

CredX - BFS Capstone Project

MID SUBMISSION

Srinivasan Narayanaswamy
Sri Yogesh
Venuder Reddy
Jeeja Gopinath

Business Objectives

"CredX" a leading credit card provider, gets thousands of credit card applications every year. But the problem is it is experiencing an increase in credit loss and therefore we have to find out the reason for this increase in credit loss and also to use the historical data to build a predictive model that can distinguish between the good and bad customers



Approach

- Reviewing provided dataset
- Look for any inconsistency like duplicate rows or application id.
- Check for NA's and missing values.
- Create WOE/Information Values for all parameters.
- We can replace missing values with WOE.
- We can have two dataset, one with WOE values and one with original values.
- We can create model using logistic regression on both the data set i.e with woe values and without woe values.



Approach

- Build other models like Random Forest to handle class imbalance.
- By looking at confusion matrix we can check cutoffs and using Gain chart model evaluation can be done.
- On our final model we can create application scorecard.



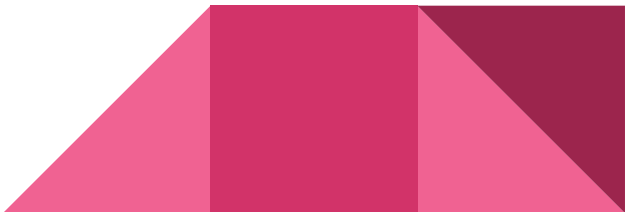
Data Understanding

We have two Data sets : demographic and credit bureau data.

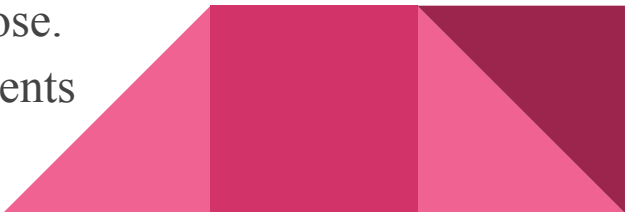
- Demographic/application data: This is obtained from the information provided by the applicants at the time of credit card application. It contains customer-level information on age, gender, income, marital status, etc.
- Credit bureau: This is taken from the credit bureau and contains variables such as 'number of times 30 DPD or worse in last 3/6/12 months', 'outstanding balance', 'number of trades', etc.



Data Understanding

- Both the data sets have equal number of observation i.e 71295, the credit data has 19 variables and demographic data has 12 variables.
 - Credit bureau data has 18 independent variable and 1 dependent variable "Performance.Tag", and all variable are of integer type.
 - The demographic data has 11 independent variable and 1 dependent variable "Performance.Tag", with 7 integer and 5 character variable.
 - Demographic :1428 na values.
 - Credit data has 3028 na values.
 - Demographic data also has blank values .
- 

Data Understanding

- Age variable has the lowest value as -3 which is clearly an error and also Income has -0.5 as its lowest value, an outlier.
 - Dependent variable has 1425 NA's, which means that this population has been denied for loan.
 - There are 3 observation have duplicate application id, we will remove those observation from our demographic data and credit data.
 - Both data set have 71295 observations, we will merge them.
 - We will then check for the percentage of NA's in performance tag and as the % is very less of the total observation we can safely remove those.
 - Other attributes like Gender, Marital.Status, No.of.dependents etc have NA's.
- 

WoE and IV Analysis

- Weight of Evidence and Information Value Analysis were conducted on the various features of the two datasets.
- The WoE and IV was computed for each variable of the demographic dataset. The numeric features were binned to compute the WoE for each bin.. The IC was also generated.
- The bins were made coarser to improve monotonicity.
- As per the standard reference for IV values there are no strong predictor variables in the demo dataset
- Income and no of Months in the current residence are two weak predictors.

Reference :

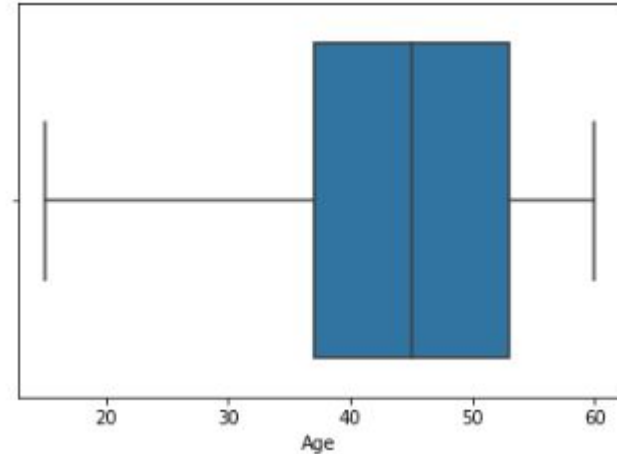
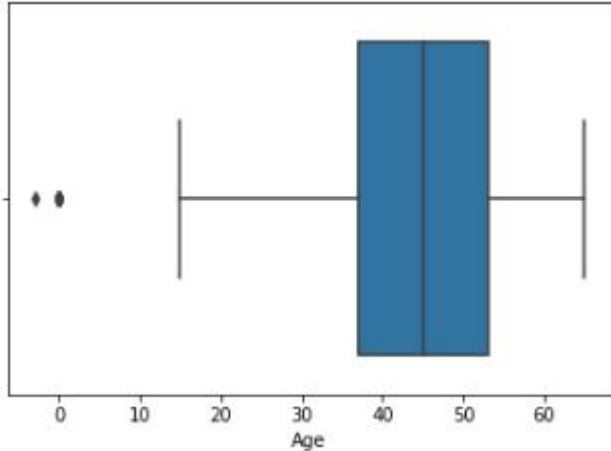
<https://www.listendata.com/2015/03/weight-of-evidence-woe-and-information.html>

Null Percentages

→ demographic and credit bureau data.

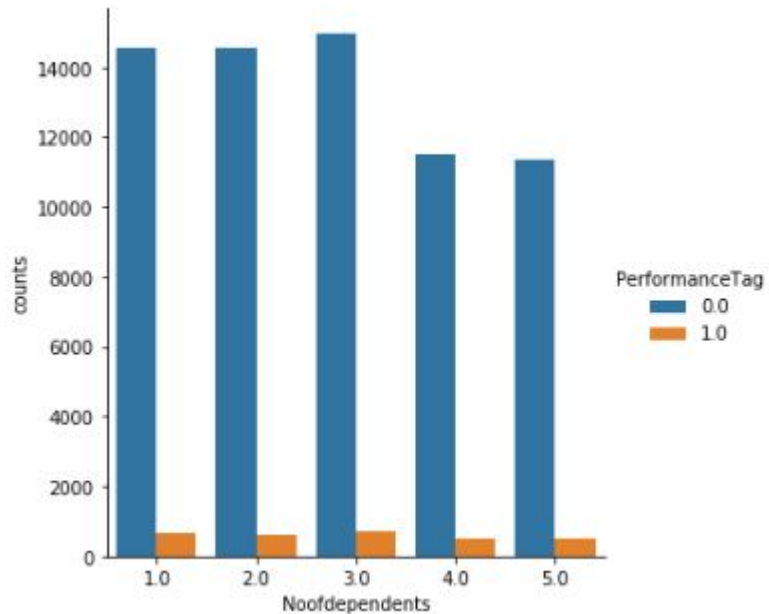
Null percentages:	
Application ID	0.0 %
No of times 90 DPD or worse in last 6 months	0.0 %
No of times 60 DPD or worse in last 6 months	0.0 %
No of times 30 DPD or worse in last 6 months	0.0 %
No of times 90 DPD or worse in last 12 months	0.0 %
No of times 60 DPD or worse in last 12 months	0.0 %
No of times 30 DPD or worse in last 12 months	0.0 %
Avgas CC Utilization in last 12 months	1.48 %
No of trades opened in last 6 months	0.0 %
No of trades opened in last 12 months	0.0 %
No of PL trades opened in last 6 months	0.0 %
No of PL trades opened in last 12 months	0.0 %
No of Inquiries in last 6 months (excluding home & auto loans)	0.0 %
No of Inquiries in last 12 months (excluding home & auto loans)	0.0 %
Presence of open home loan	0.38 %
Outstanding Balance	0.38 %
Total No of Trades	0.0 %
Presence of open auto loan	0.0 %
Performance Tag_x	2.0 %
Age	0.0 %
Gender	0.0 %
Marital Status (at the time of application)	0.01 %
No of dependents	0.0 %
Income	0.0 %
Education	0.17 %
Profession	0.02 %
Type of residence	0.01 %
No of months in current residence	0.0 %
No of months in current company	0.0 %
Performance Tag_y	2.0 %
dtype: object	

EDA & Outlier Treatment



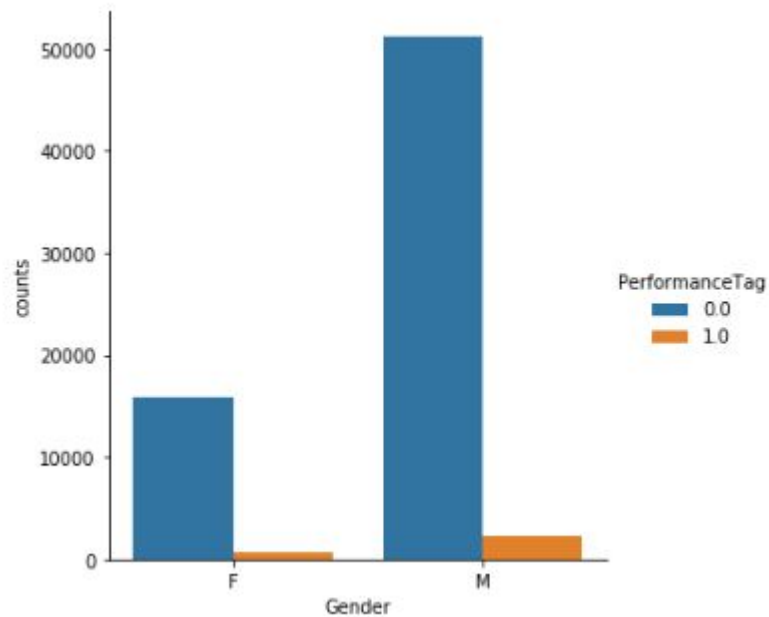
Replace age less than 15 with 16 and age greater than 60 with 60 - Agesegment

EDA & Outlier Treatment

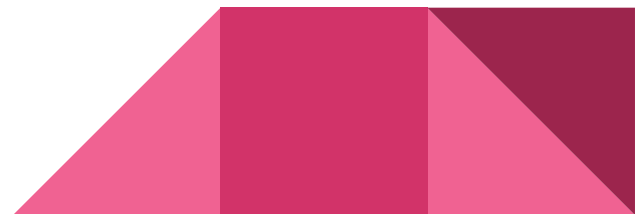


Noofdependents against count

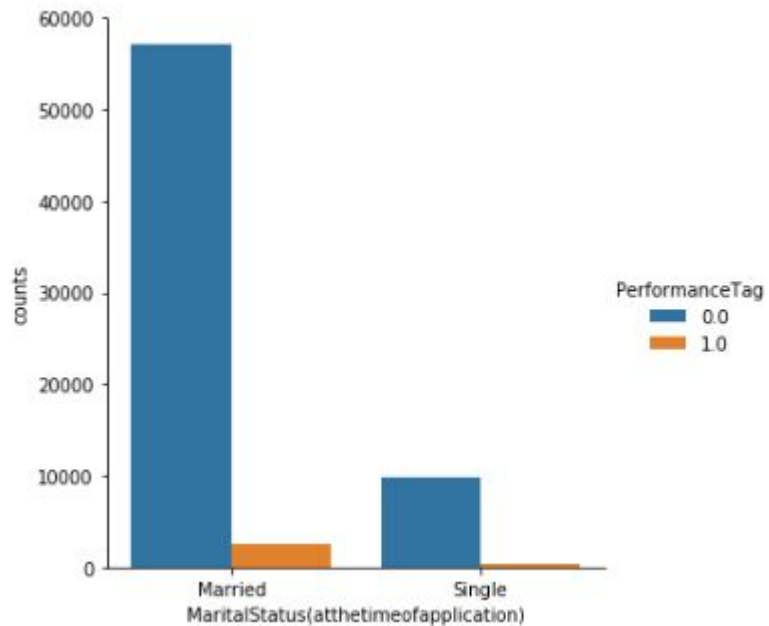
EDA & Outlier Treatment



Gender against Count



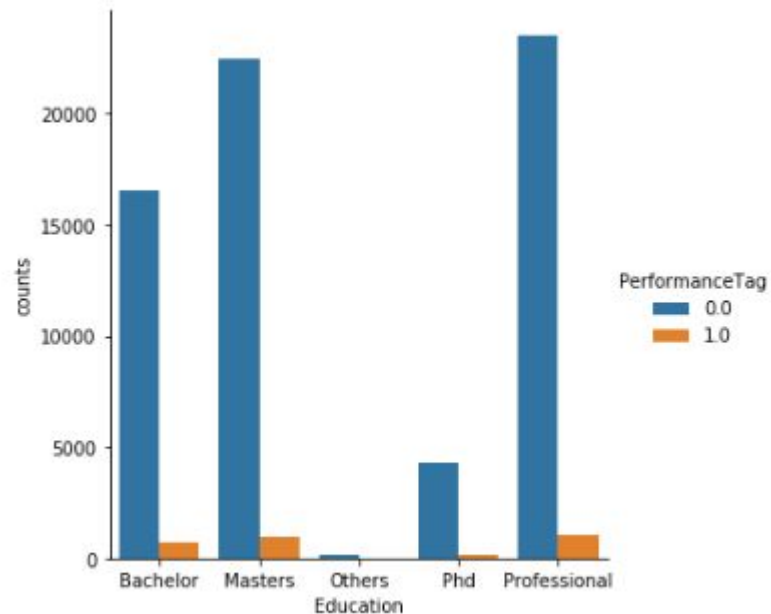
EDA & Outlier Treatment



Marital Status against count

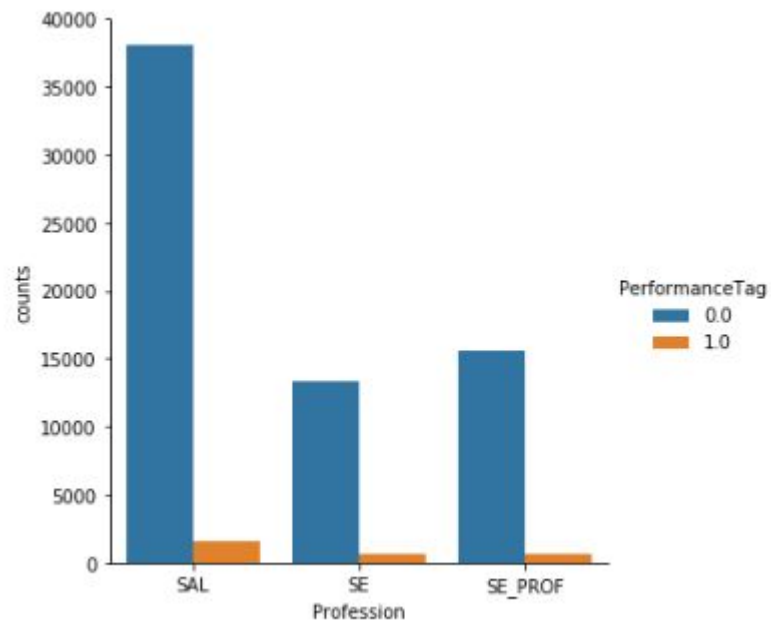


EDA & Outlier Treatment



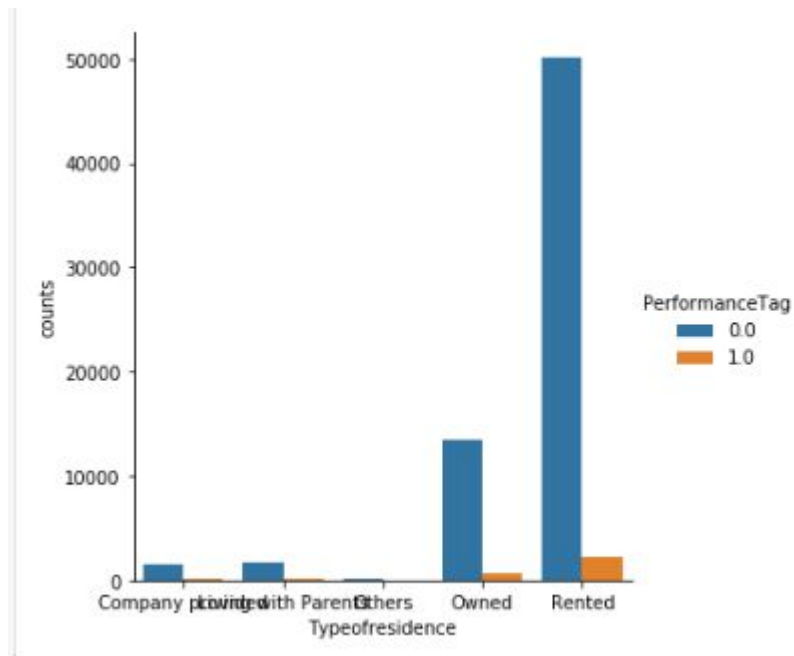
Education against count

EDA & Outlier Treatment



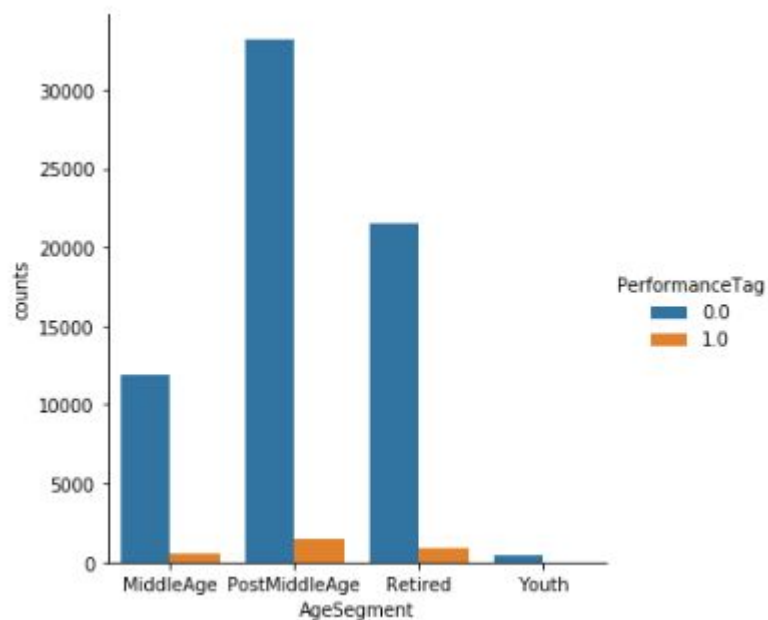
Profession againt count

EDA & Outlier Treatment

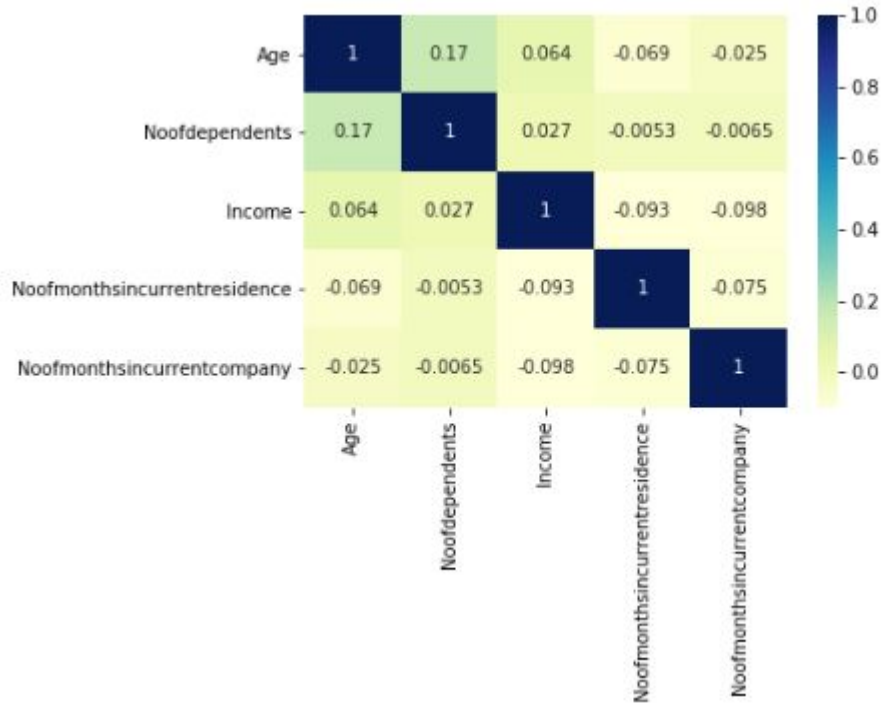


Type of residence against count

EDA & Outlier Treatment - Agesegment

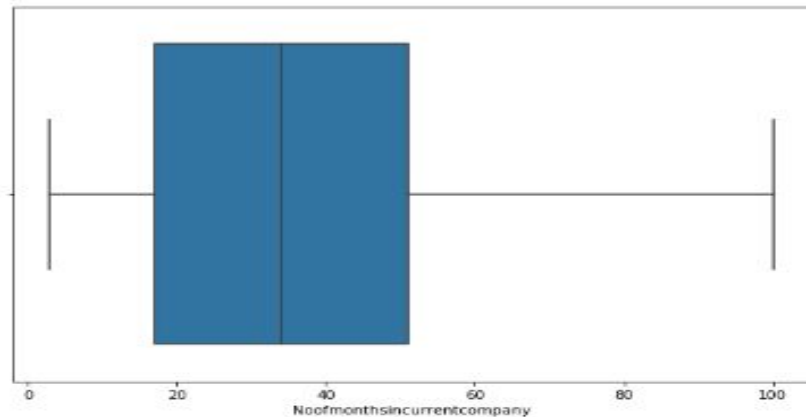
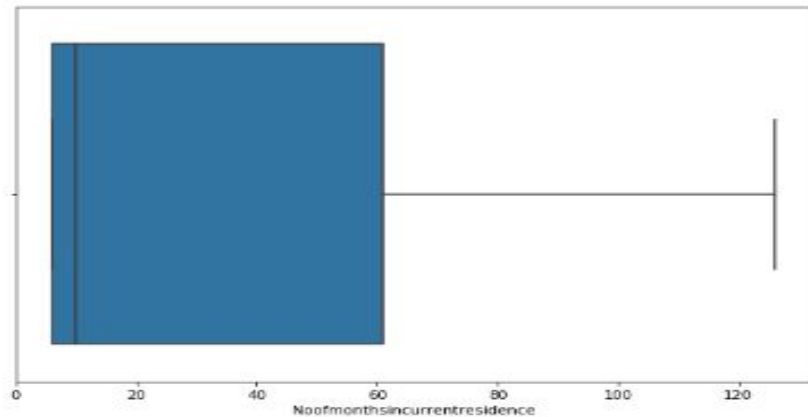
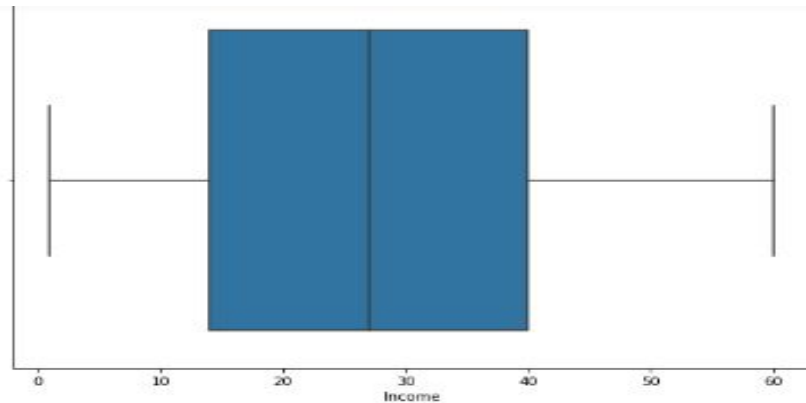
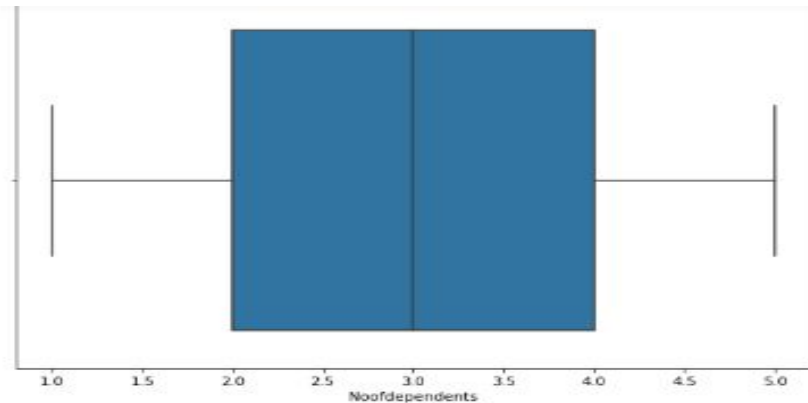


EDA & Outlier Treatment

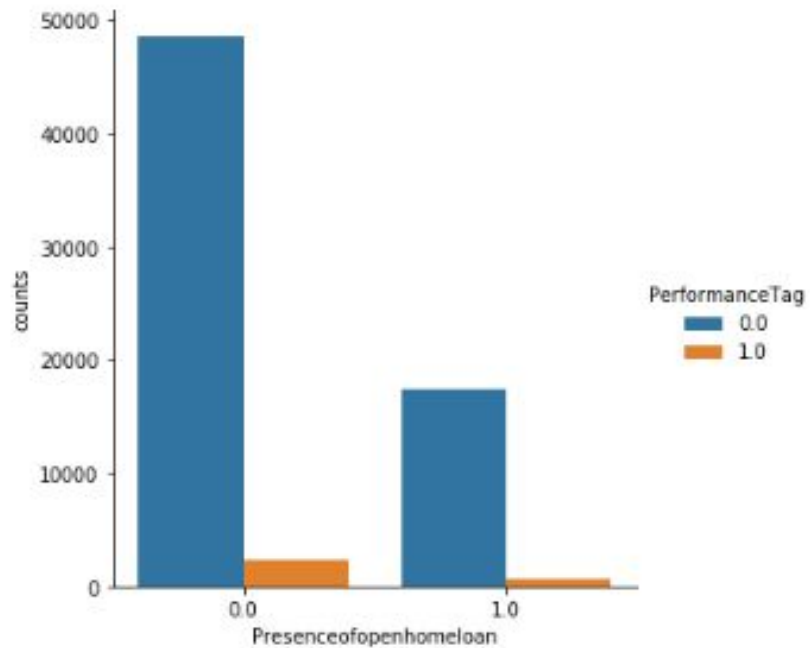


Correlation b/n numeric variables.

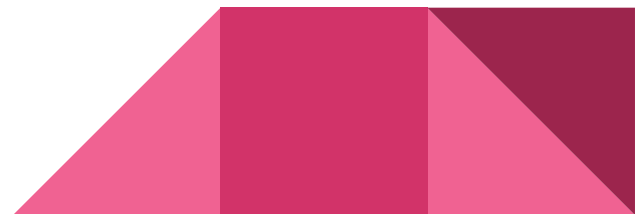
EDA & Outlier Treatment - boxplot after handling



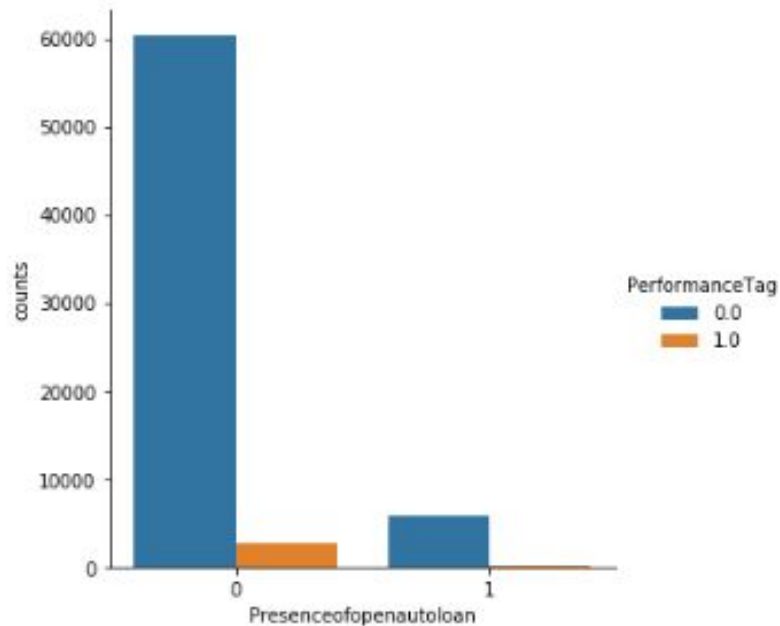
EDA & Outlier Treatment



Presense of home loan against
count

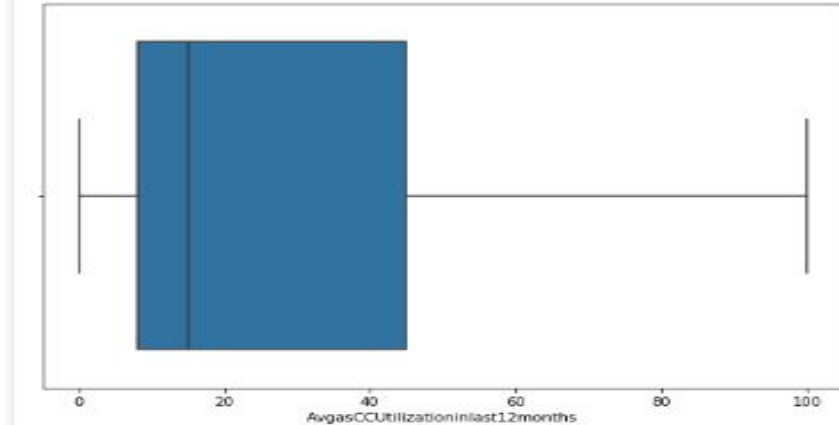
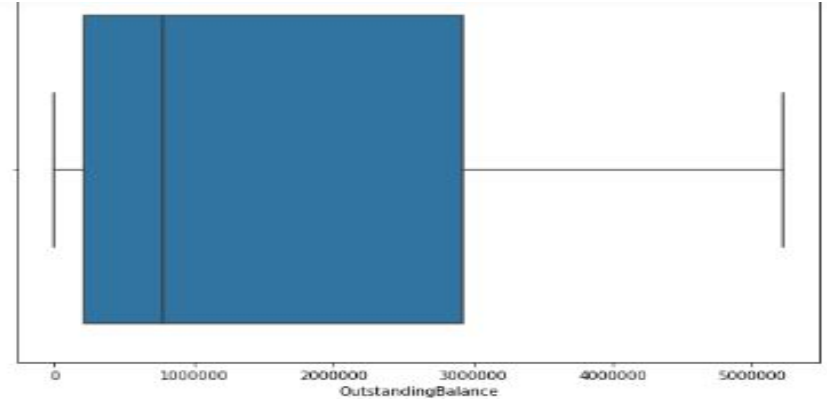
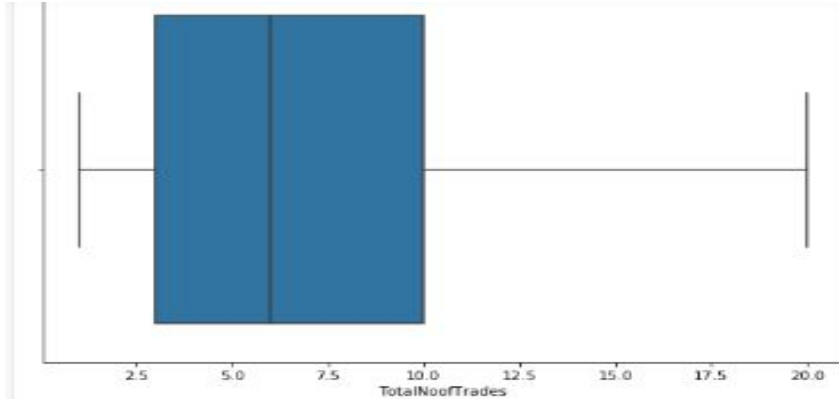


EDA & Outlier Treatment

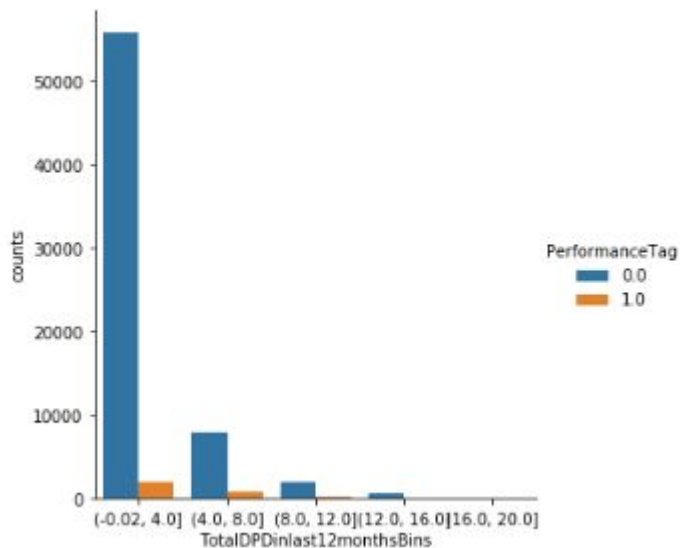
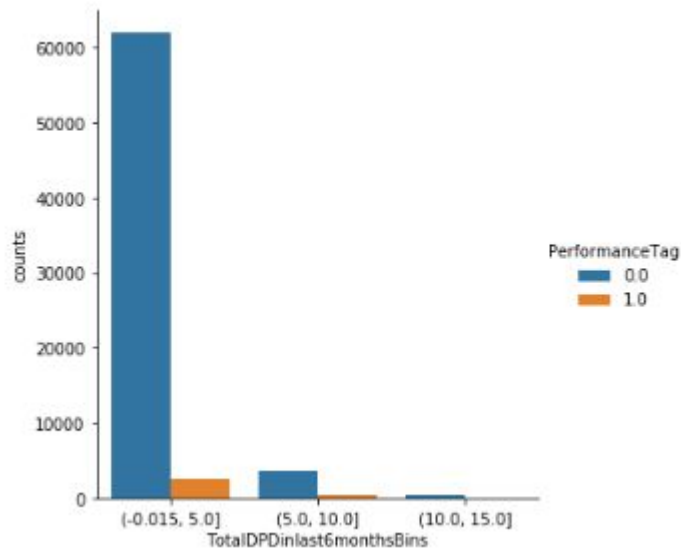


Presense of auto load against count

EDA & Outlier Treatment - boxplot after handling

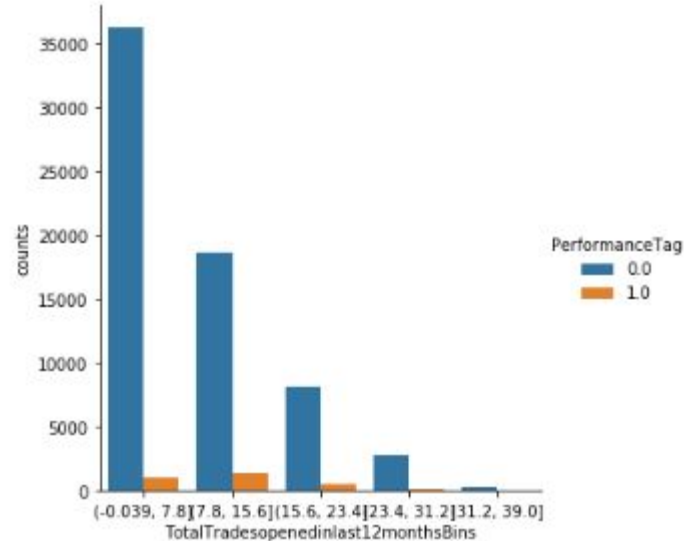
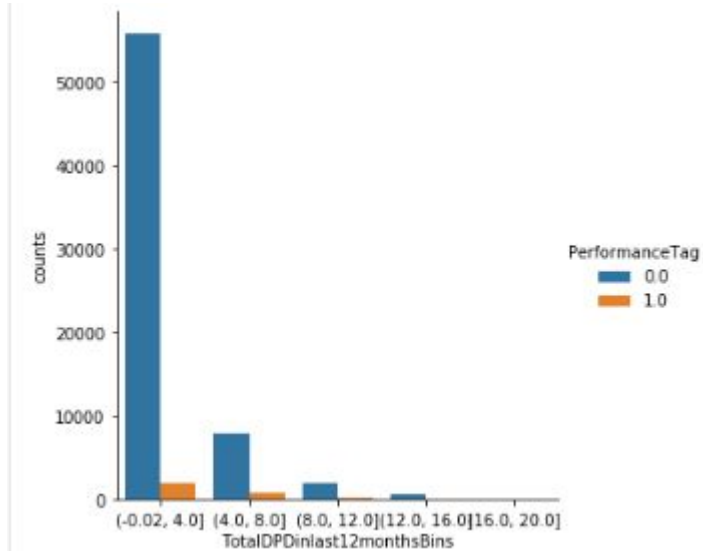


EDA & Outlier Treatment



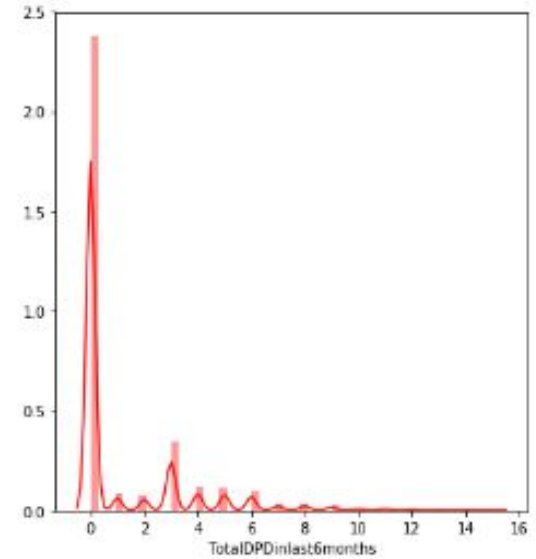
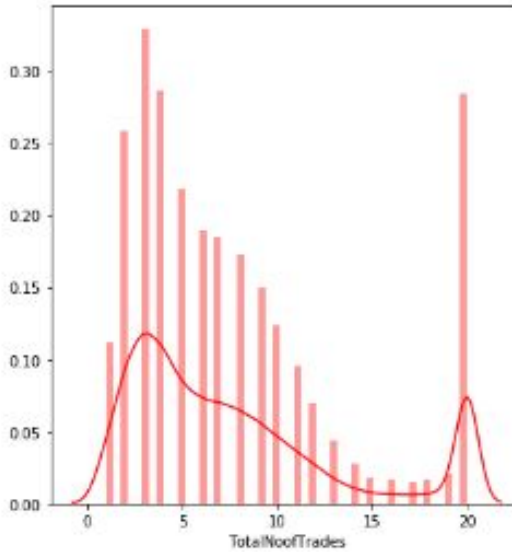
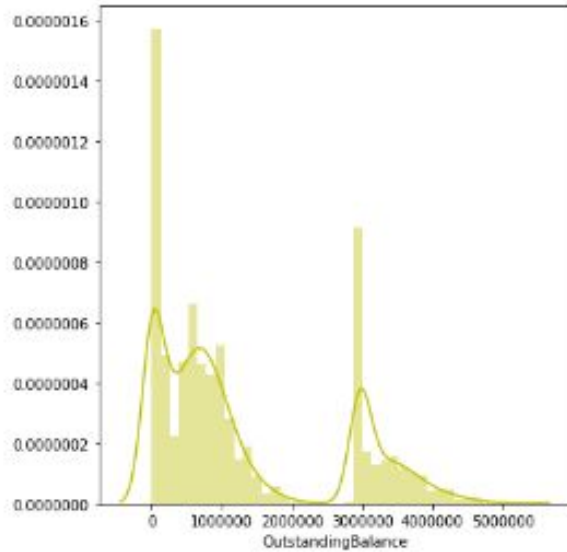
Plotting after creation of bins.

EDA & Outlier Treatment



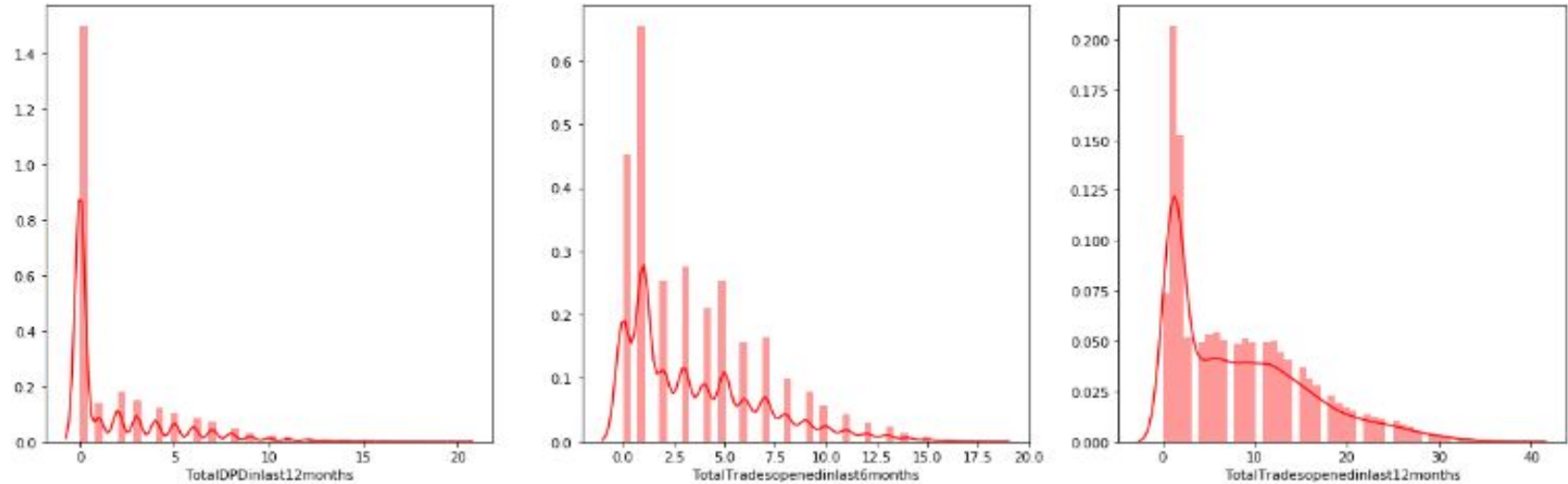
Plotting after creation of bins.

EDA & Outlier Treatment



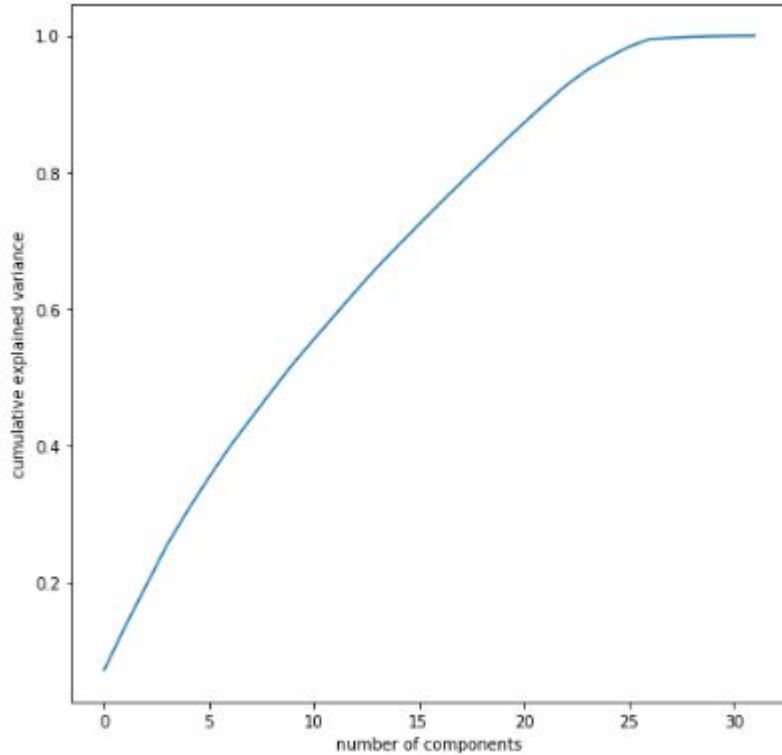
Scatter plot for the variables

EDA & Outlier Treatment



Scatter plot for the variables

Scree plot - No of components



WoE and IV Analysis

WoE values

< 0.02 useless for prediction

0.02 to 0.1 Weak predictor

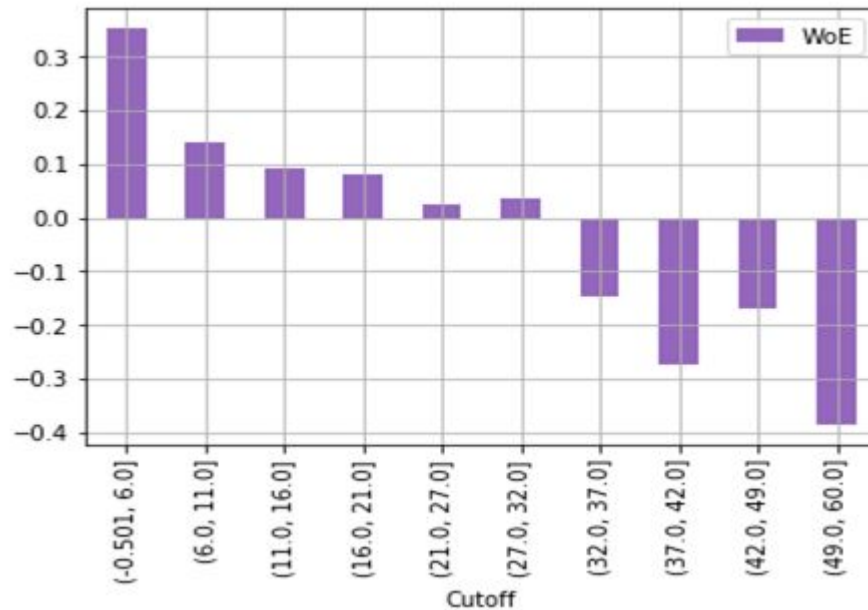
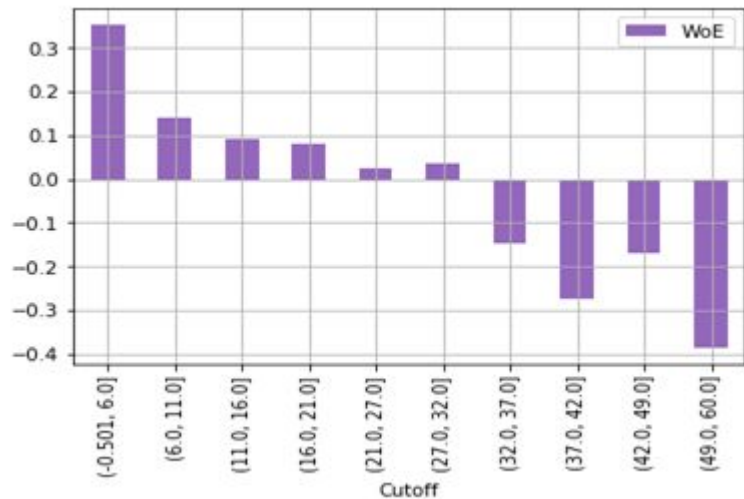
0.1 to 0.3 Medium predictor

0.3 to 0.5 Strong predictor

0.5 Suspicious or too good.

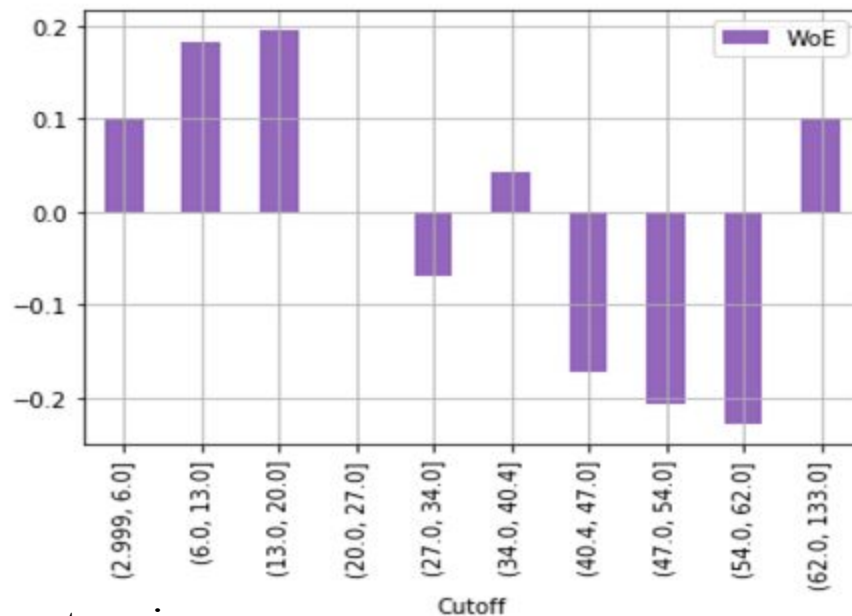


WoE and IV Analysis



Income & Age respectively. WoE values w,r,t various bins/categories.(representative of monotonicity)

WoE and IV Analysis



Income WoE values w,r,t various
bins/categories.(representative of monotonicity)

WoE & IV Analysis

IV Variables - Notable that there are few medium predictors to be further analysed.

Application ID	0.001594
90DPD6	0.162983
60DPD6	0.211539
30DPD6	0.24446
90DPD12	0.216015
60DPD12	0.18854
30DPD12	0.218892
CCUtil	0.321194
tradeOpen6	0.187406

tradeOpen12	0.294371
PLtradeOpen6	0.224342
PLtradeOpen12	0.258768
NoInq6	0.113095
NoInq12	0.245281
OpenHomeLoan	0
OutstandBalance	0.247832
TotTrades	0.23167
OpenAutoLoan	0.001665
hit_flag	0.000564

WoE & IV Analysis -Function to calculate WoE

```
def iv_woe(data, target, bins=10, show_woe=False):

    #Empty Dataframe
    newDF,woeDF = pd.DataFrame(), pd.DataFrame()

    #Extract Column Names
    cols = data.columns

    #Run NOE and IV on all the independent variables
    for ivars in cols[~cols.isin([target])]:
        if (data[ivars].dtype.kind in 'bifc') and (len(np.unique(data[ivars]))>10):
            binned_x = pd.qcut(data[ivars], bins, duplicates='drop')
            d0 = pd.DataFrame({'x': binned_x, 'y': data[target]})
        else:
            d0 = pd.DataFrame({'x': data[ivars], 'y': data[target]})
        d = d0.groupby("x", as_index=False).agg({'y': ["count", "sum"]})
        d.columns = ['Cutoff', 'N', 'Events']
        d['% of Events'] = np.maximum(d['Events'], 0.5) / d['Events'].sum()
        d['Non-Events'] = d['N'] - d['Events']
        d['% of Non-Events'] = np.maximum(d['Non-Events'], 0.5) / d['Non-Events'].sum()
        d['WoE'] = np.log(d['% of Events']/d['% of Non-Events'])
        d['IV'] = d['WoE'] * (d['% of Events'] - d['% of Non-Events'])
        d.insert(loc=0, column='Variable', value=ivars)
        print("Information value of " + ivars + " is " + str(round(d['IV'].sum(),6)))
        temp =pd.DataFrame({"Variable": [ivars], "IV": [d['IV'].sum()]}, columns = ["Variable", "IV"])
        newDF=pd.concat([newDF,temp], axis=0)
        woeDF=pd.concat([woeDF,d], axis=0)

    #Show NOE Table
    if show_woe == True:
        print(d)
    return newDF, woeDF

iv, woe = iv_woe(data = demographic, target = 'Perf_Tag', bins=10, show_woe = True)
print(iv)
print(woe)
```


Application scorecard Calculation

Application score card for odds of 10 to 1 is 400. Score increases by 20 points for doubling odds. Steps are below.

- Compute the probabilities of being defaulted by using predict function, after your modelling for entire population.
- Then, you need to compute odds for good. Since the probability computed is for rejection (bad customers), $\text{Odd}(\text{good}) = (1 - P(\text{bad})) / P(\text{bad})$
- Then, compute $\ln(\text{odd}(\text{good}))$.
- Use the following formula for computing application score card :
 $400 + \text{slope} * (\ln(\text{odd}(\text{good})) - \ln(10))$ where slope is $20 / (\ln(20) - \ln(10))$.

Future Roadmap

- Build other models (Random forest) for the given data to arrive at the optimal model and compare all the metrics
- Build models using different methods to handle class imbalance
- Develop the application scorecard.



Thank You !

