ML Classification Assignement

This report deals the creation and the evaluation of the best model in order to predict if people could be interested in a vehicle insurance provided by the insurance company. The dataset contains the information of 102351 client with the respective target variable.

Data import

First of all, open source libraries, pandas, numpy, seaborn, and matplotlib.pyplot, have been imported as pd, np, sns, and plt respectively. Then, the dataset "model.csv" has been uploaded. Thanks to the `shape` function the dataset's dimensions, equal to `(102351, 13)`, are shown.

Exploratory Data Analysis

The second step is the data exploration, in which duplicated rows and not a numbers have been searched into the dataset. In particular, `5091` not a numbers have been found in *Licence_Type* column and then they have been replaced with string "`Z`" in the original dataset. Finally, the basic statistical details of variables have been shown using `describe` function.

Split categorical and numerical variables

During the creation of a model, one of the most important thing to do is the feature selection in order to define which subset of variables could be useful to make the prediction. Thanks to the "`dtypes`" function is possible to recognize the type (categorical or numerical) of each variable. Referring to the given table and comparing the type of each variable, some of these have been changed from numerical to categorical ones. Firstly, *Region_Code* has been converted into categorical variable because even if it is a unique code for the region of the customer, it could be considered as a string that identify a specific region. Secondly, *Policy_Sales_Channel* has been changed to categorical variable for the same reason of *Region_Code*. Finally, as suggested by the table, *Driving_License* has been converted into categorical variable.

Furthermore, all the categorical variables have been added to `df_categorical` while the numerical ones have been put in to the `df_numerical`

Categorical Data

In order to analyze categorial data, the histogram has been used to visualize the distribution of the different classes on the zeros and on the ones. Equal distributions mean that the variables aren't helpful to distinguish between the two groups. While, dissimilar distributions mean that the variables behave different on the zeros and the ones so helping to recognize the right answer. Following these observation, only some variables have been considered which have been added to the dummies. In this case the chosen varibles are: *Gender*, *Region_Code*, *Previously_Insured*, *Vehicle_Age*, *Vehicle_Damage*, *Policy_Sales_Channel*.

Numerical Data

By the same token, numerical data have been explored using the histogram. It is known that *Annual_Premium* is characterized by an exponential decay. Therefore, a possible way to deal with

this behavior is to apply the logarithm to this variable. Thanks to the pair plot it could be seen the relationships between numerical variables. On the diagonal, it is shown the distribution of the variables into two different subsets, the zeros and the ones. The variables with a different distribution have been chosen to continue the analysis. Furthermore, it could never be used two variables that say exactly the same thing. For these reasons, *Age* and *LogAnnual_Premium* have been considered.

Standardize

All the variables have been put in a standard form because some measures could be in a different scale or in a different unit of measurement, in order to make comparable the different values. As a result, the variable `X_` has been created thanks to the concatenation between the two variables (`dummies` and `X_numerical`) which contain the categorical and numerical chosen ones.

Models

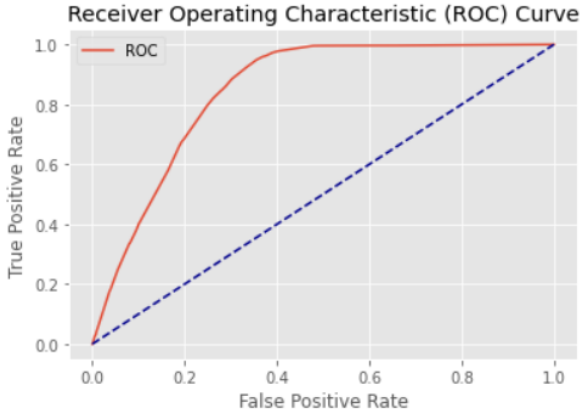
Before choosing the best model, the two sets (train and test sets) have been split. A stratify selection has been used with a `test_size` equal to 0.20. Various approaches have been proposed to solve this problem, but it can be seen from the data that the best model is the Decision Tree because f1_test and f1_train values are similar which means that there isn't overfitting. Furthermore, in this case f1_test is the highest value among all models. In brief, shown below the results of each model.

- Decision Tree

```
f1_train: 0.684863 using {'criterion': 'gini', 'max_depth': 10, 'min_samples_leaf': 10, 'min_samples_split': 20}
f1_test:  0.6852866497394094
[[10633  3343]
 [ 1367  5128]]
              precision    recall  f1-score   support

           0       0.89      0.76      0.82     13976
           1       0.61      0.79      0.69      6495

    accuracy                           0.77     20471
   macro avg       0.75      0.78      0.75     20471
weighted avg       0.80      0.77      0.78     20471
```



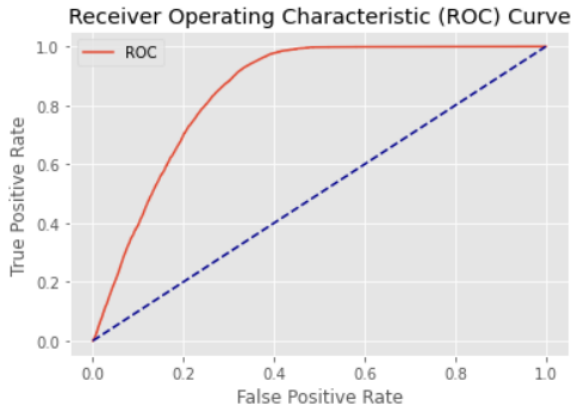

```
AUC: 0.85
```

- Logistic Regression

```
f1_train: 0.669307 using {'C': 0.1, 'max_iter': 1000}
f1_test:  0.6774572871273432
[[10911  3065]
 [ 1598  4897]]
              precision    recall  f1-score   support

           0       0.87      0.78      0.82     13976
           1       0.62      0.75      0.68      6495

    accuracy                           0.77     20471
   macro avg       0.74      0.77      0.75     20471
weighted avg       0.79      0.77      0.78     20471
```



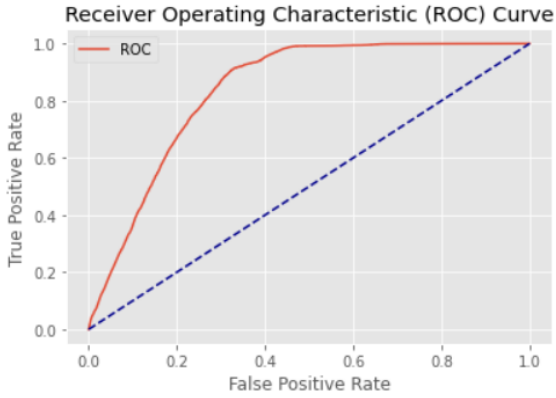

```
AUC: 0.85
```

- Random Forest

```
f1_train: 0.667833 using {'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 6, 'min_samples_split': 5, 'n_estimator
s': 30}
f1_test:  0.6644170017231475
[[11170  2806]
 [ 1868  4627]]
              precision    recall  f1-score   support

           0       0.86      0.80      0.83     13976
           1       0.62      0.71      0.66      6495

    accuracy                           0.77     20471
   macro avg       0.74      0.76      0.75     20471
weighted avg       0.78      0.77      0.78     20471
```



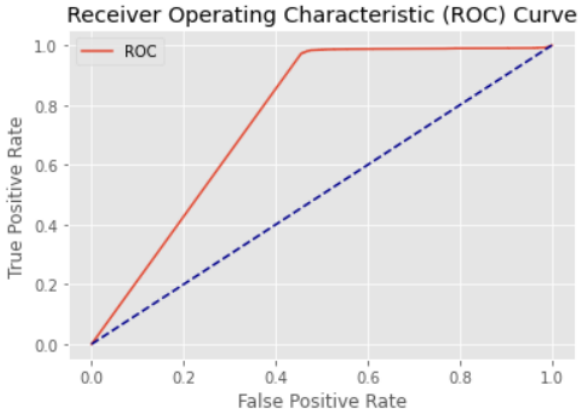

```
AUC: 0.84
```

- Naive Bayes

```
f1_score:  0.6580945003872967
f1_test:  0.6580945003872967
[[7478 6498]
 [ 123 6372]]
              precision    recall  f1-score   support

           0       0.98      0.54      0.69     13976
           1       0.50      0.98      0.66      6495

    accuracy                           0.68     20471
   macro avg       0.74      0.76      0.68     20471
weighted avg       0.83      0.68      0.68     20471
```



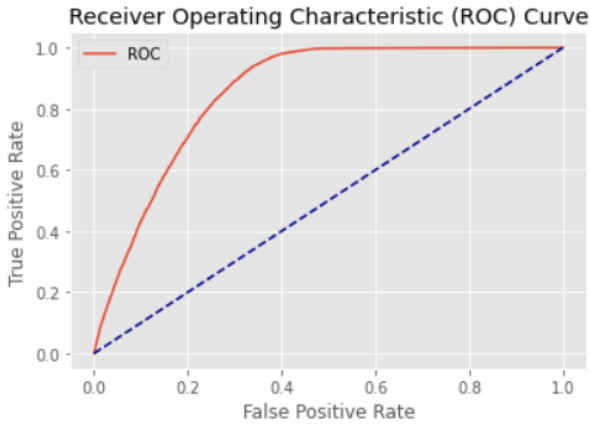

```
AUC: 0.76
```

- Multi-Layer Perceptron Classifier

```
f1_train: 0.671137 using {'alpha': 0.1, 'hidden_layer_sizes': (10, 5), 'max_iter': 2000}
f1_test:  0.6839832773627579
[[10870  3106]
 [ 1505  4990]]
              precision    recall  f1-score   support

           0       0.88      0.78      0.83     13976
           1       0.62      0.77      0.68      6495

    accuracy                           0.77     20471
   macro avg       0.75      0.77      0.75     20471
weighted avg       0.80      0.77      0.78     20471
```




```
AUC: 0.86
```

Imbalanced Data

As can be seen from the histogram the dataset is unbalanced. A function, that execute both downsampling and oversampling, has been used to balance the data, but in this way the problem of overfitting has been occurred. As follows examples of overfitting obtained using some models are reported.

- Decision Tree

```
f1_train: 0.828314 using {'criterion': 'entropy', 'max_depth': 4, 'min_samples_leaf': 10, 'min_samples_split': 20}
f1_test:  0.6893385728451284
[[12862  8102]
 [  357  9385]]
              precision    recall  f1-score   support

           0       0.97      0.61      0.75     20964
           1       0.54      0.96      0.69      9742

    accuracy                           0.72     30706
   macro avg       0.75      0.79      0.72     30706
weighted avg       0.83      0.72      0.73     30706
```

- Logistic Regression

```
f1_train: 0.828603 using {'C': 0.1, 'max_iter': 1000}
f1_test:  0.6995941496286086
[[13724  7240]
 [  606  9136]]
              precision    recall  f1-score   support

           0       0.96      0.65      0.78     20964
           1       0.56      0.94      0.70      9742

    accuracy                           0.74     30706
   macro avg       0.76      0.80      0.74     30706
weighted avg       0.83      0.74      0.75     30706
```