Health Insurance Cross Sell Prediction

Shadab Shaikh,
Data Science Trainee,
AlmaBetter, Bangalore

Abstract

Cross-selling is the act of selling products that are related or complementary to the one(s) your current customers already own or use. For example, you cross-sell prescription drug coverage to an existing Medicare customer as a way to keep costs down. Another example might be offering an auto insurance policy to a current homeowners insurance client to bundle their policies and help keep costs down.

Going through customer databases and coming up with a prediction of which customers are more likely to be interested in buying similar products is essential for businesses to strategize and scale their businesses.

1. Introduction

An insurance policy is an arrangement by which a company undertakes to provide a guarantee of compensation for 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.

Just like medical insurance, there is vehicle insurance where every year a customer needs to pay a premium of a certain amount to the insurance provider company so that in case of an unfortunate accident by the vehicle, the insurance provider company will provide a compensation (called ‘sum assured’) to the customer.

To cross-sell insurance effectively, you need to be able to identify opportunities. One way to do this is by taking a look through your client database.

2.Problem Statement

Our client is an Insurance company that has provided Health Insurance to its customers. Now they need your help in building a model to predict whether the policyholders (customers) from past year will also be interested in Vehicle Insurance provided by the company.

Building a model to predict whether a customer would be interested in Vehicle Insurance is extremely helpful for the company because it can then accordingly plan its communication strategy to reach out to those customers and optimize its business model and revenue.

Now, in order to predict whether the customer would be interested in Vehicle insurance, you have information about demographics (gender, age, region code type), Vehicles (Vehicle Age, Damage), Policy (Premium, sourcing channel) etc.

3. Data Description

Our dataset contains customer demographics, vehicle information, policy information and customer response to cross sell surveys.

Attribute Information:

- id : Unique ID for the customer
- Gender : Gender of the customer
- Age : Age of the customer
- Driving_License: 0 : Customer does not have DL, 1 : Customer already has DL
- Region_Code : Unique code for the region of the customer
- Previously_Insured : 1 : Customer already has Vehicle Insurance, 0 : Customer doesn't have Vehicle Insurance
- Vehicle_Age : Age of the Vehicle
- Vehicle_Damage :1 : Customer got his/her vehicle damaged in the past. 0 : Customer didn't get his/her vehicle damaged in the past.
- Annual_Premium : The amount customer needs to pay as premium in the year
- PolicySalesChannel : Anonymized Code for the channel of outreaching to the customer ie. Different Agents, Over Mail, Over Phone, In Person, etc.
- Vintage : Number of Days, Customer has been associated with the company
- Response : 1 : Customer is interested, 0 : Customer is not interested

Duplicate Observations:

When similar entries are not all in the same set, duplicate entries can sabotage the separation of the train, validation, and test sets. This may result in skewed performance estimations, which could make final models perform poorly. In our dataset, however, we have no duplicate rows present.

Missing Values:

The absence of data reduces statistical power, which refers to the probability that the test will reject the null hypothesis when it is false.The lost data may introduce bias into the parameter estimate process. It might make the samples less representative. Our dataset does not contain any missing values, which saves us some hassle and favors model building.

Exploratory Data Analysis, Data Cleaning and Feature Manipulation

The exploratory data analysis that we performed on our train dataset helped us to realize how different features in our dataset influence the target variable.

Outlier Treatment:

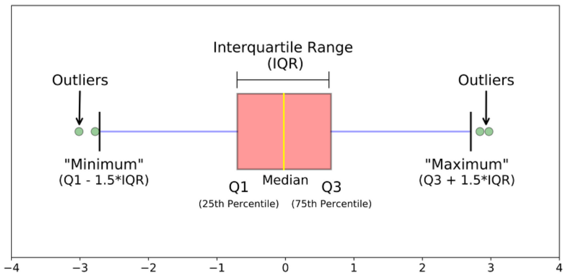
Many times in a large dataset, some features may contain values that differ significantly from other observations. In statistical terminology, such values are called Outliers.

These values may be included in the dataset due to variability in data, data entry errors, measurement errors or experimental errors.

Outliers may lead to various adverse impacts on the dataset, such as, affecting the normality of features, stretching mean and standard deviation of the data, raising problems during statistical analysis, etc.

Thus treating these outliers becomes the utmost priority before building the models.

In this project we used a method called Outlier removal using Inter Quartile Range (IQR).



Interquartile Range score is a measure of

statistical dispersion, being equal to the difference between the75th percentile and 25th percentile i.e., third quartile(Q3) and first quartile(Q1). We identify the outliers as values less than Q1 - (1.5 * IQR) or greater than Q3+(1.5 * IQR).

IQR=Q3-Q1

UNIVARIATE ANALYSIS:

Univariate analysis is an analysis used on one variable with the aim of finding out and identifying the characteristics of the variable. This analysis is the most basic analysis technique that is often used in various types of research. We can see whether the data we use at a glance is normally distributed, left-winged, right handed, there are outliers, etc. Knowing the size of concentration, size of distribution, and mean-median from a data set.

BIVARIATE ANALYSIS:

Bi-variate Analysis finds out the relationship between independent variables with single dependent variables. Using bi-variate analysis association and dissociation between variables at a predefined significance level. We can perform bi-variate analysis for any combination of categorical and continuous variables.

Feature Engineering

The approach of leveraging domain expertise to extract features (characteristics, qualities, and attributes) from unprocessed data is known as feature engineering or feature extraction. The goal is to employ these additional features to enhance the quality of machine learning process results as opposed to only giving the process raw data.

Building new features:

Feature engineering should be carried out with specific domain knowledge, so that these prove to be valuable from a business point of view.

From our dataset, we employed the 'Region_Code' column to extract the feature called 'Region' by binning the values in the original column and getting 5 distinct categories, making it a categorical feature.

Encoding Categorical Features:

Converting categorical data into integer format is the process of encoding categorical data so that the data with transformed categorical values can be delivered to the various models.

We used the following methods to encode categorical features:

Label Encoding:

Label encoding is the process of transforming the labels into a numeric form so that they may be read by machines.

We performed label encoding some features yielding the following results

- Gender: 1 = Male, 0 = No Female
- Vehicle_Damage: 1 = Yes, 0 = No

One Hot Encoding:

Data can be converted using one hot encoding as a means of getting a better prediction and preparing the data for an algorithm. With one-hot, we create a new category column for each categorical value and give it a binary value of 1 or 0.

We performed One hot encoding on:

- 'Vehicle_Age' column, to get new features i.e. 'Vehicle_Age_1-2 Year', 'Vehicle_Age_< 1 Year' and 'Vehicle_Age_> 2 Years'
- 'Region' column, to get new features i.e. 'Region_A', 'Region_B', 'Region_C', 'Region_D' and 'Region_E'

Model Building

Prerequisites:

Train-Test Split

Train test split is a model validation process that allows you to simulate how your model would perform with new data. The train_test_split() method is used to split our data into train and test sets.

Handling Class Imbalance

An imbalance occurs when one or more classes have very low proportions in the training data as compared to the other classes. In our dataset, the predictor variable 'Response' suffers from a heavy class imbalance, with the class '0' having 334399 or 87.74% observations and the class '1' having 46710 or 12.26% observations.

The Metric Trap:

One of the major issues when dealing with unbalanced datasets relates to the metrics used to evaluate our model. Using simpler metrics like accuracy score can be misleading. In a dataset with highly unbalanced classes, the classifier will always "predict" the most common class without performing any analysis of the features and it will have a high accuracy rate, obviously not the correct one. Hence we need to address the imbalance in the classes.

There are various ways to handle class imbalance problem, one of which that we are going to utilize for the scope of this project is called SMOTE.

SMOTE (Synthetic Minority Oversampling Technique) works by randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbors.

Feature Scaling

Scaling data is the process of increasing or decreasing the magnitude according to a fixed ratio, in simpler words you change the size but not the shape of the data.

Various methods of feature scaling:

1. Standardization

It calculates the z-score of each value and replaces the value with the calculated Z-score. The Z-score can be calculated by the following formula:

$$x' = \frac{x - \bar{x}}{\sigma}$$

Where σ is the variance and x̄ is the mean.The features are then rescaled with x̄ =0 and σ=1.

Library used: StandardScalar

2. Min-Max Scaling:

It is also referred to as Normalization. The features are scaled between 0 and 1. Here, the mean value remains the same as in Standardization, that is, 0. The formula is given as:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Library used: StandardScalar

3. Normalizing

It is used to rescale each sample. Each sample (i.e. each row of the data matrix) with at least one non zero component is rescaled independently of other samples so that its norm (l1 or l2) equals one.

Library used: Normalizer

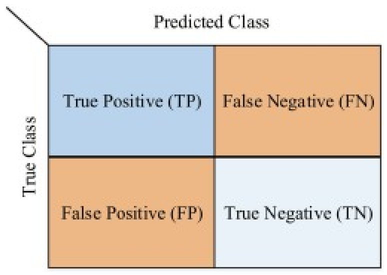
For this project, we utilized Min-Max Scaling.

Evaluation Metrics

Evaluation metrics are used to gauge how well a statistical or machine learning model is performing. Every project has to evaluate machine learning models or algorithms. To measure the performance of a classification model, a wide variety of evaluation metrics are available. We will use the following evaluation metrics to estimate the performance of our models:

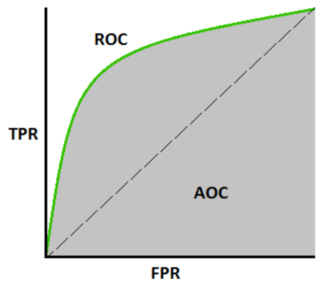
1. Confusion Matrix

Confusion matrix is a very popular measure used while solving classification problems. Confusion matrices represent counts from predicted and actual values. The output “TN” stands for True Negative which shows the number of negative examples classified accurately. Similarly, “TP” stands for True Positive which indicates the number of positive examples classified accurately. The term “FP” shows False Positive value, i.e., the number of actual negative examples classified as positive; and “FN” means a False Negative value which is the number of actual positive examples classified as negative.



2. AUC - ROC curve

AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1.



3. Accuracy

The most common metric used to evaluate the performance of a classification predictive model is classification accuracy. Typically, the accuracy of a predictive model is good (above 90% accuracy), therefore it is also very common to summarize the performance of a model in terms of the error rate of the model.

Classification accuracy is a metric that summarizes the performance of a classification model as the number of correct predictions divided by the total number of predictions.

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

4. Precision

Precision is a metric that quantifies the number of correct positive predictions made. Precision, therefore, calculates the accuracy for the minority class.

It is calculated as the ratio of correctly predicted positive examples divided by the total number of positive examples that were predicted.

$$\text{Precision} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Positive}}$$

$$= \frac{\textit{True Positive}}{\textit{Total Predicted Positive}}$$

5. Recall

Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made.Unlike precision that only comments on the correct positive predictions out of all positive predictions, recall provides an indication of missed positive predictions.In this way, recall provides some notion of the coverage of the positive class.

$$\text{Recall} = \frac{\textit{True Positive}}{\textit{True Positive} + \textit{False Negative}}$$

$$= \frac{\textit{True Positive}}{\textit{Total Actual Positive}}$$

6. F1 Score

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

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

Defining Hyperparameters

Hyperparameters are the parameters that are explicitly defined to control the learning process before applying a machine-learning algorithm to a dataset.

We defined the ranges of hyperparameters to be used in the later stages of model building.

For the scope of this project we used HalvingRandomSearchCV with these models to come up with ideal hyperparameters.

HalvingRandomSearchCV is a method for adjusting hyperparameters to find the best values for a particular model. The search strategy starts evaluating all the candidates with a small amount of resources and iteratively selects the best candidates, using more and more resources. As was already noted, a model's performance is strongly influenced by the value of its hyperparameters. Noting that there is no way to determine the best values for hyperparameters in advance, it is ideal to explore every conceivable value before deciding what the best ones are. We utilize HalvingRandomSearchCV to automate the tweaking of hyperparameters because doing it manually could take a lot of time and resources.

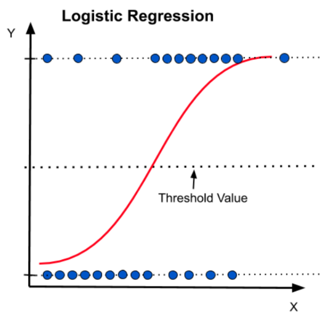
This function aids in fitting your estimator (model) to your training set by looping through predefined hyperparameters. So, from the list of hyperparameters, we may choose the best parameters in the end.

Model Selection:

We used the following models to fit to our data and predict hourly bike counts.

1. Linear Regression

Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure.



Cost function for logistic regression:

Cost(hϴ(x), Y(actual)) = - log (hϴ(x)) if y=1

-log (1- hϴ(x)) if y=0

2. Naive Bayes

It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the basis of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other. It is called Bayes because it depends on the principle of Baye's theorem.

Baye's theorem:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

● P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

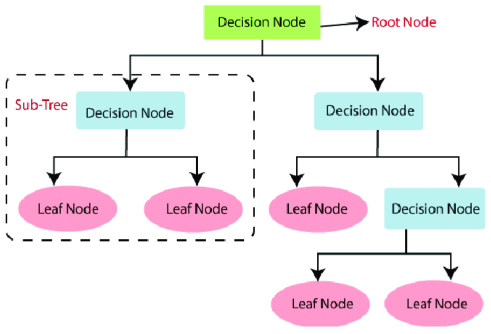
● P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true

● P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

● P(B) is Marginal Probability: Probability of Evidence.

3. Decision Trees

A supervised learning method called the decision tree can be applied to classification and regression problems. It is a tree-structured classifier, where internal nodes stand in for a dataset's features, branches for the decision-making process, and each leaf node for the classification result. The Decision Node and Leaf Node are the two nodes of a decision tree. The CART algorithm, which stands for Classification and Regression Tree algorithm, is used to construct a tree. Both numerical data and categorical data (YES/NO) can be included in a decision tree.



Following terminologies are used for decision trees.

● Root Node: Root node is from where the decision tree starts.

● Leaf Node: Leaf nodes are the final output node

● Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes

● Branch/Sub Tree: A tree formed by splitting the tree.

Ensemble Learning:

Ensemble methods in statistics and machine learning combine several learning algorithms to achieve higher predicted performance than any one of the individual learning algorithms could.

There are two main types of ensemble learning techniques that we will use for the scope of this project:

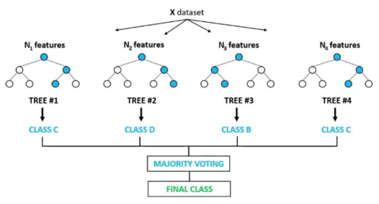
● Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

● Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, XGBoost, Gradient Boosting Machine, LightGBM, CatBoost.

4. Random Forest

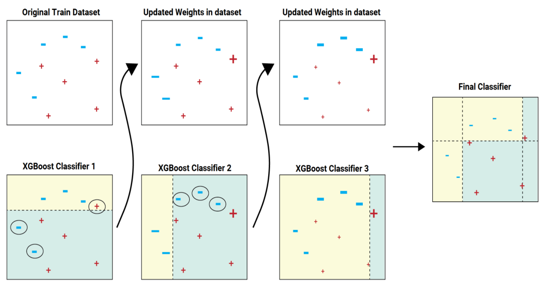
Random Forest is a classifier that uses many decision trees on different subsets of the input dataset and averages the results to increase the dataset's predicted accuracy. Instead of relying on a single decision tree, the random forest takes the prediction from each tree and predicts the outcome based on the majority votes of predictions. Higher accuracy and overfitting are prevented by the larger number of trees in the forest.

Random Forest Classifier



5. XGBoost

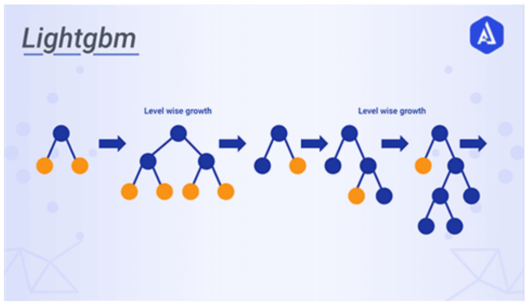
Decision trees are generated sequentially in this algorithm. Weights are important in XGBoost. Each independent variable is given a weight before being fed into the decision tree that predicts results. Variables that the tree incorrectly predicted are given more weight before being placed into the second decision tree. These distinct classifiers/predictors are then combined to produce a robust and accurate model. The boosting ensemble strategies include XGBoost, which combines the weaknesses of primary learners with the following strong and compatible learners.



7. LightGBM

LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel, distributed, and GPU learning.
- Capable of handling large-scale data



8. CatBoost

CatBoost is a recently open-source machine learning algorithm from Yandex. It can easily integrate with deep learning frameworks like Google's TensorFlow and Apple's Core ML. It can work with diverse data types to help solve a wide range of problems that businesses face today. To top it up, it provides best-in-class accuracy.

It is especially powerful in two ways:

- It yields state-of-the-art results without extensive data training typically required by other machine learning methods, and
- Provides powerful out-of-the-box support for the more descriptive data formats that accompany many business problems.

"CatBoost" name comes from two words "Category" and "Boosting".

Model Explainability

Explainability in machine learning means that you can explain what happens in your model from input to output. It makes models transparent and solves the black box problem. Explainability is the degree to which a human can understand the cause of a decision or the degree to which a human can consistently predict ML model results.

Our best model; XGBoost Classifier is a black-box model. We have employed SHAP model explainability technique for the scope of this project

Shapley's value assumes that we can compute the value of the surplus with or without each analyzed factor. The algorithm estimates the value of each factor by assessing the values of its 'coalitions'. In the case of Machine Learning, the 'surplus' is a result of our algorithm and co-operators having different input values. The goal of SHAP is to explain the prediction by computing the contribution of each feature to the final result.

Conclusion

That's it! We reached the end of our exercise.
Starting with loading the data so far we have done EDA, null values treatment, feature engineering, class imbalance handling, and then model building.
In all of these models, our accuracy revolves in the range of 82% to 90%.
Among all the other models used, the XGBoost Classifier gives the best model performance of about 90% on the test data.
We can conclude this performance to be satisfactory, and with more data and careful feature selection from a business point of view, there is a chance of getting an even greater model performance.