A

Report On

Micro Credit Loan

Use case

For

Telecom Industry

Submitted By:

Rohit Kachhal

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a beautiful dataset and it has helped me to improve my analyzation skills.

A huge thanks to my academic team “Data trained” who are the reason behind what I am doing today.

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 industry 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.
  - ❖ This use case is for 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.

- Conceptual Background of the Domain Problem
  - ❖ Telecom industries 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).

- Motivation for the Problem Undertaken

  Based on data provided, customer's repayment of loan is assessed based on different factors. By building the model, we can access those customers are highly likely to repay the loan, thereby it will be useful for those needy people who will repay the loan and also prevent the loss to the customer by avoiding loans to the defaulters.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  In this given dataset, the target variable is ‘label’ which contains two type of value: ‘1’ indicates that the loan has been payed i.e, Non-defaulter while ‘0’ indicates that the loan has not been payed i.e. defaulter. So, this problem is binary classification problem. This classification problem looks like imbalanced. For this problem, I have worked on 8 different classification model of machine learning. This dataset does not contain any null value. It is also observed that some features contain above 92% values as 0, some features contain negative value which is not possible in real world, some are useless columns for model, from one column have to extract data into several column. I used different visualization plot to see the trend of data like distplot to see the distribution and skewness in dataset, boxplot to see the outliers, barplot to see the relationship between target and features, heatmap to see the multicollinearity, ROC AUC curve to see the performance of models. From these plots, I can observe skewness, outliers which I treated accordingly. I have also done hyperparameter tuning on finalize model to improve performance of model but not always hyperparameter tuning improve the performance of model. After that I saved the model and compare the predictions.

- Data Sources and their formats

  ❖ **DataFrame**

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	last_rech_amt_ma
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
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

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

Outcome:

- ✓ This dataframe contains 209593 rows and 37 columns.
- ✓ There are no null values in the dataset.
- ✓ As data is important so, loss of data above 7-8% is not allowed.

❖ More about dataset

Data Frame Info

```
Data columns (total 37 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   Unnamed: 0            209593 non-null  int64
 1   label                 209593 non-null  int64
 2   msisdn                209593 non-null  object
 3   aon                   209593 non-null  float64
 4   daily_decr30          209593 non-null  float64
 5   daily_decr90          209593 non-null  float64
 6   rental30              209593 non-null  float64
 7   rental90              209593 non-null  float64
 8   last_rech_date_ma     209593 non-null  float64
 9   last_rech_date_da     209593 non-null  float64
 10  last_rech_amt_ma      209593 non-null  int64
 11  cnt_ma_rech30         209593 non-null  int64
 12  fr_ma_rech30          209593 non-null  float64
 13  sumamnt_ma_rech30     209593 non-null  float64
 14  medianamnt_ma_rech30  209593 non-null  float64
 15  medianmarechprebal30  209593 non-null  float64
 16  cnt_ma_rech90         209593 non-null  int64
 17  fr_ma_rech90          209593 non-null  int64
 18  sumamnt_ma_rech90     209593 non-null  int64
 19  medianamnt_ma_rech90  209593 non-null  float64
 20  medianmarechprebal90  209593 non-null  float64
 21  cnt_da_rech30         209593 non-null  float64
 22  fr_da_rech30          209593 non-null  float64
 23  cnt_da_rech90         209593 non-null  int64
 24  fr_da_rech90          209593 non-null  int64
 25  cnt_loans30           209593 non-null  int64
 26  amnt_loans30          209593 non-null  int64
 27  maxamnt_loans30       209593 non-null  float64
 28  medianamnt_loans30    209593 non-null  float64
 29  cnt_loans90           209593 non-null  float64
 30  amnt_loans90          209593 non-null  int64
 31  maxamnt_loans90       209593 non-null  int64
 32  medianamnt_loans90    209593 non-null  float64
 33  payback30             209593 non-null  float64
 34  payback90             209593 non-null  float64
 35  pcircle               209593 non-null  object
 36  pdate                 209593 non-null  object
dtypes: float64(21), int64(13), object(3)
```

Unique Values

```
Unnamed: 0              209593
label                        2
msisdn                  186243
aon                       4507
daily_decr30            147025
daily_decr90            158669
rental30                132148
rental90                141033
last_rech_date_ma         1186
last_rech_date_da         1174
last_rech_amt_ma            70
cnt_ma_rech30               71
fr_ma_rech30              1083
sumamnt_ma_rech30        15141
medianamnt_ma_rech30       510
medianmarechprebal30     30428
cnt_ma_rech90              110
fr_ma_rech90                89
sumamnt_ma_rech90        31771
medianamnt_ma_rech90       608
medianmarechprebal90     29785
cnt_da_rech30             1066
fr_da_rech30              1072
cnt_da_rech90               27
fr_da_rech90                46
cnt_loans30                 40
amnt_loans30                48
maxamnt_loans30           1050
medianamnt_loans30           6
cnt_loans90               1110
amnt_loans90                69
maxamnt_loans90              3
medianamnt_loans90           6
payback30                 1363
payback90                 2381
pcircle                      1
pdate                       82
dtype: int64
```

❖ Descriptive Statistical Analysis

	label	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_amt_ma	cnt_ma_rech30
count	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000
mean	0.875177	8112.343445	5381.402289	6082.515068	2692.581910	3483.406534	3755.847800	2064.452797	3.978057
std	0.330519	75696.082531	9220.623400	10918.812767	4308.586781	5770.461279	53905.892230	2370.786034	4.256090
min	0.000000	-48.000000	-93.012667	-93.012667	-23737.140000	-24720.580000	-29.000000	0.000000	0.000000
25%	1.000000	246.000000	42.440000	42.692000	280.420000	300.260000	1.000000	770.000000	1.000000
50%	1.000000	527.000000	1469.175667	1500.000000	1083.570000	1334.000000	3.000000	1539.000000	3.000000
75%	1.000000	982.000000	7244.000000	7802.790000	3356.940000	4201.790000	7.000000	2309.000000	5.000000
max	1.000000	999860.755168	265926.000000	320630.000000	198926.110000	200148.110000	998650.377733	55000.000000	203.000000

	fr_ma_rech30	sumamnt_ma_rech30	medianamnt_ma_rech30	medianmarechprebal30	cnt_ma_rech90	fr_ma_rech90	sumamnt_ma_rech90
count	209593.000000	209593.000000	209593.000000	209593.000000	209593.00000	209593.000000	209593.000000
mean	3737.355121	7704.501157	1812.817952	3851.927942	6.31543	7.716780	12396.218352
std	53643.625172	10139.621714	2070.864620	54006.374433	7.19347	12.590251	16857.793882
min	0.000000	0.000000	0.000000	-200.000000	0.00000	0.000000	0.000000
25%	0.000000	1540.000000	770.000000	11.000000	2.00000	0.000000	2317.000000
50%	2.000000	4628.000000	1539.000000	33.900000	4.00000	2.000000	7226.000000
75%	6.000000	10010.000000	1924.000000	83.000000	8.00000	8.000000	16000.000000
max	999606.368132	810096.000000	55000.000000	999479.419319	336.00000	88.000000	953036.000000

	medianamnt_ma_rech90	medianmarechprebal90	cnt_loans30	amnt_loans30	maxamnt_loans30	cnt_loans90	amnt_loans90	maxamnt_loans90
count	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000	209593.000000
mean	1864.595821	92.025541	2.758981	17.952021	274.658747	18.520919	23.645398	6.703134
std	2081.680664	369.215658	2.554502	17.379741	4245.264648	224.797423	26.469861	2.103864
min	0.000000	-200.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	773.000000	14.600000	1.000000	6.000000	6.000000	1.000000	6.000000	6.000000
50%	1539.000000	36.000000	2.000000	12.000000	6.000000	2.000000	12.000000	6.000000
75%	1924.000000	79.310000	4.000000	24.000000	6.000000	5.000000	30.000000	6.000000
max	55000.000000	41456.500000	50.000000	306.000000	99864.560864	4997.517944	438.000000	12.000000

	payback30	payback90
count	209593.000000	209593.000000
mean	3.398826	4.321485
std	8.813729	10.308108
min	0.000000	0.000000
25%	0.000000	0.000000
50%	0.000000	1.666667
75%	3.750000	4.500000
max	171.500000	171.500000

- Outcome:
  - ✓ Analysis clearly describes the misleading values, Outliers and skewness present in the dataset.

- Data Preprocessing Done
  - ❖ Some features like age, amount, days contain negative values which is not possible in real world, need to treat them.
  - ❖ ‘pdate’ contains values in yyyy-mm-dd so, we need to extract day, month and year.
  - ❖ 'maxamnt_loans30' should be 0,6 and 12. Except these three values, we need to replace rest value by 0.
  - ❖ Few features contain above 92% value as 0. so, drop these features.
  - ❖ Drop unwanted columns.
  - ❖ Dealing with outliers using percentile method.
  - ❖ Dealing with skewness using ‘yeo-johnson’ method of Power Transformer.

- Data Inputs- Logic- Output Relationships
  - ❖ I used barplot to visualize the impact of feature on target.
  - ❖ All data now are of either int or float.
  - ❖ I observed, in mostly features, customers are Non-Defaulters.

- Defaulters are most in case of 'aon', 'medianmarechprebal30'.

- Hardware and Software Requirements and Tools Used
  - Hardware:
    - Processor: I3 or above
    - RAM: 4 GB or above
    - Storage: 250 GB or above
  - Software:
    - Anaconda
    - Jupyter notebook etc
  - Libraries:
    - numpy
    - pandas
    - matplotlib
    - seaborn
    - model_selection: train_test_split, cross_val_score, GridSearchCV
    - datetime
    - preprocessing : PowerTransformer, StandardScaler,
    - imblearn: over_sampling using SMOTE
    - metrics: accuracy_score, classification_report, roc_curve, plot_roc_curve, confusion_matrix

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  First of all, I cleaned the data like treating the values by taking **abs()** value for those which contains negative value that is not possible in real world domain. After that, extract the value into separate columns from 'pdate' using **datetime** libraries, then replace the value in 'maxamnt_loans30' to 0 if these values are not 0,6 & 1 by using **replace().**Then, drop those column that contains above 92% value as 0 and those columns which are not useful for model by using **drop().** Finally dealing with outliers using **percentile method**, skewness using **'yeo-johnson' method of Power Transformer** and imbalancing using **over_sampling of SMOTE algorithm**.

- Testing of Identified Approaches (Algorithms)

  As the problem is binary classification problem. In machine learning there are several algorithms for classification problem. I used total 8 algorithm to build best model on this dataset. The algorithms are:

  - Logistic Regression
  - K-Neighbors Classifier
  - Decision Tree Classifier
  - Random Forest Classifier
  - Bagging Classifier
  - Gradient Boosting Classifier
  - AdaBoost Classifier
  - XGB Classifier

- Run and Evaluate selected models

  Here are the coding and insights of all models:

❖ Logistic Regression

```
lg =LogisticRegression()
lg.fit(X_train,y_train)
lg_pred = lg.predict(X_test)
lg_accuracy = accuracy_score(y_test,lg_pred)
lg_cf = classification_report(y_test,lg_pred)
lg_cm = confusion_matrix(y_test,lg_pred)
lg_train_score = lg.score(X_train,y_train)
lg_test_score = lg.score(X_test,y_test)

print('Logistic Regression')
print('------------------------------------------------------------------\n')
print('The Score on train set is    :',lg_train_score)
print('The Score on test set is     :',lg_test_score)
print('The Accuracy on test set is :',lg_accuracy)
print('The Classification report is :\n',lg_cf)
print('The Confusion matrix is :\n',lg_cm)
print('\n------------------------------------------------------------------')
```

```
Logistic Regression
------------------------------------------------------------------

The Score on train set is    : 0.7701000377721444
The Score on test set is     : 0.7733760982745618
The Accuracy on test set is : 0.7733760982745618
The Classification report is :
               precision    recall  f1-score   support

           0       0.77      0.79      0.78     55154
           1       0.78      0.76      0.77     54905

    accuracy                           0.77    110059
   macro avg       0.77      0.77      0.77    110059
weighted avg       0.77      0.77      0.77    110059

The Confusion matrix is :
 [[43586 11568]
 [13374 41531]]

------------------------------------------------------------------
```

```
lg_cv_score = cross_val_score(lg,X_train,y_train,cv=5)
print('The cross validation score is :',lg_cv_score.mean())
```

```
The cross validation score is : 0.7703920883367813
```

❖ K-Neighbors Classifier

```
kc = KNeighborsClassifier()
kc.fit(X_train,y_train)
kc_pred = kc.predict(X_test)
kc_accuracy = accuracy_score(y_test,kc_pred)
kc_cf = classification_report(y_test,kc_pred)
kc_cm = confusion_matrix(y_test,kc_pred)
kc_train_score = kc.score(X_train,y_train)
kc_test_score = kc.score(X_test,y_test)

print('K-Neighbors Classifier')
print('------------------------------------------------------------------\n')
print('The Score on train set is    :',kc_train_score)
print('The Score on test set is     :',kc_test_score)
print('The Accuracy on test set is :',kc_accuracy)
print('The Classification report is :\n',kc_cf)
print('The Confusion matrix is :\n',kc_cm)
print('\n------------------------------------------------------------------')
```

```
K-Neighbors Classifier
------------------------------------------------------------------

The Score on train set is    : 0.9251099091521516
The Score on test set is     : 0.8980274216556574
The Accuracy on test set is : 0.8980274216556574
The Classification report is :
               precision    recall  f1-score   support

           0       0.84      0.99      0.91     55154
           1       0.99      0.80      0.89     54905

    accuracy                           0.90    110059
   macro avg       0.91      0.90      0.90    110059
weighted avg       0.91      0.90      0.90    110059

The Confusion matrix is :
 [[54640   514]
 [10709 44196]]

------------------------------------------------------------------
```

```
kc_cv_score = cross_val_score(kc,X_train,y_train,cv=5)
print('The cross validation score is :',kc_cv_score.mean())
```

```
The cross validation score is : 0.8896352451382405
```

❖ Decision Tree Classifier

```
dt = DecisionTreeClassifier()
dt.fit(X_train,y_train)
dt_pred = dt.predict(X_test)
dt_accuracy = accuracy_score(y_test,dt_pred)
dt_cf = classification_report(y_test,dt_pred)
dt_cm = confusion_matrix(y_test,dt_pred)
dt_train_score = dt.score(X_train,y_train)
dt_test_score = dt.score(X_test,y_test)

print('Decision Tree Classifier')
print('-------------------------------------------------------------------\n')
print('The Score on train set is    :',dt_train_score)
print('The Score on test set is     :',dt_test_score)
print('The Accuracy on test set is :',dt_accuracy)
print('The Classification report is :\n',dt_cf)
print('The Confusion matrix is :\n',dt_cm)
print('\n-------------------------------------------------------------------')
```

```
Decision Tree Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9999922119289884
The Score on test set is     : 0.9094758266020952
The Accuracy on test set is : 0.9094758266020952
The Classification report is :
               precision    recall  f1-score   support

           0       0.90      0.92      0.91     55154
           1       0.92      0.90      0.91     54905

    accuracy                           0.91    110059
   macro avg       0.91      0.91      0.91    110059
weighted avg       0.91      0.91      0.91    110059

The Confusion matrix is :
 [[50575  4579]
 [ 5384 49521]]

-------------------------------------------------------------------
```

```
dt_cv_score = cross_val_score(dt,X_train,y_train,cv=5)
print('The cross validation score is :',dt_cv_score.mean())
```

```
The cross validation score is : 0.9042807020321186
```

❖ Random Forest Classifier

```
rc = RandomForestClassifier()
rc.fit(X_train,y_train)
rc_pred = rc.predict(X_test)
rc_accuracy = accuracy_score(y_test,rc_pred)
rc_cf = classification_report(y_test,rc_pred)
rc_cm = confusion_matrix(y_test,rc_pred)
rc_train_score = rc.score(X_train,y_train)
rc_test_score = rc.score(X_test,y_test)

print('Random Forest Classifier')
print('-------------------------------------------------------------------\n')
print('The Score on train set is    :',rc_train_score)
print('The Score on test set is     :',rc_test_score)
print('The Accuracy on test set is :',rc_accuracy)
print('The Classification report is :\n',rc_cf)
print('The Confusion matrix is :\n',rc_cm)
print('\n-------------------------------------------------------------------')
```

```
Random Forest Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9999883178934825
The Score on test set is     : 0.9528798190061695
The Accuracy on test set is : 0.9528798190061695
The Classification report is :
               precision    recall  f1-score   support

           0       0.95      0.96      0.95     55154
           1       0.96      0.95      0.95     54905

    accuracy                           0.95    110059
   macro avg       0.95      0.95      0.95    110059
weighted avg       0.95      0.95      0.95    110059

The Confusion matrix is :
 [[52852  2302]
 [ 2884 52021]]

-------------------------------------------------------------------
```

```
rc_cv_score = cross_val_score(rc,X_train,y_train,cv=5)
print('The cross validation score is :',rc_cv_score.mean())
```

```
The cross validation score is : 0.9489024653905126
```

❖ Bagging Classifier

```
bc = BaggingClassifier()
bc.fit(X_train,y_train)
bc_pred = bc.predict(X_test)
bc_accuracy = accuracy_score(y_test,bc_pred)
bc_cf = classification_report(y_test,bc_pred)
bc_cm = confusion_matrix(y_test,bc_pred)
bc_train_score = bc.score(X_train,y_train)
bc_test_score = bc.score(X_test,y_test)

print('Bagging Classifier')
print('-----------------------------------------------------------------\n')
print('The Score on train set is    :',bc_train_score)
print('The Score on test set is     :',bc_test_score)
print('The Accuracy on test set is :',bc_accuracy)
print('The Classification report is :\n',bc_cf)
print('The Confusion matrix is :\n',bc_cm)
print('\n-----------------------------------------------------------------')
```

```
Bagging Classifier
-----------------------------------------------------------------

The Score on train set is    : 0.9961410108137366
The Score on test set is     : 0.9387237754295423
The Accuracy on test set is : 0.9387237754295423
The Classification report is :
               precision    recall  f1-score   support

           0       0.93      0.95      0.94     55154
           1       0.95      0.92      0.94     54905

    accuracy                           0.94    110059
   macro avg       0.94      0.94      0.94    110059
weighted avg       0.94      0.94      0.94    110059

The Confusion matrix is :
 [[52647  2507]
 [ 4237 50668]]

-----------------------------------------------------------------
```

```
bc_cv_score = cross_val_score(bc,X_train,y_train,cv=5)
print('The cross validation score is :',bc_cv_score.mean())
```

```
The cross validation score is : 0.9354952926663328
```

❖ Gradient Boosting Classifier

```
gc = GradientBoostingClassifier()
gc.fit(X_train,y_train)
gc_pred = gc.predict(X_test)
gc_accuracy = accuracy_score(y_test,gc_pred)
gc_cf = classification_report(y_test,gc_pred)
gc_cm = confusion_matrix(y_test,gc_pred)
gc_train_score = gc.score(X_train,y_train)
gc_test_score = gc.score(X_test,y_test)

print('Gradient Boosting Classifier')
print('-----------------------------------------------------------------\n')
print('The Score on train set is    :',gc_train_score)
print('The Score on test set is     :',gc_test_score)
print('The Accuracy on test set is :',gc_accuracy)
print('The Classification report is :\n',gc_cf)
print('The Confusion matrix is :\n',gc_cm)
print('\n-----------------------------------------------------------------')
```

```
Gradient Boosting Classifier
-----------------------------------------------------------------

The Score on train set is    : 0.8994988376304015
The Score on test set is     : 0.901007641356
The Accuracy on test set is : 0.901007641356
The Classification report is :
               precision    recall  f1-score   support

           0       0.89      0.91      0.90     55154
           1       0.91      0.89      0.90     54905

    accuracy                           0.90    110059
   macro avg       0.90      0.90      0.90    110059
weighted avg       0.90      0.90      0.90    110059

The Confusion matrix is :
 [[50465  4689]
 [ 6206 48699]]

-----------------------------------------------------------------
```

```
gc_cv_score = cross_val_score(gc,X_train,y_train,cv=5)
print('The cross validation score is :',gc_cv_score.mean())
```

```
The cross validation score is : 0.8996857461244488
```

❖ AdaBoost Classifier

```
ac = AdaBoostClassifier()
ac.fit(X_train,y_train)
ac_pred = ac.predict(X_test)
ac_accuracy = accuracy_score(y_test,ac_pred)
ac_cf = classification_report(y_test,ac_pred)
ac_cm = confusion_matrix(y_test,ac_pred)
ac_train_score = ac.score(X_train,y_train)
ac_test_score = ac.score(X_test,y_test)

print('AdaBoost Classifier')
print('-------------------------------------------------------------------\n')
print('The Score on train set is    :',ac_train_score)
print('The Score on test set is     :',ac_test_score)
print('The Accuracy on test set is :',ac_accuracy)
print('The Classification report is :\n',ac_cf)
print('The Confusion matrix is :\n',ac_cm)
print('\n-------------------------------------------------------------------')
```

```
AdaBoost Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.8484830784687095
The Score on test set is     : 0.8501530997010694
The Accuracy on test set is : 0.8501530997010694
The Classification report is :
               precision    recall  f1-score   support

           0       0.84      0.87      0.85     55154
           1       0.86      0.83      0.85     54905

    accuracy                           0.85    110059
   macro avg       0.85      0.85      0.85    110059
weighted avg       0.85      0.85      0.85    110059

The Confusion matrix is :
 [[47753  7401]
 [ 9091 45814]]

-------------------------------------------------------------------
```

```
ac_cv_score = cross_val_score(ac,X_train,y_train,cv=5)
print('The cross validation score is :',ac_cv_score.mean())
```

```
The cross validation score is : 0.8513568839218285
```

❖ XGB Classifier

```
xc = xb.XGBClassifier()
xc.fit(X_train,y_train)
xc_pred = xc.predict(X_test)
xc_accuracy = accuracy_score(y_test,xc_pred)
xc_cf = classification_report(y_test,xc_pred)
xc_cm = confusion_matrix(y_test,xc_pred)
xc_train_score = xc.score(X_train,y_train)
xc_test_score = xc.score(X_test,y_test)

print('XGB Classifier')
print('-------------------------------------------------------------------\n')
print('The Score on train set is    :',xc_train_score)
print('The Score on test set is     :',xc_test_score)
print('The Accuracy on test set is :',xc_accuracy)
print('The Classification report is :\n',xc_cf)
print('The Confusion matrix is :\n',xc_cm)
print('\n-------------------------------------------------------------------')
```

```
XGB Classifier
-------------------------------------------------------------------

The Score on train set is    : 0.9560752794943984
The Score on test set is     : 0.9508536330513634
The Accuracy on test set is : 0.9508536330513634
The Classification report is :
               precision    recall  f1-score   support

           0       0.96      0.94      0.95     55154
           1       0.94      0.96      0.95     54905

    accuracy                           0.95    110059
   macro avg       0.95      0.95      0.95    110059
weighted avg       0.95      0.95      0.95    110059

The Confusion matrix is :
 [[51905  3249]
 [ 2160 52745]]

-------------------------------------------------------------------
```

```
xc_cv_score = cross_val_score(xc,X_train,y_train,cv=5)
print('The cross validation score is :',xc_cv_score.mean())
```

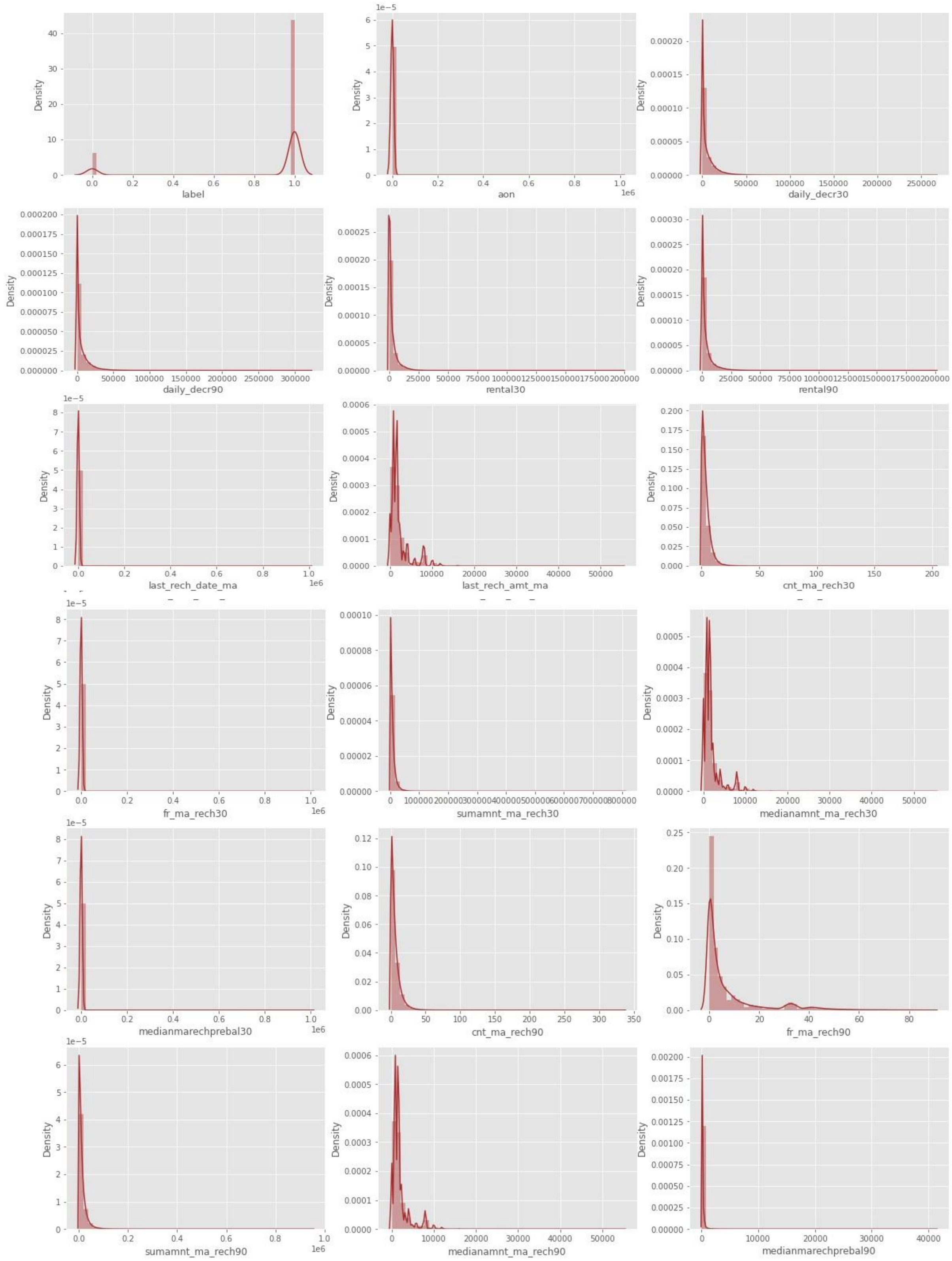
```
The cross validation score is : 0.9507131881858066
```

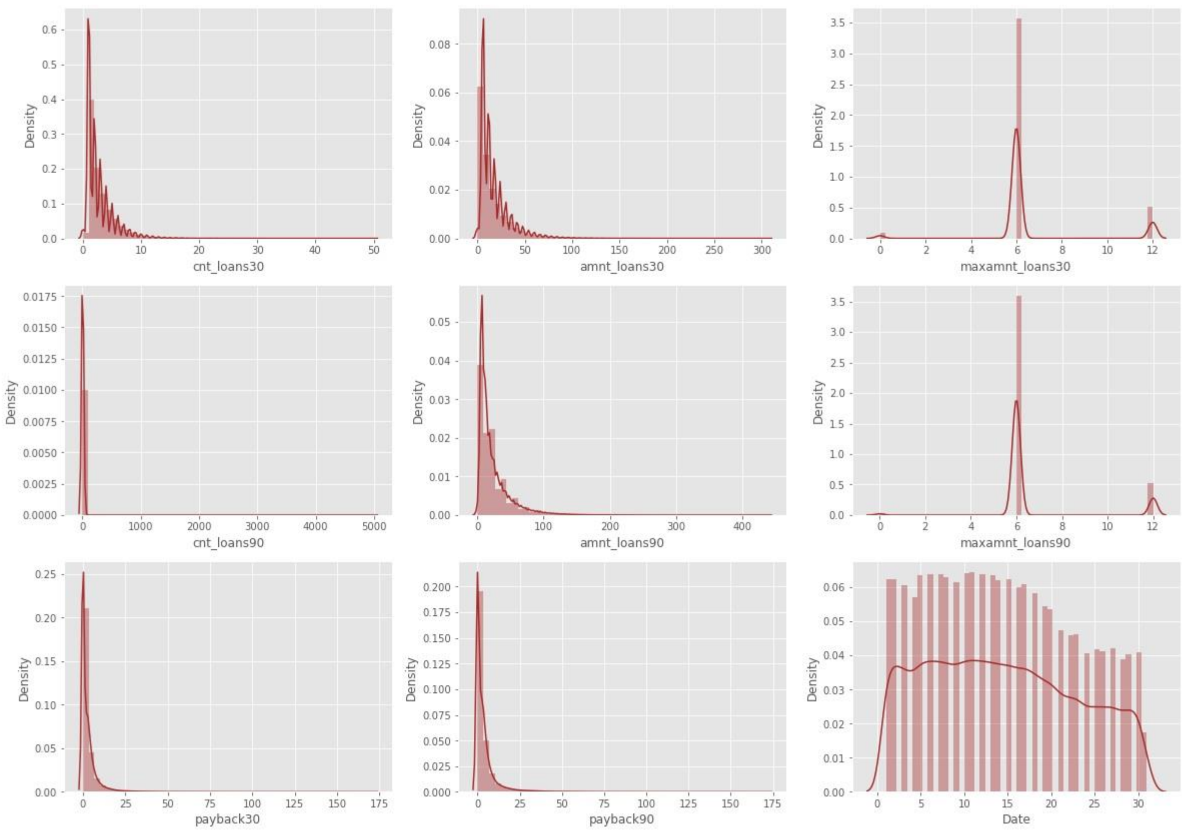
- Key Metrics for success in solving problem under consideration

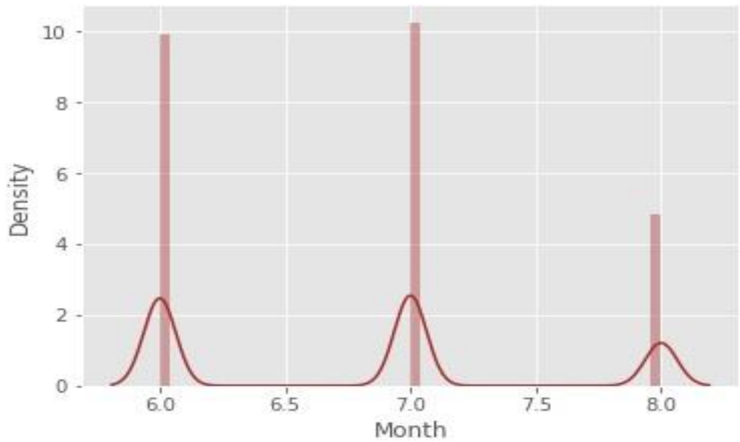
  Following are the metrics to observe good model:

  - ❖ Accuracy simply measures how often the classifier correctly predicts. Accuracy can be defined as the ratio of the number of correct predictions and the total number of predictions.
  - ❖ Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives.
  - ❖ Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive.
  - ❖ It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.
  - ❖ Cross_val_score: Cross-validation is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.
  - ❖ AUC-ROC: The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR(True Positive Rate) against the FPR(False Positive Rate) at various threshold values and separates the ‘signal’ from the ‘noise’.
  - ❖ AUC is the measure of the ability of a classifier to distinguish between classes.

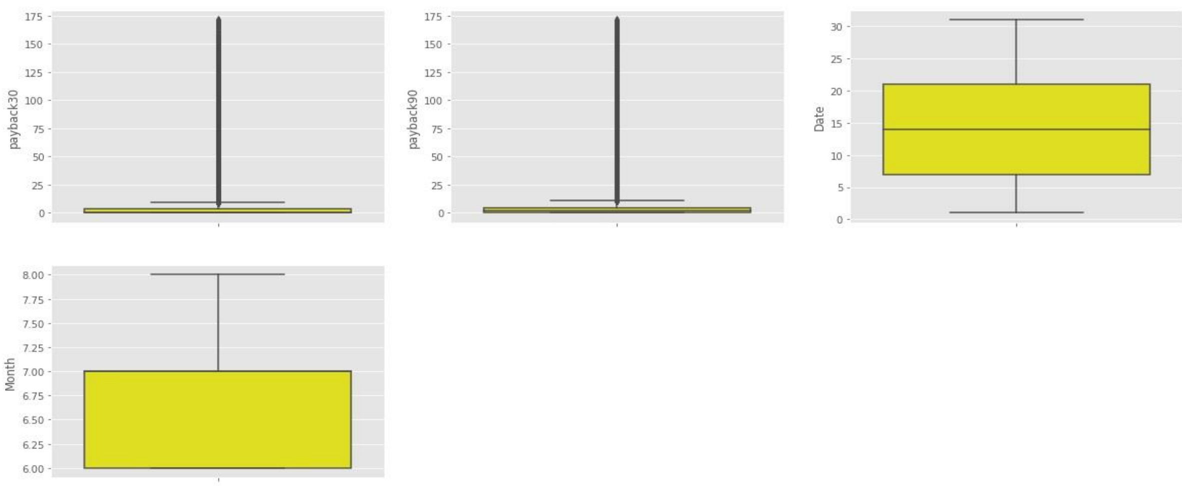
- Visualizations
  - ❖ Univariate
    - ▪ Using distplot checking distribution

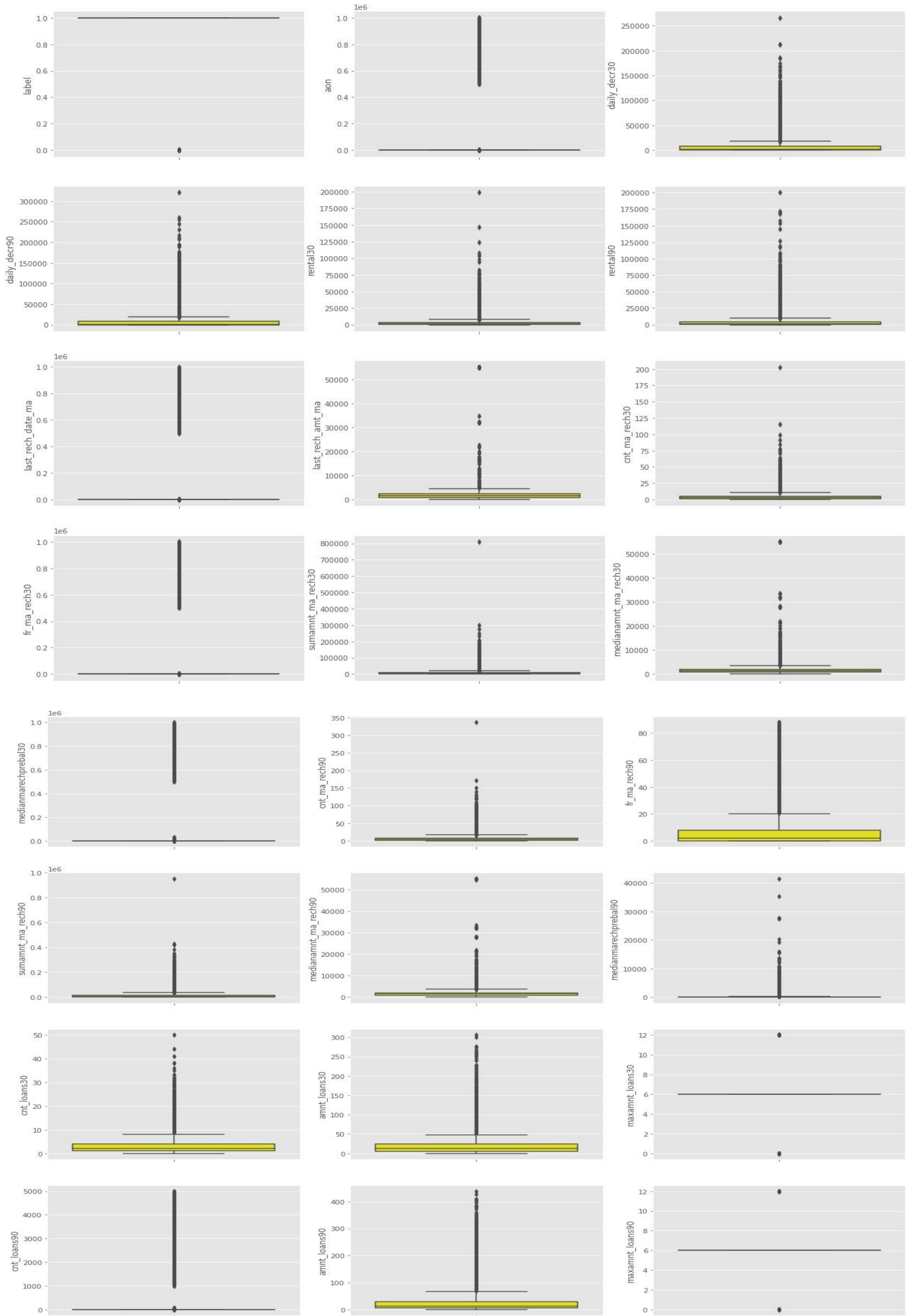




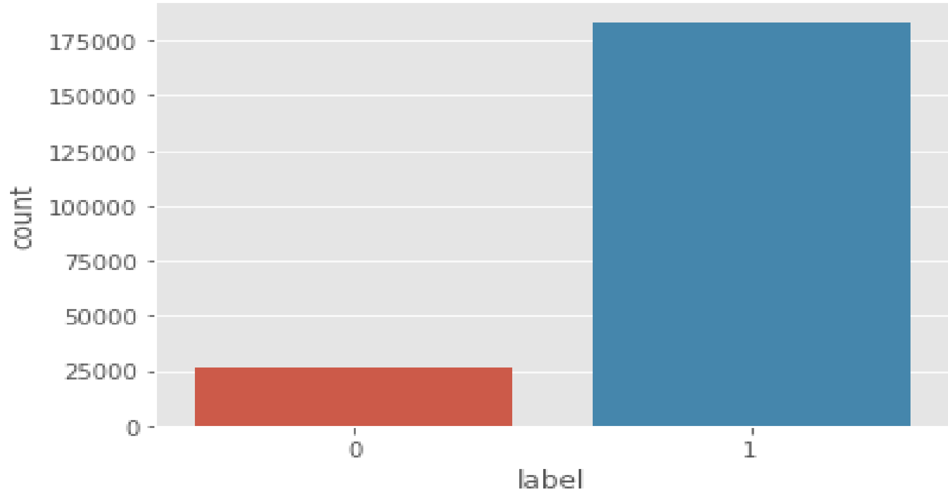


- Using boxplot checking outliers




label
aon
daily_decr30
daily_decr90
rental30
rental90
last_rech_date_ma
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_loans30
amnt_loans30
maxamnt_loans30
cnt_loans90
amnt_loans90
maxamnt_loans90

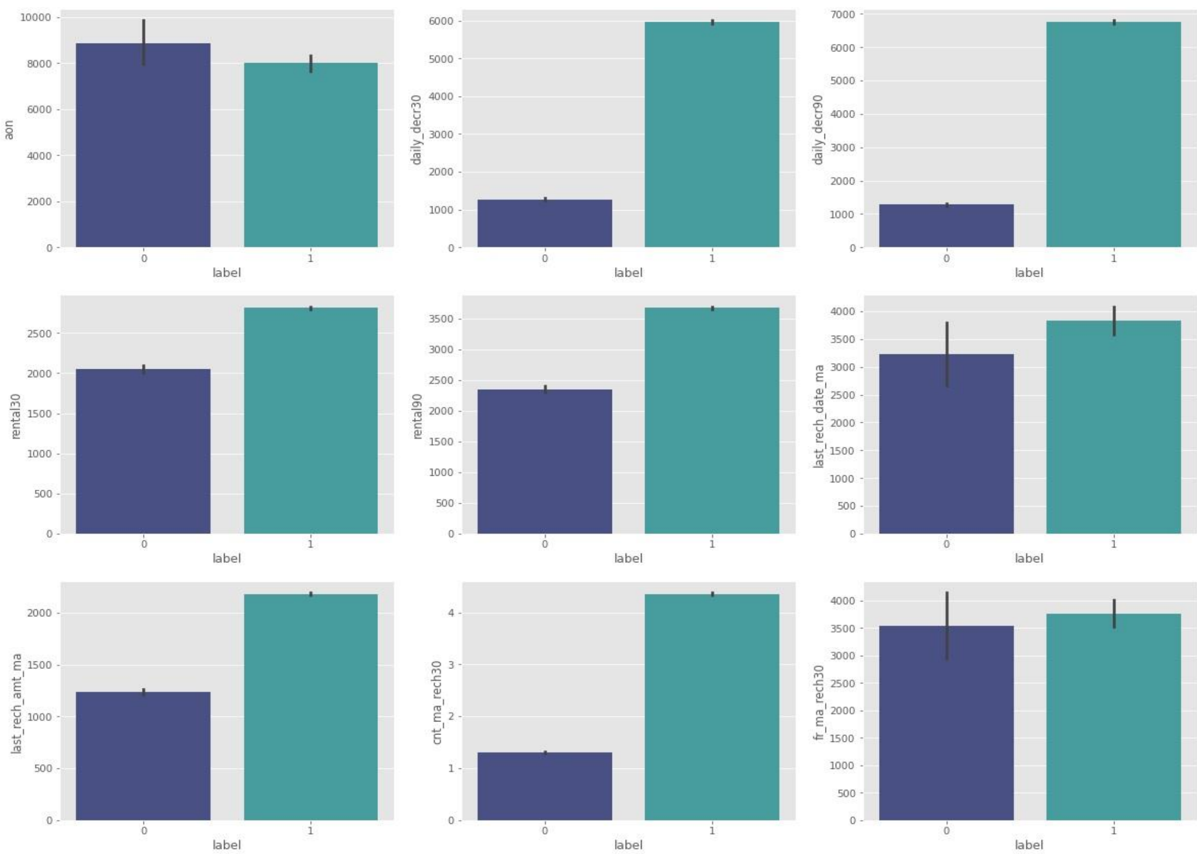
- Visualize target

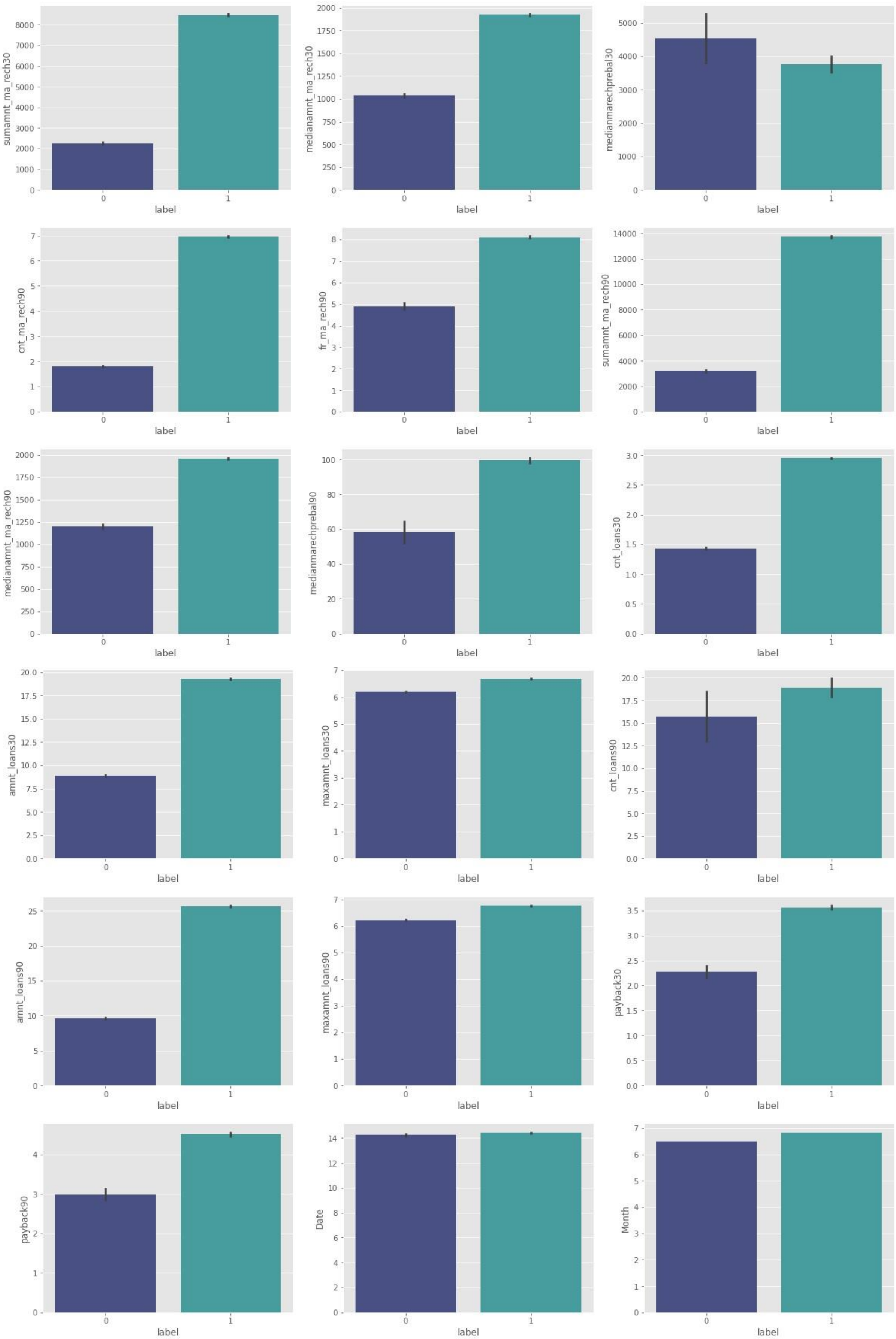


Outcome:

- ✓ Skewness is there
- ✓ Outliers is there
- ✓ Target is imbalance

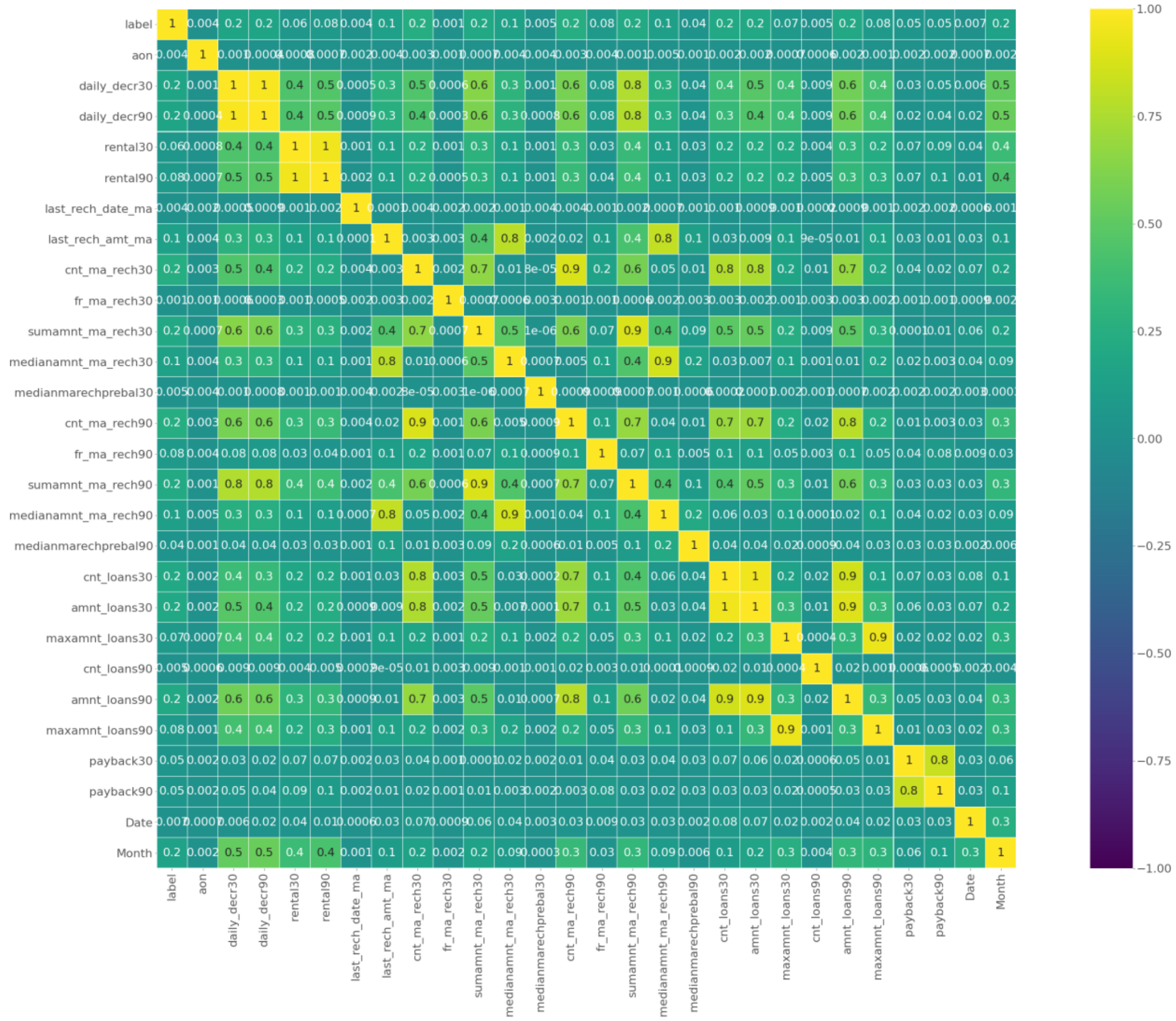
Bivariate




sumamnt_ma_rech30
label
medianamnt_ma_rech30
label
medianmarechprebal30
label
cnt_ma_rech90
label
fr_ma_rech90
label
sumamnt_ma_rech90
label
medianamnt_ma_rech90
label
medianmarechprebal90
label
cnt_loans30
label
amnt_loans30
label
maxamnt_loans30
label
cnt_loans90
label
amnt_loans90
label
maxamnt_loans90
label
payback30
label
payback90
label
Date
label
Month
label

- Outcome
  - ✓ Defaulters are more in case of 'aon','medianmarechprebal30'.
  - ✓ Non-Defaulters are more in case of 'daily_decr30','daily_decr90','rental30', 'rental90','last_rech_date_ma','last_rech_amt_ma','cnt_ma_rech30', 'fr_ma_rech30','sumamnt_ma_rech30','medianamnt_ma_rech30', 'cnt_ma_rech90','fr_ma_rech90','sumamnt_ma_rech90', 'medianamnt_ma_rech90','medianmarechprebal90','cnt_loans30', 'amnt_loans30', 'maxamnt_loans30', 'cnt_loans90','amnt_loans90', 'maxamnt_loans90', 'payback30','payback90','Date','Month'.

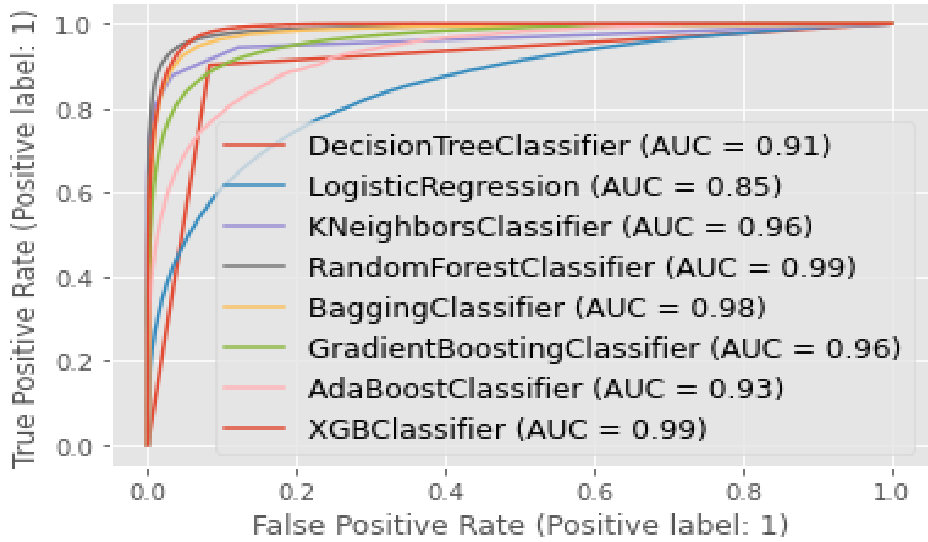
❖ Correlation matrix



- Outcome:
  - ✓ All features are vrey less correlated with target.

- Interpretation of the Results

❖ ROC-AUC Curve



Outcome:

✓ XGB Classifier & Random Forest Classifier have best score.

❖ Score of all model

	Model Name	Train Score	Test Score	Accuracy Score	Cross Validation Score
0	Logistic Regression	0.770100	0.773376	0.773376	0.770392
1	KNeighbors Classifier	0.925110	0.898027	0.898027	0.889635
2	Decision Tree Classifier	0.999992	0.909476	0.909476	0.904281
3	Random Forest Classifier	0.999988	0.952880	0.952880	0.948902
4	Bagging Classifier	0.996141	0.938724	0.938724	0.935495
5	Gradient Boosting Classifier	0.899499	0.901008	0.901008	0.899686
6	AdaBoost Classifier	0.848483	0.850153	0.850153	0.851357
7	XGB Classifier	0.956075	0.950854	0.950854	0.950713

Outcome:

✓ The difference between accuracy score and cross validation score is less for XGB Classifier as compared with Random Forest Classifier. So, XGB Classifier is final model on this dataset.

❖ Hyperparameter Tuning

```
clf = xb.XGBClassifier()

param = {'learning_rate' : np.arange(0.01,0.8),
         'max_depth' : range(2,30),
         'subsample' : np.arange(0.1,1)
         }

grd = GridSearchCV(clf,param_grid=param)
grd.fit(X_train,y_train)


clf = grd.best_estimator_

clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)

ac_con = confusion_matrix(y_test,y_pred)

print("\n Best parameter",grd.best_params_)
print("\n Confusion Matrix \n",ac_con)
print("\n Accuracy after hyper parameter tunning",accuracy_score(y_test,y_pred))
```

```
Best parameter {'learning_rate': 0.01, 'max_depth': 26, 'subsample': 0.1}

Confusion Matrix
[[51050  4104]
 [ 4373 50532]]

Accuracy after hyper parameter tunning 0.9229776756103545
```

Outcome:

- ✓ After hyperparameter tuning, not able to improve performance. So, default parameters are best parameters.

CONCLUSION

- Key Findings and Conclusions of the Study

  Mostly, the customers have the intension of repaying. There are certain cases when the customers have no intension of repayment but the number of such customers are few. With the model built, we can certainly determine those customers having intension of repayment or not.

- Learning Outcomes of the Study in respect of Data Science
  - ❖ The dataset was full of outliers, skewness and unbalanced data which was the biggest challenge to overcome. Hence, data cleaning was very important to get proper prediction.
  - ❖ While removing the outliers, in most of the scenarios Z-score was used as outlier removal technique since it performs quite well with less data loss. In this dataset major challenges was using IQR & zscore the loss of data is above the criteria so I used percentile method.

- Limitations of this work and Scope for Future Work
  - ❖ The solution can be applied to the customer having a transaction history but the model may not perform well with customer having new profile and no transaction history. Nevertheless, the model will perform well with customer having transaction history and can predict whether a person will be a defaulter or non-defaulter.
  - ❖ This data set contains data of the year 2016 belonging to PSW telecom circle.If we get data of other years along with other telecom companies we can predict on varied scenario.