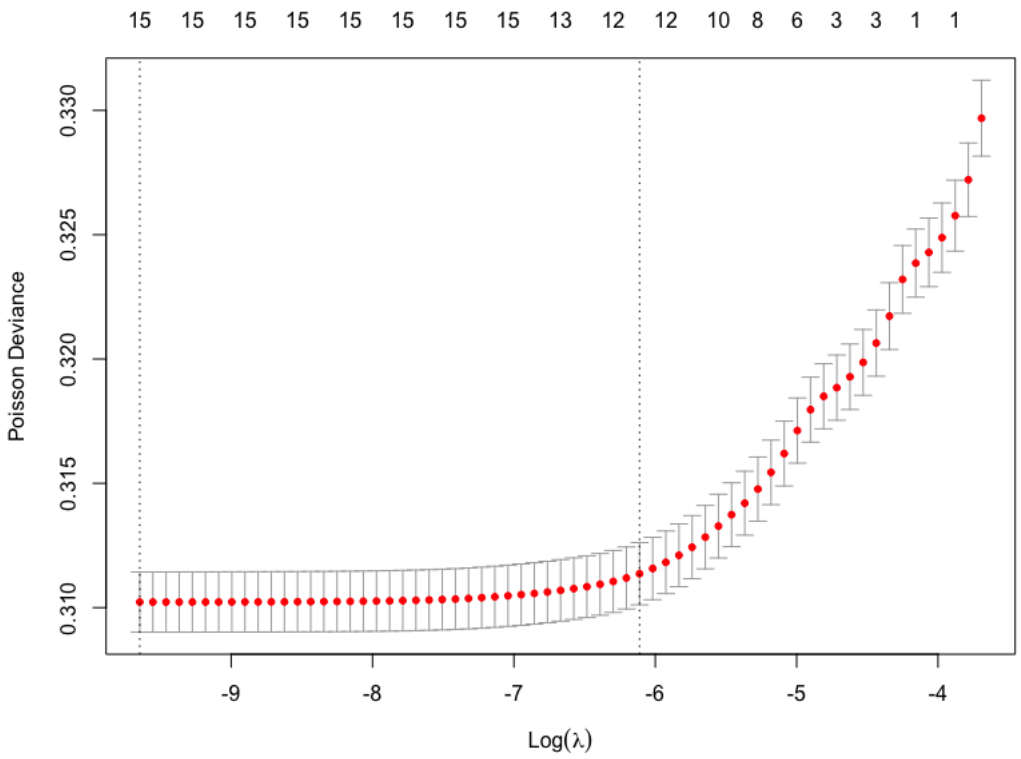


Figure 3.10: Poisson deviance vs log of the regularization parameter lambda

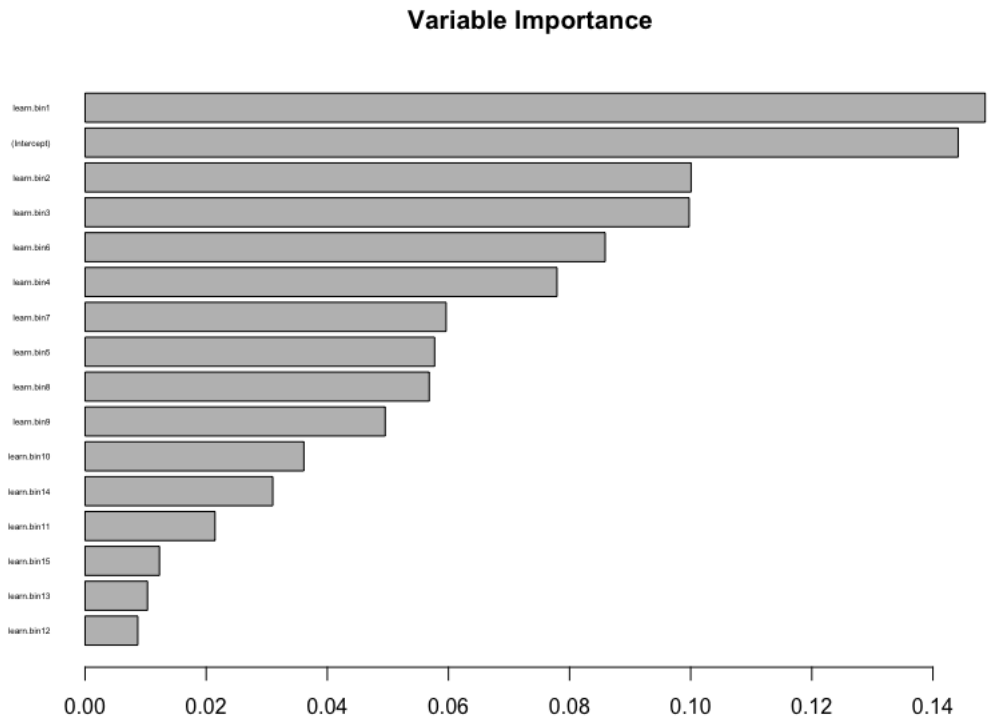


Figure 3.11: Horizontal barplots for variable importance in GLMNet with only bin variables

Analysis

The performance of the suggested hybrid GLMNet model is compared to that of Gradient Boosting (GBM) and Extreme Gradient Boosting (XGBoost) models in *R*. Two types of metrics are used, the Poisson deviance and the mean squared error (MSE), to compare the performances of the three algorithms. However, since the models have been built for claim counts using the Poisson model it is the Poisson Deviance that is minimized by all the models. We see that the lowest Poisson deviation is obtained from the hybrid GLMnet model. Table 3.3 shows the metric results obtained on test data for the three models. The row corresponding to the GLMNet with interaction between the predictors and bins yields the lowest Poisson deviance so far. In the corresponding model within each bin/subspace not only the intercept but also the coefficients for each of the predictor are distinct.

Model	Poisson Deviance	Mean Squared Error
GLM	37.2098	0.05669235
Hybrid GLMNet with predictors and bins	30.5210	0.05714413
Hybrid GLMNet with bins only	30.7533	0.05538684
Hybrid GLMNet with interaction between predictors and bins	30.4478	0.0550325
GBM	31.3043	0.06350775
XGBoost	31.9227	0.05566972

Table 3.3: Poisson deviance and MSE for GBM, XGBoost and Hybrid GLMNet models

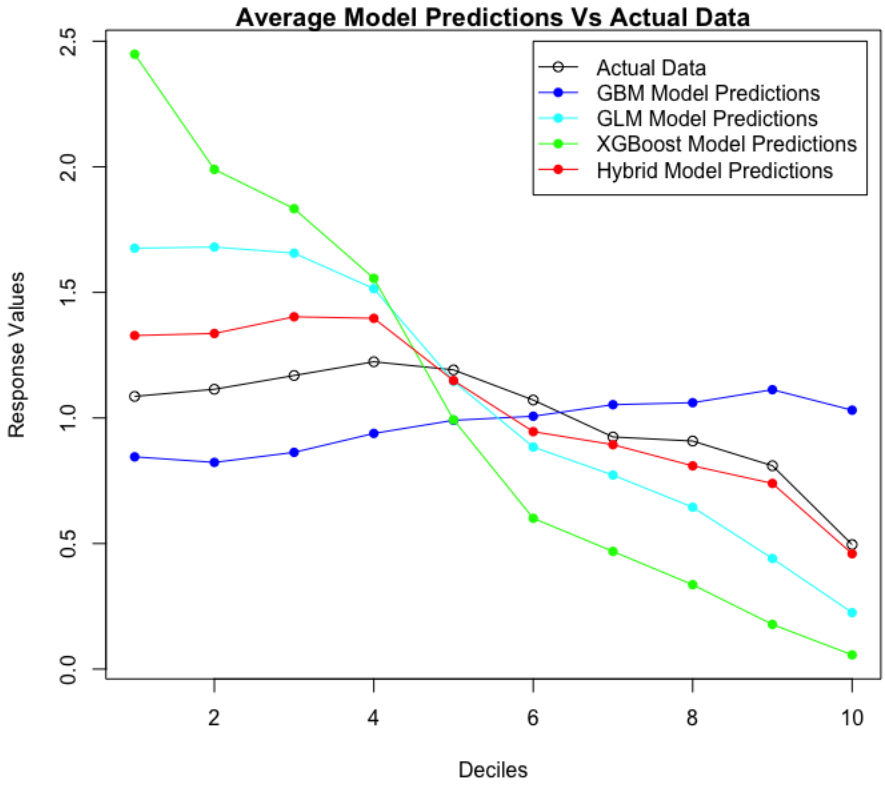


Figure 3.12: The actual data vs the predictions from all the models

The average model predictions vs. actual plots provide a visual representation that enable us to compare how well various predictive models forecast claim counts compared to actual data. They can be used to analyse how well each model's predictions match the actual data. Figure 3.12 shows how the average prediction from the GLM model (green curve) is farthest from the average actual data (black curve) over all the deciles of the predicted response. The red curve corresponding to the hybrid GLMnet model is closest to the actual data over all the deciles. The next closest curve is the GBM (blue curve). It is slightly worse than the hybrid GLMnet curve with respect to proximity

to the real data over the first five deciles. However, the red curve is much better than the GBM curve in the rest of the deciles, making it the best one. The cyan curve corresponds to XGBoost and is farther from the actual data in all the deciles, compared to the GBM and the hybrid model curves.

From the average model predictions vs. actual plot as well as the Table 3.3 it is evident that GBM and the hybrid GLMnet models outperform the XGBoost and the GLM model. The performance of the GBM model is commendable. It has a considerably low Poisson deviance and mean squared error value and most of its data points lie closer to the actual data. This is an indication that its predictions are reliable and align with the actual data. XGBoost comes in third position based on performance. Although its performance is not as commendable as the GBM, it is to be noted that the XGBoost outperformed the traditional GLM model.

It should also be noted that tuning the hyperparapeters of models like GBM and XGBoost is an extensive and extremely time consuming process. A systematic and thorough hyperparameter tuning analysis system could be the subject of another experimentation that might unveil more interesting results. For the data analyzed here, the parameters were tested at length and breadth and the best of all the results were taken into consideration.

Conclusion

The purpose of this research is to develop models to better predict insurance claim counts as a part of insurance premium pricing.

Chapter 1 develops a sampling mechanism and a classification tree is built on the obtained sample to better identify the minority class of non–zero claims. This method can be helpful to identify policyholders that are more susceptible to claim, in insurance portfolios characterized by high imbalance between the zero and non-zero classes.

Chapter 2 builds a regression tree on the training data to partition the insurance portfolio into homogeneous groups along the leaf nodes of the tree. Since a tree is a helpful algorithm in identifying the non–linear relations between the response and the predictor variables, the groupings produced by this non–parametric algorithm are then used to separate the data into homogeneous groups called bins, maximizing the heterogeneity between the groups. These bins are used as an additional predictor variables in a regression context to obtain GLM models with better predictability.

Having developed these two methods the third and final chapter centers over the idea that of mixing the first two chapter proposals. Algorithms like XGBoost and neural networks are being increasingly successful in insurance applications. Nonetheless, the interpretability of these methods is an issue that actuaries need to face. Black–box methods leave less control to the developers.

In terms of interpretation, GLM methods can help to deal with this issue. On the other hand, GLMs are parametric models that impose certain assumptions on the data. Insurance data often fulfill these assumptions, but still GLMs show less predictive accuracy, as compared to the above mentioned methods.

Under these conditions a third model is proposed, constructed as a mixture of the predictions from multiple trees. Each tree is built on a different subsample that vary in terms of the percentage of minority data in them. These tree predictions separate the data into bins, an information that is then used with a GLMNet model, improving to a large extent predictions of claim counts. GLMNet is preferred over GLM so that the model can be penalized and overfitting is controlled. The use of GLMNet performs a regularization, as expected. All the models are evaluated on unseen/test data. The results show that the insurance industry, that has to deal with problematic data, could focus on building hybrid models combining multiple methods rather than relying on a single algorithm.

Bibliography

L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996a.

L. Breiman. Bias, variance, and arcing classifiers. Technical report, Tech.Rep.460, Statistics Department, University of California, Berkeley . . . , 1996b.

L. Breiman. *Classification and Regression Trees*. Routledge, 2017.

M. J. Brockman and T. Wright. Statistical motor rating: making effective use of your data. *Journal of the Institute of Actuaries*, 119(3):457–543, 1992.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

M. Denuit, D. Hainaut, and J. Trufin. *Effective Statistical Learning Methods for Actuaries I: GLMs and Extensions*. Springer, 2019.

M. Denuit, D. Hainaut, and J. Trufin. *Effective Statistical Learning Methods for Actuaries II: Tree-Based Methods and Extensions*. Springer, 2020.

Y. Freund, R. E. Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156. Citeseer, 1996.

L. Guelman. Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications*, 39(3):3659–3667, 2012.

L. Guelman, M. Guillén, and A. M. Pérez-Marín. Random forests for uplift modeling: an insurance customer retention case. In *Modeling and Simulation in Engineering, Economics and Management: International Conference, MS 2012, New Rochelle, NY, USA, May 30-June 1, 2012. Proceedings*, pages 123–133. Springer, 2012.

L. Guelman, M. Guillén, and A. M. Pérez-Marín. Uplift random forests. *Cybernetics and Systems*, 46(3-4):230–248, 2015.

S. Haberman and A. E. Renshaw. Generalized linear models and actuarial science. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 45(4):407–436, 1996.

T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, 2009.

R. Henckaerts, M.-P. Côté, K. Antonio, and R. Verbelen. Boosting insights in insurance tariff plans with tree-based machine learning methods. *North American Actuarial Journal*, 25(2):255–285, 2021.

C. Hu, Z. Quan, and W. F. Chong. Imbalanced learning for insurance using modified loss functions in tree-based models. *Insurance: Mathematics and Economics*, 106:13–32, 2022.

S. A. Klugman, H. H. Panjer, and G. E. Willmot. *Loss models: from data to decisions*, volume 715. John Wiley & Sons, 2012.

O. Lopez, X. Milhaud, and P.-E. Thérond. A tree-based algorithm adapted to microlevel reserving and long development claims. *ASTIN Bulletin: The Journal of the IAA*, 49 ((3)):741–762, 2019.

P. McCullagh and J. Nelder. Binary data. In *Generalized Linear Models*, pages 98–148. Springer, 1989.

J. A. Nelder and R. W. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972.

A. Noll, R. Salzmann, and M. V. Wuthrich. Case study: French motor third-party liability claims. *Preprint, available at SSRN 3164764*, 2020.

W. Olbricht. Tree-based methods: a useful tool for life insurance. *European Actuarial Journal*, 2:129–147, 2012.

A. E. Renshaw. Modelling the claims process in the presence of covariates. *ASTIN Bulletin: The Journal of the IAA*, 24(2):265–285, 1994.

S. Stocksieker, D. Pommeret, and A. Charpentier. Data augmentation for imbalanced regression. *arXiv preprint arXiv:2302.09288*, 2023a.

S. Stocksieker, D. Pommeret, and A. Charpentier. Generalized oversampling for learning from imbalanced datasets and associated theory. *arXiv preprint arXiv:2308.02966*, 2023b.

R. Tibshirani. *Bias, Variance and Prediction Error for Classification Rules*. Citeseer, 1996.

M. V. Wüthrich. Machine learning in individual claims reserving. *Scandinavian Actuarial Journal*, 2018(6):465–480, 2018.

Y. Yang, W. Qian, and H. Zou. Insurance premium prediction via gradient tree-boosted tweedie compound poisson models. *Journal of Business & Economic Statistics*, 36(3): 456–470, 2018.

Appendix A

Toy Data Predictions Table

Age of Driver	Proportion of Class 0	Proportion of Class 1	Predicted class by CART	Prediction by Bayes Classifier
18	0	1	1	1
19	0.5	0.5	1	1
20	0.75	0.25	1	1
21	0.33	0.67	1	1
22	0.4	0.6	1	1
23	1	0	0	1
24	1	0	0	1
25	1	0	0	1
27	1	0	0	1
28	0.5	0.5	1	1
29	0.5	0.5	1	1
30	1	0	0	0
31	1	0	0	0
33	1	0	0	0
34	1	0	0	0
35	1	0	0	0
36	1	0	0	0
37	0.75	0.25	0	1
40	1	0	0	0
41	1	0	0	0
42	1	0	0	0
43	0.33	0.67	1	1
44	0.86	0.14	0	0
45	0.8	0.2	0	0 or 1
46	1	0	1	0
47	0.5	0.5	1	1

Continuation of Table A.1

Age of Driver	Proportion of Class 0	Proportion of Class 1	Predicted class with minsplit = 5	Prediction by Bayes Classifier
50	1	0	0	0
51	0.75	0.25	0	1
52	1	0	0	0
53	1	0	0	0
54	1	0	0	0
55	1	0	0	0
56	1	0	0	0
57	1	0	0	0
58	1	0	0	0
60	1	0	0	0
61	0.5	0.5	Female as 1, Male as 0	1
62	0.67	0.33	0	1
63	1	0	1	0
64	1	0	0	0
65	0.5	0.5	Female as 1 and Male as 0	1

Table A.1: Proportion of classes at each age

Appendix B

Summary of Different GLM Models

```
Call:
glm(formula = ClaimNb ~ VehPower + VehAge + DrivAge + BonusMalus +
    VehBrand + VehGas + Area + Density + Region, family = poisson(),
    data = learn, offset = log(Exposure))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0811  -0.3871  -0.2956  -0.1612   6.8068

Coefficients:
                                 Estimate Std. Error z value Pr(>|z
(Intercept)                   -3.967e+00  1.069e-01 -37.107  < 2e-
VehPower                       1.425e-02  3.082e-03   4.625 3.75e-
VehAge                        -3.923e-02  1.306e-03 -30.041  < 2e-
DrivAge                        6.505e-03  4.401e-04  14.782  < 2e-
BonusMalus                     2.228e-02  3.425e-04  65.064  < 2e-
VehBrandB10                   -2.759e-03  4.108e-02  -0.067   0.94
VehBrandB11                    6.832e-02  4.423e-02   1.544   0.12
VehBrandB12                    1.531e-01  1.949e-02   7.857 3.93e-
VehBrandB13                    4.437e-03  4.625e-02   0.096   0.92
VehBrandB14                   -1.705e-01  8.862e-02  -1.925   0.05
VehBrandB2                    -2.315e-02  1.709e-02  -1.355   0.17
VehBrandB3                    -1.347e-02  2.449e-02  -0.550   0.58
VehBrandB4                    -2.723e-02  3.294e-02  -0.827   0.40
VehBrandB5                     6.482e-02  2.765e-02   2.344   0.01
VehBrandB6                    -6.580e-02  3.199e-02  -2.057   0.03
VehGasRegular                  5.549e-02  1.223e-02   4.537 5.71e-
AreaB                          5.847e-02  2.440e-02   2.397   0.01
AreaC                          9.983e-02  2.023e-02   4.934 8.04e-
AreaD                          1.990e-01  2.169e-02   9.172  < 2e-
```

```
AreaE                                 2.326e-01  2.876e-02   8.088 6.07e-
AreaF                                 1.836e-01  1.006e-01   1.825   0.068
Density                               4.790e-07  4.154e-06   0.115   0.908
RegionAquitaine                      -7.661e-02  1.005e-01  -0.762   0.446
RegionAuvergne                       -2.336e-01  1.246e-01  -1.875   0.060
RegionBasse-Normandie                -6.316e-03  1.059e-01  -0.060   0.952
RegionBourgogne                       1.091e-02  1.078e-01   0.101   0.919
RegionBretagne                        9.818e-02  9.841e-02   0.998   0.318
RegionCentre                          7.422e-02  9.696e-02   0.765   0.444
RegionChampagne-Ardenne               1.675e-01  1.297e-01   1.291   0.196
RegionCorse                           6.613e-02  1.206e-01   0.548   0.583
RegionFranche-Comte                  -1.187e-01  1.760e-01  -0.675   0.500
RegionHaute-Normandie                -6.230e-02  1.154e-01  -0.540   0.589
RegionIle-de-France                  -3.806e-03  9.826e-02  -0.039   0.969
RegionLanguedoc-Roussillon           -3.862e-02  1.002e-01  -0.385   0.700
RegionLimousin                        1.806e-01  1.192e-01   1.516   0.129
RegionMidi-Pyrenees                  -1.124e-01  1.051e-01  -1.070   0.284
RegionNord-Pas-de-Calais             -1.450e-01  9.948e-02  -1.458   0.144
RegionPays-de-la-Loire                2.324e-03  9.910e-02   0.023   0.981
RegionPicardie                        5.304e-02  1.101e-01   0.482   0.629
RegionPoitou-Charentes               -4.823e-02  1.026e-01  -0.470   0.638
RegionProvence-Alpes-Cotes-D'Azur    -4.154e-03  9.748e-02  -0.043   0.966
RegionRhone-Alpes                     8.530e-02  9.718e-02   0.878   0.380
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 179154  on 542409  degrees of freedom
Residual deviance: 173424  on 542368  degrees of freedom
AIC: 228957

Number of Fisher Scoring iterations: 6
```

$$formula = "ClaimNb \sim VehAge + BonusMalus + VehBrand + DrivAge + VehGas + Region + VehPower + Density + Area + Bin"$$

```
Call:
glm(formula = as.formula(formula), family = poisson(), data = learn,
    offset = log(Exposure))

Deviance Residuals:
```

```
    Min       1Q   Median       3Q      Max
-1.5177  -0.3734  -0.2823  -0.1586   6.8368

Coefficients:
                                  Estimate Std. Error z value Pr(>|z
(Intercept)                     -3.472e+00  1.104e-01 -31.453  < 2e-
VehAge                          -2.415e-02  1.358e-03 -17.781  < 2e-
BonusMalus                       7.928e-03  6.339e-04  12.508  < 2e-
VehBrandB10                     -4.642e-02  4.110e-02  -1.129   0.258
VehBrandB11                      5.137e-02  4.420e-02   1.162   0.245
VehBrandB12                     -2.668e-01  2.287e-02 -11.665  < 2e-
VehBrandB13                     -3.509e-03  4.624e-02  -0.076   0.939
VehBrandB14                     -1.714e-01  8.862e-02  -1.934   0.053
VehBrandB2                      -2.629e-02  1.709e-02  -1.539   0.123
VehBrandB3                      -9.918e-03  2.449e-02  -0.405   0.685
VehBrandB4                      -7.843e-03  3.296e-02  -0.238   0.811
VehBrandB5                       6.466e-02  2.765e-02   2.338   0.019
VehBrandB6                      -4.399e-02  3.200e-02  -1.375   0.169
DrivAge                          7.523e-03  4.505e-04  16.701  < 2e-
VehGasRegular                   -5.555e-02  1.299e-02  -4.278 1.89e-
RegionAquitaine                 -7.014e-02  1.006e-01  -0.697   0.485
RegionAuvergne                  -2.334e-01  1.246e-01  -1.873   0.061
RegionBasse-Normandie            3.629e-02  1.059e-01   0.343   0.731
RegionBourgogne                  4.463e-02  1.078e-01   0.414   0.678
RegionBretagne                   1.314e-01  9.843e-02   1.335   0.181
RegionCentre                     9.795e-02  9.699e-02   1.010   0.312
RegionChampagne-Ardenne          1.667e-01  1.297e-01   1.285   0.199
RegionCorse                      9.069e-02  1.206e-01   0.752   0.452
RegionFranche-Comte             -7.411e-02  1.760e-01  -0.421   0.673
RegionHaute-Normandie           -3.876e-02  1.154e-01  -0.336   0.736
RegionIle-de-France             -1.305e-02  9.825e-02  -0.133   0.894
RegionLanguedoc-Roussillon      -3.106e-02  1.002e-01  -0.310   0.756
RegionLimousin                   2.047e-01  1.192e-01   1.718   0.085
RegionMidi-Pyrenees             -8.377e-02  1.051e-01  -0.797   0.425
RegionNord-Pas-de-Calais        -1.107e-01  9.949e-02  -1.112   0.266
RegionPays-de-la-Loire           2.380e-02  9.912e-02   0.240   0.810
RegionPicardie                   7.210e-02  1.101e-01   0.655   0.512
RegionPoitou-Charentes          -3.842e-02  1.027e-01  -0.374   0.708
RegionProvence-Alpes-Cotes-D'Azur 1.384e-03  9.749e-02   0.014   0.988
RegionRhone-Alpes                1.093e-01  9.720e-02   1.125   0.260
VehPower                         2.001e-02  3.056e-03   6.546 5.92e-
Density                         -1.219e-06  4.148e-06  -0.294   0.768
AreaB                            6.207e-02  2.440e-02   2.544   0.011
```

```
AreaC                                1.061e-01  2.023e-02   5.244 1.57e-
AreaD                                2.025e-01  2.170e-02   9.333  < 2e-
AreaE                                2.327e-01  2.874e-02   8.097 5.62e-
AreaF                                2.032e-01  1.005e-01   2.021   0.04
Bin1                                 1.220e+00  3.880e-02  31.434  < 2e-
Bin3                                 4.673e-01  2.044e-02  22.868  < 2e-
Bin4                                 3.106e-01  4.016e-02   7.734 1.04e-
Bin5                                 2.254e+00  2.829e-02  79.686  < 2e-
Bin6                                 1.153e+00  4.197e-02  27.469  < 2e-
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 179154  on 542409  degrees of freedom
Residual deviance: 167811  on 542363  degrees of freedom
AIC: 223354

Number of Fisher Scoring iterations: 6
```

$$formula = "ClaimNb \sim VehAge + BonusMalus + VehBrand + DrivAge + VehGas + Region + VehPower + Density + Area + Bin + Bin * VehAge + Bin * BonusMalus + Bin * VehBrand + Bin * DrivAge + Bin * VehGas + Bin * Region + Bin * VehPower + Bin * Density + Bin * Area"$$

```
Call:
glm(formula = as.formula(formula), family = poisson(), data = learn,
    offset = log(Exposure))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6037  -0.3696  -0.2807  -0.1582   6.8550

Coefficients: (26 not defined because of singularities)
                                      Estimate Std. Error z value Pr
(Intercept)                         -4.712e+00  3.126e-01 -15.073  <
VehAge                              -2.898e-02  1.787e-03 -16.217  <
BonusMalus                           3.153e-02  5.192e-03   6.073 1.
VehBrandB10                         -6.569e-02  5.328e-02  -1.233 0.
VehBrandB11                         -3.321e-02  6.194e-02  -0.536 0.
VehBrandB12                         -1.838e-01  3.000e-02  -6.127 8.
VehBrandB13                         -4.877e-02  6.231e-02  -0.783 0.
```

VehBrandB14	-2.530e-01	1.191e-01	-2.125	0
VehBrandB2	-2.448e-02	2.270e-02	-1.079	0
VehBrandB3	-7.135e-02	3.558e-02	-2.005	0
VehBrandB4	8.158e-03	4.617e-02	0.177	0
VehBrandB5	6.472e-02	3.810e-02	1.698	0
VehBrandB6	-9.289e-02	4.510e-02	-2.060	0
DrivAge	7.114e-03	6.710e-04	10.602	<
VehGasRegular	-4.837e-02	1.761e-02	-2.747	0
RegionAquitaine	-2.386e-02	1.528e-01	-0.156	0
RegionAuvergne	-9.504e-02	1.901e-01	-0.500	0
RegionBasse-Normandie	1.392e-01	1.579e-01	0.882	0
RegionBourgogne	8.627e-02	1.643e-01	0.525	0
RegionBretagne	2.111e-01	1.488e-01	1.419	0
RegionCentre	1.688e-01	1.473e-01	1.146	0
RegionChampagne-Ardenne	9.269e-02	2.173e-01	0.427	0
RegionCorse	2.997e-01	1.895e-01	1.581	0
RegionFranche-Comte	4.995e-02	2.775e-01	0.180	0
RegionHaute-Normandie	2.695e-02	1.740e-01	0.155	0
RegionIle-de-France	9.769e-03	1.509e-01	0.065	0
RegionLanguedoc-Roussillon	4.078e-02	1.528e-01	0.267	0
RegionLimousin	-1.280e-02	1.858e-01	-0.069	0
RegionMidi-Pyrenees	-1.177e-01	1.620e-01	-0.727	0
RegionNord-Pas-de-Calais	-1.275e-01	1.516e-01	-0.841	0
RegionPays-de-la-Loire	2.322e-03	1.503e-01	0.015	0
RegionPicardie	1.028e-01	1.736e-01	0.592	0
RegionPoitou-Charentes	-6.512e-02	1.555e-01	-0.419	0
RegionProvence-Alpes-Cotes-D'Azur	7.104e-02	1.484e-01	0.479	0
RegionRhone-Alpes	1.481e-01	1.479e-01	1.002	0
VehPower	2.362e-02	4.518e-03	5.228	1
Density	4.478e-06	7.073e-06	0.633	0
AreaB	3.565e-02	3.264e-02	1.092	0
AreaC	1.290e-01	2.703e-02	4.772	1
AreaD	2.176e-01	3.044e-02	7.149	8
AreaE	2.883e-01	4.417e-02	6.527	6
AreaF	1.956e-01	1.736e-01	1.127	0
Bin1	3.012e+00	5.966e-01	5.050	4
Bin3	8.250e-01	4.016e-01	2.054	0
Bin4	3.122e-02	1.105e+00	0.028	0
Bin5	5.996e+00	4.184e-01	14.332	<
Bin6	2.166e+00	4.571e-01	4.738	2
VehAge:Bin1	NA	NA	NA	
VehAge:Bin3	7.574e-03	2.988e-03	2.535	0
VehAge:Bin4	NA	NA	NA	

VehAge:Bin5	NA	NA	NA	
VehAge:Bin6	2.118e-02	4.969e-03	4.262	2
BonusMalus:Bin1	-2.822e-02	5.801e-03	-4.864	1
BonusMalus:Bin3	-2.268e-02	5.288e-03	-4.289	1
BonusMalus:Bin4	-1.267e-02	5.932e-03	-2.136	0
BonusMalus:Bin5	-3.460e-02	5.376e-03	-6.436	1
BonusMalus:Bin6	-1.817e-02	5.395e-03	-3.369	0
VehBrandB10:Bin1	NA	NA	NA	
VehBrandB11:Bin1	NA	NA	NA	
VehBrandB12:Bin1	NA	NA	NA	
VehBrandB13:Bin1	NA	NA	NA	
VehBrandB14:Bin1	NA	NA	NA	
VehBrandB2:Bin1	NA	NA	NA	
VehBrandB3:Bin1	NA	NA	NA	
VehBrandB4:Bin1	NA	NA	NA	
VehBrandB5:Bin1	NA	NA	NA	
VehBrandB6:Bin1	NA	NA	NA	
VehBrandB10:Bin3	-5.413e-02	9.309e-02	-0.581	0
VehBrandB11:Bin3	1.225e-01	9.950e-02	1.231	0
VehBrandB12:Bin3	-1.773e-01	5.048e-02	-3.512	0
VehBrandB13:Bin3	4.027e-02	1.020e-01	0.395	0
VehBrandB14:Bin3	4.597e-02	1.977e-01	0.232	0
VehBrandB2:Bin3	-1.334e-02	3.810e-02	-0.350	0
VehBrandB3:Bin3	1.111e-01	5.390e-02	2.061	0
VehBrandB4:Bin3	1.483e-02	7.134e-02	0.208	0
VehBrandB5:Bin3	1.857e-02	6.038e-02	0.308	0
VehBrandB6:Bin3	1.126e-01	6.997e-02	1.609	0
VehBrandB10:Bin4	1.154e-01	2.063e-01	0.560	0
VehBrandB11:Bin4	-9.401e-02	2.409e-01	-0.390	0
VehBrandB12:Bin4	NA	NA	NA	
VehBrandB13:Bin4	5.418e-01	2.452e-01	2.209	0
VehBrandB14:Bin4	1.019e+00	3.670e-01	2.777	0
VehBrandB2:Bin4	7.444e-02	1.047e-01	0.711	0
VehBrandB3:Bin4	3.000e-01	1.289e-01	2.327	0
VehBrandB4:Bin4	3.326e-01	1.653e-01	2.012	0
VehBrandB5:Bin4	-2.068e-02	1.920e-01	-0.108	0
VehBrandB6:Bin4	3.009e-01	1.824e-01	1.650	0
VehBrandB10:Bin5	NA	NA	NA	
VehBrandB11:Bin5	NA	NA	NA	
VehBrandB12:Bin5	NA	NA	NA	
VehBrandB13:Bin5	NA	NA	NA	
VehBrandB14:Bin5	NA	NA	NA	
VehBrandB2:Bin5	NA	NA	NA	

VehBrandB3:Bin5	NA	NA	NA	
VehBrandB4:Bin5	NA	NA	NA	
VehBrandB5:Bin5	NA	NA	NA	
VehBrandB6:Bin5	NA	NA	NA	
VehBrandB10:Bin6	1.621e-02	1.827e-01	0.089	0
VehBrandB11:Bin6	2.170e-01	1.422e-01	1.526	0
VehBrandB12:Bin6	-4.445e-01	9.155e-02	-4.855	1
VehBrandB13:Bin6	1.002e-01	1.776e-01	0.564	0
VehBrandB14:Bin6	1.979e-01	3.759e-01	0.526	0
VehBrandB2:Bin6	1.039e-02	6.099e-02	0.170	0
VehBrandB3:Bin6	1.108e-01	8.252e-02	1.342	0
VehBrandB4:Bin6	-3.137e-01	1.292e-01	-2.428	0
VehBrandB5:Bin6	-3.547e-02	9.527e-02	-0.372	0
VehBrandB6:Bin6	7.427e-02	1.099e-01	0.676	0
DrivAge:Bin1	-1.883e-03	2.957e-03	-0.637	0
DrivAge:Bin3	1.074e-02	1.039e-03	10.336	<
DrivAge:Bin4	-1.163e-03	2.987e-03	-0.389	0
DrivAge:Bin5	-1.908e-02	1.776e-03	-10.746	<
DrivAge:Bin6	-1.282e-02	1.875e-03	-6.835	8
VehGasRegular:Bin1	NA	NA	NA	
VehGasRegular:Bin3	-4.735e-02	2.870e-02	-1.650	0
VehGasRegular:Bin4	-1.037e-01	8.291e-02	-1.250	0
VehGasRegular:Bin5	NA	NA	NA	
VehGasRegular:Bin6	8.310e-02	4.738e-02	1.754	0
RegionAquitaine:Bin1	-4.367e-01	4.569e-01	-0.956	0
RegionAuvergne:Bin1	-1.217e+00	5.616e-01	-2.167	0
RegionBasse-Normandie:Bin1	-5.688e-01	5.408e-01	-1.052	0
RegionBourgogne:Bin1	-5.754e-01	4.981e-01	-1.155	0
RegionBretagne:Bin1	-1.060e+00	5.222e-01	-2.030	0
RegionCentre:Bin1	-1.349e+00	5.112e-01	-2.639	0
RegionChampagne-Ardenne:Bin1	-5.529e-01	5.316e-01	-1.040	0
RegionCorse:Bin1	-1.339e+00	5.515e-01	-2.429	0
RegionFranche-Comte:Bin1	5.277e-01	6.227e-01	0.848	0
RegionHaute-Normandie:Bin1	-1.033e+00	6.039e-01	-1.710	0
RegionIle-de-France:Bin1	-5.673e-01	4.455e-01	-1.273	0
RegionLanguedoc-Roussillon:Bin1	-2.238e-01	4.477e-01	-0.500	0
RegionLimousin:Bin1	9.184e-02	5.159e-01	0.178	0
RegionMidi-Pyrenees:Bin1	-6.662e-01	4.729e-01	-1.409	0
RegionNord-Pas-de-Calais:Bin1	-4.647e-01	4.540e-01	-1.024	0
RegionPays-de-la-Loire:Bin1	1.484e-01	4.592e-01	0.323	0
RegionPicardie:Bin1	-4.173e-01	4.875e-01	-0.856	0
RegionPoitou-Charentes:Bin1	1.093e-01	4.818e-01	0.227	0
RegionProvence-Alpes-Cotes-D'Azur:Bin1	-5.952e-01	4.434e-01	-1.342	0

RegionRhone-Alpes:Bin1	-5.943e-01	4.465e-01	-1.331	0
RegionAquitaine:Bin3	2.912e-01	2.815e-01	1.035	0
RegionAuvergne:Bin3	3.078e-01	3.351e-01	0.918	0
RegionBasse-Normandie:Bin3	1.256e-01	2.915e-01	0.431	0
RegionBourgogne:Bin3	3.055e-01	2.964e-01	1.030	0
RegionBretagne:Bin3	2.281e-01	2.769e-01	0.824	0
RegionCentre:Bin3	2.102e-01	2.741e-01	0.767	0
RegionChampagne-Ardenne:Bin3	5.152e-01	3.576e-01	1.441	0
RegionCorse:Bin3	-4.898e-02	3.350e-01	-0.146	0
RegionFranche-Comte:Bin3	3.684e-02	4.618e-01	0.080	0
RegionHaute-Normandie:Bin3	3.242e-01	3.090e-01	1.049	0
RegionIle-de-France:Bin3	3.313e-01	2.776e-01	1.194	0
RegionLanguedoc-Roussillon:Bin3	1.761e-02	2.823e-01	0.062	0
RegionLimousin:Bin3	7.118e-01	3.202e-01	2.223	0
RegionMidi-Pyrenees:Bin3	2.850e-01	2.936e-01	0.971	0
RegionNord-Pas-de-Calais:Bin3	4.384e-01	2.793e-01	1.570	0
RegionPays-de-la-Loire:Bin3	3.526e-01	2.782e-01	1.267	0
RegionPicardie:Bin3	3.440e-01	3.019e-01	1.140	0
RegionPoitou-Charentes:Bin3	3.621e-01	2.849e-01	1.271	0
RegionProvence-Alpes-Cotes-D'Azur:Bin3	1.875e-01	2.755e-01	0.680	0
RegionRhone-Alpes:Bin3	2.350e-01	2.746e-01	0.856	0
RegionAquitaine:Bin4	4.727e-01	1.040e+00	0.455	0
RegionAuvergne:Bin4	9.750e-01	1.135e+00	0.859	0
RegionBasse-Normandie:Bin4	1.151e+00	1.038e+00	1.108	0
RegionBourgogne:Bin4	6.431e-01	1.068e+00	0.602	0
RegionBretagne:Bin4	8.690e-01	1.021e+00	0.851	0
RegionCentre:Bin4	1.063e+00	1.015e+00	1.047	0
RegionChampagne-Ardenne:Bin4	2.673e+00	1.066e+00	2.508	0
RegionCorse:Bin4	1.377e+00	1.173e+00	1.174	0
RegionFranche-Comte:Bin4	1.152e+00	1.443e+00	0.798	0
RegionHaute-Normandie:Bin4	8.069e-01	1.076e+00	0.750	0
RegionIle-de-France:Bin4	1.115e+00	1.023e+00	1.090	0
RegionLanguedoc-Roussillon:Bin4	5.292e-01	1.031e+00	0.513	0
RegionLimousin:Bin4	9.234e-01	1.241e+00	0.744	0
RegionMidi-Pyrenees:Bin4	3.306e-01	1.074e+00	0.308	0
RegionNord-Pas-de-Calais:Bin4	7.911e-01	1.029e+00	0.769	0
RegionPays-de-la-Loire:Bin4	9.252e-01	1.028e+00	0.900	0
RegionPicardie:Bin4	8.656e-01	1.076e+00	0.804	0
RegionPoitou-Charentes:Bin4	7.251e-01	1.048e+00	0.692	0
RegionProvence-Alpes-Cotes-D'Azur:Bin4	6.523e-01	1.017e+00	0.641	0
RegionRhone-Alpes:Bin4	8.170e-01	1.017e+00	0.803	0
RegionAquitaine:Bin5	-3.232e-01	2.808e-01	-1.151	0
RegionAuvergne:Bin5	-6.794e-01	3.572e-01	-1.902	0

```
RegionBasse-Normandie:Bin5              -1.243e+00  4.051e-01  -3.070 0
RegionBourgogne:Bin5                    -3.457e-01  3.089e-01  -1.119 0
RegionBretagne:Bin5                     -5.633e-01  3.061e-01  -1.840 0
RegionCentre:Bin5                       -1.112e+00  3.341e-01  -3.328 0
RegionChampagne-Ardenne:Bin5            -3.445e-01  3.659e-01  -0.942 0
RegionCorse:Bin5                        -4.650e-01  3.202e-01  -1.452 0
RegionFranche-Comte:Bin5                -1.037e+00  6.126e-01  -1.693 0
RegionHaute-Normandie:Bin5              -2.359e-01  3.461e-01  -0.682 0
RegionIle-de-France:Bin5                -1.837e-01  2.698e-01  -0.681 0
RegionLanguedoc-Roussillon:Bin5         -4.134e-02  2.751e-01  -0.150 0
RegionLimousin:Bin5                      9.574e-02  3.442e-01   0.278 0
RegionMidi-Pyrenees:Bin5                 1.680e-01  2.851e-01   0.589 0
RegionNord-Pas-de-Calais:Bin5           -1.120e-01  2.770e-01  -0.405 0
RegionPays-de-la-Loire:Bin5             -1.064e-01  2.856e-01  -0.372 0
RegionPicardie:Bin5                     -3.841e-01  3.181e-01  -1.207 0
RegionPoitou-Charentes:Bin5             -3.802e-01  3.234e-01  -1.175 0
RegionProvence-Alpes-Cotes-D'Azur:Bin5 -1.742e-01  2.683e-01  -0.649 0
RegionRhone-Alpes:Bin5                  -3.229e-01  2.706e-01  -1.193 0
RegionAquitaine:Bin6                    -4.621e-01  3.236e-01  -1.428 0
RegionAuvergne:Bin6                     -9.638e-02  4.337e-01  -0.222 0
RegionBasse-Normandie:Bin6              -3.520e-01  3.446e-01  -1.022 0
 [ reached getOption("max.print") -- omitted 52 rows ]
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 179154  on 542409  degrees of freedom
Residual deviance: 166649  on 542184  degrees of freedom
AIC: 222550

Number of Fisher Scoring iterations: 6
```