Fliprobo

Micro-Credit Defaulter Model

Report

Submitted by:

Arjun Verma,

Intern Data Scientist

ACKNOWLEDGMENT

I would like to express my greatest appreciation to the all individuals who have helped and supported me throughout the project. I am thankful to Fliprobo team for their ongoing support during the project, from initial advice, and encouragement, which led to the final report of this project.

A special acknowledgement goes to my institute Datatrained who helped me in completing the project and learning concepts.

I wish to thank my parents as well for their undivided support and interest who inspired me and encouraged me to go my own way, without whom I would be unable to complete my project.

Below following are the other references:

www.towardsdatascience.com

www.medium.com

www.stackoverflow.com

Datatrained Lectures

INTRODUCTION

➢ Business Problem Framing

A Microfinance Institution (MFI) is an organization that offers financial services to low income populations. MFS becomes very useful when targeting especially the unbanked poor families living in remote areas with not much sources of income. The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on.

Many microfinance institutions (MFI), experts and donors are supporting the idea of using mobile financial services (MFS) which they feel are more convenient and efficient, and cost saving, than the traditional high-touch model used since long for the purpose of delivering microfinance services. Though, the MFI industries is primarily focusing on low income families and are very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.

Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.

We are working with one such client that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.

They understand the importance of communication and how it affects a person's life, thus, focusing on providing their services and products to low income families and poor customers that can help them in the need of hour.

They are collaborating with an MFI to provide micro-credit on mobile balances to be paid back in 5 days. The Consumer is believed to be defaulter if he deviates from the path of paying back the loaned amount within the time duration of 5 days. For the loan amount of 5 (in Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

The sample data is provided to us from our client database. It is hereby given to you for this exercise. In order to improve the selection of customers for the credit, the client wants some predictions that could help them in further investment and improvement in selection of customers.

Build a model which can be used to predict in terms of a probability for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan. In this case, Label '1' indicates that the loan has been payed i.e. Non- defaulter, while, Label '0' indicates that the loan has not been payed i.e. defaulter.

➢ Conceptual Background of the Domain Problem

We are required to model the label of defaulter with the available independent variables. This model will be used by the management to understand how exactly the customer intention for loan vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas. Further, the model will be a good way for the management to understand the customer intention for the returning loan.

Solution we find that building a machine learning model that can predict upcoming new customer that have taken loan is a defaulter or not from previous dataset. Here we implement 9 models and find out best machine learning models.

➢ Review of Literature

1. **Here we categorized data into three parts one for training model, second one for cross validation and last one for testing the model and deploying this model for prediction upcoming customers goodwill.**

2. **Loan defaulter labels are depends on the various features which we will show later observations how different feature impact customer intention for returning loan.**

3. **For building a best model for prediction we did EDA and several mandatory requirement procedures for enhancing and improving model accuracy to predict loan labels.**

➢ Motivation for the Problem Undertaken

Genuinely it's a need of the financial sector to complete their goal with higher revenue and low expenditure. Hence this model can brings higher revenue because we can predict upcoming goodwill of the customer and take action accordingly.

➢ Mathematical/ Analytical Modeling of the Problem

Data is statistically analysed through variance inflation factor. Analysed through correlation, CHI2 hypothesis testing and multicollinearity. Graphical modelling done through seaborn and matplotlib to understanding how different features impact dataset.

➢ Data Sources and their formats

Datasets are provided by fliprobo for building machine learning model to predict the customer defaulter label based on given parameter.

Dataset are in three parts one is for training model, second one is for cross validating and last one for predicting.

Train dataset: Dataset is having 179760 rows and 85 columns including target.

Cross Validation dataset: CV dataset having 77040 rows and 85 columns.

Test dataset: Dataset is having 11058 rows and 85 columns.

The information about features are as follows

```
'Unnamed: 0', 'label', 'msisdn', 'aon', 'daily_decr30',
'daily_decr90','rental30', 'rental90', 'last_rech_date_ma',
'last_rech_date_da','last_rech_amt_ma', 'cnt_ma_rech30',
'fr_ma_rech30','sumamnt_ma_rech30', 'medianamnt_ma_rech30',
'medianmarechprebal30','cnt_ma_rech90', 'fr_ma_rech90',
'sumamnt_ma_rech90','medianamnt_ma_rech90', 'medianmarechprebal90',
'cnt_da_rech30','fr_da_rech30', 'cnt_da_rech90', 'fr_da_rech90',
'cnt_loans30','amnt_loans30','maxamnt_loans30','medianamnt_loans30',
'cnt_loans90','amnt_loans90','maxamnt_loans90','medianamnt_loans90',
'payback30','payback90', 'pcircle', 'pdate'
```

Variables in details

- Variable Definition label: Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}
- msisdn: mobile number of user
- aon: age on cellular network in days
- daily_decr30: Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
- daily_decr90: Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
- rental30: Average main account balance over last 30 days

- rental90: Average main account balance over last 90 days
- last_rech_date_ma: Number of days till last recharge of main account
- last_rech_date_da: Number of days till last recharge of data account
- last_rech_amt_ma: Amount of last recharge of main account (in Indonesian Rupiah)
- cnt_ma_rech30: Number of times main account got recharged in last 30 days
- fr_ma_rech30: Frequency of main account recharged in last 30 days
- sumamnt_ma_rech30: Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
- medianamnt_ma_rech30: Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
- medianmarechprebal30: Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
- cnt_ma_rech90: Number of times main account got recharged in last 90 days
- fr_ma_rech90: Frequency of main account recharged in last 90 days
- sumamnt_ma_rech90 Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)
- medianamnt_ma_rech90: Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)
- medianmarechprebal90: Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)
- cnt_da_rech30: Number of times data account got recharged in last 30 days
- fr_da_rech30: Frequency of data account recharged in last 30 days
- cnt_da_rech90: Number of times data account got recharged in last 90 days
- fr_da_rech90: Frequency of data account recharged in last 90 days
- cnt_loans30: Number of loans taken by user in last 30 days
- amnt_loans30: Total amount of loans taken by user in last 30 days
- maxamnt_loans30: maximum amount of loan taken by the user in last 30 days
- medianamnt_loans30: Median of amounts of loan taken by the user in last 30 days
- cnt_loans90: Number of loans taken by user in last 90 days
- amnt_loans90: Total amount of loans taken by user in last 90 days
- maxamnt_loans90: maximum amount of loan taken by the user in last 90 days
- medianamnt_loans90: Median of amounts of loan taken by the user in last 90 days
- payback30: Average payback time in days over last 30 days
- payback90: Average payback time in days over last 90 days
- pcircle: telecom circle
- pdate: date

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma	cnt
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	1539	
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	5787	
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	1539	
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	947	
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	2309	

cnt_ma_rech30	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_rech30	medianmarechprebal30	cnt_ma_rech90	fr_ma_rech90	sumamnt_ma_rech90	
2	21.0	3078.0	1539.0	7.50	2	21	3078	
1	0.0	5787.0	5787.0	61.04	1	0	5787	
1	0.0	1539.0	1539.0	66.32	1	0	1539	
0	0.0	0.0	0.0	0.00	1	0	947	
7	2.0	20029.0	2309.0	29.00	8	2	23496	

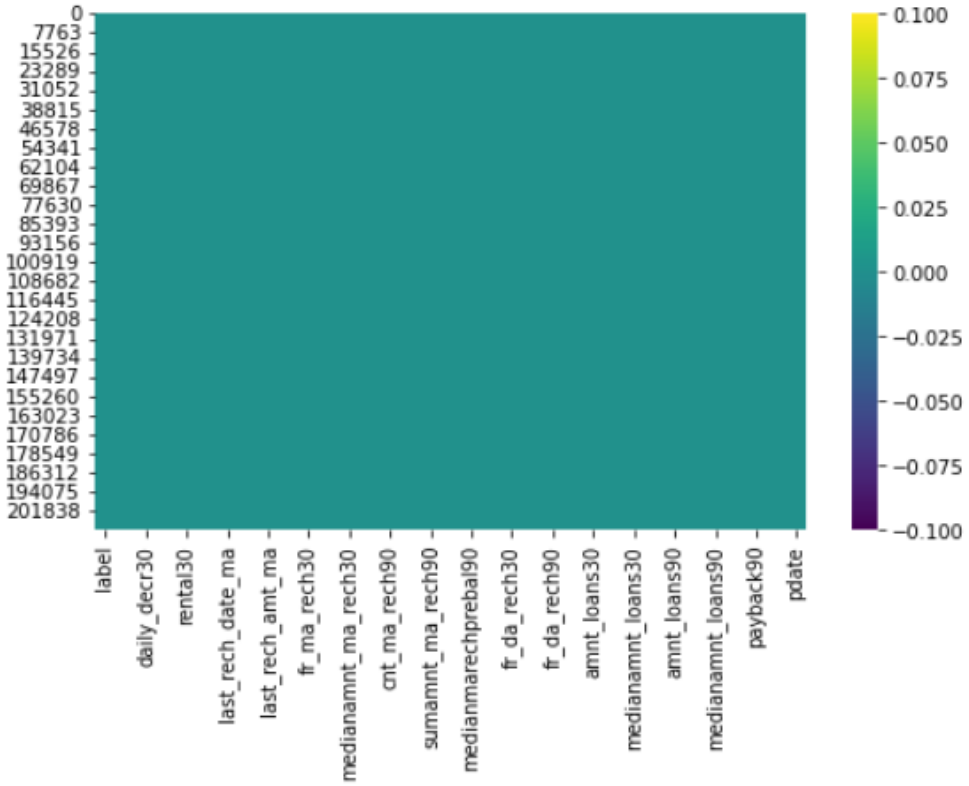
medianamnt_ma_rech90	medianmarechprebal90	cnt_da_rech30	fr_da_rech30	cnt_da_rech90	fr_da_rech90	cnt_loans30	amnt_loans30	maxamnt_loans30	r
1539.0	7.50	0.0	0.0	0	0	2	12	6.0	
5787.0	61.04	0.0	0.0	0	0	1	12	12.0	
1539.0	66.32	0.0	0.0	0	0	1	6	6.0	
947.0	2.50	0.0	0.0	0	0	2	12	6.0	
2888.0	35.00	0.0	0.0	0	0	7	42	6.0	

maxamnt_loans30	medianamnt_loans30	cnt_loans90	amnt_loans90	maxamnt_loans90	medianamnt_loans90	payback30	payback90	pcircle	pdate
6.0	0.0	2.0	12	6	0.0	29.000000	29.000000	UPW	2016-07-20
12.0	0.0	1.0	12	12	0.0	0.000000	0.000000	UPW	2016-08-10
6.0	0.0	1.0	6	6	0.0	0.000000	0.000000	UPW	2016-08-19
6.0	0.0	2.0	12	6	0.0	0.000000	0.000000	UPW	2016-06-06
6.0	0.0	7.0	42	6	0.0	2.333333	2.333333	UPW	2016-06-22

Dataset Information

- 'msisdn', 'pcircle', 'pdate' columns are objects type while rest of the columns are of numerical types.

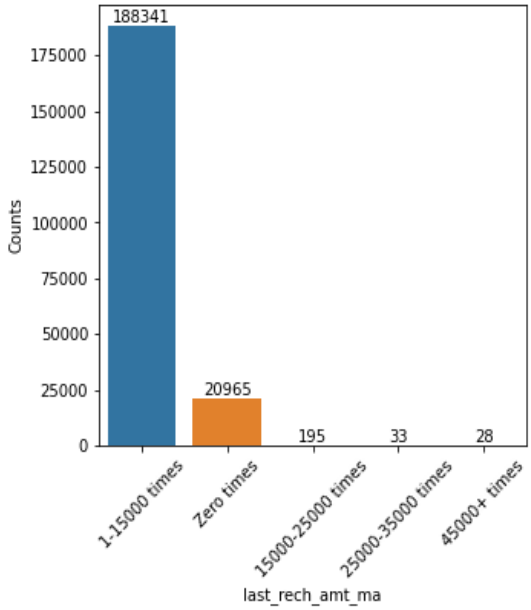
Checking Null Values of the dataset

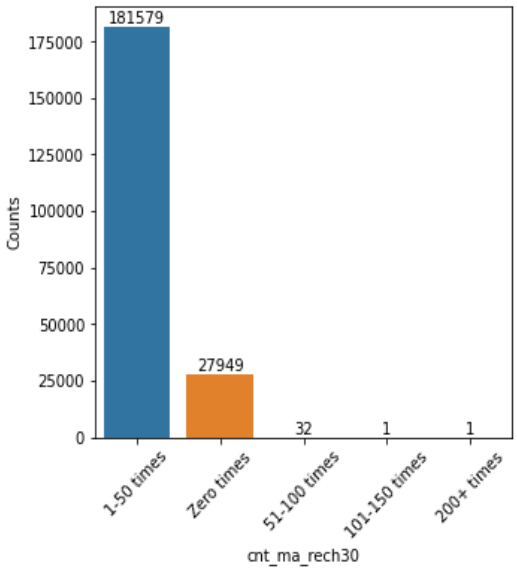


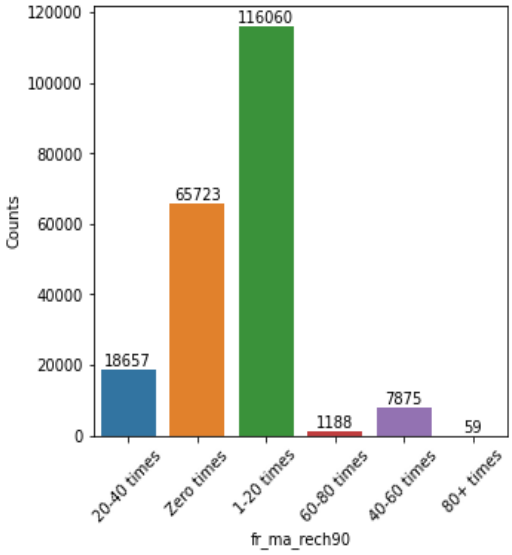
- As per dataset there is no null values present.
- We converted some of parameters into sub parameters for better understanding.
- Drop duplicated values.

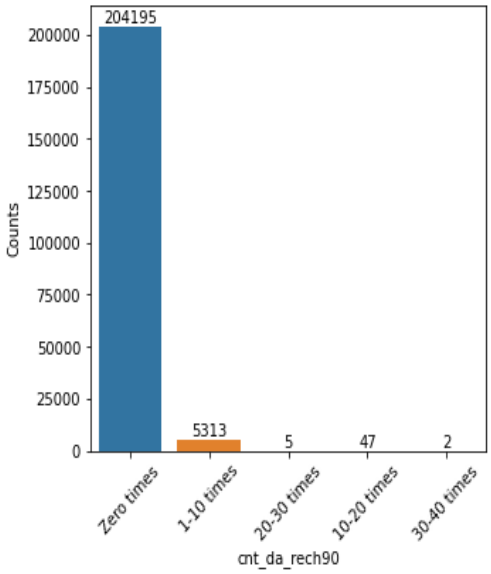
Visualizing Parameters for better understandings

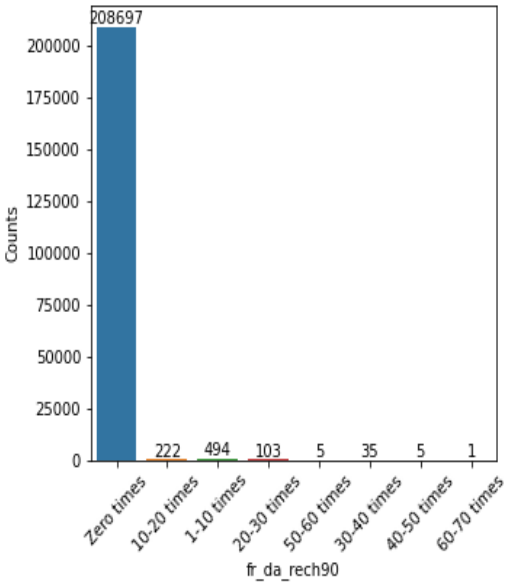
Bar chart

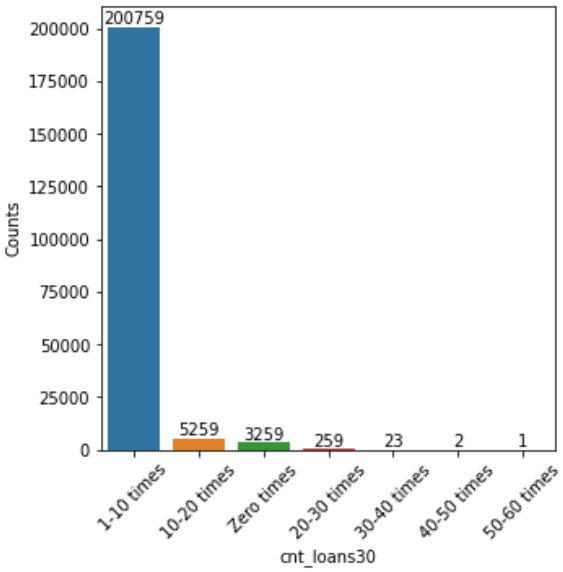


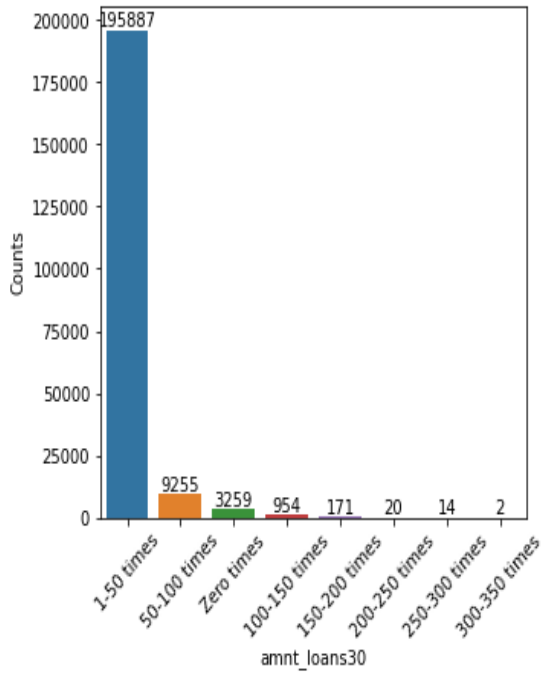


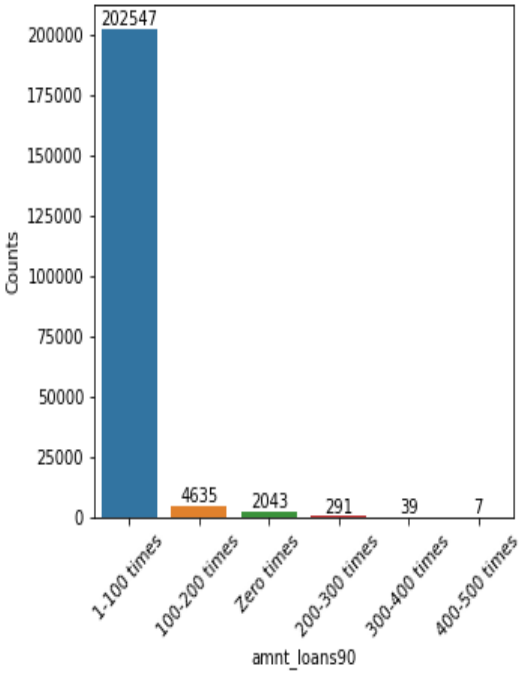


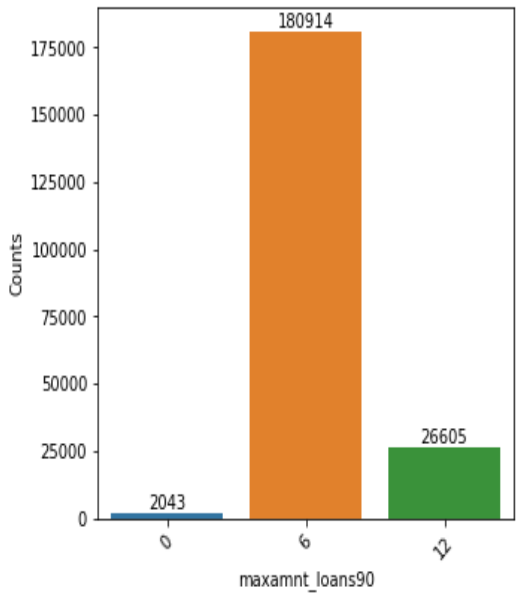


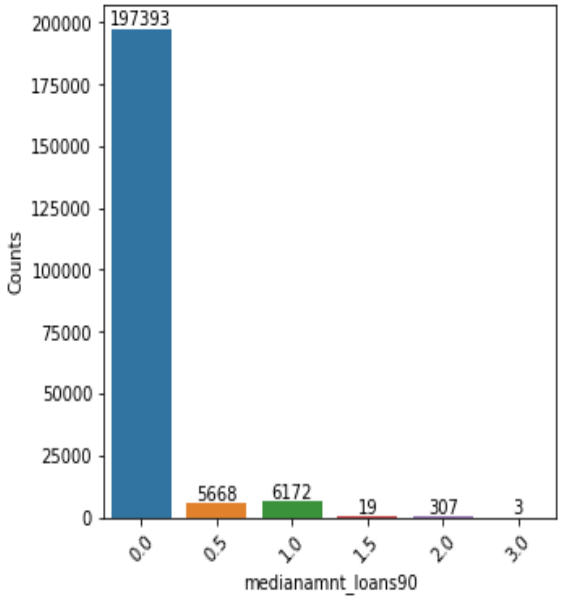

208697
222
494
103
5
35
5
1
Counts
Zero times
10-20 times
1-10 times
20-30 times
50-60 times
30-40 times
40-50 times
60-70 times
fr_da_rech90

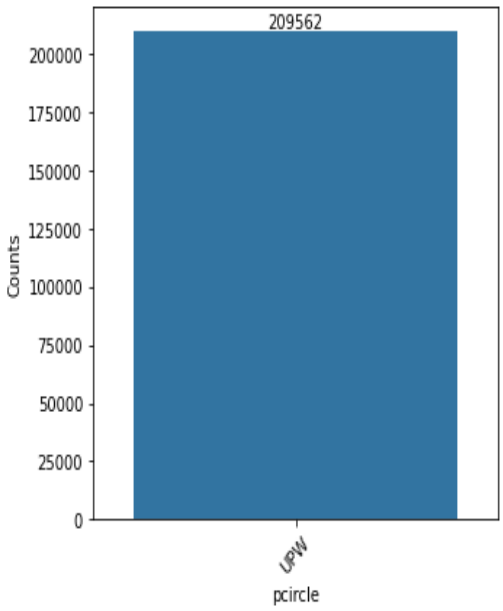

200759
5259
3259
259
23
2
1
Counts
1-10 times
10-20 times
Zero times
20-30 times
30-40 times
40-50 times
50-60 times
cnt_loans30

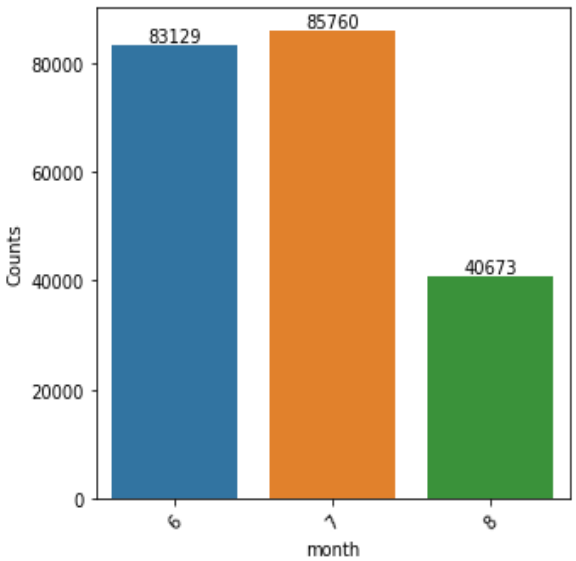

195887
9255
3259
954
171
20
14
2
Counts
1-50 times
50-100 times
Zero times
100-150 times
150-200 times
200-250 times
250-300 times
300-350 times
amnt_loans30


202547
4635
2043
291
39
7
Counts
1-100 times
100-200 times
Zero times
200-300 times
300-400 times
400-500 times
amnt_loans90

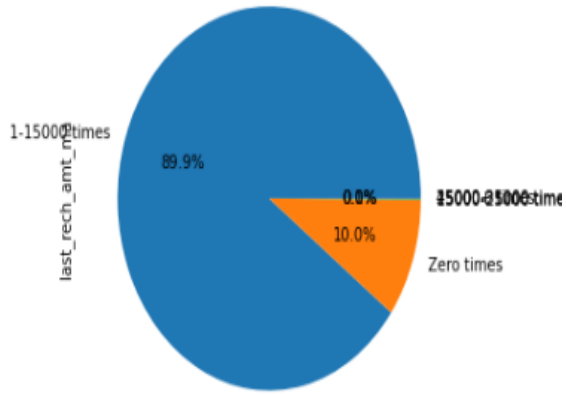

180914
26605
2043
Counts
maxamnt_loans90

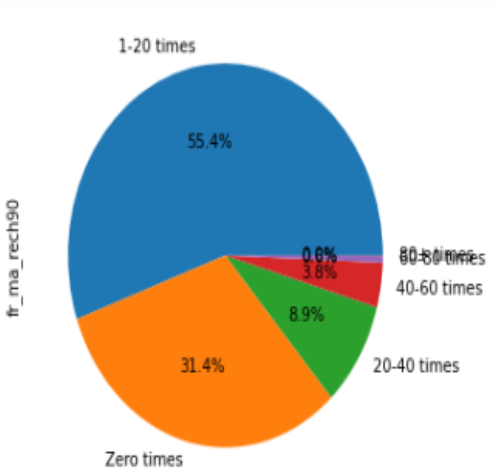

197393
5668
6172
19
307
3
Counts
medianamnt_loans90

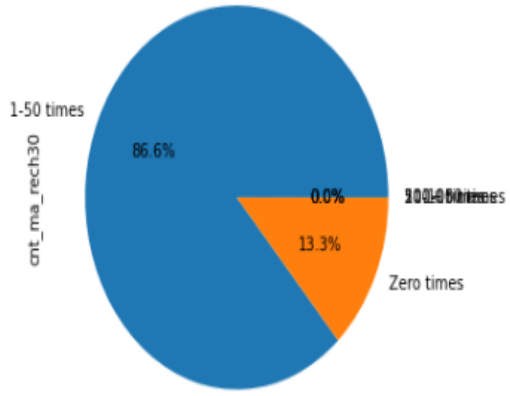

209562
Counts
UPW
pcircle

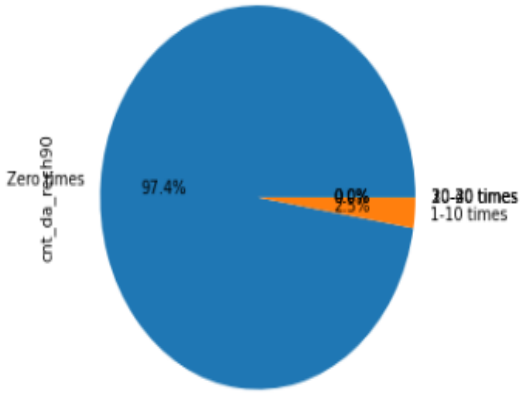

83129
85760
40673
Counts
month

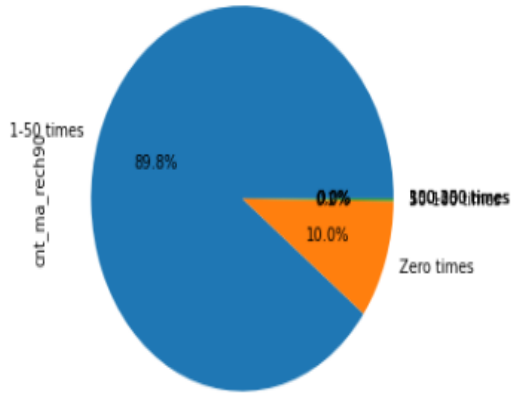
Pie Chart

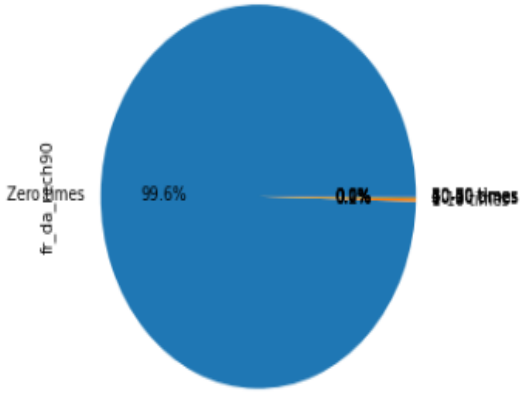

89.9%
10.0%
Zero times

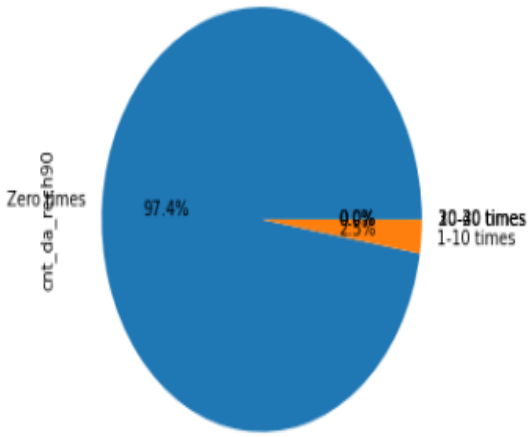

1-20 times
55.4%
40-60 times
8.9%
20-40 times
31.4%
Zero times

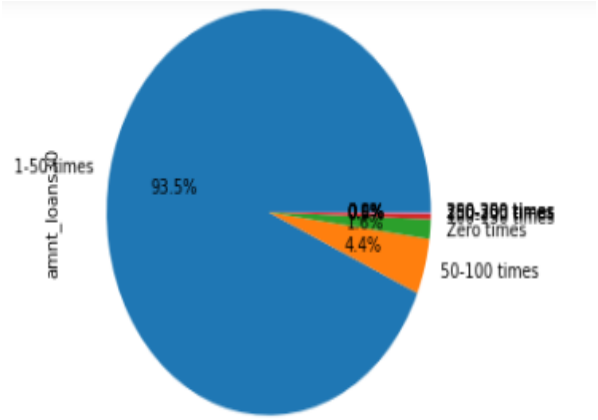

1-50 times
86.6%
13.3%
Zero times

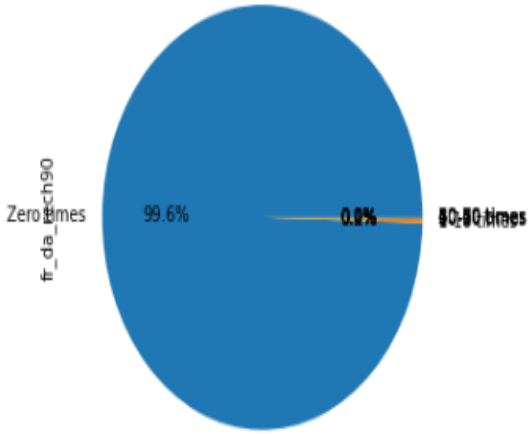

97.4%
1-10 times

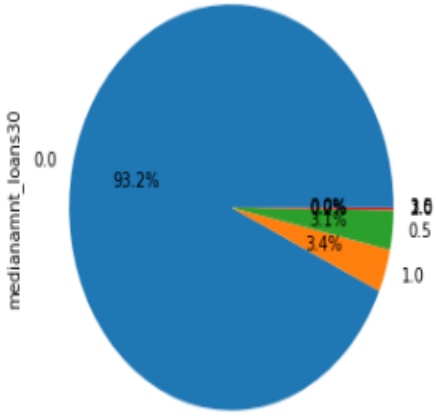

1-50 times
89.8%
10.0%
Zero times

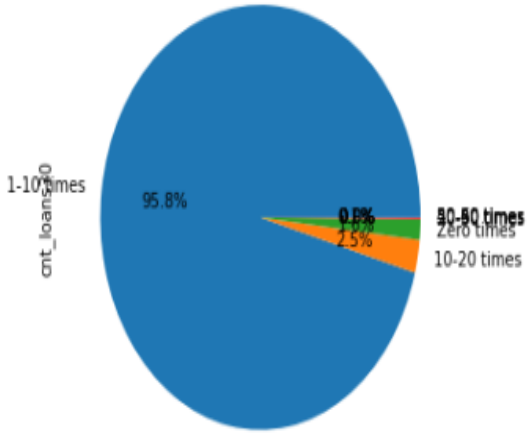

Zero times
99.6%

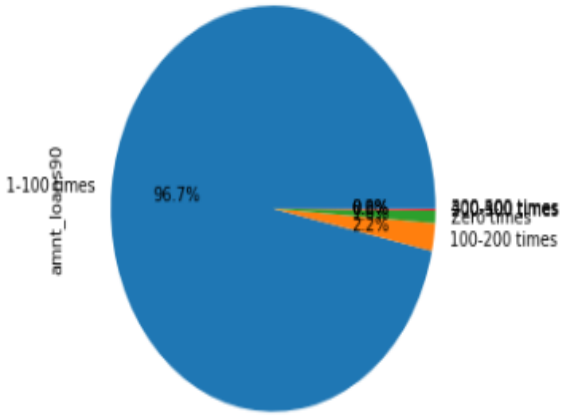

97.4%
2.5%
1-10 times

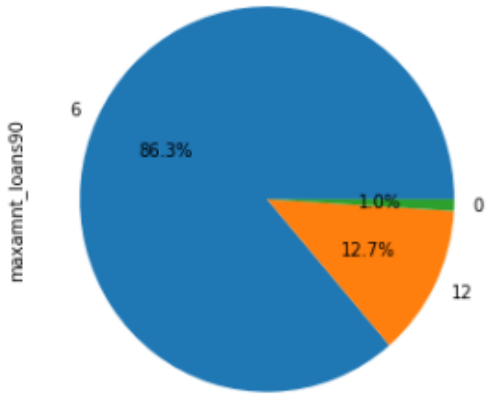

93.5%
4.4%
Zero times
50-100 times

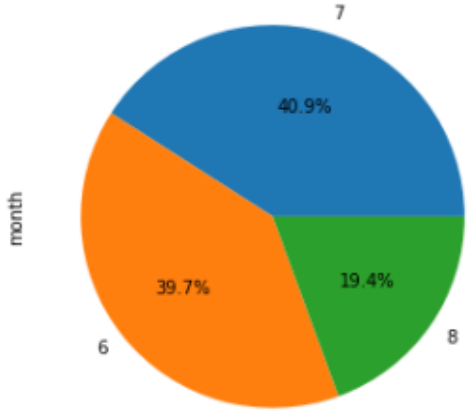

99.6%

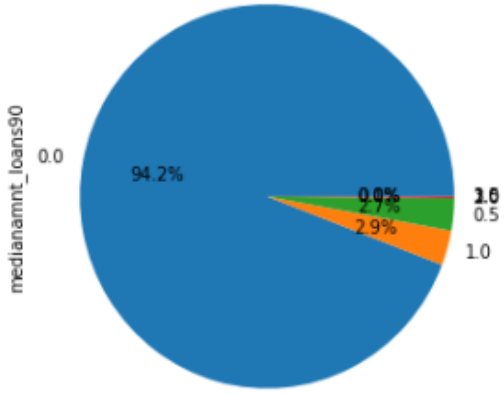

0.0
93.2%
3.4%
0.5
1.0

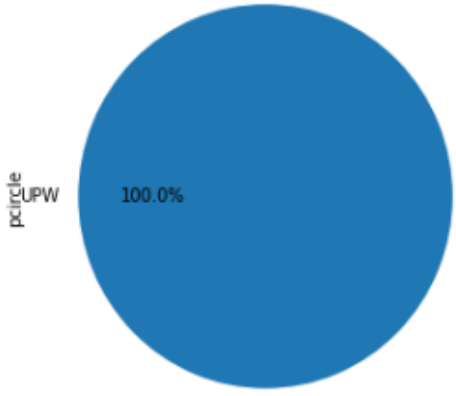

95.8%
2.5%
Zero times
10-20 times


96.7%
2.2%
100-200 times

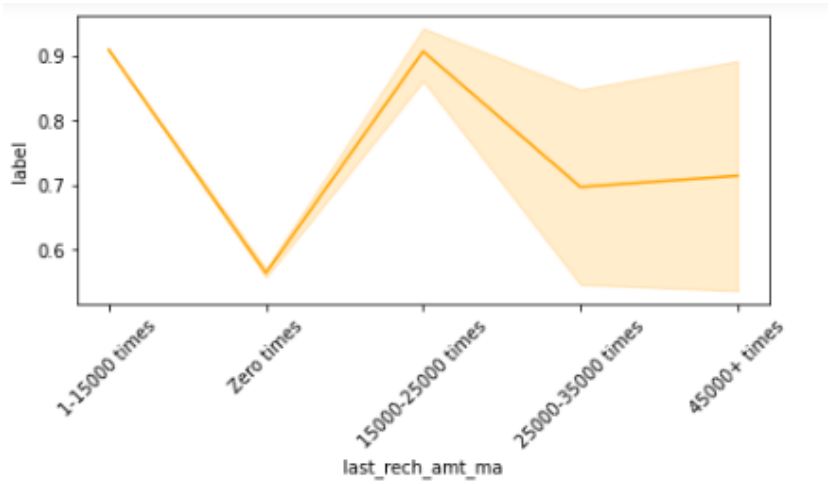

maxamnt_loans90
6
86.3%
1.0%
0
12.7%
12

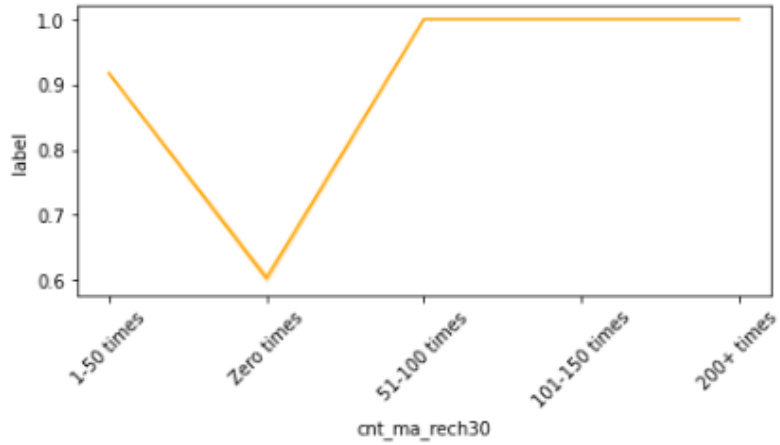

month
7
40.9%
39.7%
19.4%
6
8

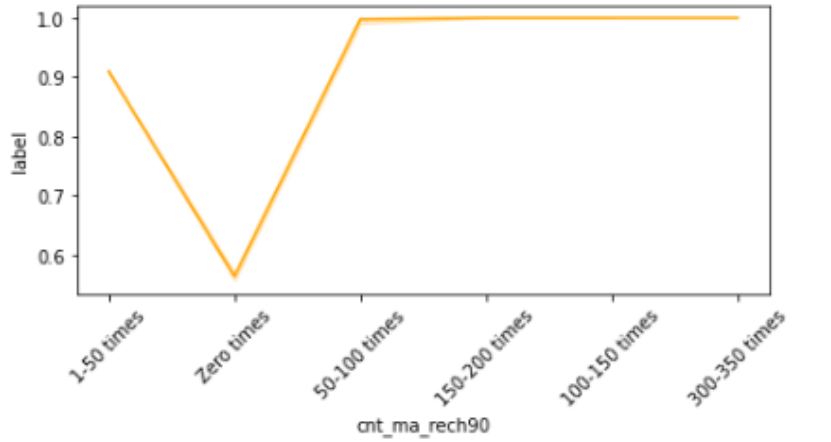

medianamnt_loans90
0.0
94.2%
2.7%
0.5
2.9%
1.0

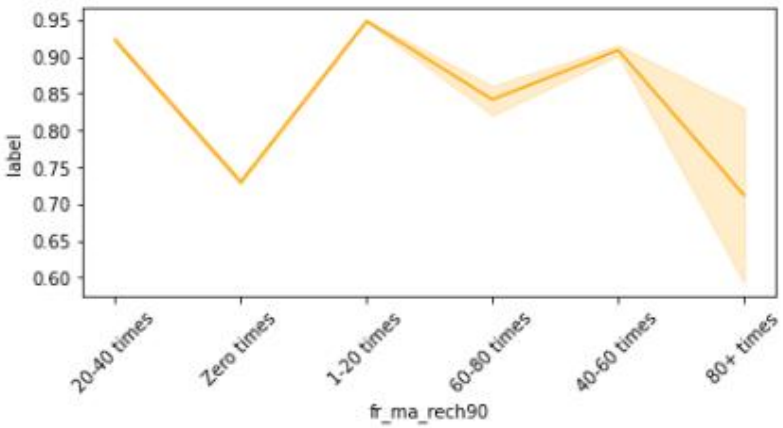

pcircle
UPW
100.0%

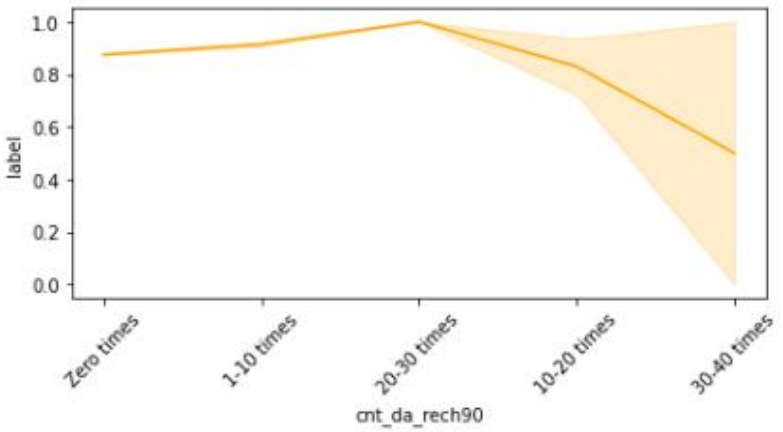
Line Chart

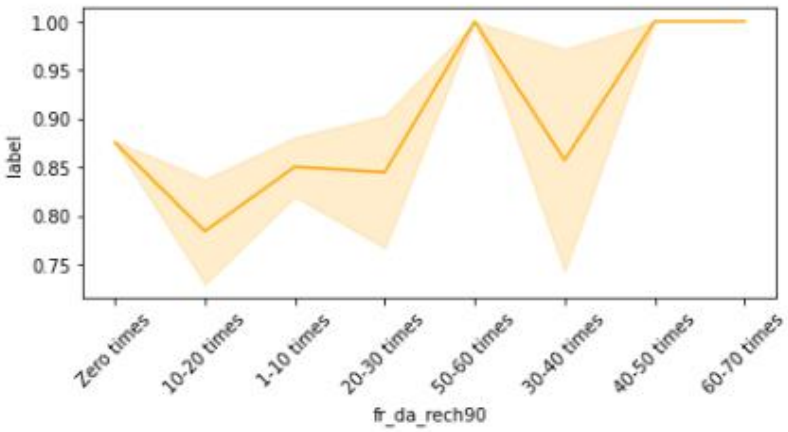

label
0.9
0.8
0.7
0.6
1-15000 times
Zero times
15000-25000 times
25000-35000 times
45000+ times
last_rech_amt_ma

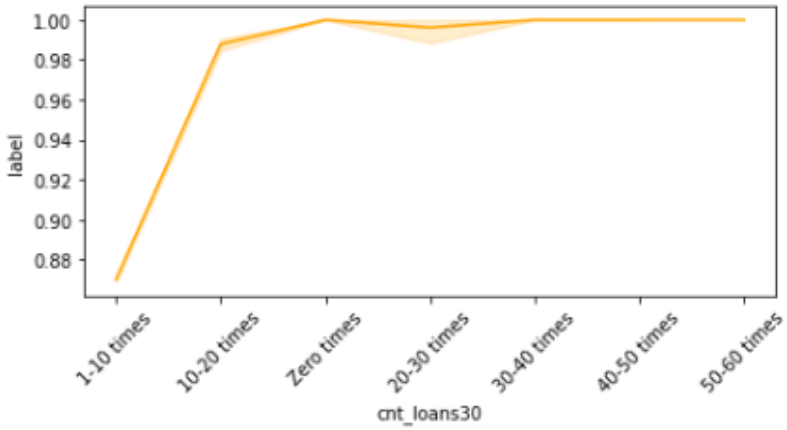

label
1.0
0.9
0.8
0.7
0.6
1-50 times
Zero times
51-100 times
101-150 times
200+ times
cnt_ma_rech30

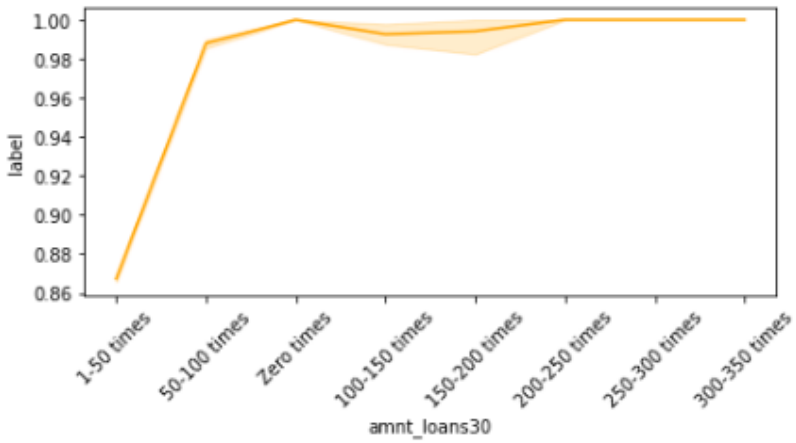

label
1.0
0.9
0.8
0.7
0.6
1-50 times
Zero times
50-100 times
150-200 times
100-150 times
300-350 times
cnt_ma_rech90

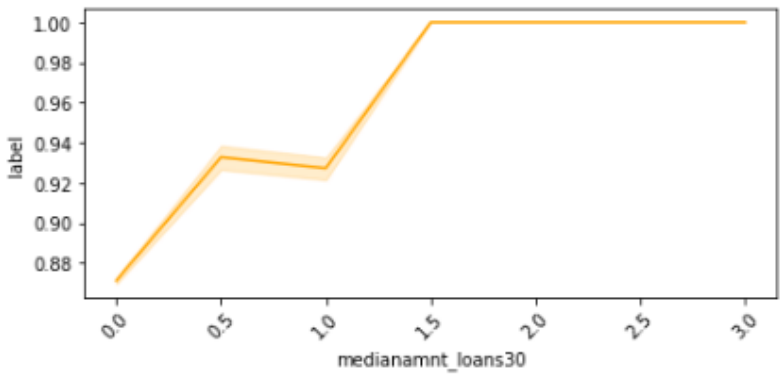

0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
label
20-40 times
Zero times
1-20 times
60-80 times
40-60 times
80+ times
fr_ma_rech90

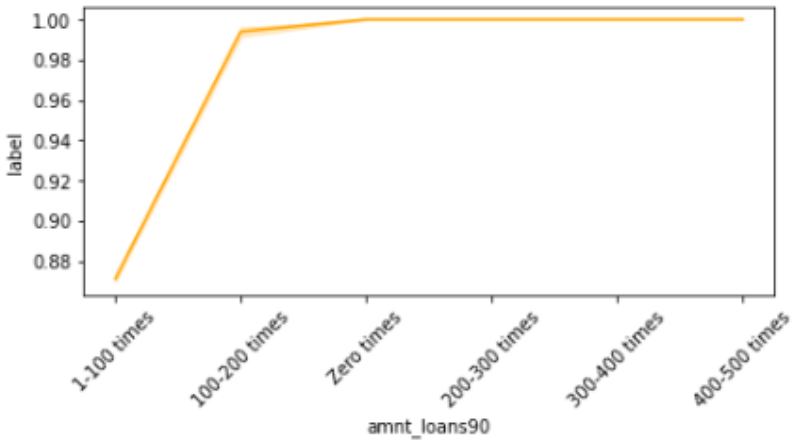

1.0
0.8
0.6
0.4
0.2
0.0
label
Zero times
1-10 times
20-30 times
10-20 times
30-40 times
cnt_da_rech90

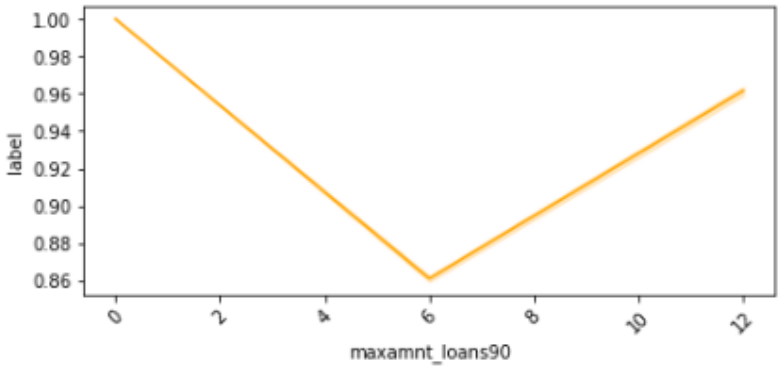

1.00
0.95
0.90
0.85
0.80
0.75
label
Zero times
10-20 times
1-10 times
20-30 times
50-60 times
30-40 times
40-50 times
60-70 times
fr_da_rech90

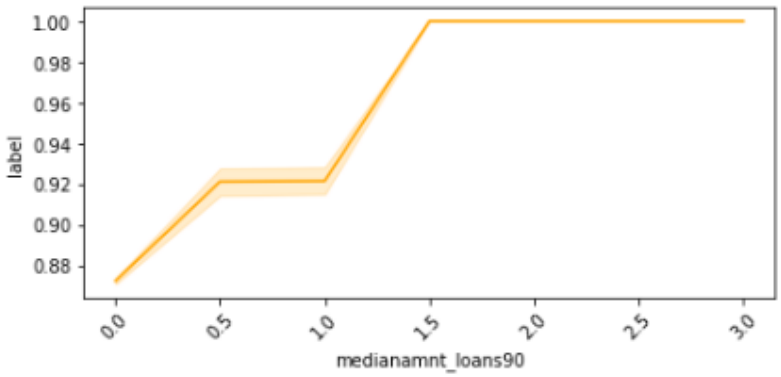

label
1.00
0.98
0.96
0.94
0.92
0.90
0.88
1-10 times
10-20 times
Zero times
20-30 times
30-40 times
40-50 times
50-60 times
cnt_loans30

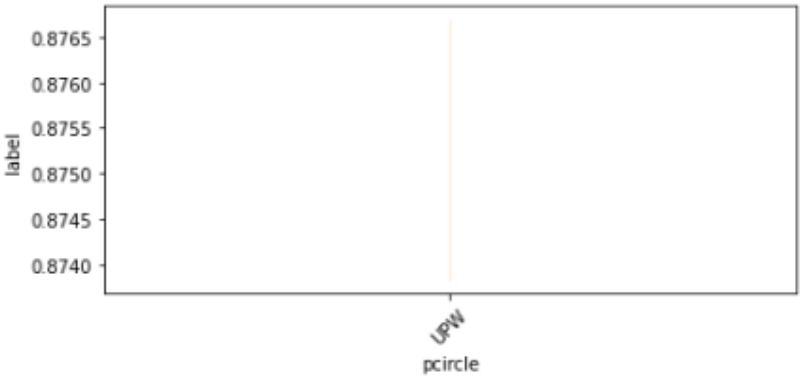

label
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
1-50 times
50-100 times
Zero times
100-150 times
150-200 times
200-250 times
250-300 times
300-350 times
amnt_loans30

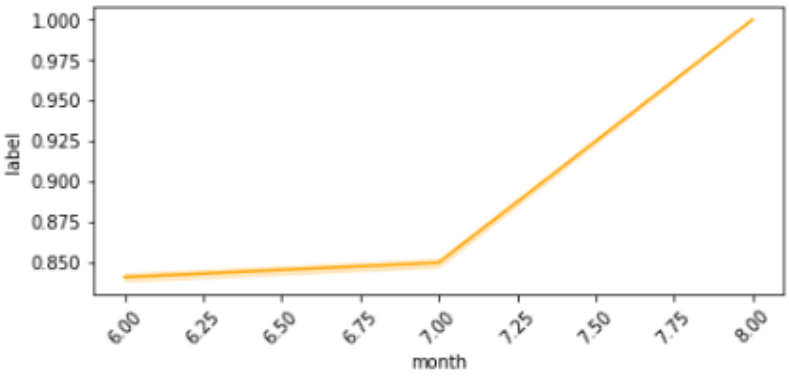

label
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.0
0.5
1.0
1.5
2.0
2.5
3.0
medianamnt_loans30


1.00
0.98
0.96
0.94
0.92
0.90
0.88
label
1-100 times
100-200 times
Zero times
200-300 times
300-400 times
400-500 times
amnt_loans90


1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
label
0
2
4
6
8
10
12
maxamnt_loans90


1.00
0.98
0.96
0.94
0.92
0.90
0.88
label
0.0
0.5
1.0
1.5
2.0
2.5
3.0
medianamnt_loans90

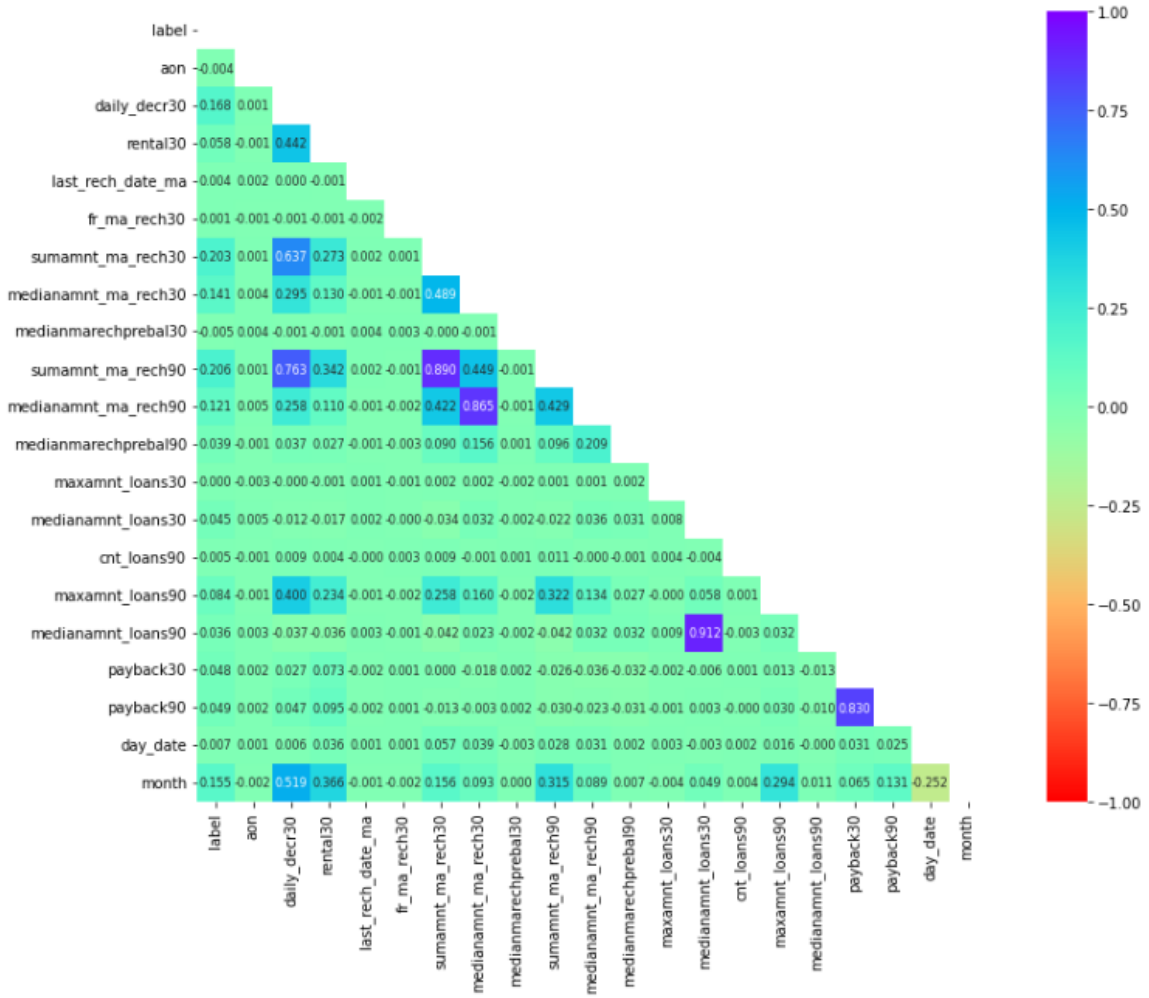




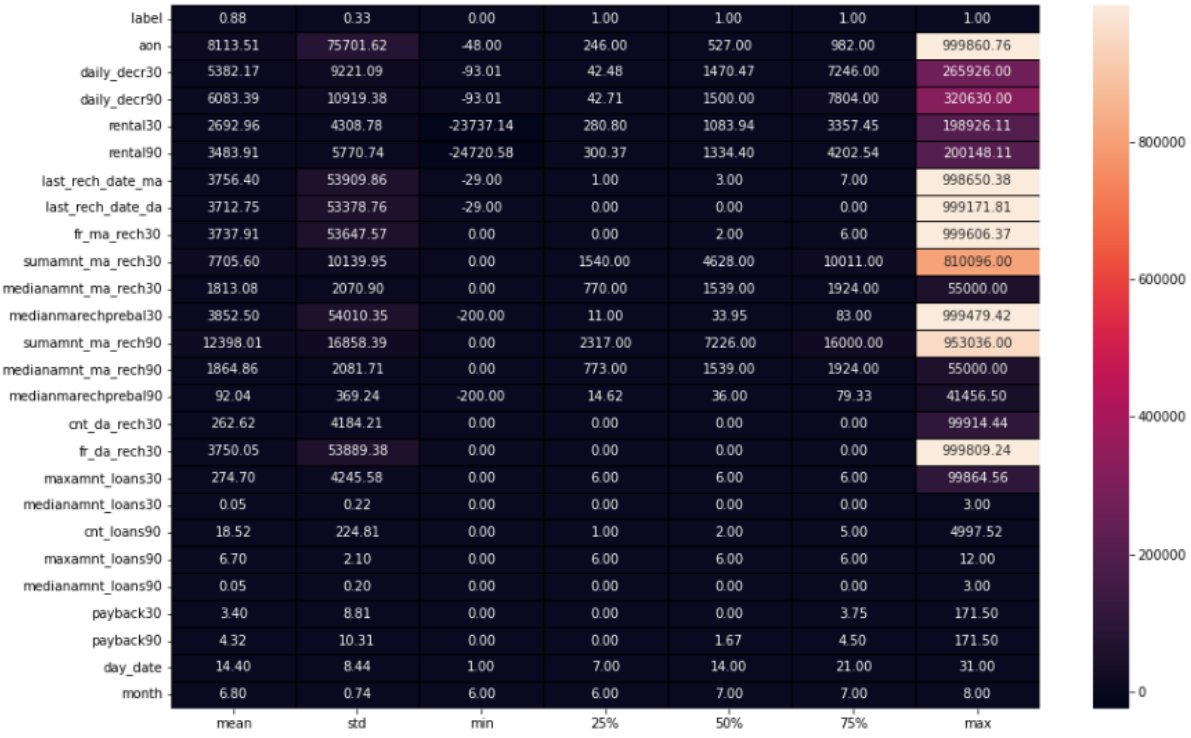
Observations: -

- **In last_rech_amt_ma having more counts in 1-15000 times while very less in 45000+ cases.**
- **Most of the cases are of 1-50 times in cnt_ma_rech30.**
- **1-20 times cases are more from rest of cases in fr_ma_rech90.**
- **In cnt_da_rech90 and fr_da_rech most of the cases are of 0 times.**
- **cnt_loans are more in 1-10 times as per dataset.**
- **Most of the customer takes 1-50 times amnt_loans30 in 30 days.**
- **Most of the customer medianamnt_loans30 are zero as per dataset..**
- **In 90 days most of the customers takes 1-100 times loans.**
- **maxamnt_loans90 are 6 lakhs in 90 days.**
- **medianamnt_loans90 are having zero cases more.**
- **Months and pcircle are show not miserable changes as per dataset.**

Correlation of the Dataset



Describe of the Dataset



Skewness

We have remove skewness through power transformer

Outliers

We have applied Z score and Interquartile method for outlier removal but both shows very high amount of data loss upto 50 percent hence we can't consider it.

Chi2

We applied chi2 test for taking import variable for model building.

Feature selection

Checking Mutlicollinearity

```
import statsmodels.api as sm
from scipy import stats
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
def calc_vif(x):

    vif = pd.DataFrame()
    vif['Variance'] = x.columns
    vif["VIF Factor"] = [variance_inflation_factor(x.values, i) for i in range(x.shape[1])]
    return vif
```

	Variance	VIF Factor
0	aon	1.009571
1	daily_decr30	3.664129
2	rental30	1.711770
3	last_rech_date_ma	1.003976
4	fr_ma_rech30	1.003961
5	sumamnt_ma_rech30	8.448510
6	medianamnt_ma_rech30	7.773267
7	medianmarechprebal30	1.004025
8	sumamnt_ma_rech90	11.467420
9	medianamnt_ma_rech90	7.552948
10	medianmarechprebal90	1.115317
11	maxamnt_loans30	1.003637
12	medianamnt_loans30	1.055908
13	cnt_loans90	1.005900
14	payback30	3.715456
15	payback90	3.816335
16	day_date	1.874745

Removing Skewness

```
x[num_col] = pw.fit_transform(x[num_col])
```

```
x[num_col].skew()   # done in main dataframe
```

```
aon                    1.654672
daily_decr30          -6.578784
rental30              -1.020249
last_rech_date_ma     -5.361687
fr_ma_rech30           0.164892
sumamnt_ma_rech30     -0.290988
medianamnt_ma_rech30  -0.189169
medianmarechprebal30  -0.118247
sumamnt_ma_rech90     -0.191753
medianamnt_ma_rech90  -0.044012
medianmarechprebal90   7.507637
maxamnt_loans30       -1.679963
medianamnt_loans30     3.447419
cnt_loans90            0.105503
payback30              0.298419
payback90              0.210764
day_date              -0.156104
dtype: float64
```

Standard Scalling

Standard Scaler

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x[num_col] = scaler.fit_transform(x[num_col])
```

```
x.head()
```

	aon	daily_decr30	rental30	last_rech_date_ma	last_rech_amt_ma	cnt_ma_rech30	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_rech30	media
0	-0.177142	0.331978	-0.580876	-0.025529	0.0	0.0	1.603221	-0.178281	0.208319	
1	0.036408	1.115924	0.339622	0.191337	0.0	0.0	-1.129108	0.219532	1.632552	
2	-0.034699	-0.007711	-0.367064	0.000026	0.0	0.0	-1.129108	-0.535808	0.208319	
3	-0.199242	-1.031704	-0.603147	0.299322	0.0	4.0	-1.129108	-1.902672	-1.962288	
4	0.114830	-0.682563	-0.310574	0.021212	0.0	0.0	0.188668	1.262697	0.574988	

Model Building and Results

1. We have find first random state.
2. We divided data into 3 parts train, cv and test for better model prediction.
3. We use 9 models and find results as below.

```
# ********************** LogisticRegression() ***************************
# cv_accuracy 0.81452492211838
# test_accuracy 0.8140253320976213

# ********************** SGDClassifier() ***************************
# cv_accuracy 0.8089693665628245
# test_accuracy 0.808446455505279

# ********************** GaussianNB() ***************************
# cv_accuracy 0.6339823468328141
# test_accuracy 0.6329389958022134

# ********************** RandomForestClassifier() ***************************
# cv_accuracy 0.9398753894080997
# test_accuracy 0.9401315669919497

# ********************** AdaBoostClassifier() ***************************
# cv_accuracy 0.8407969885773624
# test_accuracy 0.843028221483218

# ********************** GradientBoostingClassifier() ***************************
# cv_accuracy 0.8903426791277259
# test_accuracy 0.8914118010503552

# ********************** BaggingClassifier() ***************************
# cv_accuracy 0.9269210799584632
# test_accuracy 0.9259844809100656

# ********************** DecisionTreeClassifier() ***************************
# cv_accuracy 0.8949896157840083
# test_accuracy 0.8950008177506406

# ********************** XGBClassifier() ***************************
# cv_accuracy 0.9474039460020769
# test_accuracy 0.9472459975649203
```

From above all model we find that XGBClassifier shows similar CV and testing accuracy result with least difference hence we are going to consider it for model building.

Final Model XGBClassifier

```
params = {'base_score': [0.5, 1, 1.5], 'booster' : ['gbtree','gblinear ', 'dart', None], 'colsample_bylevel': [1,2,3],
          'n_jobs': [1, 2, 3]}
```

```
gcv = GridSearchCV(estimator = XGBClassifier(), param_grid = params)
gcv.fit(train, train_output)
gcv.best_params_

#model gives better prediction with params hence we dont consider it.
```

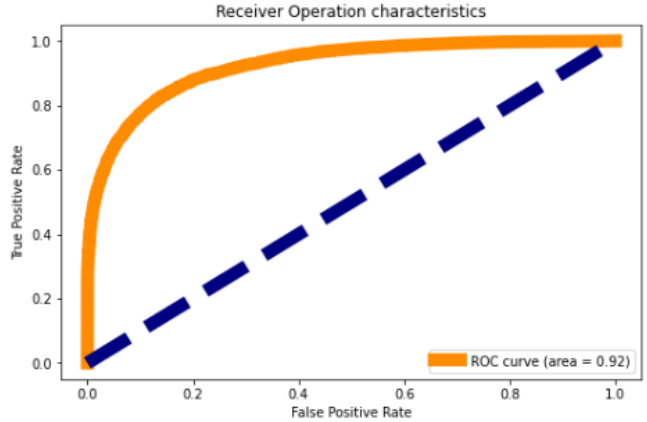
```
final_model = XGBClassifier()
final_model.fit(train, train_output)
model_pred_test = final_model.predict(test)
test_acc = accuracy_score(model_pred_test, test_output)
print('test_accuracy', test_acc, '\n')
print('Classification Report: \n', classification_report(model_pred_test, test_output) )
print('Confusion Matrix: \n', confusion_matrix(model_pred_test, test_output) )
print("\n")
```

```
test_accuracy 0.9472459975649203

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.95      0.95     54557
           1       0.95      0.94      0.95     55501

    accuracy                           0.95    110058
   macro avg       0.95      0.95      0.95    110058
weighted avg       0.95      0.95      0.95    110058

Confusion Matrix:
 [[51890  2667]
 [ 3139 52362]]
```

Using Predict Proba function for finding accuracy (i.e Getting approved loan)

```
final_pred_prob = model.predict_proba(test)[:, 1]
```

```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresolds = roc_curve(test_output, final_pred_prob)
roc_auc = auc(fpr, tpr)

plt.figure(figsize = (8,5))
plt.plot(fpr, tpr, color = 'darkorange', lw = 10, label = "ROC curve (area = %0.2f)" % roc_auc)
plt.plot([0,1],[0,1], color = 'navy', lw = 10, linestyle = '--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Receiver Operation characteristics")
plt.legend(loc = "lower right")
plt.show()
```



Model Deployment

Deploy Model

```
import pickle

filename = "Microcredit.pkl"
pickle.dump(final_model, open(filename, 'wb'))
```

Loading Model

```
load = pickle.load(open('Microcredit.pkl', 'rb'))
result = load.score(test, test_output)
print(result)
```

```
0.9472459975649203
```

```
conclusion = pd.DataFrame()
conclusion['Predicted Microcredit label'] = np.array(final_model.predict(test))
conclusion['Actual Microcredit label'] = np.array(test_output)
```

```
conclusion.sample(10)
```

	Predicted Microcredit label	Actual Microcredit label
3971	1	1
92599	1	1
51038	1	1
67833	1	1
71170	0	0
50984	0	0
44452	1	1
47929	1	1
31616	1	1
79104	0	0

➢ Hardware and Software Requirements and Tools Used

Operating System: Window 11

RAM: 8 GB

Processor: i5 10th Generation

Software: Jupyter Notebook

Python Libraries: Mainly

Pandas: This library used for dataframe operations .

Numpy: This library gives statistical computation for smooth functioning .

Matplotlib: Used for visualization.

Seaborn: This library is also used for visualization.

Sklearn: This library having so many machine learning module and we can import them from this library.

Pickle: This is used for deploying the model.

Xgboost: Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library

Lightgbm: Light version of Gradient Boosting Machine.

CONCLUSION

Key Findings and Conclusions of the Study

This project has built a model that can predict upcoming customer goodwill for returning loan. Due to this company can reduces loses in adding new customer. The challenge to find defaulter with many number of features dataset in machine learning.