Insurance Claims Prediction and Risk Profiling

Rajput, Kabir *20560795*
Relekar, Akhil *20656087*
Sami, Muhammad Talha *20581658*

Abstract

With the recent issuance of virtual banking and virtual insurance licenses on Hong Kong by the Hong Kong Monetary Authority and the Insurance Authority of Hong Kong respectively, there has been a surge in the new entrants to the banking and insurance market.

Traditional banks and insurance companies face an existential threat to their businesses due to the lower operating costs that these virtual businesses have. Apart from the lower costs, these businesses also have access to wide ranging data from customer traits, personality types and preferences.

Our group tries to leverage insurance related data collected from 3rd party companies and insurance companies to classify customers for risk profiling and further targeting insurance products using recommendation system and finally predicting the claims for the customers.

1 Problem

Big Data technologies are applied to predict risks and claims, to monitor and to analyze them in order to develop effective strategies for customers attraction and retention.

- Fraud detection
- Price optimization
- Personalized marketing
- Customer segmentation
- Risk assessment
- Claims prediction

Compared to other segments, travel insurance adopts big data and AI technologies particularly well. The relatively low policy price makes purchasing insurance a fairly quick decision, so this industry deals with an impressive number of requests. Technology can speed up the interaction with customers, give more tailored products and services, automate simple communication, improve customer satisfaction, and quickly configure the most beneficial offer.

2 Dataset

We are using three datasets for this project. One contains health records of customers like age, bmi, is the customer a smoker and then finally it shows the charges that the insurance company charges the customer based on the above factors. We used this dataset[1] for risk profiling of customers.
The second dataset is a claims prediction challenge for auto-insurance. The dataset contains variables related to the car like car-model, purchase date, etc which are used to predict the insurance claim. The dataset[2] is used for claims prediction.
The third dataset[3] is used for fraud detection which contains variables like claim amount, policy category, policy deductible, insurance date. These variables are used to detect if the claim that the customer is claiming is a fraud case or not.

3 Execution Plan

1. Collect data & extract features
    - Data cleansing and preprocessing
        - Remove nulls and corrupted data
        - Convert textual data to numeric data
        - Categorical Type Conversion
2. Determine a model
    - Linear regression (loss / optimization) > sales prediction
    - Classification (logistic regression / sigmoid function) > claim prediction
    - Nearest neighbour > claim prediction
    - Decision tree > claim prediction

- Clustering (k-means / hierarchical) > customer segmentation

3. Train the model
   - Compare model performance and find out which model has the best performance / fits the data well
   - Measures to avoid underfitting / overfitting
     - Cross validation
     - Regularization
     - Ensemble learning
   - Find out outliers if there is any

Outcome

- Build a model that is able to accurately predict the claim status of a potential client
- Insights for product marketing / cost saving

4 Methodology

The process of data preparation has been streamlined and generalized so that it can be applied for all the different use cases that are discussed in the project. Dataset[2] from Kaggle has been chosen to show the methodology of each process in this section where the goal is to predict Bodily Injury Liability Insurance claim payments based on the characteristics of the insured customer's vehicle.

```
Blind_Make         1
Blind_Model        1
Blind_Submodel     1
Cat1               5
Cat2             352
Cat4             512
Cat5             512
Cat6               5
Cat7             610
Cat11              1
Cat12              1
OrdCat             1
dtype: int64
```

Figure 1: Summary of the missing data

4.1 Phase 1: Data Pre-processing

To mimic the nature of real data collection process, the training dataset provided by Kaggle contained missing values (coded as "?"). The following two approaches were taken to handle the missing data:

- ***Deletion/Omission:*** Although getting rid of data is not the preferred choice, there were some variables that were dropped off because they were missing more than threshold (custom set) number of data (10% of total variable size in this case).
- ***Data Imputation:*** The resulting dataset had only a few rows of missing values which was recoverable with the help of imputation/interpolation techniques supported by the pandas and scikit library.
- ***Outlier Detection & Removal:*** With the completed dataset, we next go through the process of Outlier Detection & Removal. There were two types of analysis that were followed to detect the outliers - Univariate (one variable outlier analysis) and Logical analysis. The univariate detection was implemented with the help of Interquartile Range (IQR) value of the particular variable. The IQR was used to identify outliers by defining limits on the sample values that are a factor k (k=2 in our case) of the IQR below the 25th percentile or above the 75th percentile. The second type of analysis used logical reasoning and domain knowledge where data points were filtered out that didn't fit any logical profile for the particular variable.

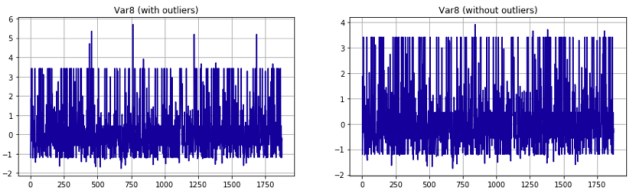

Figure 2: Variable 'Var8' (Before and after Outlier removal)

- ***Handling categorical variables:*** Categorical Data is the data that generally takes a limited number of possible values which doesn't necessarily have to be numeric in nature. All machine learning models are some kind of mathematical model that need numbers to work with. So in order to make use of categorical data as input for the machine learning model it had to first go through some encoding/conversion process. This was done (based on the nature of the feature) using One Hot Encoder (sklearn)/get dummy variables (pandas) where each category of the feature was converted into a new dummy feature with values 1 (present) or 0 (absent).

The result of data pre-processing phase gives clean dataset with all features containing only numerical values.

4.2 Phase 2: Pre-modeling Data preparation

This phase helps us to curate all data so that the machine learning model has to process only what is necessary and relevant to solve the problem.

- ***Feature Study:*** The features of the cleaned dataset are then studied and analysed to understand its distribution and balance. As per the figure 3, quick visual exploration

concludes that all the original numerical features seems to have good distribution of data within its value range except one (feature: "Claim Amount" with majorly zero as its value).

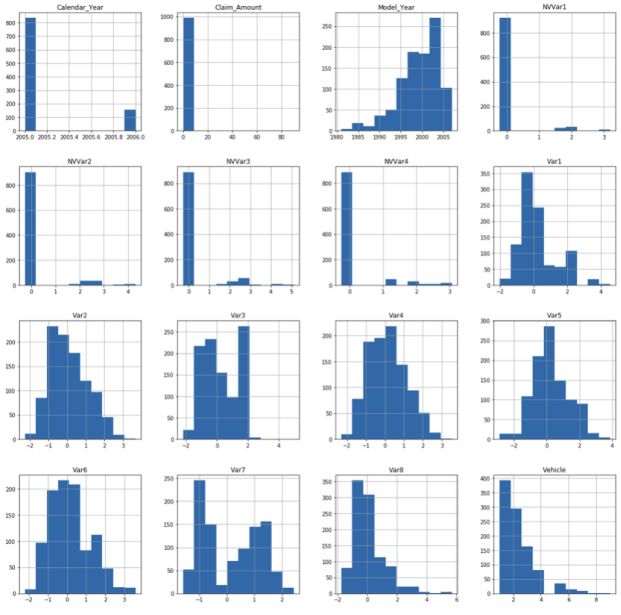

Figure 3: Distribution plot of the numerical features

To improve the prediction power of our model, the dataset was made to undergo the process of resampling. Chosen dataset for the analysis is actually a subset of the original dataset and since the size is small (1000 rows) Oversampling (Synthetic Minority Over-Sampling Technique for Regression with Gaussian Noise) has been applied to balance out the zeroes and non-zero values as shown in figure 4. The downside to this is that the model will have bias but it is better than model that mostly just predicts zeroes.

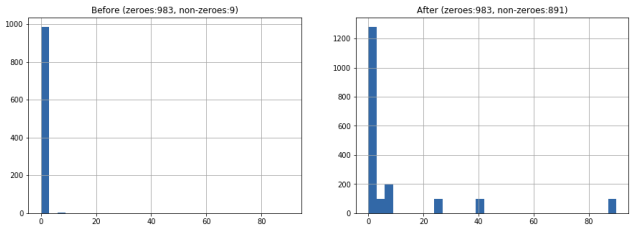

Figure 4: Target variable ('Claim Amount') distribution before and after oversampling

- ***Feature Selection:*** Not all features contribute towards enhancing the model's prediction power so for the sake of the following reasons, the entire feature set is filtered to get the most useful features:
  - *Reduces the complexity -> reduces the chances of overfitting the model*
  - *Reduced number of features -> enables model to train faster*

The features are selected based on their level of correlation with the target variable. Figure 5 illustrates the correlation of all features against all features in the form a matrix plot.

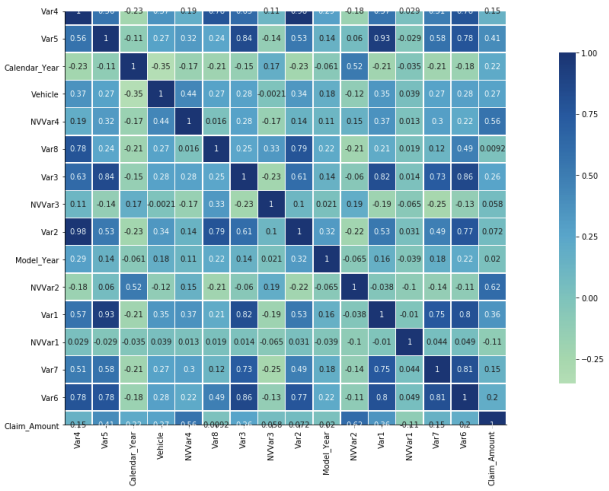

Figure 5: Correlation matrix plot of the features including the target variable

The effective features are picked using a Filter method (SelectKBest features using f regression scoring) that utilizes univariate statistics ('relevance" with target variable quantified as feature importance score).

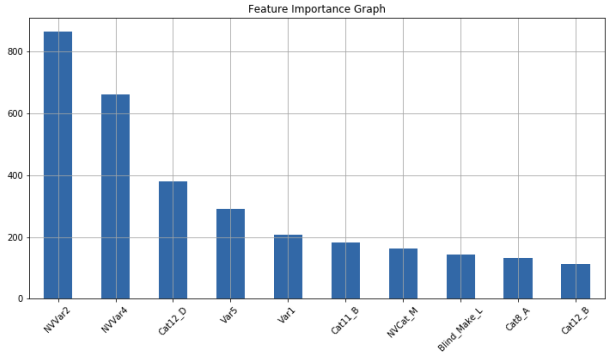

Figure 6: Bar plot of the top 10 features according to filter method

A quick glance at figure 5 and 6 shows that features were selected because of their relative high correlation scores. This approach was chosen to perform the feature selection because it is model independent (low chance

of overfitting) and helps in the later stage during the comparative study of different models.

- ***Train/Test Split:*** In order to really understand and gauge the performance of the model, the given data set is split into train and test data (using the train test split function from sklearn library) so that the model is fit on train data and test its predictions against the test data. In the chosen scenario, the dataset was split as 80/20 (80% of dataset as train data and 20% as test data) while maintaining the general distribution of the features.

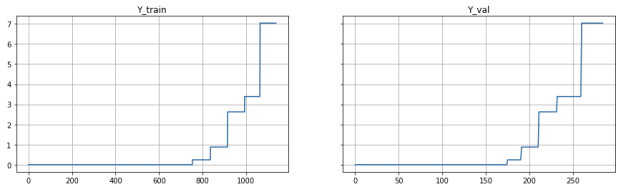

Figure 7: 80/20 train/test split of Target variable ('Claim Amount')

- ***Normalization:*** All features don't necessarily have the same value range within the dataset and this can affect the way how some of the models learn to predict. So to eliminate this kind of bias, all the features were normalized. The features in the chosen dataset already had a good distribution within the range, so a simple min-max scaler (from sklearn) was used which just shifted and bounded all features to be of values within [0,1] while retaining the original distribution shape.

4.3 Phase 3: Modelling and Analysis

In this phase, we employed a number of different modelling techniques for each of the different use cases under consideration.

- ***Evaluation Metric(s):*** For any given use case, we used the same metric to evaluate performance across all the models; this allowed us to objectively compare model performance across the different models.
  After iterating through a few different metrics, we selected Root Mean Squared Error as the evaluation metric across all our use cases. Its simplicity of use and well-rounded capturing of both overestimates and underestimates in predictions were the major reasons behind this choice.

- ***Standard techniques:*** Before we delve into the models, we will introduce a few high-level ensemble techniques that were employed across the different models to improve results:
  - *Bagging:* Bagging is the technique whereby models are bootstrapped first by training them on subsets of data, and then aggregated through an averaging or pooling mechanism in case of regression, or a voting mechanism in case of classification. This reduces the variance of the sample on which the model is being trained.
  - *Boosting:* Boosting is the technique where successive models are trained after increasing weights of examples that were misclassified or regressed with high residuals in the preceding batches. This helps reduce the bias of the training data.

- ***Models Used:*** Across the different use cases, we made use of Decision Trees, ExtraTree and ExtraTrees regressors, Random Forests regressor, K Neighbors regressor, AdaBoost, and XGBoost. But in this report, we will only focus on the 4 models from which we were able to derive the most meaningful results and insights: K Neighbors Regressor, Decision Trees, Random Forests, Extra Trees and XGBoost.
  - *K Neighbors Regressor:* K Neighbors Regressor uses a similarity measure to determine distance between the feature vector of a point and the feature vectors of other points already observed in training. This similarity measure could simply be based on the Euclidean distance between the vectors. It then predicts the value of the current example by taking an average of the values of the K points with the least distance (or most similarity) from the current example.
  - *Decision Trees:* Decision Trees is a model that splits the feature vector recursively, selecting a binary (Yes/No) at every level, until the prediction label or regression result can be obtained. The node splitting in decision trees is usually based on information gain, which is defined as the loss of entropy between successive levels of the tree.
  - *Random Forest Regressor:* Decision trees tend to overfit on the sample data. One way of overcoming that is by creating different decision trees on different subsamples of data, and then aggregating them through an averaging/voting mechanism. This kind of model is referred to as Random Forest Regressor.
  - *Extra Trees Regressor:* One model which takes this spirit of reducing sample variance even further is ExtraTrees regressor, which works in a similar way as Random Forests, but splits each tree randomly instead on a preselected information gain metric.
  - *XG Boost:* In XGBoost, a gradient descent procedure is used to minimize the loss when adding trees.

Traditionally, gradient descent is used to minimize a set of parameters, such as the coefficients in a regression equation or weights in a neural network. After calculating error or loss, the weights are updated to minimize that error. In our case, instead of parameters, we had weak learner sub-models or more specifically decision trees. After calculating the loss, to perform the gradient descent procedure, we add a tree to the model that reduces the loss (i.e. follow the gradient). We do this by parameterizing the tree, then modifying the parameters of the tree and moving in the right direction (i.e. reducing the residual loss).

- ***Hyperparameter Optimization:*** In some of our models like Random Forest Regressor, we employed techniques like GridSearchCV to find hyperparameters (such as max-depth of the tree, max number of estimators at any given node etc.) that were optimized for our dataset.
- ***Cross Validation:*** In two out of the three use cases we researched upon, the datasets available to us were not very large. Thus, we used cross-validation, especially when optimizing hyperparameters, to reduce the likelihood of overfitting on the sample data.

4.4 Phase 4: Analysis of results

- ***Use Case 1: Predicting amount claimed from bodily injuries***
  Best Model (with larger dataset): K Neighbors Regressor
  Best models when we tested with smaller dataset: Extra Trees Regressor, Adaboost, Bagging Regressor. (These were the results we mentioned during our presentation as we had not finished our testing on the larger dataset by then.)
  By testing with the larger dataset and observing the difference in model performance, it was clear that some of the models that seemed to perform well on the smaller dataset were actually overfitting on the sample. We then incorporated 10-Fold Cross Validation on our larger dataset to further reduce the likelihood of overfitting.
- ***Use Case 2: Risk Profiling and regressing for Cost from Customer***
  Best Model: Random Forest Regressor
  The model allowed us to assess the risk for a customer thereby allowing us to profile the customer into clusters. For example, our system charges the customer Billy a higher amount than Dennis even though he is younger than Dennis since Billy has a higher BMI, has 2 kids and smokes.
- ***Use Case 3: Detecting Fraud***
  Best Model: Decision Trees
  Using the decision tree model, we are able to get a 95 percent accuracy on the case where there is no fraud and are able to predict with a 57 percent chance that it is a case of fraud. Since fraud detection data is usually skewed to the cases where there is no fraud, the fact that our algorithm can detect fraud cases with 57 percent accuracy is pretty good in the fraud landscape.

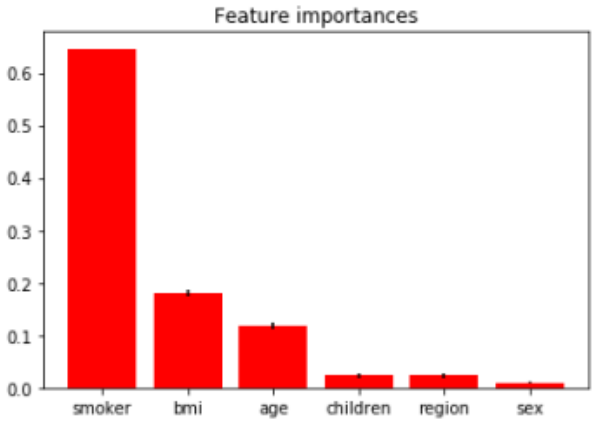

Figure 8: Feaure importance for health-care dataset

```
Predicting on new data


Billy -  ['male', 'yes', 'southeast', 25, 30.5, 2]
Cost for Billy =  33952.10115719998


Dennis -  ['female', 'no', 'southeast', 45, 19, 0]
Cost for Dennis =  9197.550964249998
```

Figure 9: Final Insurance Cost Prediction

5 Conclusion

This project allowed us to cover the whole Insurance pipeline where we got to work on three distinct problems within the Insurance Industry: from profiling risk to predicting claim amounts to detecting fraud. In each of the use cases, we were able to find models that were able to make predictions with a significant degree of precision.
Many of the models that performed well across these use cases were tree-based models, which seems to suggest that the generative functions for these use cases are step-wise-like functions spanning across different features of the customer profile and claim modalities. We hope that this insight, alongside the results captured above, will help take the research in this field to the next level.

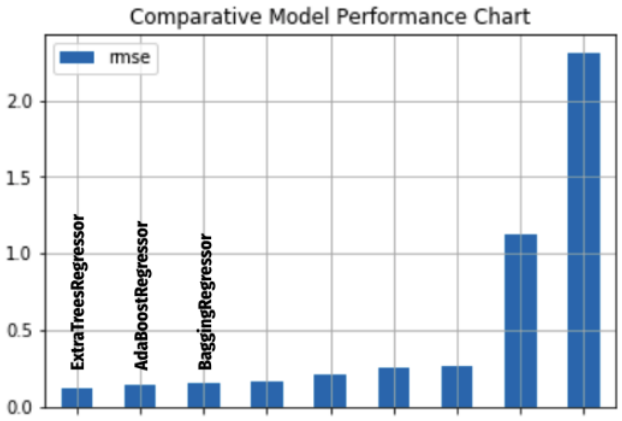

Figure 10: Bar plot showing the RMSE of various models

6 Appendix (Code)

- *Bodily Injury Claim predictor:* https://github.com/arelekar2/Insurance-Claim-Predictor
- *Health Care Insurance Cost predictor:* https://github.com/KabirRajput/Insurance-Cost-Analysis
- *Insurance Fraud Detection:* https://github.com/KabirRajput/Predicting-Insurance-Fraud
  - code inspired from Dataset[3]

7 References (Datasets)

1. https://www.kaggle.com/flagma/health-care-cost-analysys-prediction-python/data
2. https://www.kaggle.com/c/ClaimPredictionChallenge/data
3. https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/4954928053318020/1058911316420443/167703932442645/latest.html

8 Acknowledgements

- ***Akhil Relekar:*** Data Collection and preprocessing for the different use cases discussed in the report. Contributed by creating relevant slides for the group presentation and by writing relevant parts of the final report.
- ***Kabir Rajput:*** Modelling and analysis for the Health Care Insurance Cost predictor and the Fraud detector. Contributed by creating relevant slides for the group presentation and by writing relevant parts of the final report.
- ***Muhammad Talha Sami:*** Modelling and Analysis for the Bodily Injury Claim predictor. Contributed by creating relevant slides for the group presentation and by writing relevant parts of the final report.