**Health Insurance Cross-Sell Prediction**

**Kunal Agrawal**

**Abstract:**

It is a widely known fact that repeat customers are more profitable than new customers. Given the high cost of acquiring new customers, companies today want to sell more items to their existing customer base.

In Insurance Industries, there is a very high cost that comes along with the new customers as there are a lot of commissions involved and thus diminishing initial returns. In most of the cases, a customer only becomes profitable after 3 years of premium payments.

So it makes sense for them to look into their existing portfolio of customers and to predict in which cases they might be able to cross sell or better up-sell their products. Also, since there is cost associated with each and every policy-holder, there is an utter need to target genuine customers who will be willing to avail these services.

***Keywords: machine learning, cross-selling, Predictive modelling, Classification***

**1. 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 the past year will also be interested in Vehicle Insurance provided by the company.

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 compensation (called ‘sum assured’) to the customer.

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 optimise its business model and revenue.

We were provided with information about demographics (gender, age, region code type), Vehicles (Vehicle Age, Damage), Policy (Premium, sourcing channel) etc in order to predict customers who would be interested in buying

Various variables provided are as:

* 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

**2. Introduction**

The cost of servicing existing customers is comparatively very less when compared to acquiring new customers. It is normal practice in the industry to foster stronger relationships with their customers and thus does cross-selling or up-selling of their various products to them.

### Our goal here is to build a predictive model, which could help our client in predicting the policy-holders willing to take Motor Insurance proactively.

## **3. Cross-Sell Dependency**

The Sales of any store can normally depend basically on a number of factors:

* Customers possessing vehicle
* Need of customers: previous insurance validity near expiration, buying a new vehicle
* Local Government’s mandate for buying motor insurance

# **4. Factors Affecting Motor Insurance Sales as per dataset**

## **Gender**

## There was not much difference in Motor Insurance buying interest in terms of Gender. However, a slightly greater proportion of males (around 13.8% in comparison to 10.4% for females) were interested in Motor Insurance.

## **Driving License**

## It was clearly observed in the dataset that if a customer possesses a Driving License then he/she is willing to take Motor insurance. Most of the cases included were of persons who had Driving License.

* **Previously Insured**

Most of the policy-holders who were interested in Motor Insurance were those who were not previously insured. Among the people who did not have Motor Insurance, more than 22% were willing to take Motor Insurance.

* **Vehicle Age**

It was observed that most customers who were willing to take Motor Insurance were having old vehicles with Age greater than 1 year. For greater than 2 years, around 30% and for vehicles between the age one to two years, 18% people were interested.

* **Vehicle Damage**

Most of the policyholders who were interested in willing to take Health Insurance, were the people with damaged vehicles. Around 24% people with damaged vehicles were interested in buying in motor insurance with the company.

**4. Steps involved:**

* **Exploratory Data Analysis**

After loading the dataset we performed this method by comparing our target variable that is with other independent variables. This process helped us figuring out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

* **Duplicate Values Elimination**

There were around 269 values present in our dataset containing duplicate values and we eliminated those values to make our model robust.

* **Null values Treatment**

We checked our datasets for Null values, there were no null values present in our dataset, Hence, no null value treatment was required for modelling purposes

* **Outlier Treatment**

We checked for outliers for the Vintage column, however, no outliers were observed for the same.

* **Discretization of Age and Annual Premium**

We performed Discretization on Age and Annual premium columns to convert it into discrete form. We did this by creating a set of contiguous intervals (or bins) that go across the range of our desired variables.

* **Encoding of categorical columns**

We used One Hot Encoding to produce binary integers of 0 and 1 to encode our categorical features as there was no specific order for those columns and to make our model better predict results we changed the entries into various columns.

We also encoded “Vehicle Damage” and “Gender” into 0 and 1 to convert it into numerical values.

* **Feature Selection**

We removed Driving\_License variable as there was a huge data imbalance present. Moreover, most of the policy-holders had driving licenses and they were the ones who were willing to buy Motor Insurance.

* **Standardization of features**

We tried to scale our data into a uniform format that would allow us to utilize the data in a better way while performing fitting and applying different algorithms to it. That is why we implemented MinMaxScaler on our features.

By using this, a level of consistency or uniformity was maintained within the selected environment.

* **Handling Class Imbalance**

There was a huge imbalance in the class as few policy-holders were interested in Motor policy. Hence, we used SMOTE technique to balance the class balance present in the dependent variable.

* **Fitting different models**

For modelling, we tried various classification algorithms like:

1. **Logistic Regression**
2. **Decision Tree Classifier**
3. **Random Forest Classifier**
4. **Light GBM Classifier**
5. **Naive Bayes Algorithm**
6. **XG Boost**
7. **Stacking** 
   1. **Level1: Logistic Regression, Random Forest, Light GBM**
   2. **Level2: XGBoost**

* **Tuning the hyperparameters for better accuracy**

Tuning the hyperparameters of respective algorithms is necessary for getting better accuracy and to avoid overfitting in case of tree-based models.

We also experimented with various Bagging function values and different classifiers.

* **Feature Importance**

We also checked for Feature Importance for various tree-based models using importances.

**7.1. Algorithms:**

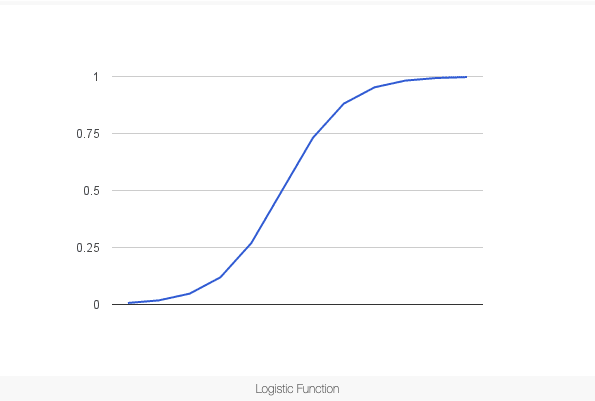
1. **Logistic Regression:**

It is one of the most sought classification methods mostly in the case of binary classification. One of the most important functions that we should know is the Logistic function.

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It’s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value), where e is the base of the natural logarithms



Input values (x) are combined linearly using weights or coefficient values to predict an output value (y). However, in logistic regression output value being modelled is a binary value (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

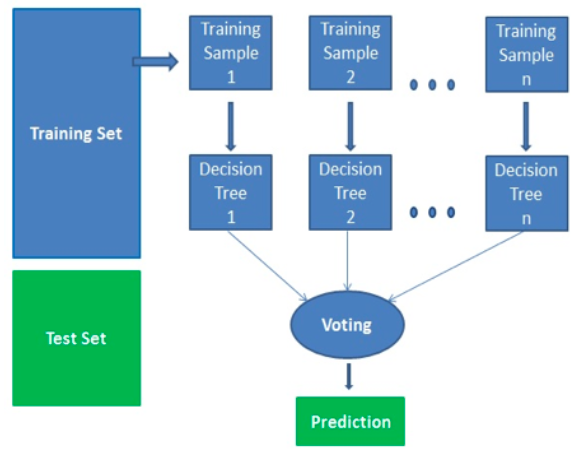
y = e^(b0 + b1\*x) / (1 + e^(b0 + b1\*x))

Logistic Regression models the probability for the base class. Finally, we snap the probabilities to a binary class value.

1. **Random Forest Classifier:**

Random Forest is a supervised machine learning algorithm which means it is used on a labelled dataset. We can say that just like a forest which comprises many trees. Random forest algorithm comprises many decision trees.

It is basically an ensemble technique which is based on the philosophy of divide and conquer methodology. It generates small decision trees using random subsamples of the dataset where the collection of the generated decision tree is defined as forest. Every individual tree is created using an attribute selection indicator such as gini Index, entropy and information gain etc.



1. **XGBoost Classifier**

Extreme Gradient Boosting provides an efficient and effective implementation of the gradient boosting algorithm. It is an open-source library. It can be directly used for regression predictive modelling.

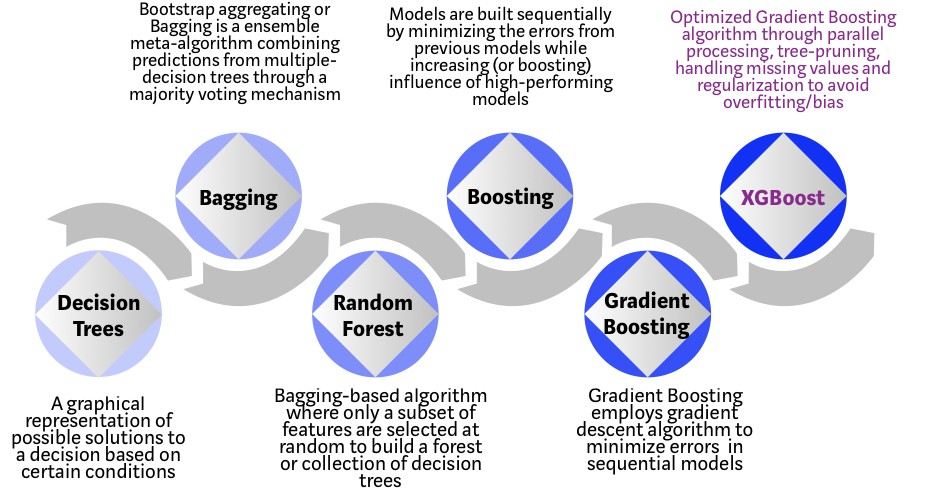
Let’s first talk about Gradient boosting. It is a class of ensemble machine learning algorithms that can be used for classification or regression predictive modelling problems.

Ensembles are constructed from decision tree models. Weak learners are added one at a time to the ensemble and fit to correct the prediction errors made by prior models. It is referred to as boosting.

Extreme Gradient Boosting is designed to be both computationally efficient and highly effective as it runs parallelly by incorporating all the cores of the system.

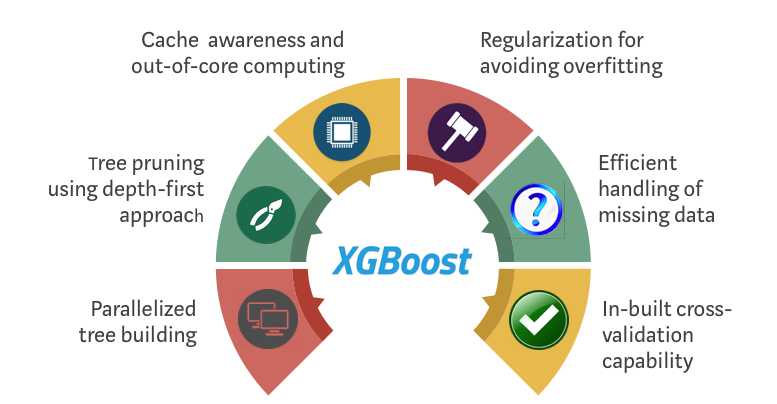
It can run on both single and distributed systems.

Let’s see the evolution of XG Boost.



Source:https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

How XGBoost optimizes GBM Algorithm



Source:https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

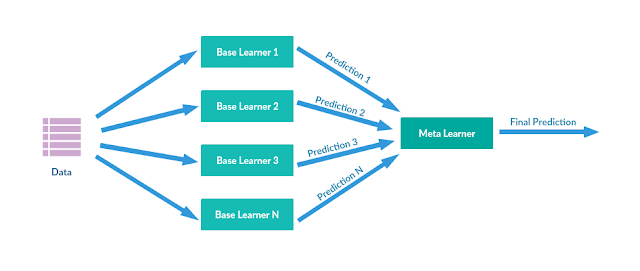
1. **Light GBM**

LightGBM is a gradient boosting framework based on decision trees to increase the efficiency of the model and reduce memory usage. It uses two novel techniques: Gradient-based One Side Sampling and Exclusive Feature Bundling (EFB) which fulfils the limitations of histogram-based algorithm that is primarily used in all GBDT (Gradient Boosting Decision Tree) frameworks.

LightGBM extends the gradient boosting algorithm by adding a type of automatic feature selection as well as focusing on boosting examples with larger gradients. This can result in a dramatic speedup of training and improved predictive performance.

1. **Stacking**

Stacking uses one or more models at the first level, makes predictions at the first level using these models and then uses these predictions as features to fit one or more second-level models on the top. To avoid overfitting, cross-validation is usually used to predict the OOF (out-of-fold) part of the training set.



1. **Naive Bayes**

We need to first talk about Bayes’ Theorem before talking about Naive Bayes Algorithm.

Bayes’ Theorem provides a way that we can calculate the probability of a piece of data belonging to a given class, given our prior knowledge. Bayes’ Theorem is stated as:

P(class|data) = (P(data|class) \* P(class)) / P(data)

where P(class|data) is the probability of class given the provided data.

Naive Bayes is a classification algorithm for binary (two-class) and multiclass classification problems. It is called Naive Bayes or idiot Bayes because the calculations of the probabilities for each class are simplified to make their calculations tractable.

Rather than attempting to calculate the probabilities of each attribute value, they are assumed to be conditionally independent given the class value.

This proves to be a very strong assumption that is most unlikely in real data which means that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

**7.2. Model performance:**

The model can be evaluated by various metrics such as:

1. **Confusion Matrix and Classification Matrix:**

The Confusion Matrix is NXN matrix containing the number of correct and incorrect predictions. The rows of the matrix represent the real classes, while the columns represent the predicted classes.

Categories present in Confusion Matrix are:

1. True Positive (TP): The model predicted positive, and the real value is positive.
2. True Negative (TN): The model predicted negative, and the real value is negative.
3. False Positive (FP): The model predicted positive, but the real value is negative (Type I error).
4. False Negative (FN): The model predicted negative, but the real value is positive (Type II error).

Following Metrics are used to monitor the performance of the model:

1. **Accuracy**: It returns the proportion of correct predictions.

(TP + TN)/(TP + FP + TN + FN)

1. **Precision**: It returns the proportion of true positives among all values predicted as Positives.

TP/(TP +FP)

1. **Recall**: It returns the proportion of Positive values correctly predicted.

TP/(TP + FN)

1. **Specificity**: It returns the proportion of Negative Values correctly predicted.

TN/(TN + FP)

1. **F1-Score**: It is the Harmonic mean of Precision and Recall. Often used to compare classifiers.

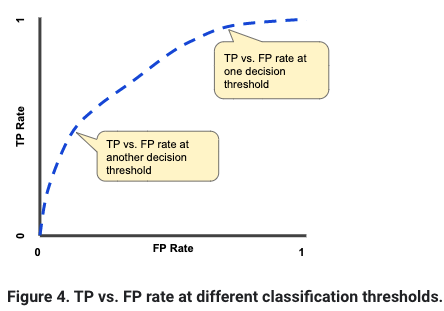
(2 x Precision x Recall)/(Precision + Recall)

* **ROC-AUC Scores:**

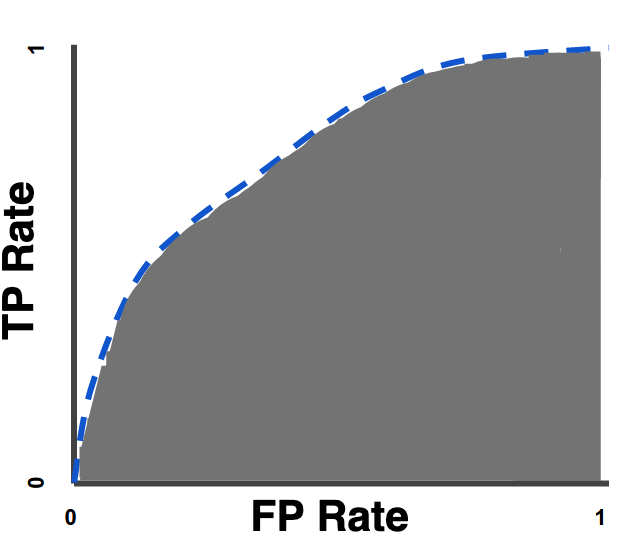
A ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

1. True Positive Rate
2. False Positive Rate

Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.



AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve. The higher the AUC Scores, the better our model is able to predict the results.



**7.3. Hyper Parameter Tuning:**

Hyperparameters are sets of information that are used to control the way of learning an algorithm. Their definitions impact parameters of the models, seen as a way of learning, change from the new hyperparameters. This set of values affects the performance, stability and interpretation of a model. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem. Hyperparameters alter the way a model learns to trigger this training algorithm after parameters to generate outputs.

We used Grid Search CV and Randomized Search CV for hyperparameter tuning. This also results in cross-validation and in our case we divided the dataset into different folds.

1. **Grid Search CV-**Grid Search combines a selection of hyperparameters established by the scientist and runs through all of them to evaluate the model’s performance. Its advantage is that it is a simple technique that will go through all the programmed combinations. The biggest disadvantage is that it traverses a specific region of the parameter space and cannot understand which movement or which region of the space is important to optimize the model.
2. **Randomized Search CV-** In Random Search, the hyperparameters are chosen at random within a range of values that it can assume. The advantage of this method is that there is a greater chance of finding regions of the cost minimization space with more suitable hyperparameters since the choice for each iteration is random. The disadvantage of this method is that the combination of hyperparameters is beyond the scientist’s control

**8. Conclusion:**

Starting with loading the data so far we have done EDA, null values treatment, encoding of categorical columns, feature selection and then model building.

The model scores of various implemented models were as follows:

| **Model** | **Train AUC-ROC Score** | **Test AUC-ROC Score** | **Recall TEST for Class 1** | **Conclusion** |
| --- | --- | --- | --- | --- |
| Logistic Regression | 0.7963 | 0.7974 | 0.92 | Basic Model. |
| Logistic Regression (Hyperparameter Tuned) | 0.7963 | 0.7974 | 0.92 | Giving results almost similar without using hyper-parameters |
| Decision Tree | 0.8083 | 0.8087 | 0.89 | Moderate score. No parameter tuning was employed. Neither under fit nor overfit |
| Decision Tree (Hyperparameter Tuned) | 0.9703 | 0.8630 | 0.86 | Better ROC Score. Huge over-fitting approved |
| Random Forest | 0.8191 | 0.8190 | 0.93 | Not much improvement over basic decision tree. Neither under fit nor overfit |
| Light GBM | 0.8385 | 0.8382 | 0.92 | High improvement after using Gradient Boosting algorithm. Neither under fit nor overfit |
| Naive Bayes | 0.7973 | 0.7983 | 0.88 | Lower score |
| XGBM | 0.9144 | 0.8985 | 0.92 | Best Model |
| Stacking | 0.8385 | 0.8391 | 0.92 | Moderate score in comparison to other Esemble Techniques |

Data was fit properly within the dataset and neither underfitting nor overfitting was observed in the dataset. As evident from the above set, Models achieved very high scores. Xtreme GBM has very close model scores. Light GBM and Stacking is giving almost similar results.

**References-**

1. MachineLearningMastery
2. Greatlearning
3. GeeksforGeeks
4. Analytics Vidhya
5. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d
6. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc