**INDEX**

[1. Exploratory data analysis 2](#_Toc17729235)

[i. Shape of train and test data 2](#_Toc17729236)

[ii. Missing value analysis 2](#_Toc17729237)

[iii. Univariate analysis 2](#_Toc17729238)

[iv. Bi-variate analysis 7](#_Toc17729239)

[v. Outlier analysis 8](#_Toc17729240)

[vi. Imbalanced target class 9](#_Toc17729241)

[SMOTE 9](#_Toc17729242)

[2. MODEL BUILDING 11](#_Toc17729247)

[3. Measuring the accuracy of model 12](#_Toc17729248)

[4. Approach 1(all features included) 13](#_Toc17729249)

[1. Logistic regression- 13](#_Toc17729250)

[Assumptions – 14](#_Toc17729251)

[Results: 14](#_Toc17729252)

[2. Naïve Bayes 14](#_Toc17729253)

[Assumptions 14](#_Toc17729254)

[Results 15](#_Toc17729255)

[3. RandomForest 15](#_Toc17729256)

[Results 16](#_Toc17729257)

[RESULTS – 16](#_Toc17729258)

[4. Alternative approach(Approach 2) 17](#_Toc17729259)

[5. RandomForest feature importance 19](#_Toc17729260)

**REPORT : Predict if customer will make a transaction(irrespective of amount ) in future or not?**

# Exploratory data analysis

## Shape of train and test data

((200000, 202), (12886, 201))

-----Summary of columns

Total col - 201

continous – 201

-------------------------

categorical – 0

## Missing value analysis

No Missing values in train

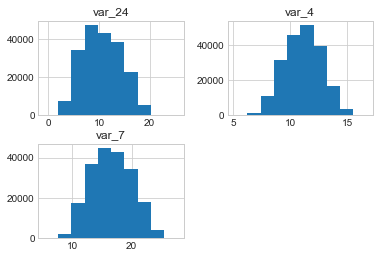
Missing values exist in test

| **missing\_per** |
| --- |
| **var\_10** | 0.0005 |
| **var\_11** | 0.0005 |
| **var\_12** | 0.0005 |
| **var\_13** | 0.0005 |
| **var\_14** | 0.0005 |

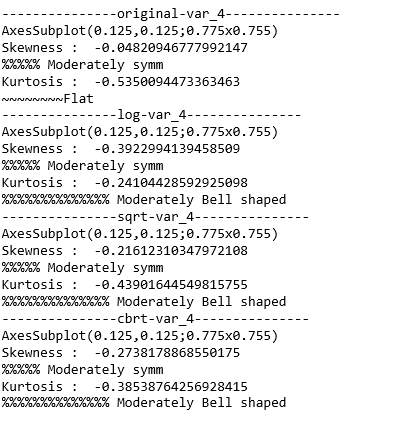
Filled the missing values with mean

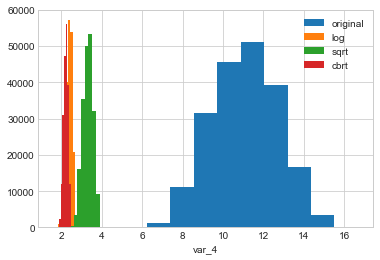
## Univariate analysis

Plot histograms for a few columns

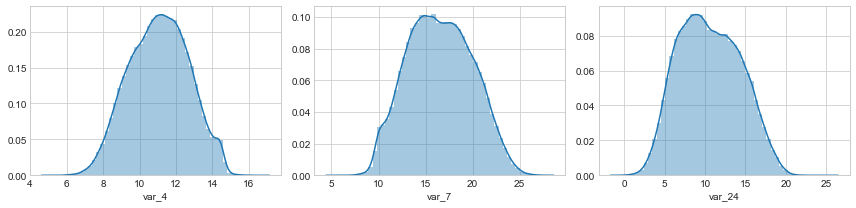


1. Skew and kurtosis for var\_4

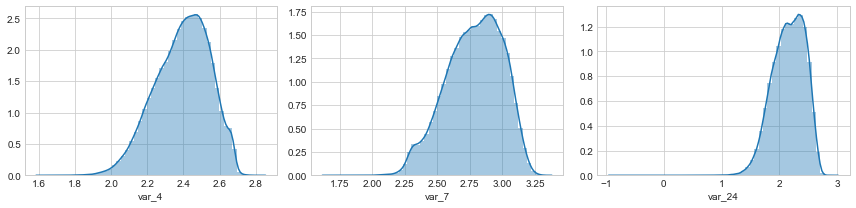




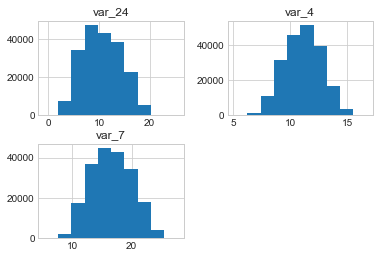
Original was flat and log was moderately bell shaped, so we choose for this column of train data



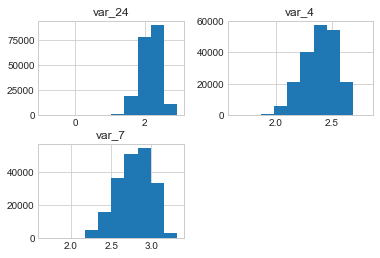
After processing:



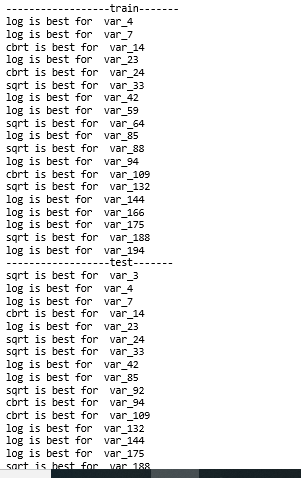
BEFORE



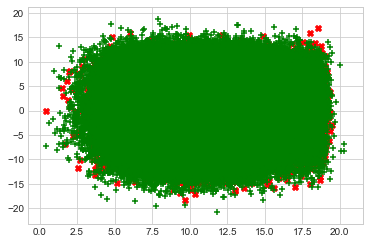
AFTER



SUMMARY for all the columns

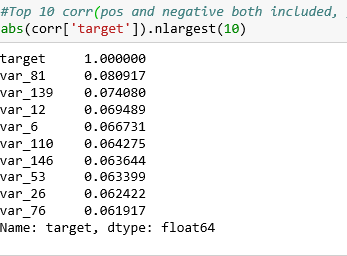


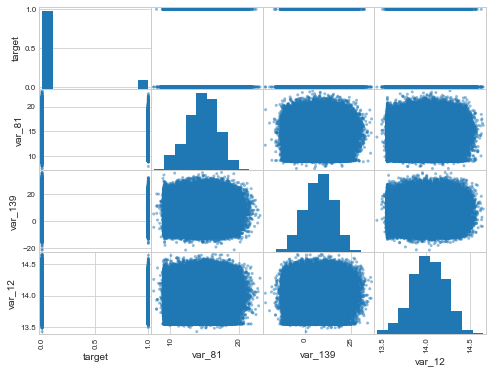
## Bi-variate analysis

1. Scatterplot of var\_0 between 2 target classes( 0 and 1)

Difficult to draw a boundary b/w them, they are overlapping

1. We need to plot the scatter plots, but we will go ahead with most correlated features only

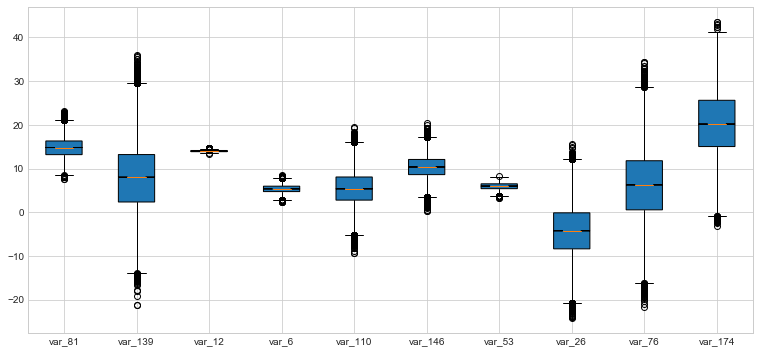




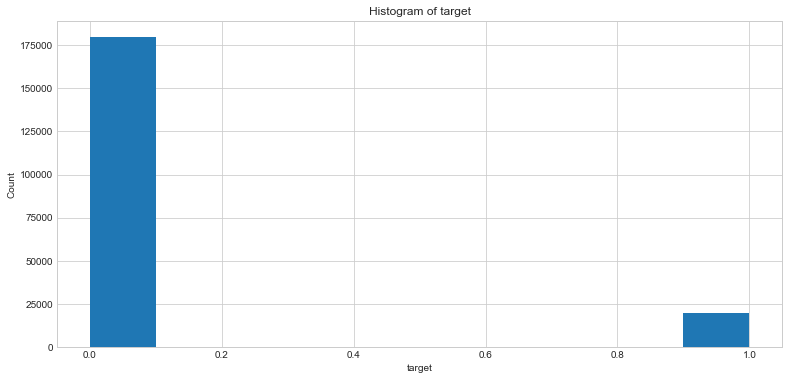
The values are very dispersed and not very strongly correlated with the target variable

## Outlier analysis

Take a subset of most correlated features (with target) and plot, since original data is huge



## Imbalanced target class



* Positive cases(people doing a transaction) are very less as compared to negative class.
* Since outliers are taken care of, need to balance these classes using SMOTE

**# Analysis:**

* Negative case : target = 0 means customer will not make a transaction in future
* Positive case : target = 1 means customer will make a transaction in future

Data is **imbalanced**, need to use techniques:

-- under sampling - randomly select a sub-sample from negative class to make equal to positive class

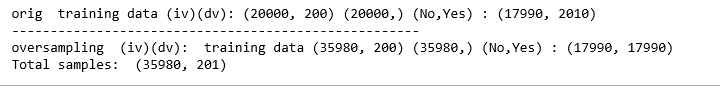
--oversampling - SMOTE to synthesize positive cases

But here, since positive cases are very less and decreasing our sample set might affect our decision, I choose SMOTE

### SMOTE

oversampling to increase the minority class samples i.e. target = 1(Will make a Transaction)

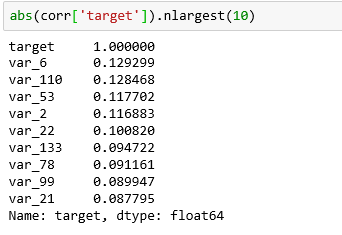
**Prerequisite** - See outliers and fix those, or else outlier values will affect data synthesis and more outlier values will be present in resultant dataset

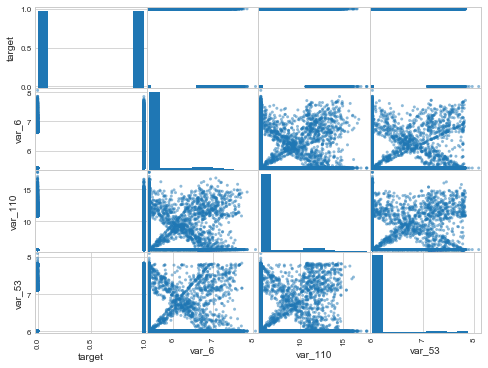


We have taken a stratified sample of 20k observations and then ran SMOTE,

Total number of sample increases as a result SMOTE, as it produces first positive cases equal to negative 17990 and then it creates positive cases for the old 2010 negatives we had.

After SMOTE, we again plot scatter plots with most correlated features wrt target





# ## The correlation of target increased to a max of 0.12 for this sub-sample, but overall again not much correlation

# LogisticRegression depends on the how well we can draw a boundary between target class, so wont do any good here

# Naive Bayes suits the cases where features are not dependent on each other

# RandomForest has no assumptions of normalized data, or any other, so might be the best here

# MODEL BUILDING

Now that we are done with cleaning of data, we start building the model

We need to split our data for taining and testing.

Way1 is using random sampling

Way2 is using Kfolds cross validation( I have used a framework to abstract, so that different classifiers can be run with lesser code, using OOPS)

I have used both ways in python notebook, here I wil be pasting results for cross validation with 5 folds and taking average of errors in each fold.

# Measuring the accuracy of model

classification metrics is not a good indication for imbalanced datasets Rather we use ROC

Recall is also called sensitivity(TP rate - actual positive classes which were correctly classified)

Specificity is also called TN rate(actual negative classes which were correctly classified)

F1 score = combination of recall and specificity

Depending on the kind of problem you want to solve you may want to maximize either sensitivity or specificity.

confusion matrix considers only a single threshold value to generate a curve vs all possibe classification thresholds are considered for ROC curve

ROC compares TPR(y-axis) vs FPR(x-axis) , More the area under the curve, better the model

EXAMPLE :

Paper published in a Journal positives(admitted) = 250

negatives(not admitted) =250

Based on prob threshold(>0.6 generally) which we set, we classify into classes.

If your model did not do well, then distribution plot for both classes will be overalapping for a significant area and the roc curve will be a 45 degree line, telling that model just did random guessing

TPR = TP / all positives FPR = FP/ all negatives

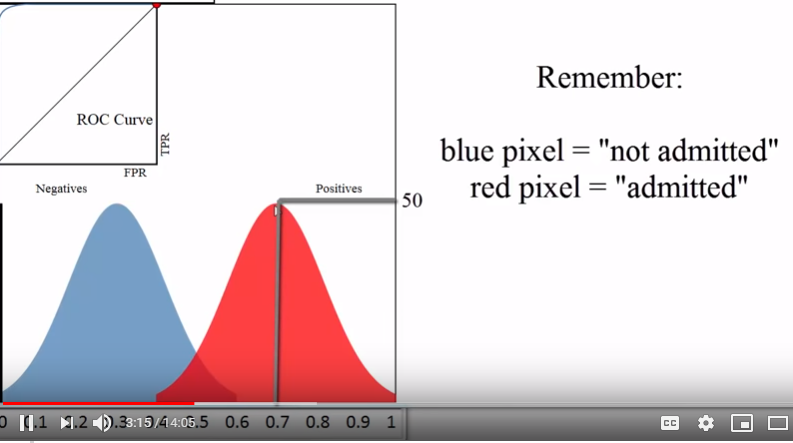
Curve generation E.g. If threshold is say 0.8,then the area to the right of the curve is admitted and to the left is not admitted

TPR = 50(red pixels to the right of threshold line)/250(total red(pos) pixels)

FPR = 0(blue pixels to the right of threshold line)/250(total red(pos) pixels)

plot (FPR,TPR) --> (0,0.2) E.g.

If threshold is say 0.5, then the area to the right of the curve is admitted and to the left is not admitted TPR = 230(red pixels to the right of threshold line)/250(total red(pos) pixels)=0.94 FPR = 125(blue pixels to the right of threshold line)/250(total red(pos) pixels)=0.5 plot (FPR,TPR) --> (0.5,0.94) All the possible classificaton thresholds are considered and points are plotted to fit a curve



So for good model, curve will be hugging the topmost left corner, more area under the curve and auc should be relatively higher

# Approach 1(all features included)

## Logistic regression-

Works with probabilities of one observation. It uses Logistic function to estimate the prediction.

In this logit(p^) is calculated using w0^+w1^x1+w2^x2……….

Where w0^,w1^,w2^ are regression coefficients and are calculated for all unique values(for all categ values) of that variable

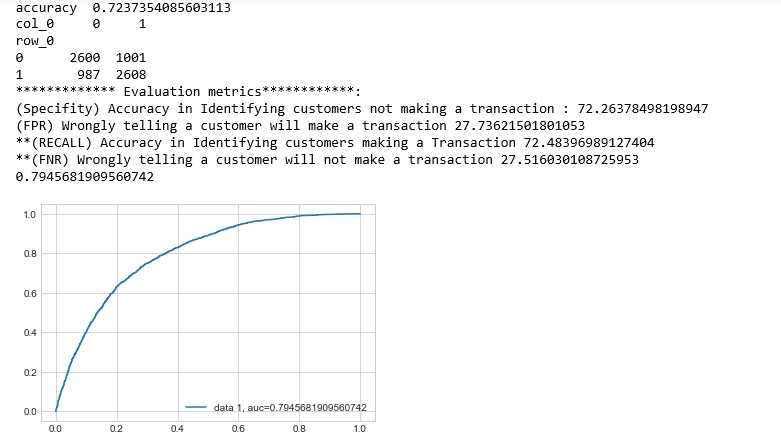
Value of logit is used to calculate probability .

If prob < 0.5 class 1, or else class 2

### Assumptions –

* Balanced target variable- used SMOTE to achieve that
* Absence of multi-collinearity – correlation b/w IV variables is very less here
* Absence of outliers – we have removed these

### Results:



Auc = 0.80

PS: \*\* marked are important ones to consider

## Naïve Bayes

This algorithm is also based on probability

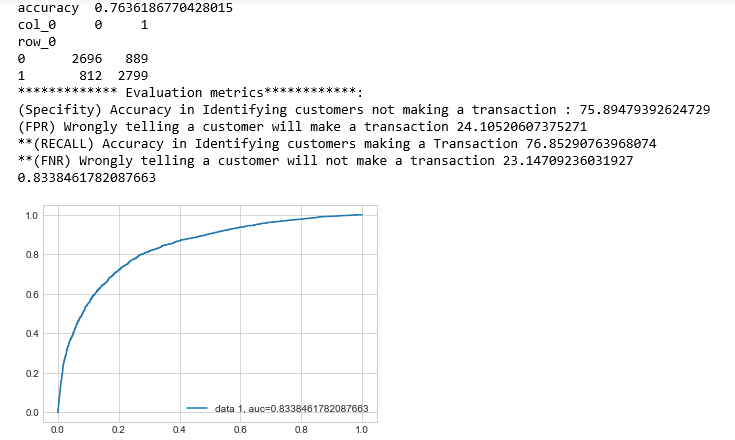
Conditional probability of x’ is calculated

P(yes/x’) = prob of yes provided x’ has happened and p(No/x’) is calculated and the greater value is chosen as the label of x’

### Assumptions

* Zero frequency problem - For a new category, that we don’t have any data for that new category in training set.
* Assumption of IV variables

### Results



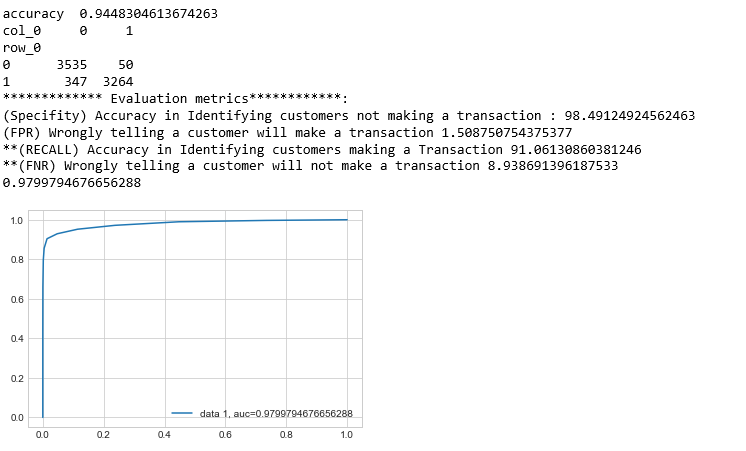
Auc = 0.83 better than logistic regression

## RandomForest

This algorithm works on GINI index , variable with low GINI index is chosen as the root and then expanded and at each level this is repeated

RandomForest doesn’t assume any normalized data, not affected by outliers, and not even colinearity and hence performs the best

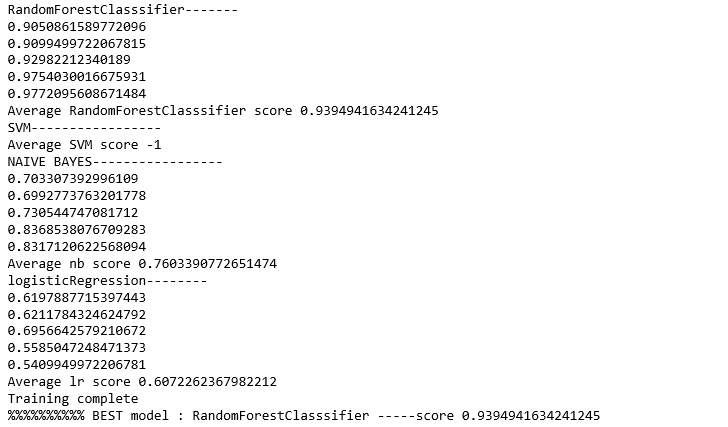
### Results



Auc is 0.979 (This is by far the best)

## RESULTS –

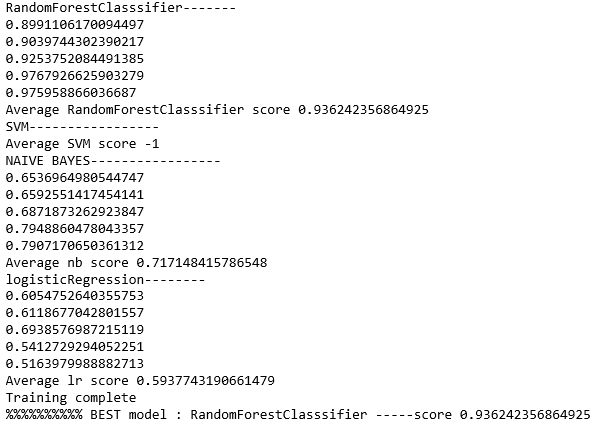
Below I have pasted the model scores in 5 folds using cross-validated k fold strategy



# Alternative approach(Approach 2)

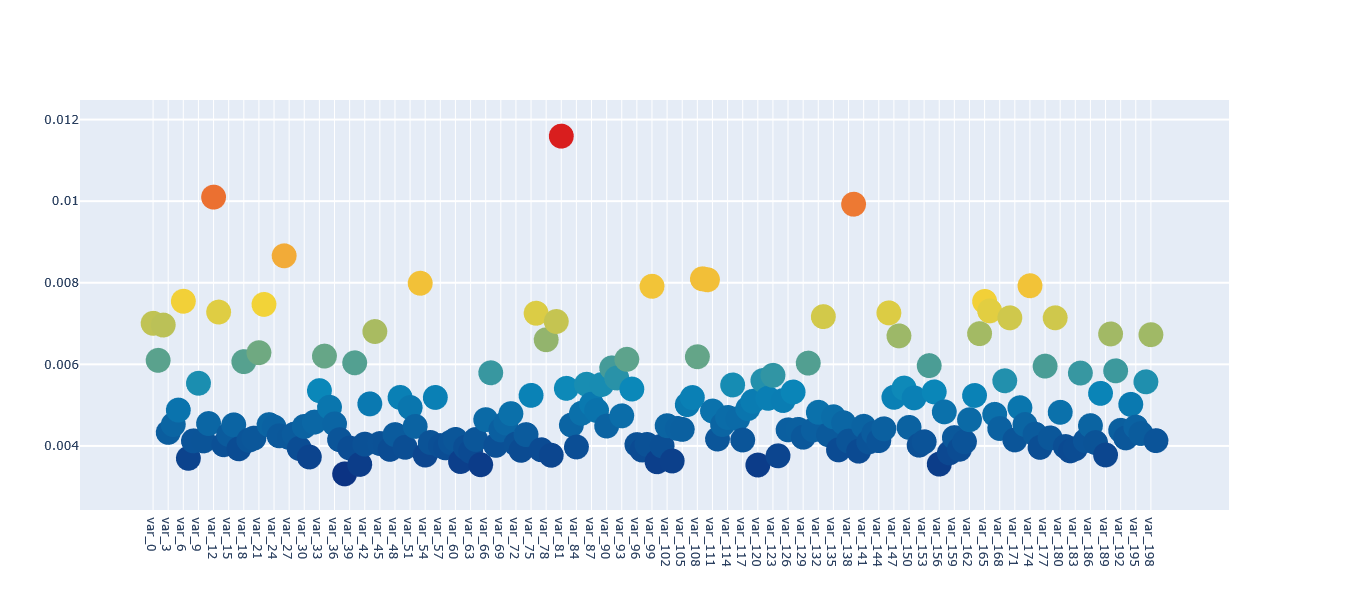
Top 100 features

Let us try taking best 100 features(top 100 correlation values wrt target) into train\_top100 dataframe and compare them train dataframe(all 201 features) and see if any significant difference exists



Model gives best performance when all features are passed, rather than top 100 correlated one, since the corr itself is very low

# RandomForest feature importance



Clearly , var\_12, var\_81 and var\_139 are more important than rest