Car Insurance Claims Classification

Hamed Vaheb
hamed.vaheb.001@student.uni.lu

January 16, 2023

Abstract

This report describes the project of analysis and prediction of the car insurance claims dataset. The dataset includes historical data of the policyholders (e.g., age, gender, vehicle details, etc.), among which the variable of interest is the outcome of insurance, indicating whether a customer's claim is approved or not. The `aiinsurance` R package [1] is developed to make this work reproducible, accessible, and equipped with advanced features, e.g., a pipeline that performs the main processes of this work in a single command, and an interactive app that displays the performance of the classification models, logistic regression and random forest, which are implemented using the package's functions and their purpose was to classify the outcomes based on solely the historical data. The results indicate promising prediction performance. In addition to classification, informative insights from the dataset and models have been drawn, e.g., statistical signifcance of the variables playing more important role in prediction.

1 Introduction

Rapid advances in artificial intelligence (AI) and machine learning are creating products and services with the potential not only to change the environment in which actuaries operate but also to provide new opportunities within actuarial science [2].

The use of statistical learning models has been a common practice in actuarial science since the 1980s. It was not long after that time when the field adopted classical models, e.g., linear models and generalized linear models (GLMs). While actuaries use GLMs frequently in practice, it was in the past few years that the use of AI and machine learning, and hence more modern models garnered significant attention in the field [3].

The goal of this work is to predict status of policyholders' claims. The status lies at the `outcome` column of the car insruance dataset, indicating whether a customer has claimed his loan or not.

A classical linear model and a modern nonlinear model is used and compared. The former model is logistic regression, which is an example of GLMs, accommodated to classification setting, i.e., for predicting discrete classes, in this work's case, status of claims. The latter model is random forest, which is a nonlinear tree-based model.

The remainder of this work is organized as the following: In section 2, main concepts included in the work are explained. In 2.1, the concept of car insurance is introduced followed by an elaboration on claims and how they interact among an insurer and a policyholder. Subsequently, the two machine learning models, logistic regression 2.2, and random forest 2.3 are briefly explained. As a contribution of this work, the `aiinsurance` package [1], introduced in 3, is developed, which automates many of the classification tasks, and uses advanced features, e.g., a pipeline connecting this work's processes and interactive app. In 4, data is explored, with stress on the effect of categorical variables as well as missing values, both of which will be treated in 5, which contains all the preprocessing steps. These steps are required before fitting the two classification models used, random forest and logit, with their results presented in 6. Finally, the models' performance are evaluated in 7.

2 Preliminary Concepts

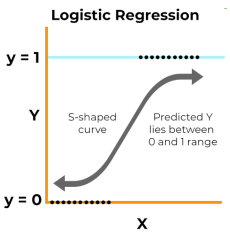
2.1 Car Insurance Claims

Car insurance is a type of insurance policy that financially protects drivers in the event of an accident or theft. There are several types of car insurance coverage, including liability insurance, collision insurance, comprehensive insurance, and personal injury protection.

An insurance claim is a formal request by a policyholder to an insurance company for coverage or compensation for a covered loss or policy event, in this work's case, a car accident. The insurance company either accepts or rejects the claim. If it is approved, the insurance company will issue payment to the insured or an approved interested party on behalf of the insured.

Predicting the outcome of claims can be utilized to better understand the customer strata and incorporate the findings throughout the insurance policy enrollment (including the underwriting and approval or rejection stages), triage claims and automate where possible, gradually obviating the need for human interaction, and optimize the entire insurance policy enrollment process flow [4].

2.2 Model: Logistic Regression

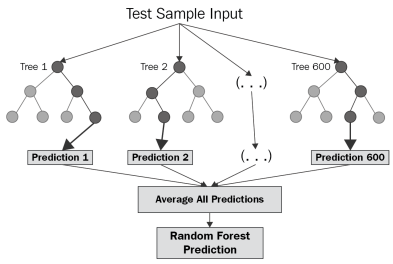


Logistic regression (logit), which is a type of GLM, accommodated to classification setting (for predicting discrete classes), uses the logistic function to model the probability of the binary outcome by creating a model that takes in the input variable (e.g., policyholder's information) and produces a probability that a "outcome" (the variable we aim to predict) is 1. The "outcome" in this work is whether a customer's claim is approved or not. The logistic function is defined as the following:

$$\frac{1}{1+e^{-x}}$$

This probability can then be used to make a prediction about the outcome. For instance, if the probability that the policyholder's claim will be approved is greater than a certain threshold (e.g., 0.5), we predict that the claim will be approved. In another words, the goal of logit is to find the best set of coefficients for a set of independent variables that maximizes the likelihood (measuring how well parameters of a model fit the data) of the observed data.

2.3 Model: Random Forest



An ensemble learning model is a model that is constructed from multiple models, to obtain combined results, expected to be improved compared to any of the constituent models. The random forest is an ensemble model built from decision trees, i.e., flowchart-like tree structures, wherein each internal node represents a "test" on a variable (e.g., vehicle type), each branch represents the outcome of the test, and each leaf node represents a class label (e.g., 1 for claim approval and 0 for claim rejection). The intuition behind a decision tree is to recursively split the data into subsets based on the values of the input variables, such that the subsets are as "pure" as possible in terms of their class labels, which means the most amount of information is extracted from them. The less pure a subset is, the more the data belongs to the same class, and the more pure it is, the more data is evenly split among all classes. The goal is to create a tree that can accurately classify new examples by traversing the tree from the root to a leaf node, making decisions at each internal node based on the values of the input variables.

As a single random tree might not be able to capture the proper inherent complexity of a data, and may either overfit (become too complex and remembers data rather than learning from it) or underfit (become too simple and hence unable to learn the inherent complicated patterns in the data), by training multiple decision trees and combining their

predictions by taking a majority vote, a random forest is able to capture a more robust and accurate representation of the data. The randomness in the random forest comes from randomly selecting subsets of the data to train each decision tree, and randomly selecting subsets of input variables to consider at each split point in the decision tree. This helps to decorrelate the trees, i.e., reducing the correlation or dependence between the trees, consequently, make the model more robust to overfitting, i.e., becoming too complex and remembers data rather than learning from it.

3 `aiinsurance` Package

AI INSURANCE

The `aiinsurance` R package [1] is developed to make this work reproducible, accessible, and equipped with advanced features.

Instructions on how to install and use the package's functions and features is provided in the README part of package's Github repository.

Noteworthy features of the package are explained in the following:

1. Multiple functions that facilitated and automated processes in 5 and 5.1. The functions are well documented and numerous unit tests are conducted on them. This package was aimed to be developed as generalizable as possible, so that hopefully we little effort, it can be used for other datasets with classification setting.

2. Both raw dataset and processed train and tests (explained in 5.4) are incorporated in the package.

3. A pipeline that connects the main processes of this work and is executable with a single command. The pipeline takes the processed train and test datsets and outputs the evaluation plots in 7.

4. An interactive shiny app that after performing the same tasks in pipeline, displays the evaluation plots in 7 based on the type of the model and the type of the plot that user prefers

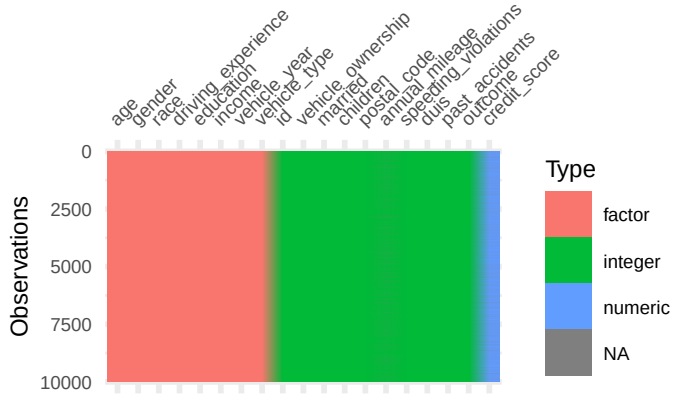
4 Exploratory Data Analysis (EDA)

An overview of the car insurance dataset is provided in the following:

```
## Rows: 10,000
## Columns: 19
## $ id                 <int> 569520, 750365, 199901, 478866, 731664, 877557, 93~
## $ age                <chr> "65+", "16-25", "16-25", "16-25", "26-39", "40-64"~
## $ gender             <chr> "female", "male", "female", "male", "male", "femal~
## $ race               <chr> "majority", "majority", "majority", "majority", "m~
## $ driving_experience <chr> "0-9y", "0-9y", "0-9y", "0-9y", "10-19y", "20-29y"~
## $ education          <chr> "high school", "none", "high school", "university"~
## $ income             <chr> "upper class", "poverty", "working class", "workin~
## $ credit_score       <dbl> 0.6290273, 0.3577571, 0.4931458, 0.2060129, 0.3883~
## $ vehicle_ownership  <int> 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1,~
## $ vehicle_year       <chr> "after 2015", "before 2015", "before 2015", "befor~
## $ married            <int> 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1,~
## $ children           <int> 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,~
## $ postal_code        <int> 10238, 10238, 10238, 32765, 32765, 10238, 10238, 1~
## $ annual_mileage     <int> 12000, 16000, 11000, 11000, 12000, 13000, 13000, 1~
## $ vehicle_type       <chr> "sedan", "sedan", "sedan", "sedan", "sedan", "seda~
## $ speeding_violations <int> 0, 0, 0, 0, 2, 3, 7, 0, 0, 0, 6, 4, 4, 0, 0, 0, 10~
## $ duis               <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 1, 0, 2, 0, 2,~
## $ past_accidents     <int> 0, 0, 0, 0, 1, 3, 3, 0, 0, 0, 7, 0, 2, 0, 1, 0, 1,~
## $ outcome            <int> 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,~
```

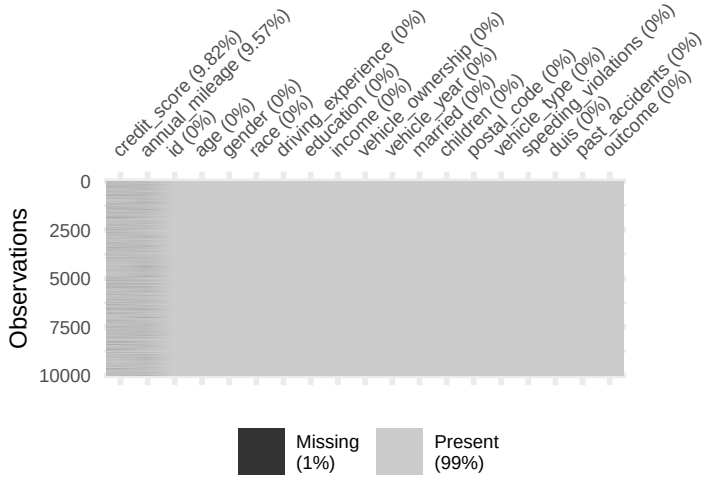
The dataset has 19 variables (columns) and 10,000 observations (rows).

The following plot visualizes proportion of data types within the dataset.



Variables with `factor` types (denoted red in the plot) represent categorical variables containing classes as characters, e.g., `gender` variable contains characters `male` or `female`, and `age` variable contains four age groups, `16-25`, `26-39`, `40-64`, `65+`, etc. However, there are some categorical variables in the integer type as well, e.g., `married` which has values {1,0}, with 1 corresponding to being married or 0 to not being married. The categorical columns, should be treated and mapped to proper numeric representations, elaborated in 5.3.

In the following, the proportion and location of missing values are visualized.



Evidenced by the plot, the columns containing missing values are the following:

- `credit_score`
- `annual_mileage`

The reason that the dataset is better not to contain missing values are explained in the following:

1. Many machine learning algorithms cannot handle missing values.
2. Missing values introduce bias or lead to inaccurate conclusions if the missing data is not missing at random.
3. Missing values increase the variance of the estimate obtained from the data, which means the model's predictions will be more distant from actual values, caused by overfit (i.e., model becoming too complex and remembers data rather than learning from it).

Treating missing values is expalined in detail in 5.1.

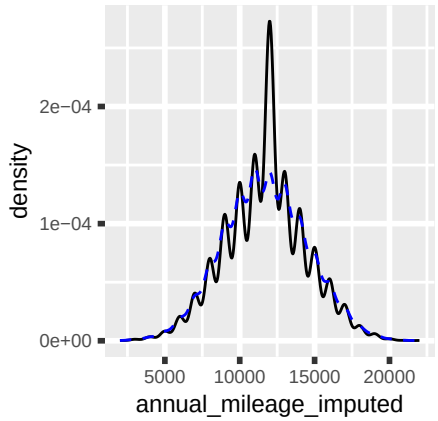
5 Preprocessing

5.1 Missing values

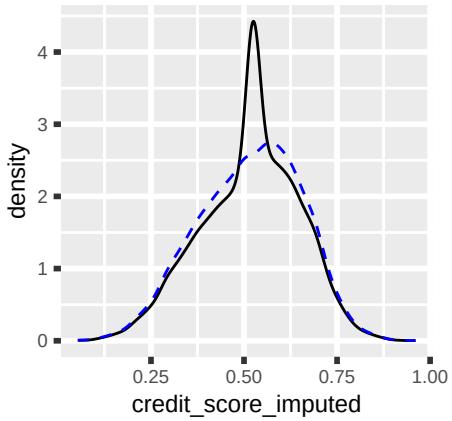
To proceed the processing and modeling processes, missing values of each variable are imputed with median of that particular variable. The reason that median is chosen as the statistical quantity for filling the missing values is that median is robust to outliers, i.e., the data that are extremely different from existing data will not drastically change the value of median.

When filling the missing values, one must exercise prudence. Imputing the missing values in a variable with some statistical property of that variable might lead to unwanted changes in nature of the variable, e.g., its distribution.

To observe the changes in imputed variables, the density plot of the variables `credit_score` and `annual_mileage` are plotted alongside their imputed versions in the following.



Density plot of annual_mileage

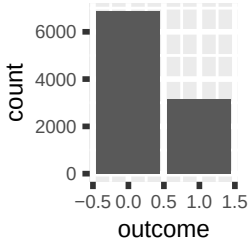


Density plot of credit_score

The dashed blue lines indicate the imputed versions. Evidenced by the plots, the `credit_score` variable's imputed version has a slight shift in its mean. Notwithstanding, this work resorted to median for filling missing values, due to good performance of the models and due to computational limit of using more advanced methods, of which a prominent one is multiple imputation using chained equations [5].

5.2 Class Imabalance

The proportion of outcome classes are shown in the following:



Proportion of outcome classes

The number of class 0 and class 1 labels are respectively 6867, 3133. Imbalance ratio is defined as size of minority class (1 in our case) over size of majority class (0). Therefore, this quantity is 1 for a fully balanced binary variable. Imbalance ratio for the outcome variable in the dataset is 0.45624

To make the dataset balanced, oversampling methods can be used, which are methods that generate samples from observations with the minority class (having outcome equal to 1). The key challenge in these methods is that the samples should be similar to the original dataset, as their information, i.e., distribution and other statistical properties should align with the original data, however, they would better be different to a small reasonable extent too, so as

to resemble new data available in the dataset, not merely a copy of what exists therein. The oversampling method that is used for this work is the RaCog algorithm. After oversampling using the mentioned method, the number of class 0 and class 1 labels would become respectively 5484, 5516, leading to a imbalance ratio approximating 1. Prior to explaining the algorithm in 5.6, the process of representing the categorical columns (variables) of the dataset as numerical ones is explained in 5.3, as this process is a prerequisite for many oversampling methods. For RaCog, the dataset should be discretized and numeric variables, and when one-hot-encoded, the dataset would satisfy this property.

5.3 Encoding Categorical Columns

As mentioned earlier, a prerequisite of RaCog algorithm is a discretized and numeric dataset. In addition to this, there are other reasons for encoding the dataset to a one containing only numbers. Firstly, most machine learning algorithms require that input and output variables are numbers. Secondly, even if the dataset only contain numbers, and yet a categorical variable is represented with natural numbers, the following two cases might occur: Either the variable has a natural order, e.g., `age` which contains four age groups can be represented with numbers {1,2,3,4}, such that higher numbers correspond to older age groups. The second case is the case of variables `vehicle_type`, `gender`, `race`, etc that are not suitable to be represented with natural numbers, as higher or lower numbers does not indicate any inherent order, and this might mislead a machine learning model.

One-hot-encoding is used to encode (represent) categorical variables as numerical values but not by natural numbers, rather by binary 0 and 1 values, which would omit any implicit indication of order existence in the variables. This would overcome the ordering issue mentioned earlier. One-hot-encoding is done by creating a new binary column for each unique category in the data, with a value of 1 in the column corresponding to that particular category, and 0 in all other columns.

5.4 Train/Test Split

After encoding the dataset, it is splitted to train and test datasets, with 80% proportion for the former and 20% for the latter. The oversampling method in 5.6, the classification models in 6, and the hyperparameter tuning in 6.3 are all done using the train data, and the evaluation processes in 7 is done on the test data, which is the data that is unobserved by the models.

5.5 Normalize

As this point data is normalized as well, meaning that numerical variables are transformed to a homogeneous scale of values. The normalizing uses train dataset to be fitted to, i.e., to use its values for scaling. But after scaling by using statistical properties of only the train dataset, the scaler will transform both train and test dataset. Otherwise, if statistical properties of test is used during fitting the scaler, data leakage occurs and the models can cheat by using footprint of test data left on normalized train data.

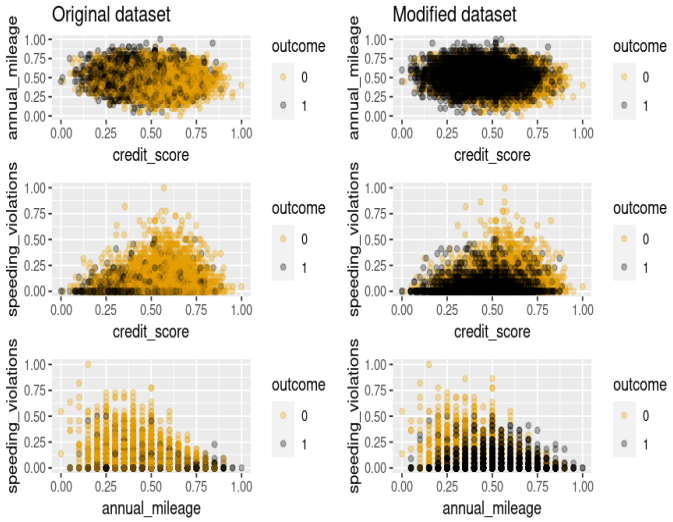
5.6 Oversampling minority class using RaCog

Existing oversampling approaches for addressing imbalanced dataset (explained in 5.2) typically do not consider the probability distribution of the minority class while synthetically generating new samples. This leads to poor representation of the minority class, and hence to poor classification performance.

Rapidly converging Gibbs algorithm (RaCog) uses the joint probability distribution of input variables as well as Gibbs sampling to generate new minority class samples. Gibbs sampling is a Markov Chain Monte Carlo (MCMC) method [6] that is used to sample from a multi-dimensional distribution. The basic idea behind Gibbs sampling is that in order to sample from a probability distribution with multiple random variables, instead of directly sampling from the joint distribution of all the variables, the algorithm iteratively sample from the conditional distributions of each variable given the current values of the other variables. By dint of this, the computational difficulties of sampling from the full joint distribution are avoided. The result of the algorithm is a sequence of samples from the joint distribution of all the variables, which can be used to estimate various properties of the distribution, such as the mean and variance of each variable.

In summary, Gibbs sampling is a way to iteratively sample from the conditional distributions of each variable given the current values of the other variables, to estimate properties of a multi-dimensional distribution. For more rigorous and detailed information on RaCog alogithm, the interested reader is referred to [7].

In following, a grid of one to one variable comparison is presented, wherein the the prior imbalanced dataset is placed next to the balanced one, for three variables (or six pair of variables).



6 Classification Models

6.1 Logistic Regression

Among results of fitting logistic regression (logit) on the dataset, the significance level of variables are reported, obtained by perturbing a small change in the variable under study while holding all other variables constant. The most significant variables that affects the prediction of outcomes, i.e., 1 for approval and 0 for or rejection of car insurance claims are listed below. The high significance levels were determined by observing small p-values. However, the direction of the variables' impact remains yet to be determined. The aim of deriving direction of impact is to answer the following question. Would a significant variable have an effective role in increasing the chance of approval (class 1), or increasing the chance of rejection (class 0)? The answer of this question lies at the sign of the coefficient of each variable in the model. Therefore, the direction of significance are also reported below. Positive effect means the variable increased chance in class 1.

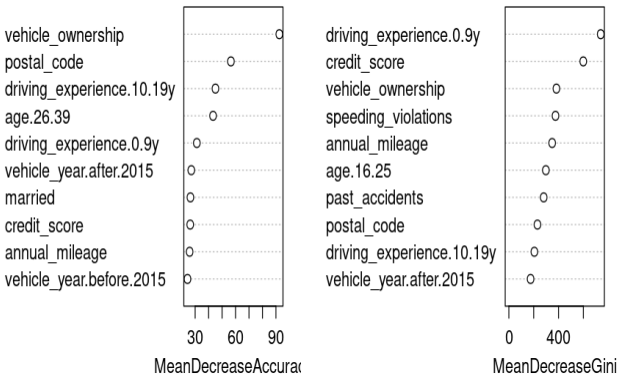
- `driving_experience.0.9y`: Having driving experience between 0-9 years, positive.
- `vehicle_ownership`: Whether the policyholder owns the vehicle or not, negative.
- `vehicle_year.after.2015`: The vehicle was built after 2015, negative.
- `gender.female`: The policyholder being female, negative.
- `postal_code`: Postal code of the policyholder.

6.2 Random Forest

The random forest model can also report significance level of variables using different measures, two of which will be reported, accuracy, and Gini.

The following plot visualizes significance level of variables. The left part shows how much accuracy the model loses by excluding each variable. The more the accuracy suffers (higher `MeanDecreaseAccuracy`), the more important the variable is for the successful classification. The variables are presented in descending quantity order of importance. The right part shows the mean decrease in Gini coefficient. The Gini coefficient is a measure of how each variable contributes to the homogeneity of the nodes and leaves in the resulting random forest. Again, higher `MeanDecreaseGini` indicates more significance.

A salient point to take into account is that almost all variables for which the logit model 6.1 considered high significance level, also had high significance level in random forest 6.2, judged by at least one of the accuracy or Gini measurements. The only variable considered significant by logit but not by random forest was `gender.female`. And the variables that were considered significant by random forest but not by logit were the following: `age`, `speeding_violations`, and `credit_score`.



6.3 Hyper-parameter tuning

Although the classification models and in general machine learning models learn to find the best parameters of a given model, there are some parameters that are not learned during the training process, therefore they should be chosen by the human. Using intuition, trial and error, and relying on domain knowledge are common practices to find the best set of hyperparameters. Another essential method is to do a systematic search and try all (or a rich subset of) possible combinations. This is usually computationally intensive. In this work, the following hyperparameters were chosen by search:

1. **Logit:**

- λ: The parameter that controls the amount of regularization applied to the modeling logistic regression derived from generalized linear model.

2. **Random Forest:**

- `ntree`: Number of trees
- `mtry`: Number of variables randomly sampled as candidates at each split

7 Evaluation

In the following, various evaluation metrics are employed to measure, compare, and report the classification performance of the models. The test set (explained in 5.4) is used for evaluating the models. The prediction of the model (given test set as input and outputting outcome variable) are compared with the test set's real values of outcome.

7.1 Confusion Matrix

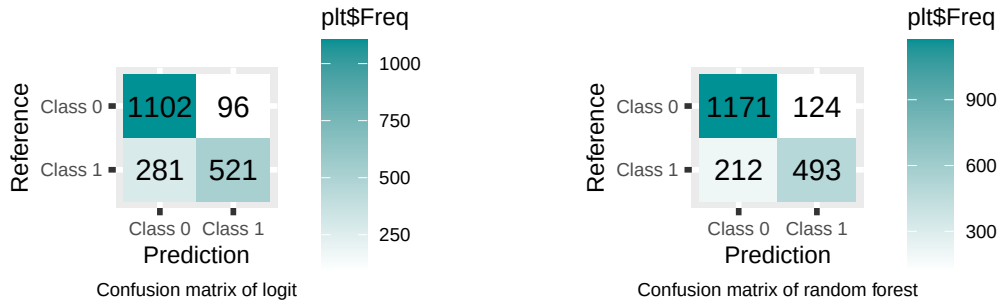
The confusion matrix is made up of four different cells, each representing the number of correctly and incorrectly classified instances of a particular class. The cells are typically arranged in the following format:

True Positive (TP)	False Positive (FP)
False Negative (FN)	True Negative (TN)

The four quantities are defined below:

- **True Positive (TP)**: cases where the model predicted the positive class, and the actual output was also positive.
- **False Positive (FP)**: cases where the model predicted the positive class, but the actual output was negative.
- **False Negative (FN)**: cases where the model predicted the negative class, but the actual output was positive.
- **True Negative (TN)**: cases where the model predicted the negative class, and the actual output was also negative.

In the following plot, the confusion matrix for both models are visualized.



Both models demonstrate similar performance, with difference that logit have more misclassification cases for predicting class 0, and more correct classification cases for class 1, whereas random forest have more misclassification cases for predicting class 1, and more correct classifications for class 0.

7.2 Metrics derived from confusion matrix

The formulas for the evaluation metrics needed for the rest of this work are defined in the following, derived from terms existing in the confusion matrix explained earlier. Two new terms used here are condition positive (P), i.e., the number of real positive cases in the data, and condition negative (N), i.e., the number of real negative cases in the data.

- Accuracy:
$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$
- Precision (positive predictive value or PPV):
$$PPV = \frac{TP}{TP + FP}$$
- FDR (false discovery rate):
$$FDR = 1 - PPV$$
- Recall (True Positive Rate or TPR)
$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$
- Fall-out (False Positive Rate or FPR):
$$\frac{FP}{N} = \frac{FP}{FP + TN}$$

Metric	Logit	Random Forest
Accuracy	0.8115	0.832
FDR	0.1555916	0.2009724
Precision	0.8444084	0.7990276
Recall	0.6496259	0.6992908

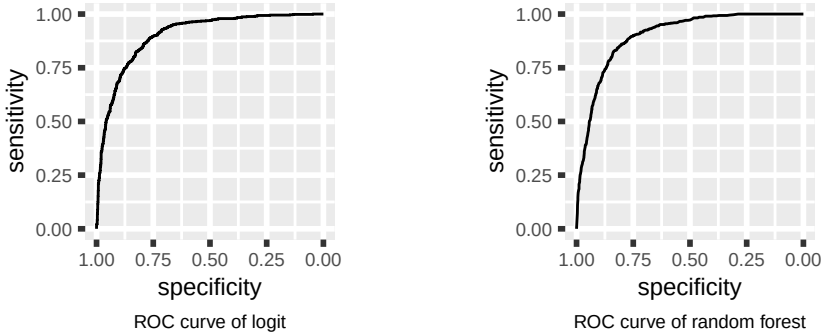
Table 1: Comparison of Evaluation Metrics for Two Models

From insurance company's perspective, the most prominent loss would be to approve a claim falsely. This is equivalent to misclassify an outcome to class 1 which originally belongs to class 0. But rejecting a claim by false wouldn't have the same financial damage. Therefore, a quantity that concentrate on the number misclassifications to label 1 would

be valuable. From the reported evaluation metrics, of particular note is the FDR (false discovery rate), as it shows exactly quantity discussed. The lower this quantity is, the less falsely approved claims will occur. This quantity is lower in logit, evidenced by 1. If judged by this metrics, the logit is preferable over random forest.

7.3 ROC Curve

An ROC curve plots the recall (true positive rate) against the false positive rate (FPR) at different classification thresholds. The curve is visualized for the two models in the following:



Both models demonstrate similar ROC curves, and their curves are inclined toward the upper left of the plot, the part corresponding the least mistakes and most correct classifications.

8 Conclusion

This report described the project of analysis of the car insurance claims dataset, and prediction of claims' outcomes (approval or rejection) based on historical data of policyholders. The classification models used in this work were random forest and logit, which showed promising performance, with most evaluation metrics being more than 70 percent. Since the insurance company would be more concerned about falsely approving a claim, the FDR metrics was reported, with lower value for logit, implying a preferred model The `aiinsurance` R package [1] is developed to make this work reproducible, accessible, and equipped with advanced features, e.g., a pipeline that performs the main processes of this work in a single command, and an interactive app that displays the performance of the models. Judged by both models, the following variables had the most effective role in prediction: Having driving experience between 0-9 years would make the claim more likely to accept. In contrast, not owning the vehicle, the vehicle being built after 2015, and the policyholder being female would make the claim more likely to be rejected.

References

[1] Hamed Vaheb. aiinsurance Package. Available at http://github.com/berserkhmdvhb/aiinsurance.

[2] Ronald Richman. Ai in actuarial science – a review of recent advances – part 1. *Annals of Actuarial Science*, 15(2):207–229, 2021.

[3] Christopher Blier-Wong, Hélène Cossette, Luc Lamontagne, and Etienne Marceau. Machine learning in p&c insurance: A review for pricing and reserving. *Risks*, 9(1), 2021.

[4] Mohamed Hanafy and Ruixing Ming. Classification of the insureds using integrated machine learning algorithms: A comparative study. *Applied artificial intelligence*, 36(1), 2022.

[5] Ian R. White, Patrick Royston, and Angela M. Wood. Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in Medicine*, 30(4):377–399, 2011.

[6] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[7] Barnan Das, Narayanan C. Krishnan, and Diane J. Cook. Racog and wracog: Two probabilistic oversampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):222–234, 2015.