Cross-Selling Vehicle Insurance to Current Health Insurance Policy Owners

By Isaac Yoo, Alexandru Rudi, Tae Yeon Kim

1 Abstract

This paper aims to understand consumer behavior in the cross-selling of vehicle insurance to current health insurance policy owners. Our dataset consists of features that detail the vehicular history of numerous individuals, with the label being their interest in vehicle insurance from their health insurance provider [6]. The dataset is preprocessed to improve interpretability and account for random noise. Our models consist of the Decision Tree, Random Forest, Gradient Boosting, Logistic Regression, CatBoost, and XGBoost classifiers. Hyperparameter tuning is done through the K-Fold Cross-Validation technique. Due to a large imbalance within the labels, optimizations are made for the F1 score rather than accuracy/area under the curve. Our best performing model was CatBoost, obtaining an F1 score of 0.455 while maintaining a solid AUC of 0.856. We have obtained very similar performance from XGBoost and Random Forest, and even the simple Decision Tree model ended up being very competitive, obtaining an F1 score of 0.444. These results are on par with the other results on Kaggle, where the largest F1 value obtained is 0.459, using XGBoost.

2 Introduction

Healthcare has been at the core of discussions all around the world for decades. Recent months have proven to bring this topic to the forefront of all frontiers. The COVID-19 pandemic has certainly demonstrated how quickly our everyday norms can be attacked – both in terms of how we interact with one another and how we seek care. In addition to being ready to prevent such epidemics, we must also be wary of how to provide the utmost aid to those affected by illnesses such as this. While we may be ready for the unexpected, such preparedness can only reach so far. There is no way to predict what predicaments we may come across. To circumvent this risk, insurance policies provide protection against these unforeseeable situations in exchange for a premium. While this concept is appealing in theory, financial circumstances and personal preferences influence the purchasing behavior of such policies.

Insurance does not stop at healthcare. It is evident everywhere – policies are widely available for smartphones, appliances, homes, and lives. Particularly, auto insurance is widely advertised and purchased worldwide. With the average American driving over 13,000 miles per year, constant risk exists on the road. As an active vehicle insurance is required for 49 out of 50 states within the United States, how much effort are Americans willing to invest into seeking a reliable policy? An unidentified health insurance company seeks to delve deeper into this topic. Perhaps, current health insurance owners may be interested in purchasing vehicle insurance as well from the same company.

Our project aims to capture unbiased data regarding interest in vehicle insurance from their health insurance provider. This is achieved through cross-selling: the sale of a novel product or service to an existing customer. By observing current customers who own health insurance, we analyze a wide variety of individuals from all around the country – regardless of race, gender, or ethnicity – who already demonstrate interest in insurance policies. The cross-selling of related products is an area of great interest to many companies, for which it is crucial to be able to gauge public interest in new products. Cross-selling can also help improve customer satisfaction and build loyalty between the customer and provider.

The COVID-19 pandemic encourages further studies into this subject, as consumers seek budget-friendly and reliable coverage. As disruptions occur in many aspects of our lives, individuals seek to be protected from the unexpected in every aspect. There is not one single explanation for why an individual may be interested in purchasing vehicle insurance; rather, it constitutes a multitude of factors. By observing the numerous features recorded among the various health insurance policy owners regarding their vehicular history, we aim to better understand what exactly influences a consumer's decision to purchase vehicle insurance in addition to their health insurance policy.

3 Background

Kaggle Notebook Number	Result
1	AUC = 0.868 (LightGBM)
2	F1 = 0.459, AUC = 0.859 (XGBoost)
3	AUC = 0.876 (XGBoost)
4	AUC = 0.860 (CatBoost)
5	AUC = 0.859 (CatBoost)

Looking at the most popular Kaggle notebooks for this problem, we find a couple of important observations. Most of the approaches are trying to optimize for AUC, as that is the goal of the original Kaggle problem as well. The best result was an AUC value of 0.876, using XGBoost [3]; however, the margin of improvement was very slim, as almost every solution managed to obtain an AUC score of at least 0.86. Looking at the discussions and comments, we see a few threads on the optimal evaluation metric, some suggesting that F1 score would be a better metric. Of the top ten most voted blogs, nine optimize for AUC while the other one optimizes for F1 score, obtaining a best score of 0.459, again using XGBoost [2]. However, comparing the different approaches we see in that notebook, we see that the margin of improvement is much better for F1, as even some of the strong models used like Linear Regression, which does well on AUC, only achieves an F1 score of 0.1.

Another observation is that all of the best solutions utilize a variation of Gradient Boosting models [1, 4, 5], which suggest that these models are the most fit for this dataset.

4 Methods

The learning algorithms utilized are Logistic Regression, Random Forest, Decision Tree, and Gradient Boosting (sklearn, XGBoost, CatBoost). Since the dataset we are working with has its dependent variable (or response label) in binary, the Logistic Regression classifier is useful in predicting the probabilities of the two categorical response variables and aids in the classification task. The Decision Tree classifier is another intuitive method used for classification tasks – the data is continuously split according to a parameter. We utilize the Decision Tree model to predict labels given a wide array of both categorical and continuous features. The Random Forest classifier is the third learning algorithm utilized. This ensemble machine learning algorithm is known to perform well in a wide range of classification problems by utilizing bootstrapping to provide better performance than a single Decision Tree. Since we are working with quite a large dataset, we expect the Random Forest algorithm will yield a higher performance by averaging all the different, unpruned trees in the ensemble. The fourth learning algorithm is the Gradient Boosting classifier which is mainly used for classification tasks. This ensemble learning algorithm combines multiple models together by finding the gradients of the loss function and training a new model based on the gradients. The repetition of this procedure leads to increasingly accurate predictions, and we expect that this will be the case for our classification task.

The K-Fold Cross-Validation technique is incorporated to tune the hyperparameters of each model. We aim to compare the performances of the aforementioned models based on F1 score, which is the weighted average of precision and recall. We also compare the F1 scores of the models on different subsets of features in order to determine the irrelevant ones. Given the results, we optimize our approaches and incorporate more powerful models such as XGBoost and CatBoost in order to increase the F1 score. XGBoost is an advanced implementation of Gradient Boosted Decision Trees that provides high execution speed and performance. CatBoost is another high performance implementation of Gradient Boosting that is compatible with various data types and reduces the need for extensive hyperparameter tuning. By further optimizing our models, we aim to produce more competitive models and reach higher accuracy.

5 Experiments/Results

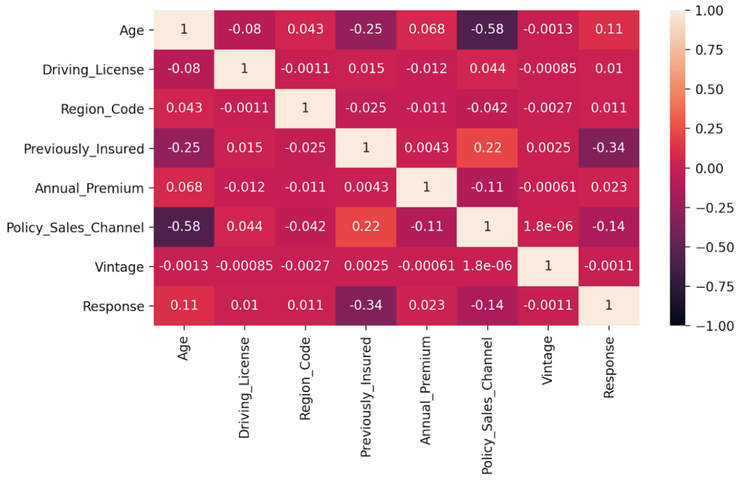
5.1 Data Description

The Kaggle dataset comprises ten features – ID, Gender, Age, Driving License, Previously Insured, Vehicle Age, Vehicle Damage, Annual Premium, Policy Sales Channel and Vintage [6]. Two features (Driving License and Previously Insured) are binary values and three features (Gender, Vehicle Age, Vehicle Damage) are categorical. The response label is recorded in binary – 0 to represent no interest and 1 to represent interest in vehicle insurance. The dataset has a total of 381,109 samples with no missing feature values or labels, thus not requiring extra cleanup. The dataset did not already come with training and testing datasets, so we split the data into testing and validation sets using a 0.7 to 0.3 ratio for the train/test split. The training dataset has 266,776 samples, and the validation dataset has 114,333 samples.

5.2 Exploratory Data Analysis, Preprocessing, Feature Extraction, Feature Selection

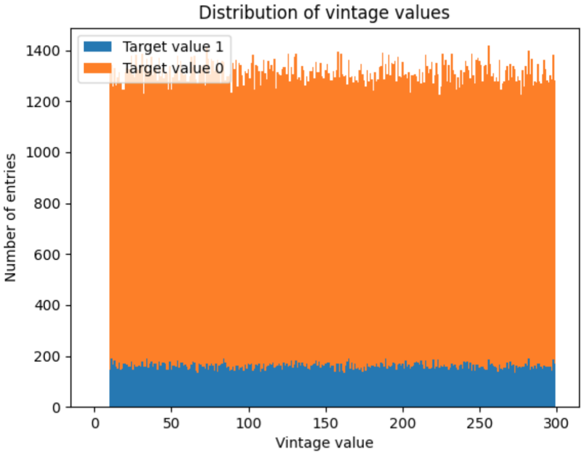
To better work with our dataset, we modify our features to preprocess them before utilizing them for our experiments. Gender and vehicle damage initially has two categories each, and they are converted to binary values of 0 and 1. Vehicle damage initially has three categories, and they are converted to ternary values of 0, 1, and 2. The dataset is then normalized to a mean of 0 and a standard deviation of 1. As the feature containing customer ID does not provide much use to the purpose of our study, this column is dropped.

In order to determine if there are correlations between the features themselves or between the features and response labels, we calculate the Pearson correlation and create a heatmap. While our plan was to drop any redundant features and features that correlated very strongly with each other, the majority of the correlations have an absolute value below 0.5. Thus, we do not have to drop any features due to redundancy.

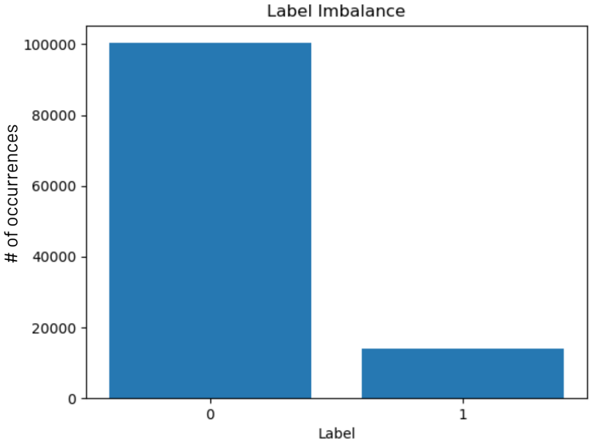


	Age	Driving_License	Region_Code	Previously_Insured	Annual_Premium	Policy_Sales_Channel	Vintage	Response
Age	1	-0.08	0.043	-0.25	0.068	-0.58	-0.0013	0.11
Driving_License	-0.08	1	-0.0011	0.015	-0.012	0.044	-0.00085	0.01
Region_Code	0.043	-0.0011	1	-0.025	-0.011	-0.042	-0.0027	0.011
Previously_Insured	-0.25	0.015	-0.025	1	0.0043	0.22	0.0025	-0.34
Annual_Premium	0.068	-0.012	-0.011	0.0043	1	-0.11	-0.00061	0.023
Policy_Sales_Channel	-0.58	0.044	-0.042	0.22	-0.11	1	1.8e-06	-0.14
Vintage	-0.0013	-0.00085	-0.0027	0.0025	-0.00061	1.8e-06	1	-0.0011
Response	0.11	0.01	0.011	-0.34	0.023	-0.14	-0.0011	1

We then question if the Vintage feature – which informs us how many days the customer has been associated with the company – provides useful information. To test this, we chart the target labels for each value of the feature. As per the figure, a lot of randomness is evident, suggesting that the Vintage feature is random noise. Thus, a second dataset is created without this feature.



Upon analyzing the target value distribution, we notice that the target labels have a significant imbalance, with 87% of labels being 0s. Due to this, we optimize for the F1 score instead of accuracy or area under the curve. As evident from the results on Kaggle, the best improvements in accuracy are about 1%, while the F1 value can be significantly increased. This is reasonable in terms of real-world application, as a lot more value would be placed on missing an interested customer.



5.3 Modeling Choices

We utilize four models – Logistic Regression, Decision Tree, Random Forest and Gradient Boosting. To find the optimal hyperparameters for all four models, we call RandomizedSearchCV on the model. Using the results from the search, we fit the model using the training data to obtain the F1 score. Due to our decision to further optimize our techniques to produce more competitive models, we repeat the exact same procedure to find the optimal hyperparameters for the XGBoost and CatBoost implementations in the latter half of our experimentation. All of the aforementioned models perform classification tasks, so these models are appropriate to use with the dataset because we aim to predict the binary response label (either 0 or 1).

The next step of the experiment is to decide on which models to continue with and how they can be further optimized. Using the finalized models, we evaluate the F1 scores, AUC scores, and AUPRC scores. Additionally, we plot the ROC curves and Precision-Recall curves for better result analysis.

5.4 Empirical Results and Comparisons

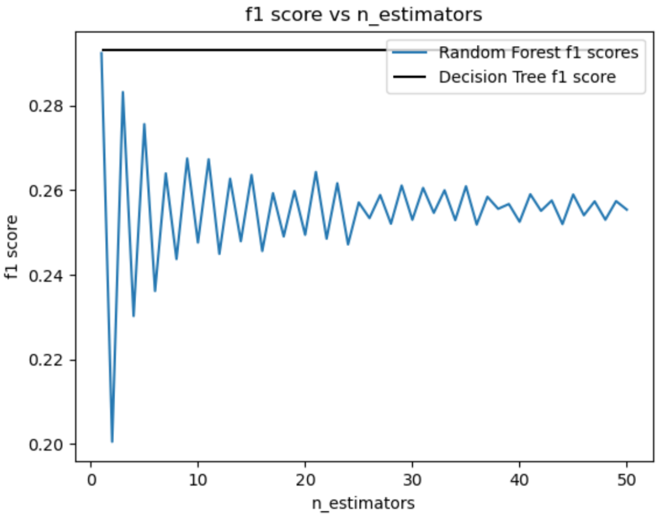
We start with the 4 models available in sklearn - Linear Regression, Decision Tree, Random Forest, Gradient Boosting. Our first test is comparing the two datasets (with and without the noisy feature). The following are the results after running the Cross Validation to tune the basic parameters (using k=5).

Model	F1 Score Train Data (CV)	F1 score Train Data w\o Vintage (CV)
Logistic Regression	0.005	0.005
Random Forest	0.187	0.251
Decision Tree	0.302	0.295
Gradient Boosting	0.309	0.302

The first observation we make is that the Logistic Regression model does not manage to perform well on F1 score, probably due to the data not being very well linearly separable. For this reason, we drop this model from further results.

The second observation is that removing the noisy feature highly increases the performance of Random Forest, while the other two models are very slightly hurt by the change. Since the decrease is not too big, we decide to continue without the feature.

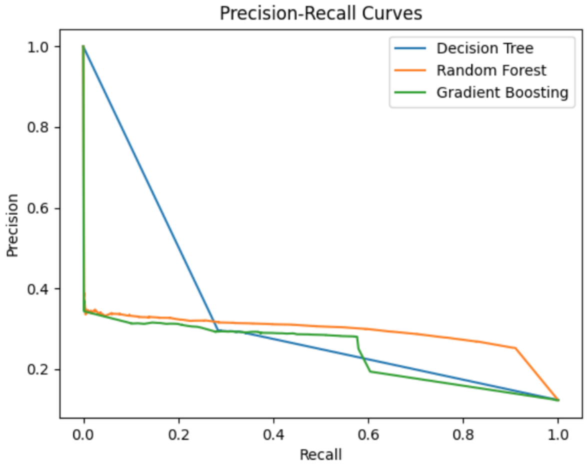
The third observation we make is that Random Forest performs worse than just a simple Decision Tree. After further testing, we find that decreasing the number of estimators of Random Forest increases the F1 score. This is probably because the target labels are unbalanced, so averaging many trees makes it more likely that we will predict a 0 for every entry, thus decreasing the F1 score.



Our next step is running these models with the found optimal parameters on the test data.

Model	F1 Score Train Data (CV)	F1 score Test Data
Random Forest	0.251	0.252
Decision Tree	0.295	0.293
Gradient Boosting	0.302	0.300

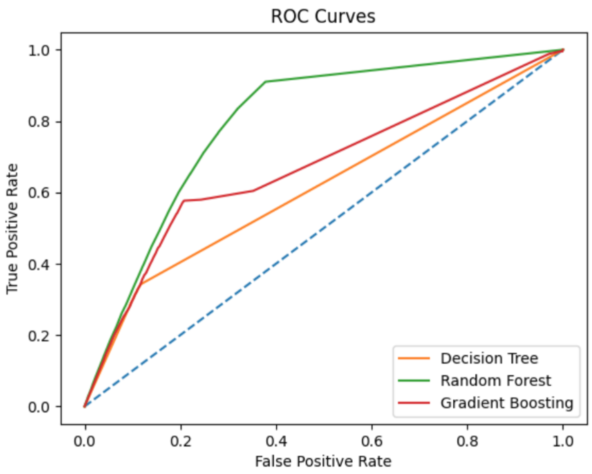
We see similar results (above), with Gradient Boosting doing slightly better than the Decision Tree, with Random Forest being significantly below both for the aforementioned reasons.



Model	AUPRC
Random Forest	0.164
Decision Tree	0.173
Gradient Boosting	0.175

Plotting the Precision-Recall curve (above), we can see very similar results, with Gradient Boosting being slightly in the lead.

For completeness, we also graph the ROC of the three models, even though we do not expect good performance since our models have been optimized for F1 score. One thing to note is that Random Forest's disadvantage for F1 score becomes an advantage here, as it performs much better in terms of AUC.
(Please see below)



Model	AUC
Random Forest	0.796
Decision Tree	0.613
Gradient Boosting	0.670

Following these initial results, we start optimizing our approach. We begin by carefully going over all the parameters that we can optimize in our models. We observe the list of parameters in the sklearn documentation and try them all. Most do not have any significant impact, except for one which dramatically improves our results — 'classweight'. This parameter gives a higher weight to one of the target values – in our case, we have the best results when we weigh a value of 1 three times more than a value of 0. Since we are optimizing for F1 score in an unbalanced dataset, it becomes obvious why this optimization improves the result this much. However, the parameter only exists in Decision Trees and Random Forest, so for Gradient Boosting, we have to use more advanced implementations than sklearn. We pick XGBoost and CatBoost. These are our final results:

Model	Best Parameters

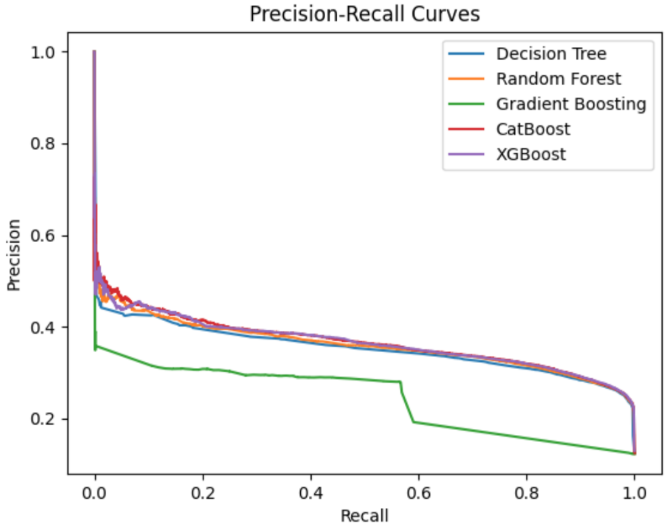
Decision Tree	min_samples_leaf=75, max_depth=10 criterion='gini',
	class_weight={0: 1, 1: 3}

Random Forest	n_estimators=10, min_samples_leaf=50, max_depth=10,
	criterion='entropy', max_features=0.5, class_weight={0: 1, 1: 3}

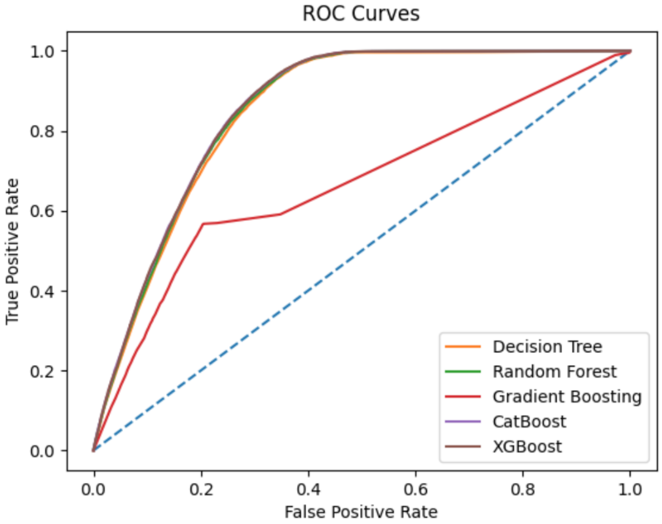
XGBoost	scale_pos_weight=3, n_estimators=50, max_depth=4,
	reg_lambda=1, gamma=0.1, eta=0.25, alpha=0

CatBoost	scale_pos_weight=3, max_depth=4,
	learning_rate=0.25, iterations=100

Model	F1 Score Train Data (CV)	F1 Score Test Data
Decision Tree	0.448	0.444
Random Forest	0.452	0.450
XGBoost	0.457	0.454
CatBoost	0.457	0.455



Model	AUPRC
Decision Tree	0.264
Random Forest	0.269
XGBoost	0.273
CatBoost	0.273



Model	AUC
Decision Tree	0.849
Random Forest	0.853
XGBoost	0.856
CatBoost	0.856

6 Discussion

In the end, we manage to obtain an F1 score of 0.45, replicating the results from the other attempts on Kaggle. Not only that, but our models have a surprisingly good showing on AUC as well, being only slightly below other models on Kaggle that were specifically trained for AUC.

In the end, CatBoost obtains the best scores for both F1 and AUC, very closely followed by XGBoost. Surprisingly, by introducing the new parameter, we have "fixed" Random Forest's issue, as it has no trouble matching the performance of the other models this time – placing in a solid third place. Finally in last place, we have the Decision Tree, which performs surprisingly well. Neither Random Forest nor the two advanced Gradient Boosting models are able to significantly over perform the simple Decision Tree model. One reason might be the fact that these ensemble models are designed to improve accuracy and AUC, not F1 score. On the other hand, the Decision Tree performs just as well on AUC as well, so perhaps this dataset simply does not benefit greatly from the additional optimization provided by the ensemble models.

The training time for each model is very small - less than a minute for each. This is because the optimal number of estimators turns out to be relatively small for each ensemble model due to the same reason as before - averaging many trees results in predicting each target value as '0' due to the unbalanced dataset.

7 Contributions

Contributions to this project were equally split among all group members. The bulk of the project was done together through Zoom meetings, including the Spotlight and Final Presentation slides and script, proposal, and final paper. In terms of programming, Tae Yeon was responsible for data processing through feature cleanup and correlation observation, Alexandru was responsible for the various models and hyperparameter tuning to ensure optimal performance, and Isaac was responsible for evaluating the models through scores, figures, and graphs.

8 Code

Click here to see the code files.

References

[1] Yashvi Patel. *Vehicle Insurance EDA and boosting models*. Kaggle, 11 September 2020.

[2] Kostiantyn Isaienkov. *Insurance Prediction. EDA and modeling (acc. 88%)*. Kaggle, September 2020.

[3] Jacob Jaszczyk. *Actuarial study: EDA,PCA,Cluster,Estimation (0.88)*. Kaggle, September 2020.

[4] Roshan Kumar G. *Rank 10 solution cross sell prediction hackathon*. Kaggle, October 2020.

[5] Anmol Kumar. *Vehicle Insurance - EDA, LGBM vs Catboost - 85.83%*. Kaggle, September 2020.

[6] Anmol Kumar. *Health Insurance Cross Sell Prediction*. Kaggle, 11 September 2020.