Health Insurance Cross Sell Prediction

Mahin Arvind C
Almabetter, Bangalore

Abstract

Acquiring a new customer is often considered more expensive than retaining an existing customer. It is known that the success rate of selling a product to a pre-existing client is higher in comparison to a new client.
The idea of cross-selling seeks to leverage these observations by selling additional products to existing customers. Cross-selling to existing clients has been one of the primary methods of generating new revenue for many businesses.
In this project, we try to identify potential buyers of vehicle insurance among existing clients who have already bought health insurance. Details about customer demographics, vehicle information and data from previous purchases obtained from the Health Insurance Cross Sale dataset is used to identify potential buyers. The efficiency of standard machine learning techniques namely Logistic Regression, Naive Bayes, Decision Trees, Bagging and Boosting ensembles are implemented and their performances are compared.

1. Problem Statement

In this project we'll be using the Health Insurance Cross Sale dataset to

1. Understand Vehicle Insurance Cross Sale Responses,
2. Apply machine learning techniques to identify potential Vehicle Insurance clients among pre-existing Health Insurance buyers, and
3. Provide reasonable explanations from the best classifying model to understand factors affecting these responses.

2. Introduction

An insurance policy is an arrangement by which a company undertakes to provide a guarantee of compensation for a specified loss, damage, illness, or death in return for the payment of a specified premium. A premium is a sum of money that the customer needs to pay regularly to an insurance company for this guarantee.
The insurance company that has provided health insurance to its customers, collected anonymized data about the demographics, vehicles, policies and previous purchases of the clients whose cross-sale response was recorded.
The goal is to use this data to build a model to predict whether the policyholders from the past year will also be interested in Vehicle Insurance provided by the company.
Building such a model to predict whether a customer would be interested in Vehicle Insurance is extremely helpful for the company as it can appropriately plan its communication strategy to reach out to customers, thereby optimizing its business model and increasing revenue.

3. Dataset Description

3.1. Data Preparation

The Health Insurance Cross Sale dataset contains the vehicle insurance cross-sale responses and details of 381,109 clients who had previously purchased Health Insurance. The dataset possesses records of customer demographics, vehicle details, medium of approach and policy details (listed in **Table. 1)**.

No missing values were observed in the dataset but it had 269 duplicates which were removed.

Table 1: Attribute Description

Feature	Type	Range
id	Continuous	0 - 381108
Gender	Categorical	Male, Female
Age	Continuous	20,21,..85
Driving License	Categorical	0,1
Region_Code	Categorical	0,1,2,...52
Previously_Insured	Categorical	0,1
Vehicle_Age	Categorical	'> 2 Years', '1-2 Years', '< 1 Year'
Annual_Premium	Continuous	[2630,540165]
PolicySalesChannel	Categorical	1,2,..163
Vintage	Continuous	[10,299]
Response	Categorical	0,1

3.2. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is the most common methodology used to summarize and visually analyze the data using different parameters.
The Pandas package has been utilized in data handling while Matplotlib and Seaborn libraries have been used for data visualization.
The count of positive and negative responses of clients is plotted in **Fig. 1a** (bottom right). Among the approached clients, only 12.25 per cent purchased the Vehicle Insurance. This essentially means that time and resources are being spent on ten customers for each successful vehicle insurance sale.

Most importantly, we must make note of the imbalance in responses which means there are fewer examples of the class we are interested in predicting.
It is also observed from **Fig. 1a** that there are a very low number of customers without a driving license, the dataset's gender make-up is fairly even and most customers approached owned a vehicle under two years of age. The distribution and outlier plot of numerical features is shown in **Fig. 1b** and **Fig. 1c.**
Fig. 2 and **Fig. 3** show most policyholders of the health insurance company come from three region codes: 28, 8 and 41, and the policy sales channels number 152, 26.0 and 124 are the most frequently used policy channels in the company. Upon finding the Cramer's V score between categorical variables, a strong association was observed in prior vehicle damage, previous vehicle insurance status and the age of the vehicle with the response of the approached client. **Fig. 4, 5** and **6** show the response rate for prior previous vehicle insurance status, vehicle damage and the age of the vehicle in a stacked column chart. We can see that the response rates are very low for vehicle owners whose car age is less than a year. Most positive respondents are vehicle owners with previous vehicle damages and no ongoing insurance. **Fig. 7, 8** and **9** show the scatter plot of numerical features in the dataset. Here, the blue blots indicate positive responses while the pink blots indicate negative responses.

3.2. Feature Engineering

Attributes such as Region_Code and Policy_Channel have over 50 categories each. Therefore, categories that occur less than 5 per cent of the time have been binned together and recognised as 'Rare' as they don't offer any significant insight, while sparing computational time, concurrently.
The age of the client's vehicle has been categorized into under one year, under two years and over two years in the form of string. These categories are better represented numerically, therefore, they were numerically encoded as one, two and seven years as an average vehicle age.

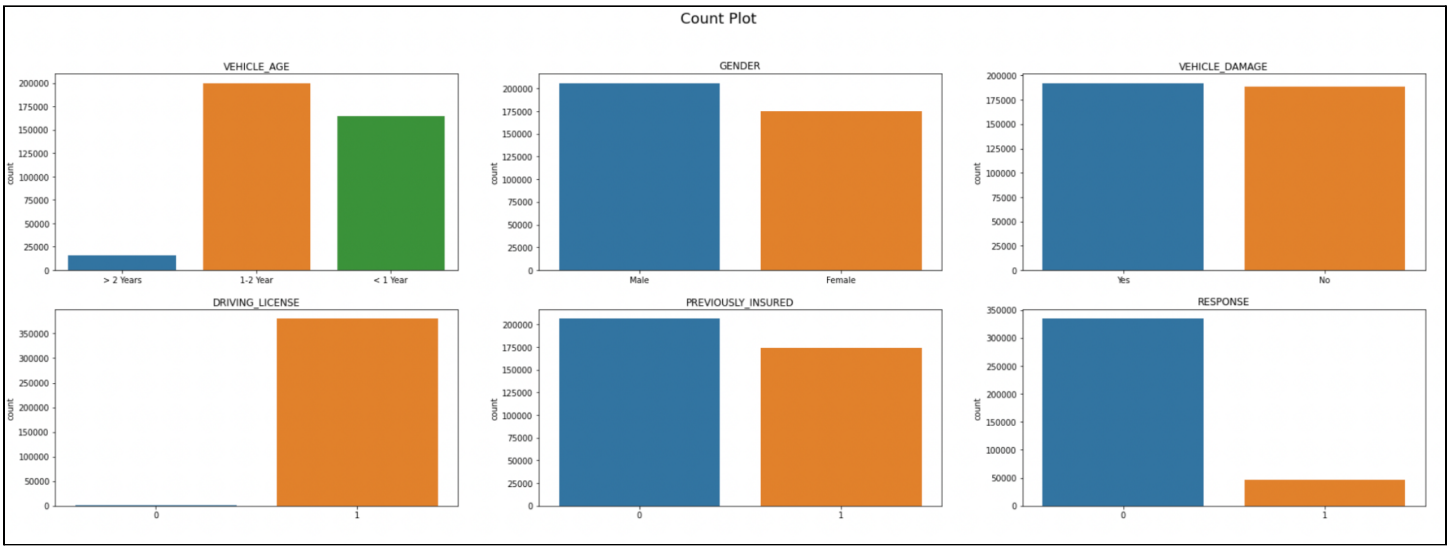

Figure 1a

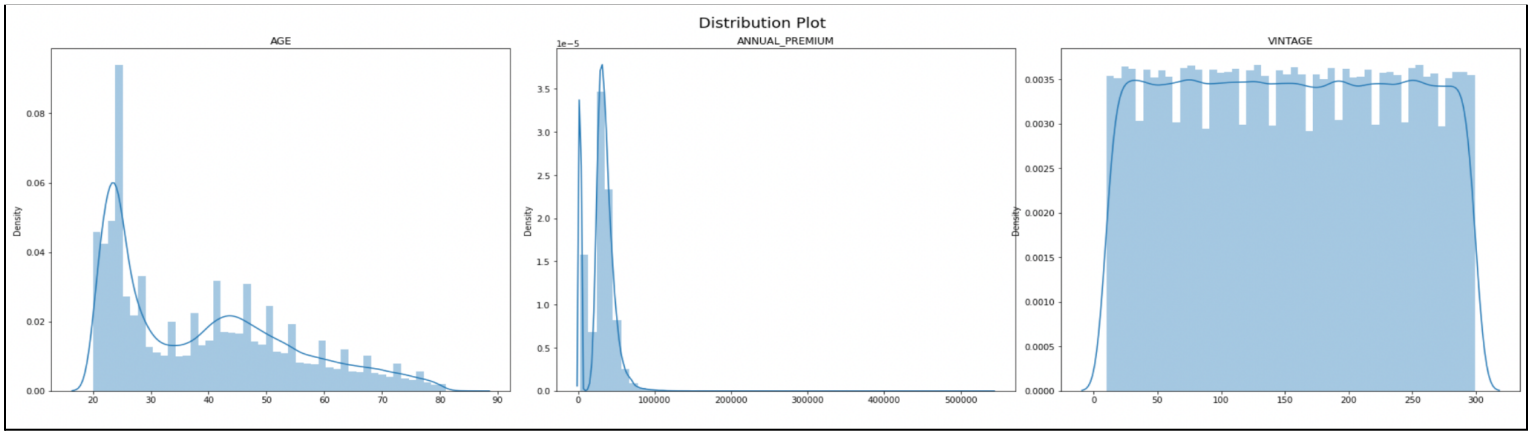

Figure 1b

Fig. 1a shows the occurrences of different categories belonging to the categorical features of the dataset. **Fig. 1b** shows the distribution of numerical features in the dataset.

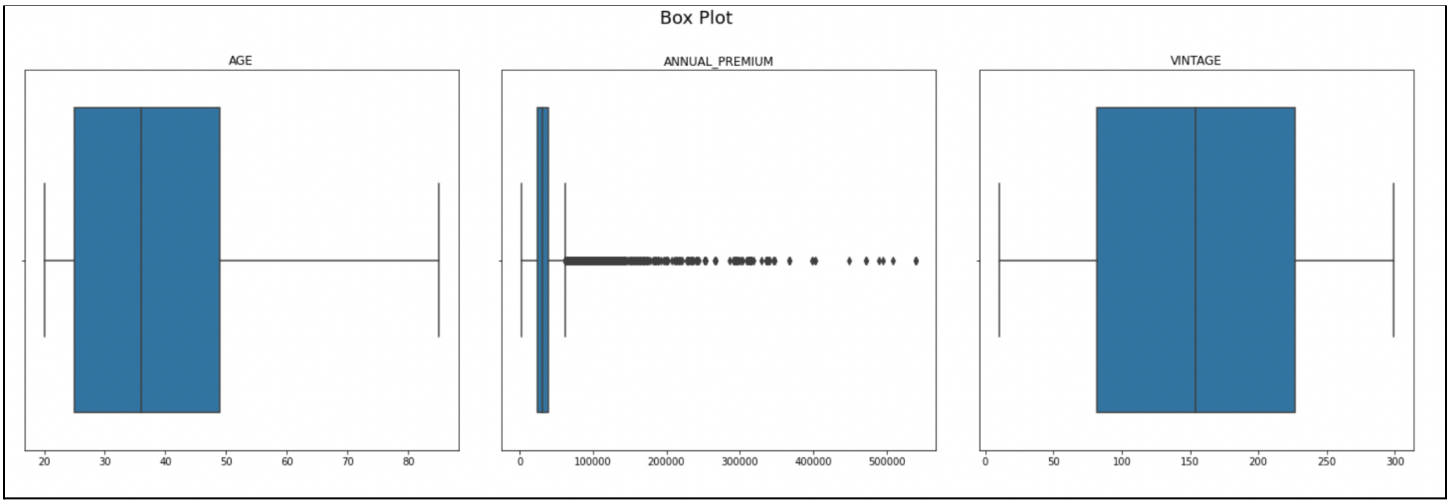

Figure 1c

Feature name	Cramer's V score	Interpretation
Vehicle Damage	0.3544	Very Strong
Previously Insured	0.3412	Very Strong
Vehicle_Age	0.2219	Strong
Gender	0.0525	Weak
Driving License	0.0102	None/Very Weak
Policy Sales Channel	0.2634	Very Strong
Region Code	0.1381	Moderate

Table 2: Test of Association with Cross Sale Response

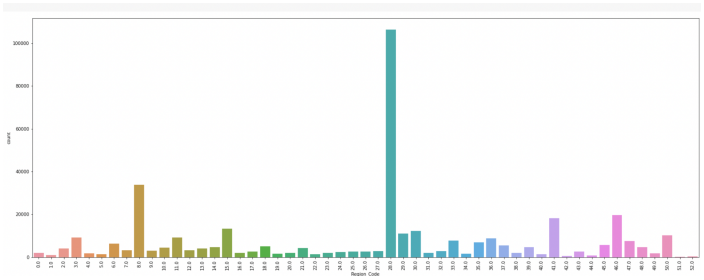

Figure 2

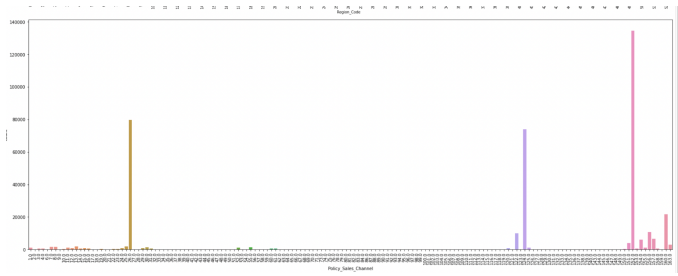

Figure 3

Fig. 1c shows the box plot of numerical features in the dataset. **Table. 2** displays the Cramer's V association scores and interpretation of categorical features and the target variable. **Fig. 2** shows the region codes of different health insurance clients. **Fig. 3** shows the medium of communication used to approach the customer for the cross sale.

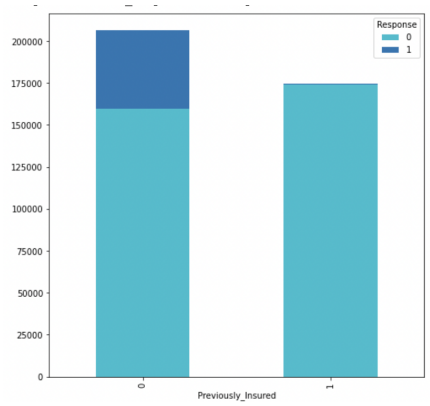

Figure 4

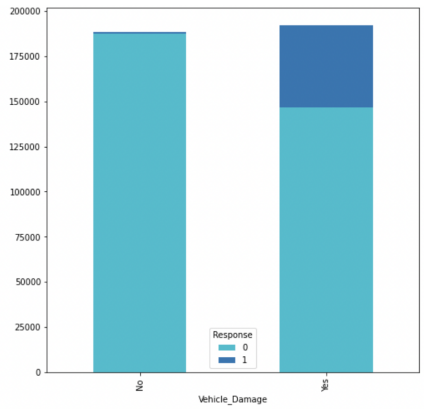

Figure 5

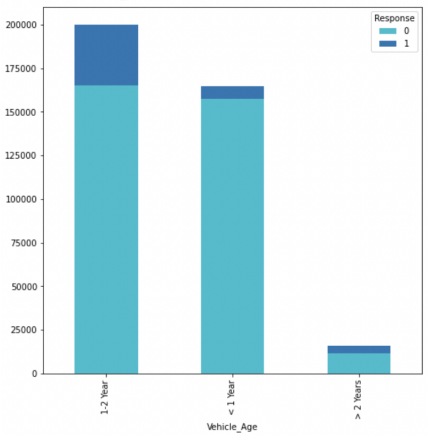

Figure 6

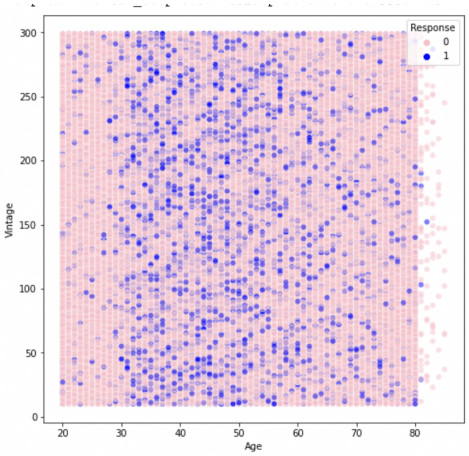

Figure 7

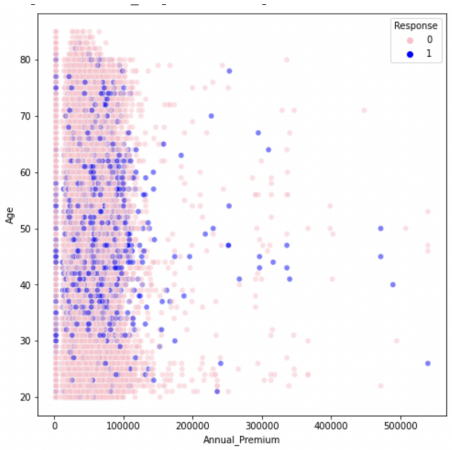

Figure 8

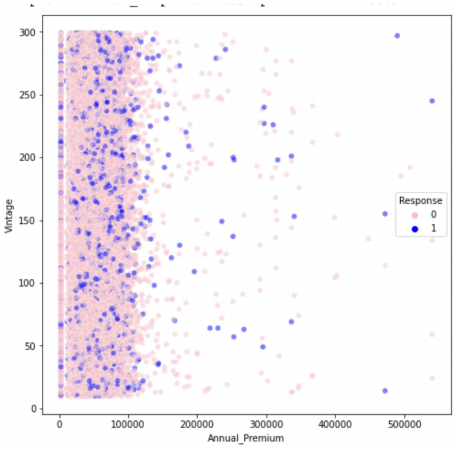

Figure 9

Fig. 4, 5 and **6** show the stacked column chart of cross-sale responses in Previous Insurances, Vehicle Damages and Vehicle Age. **Fig. 7** shows the scatter plot of vintage and customer age. **Fig. 8** shows the scatter plot of customer age and the annual premium of customers. **Fig. 9** shows the scatter plot of the vintage and the annual premium of the customer

Annual Premium is the only feature with numerical outliers. These outliers are capped at their lower and upper quartiles.

In order to make the categorical features interpreted as categories, categorical features are One Hot Encoded. One Hot Encoding converts each category in a categorical attribute into its own column taking values of 0 and 1 (1 indicating occurrence and 0 indicating absence of the category).

4. Methodology

The performance of ten machine learning algorithms are evaluated and compared to identify potential vehicle insurance buyers among existing health insurance clients

4.1. Logistic Regression

The logistic model is a statistical model that models the probability of an event taking place by having the log odds for the event as a linear combination of one or more independent variables.
Logit is a sigmoid function that produces an output between 1 and 0. The logistic function, $p(x)$ is given by

$$P(y^{(i)} = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}))}$$

If the estimated probability is greater than 50%, then the model predicts that the instance belongs to that class (called the positive class, labelled "1"), if not, it predicts that it does not belong to that class (i.e., it belongs to the negative class, labelled "0"), making it a binary classifier.

4.4. Decision Tree

A decision tree consists of a set of tree-structured decision tests working in a divide-and-conquer way. Each non-leaf node is associated with a feature test also called a split. Data falling into the node will be split into different subsets according to their different values on the feature test. Each leaf node is associated with a label, which will be assigned to instances falling into this node. In prediction, a series of feature tests are conducted starting from the root node, and the result is obtained when a leaf node is reached.
Decision tree learning algorithms are recursive processes. The key of a decision tree algorithm lies in selecting the splits and the feature-value pair which will cause the largest information gain to be selected for the split.

4.5. Gradient Boosting

In Gradient Boosting, decision trees are used as the weak learners. These weak learners solve the problem of machine learning by transforming the data into a tree representation. Gradient boosting Regression calculates the difference between the current prediction and the known correct target value each time. This difference is called the residual. Gradient boosting Regression trains a weak model that maps features to that residual. The residual predicted by a weak model is added to the existing model input and thus this process nudges the model towards the correct target. Repeating this step, again and again, improves the overall model prediction.

4.6. CatBoost

CatBoost is a member of the family of Gradient Boosting Decision Trees and is well-suited to machine learning tasks involving categorical, heterogeneous data. CatBoost encodes categorical values in order to alleviate the problem of target leakage and also chooses the most efficient combination of features during training.

CatBoost uses the Ordered TS method for encoding new features it generates from feature combinations when the combination of features includes a categorical variable[3]. CatBoost uses the following steps:

1. Dividing the records to subsets randomly,
2. Converting the labels to integer numbers, and
3. Transforming the categorical features to numerical, as:

$$avgTarget = \frac{countInClass + prior}{totalCount + 1}$$

where, 'countInClass' is the number of ones in the target for a given categorical feature, 'totalCountis' the number of previous objects and prior is specified by the starting parameter[4].

4.7. Light Gradient Boosting Machine

In LightGBM, the decision trees are grown leaf-wise, instead of checking all of the previous leaves for each new leaf. All the attributes are sorted and grouped as bins. This implementation is called histogram implementation. LightGBM has several advantages such as better accuracy, faster training speed, and is capable of large-scale data handling as it is supported by GPU learning[4].

4.8. Bagging

The Bagging algorithm consists of two key components, Bootstrapping and Aggregating. Bagging adopts the bootstrap distribution for generating different base learners. In a given training data set containing *m* number of training examples, a sample of *m* training examples will be generated by sampling with replacement. Some original examples appear more than once, while some original examples are not present in the sample. By applying the process *T* times, *T* samples of m training examples are obtained. Then, from each sample, a base learner can be trained by applying the base learning algorithm.

The most popular strategies adopted by Bagging to aggregate the outputs of the base learners are voting for classification and averaging for regression.

Given a dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ with base learning algorithm, L and number of base learners T, the steps undergone to create bagging are[2]:

1. for t = 1, . . . , T :
2. ht = L(D,Dbs) % Dbs is the bootstrap distribution
3. end

Output:

$$\boldsymbol{H(x)} = \arg\max_{y \in \mathcal{Y}} \sum_{t=1}^{T} \mathbb{I}(h_t(\boldsymbol{x}) = y)$$

4.9. Random Forest

Random Forests build a predictor ensemble with a set of decision trees that grow in randomly selected subspaces of data. The final predicted result is the majority vote taken after predictions of individual decision trees. They are fast, simple to implement, and can handle many input variables without overfitting.

Given a dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ with a feature subset size K, the steps undergone to create a random tree in a random forest algorithm is:

1. N ←create a tree node based on D;
2. if all instances in the same class then return N
3. F ←the set of features that can be split further;
4. if F is empty then return N
5. F˜ ←select K features from F randomly;
6. N.f ←the feature which has the best split point in F˜;
7. N.p ←the best split point on N.f;

8. Dl ←subset of D with values on N.f smaller than N.p ;
9. Dr ←subset of D with values on N.f no smaller than N.p ;
10. Nl ←call the process with parameters (Dl, K);
11. Nr ←call the process with parameters (Dr, K);
12. return N

Random forest algorithm is regarded as an extension of the Bagging algorithm with the difference incorporation of random feature selection. The decision boundaries of RF and its base classifiers are more flexible, leading to a better generalisation ability)}[2].

4.10. Gaussian Naive Bayes

Naive Bayes is a group of classification algorithms whose principle is based on the Bayes theorem. The fundamental Naive Bayes assumption is that each feature makes an independent and equal contribution to the outcome. Gaussian Naive Bayes is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data. The likelihood of the features is assumed to be Gaussian, hence, conditional probability is given by:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

5. Sampling Techniques:

As the provided dataset is highly imbalanced, we introduce different resampling techniques to assist the model to learn better and thereby, perform better. Three resampling techniques were performed on the training data and evaluated in this project. They are

5.1 Undersampling:

Disproportionate data can be resampled to have an equal number of classes by removing members belonging to the majority class. Tomek Links is an undersampling method that removes boundary instances as they will have nearest neighbours from opposite classes.

5.2 Oversampling:

One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique or SMOTE. This approach is effective because new synthetic examples from the minority class are created. These examples are plausible and relatively close in feature space to existing examples from the minority class.

5.3 Combination of Oversampling and Undersampling:

Oversampling methods duplicate or create new synthetic examples in the minority class, whereas undersampling methods delete examples in the majority class. Both types of sampling can be effective when used in isolation, although it can be more effective when both types of methods are used together. SMOTE is combined with Edited Nearest Neighbors undersampling as it is more aggressive at downsampling the majority class, giving an in-depth cleaning.

6. Evaluation Indices:

In order to quantitatively evaluate the performance of the classifier, classification metrics are used to

gauge model performance. The classification metrics used in this project are

True Positives: TP is the number of positive classes that were predicted to be positive by the classifier.

True Negatives: TN is the number of negative classes that were predicted to be negative by the classifier.

False Positives: FP is the number of negative classes that were predicted to be positive by the classifier.

False Negatives: FP is the number of positive classes that were predicted to be negative by the classifier.

Accuracy: Accuracy is a measure of how often the classifier makes correct predictions. It is defined as the ratio of the number of correct predictions and the total number of predictions.

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

High accuracy doesn't always reflect a model's performance. Often in an imbalanced classification problem, the interest is in predicting the minority class. In such a case the accuracy would be a misleading metric as arbitrarily predicting all members as belonging to the majority class would also increase the accuracy effectively.

Precision: Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positives are a higher concern than False Negatives. Precision for a label is defined as the number of true positives divided by the number of predicted positives

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

Recall: Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negatives are of higher concern than False Positives.

Recall for a label is defined as the number of true positives divided by the total number of actual positives.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

F1 score: The F1 score combines the precision and recall of a classifier into a single metric by taking their harmonic mean. It is primarily used to compare the performance of two classifiers.

$$F1 = 2.\frac{Precision \times Recall}{Precision + Recall}$$

The F1 score punishes extreme values more. F1 Score is used in cases where FP and FN are equally costly.

F2 score: The F2 score is a type of F measure that combines the precision and recall of a classifier in a way where it gives more weightage to recall and less weightage to precision.

$$F2.0\ Measure = \frac{(5 \text{ x } Precision \text{ x } Recall)}{(4 \text{ x } Precision + Recall)}$$

F2 measure puts more attention on minimizing false negatives than minimizing false positives.

AUC-ROC: Receiver Operating Characteristics is a curve that visualizes the tradeoff between true positive rate (TPR) and false-positive rate (FPR). AUC-ROC measures the entire two-dimensional area underneath the entire ROC curve. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0 and one whose predictions are hundred per cent correct has an AUC of 1. AUC is scale-invariant. It measures how

well predictions are ranked, rather than their absolute values.

From a profit-centric perspective, we are looking to identify all potential buyers among the existing clients as losing out on potential buyers would be more expensive than not being able to close a sale on a misidentified customer. Therefore, among the performance metrics, the most emphasis is being given to the F2 score as it gives more attention to minimizing false negatives.

6. Model Development

In order to find and decrease the error values while fitting a model, it is necessary to find optimal tuning parameters for the classifying algorithm.

The classifiers are tuned using the HalvingGridSearchCV package. It is a form of GridSearchCV that uses the successive halving technique. Successive halving is an iterative selection process where all parameter combinations are evaluated with a small number of resources during the initial iteration. After evaluation, only selected parameter combinations allocated more resources in the forthcoming iterations.

Hence after using this function, the performance for the implemented combination of hyperparameters is obtained, from which the best is picked. The hyperparameters used to optimize ensemble models are the number of trees in the ensemble and the maximum depth of trees.

As hyperparameter tuning is a time-consuming process, it will be efficient to tune the best performing classifiers producing the best F2 score.

7. Results and Discussion

The classifying models that were trained and evaluated were Logistic Regression, Gaussian Naive Bayes, Decision Tree, Random Forest, Gradient Boosting, Bagging, CatBoost and LightGBM. Their performances were assessed primarily using the F2 score while metrics like F1 score, Accuracy, Precision and Recall were used to give additional context to the model performance. The classifiers were trained using four versions of the training data, the first being the original training data and three resampled training datasets, namely SMOTE, Tomek Links and SMOTE-ENN.

Table 3 displays the evaluation report of the models with no introduction of sampling techniques. Gaussian Naive Bayes was the best performing classifier with F2 and F1 scores of 0.628088 and 0.432746.

Table 4 displays the evaluation report of the models that were trained on the dataset resampled using the Synthetic Minority Oversampling Technique. Upon oversampling, a significant increase in performance is observed in all classifiers. Logistic Regression, Gaussian Naive Bayes and Gradient Boosting produce the best F2 and F1 scores of 0.62838, 0.63490, 0.629249 and 0.417279, 0.426209, 0.432196, respectively.

Table 5 displays the evaluation report of the models that were trained on the dataset resampled using Tomek Link Undersampling. Although a notable increase in performance is observed in some models, the results in comparison to other techniques are subpar. The best performing model was the Gaussian Naive Bayes model with F2 and F1 scores of 0.432612 and 0.284894.

Table 5 displays the evaluation report of the models that were trained on dataset resampled using a combination of oversampling and selective undersampling techniques with SMOTE and Edited Nearest Neighbors. Model performance on SMOTE-ENN sampled dataset was competent with SMOTE. The ensemble models saw the highest performance when trained on the SMOTE-ENN dataset. LightGBM, Gradient Boosting and Gaussian Naive Bayes had the highest performances in this sample with F2 and F1 scores of 0.633123, 0.632103, 0.630923 and 0.438795, 0.424206, 0.415549.

Model Name	Accuracy	Recall	Precision	F1 Score	F2 Score	ROC AUC
Logistic Regression	0.877298	0.000107	0.090909	0.000214	0.000134	0.846898
Gaussian Naive Bayes	0.711257	0.898468	0.285011	0.432746	0.628088	0.825207
DecisionTree	0.823075	0.303417	0.288934	0.295998	0.300405	0.601312
Random Forest	0.85809	0.165899	0.33895	0.222766	0.184765	0.822176
Gradient Boosting	0.877403	0.000107	0.333333	0.000214	0.000134	0.854843
Bagging	0.85809	0.147156	0.325592	0.2027	0.165271	0.785966
CatBoost	0.876667	0.028382	0.451448	0.053406	0.034928	0.852722
LightGBM	0.877232	0.003427	0.410256	0.006798	0.004275	0.85508

Table 3: Model Classification Report without Sampling

Model Name	Accuracy	Recall	Precision	F1 Score	F2 Score	ROC AUC
Logistic Regression	0.675375	0.948163	0.267503	0.417279	0.62838	0.846811
Gaussian Naive Bayes	0.688885	0.942594	0.275358	0.426209	0.634901	0.825734
DecisionTree	0.816248	0.325693	0.283121	0.302919	0.316185	0.606111
Random Forest	0.808699	0.45475	0.30934	0.368209	0.415671	0.823037
Gradient Boosting	0.708815	0.904038	0.283979	0.432196	0.629249	0.851186
Bagging	0.841193	0.231766	0.305348	0.263517	0.243502	0.77262
CatBoost	0.831898	0.420049	0.346742	0.379892	0.403009	0.845253
LightGBM	0.762643	0.756667	0.308893	0.438697	0.586599	0.84619

Table 4: Model Classification Report with SMOTE Sampling

Model Name	Accuracy	Recall	Precision	F1 Score	F2 Score	ROC AUC
Logistic Regression	0.875998	0.500758	0.003641	0.193182	0.007148	0.016931
Gaussian Naive Bayes	0.711099	0.791695	0.898468	0.284894	0.432612	0.329961
DecisionTree	0.813675	0.615197	0.352254	0.287676	0.316707	0.298625
Random Forest	0.845631	0.587615	0.245796	0.327343	0.280768	0.306974
Gradient Boosting	0.87668	0.507597	0.018636	0.430693	0.035725	0.07943
Bagging	0.849688	0.575139	0.211417	0.325743	0.256414	0.293951
CatBoost	0.873018	0.536238	0.090072	0.416956	0.148142	0.241597
LightGBM	0.875499	0.519547	0.047981	0.429942	0.086328	0.165864

Table 5: Model Classification Report with Tomek Link undersampling

Model Name	Accuracy	Recall	Precision	F1 Score	F2 Score	ROC AUC
Logistic Regression	0.665319	0.961872	0.263241	0.413357	0.628350	0.846899
Gaussian Naive Bayes	0.667590	0.964014	0.264860	0.415549	0.630923	0.824724
Decision Tree	0.771531	0.615080	0.293745	0.397605	0.504666	0.704235
Random Forest	0.748214	0.778837	0.298216	0.431291	0.588988	0.834246
Gradient Boosting	0.687572	0.938845	0.274006	0.424206	0.632103	0.851881
Bagging	0.782481	0.633394	0.310300	0.416538	0.524226	0.816258
CatBoost	0.753190	0.819000	0.308895	0.448596	0.615661	0.851047
LightGBM	0.718307	0.898361	0.290292	0.438795	0.633123	0.852391

Table 6: Model Classification Report with SMOTE-ENN sampling

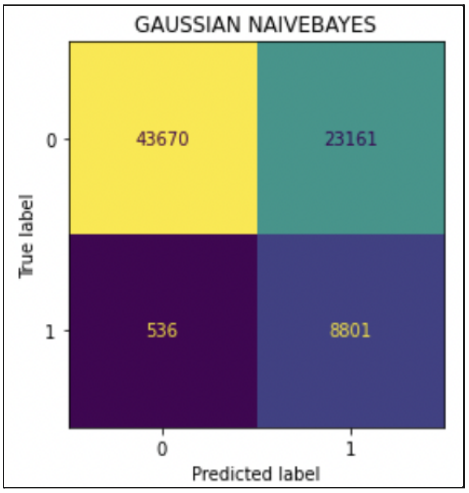

Figure 10

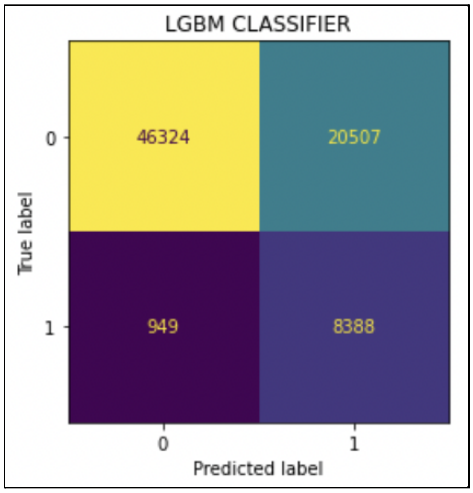

Figure 11

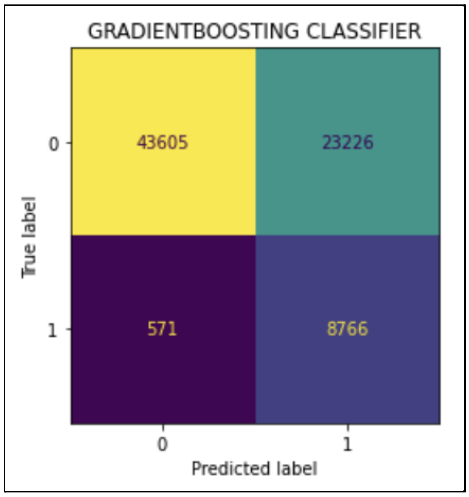

Figure 12

Fig. 10 Shows the confusion matrix of Gaussian Naive Bayes model trained on training set resampled using SMOTE. **Fig. 10** Shows the confusion matrix of the LightGBM model trained on a training set resampled using SMOTE-ENN. **Fig. 10** Shows the confusion matrix of the Gradient Boosting model trained on a training set resampled using SMOTE-ENN

Among the sampling techniques SMOTE-ENN sampling provided the highest improvement in all the tree-based models while SMOTE had the highest improvement in Logistic Regression and Gaussian Naive Bayes models.

The top three performing models were SMOTE sampled Gaussian Naive Bayes, SMOTE-ENN sampled LightGBM and SMOTE-ENN sampled Gradient Boosting classifier. Their confusion matrices are depicted in **Fig 10, 11** and **12**.

Hyperparameter tuning was done to the ensemble models using a combination of different estimators and the depth of the ensemble models. This led to no improvement in model performance, making the SMOTE-trained Gaussian Naive Bayes model the best performing classifier.

The feature importance chart of the Gaussian Naive Bayes model is shown in **Fig. 13**. It can be observed that Previous Insurances, Vehicle Damages and policy channel number 152 have high predictive power in the finalized model.

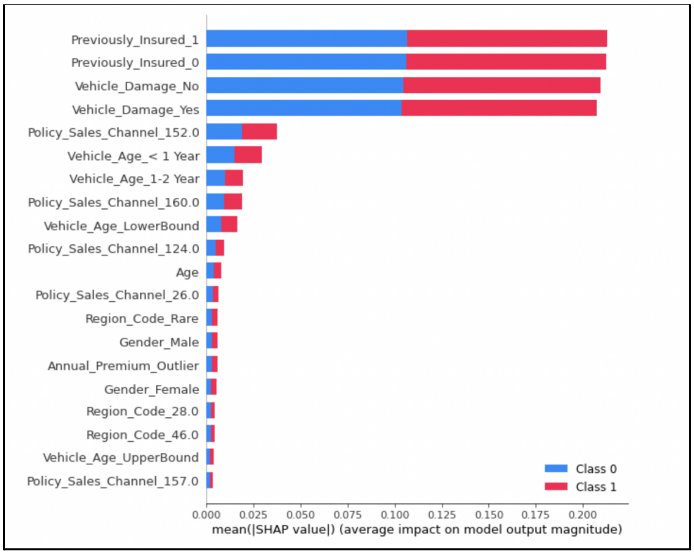

Figure 13: Gaussian Naive Bayes Feature Importance Chart

8. Conclusions

Upon Exploratory Data Analysis, we found that there was a strong association in cross-sale response with the customer's vehicle's age, existing vehicle damages and the presence of any ongoing vehicle insurance. We found that most customers that purchased vehicle insurance had existing damages in their vehicles, were not previously insured and owned vehicles aged over a year. It was also observed that most policyholders of the health insurance company come from three region codes: 28, 8 and 41. The policy sales channels most preferred to reach out to the clients were channel numbers 152, 26 and 124. Oversampling technique and a combination of oversampling and undersampling techniques provided the best results when training models on an imbalanced dataset.

Upon model deployment, tuning and evaluation, we found that the best performing model was the SMOTE sampled Gaussian Naive Bayes classifier.

This classifier had an F2 score of 0.634614, precision of 27.58 per cent and recall of 94.02 per cent. The finalized model's predictions are dependent on the customer's previous insurance status and existing damages on the vehicle.

When the dataset was explored earlier in the project, approximately, one in every ten clients had a positive cross-sale response when they were approached for cross sale. From the precision and recall scores, we can tell that the model has correctly identified 94.02 per cent of the buyers and has a success rate of 27.58 per cent in predicting positive cross-sale responders. This means that, approximately, three out of every ten predicted buyers produced a positive response.

Therefore, the model not only helped identify a large part of the potential vehicle insurance buyers among the existing list of policyholders but also increased the success rate of cross-sales, helping the company save a significant amount of time and resources by generating better leads.

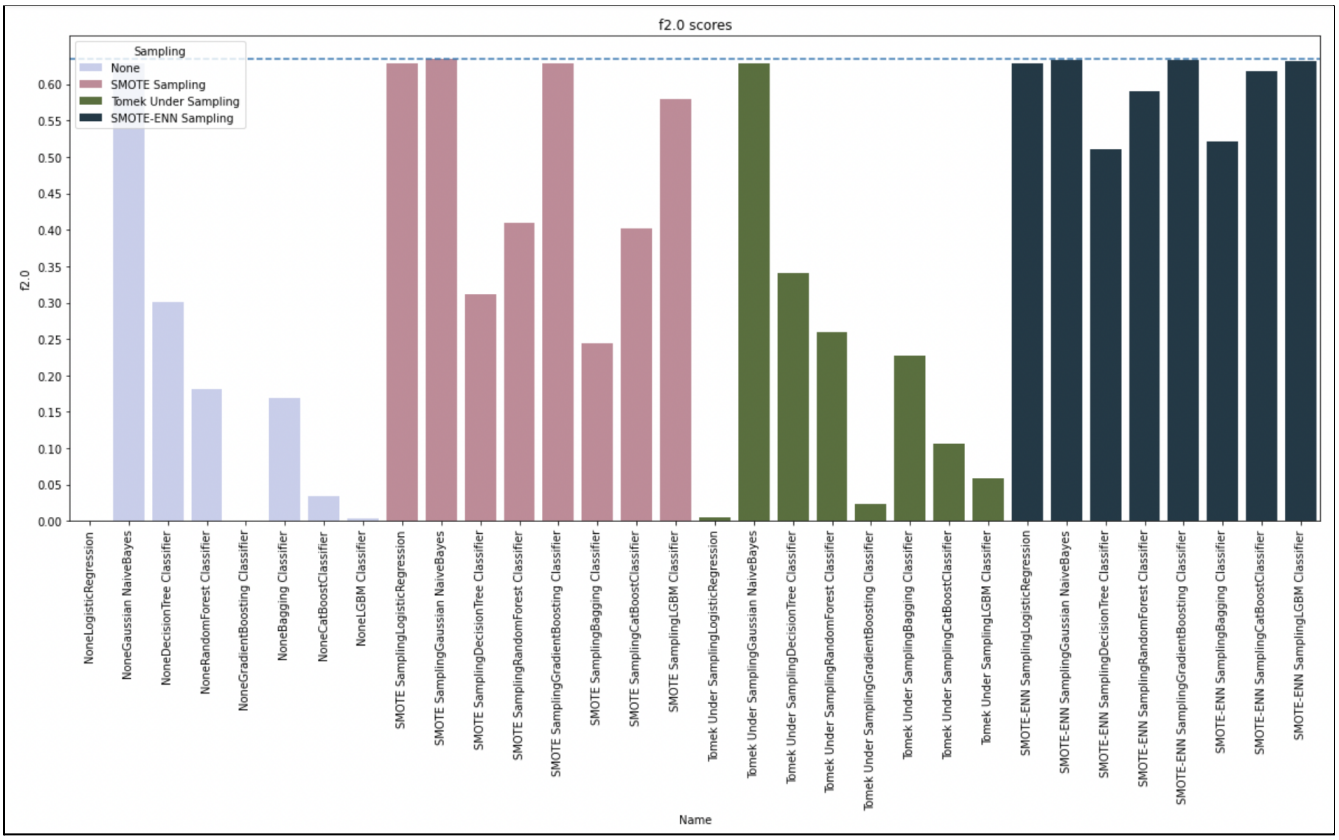

Figure 14: F2 Score Chart of Classifiers trained in different samples

9. References

1. Christopher M. Bishop, "Pattern Recognition and Machine Learning", Pg. 137-139
2. Zhi-Hua Zhou, "Ensemble Methods Foundations and Algorithms", Pg. 57-58
3. John T. Hancock and Taghi M. Khoshgoftaar, "CatBoost for big data: An Interdisciplinary Review"
4. Essam Al Daoud, "Comparison between XGBoost, LightGBM and CatBoost Using a Home Credit Dataset".
5. Jason Brownlee, "Imbalanced Classification with Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning".