IME 672: DATA MINING GROUP 16 PROJECT REPORT

PROBLEM STATEMENT:-

Porto Seguro is one of the Brazil's largest auto and homeowner insurance companies. Porto Seguro Auto insurance protects the vehicle and offered several advantages to the insured persons in contracting and renovation procedures, such as free check-ups in Porto Seguro Auto centres, services for residences, discounts in restaurants and parking lots.Inaccuracies in car insurance company's claim predictions raise the cost of insurance for good drivers and reduce the price for bad ones. Therefore a competition is sponsored by Porto Seguro in which we need to build a model that **predicts the probability that a driver will initiate an auto insurance claim in the next year**

DATA UNDERSTANDING:-

In the train and test data,

- features that belong to similar groupings are tagged as such in the feature names (e.g., ind, reg, car, calc).
- In addition, feature names include the postfix bin to indicate binary features and cat to indicate categorical features.
- Features without these designations are either continuous or ordinal.
- Values of -1 indicate that the feature was missing from the observation.
- The target columns signifies whether or not a claim was filed for that policy holder. Each row corresponds to a policy holder

In total, our *training* data has 59 variables, including *id* and *target*.
Observations: 595,212
We find that majority of cases has no filed claim. With less than 4% of policy holders filing a claim the problem is heavily imbalanced.
There are no duplicate rows in our dataset

Ind - individual or driver
Reg- region,
Car- car
Calc- calculated feature.'

We find a very similar structure for the test data set.

DATA PREPROCESSING

Before applying algorithm on our data to classify it, we will first convert pre-process our data. The data we have is incomplete, inconsistent, and lacks in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues. It prepares raw data for further processing.

Numeric attribute

We have calculated mean, median and quantiles for each numeric attribute to better understand our data. It is helpful in making comparisons between attributes. It also helps in making further statistical analysis.

Categorical attribute

We have not used mean, median and quantiles as a way to understand data for categorical variables. As our variable is of integer data type and our mean calculated will be float therefore it is difficult to generalise our data.The important disadvantage of mean is that it is sensitive to extreme values/outliers.
We have converted categorical variables into factors and have used mode for data understanding

Binary attributes

Similarly, we have converted binary variables into logical values of true and false.

Target attribute

For the target variable we have used :1 if claim has been filled and 0 if not. We have chosen factor format for this variable.

Missing Values

- There are 84648 missing values in our training set and 1270295 in testing data.
- In reg variables, only ps_reg_03 has missing value.
- There is no missing value in calc variables.
- The features *ps_car_03_cat* and *ps_car_05_cat* have the largest number of NAs. They also share numerous instances where NAs occur in both of them for the same row
- Our data contains about 2.5% of missing value in total

Missing values problems

We have seen our data contains many missing values. It can be caused by various factors:

- High cost involved in measuring variables
- Failure of sensors
- Reluctance of respondents in answering certain questions or
- An ill-designed questionnaire

Missing values in data is a serious data quality problem. It reduces the performance of data mining algorithms

Handling Missing Values

---Method 1

Ignoring the tuples which have one or more missing values- After doing this our data becomes highly reduced. Many tuples are removed due to this method. Number of tuples reduced from 595212 to 124931. Thus, information is lost and it will affect performance of our algorithm. Hence, we will discard this method.

---Method 2

Imputing Missing Values with mean/median for numerical attributes- We can replace the missing values of a numeric attribute by the mean/median of that attribute. The **mean** takes account of all values to **calculate** the **average**. Very small or very large values can affect the **mean**. The **median** is not affected by very large or very small values. If there is an even number of numbers, the **median** is found by averaging the two middle numbers. This is the main disadvantage of mean/median. Hence, we have not used this method to replace missing value.

---Method 3

Multivariate Imputation by chained equations (MICE)- It has emerged as a principled method of dealing with missing data. The chained equations approach is very flexible and can handle variables of varying types (e.g., continuous or binary) as well as complexities such as bounds or survey skip patterns. Multiple imputation involves filling in the missing values multiple times, creating multiple "complete" datasets. Because multiple imputation involves creating multiple predictions for each missing value, the analyses of multiply imputed data take into account the uncertainty in the imputations and yield accurate standard errors.
We have used this method for imputing missing values.

Id is a redundant attribute so we remove it from our dataset

Data visualisation

We check the correlations between **interval variables**. A heatmap is a good way to visualize the correlation between variables.

INDIVIDUAL FEATURE VISUALISATION

- We find that some of the binary features are very unbalanced; with "FALSE" accounting for the vast majority of cases. This is particularly true for the *ps_ind* sequence from "10" to "13".

- For the three features *ps_ind_16_bin*, *ps_calc_16_bin*, and *ps_calc_17_bin* we find that the . "TRUE" values are in fact dominating.

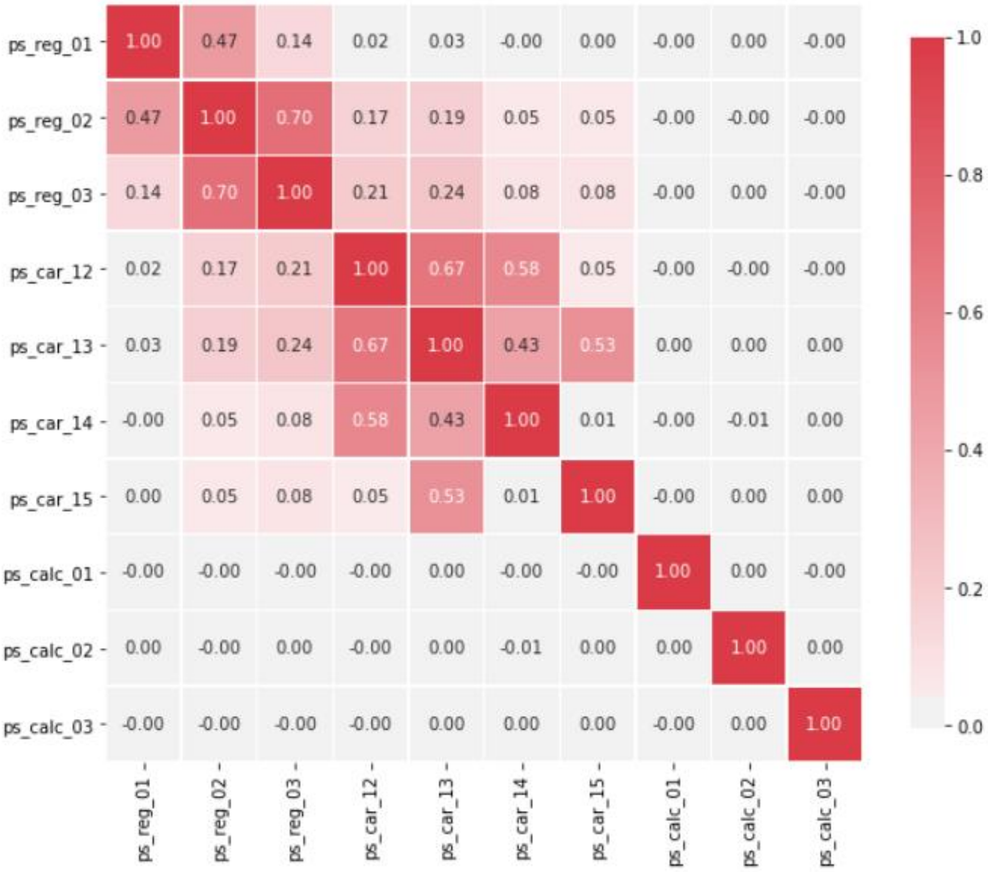
- We find that some categorical features have only very few levels, down to 2 levels (+ NA) for three of them. In others we have up to 11 levels, some of which are clearly dominating the (logarithmic) plots.

- The integer features for the "ind" and "car" groups are best visualised in a categorical-style barplot, because their ranges are not very large. We are using log axes for some.

- We find that again there are large differences in frequencies, in particular for *ps_ind_14* and *ps_car_11* where "0" and "3" are the dominating values, respectively.

- Whereas most of the “calc” integer features can still be visualised best using barplots, for three of them a histogram is a better choice.

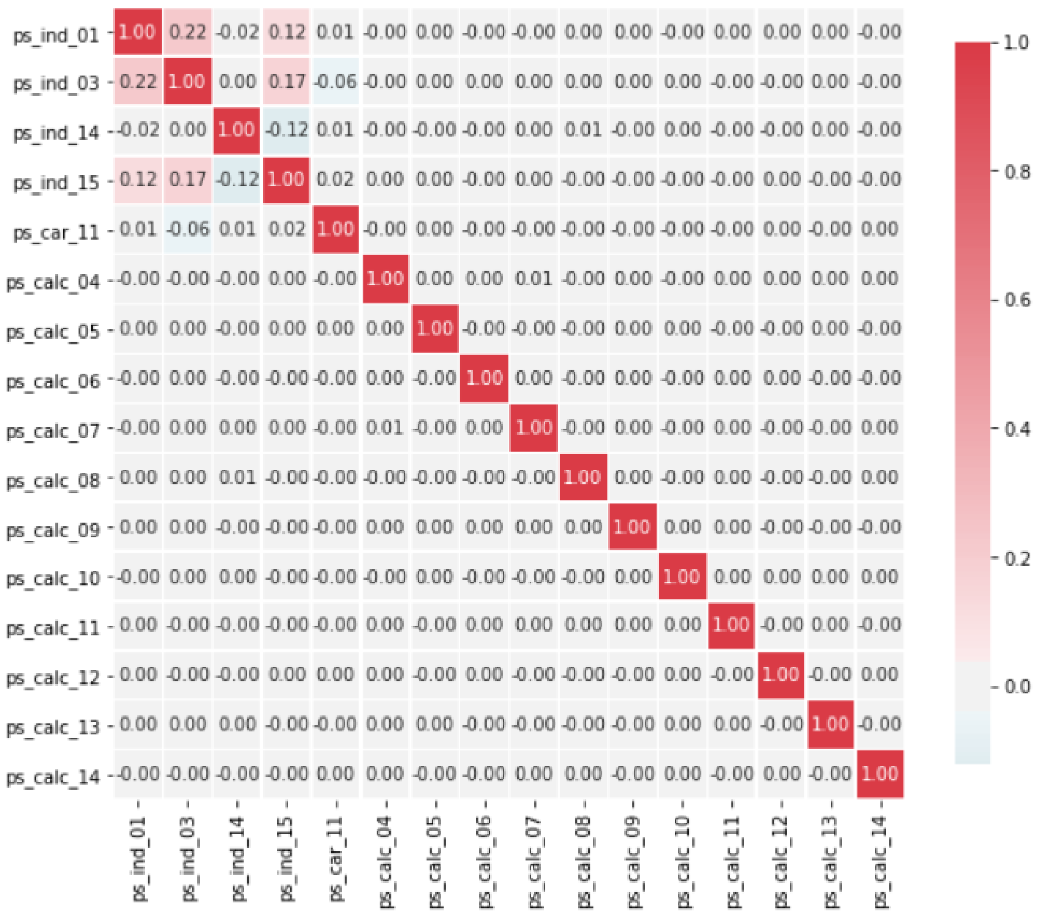
- For the floating point features we choose histograms to get a first impression of their distributions.



There is a strong correlation between the variables:

- ps_reg_02 and ps_reg_03 (0.7)
- ps_car_12 and ps_car13 (0.67)
- ps_car_12 and ps_car14 (0.58)
- ps_car_13 and ps_car15 (0.67)

Next we check the correlation between **ordinal variables:**



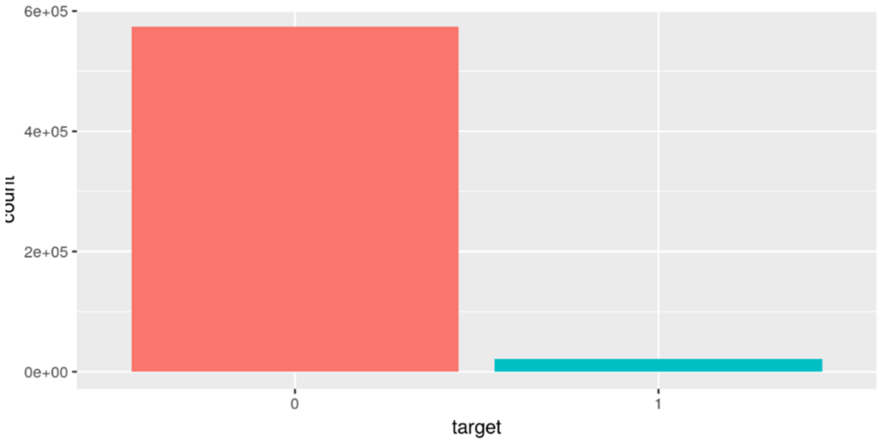
	ps_ind_01	ps_ind_03	ps_ind_14	ps_ind_15	ps_car_11	ps_calc_04	ps_calc_05	ps_calc_06	ps_calc_07	ps_calc_08	ps_calc_09	ps_calc_10	ps_calc_11	ps_calc_12	ps_calc_13	ps_calc_14
ps_ind_01	1.00	0.22	-0.02	0.12	0.01	-0.00	0.00	-0.00	-0.00	0.00	0.00	-0.00	0.00	0.00	0.00	-0.00
ps_ind_03	0.22	1.00	0.00	0.17	-0.06	-0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.00	-0.00	0.00	-0.00
ps_ind_14	-0.02	0.00	1.00	-0.12	0.01	-0.00	-0.00	-0.00	0.00	0.01	-0.00	0.00	-0.00	-0.00	0.00	-0.00
ps_ind_15	0.12	0.17	-0.12	1.00	0.02	0.00	0.00	-0.00	0.00	-0.00	-0.00	-0.00	-0.00	0.00	-0.00	-0.00
ps_car_11	0.01	-0.06	0.01	0.02	1.00	-0.00	0.00	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	0.00	-0.00	0.00
ps_calc_04	-0.00	-0.00	-0.00	0.00	-0.00	1.00	0.00	0.00	0.01	-0.00	0.00	-0.00	0.00	0.00	0.00	0.00
ps_calc_05	0.00	0.00	-0.00	0.00	0.00	0.00	1.00	-0.00	-0.00	-0.00	0.00	0.00	-0.00	-0.00	0.00	-0.00
ps_calc_06	-0.00	0.00	-0.00	-0.00	-0.00	0.00	-0.00	1.00	0.00	-0.00	0.00	-0.00	0.00	0.00	-0.00	-0.00
ps_calc_07	-0.00	0.00	0.00	0.00	-0.00	0.01	-0.00	0.00	1.00	-0.00	0.00	-0.00	0.00	-0.00	-0.00	0.00
ps_calc_08	0.00	0.00	0.01	-0.00	-0.00	-0.00	-0.00	-0.00	-0.00	1.00	0.00	-0.00	0.00	-0.00	-0.00	0.00
ps_calc_09	0.00	0.00	-0.00	-0.00	-0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	-0.00	-0.00	-0.00
ps_calc_10	-0.00	0.00	0.00	-0.00	-0.00	-0.00	0.00	-0.00	-0.00	-0.00	0.00	1.00	0.00	0.00	0.00	0.00
ps_calc_11	0.00	-0.00	-0.00	-0.00	-0.00	0.00	-0.00	0.00	0.00	0.00	0.00	0.00	1.00	-0.00	0.00	-0.00
ps_calc_12	0.00	-0.00	-0.00	0.00	0.00	0.00	-0.00	0.00	-0.00	-0.00	-0.00	0.00	-0.00	1.00	-0.00	0.00
ps_calc_13	0.00	0.00	0.00	-0.00	-0.00	0.00	0.00	-0.00	-0.00	-0.00	-0.00	0.00	0.00	-0.00	1.00	-0.00
ps_calc_14	-0.00	-0.00	-0.00	-0.00	0.00	0.00	-0.00	-0.00	0.00	0.00	-0.00	0.00	-0.00	0.00	-0.00	1.00

For the ordinal variables we do not see many correlations.

Modelling and Fitting the dataset

1. Decision Tree

We implemented the Decision Tree algorithm for our problem but **due to highly skewed classification(target) training dataset**, our model gets biased and yielded very poor accuracy.
The number of "FALSE" classes in target attribute are much more than "TRUE" class. (Shown in below histogram)
This reason can be accounted for as a failure of the Decision Tree model.



Here are some of the **evaluation metrics** obtained for Decision Tree:

1. Prediction Vs Actual table

Prediction Vs Actual	0	1
0	430200 (TN)	16209 (FN)
1	0 (FP)	0 (TP)

Observation: It is clearly observable that our model here is predicting all the values in test split set as zero, so even though this model has 96.37 % it is not good for use in this case.

2. Naive Bayes

The next model we created using Naive Bayes algorithm and to our fortune, it gave us the accuracy of **92.25%** in our first run. We predicted values on training split set and test split set and both gave us similar accuracies which provides a clue that **no problem of overfitting** has occurred.

We read some healthcare problem where they used naive bays to figure out whether a particular medical test will be positive or negative based on some given information. We observed our dataset and their dataset similar to ours.

In such cases, analyst used Naive Bayes Algorithm to train their model.

Here are some of the **evaluation metrics** obtained for Decision Tree:

1. Prediction Vs Actual table

Prediction Vs Actual	0	1
0	136752	4972
1	6566	513

2. Precision - In this case, we get the precision value of 0.07246786

3. Recall - In this case, we get recall value of 0.0935278

4. F1 score - A single number of evaluation metrics obtained from precision and recall: 0.08166189

3. XGBOOST

One of the most popular algorithms at the present time- Boost. We created a model and got an accuracy of **96.42%** which is more than any of the algorithms we have used above. We also ran our model on training split set and similar accuracies were obtained for both training and test split set hence giving us the clue that **no overfitting** has occurred. Also, note from the following table that it's not like decision tree which was predicting all tuples target as zero.

Here are some of the **evaluation metrics** obtained for Decision Tree:

1. Prediction vs Actual table

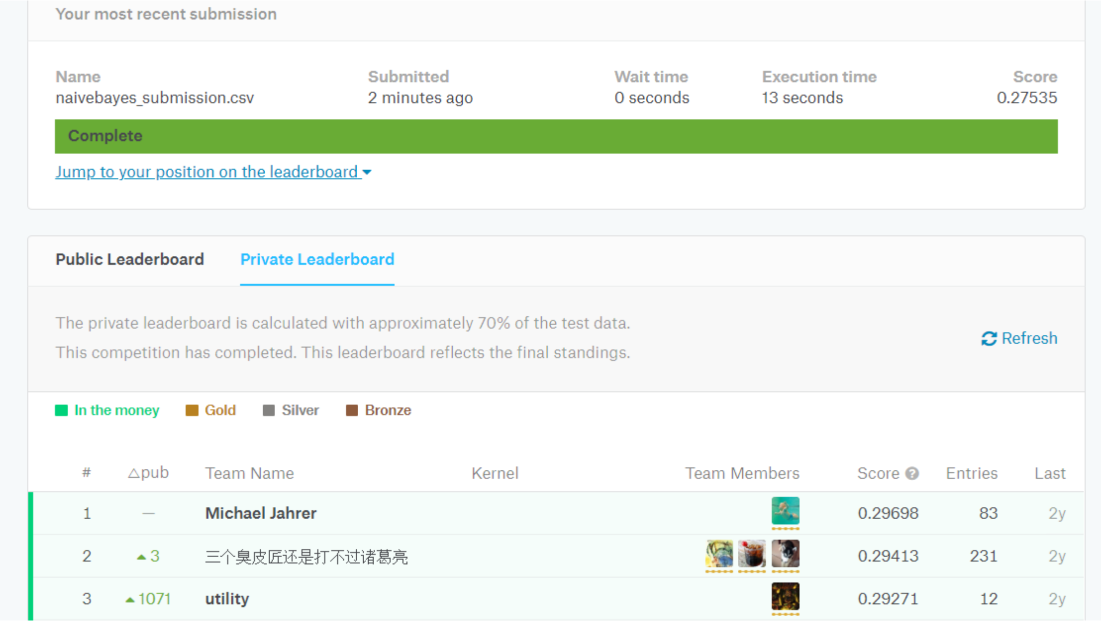
Prediction Vs Actual	0	1
0	143477	5316
1	7	3

2. Precision - Value obtained**: 0.3**
3. Recall - Value obtained: 0.005
4. F1 score - Value obtained: 0.00112

Which model to Choose?

We decide to choose the Naive Bayes model a better version then rest we trained for our problem. Naive Bayes gives us better F1 score and Recall value with good accuracy.

As F1 score =2*(precision*recall)/(precision+recall) so it give better realisation for our answer.



Your most recent submission

Name	Submitted	Wait time	Execution time	Score
naivebayes_submission.csv	2 minutes ago	0 seconds	13 seconds	0.27535

Complete

Jump to your position on the leaderboard ▾

Public Leaderboard   Private Leaderboard

The private leaderboard is calculated with approximately 70% of the test data.
This competition has completed. This leaderboard reflects the final standings.

Refresh

🟩 In the money 🟧 Gold ■ Silver 🟫 Bronze

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	Michael Jahrer			0.29698	83	2y
2	▲3	三个臭皮匠还是打不过诸葛亮			0.29413	231	2y
3	▲1071	utility			0.29271	12	2y

We got score 0.27535 while highest scorer got 0.29698 on Kaggle.

NOTE: -

Such high accuracy is not accounted by overfitting rather it is a result of highly skewed class in Naïve Bayes. It is natural as in our training set target attribute has 2 levels of value and one of which constitutes 97% of target. We showed such skewness in above Histogram plot.