Open IIT Data Analytics

09.09.2019

—

Team Name - RunTime Terror

Shubham Ekapure

Naman Paharia

Varun Madhavan

Arusarka Bose

Umang Aditya

INDEX

THE INDUSTRY ANALYSIS

Vehicle insurance (commonly known as car insurance, motor insurance, or auto insurance) is insurance for cars, trucks, motorcycles etc. Its primary use is to provide financial protection against physical damage or bodily injury resulting from traffic collisions and against liability that could also arise from incidents in a vehicle. Vehicle insurance additionally offers financial protection against theft of the vehicle, and against damage to the vehicle sustained from events other than traffic collisions, such as keying, weather or natural disasters, and damage sustained by colliding with stationary objects.

The Automobile Insurance industry consists of Personal liability insurance, personal collision/comprehensive insurance, commercial liability insurance, commercial collision/comprehensive insurance and investment income.

This industry makes revenue through the fear of any risk in the public, basically, a higher number of insurance buyers and lesser claims means more revenue for the industry. Hence, the terms and conditions regarding any auto-insurance play a very important role. The future looks promising for the life insurance industry with several changes in the regulatory framework which will lead to further change in the way the industry conducts its business and engages with its customers.

THE FEATURES

To predict Customer Lifetime value most important features are:

Monthly Premium Auto -It is the premium paid by the customer to the company per month.
It has a correlation of 0.62
It has a mean value of 93.29

Months since policy inception - The total number of month customer pays the premium to the company.
It has a mean of 48.06

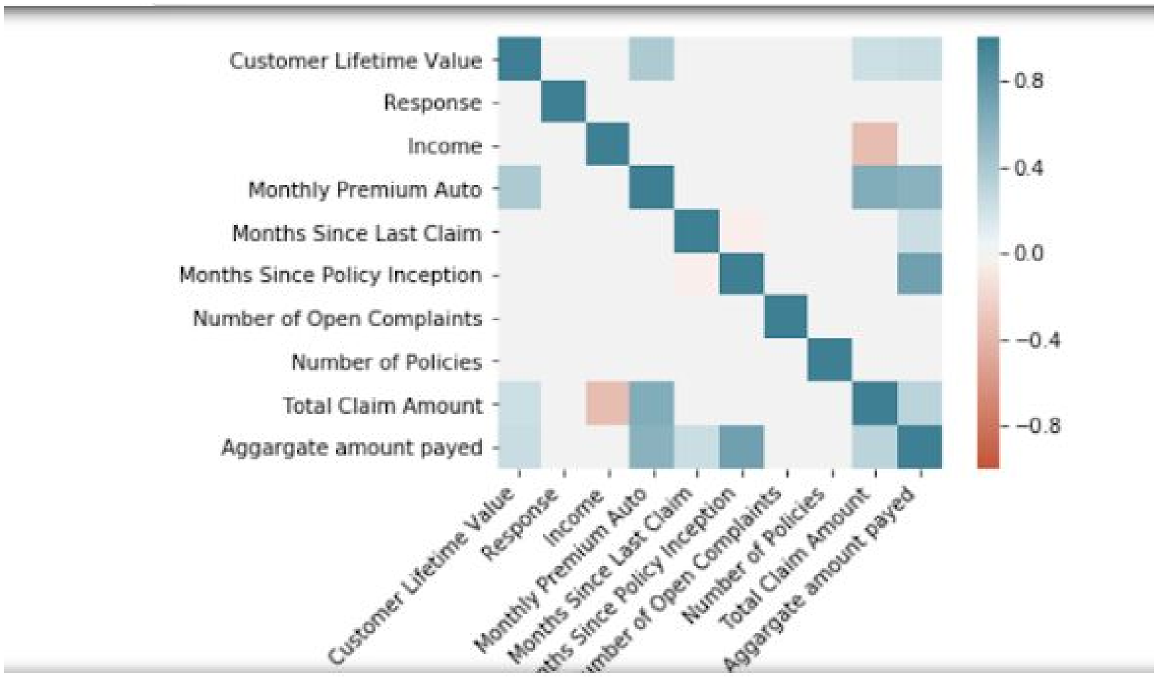
Income of the customer - income varies from 0 to 99981, has a standard deviation of 30380, mean value of 37657 and skewness of 0.2868.

Total claim amount - The money claimed by the policyholder from the company. Mean value- 434.09, standard deviation of 290.5, correlation with CLV is 0.54.

The number of open complaints - The company has maximum profit from the customers who have 0 or 1 open complaints with an aggregate of $55M. Clearly the open complaints determine how well does the company behave with the customer.

Vehicle - The type and size of vehicle affects the CLV prominently. 50% of the insurance holders have four-door car, and it corresponds to 33% of aggregate money made by the company. The average CLV of luxury vehicles is more than normal ones. 70% of the customers have a med-size vehicle and Large vehicles are prone to higher claims.

Gender - Gender composition is almost equal with 51% of females in the dataset.



This is the Pearson's r correlation plot.The Pearson product-moment correlation coefficient is a measure of the strength of the linear relationship between two variables. It is referred to as Pearson's correlation or simply as the correlation coefficient. With only the numerical variables being taken into consideration, this plot justifies the choice of variables in the final model. Few categorical variables like Response, Coverage are also quantified and are further proved to be a good measure of evaluation.

MODELS APPLIED

CLV is a continuous variable and based on its relationship with other features of the data-set like income, employment etc. we can apply various machine learning models to it.

Multivariate Linear Regression

- The model didn't perform as expected because of non-linear relationship which could be observed with most of the features like months since the last claim, the number of open complaints. But a positive accuracy meant that we had features which followed linear relationship like monthly premium.

Decision Tree Model

- Decision trees can sometimes be beneficial for regressions as well. The roots divided on the basis of root squared error and hence we got a regression tree. Effective features like location code, marital status, employment came into the scheme of model development and that increased the accuracy by 40%. The error in the prediction was also diminished but the chances of high variance and overfitting increase if the depth is increased. Even a small change in input data can at times, cause large changes in the tree. Decision trees moreover, examine only a single field at a time, leading to rectangular classification boxes.

This model gave us an R2 score of 0.99 (overfitting) on the train set and 0.47 on the test set.

Gradient Boosting Regressor

- A model that uses boosting with simple Decision Trees.

This model gave us an r2 score of 0.73 on the train set and 0.68 on the test set.

XGBoost

- XGBoost is an implementation of Gradient Boosted Decision Trees that have given a spectacular performance in Kaggle competitions.

This model gave us an r2 score of 0.96 on the train set and 0.71 on the test set.

Neural Network - We also tried to fit a fully connected neural network to the data, choosing the number of layers and the number of units per layer using hyperopt trained to minimize root mean square error.
This model gave us an r2 score of 0.68 on the train set and 0.47 on the test set.

Random Forest Regressor - Random forests are a strong modelling technique and much more robust than a single decision tree. They aggregate many decision trees to limit overfitting as well as error due to bias. The performance was boosted up by 60%, the MSE dropped significantly and overall without any features scaling this was the best model as per as the accuracy. Inclusion of many trees based on unrelated categorical features like gender, response and policy type enhanced the performance.
We also used the hyperopt library for Bayesian Hyperparameter Optimization for the Random Forest Model.
This model gave us an r2 score of 0.96 on the train set and 0.71 on the test set.

DATA CLEANSING AND VARIABLE SELECTION

DATA CLEANING:

We analysed the data and calculated the Z Score. After that, the outliers with z-score>3 were removed but as a consequence, the overall performance of the model was decreased and so it was pretermitted.

The variable of customer ID and Effective to date were dropped. Customer ID, each being unique didn’t have any relationship whatsoever. The effective date corresponds to all the policies that die out in January and February i.e. basically a portion of the whole data that the company provided.

VARIABLE SELECTION:

We found the model weights and that Number of Policies, Monthly Premium Auto, Vehicle Class, Total Claim Amount and Income had the highest correlation

We did standardisation for preprocessing of the data.

Based on the monthly premium and the number of months of policy inception we figured out a new variable called *Aggregate amount paid* which was basically the product of total months and monthly premium. We took into account the fact that some people extend their policies as-well. This resulted in improved accuracy of 0.22%.

MODEL USED

Random Forest Regressor - Random Forest Regressor has been used as the running model of the python code.
Random Forest is an ensemble machine learning technique capable of performing both regression and classification tasks using multiple decision trees and a statistical technique called bagging. Bagging along with boosting are two of the most popular ensemble techniques which aim to tackle high variance and high bias. A RF instead of just averaging the prediction of trees it uses two key concepts that give it the name random:

1. Random sampling of training observations when building trees
2. Random subsets of features for splitting nodes

In other words, Random forest builds multiple decision trees and merge their predictions together to get a more accurate and stable prediction rather than relying on individual decision trees.

Random sampling of training observations:

Each tree in a random forest learns from a random sample of the training observations. The samples are drawn with replacement, known as bootstrapping, which means that some samples will be used multiple times in a single tree. The idea is that by training each tree on different samples, although each tree might have high variance with respect to a particular set of training data, overall, the entire forest will have lower variance but not at the cost of increasing the bias. In Sklearn implementation of Random forest the sub-sample size of each tree is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True. If bootstrap=False each tree will use exactly the same dataset without any randomness.

Random Subsets of features for splitting nodes:

The other main concept in the random forest is that each tree sees only a subset of all the features when deciding to split a node. In Sklearn this can be set by specifying max_features = sqrt(n_features) meaning that if there are 16 features, at each node in each tree, only 4 random features will be considered for splitting the node.

General Working Steps:

Step 1: Samples are taken repeatedly from the training data so that each data point is having an equal probability of getting selected, and all the samples have the same size as the original training set.
Let's say we have the following data:
x= 0.1,0.5,0.4,0.8,0.6, y=0.1,0.2,0.15,0.11,0.13 where x is an independent variable with 5 data points and y is the dependent variable.
Now Bootstrap samples are taken with replacement from the above data set.
n_estimators is set to 3 (no of tree in random forest), then:
The first tree will have a bootstrap sample of size 5 (same as the original dataset), assuming it to be:
x1={0.5,0.1,0.1,0.6,0.6} likewise
x2={0.4,0.8,0.6,0.8,0.1}
x3={0.1,0.5,0.4,0.8,0.8}

Step 2: A Random Forest Regressor model is trained at each bootstrap sample drawn in the above step, and a prediction is recorded for each sample.

Step 3: Now the ensemble prediction is calculated by averaging the predictions of the above trees producing the final prediction.

Parameters used in our Model :

Number of trees in the forest =100
Maximum depth of the tree =20
Minimum number of samples required to split an internal node =2
Seed for the random number generator =0

STATISTICAL TESTS

R-squared Score - It is a statistical measure of how close the data are to the fitted regression line, it is the percentage of the response variable variation that is explained by a linear model.

R-squared = Explained variation / Total variation

R-squared result of the model is 0.7107

RMSE Score - The RMSE is the square root of the variance of the residuals, it can be interpreted as the standard deviation of the unexplained variance.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

RMSE score of the model is 3839.76

MAE Score -Mean Absolute Error (MAE) is a measure of the difference between two continuous variables.

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

MAE score of the model is 1470.28

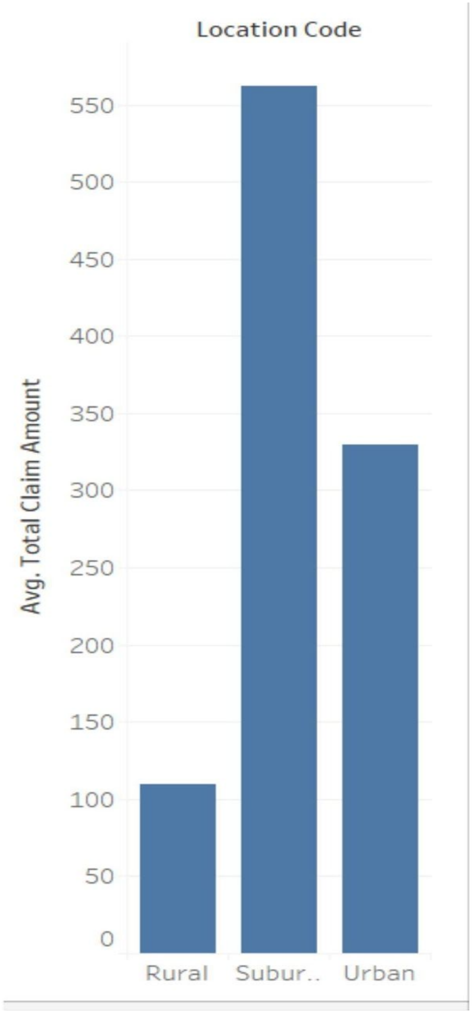
MSE Score - Mean Squared Error basically measures the average squared error of our predictions.

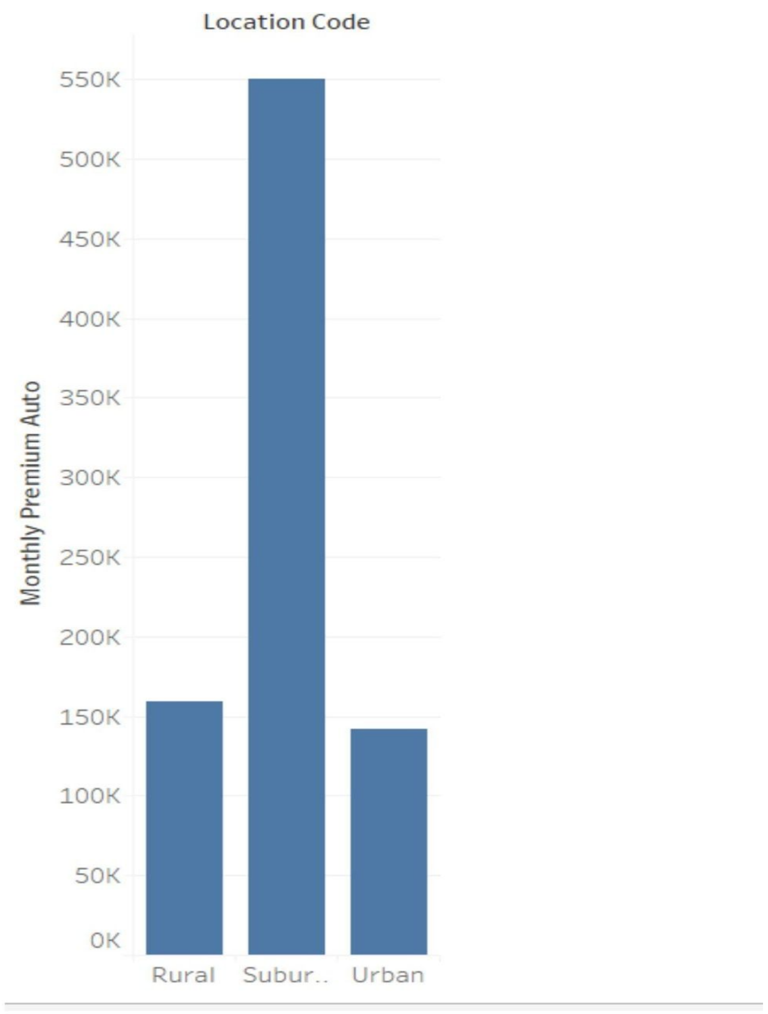
$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

MSE score of the model is 14743785.17

INSIGHTS FOR THE COMPANY

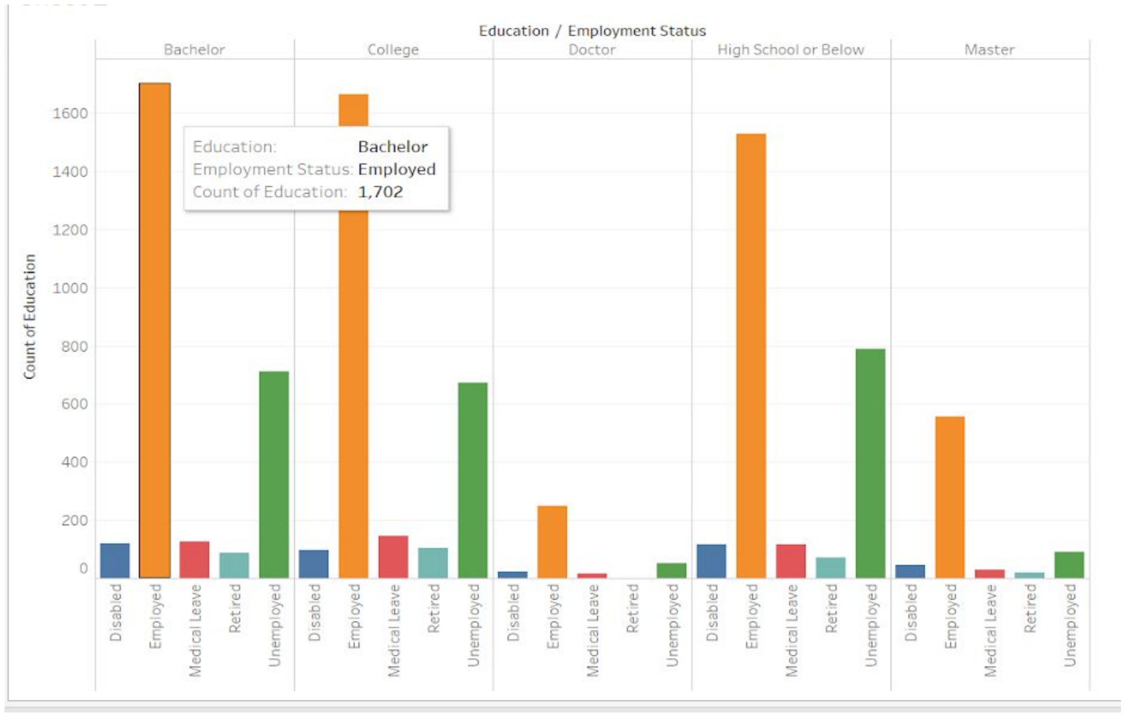
Effective marketing in the semi-urban and rural sector can be more profitable to the company because they pay a higher amount of monthly premium and the public in rural sector claim lesser money than the people in the urban sector. Suburban region has a maximum population of insurance takers and that relates to the high CSV.
The graphs here quantify the same:

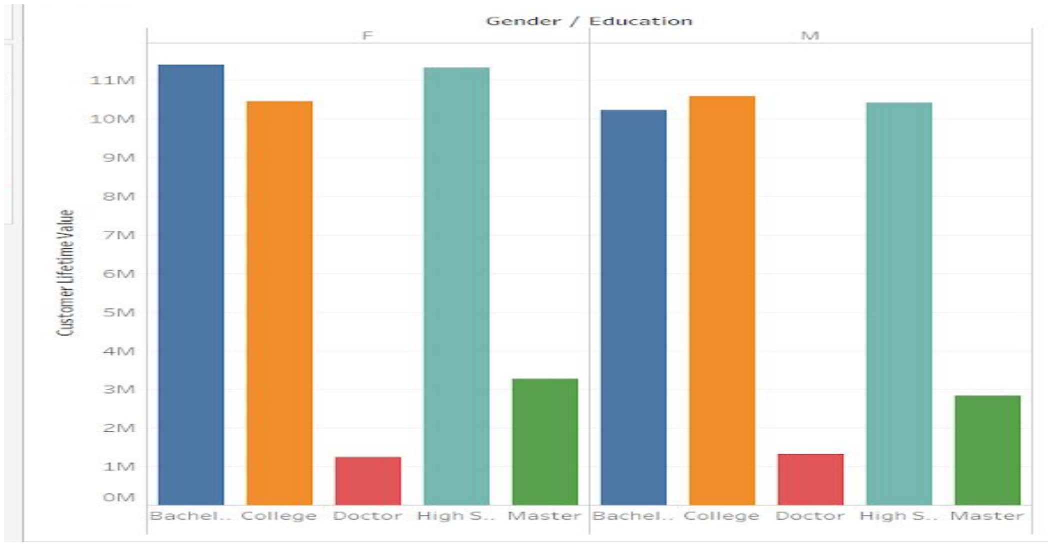




*The Monthly Premium Auto is the aggregate sum of all premiums falling in that category.

An educated person is generally financially stable because of his/her employment. But the bigger difference lies in the gender of the educated person. Generally, *male educated bachelors* and *high school or below* have a higher tendency to take insurances. In the case of females population, the focus must be on *educated bachelors* and *college-going ones*.

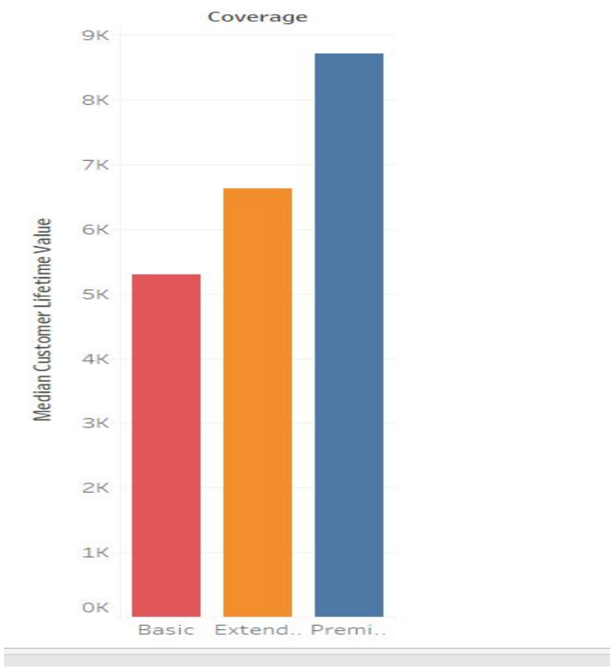


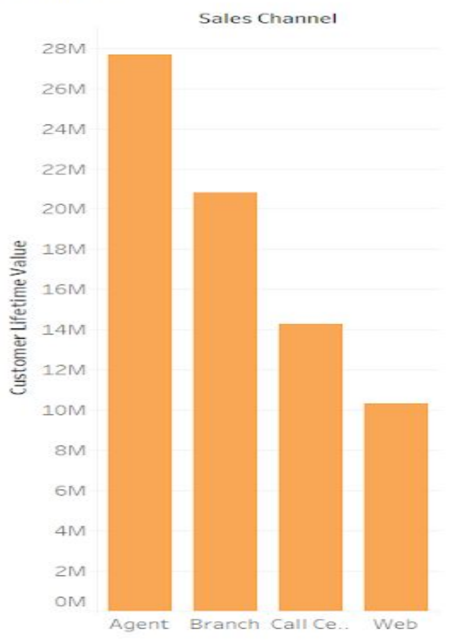


It's definitely not worth wasting resources on doctors because most of them receive such insurance benefits under the government. **Employed** people with **Masters** are the ones who should be focussed as they correspond to very little of the total profitability.

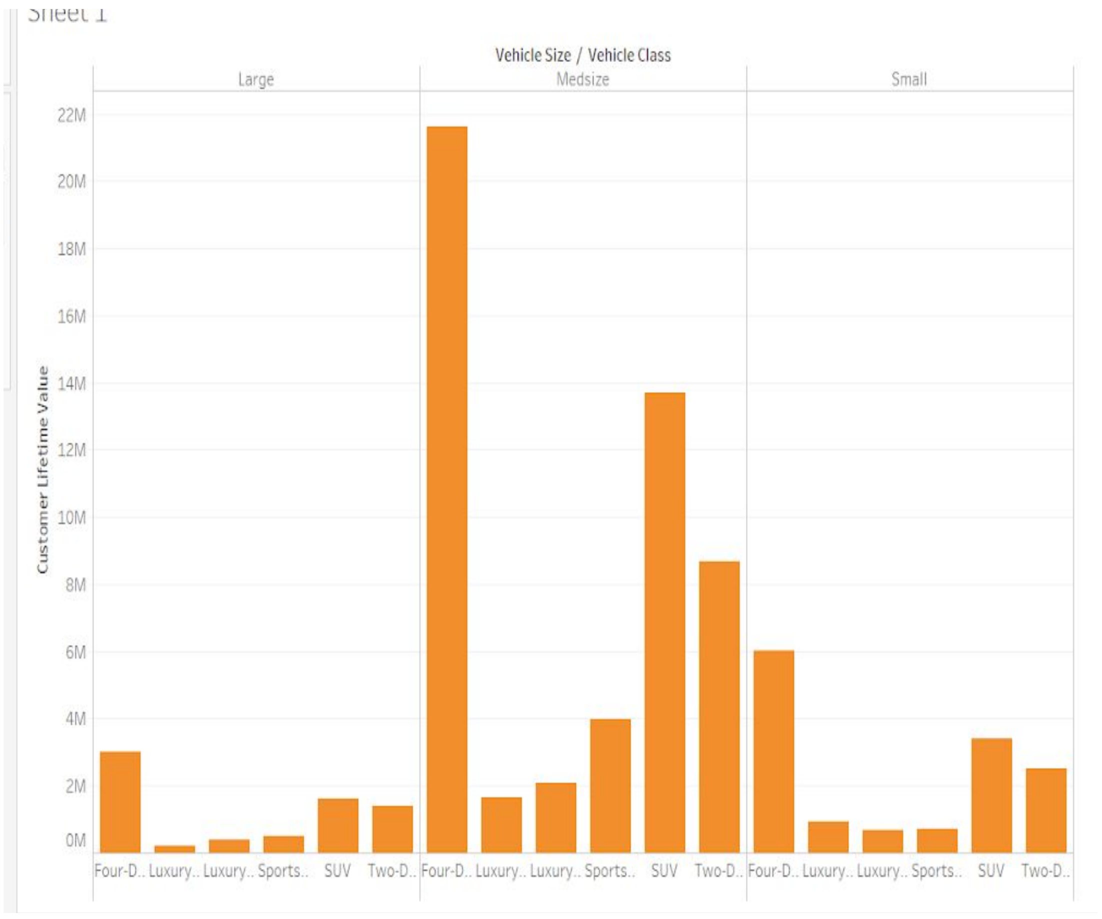
The customers with premium coverage are the best to focus on. Proving some discounts for renewal of Premium insurance holders can be strategically very fruitful. As the insurance type changes from basic to extended, growth of 10% can be observed and for premium, the growth rate is 20%.

With the enormous reach that the internet has today, straying online is one of the best options for an insurance company. Till that time engaging more and **cheaper agents**, developing **branches** are going to be of great help to the company.





Generally, a greater segment of customers have of four-door cars, SUVs and so these vehicles because of the higher number of insurances being sold are more profitable. The segment of luxury vehicles isn't profiting the company because the claim is generally higher. Hence to make this sector profiting a different policy type needs to be issued with a higher premium. **Medium-size and Large-size vehicles are generally the ones that are more profitable. On the contrary large vehicles are prone to more frequent and costly claims.**



ANNEXURE

IMPORTS/LIBRARIES USED

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor
from xgboost import XGBRegressor
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from hyperopt import fmin,tpe,hp, STATUS_OK
```

HYPEROPT

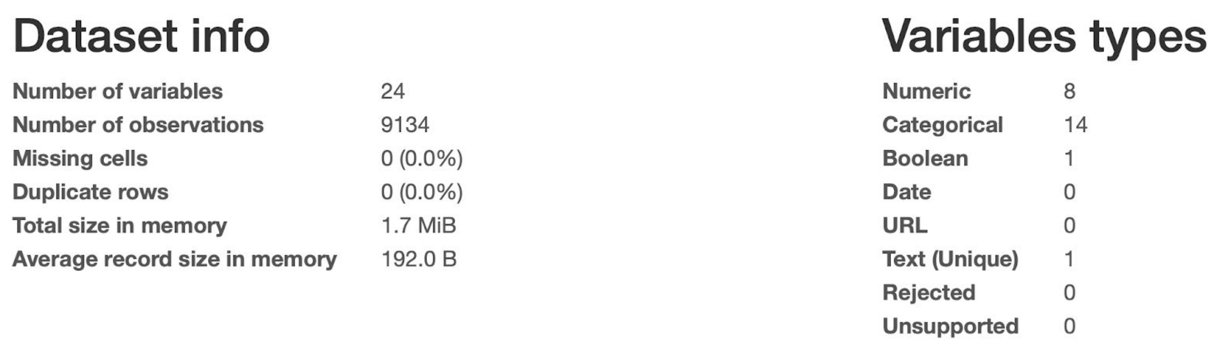
Hyperopt is a library used for Bayesian Hyperparameter Optimization, to select the hyperparameters that give the best results. We used hyperopt for the best performing models, RandomForests and the fully connected Neural Network. RandomForests performed better as the Neural Network tended to overfit the data.

(Code for hyperopt for RandomForests attached)

```
best_r2 = -10
def f(space):
    global best_nest, best_m_d, best_r2, rmse
    nest = space['nest']
    #lr = space['lr']/100
    m_d = space['m_d']

    regressor = RandomForestRegressor(n_estimators = int(nest), random_state = 0,
max_depth = m_d, min_samples_split = 2).fit(X_train, Y_train)
    results = train_test(regressor, X_train, X_test, Y_train, Y_test)

    r2 = results['r2']
    if (r2 > best_r2):
        best_r2 = r2
        best_nest = nest
        #best_lr = lr
        best_m_d = m_d
        rmse = results['RMSE']
        print('best_nest = {}'.format(best_nest))
        print('best_r2 = {}'.format(best_r2))
        print('best_m_d = {}'.format(m_d))

    return {'loss': rmse, 'status': STATUS_OK }

space = {
    'nest': hp.quniform('nest', 100, 1000,10),
    'm_d': hp.quniform('i',5,20,5)
    }

best = fmin(fn=f,space=space,algo=tpe.suggest,max_evals=150)
```

Data Information

PANDAS PROFILING REPORT

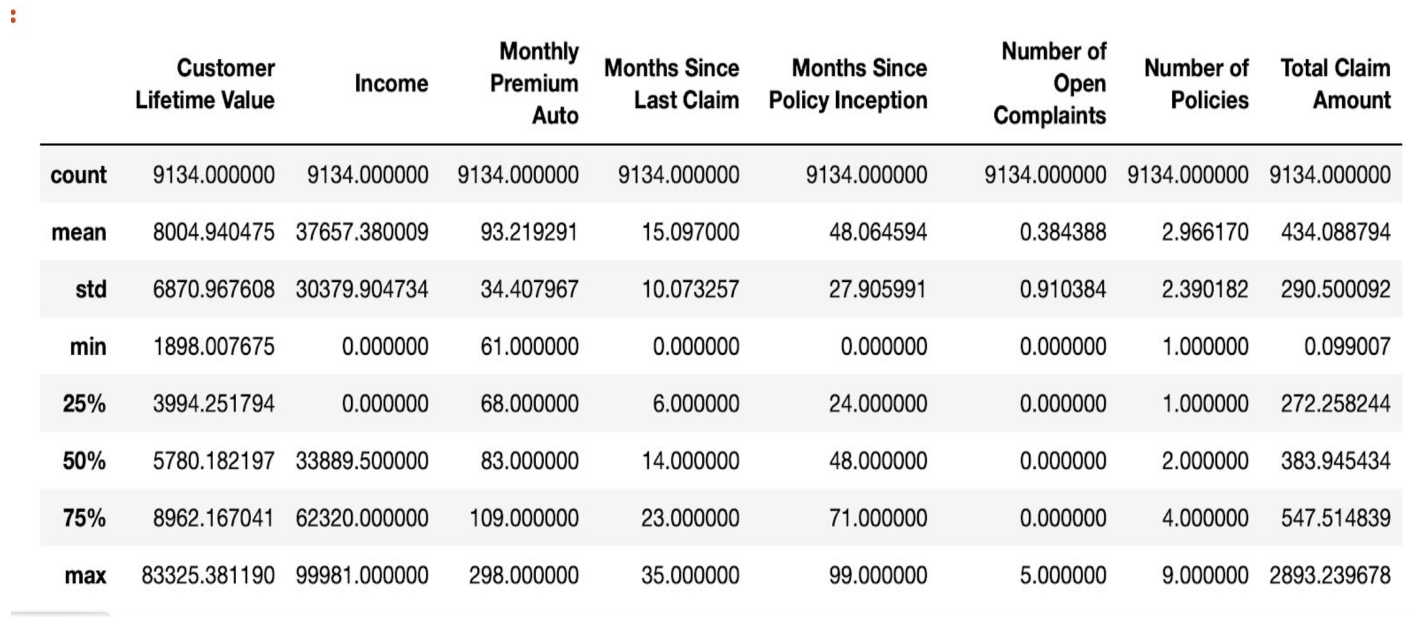


Dataset info

Number of variables	24
Number of observations	9134
Missing cells	0 (0.0%)
Duplicate rows	0 (0.0%)
Total size in memory	1.7 MiB
Average record size in memory	192.0 B

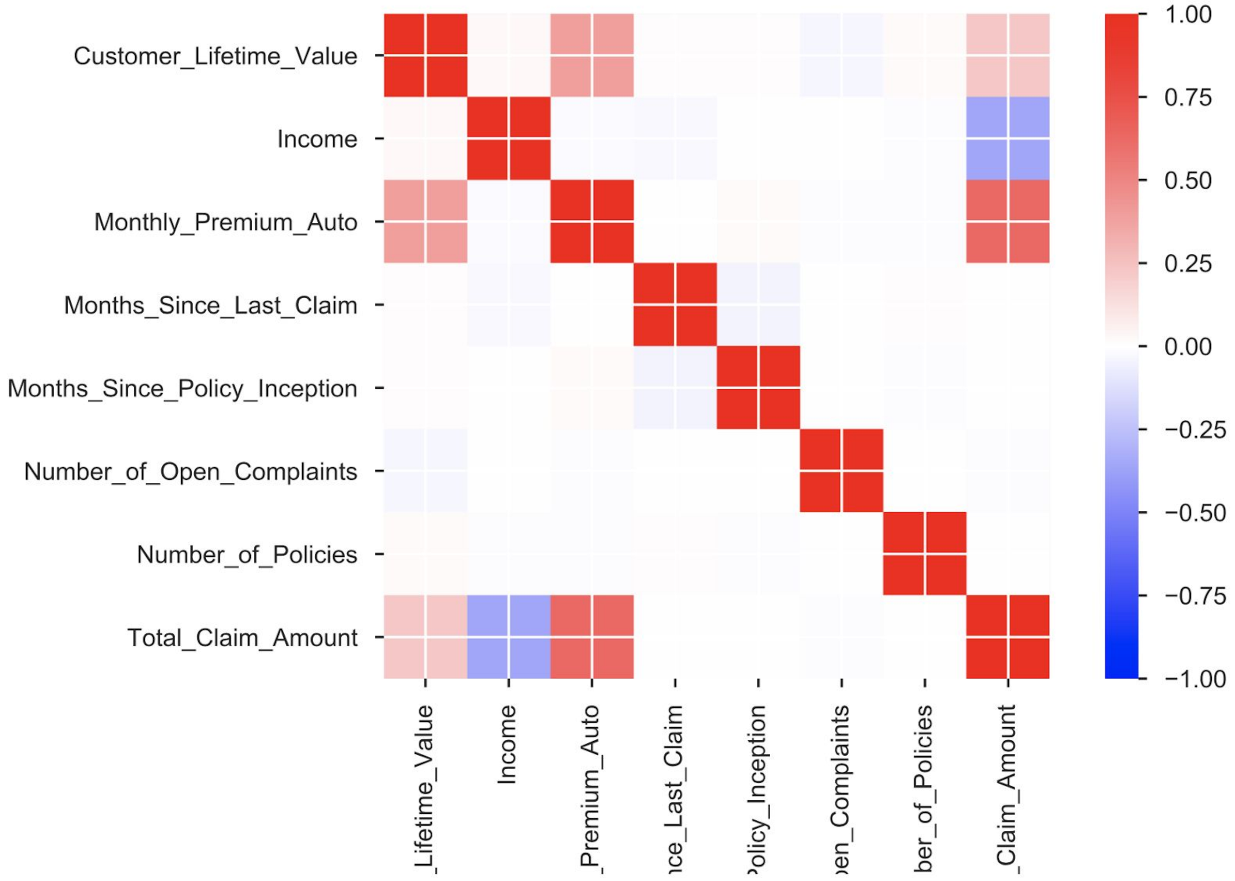
Variables types

Numeric	8
Categorical	14
Boolean	1
Date	0
URL	0
Text (Unique)	1
Rejected	0
Unsupported	0



	Customer Lifetime Value	Income	Monthly Premium Auto	Months Since Last Claim	Months Since Policy Inception	Number of Open Complaints	Number of Policies	Total Claim Amount
count	9134.000000	9134.000000	9134.000000	9134.000000	9134.000000	9134.000000	9134.000000	9134.000000
mean	8004.940475	37657.380009	93.219291	15.097000	48.064594	0.384388	2.966170	434.088794
std	6870.967608	30379.904734	34.407967	10.073257	27.905991	0.910384	2.390182	290.500092
min	1898.007675	0.000000	61.000000	0.000000	0.000000	0.000000	1.000000	0.099007
25%	3994.251794	0.000000	68.000000	6.000000	24.000000	0.000000	1.000000	272.258244
50%	5780.182197	33889.500000	83.000000	14.000000	48.000000	0.000000	2.000000	383.945434
75%	8962.167041	62320.000000	109.000000	23.000000	71.000000	0.000000	4.000000	547.514839
max	83325.381190	99981.000000	298.000000	35.000000	99.000000	5.000000	9.000000	2893.239678

CORRELATION HEATMAP



NORMALITY TEST/STANDARDIZATION OF DATA

Normality tests are used to determine if a data set is well-modeled by a normal distribution and to compute how likely it is for a random variable underlying the data set to be normally distributed. Since we have standardized the data, the mean of each column is zero and standard deviation is 1.

PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA is done to find the most relevant features for prediction. The features Number of Policies, Monthly Premium Auto, Vehicle Class, Total Claim Amount and Income were the most important.

APPLYING NEURAL NETWORK OF THE GIVEN DATA

```
from sklearn.ensemble import IsolationForest
clf = IsolationForest(max_samples = 100, random_state = 42)
clf.fit(train)
y_noano = clf.predict(train)
y_noano = pd.DataFrame(y_noano, columns = ['Top'])
y_noano[y_noano['Top'] == 1].index.values
train = train.iloc[y_noano[y_noano['Top'] == 1].index.values]
train.reset_index(drop = True, inplace = True)
print("Number of Outliers:", y_noano[y_noano['Top'] == -1].shape[0])
print("Number of rows without outliers:", train.shape[0])
# Columns for tensorflow
feature_cols = [tf.contrib.layers.real_valued_column(k) for k in FEATURES]
# Training set and Prediction set with the features to predict
training_set = train[COLUMNS]
prediction_set = train['Customer Lifetime Value']
# Train and Test
```

```
x_train, x_test, y_train, y_test = train_test_split(training_set[FEATURES] , prediction_set,
test_size=0.33, random_state=42)

y_train = pd.DataFrame(y_train, columns = [LABEL])

training_set = pd.DataFrame(x_train, columns = FEATURES).merge(y_train, left_index = True,
right_index = True)

training_set.head()

# Training for submission

training_sub = training_set[col_train]
```