PROJECT REPORT ON:

“MICRO CREDIT DEFAULTER PROJECT”

SUBMITTED BY

Safiya Firdose Khan

ACKNOWLEDGMENT

It is my deepest pleasure and gratification to present this report. Working on this project was an incredible experience that has given me a very informative knowledge regarding the data analysis process. All the required information and dataset are provided by **Flip Robo Technologies** (Bangalore) that helped me to complete the project. I want to thank my SME **KHUSHBOO GARG** for giving the dataset and instructions to perform the complete case study process.

Contents:

1.INTRODUCTION

1.1 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.

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

1.2 Conceptual Background of the Domain Problem

MFS 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).

1.3 Review of Literature

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. 4 | P a g e MICROCREDIT DEFAULTER PROJECT Today, microfinance is widely accepted as a poverty-reduction tool, representing $70 billion in outstanding loans and a global outreach of 200 million clients.

1.4 Motivation for the Problem Undertaken

We understand the importance of communication and how it effects a person's life and lack of communication can cause lot of uncertain problems so we want to work in order to bridge this gap between people. 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.

2. Analytical Problem Framing

2.1 Mathematical/ Analytical Modeling of the Problem

We need to build a Machine Learning 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 the dataset, the Label '1' indicates that the loan has been paid i.e., non-defaulter, while, Label '0' indicates that the loan has not been paid i.e., defaulter.

Clearly it is a binary classification problem where we need to use classification algorithms to predict the results. There were no null values in the dataset. There were some unwanted entries like more than 90% of zero values present in some of the columns which means these customers have no loan history so, I have dropped those columns. I found some negative values while summarizing the statistics of the dataset, I have converted them into positive. To get better insights on features, I have used some plots like pie plot, count plot, bar plot, distribution plot, box plots etc. There were lots of skewness and outliers present in our dataset which need to be cleaned using appropriate techniques and balanced the data. At last, I have built many classification models to predict the defaulter level at the institution.

2.2 Data Sources and their formats

The data set provided by Flip Robo was in the format of CSV (Comma Separated Values). The dimension of the dataset is 209593 rows and 37 columns including the target variable "label".

The features information is as follows:

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 : User Identifier

2.3 Data Preprocessing Done

Data pre-processing is the process of converting raw data into a well-readable format to be used by Machine Learning model. Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre-process our data before feeding it into our model. I have used following pre-processing steps:

- Importing necessary libraries and loading dataset as a data frame.
- Used pandas to set display maximum columns ensuring not to find any truncated information.
- Checked some statistical information like shape, number of unique values present, info, finding zero values etc.

- Checked for null values and did not find any null values.
- Dropped some unwanted columns like Unnamed:0, pcircle, msisdn as they are of no use for prediction.
- Dealt with zero values by verifying the percentage of zero values in each column and decided to discard the columns having more than 90% of zero values.
- Converted time variable “pdate” from object into datetime and extracted Day, Month and Year for better understanding. Checked value counts for each and dropped “Year” column as it contains unique value throughout the dataset.
- Checked unique values and value counts of target variable.
- While checking the statistical summary of the dataset, I found some columns having negative values which were invalid and unrealistic so decided to convert negative values into positive using absolute command.
- Visualized each feature using seaborn and matplotlib libraries by plotting several plots.
- Identified outliers using box plots and I tried to remove them using both Zscore and IQR method and got huge data loss of around 19% and 63% respectively, so removed outliers using percentile method by setting data loss to 2%.
- Checked for skewness and removed skewness in numerical columns using power transformation method (yeo-johnson).
- Used Pearson’s correlation coefficient to check the correlation between label and features. With the help of heatmap, correlation bar graph was able to understand the Feature vs Label relativity and insights on multicollinearity amongst the feature columns.
- Separated feature and label data and feature scaling is performed using MinMaxScalar method to avoid any kind of data biasness.
- Since the dataset was imbalanced, so I performed Oversampling method using SMOTE to balance the data.
- Checked for the best random state to be used on our Classification Machine Learning model pertaining to the feature importance details.
- Finally, created classification model along with evaluation metrics.

2.4 Data Inputs- Logic- Output Relationships

The dataset consists of label and features. The features are independent and label is dependent as the values of our independent variables changes as our label varies.

• To analyze the relation between features and label I have used many plotting techniques where I found some of the columns having strong relation with label.

• The visualization helped me to understand that maximum distribution is for non-defaulter for all the features & maximum defaulter list are from people who have Average payback time in days over last 30 & 90 days, also frequency of recharge done in the main account since last 90 days. So, the features, which I have kept after dropping few are having some kind of relationship with the output.

• I have checked the correlation between the label and features using heat map and bar plot. Where I got the positive correlation between the label and features and there was no much relation.

2.5 Hardware and Software Requirements and Tools Used

While taking up the project we should be familiar with the Hardware and software required for the successful completion of the project. Here we need the following hardware and software.

Hardware required: -

- Processor — core i5 and above
- RAM — 8 GB or above
- SSD — 250GB or above

Software/s required: -

- Programming language: Python
- Distribution: Anaconda Navigator
- Browser based language shell: Jupyter Notebook

Libraries required :-

In [1]:

```
1 import numpy as np
2 import pandas as pd
3 import seaborn as sns
4 import matplotlib.pyplot as plt
5
6 pd.pandas.set_option('display.max_columns',None)
7
8 import warnings
9 warnings.filterwarnings('ignore')
```

- ✓ **Numpy:** It is defined as a Python package used for performing the various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using Numpy arrays are faster than the normal Python array.

- ✓ **Pandas:** Pandas is a popular Python-based data analysis toolkit which can be imported using import pandas as pd. It presents a diverse range of utilities, ranging from parsing multiple file formats to converting an entire data table into a numpy matrix array. This makes pandas a trusted ally in data science and machine learning.

- ✓ **Seaborn:** Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

- ✓ **Matplotlib:** matplotlib.pyplot is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

Importing the dataset :-

As the dataset is very large, I was facing memory issues while trying to read the dataset. So, I have divided the dataset into chunks and read the chunks. Finally, I have combined the chunks, so as to get the entire dataset.

In [2]:
```
# As the dataset is very large, let's divide the dataset into chunks

ChunkSize = 10
for chunk in pd.read_csv('Data file.csv', chunksize=ChunkSize):
    print(chunk.shape)
    print("*"*80)
    print(chunk.head(2))
    print("*"*80)
    break
```

In [3]:
```
MyList = []
ChunkSize = 10
for chunk in pd.read_csv('Data file.csv', chunksize=ChunkSize):
    MyList.append(chunk)
```

In [7]:
```
# Combining the chunks

df1 = pd.concat(MyList, axis=0)
```

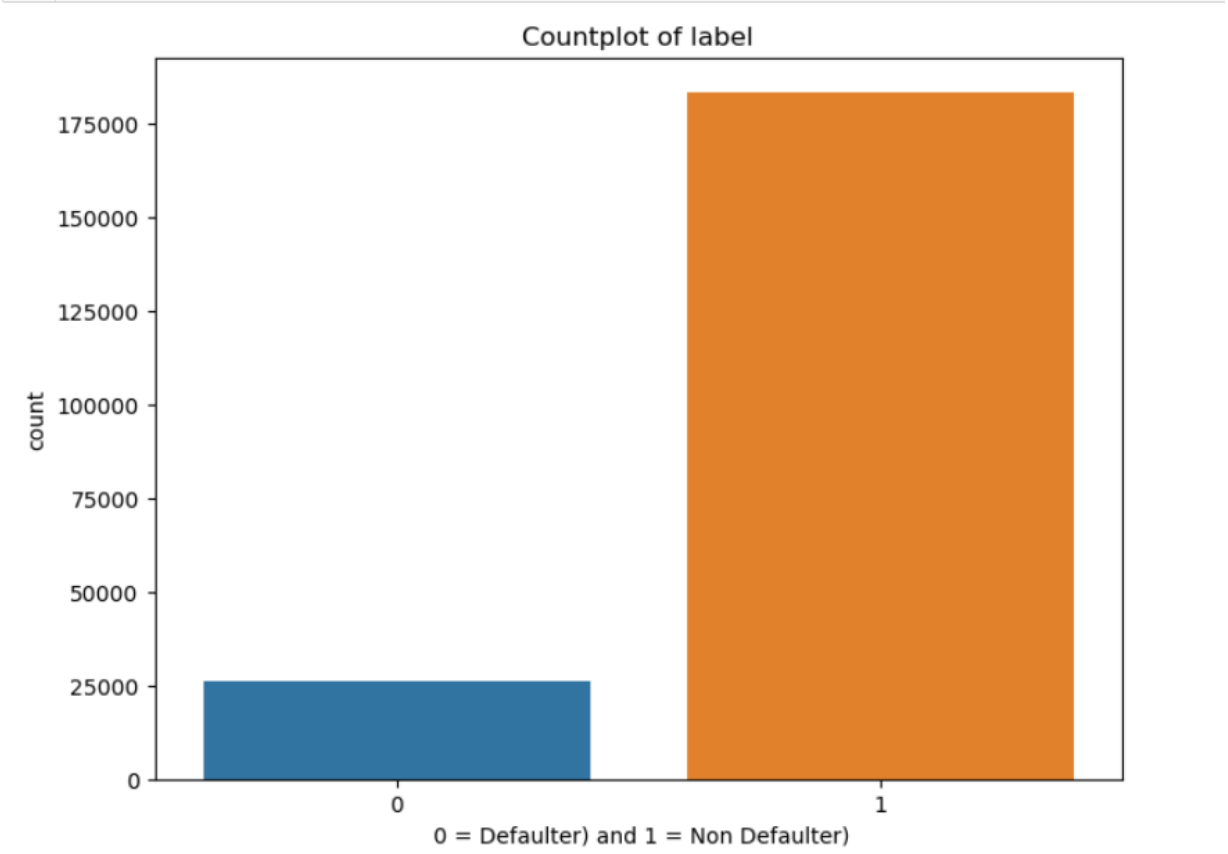
In [8]:
```
df1.shape
```

Out[8]: (209593, 37)

3. Data Analysis and Visualization

3.1 Identification of possible problem-solving approaches (methods)

I have used both statistical and analytical approaches to solve the problem which mainly includes the pre-processing of the data also used EDA techniques and heat map to check the correlation of independent and dependent features. Also, before building the model, I made sure that the input data is cleaned and scaled before it was fed into the machine learning models. The data mainly had class imbalancing issue, so applied SMOTE method to balance the dataset.



For this particular project we need to predict whether the user paid back the credit loan amount within 5 days of issuing the loan. In this dataset, label is the target variable, which consists of two categories, defaulters and non-defaulters. Which means our target column is categorical in nature so this is a classification problem.

I have used many classification algorithms and got the prediction results. By doing various evaluations I have selected Gradient Boosting Classifier as best suitable

algorithm to create our final model as it is giving least difference in accuracy score and cross validation score among all the algorithms used.

In order to get good performance and to check whether my model getting over-fitting and under-fitting I have made use of the K-Fold cross validation and then hyper parameter tuning on best model. Then I saved my final model and loaded the same for predictions.

3.2 Testing of Identified Approaches (Algorithms)

Since label is my target variable which is categorical in nature, from this I can conclude that it is a classification type problem hence I have used following classification algorithms. After the pre-processing and data cleaning I left with 27 columns including target and with the help of feature importance bar graph I used these independent features for model building and prediction. The algorithms used on training the data are as follows:

- Decision Tree Classifier
- Random Forest Classifier
- Extra Trees Classifier
- Gradient Boosting Classifier
- Extreme Gradient Boosting Classifier (XGB)
- Bagging Classifier

From all of these above models, Gradient Boosting Classifier is giving me good performance.

3.3 Key Metrics for success in solving problem under consideration

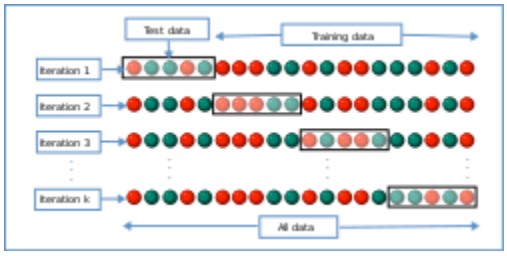
I have used the following metrics for evaluation:

- **Accuracy score:** Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition: Accuracy=Number of correct predictions /Total number of predictions For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

Accuracy=(TP+TN)/(TP+TN+FP+FN)

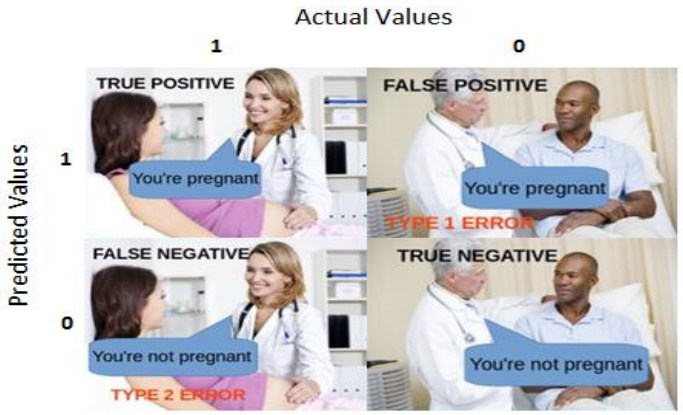
Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives

- **Cross Validation Score:** Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

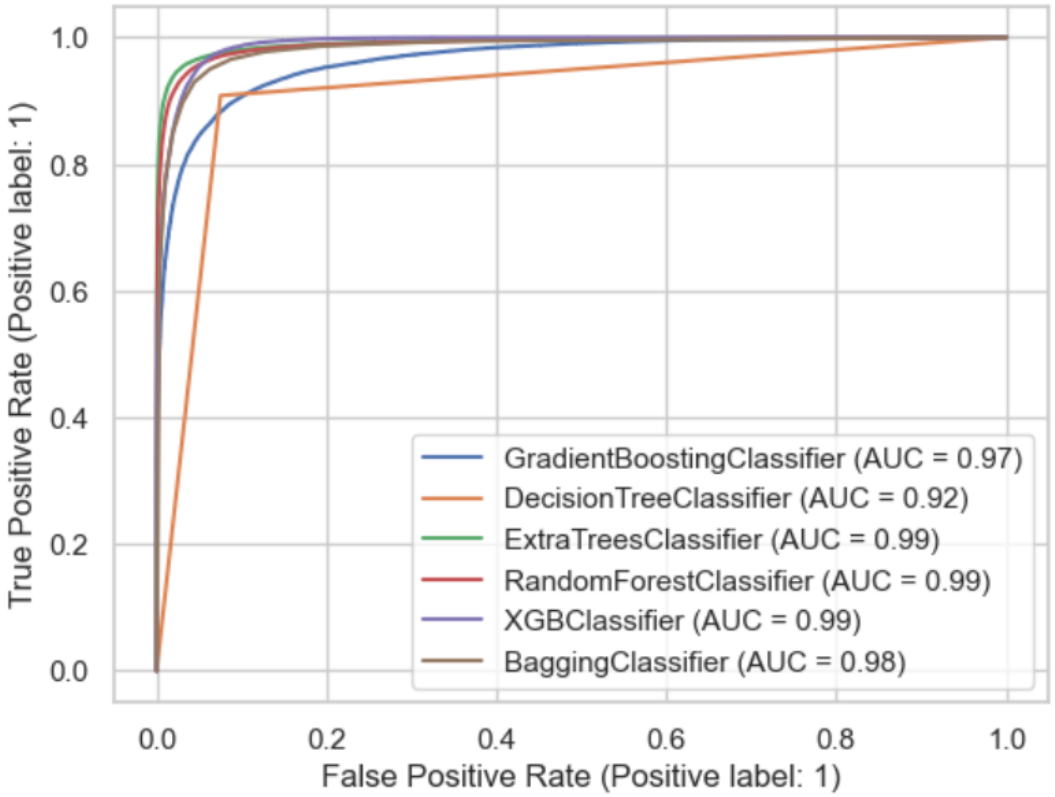


- **Confusion matrix:** A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. Well, it is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values. It is extremely useful for measuring Recall, Precision, Specificity, Accuracy, and most importantly AUC-ROC curves. Let’s understand TP, FP, FN, TN in terms of pregnancy analogy. Log loss: Log Loss is the most important classification metric based

on probabilities. For any given problem, a lower log-loss value means better predictions.



- **AUC-ROC CURVE:** AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1.

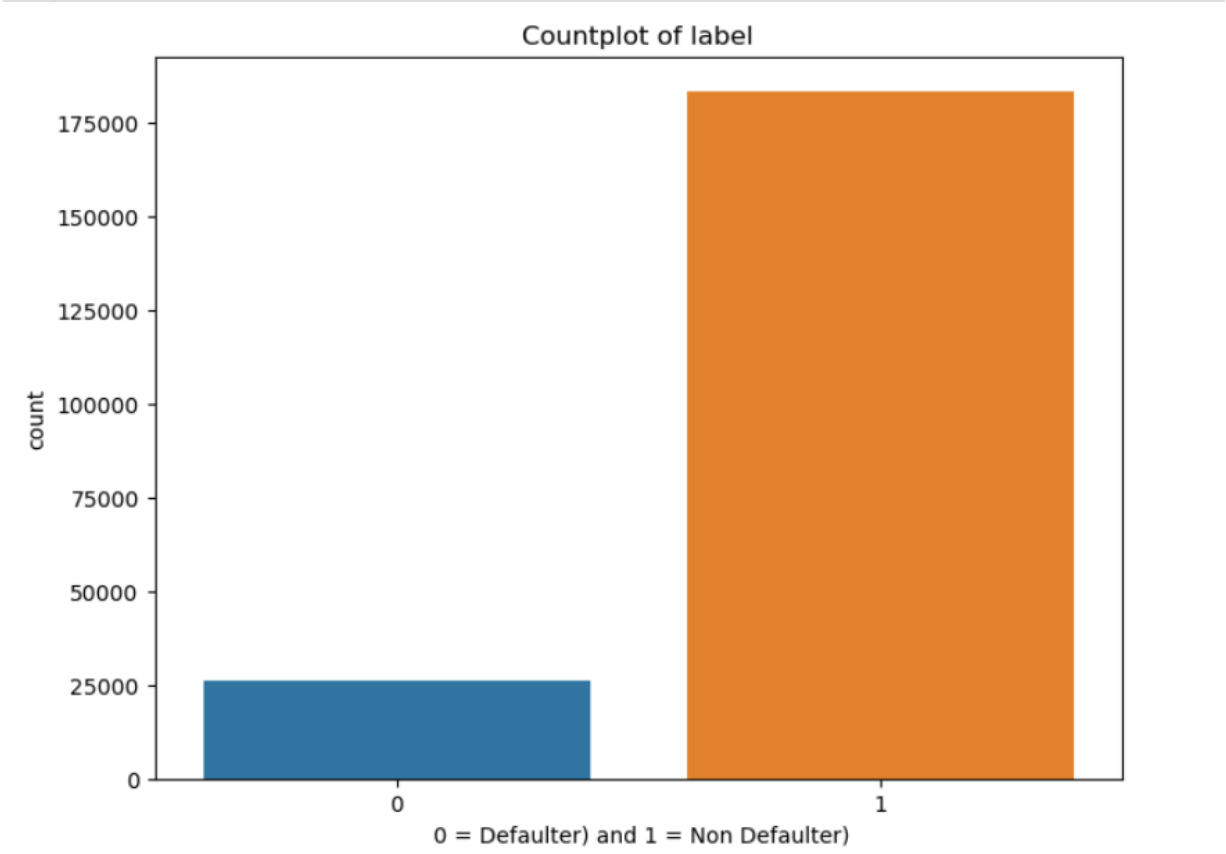

True Positive Rate (Positive label: 1)
False Positive Rate (Positive label: 1)
0.0
0.2
0.4
0.6
0.8
1.0
GradientBoostingClassifier (AUC = 0.97)
DecisionTreeClassifier (AUC = 0.92)
ExtraTreesClassifier (AUC = 0.99)
RandomForestClassifier (AUC = 0.99)
XGBClassifier (AUC = 0.99)
BaggingClassifier (AUC = 0.98)

3.4 Visualizations

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed.

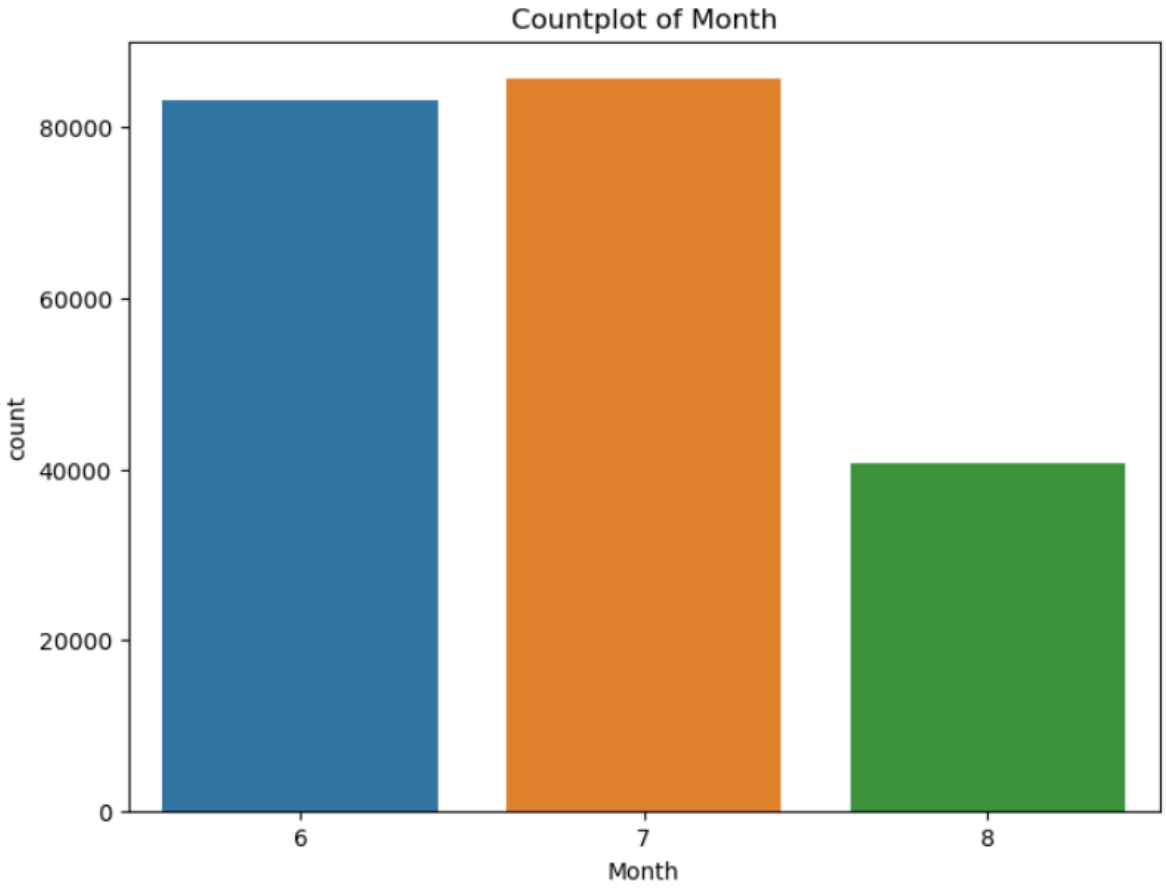
Univariate Analysis:

Count plot of label:



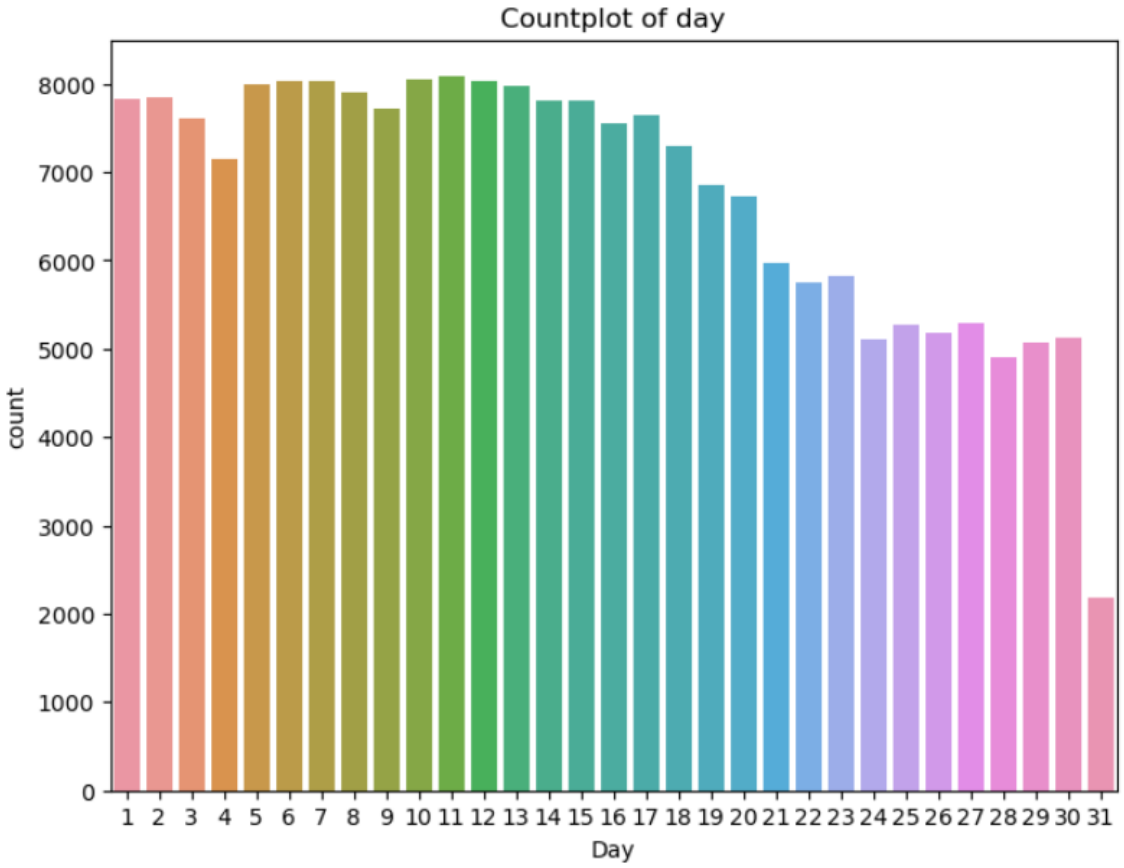
As we can see, majority of the customers are non-defaulters.

Count plot of Month:



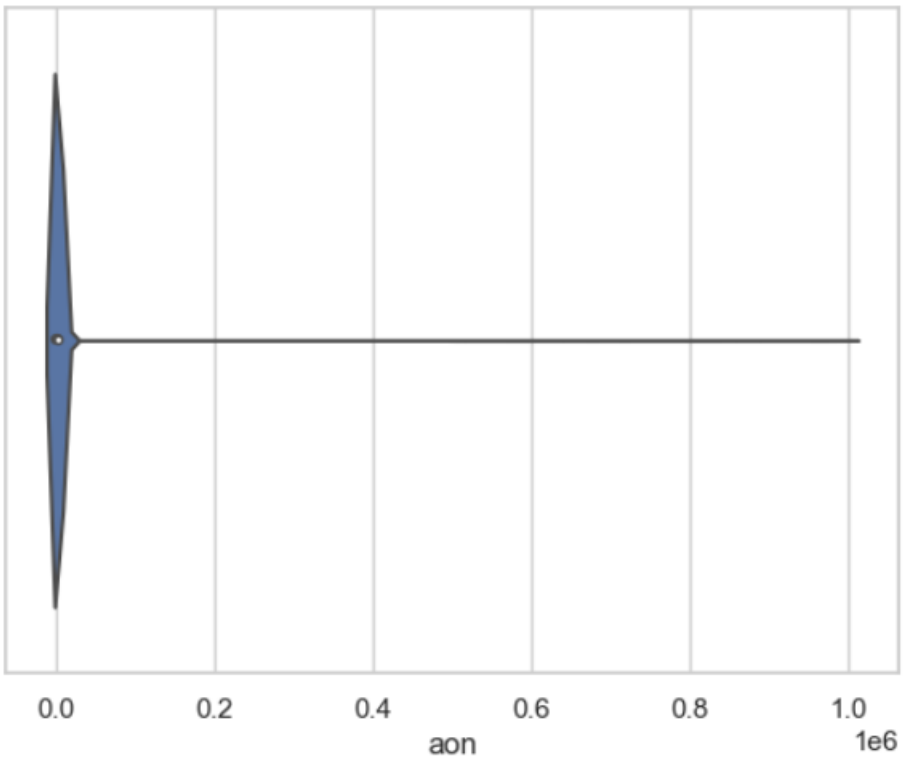
As we can see, majority of the customers (85765) have taken credit in the 7th month.

Countplot of day:



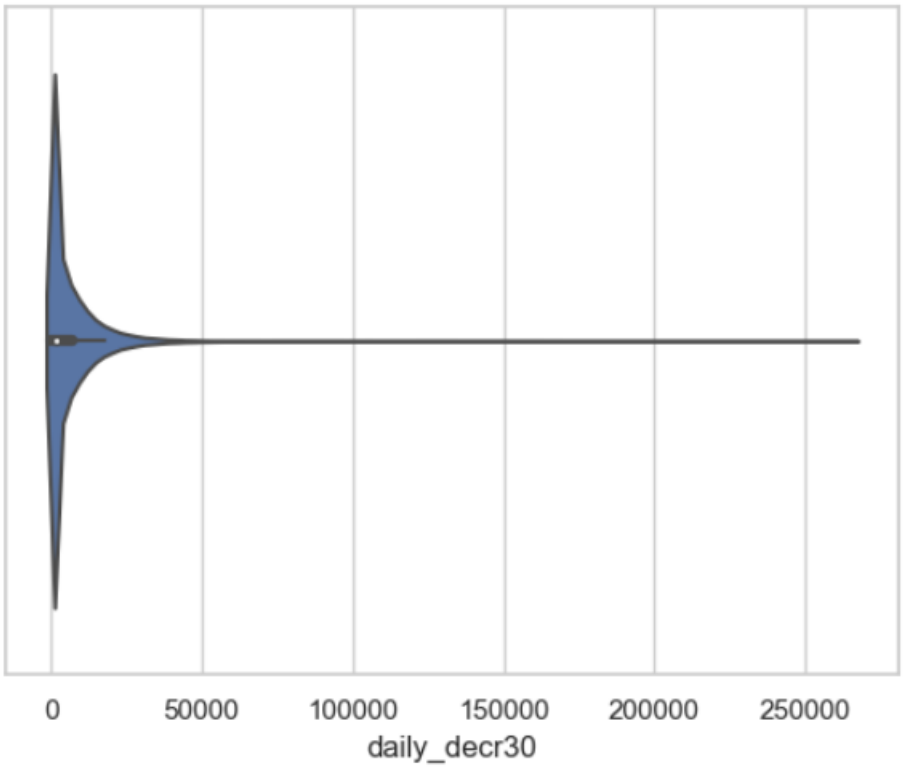
As we can see, majority of the customers (8092) have taken credit on the 11th day of the month.

Violin plot of aon:



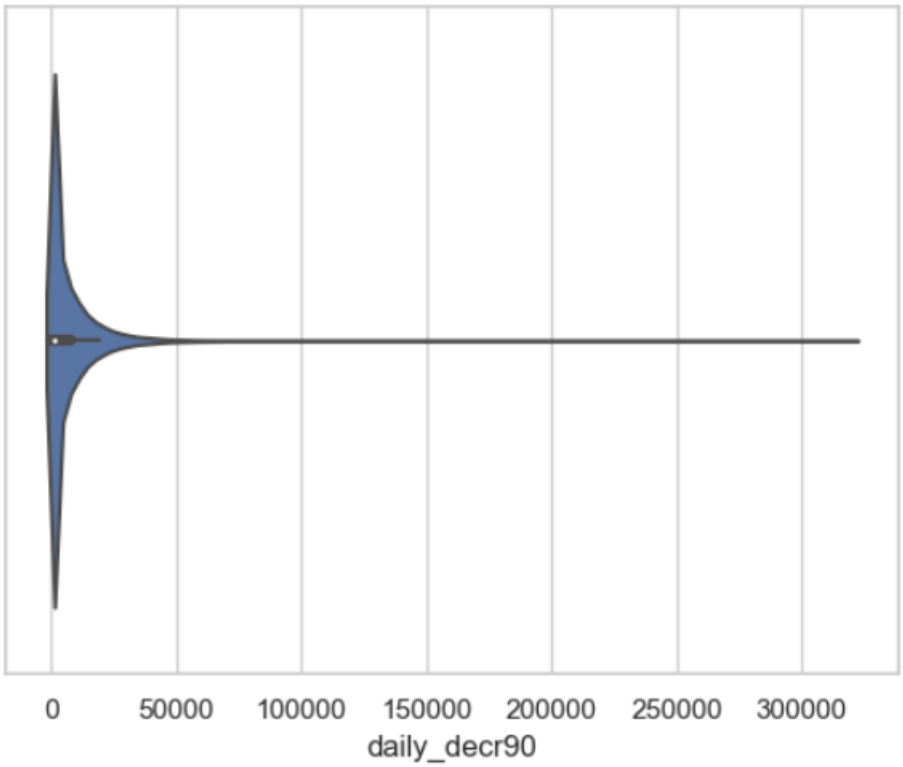
As we can see, age on cellular network in days of maximum customers is 95 days.

Violin plot of daily_decr30:



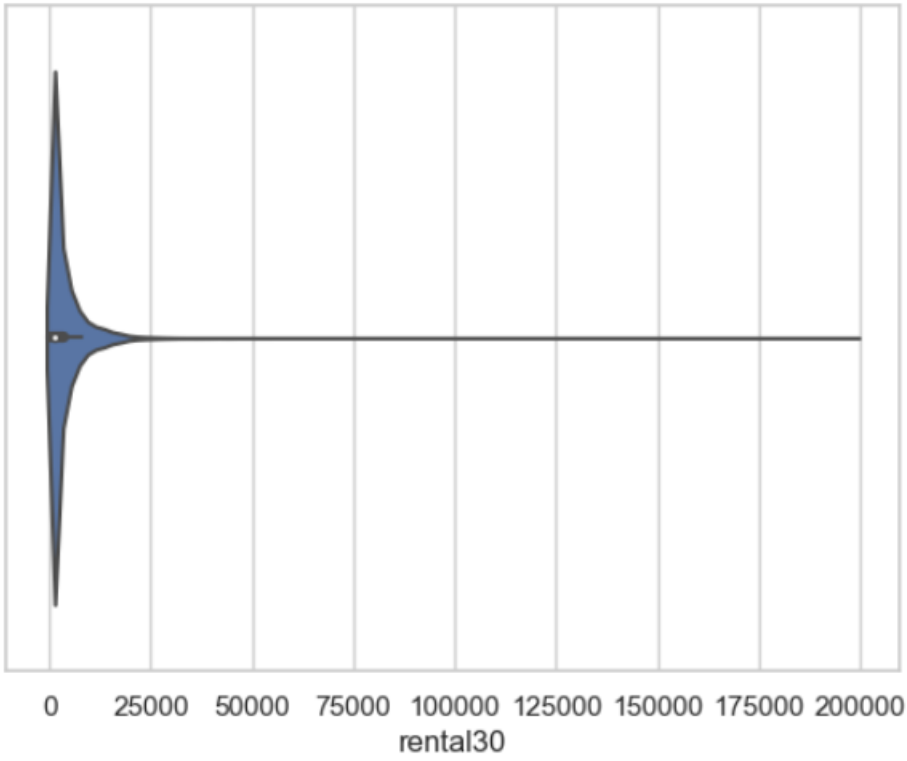
As we can see, daily amount spent from main account, averaged over last 30 days for majority of the customers is 0.

Violin plot of daily_decr90:



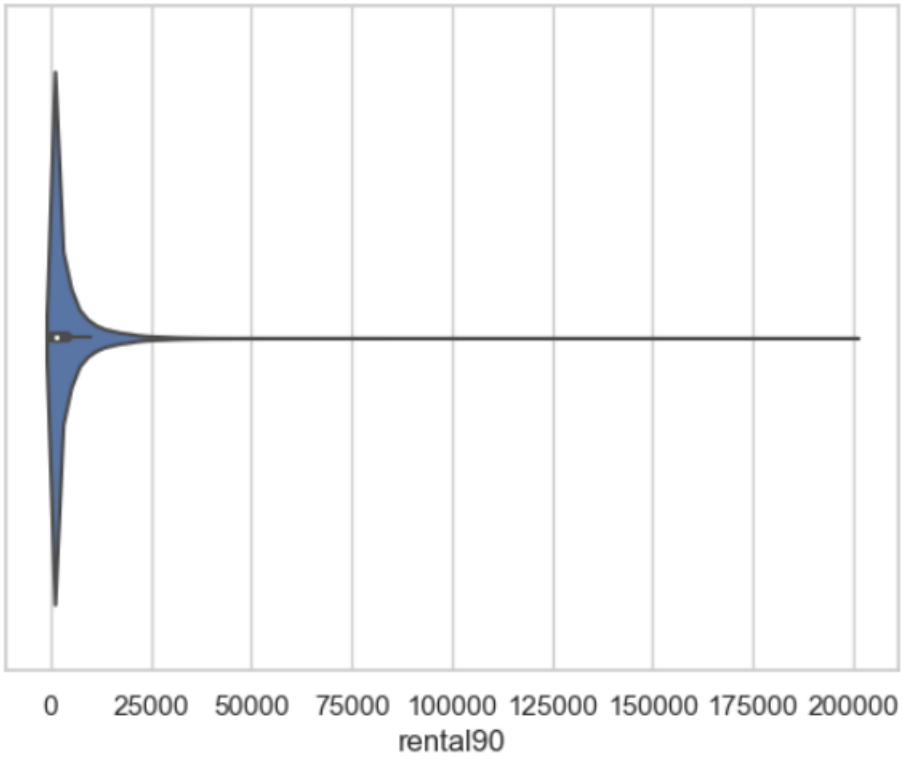
As we can see, daily amount spent from main account, averaged over last 90 days for majority of the customers is 0.

Violin plot of rental30:



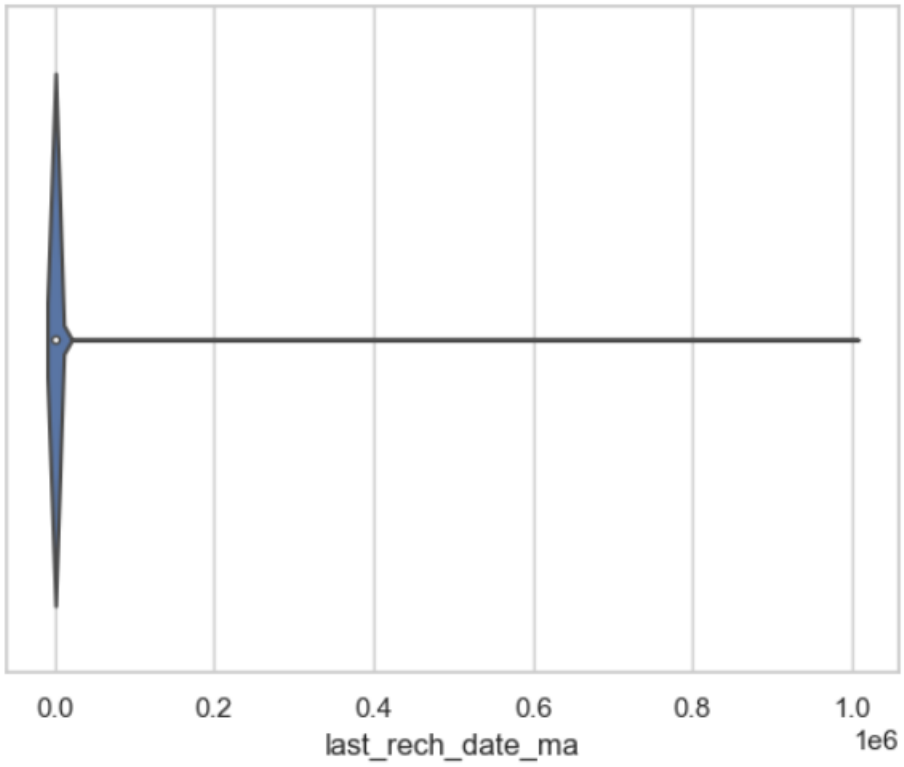
As we can see, average main account balance over last 30 days for majority of the customers is 0.

Violin plot of rental90:



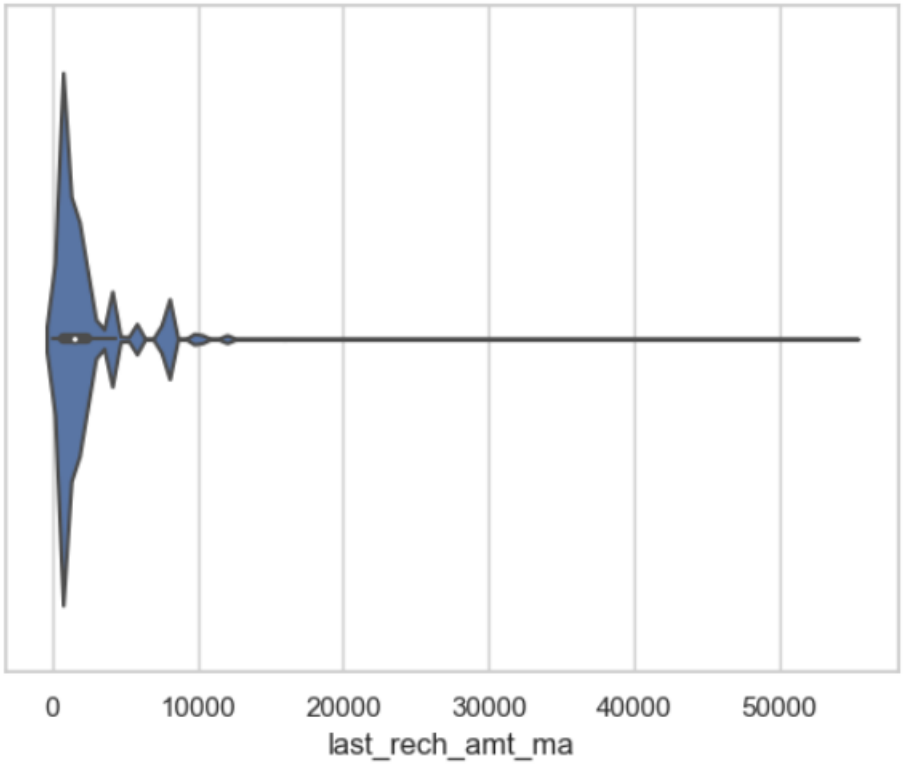
As we can see, average main account balance over last 90 days for majority of the customers is 0.

Violin plot of last_rech_date_ma:



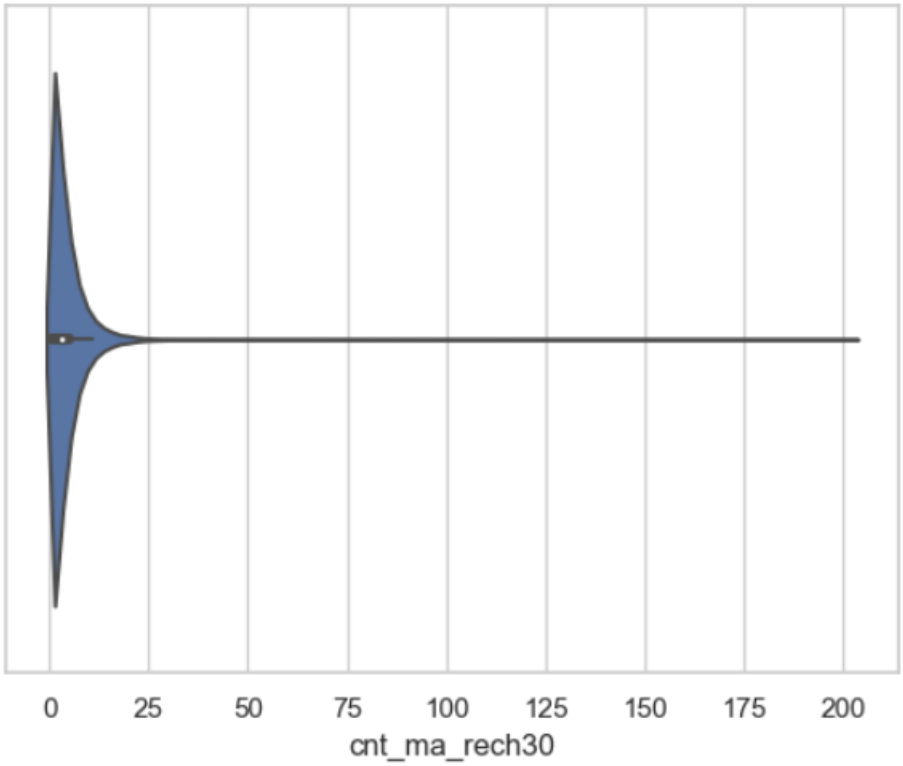
As we can see, number of days till last recharge of main account for majority of the customers is 1.

Violin plot of last_rech_amt_ma:



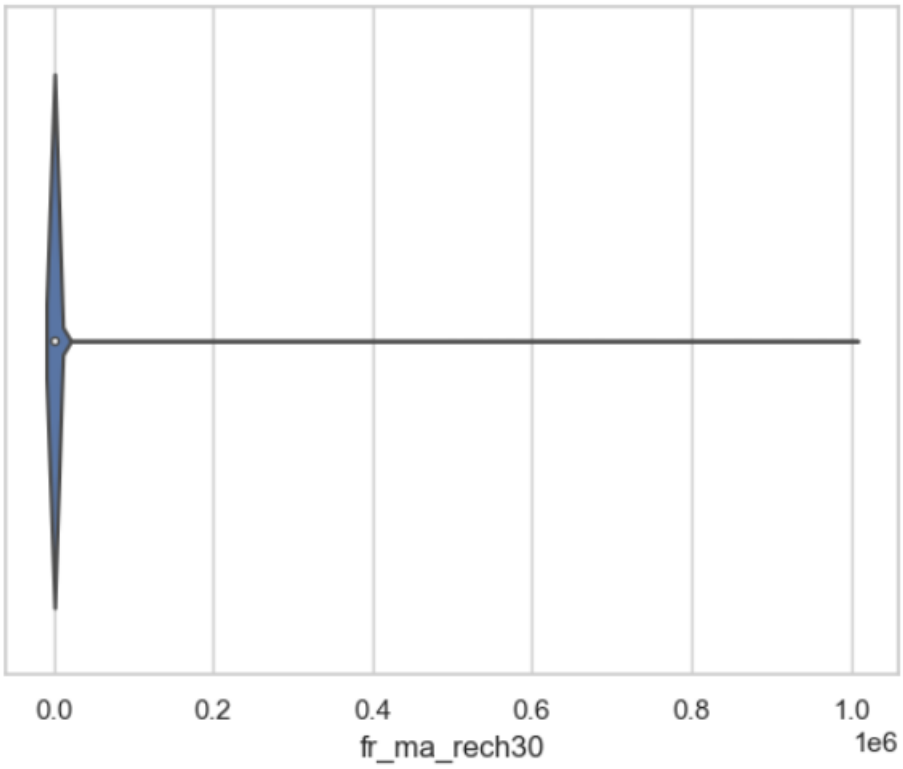
As we can see, amount of last recharge of main account for majority of the customers is 1539.

Violin plot of cnt_ma_rech30:



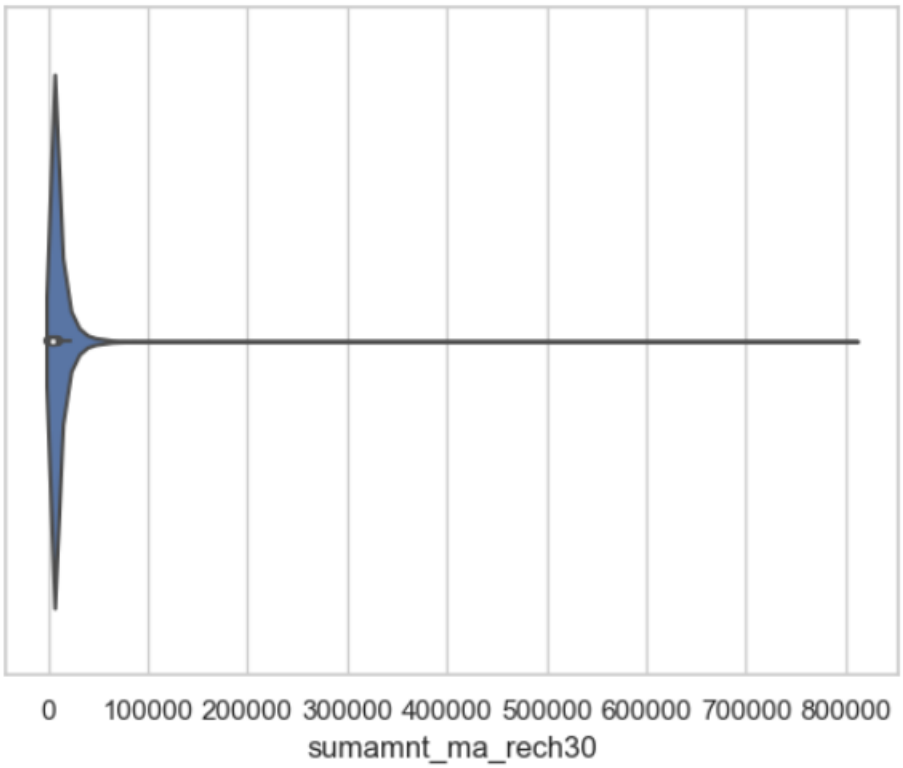
As we can see, number of times main account got recharged in last 30 days for majority of the customers is 1.

Violin plot of fr_ma_rech30:



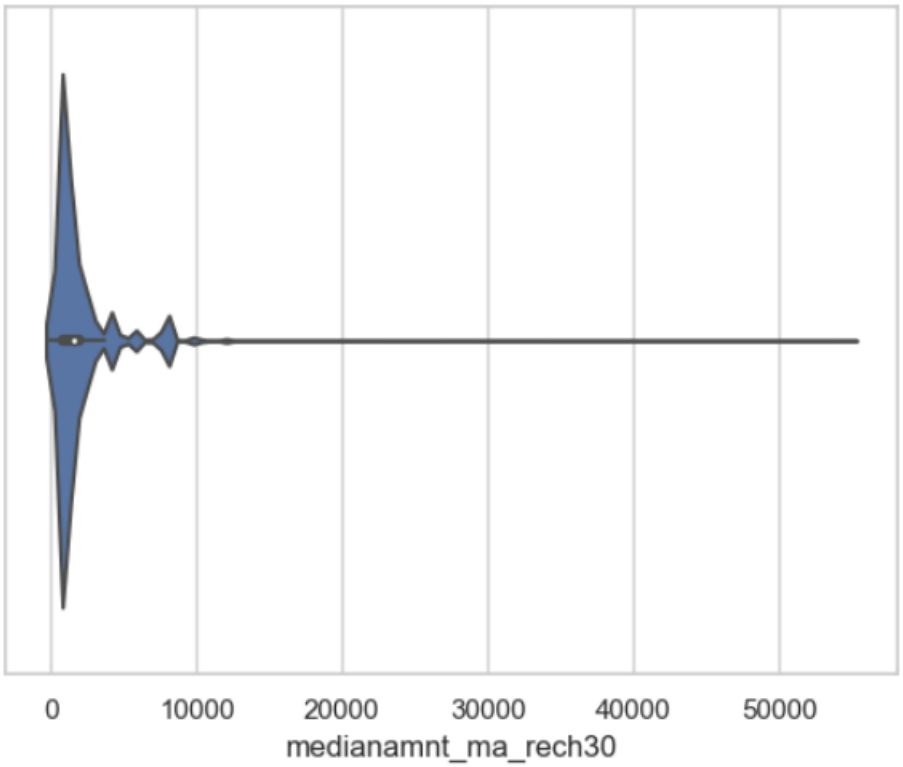
As we can see, frequency of main account recharged in last 30 days for majority of the customers is 0.

Violin plot of sumamnt_ma_rech30:



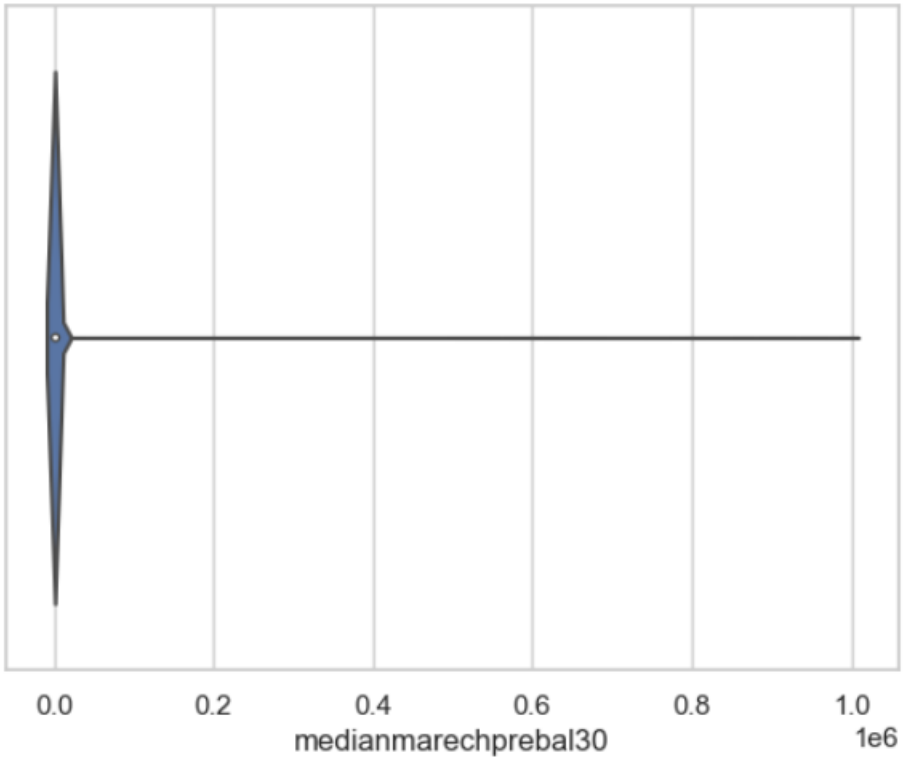
As we can see, total amount of recharge in main account over last 30 days for majority of the customers is 0.

Violin plot of medianamnt_ma_rech30:



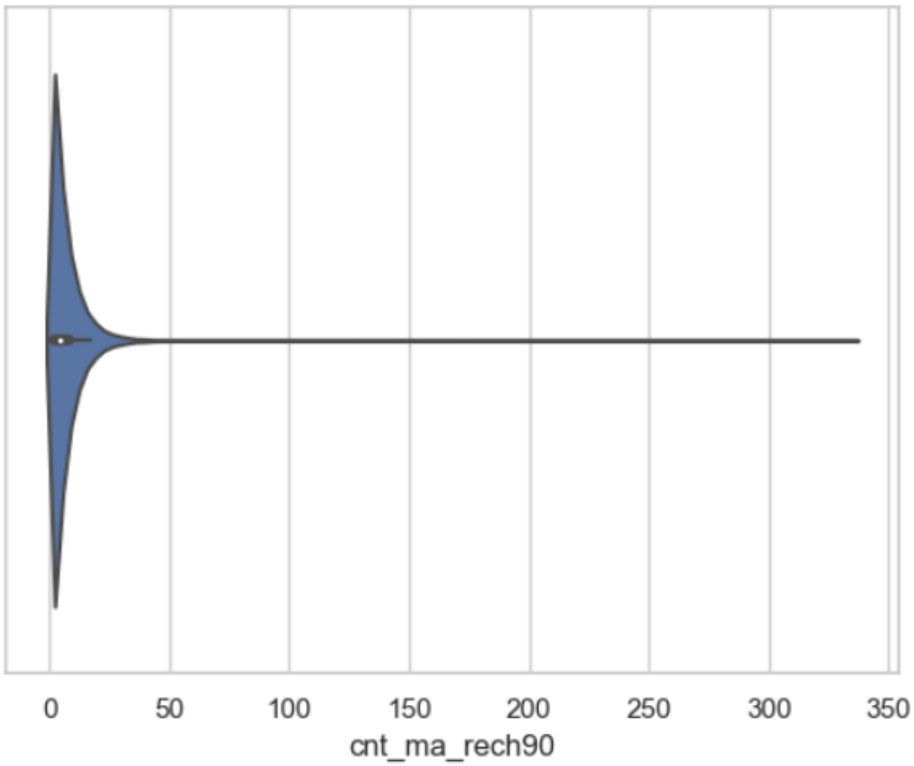
As we can see, median of amount of recharges done in main account over last 30 days at user level for majority of the customers is 1539.

Violin plot of medianmarechprebal30:



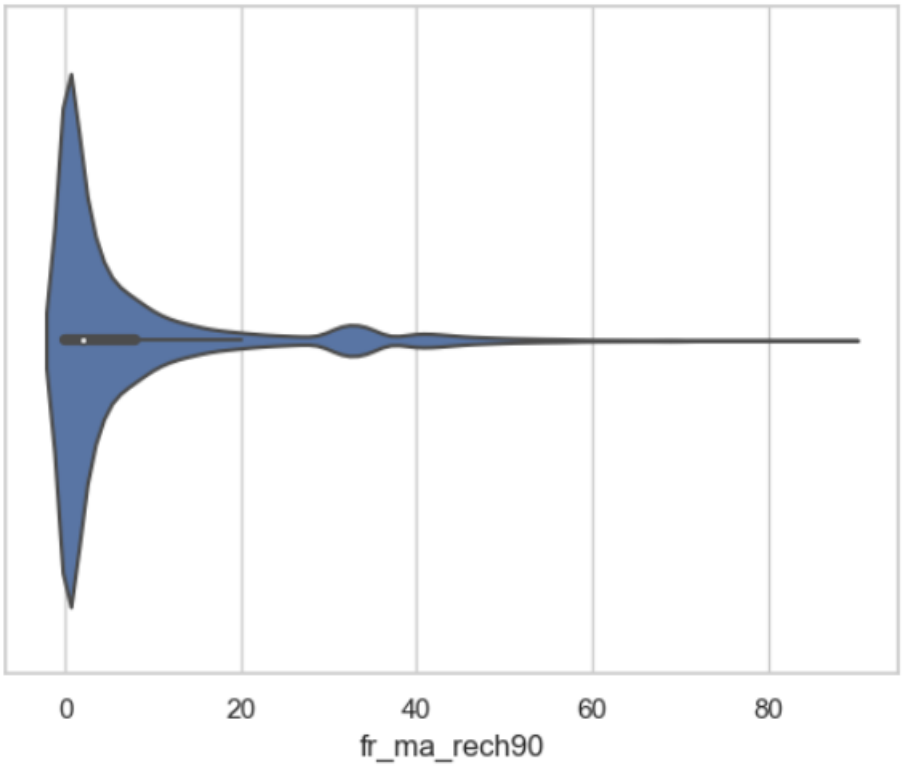
As we can see, median of main account balance just before recharge in last 30 days at user level for majority of the customers is 0.

Violin plot of cnt_ma_rech90:



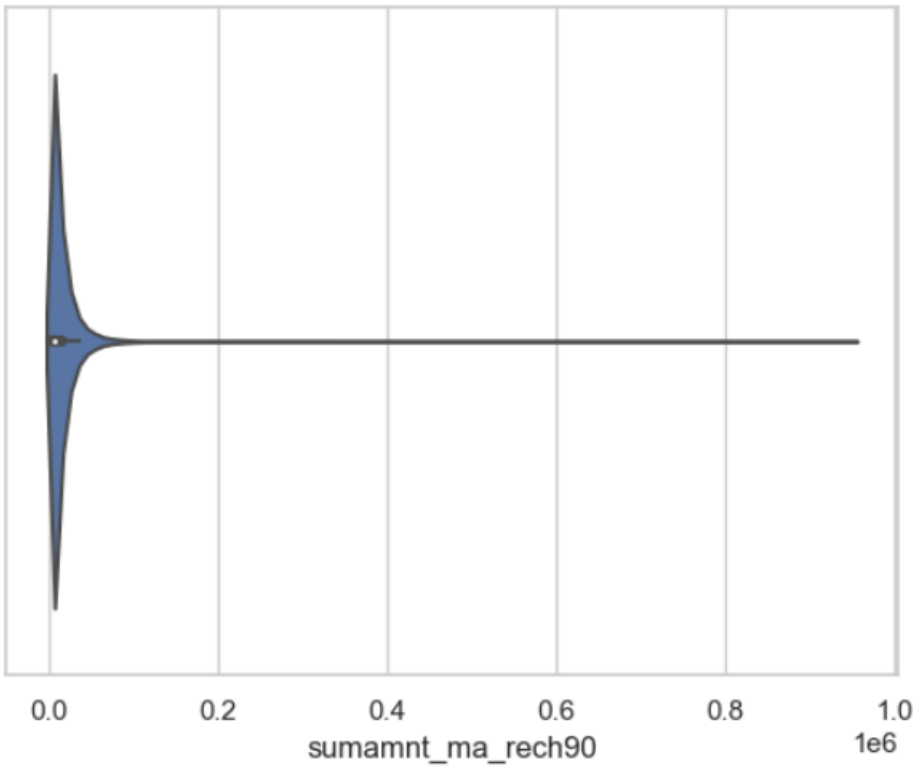
As we can see, number of times main account got recharged in last 90 days for majority of the customers is 1.

Violin plot of fr_ma_rech90:



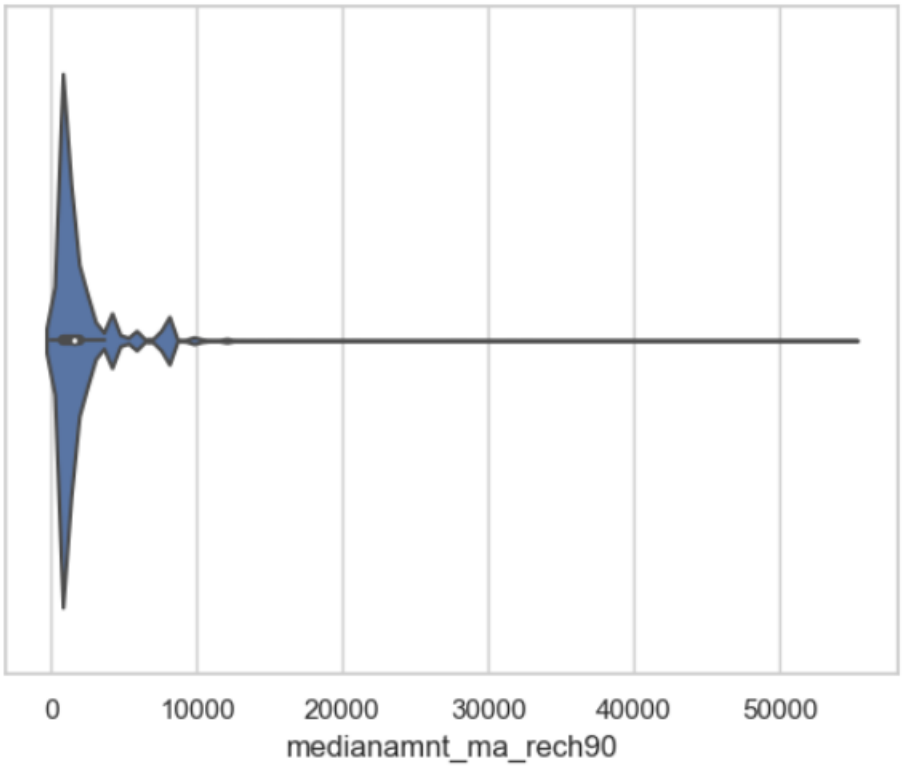
As we can see, frequency of main account recharged in last 90 days for majority of the customers is 0.

Violin plot of sumamnt_ma_rech90:



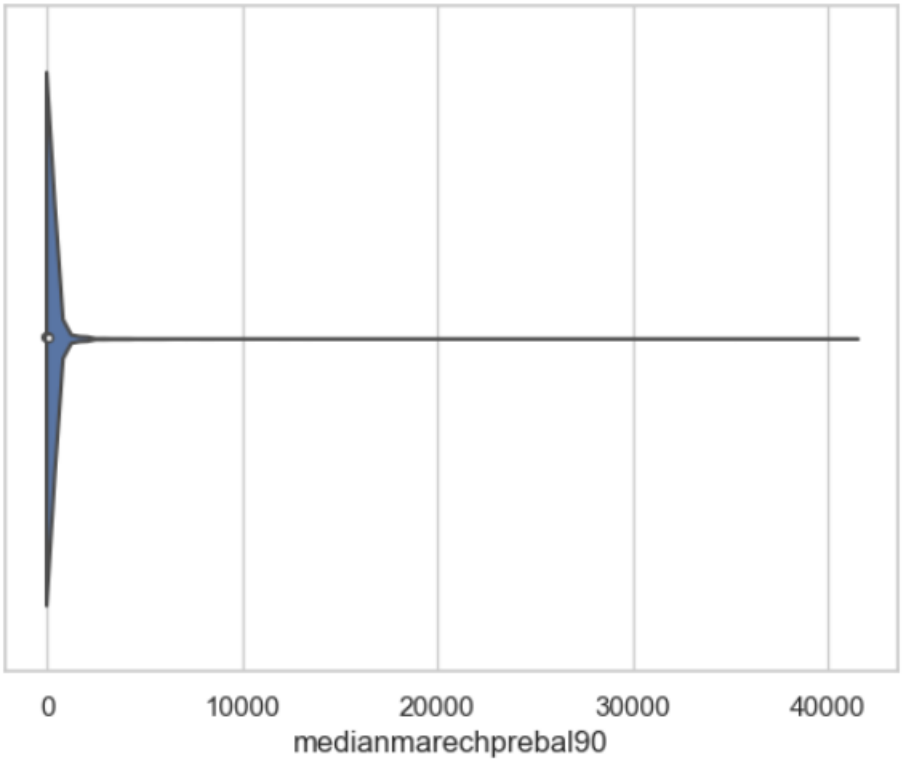
As we can see, total amount of recharge in main account over last 90 days for majority of the customers is 0.

Violin plot of medianamnt_ma_rech90:



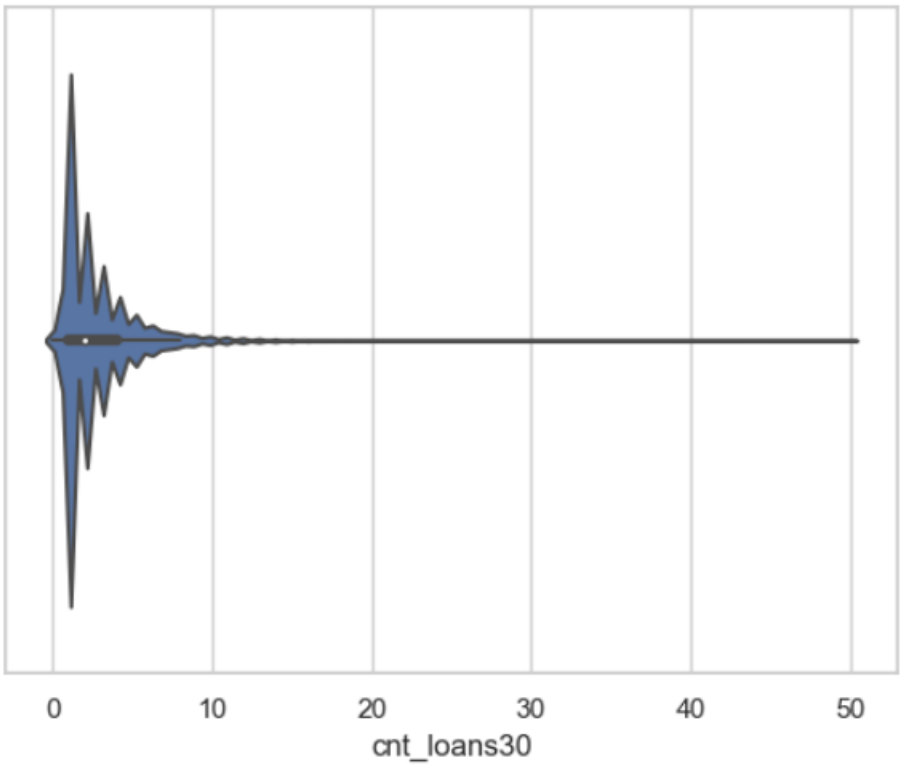
As we can see, median of amount of recharges done in main account over last 90 days at user level for majority of the customers is 1539.

Violin plot of medianmarechprebal90:



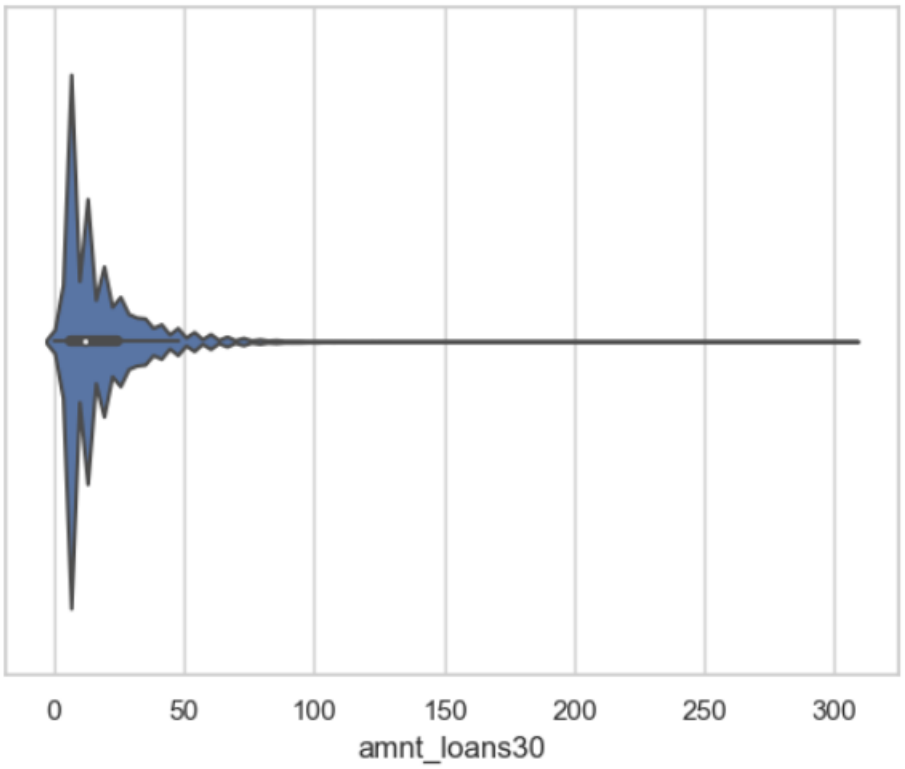
As we can see, median of main account balance just before recharge in last 90 days at user level for majority of the customers is 0.

Violin plot of cnt_loans30:



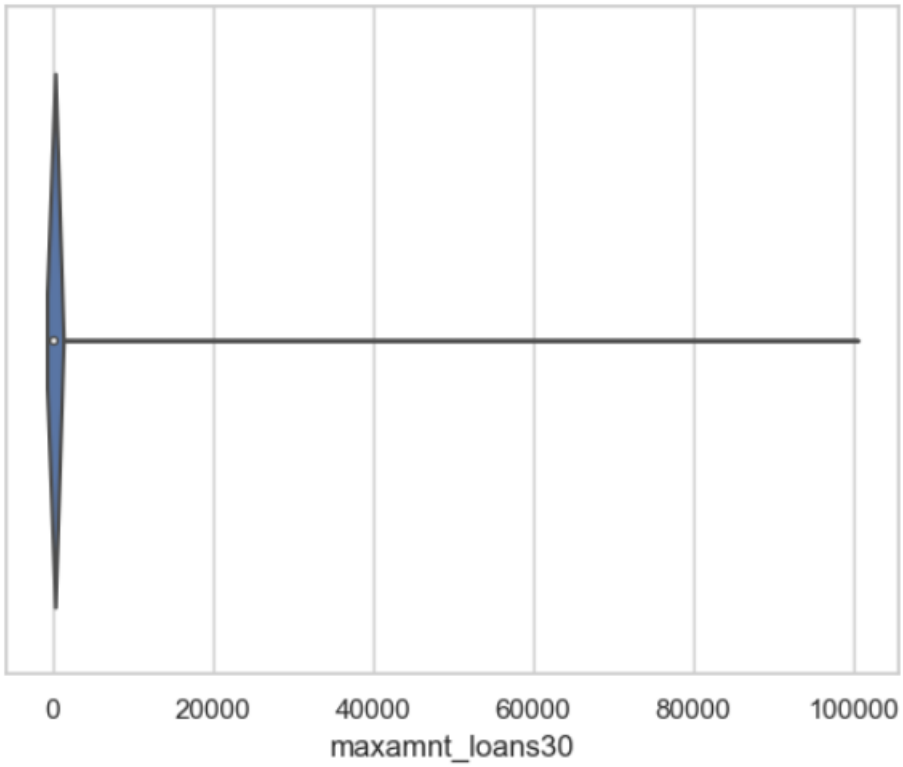
As we can see, number of loans taken by user in last 30 days for majority of the customers is 1.

Violin plot of amnt_loans30:



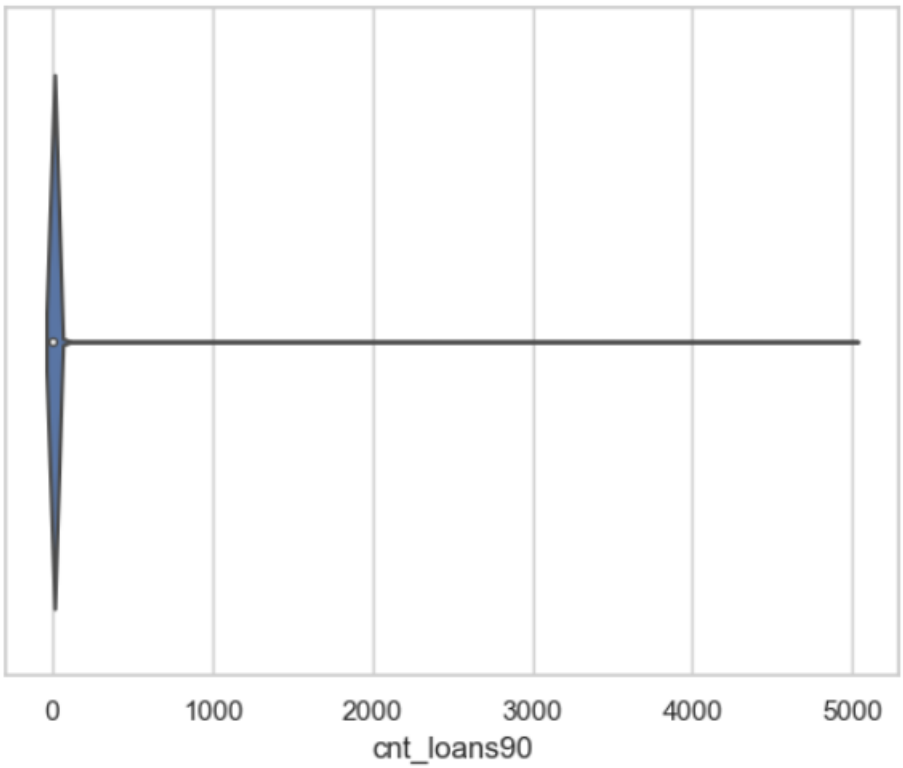
As we can see, total amount of loans taken by user in last 30 days for majority of the customers is 6.

Violin plot of maxamnt_loans30:



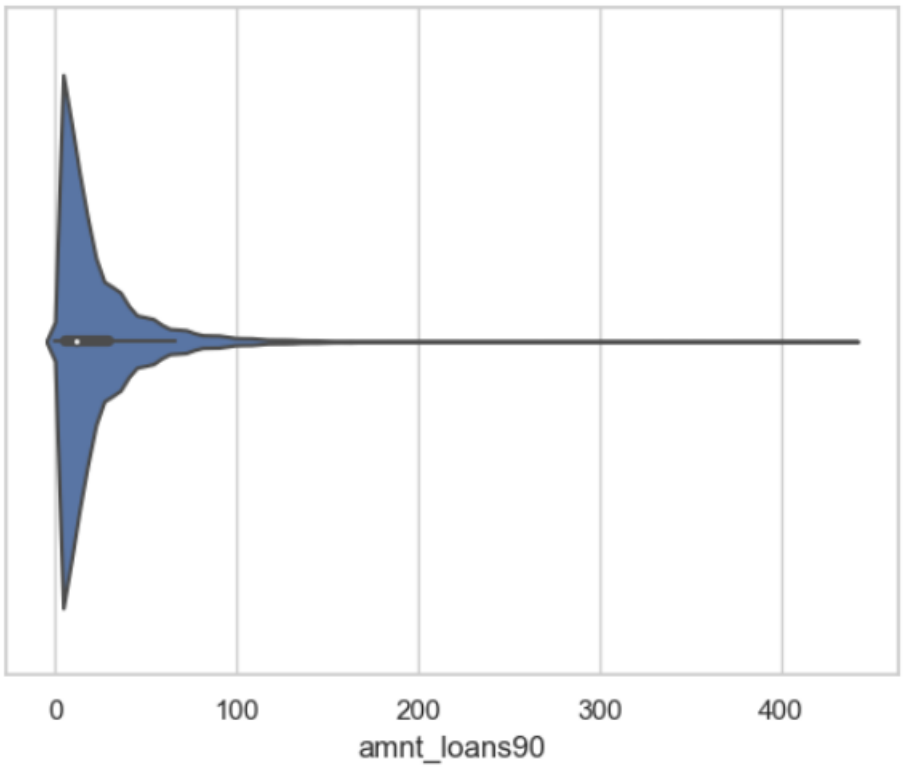
As we can see, maximum amount of loan taken by the user in last 30 days for majority of the customers is 6.

Violin plot of cnt_loans90:



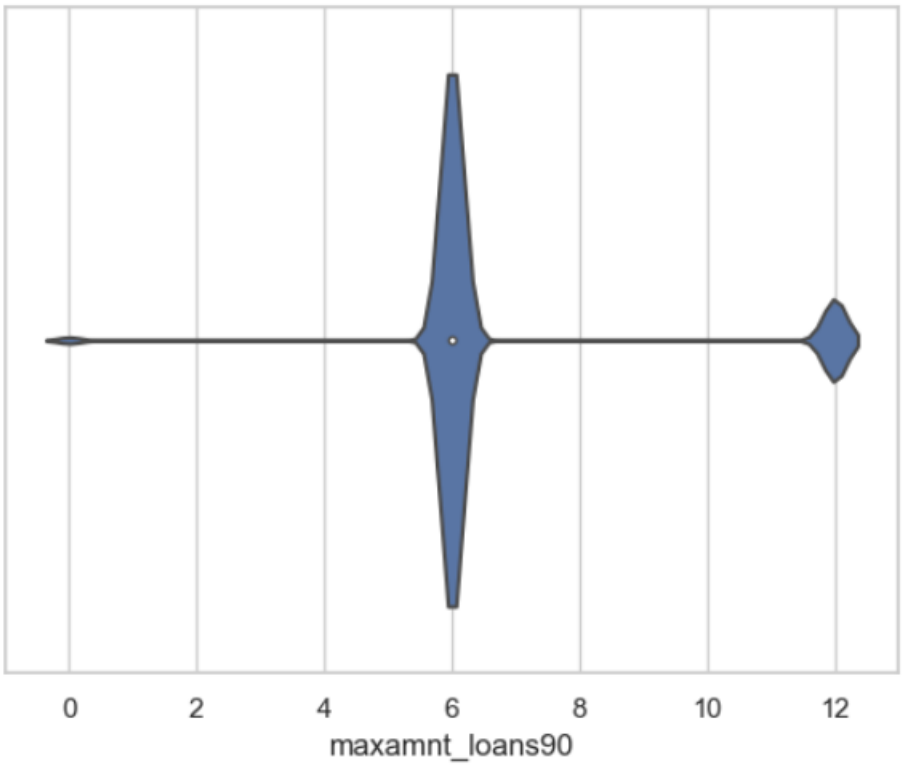
As we can see, number of loans taken by user in last 90 days for majority of the customers is 1.

Violin plot of amnt_loans90:



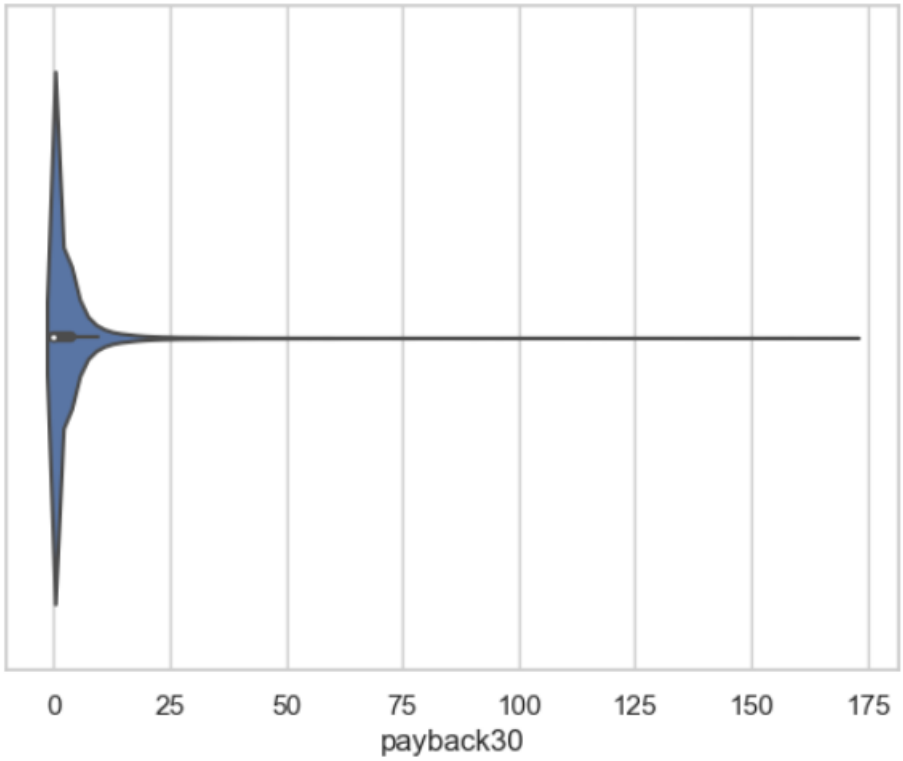
As we can see, total amount of loans taken by user in last 90 days for majority of the customers is 6.

Violin plot of maxamnt_loans90 :



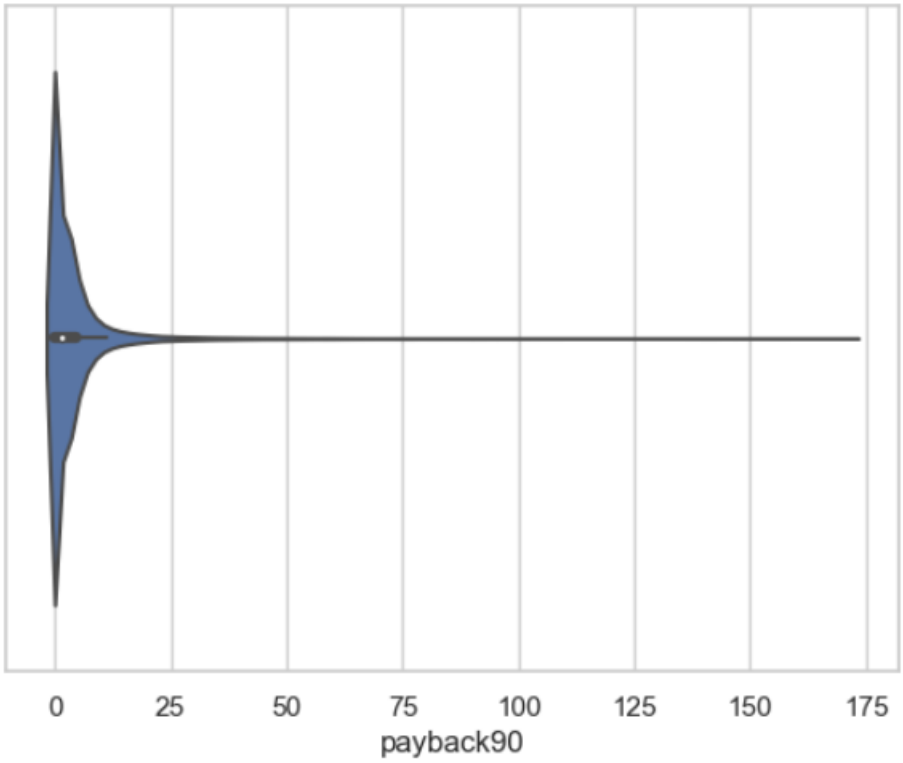
As we can see, maximum amount of loan taken by the user in last 90 days for majority of the customers is 6.

Violin plot of payback30:



As we can see, average payback time in days over last 30 days for majority of the customers is 0.

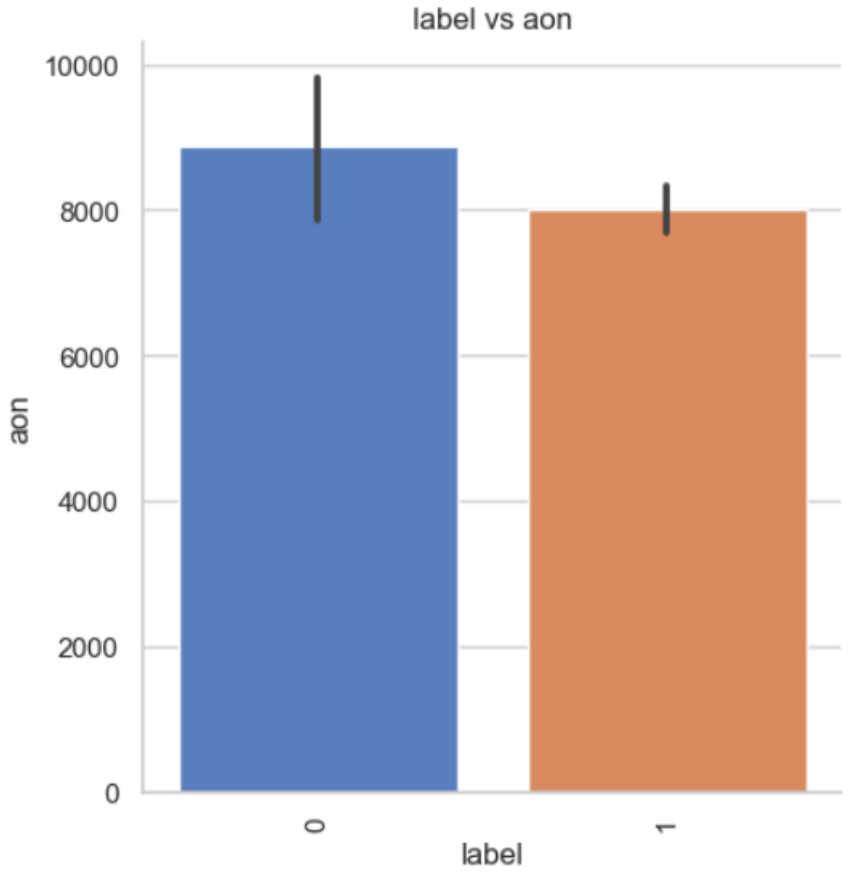
Violin plot of payback90:



As we can see, average payback time in days over last 90 days for majority of the customers is 0.

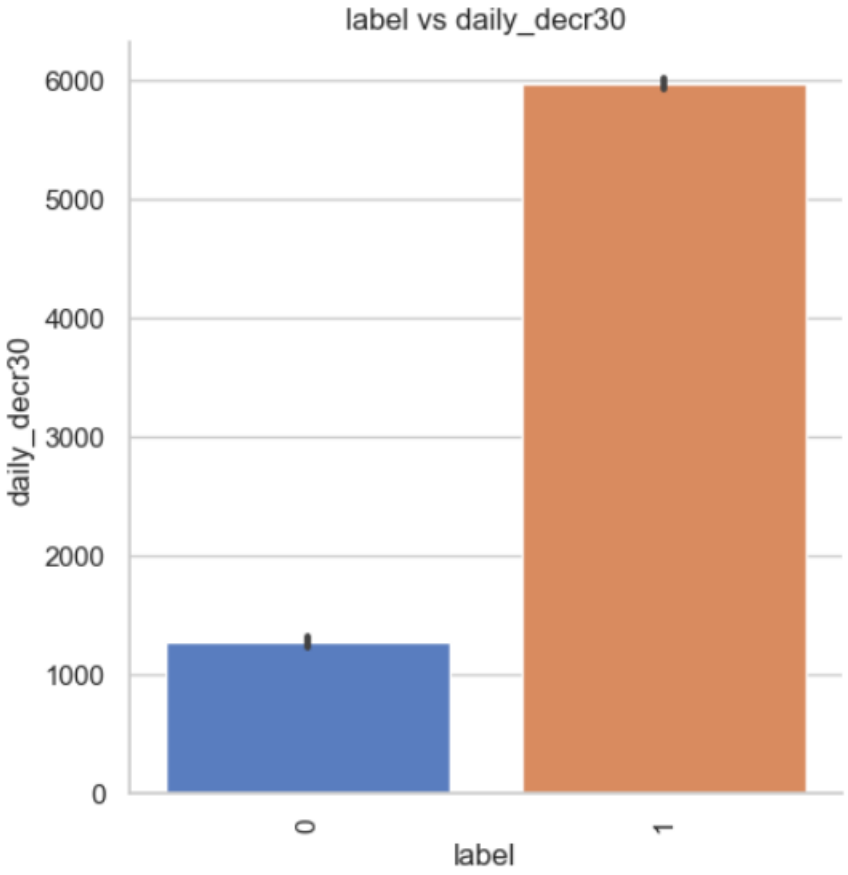
Bivariate Analysis:

Factor plot of label vs aon:



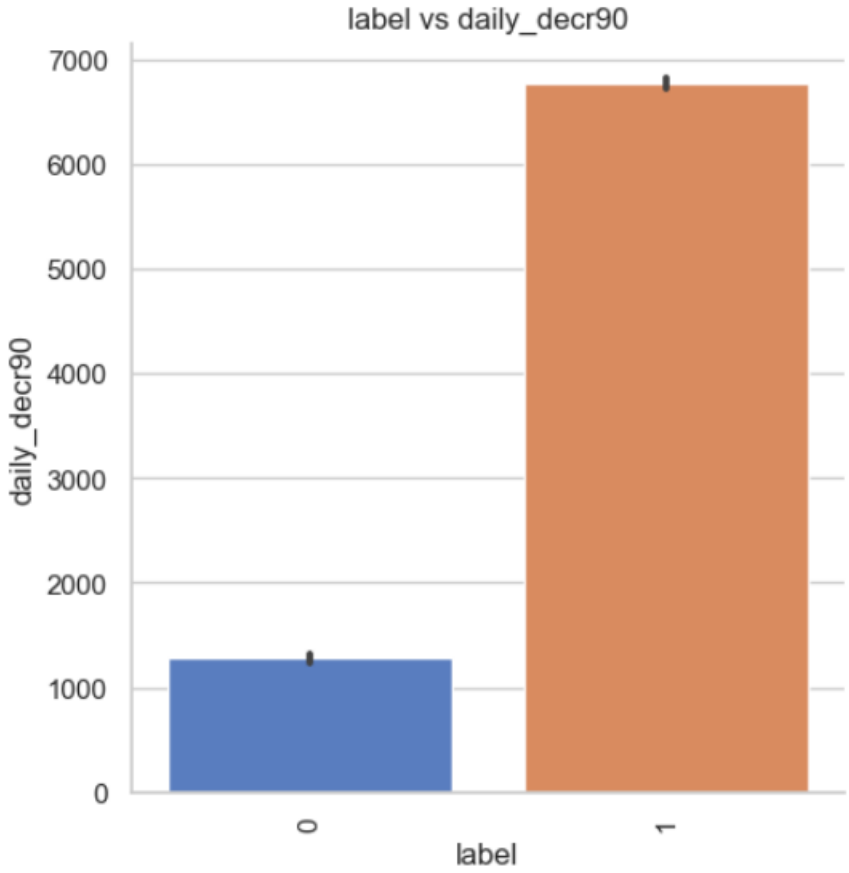
As the age on cellular network increases, the chances of default increases.

Factor plot of label vs daily_decr30:



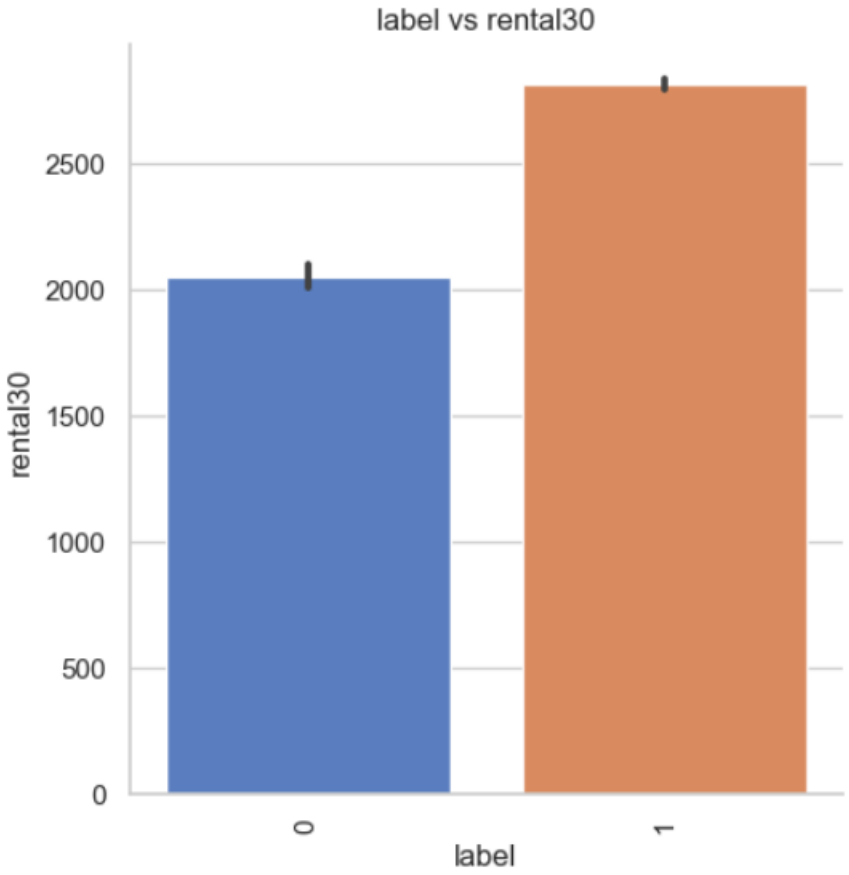
As the daily amount spent from main account, averaged over last 30 days increases, the chances of default decreases.

Factor plot of label vs daily_decr90:



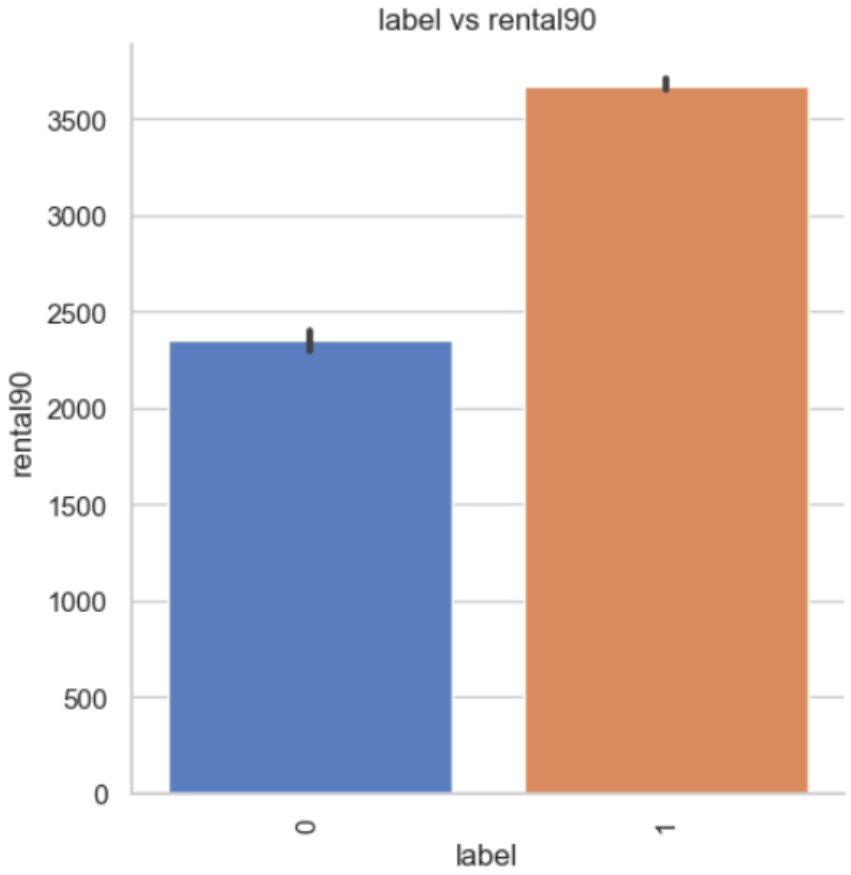
As the daily amount spent from main account, averaged over last 90 days increases, the chances of default decreases.

Factor plot of label vs rental30:



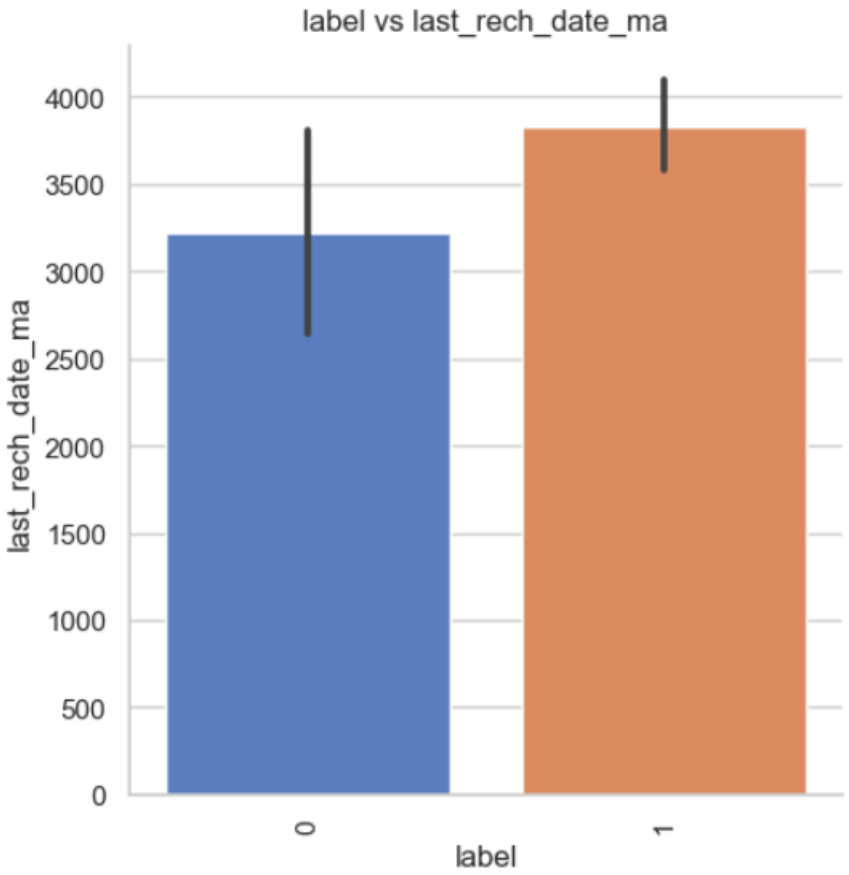
As the average main account balance over last 30 days increases, the chances of default decreases.

Factor plot of label vs rental90:



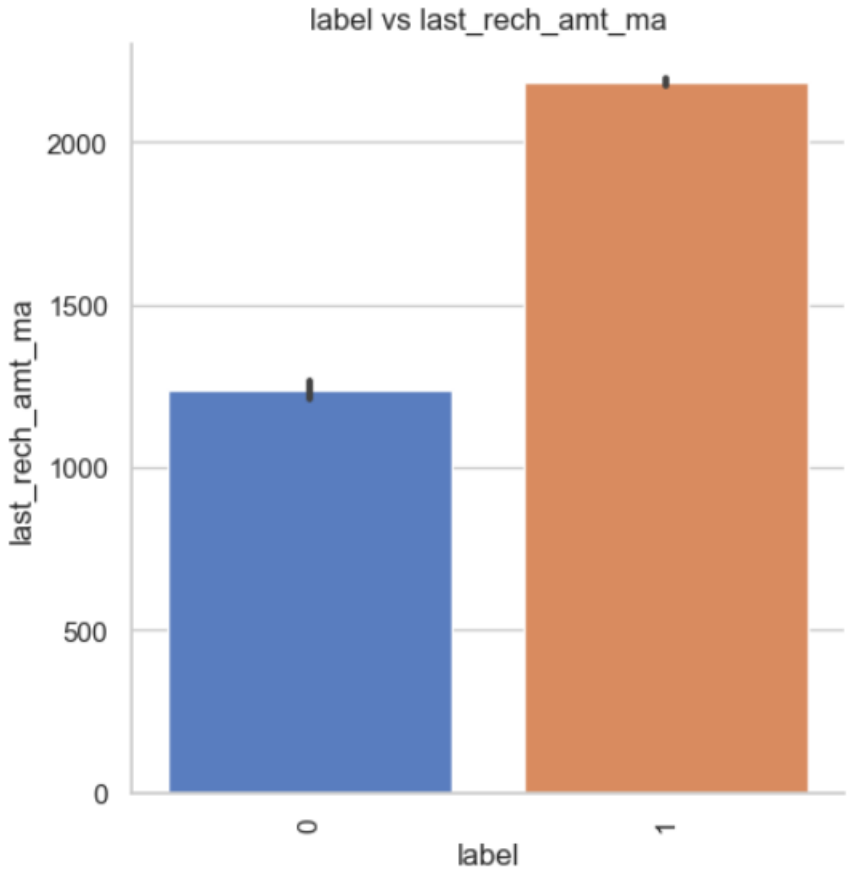
Average main account balance over last 90 days increases, the chances of default decreases.

Factor plot of label vs last_rech_date_ma:



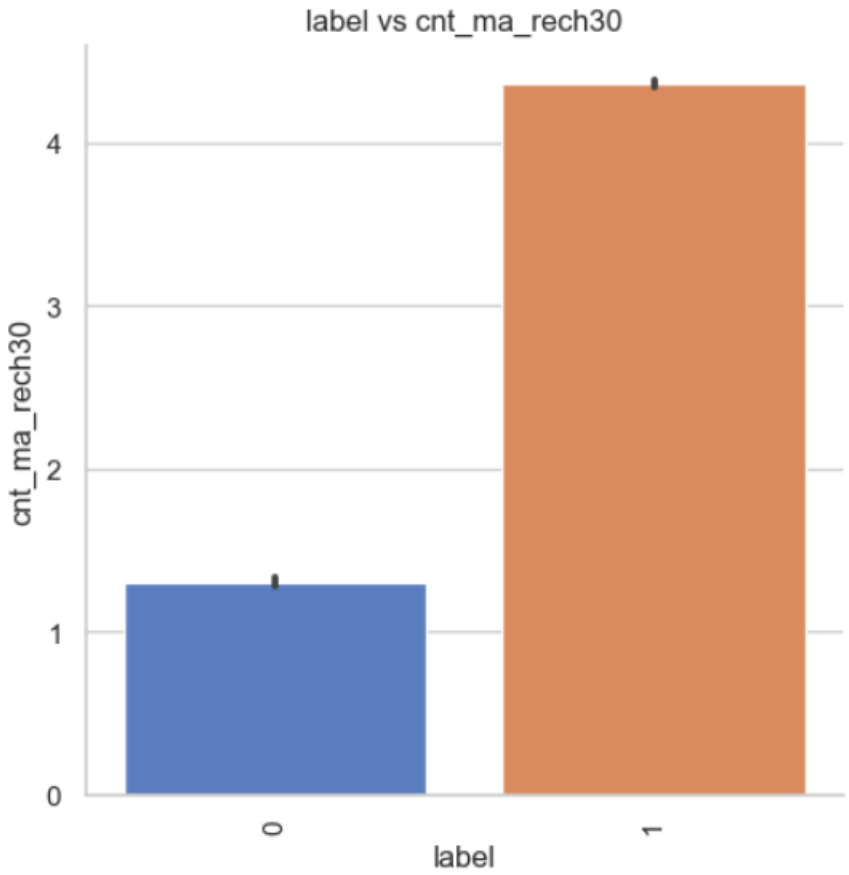
As the number of days till last recharge of main account increases, the chances of default decreases.

Factor plot of label vs last_rech_amt_ma:



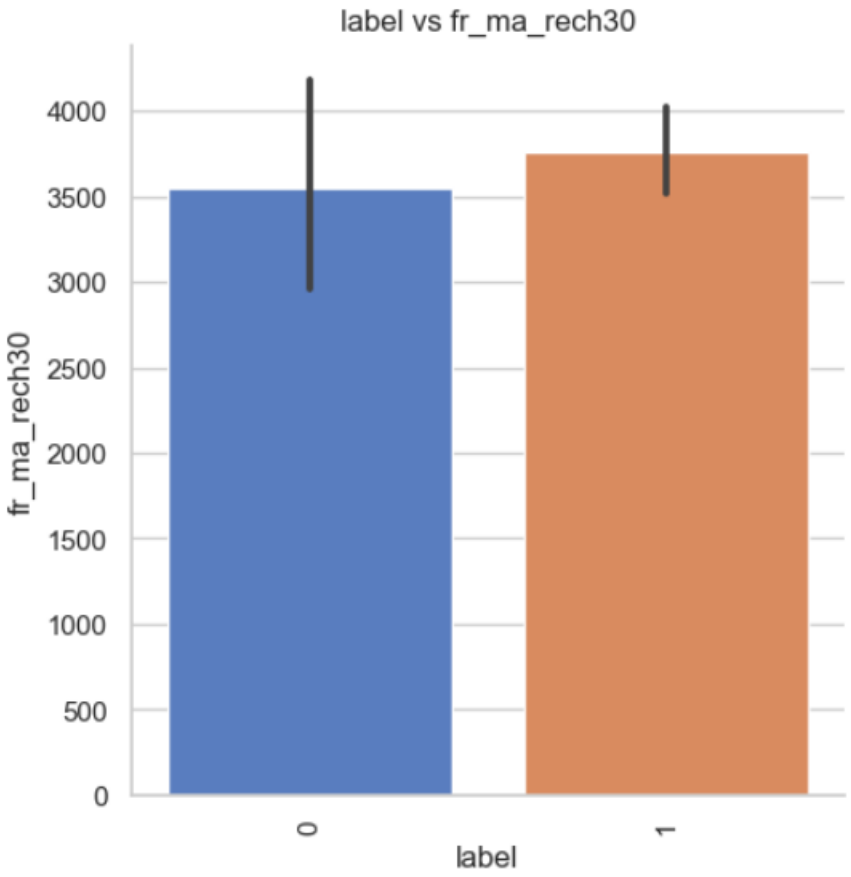
As the amount of last recharge of main account increases, the chances of default decreases.

Factor plot of label vs cnt_ma_rech30:



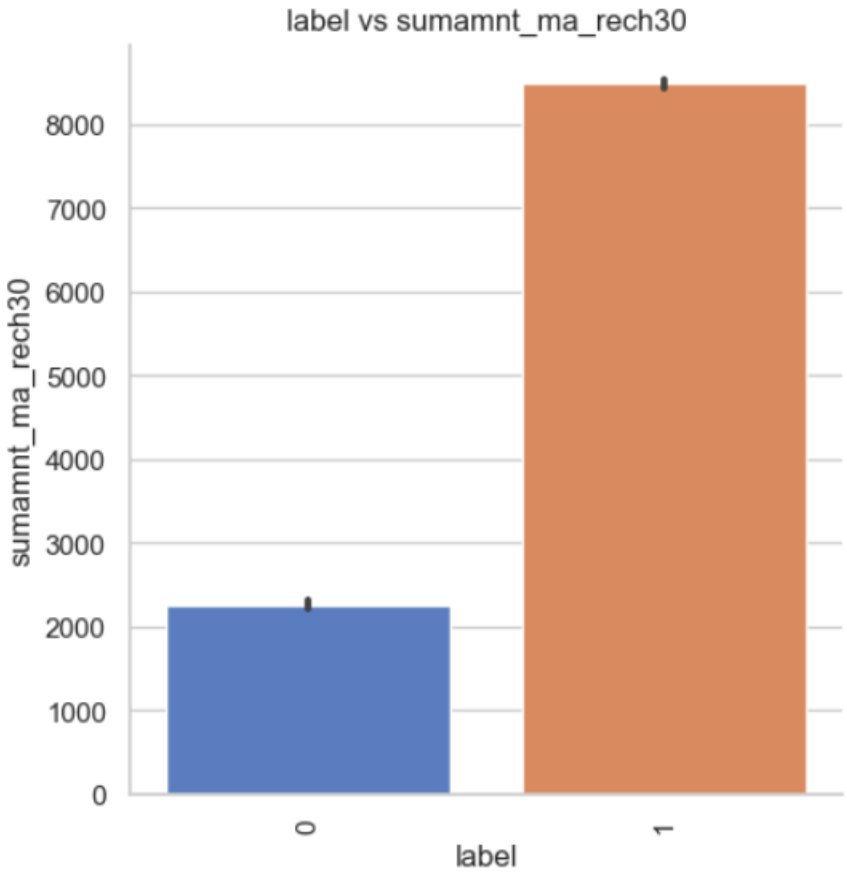
As the number of times main account got recharged in last 30 days increases, the chances of default decreases.

Factor plot of label vs fr_ma_rech30:



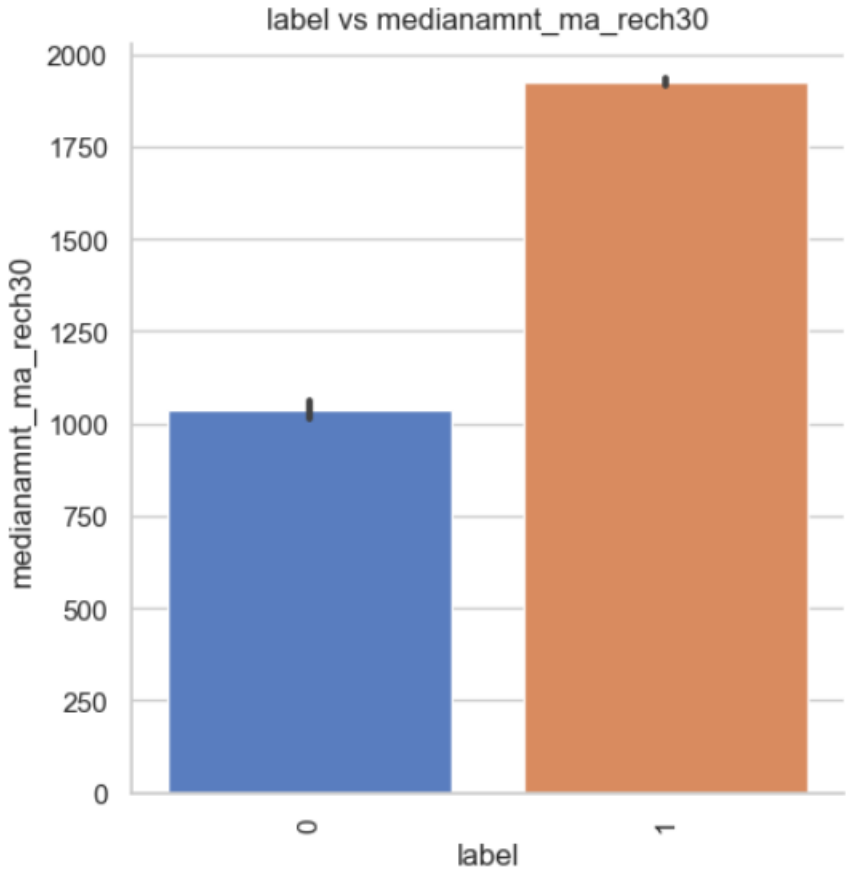
As the frequency of main account recharged in last 30 days increases, the chances of default decreases.

Factor plot of label vs sumamnt_ma_rech30:



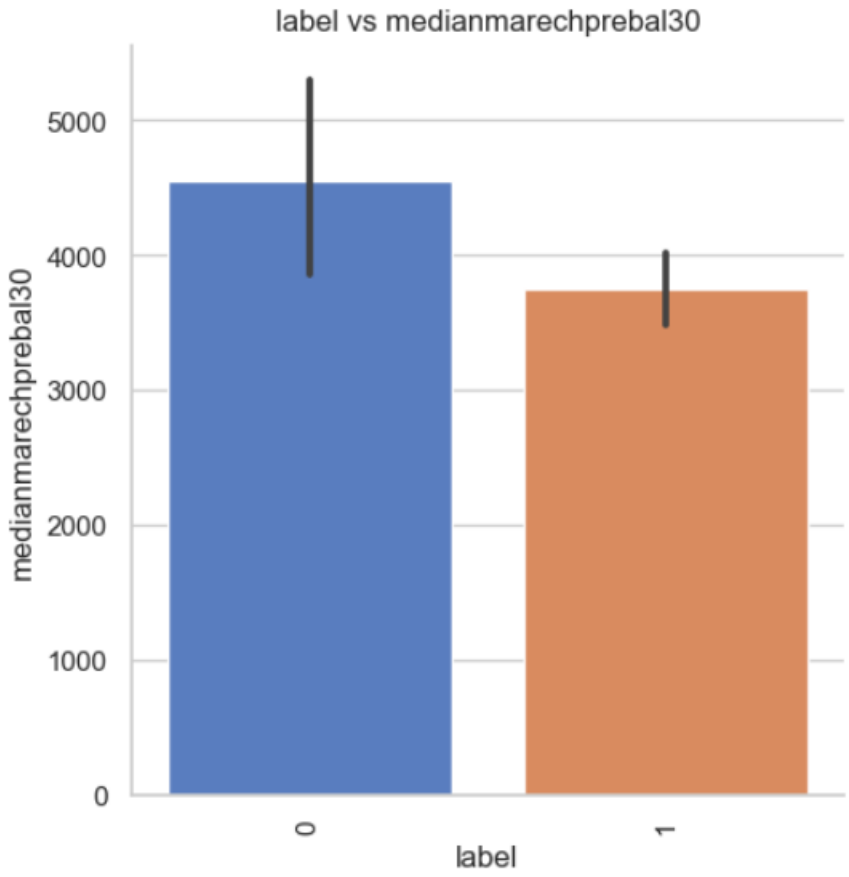
As the total amount of recharge in main account over last 30 days increases, the chances of default decreases.

Factor plot of label vs medianamnt_ma_rech30:



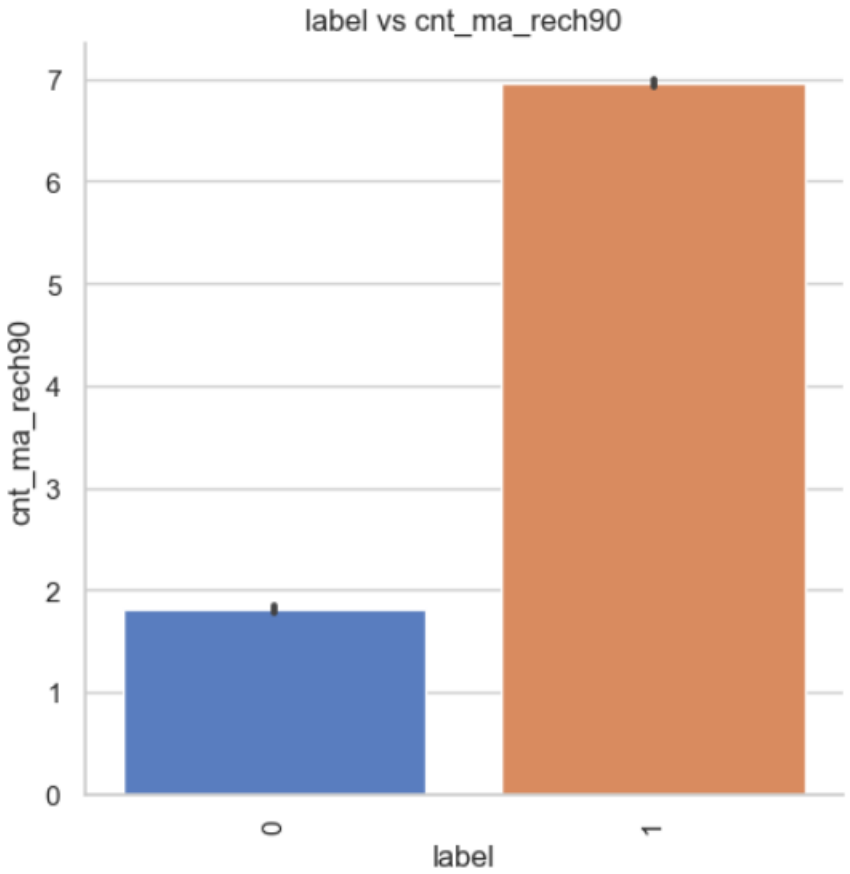
As the median of amount of recharges done in main account over last 30 days at user level increases, the chances of default decreases.

Factor plot of label vs medianmarechprebal30:



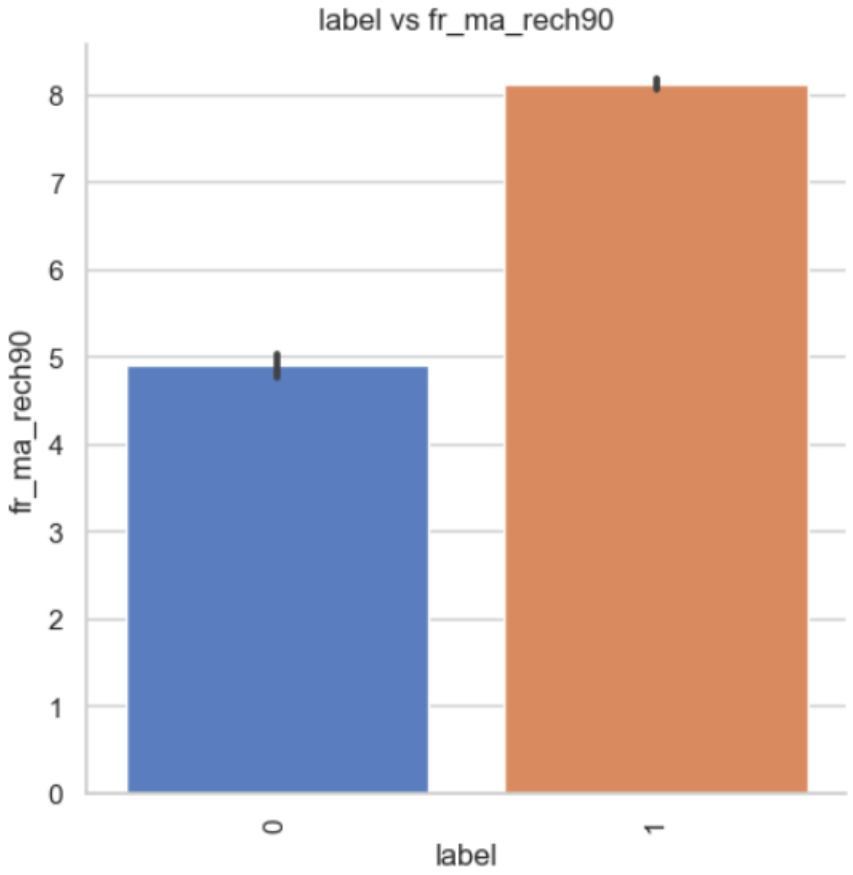
As the median of main account balance just before recharge in last 30 days at user level increases, the chances of default increases.

Factor plot of label vs cnt_ma_rech90:



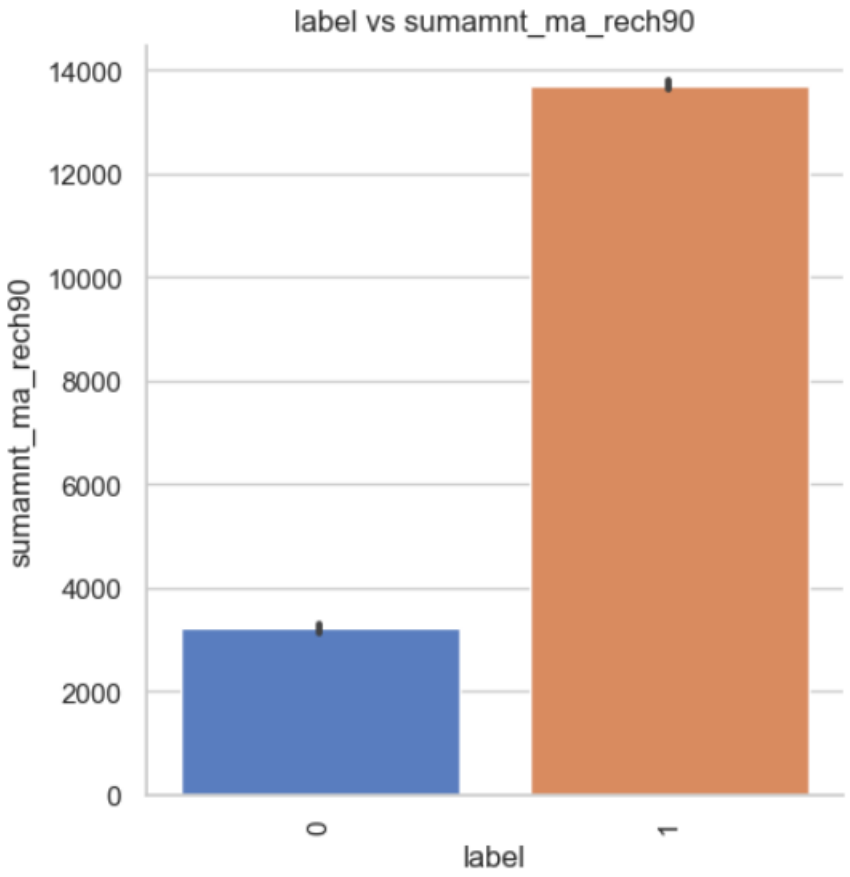
As the number of times main account got recharged in last 90 days increases, the chances of default decreases.

Factor plot of label vs fr_ma_rech90:



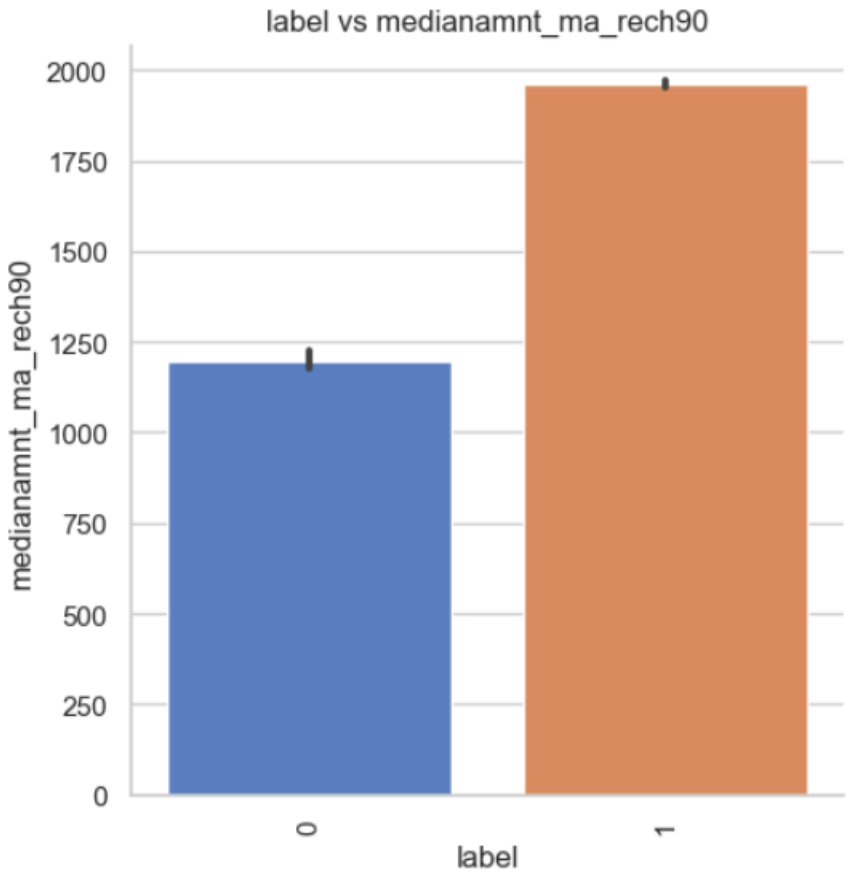
As the frequency of main account recharged in last 90 days increases, the chances of default decreases.

Factor plot of label vs sumamnt_ma_rech90:



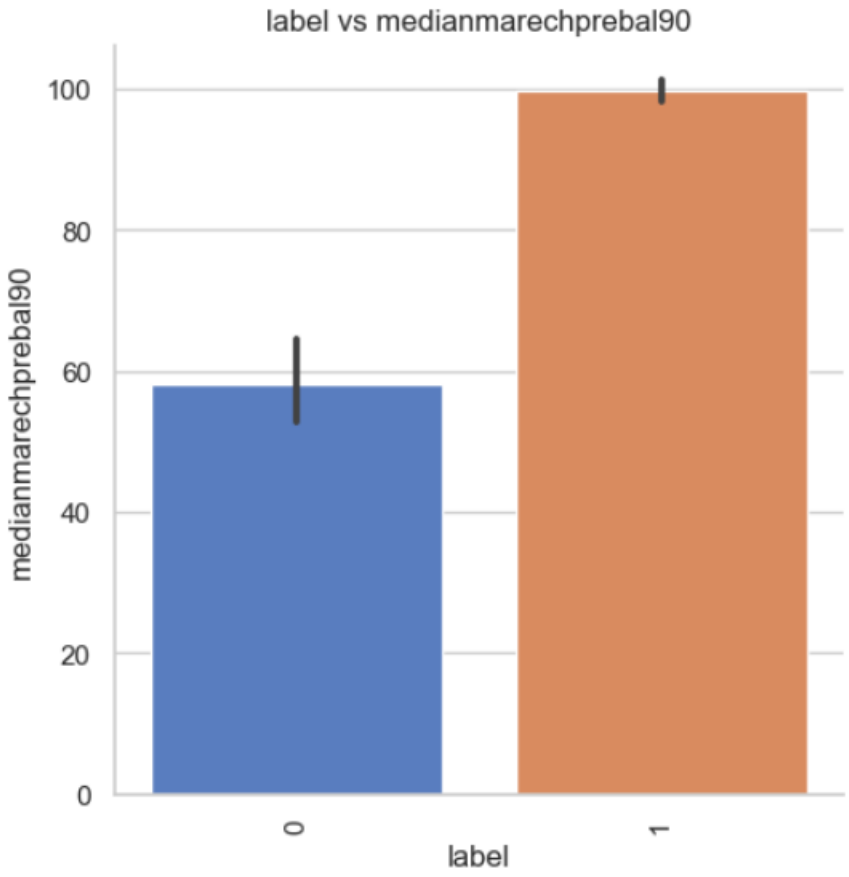
As the total amount of recharge in main account over last 90 days increases, the chances of default decreases.

Factor plot of label vs medianamnt_ma_rech90:



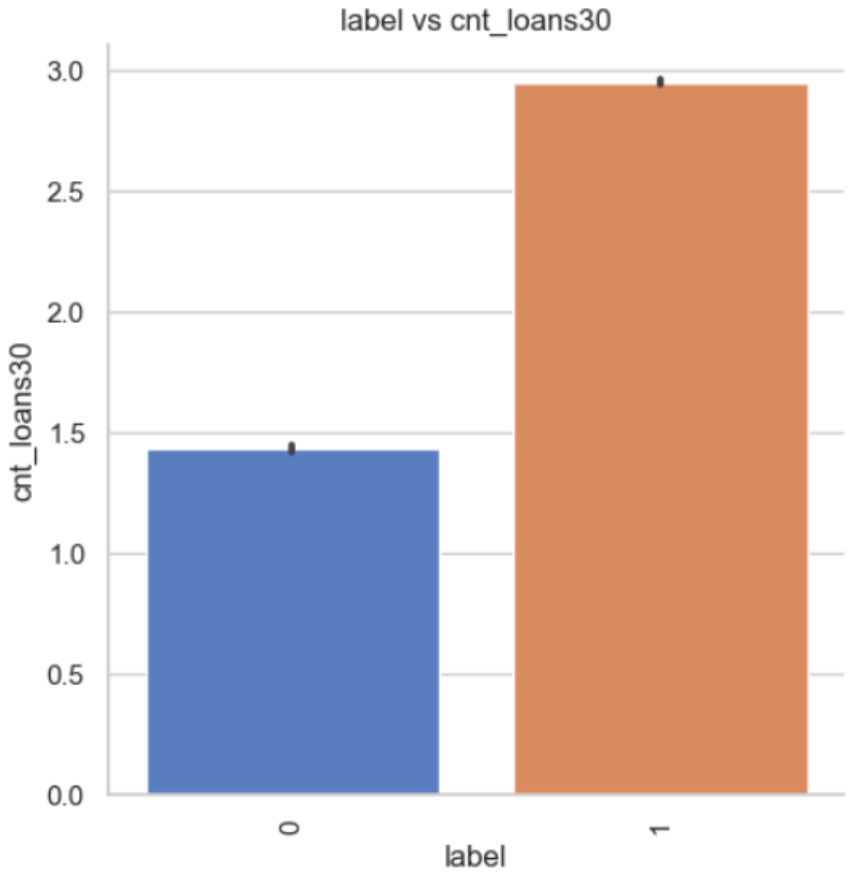
As the median of amount of recharges done in main account over last 90 days at user level increases, the chances of default decreases.

Factor plot of label vs medianmarechprebal90:



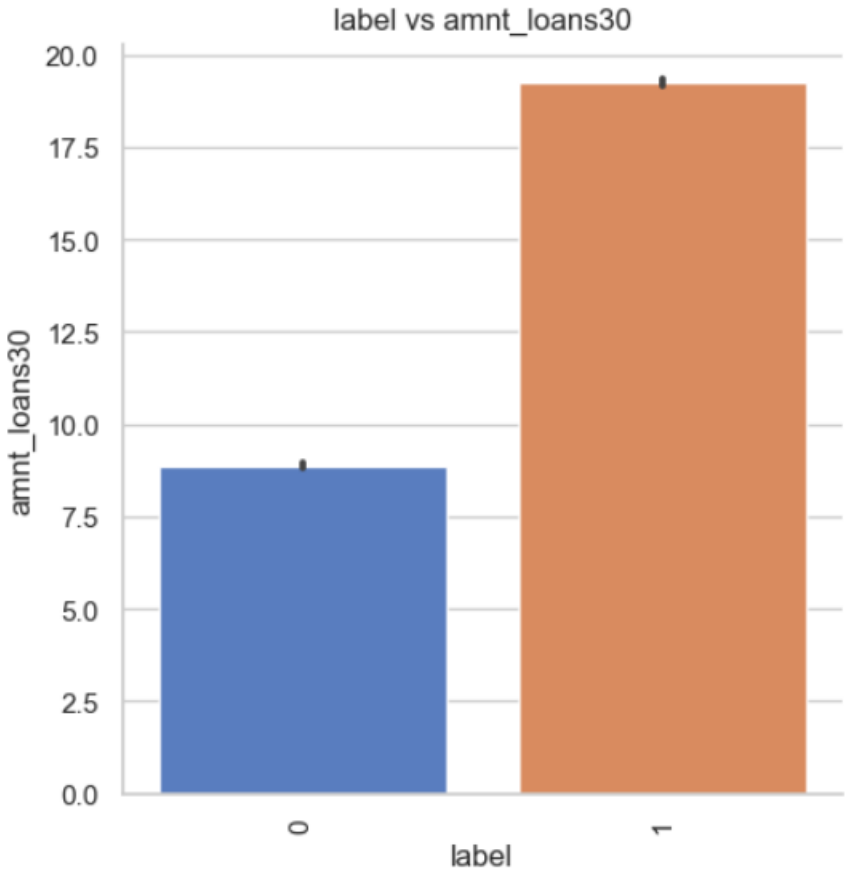
As the median of main account balance just before recharge in last 90 days at user level increases, the chances of default decreases.

Factor plot of label vs cnt_loans30:



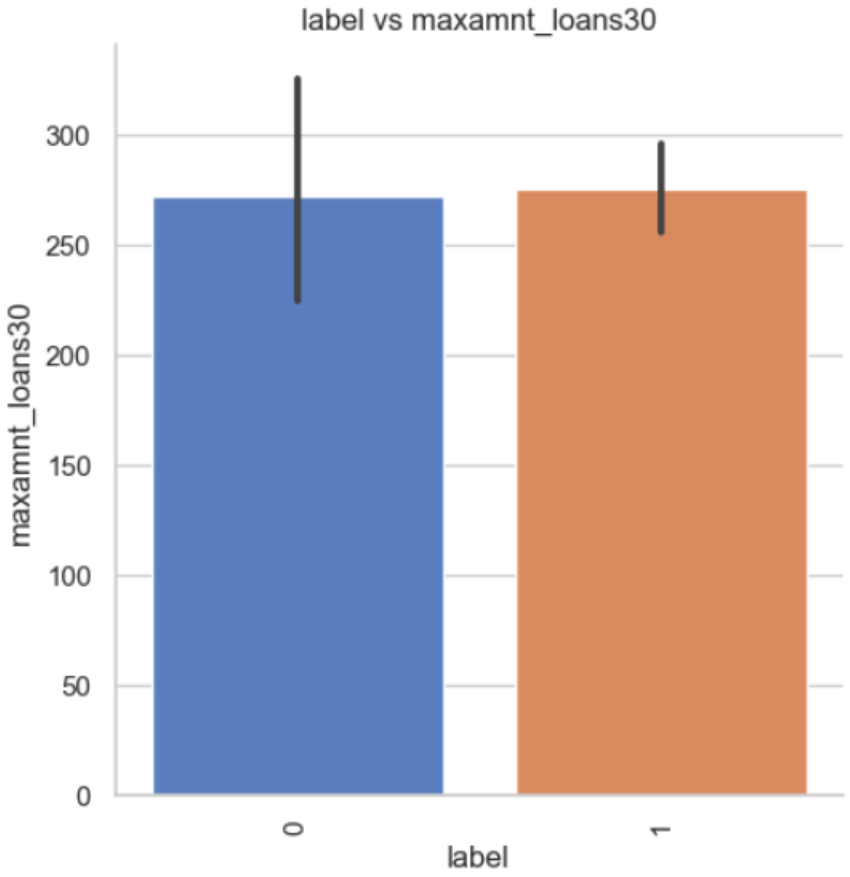
As the number of loans taken by user in last 30 days increases, the chances of default decreases.

Factor plot of label vs amnt_loans30:



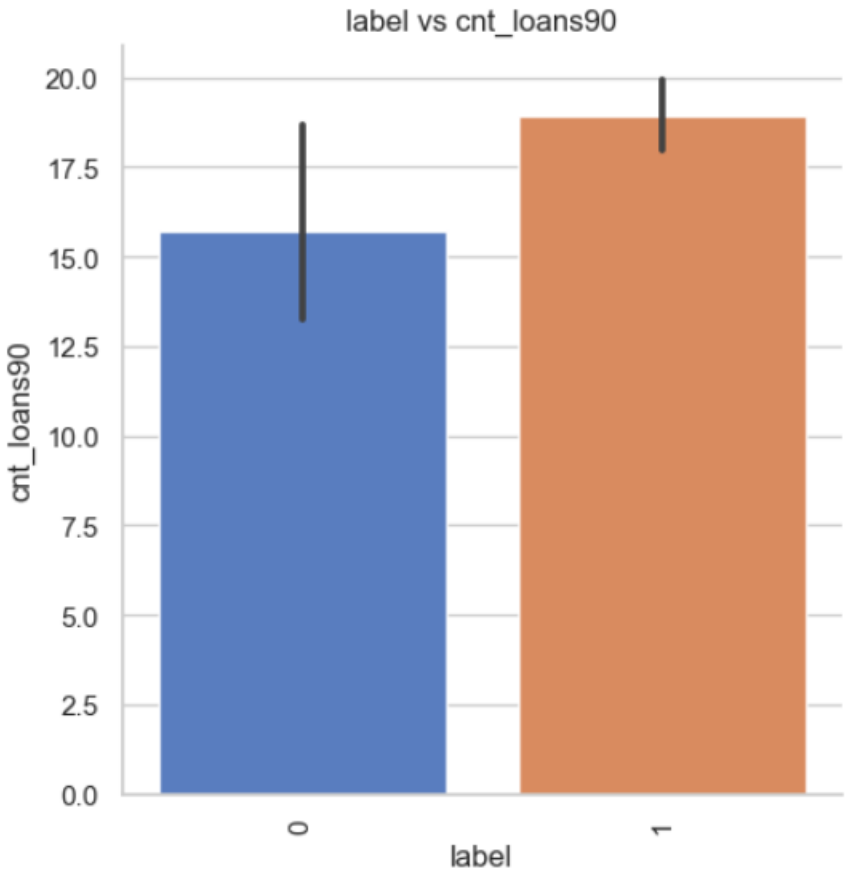
As the total amount of loans taken by user in last 30 days increases, the chances of default decreases.

Factor plot of label vs maxamnt_loans30:



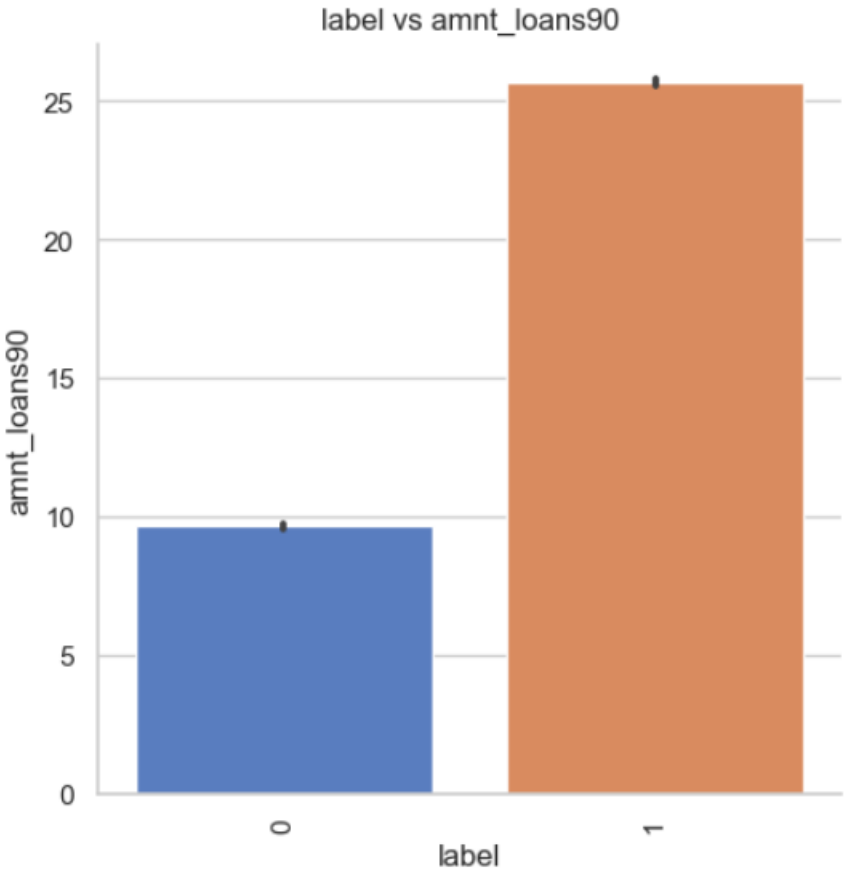
As the maximum amount of loan taken by the user in last 30 days increases, the chances of default decreases slightly.

Factor plot of label vs cnt_loans90:



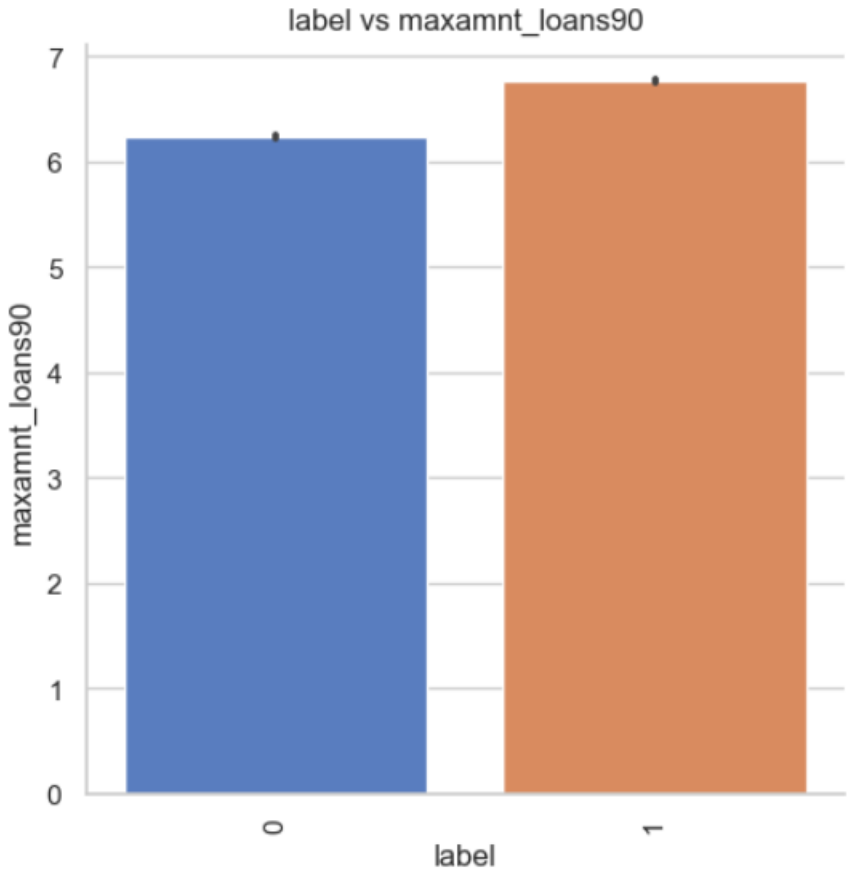
As the number of loans taken by user in last 90 days increases, the chances of default decreases.

Factor plot of label vs amnt_loans90:



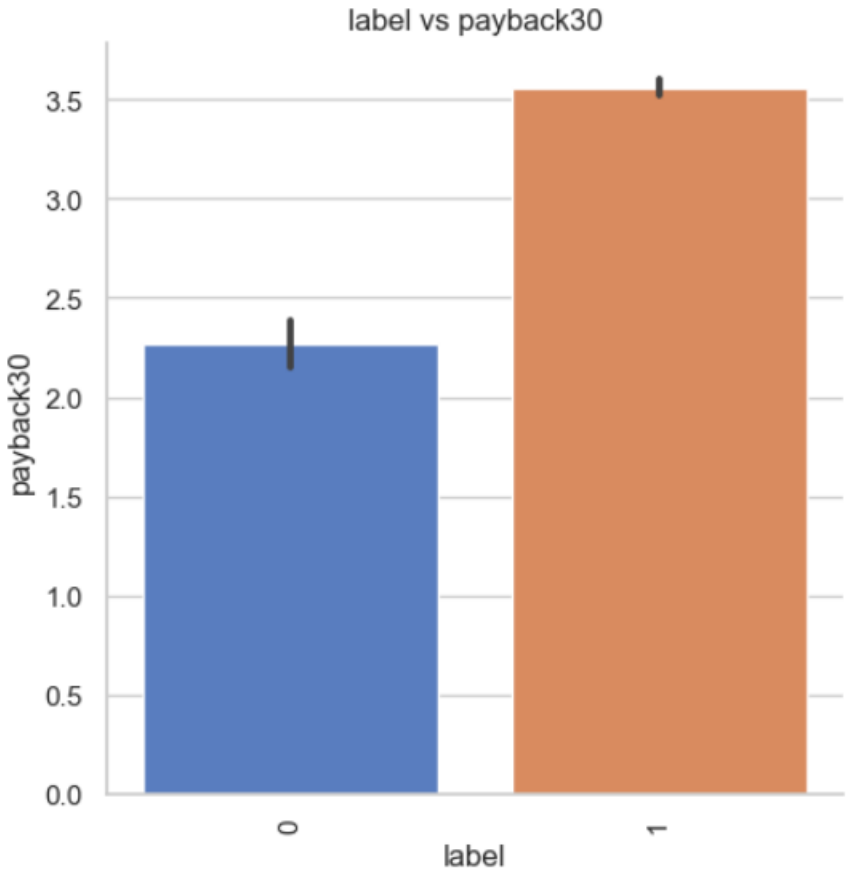
As the total amount of loans taken by user in last 90 days increases, the chances of default decreases.

Factor plot of label vs maxamnt_loans90:



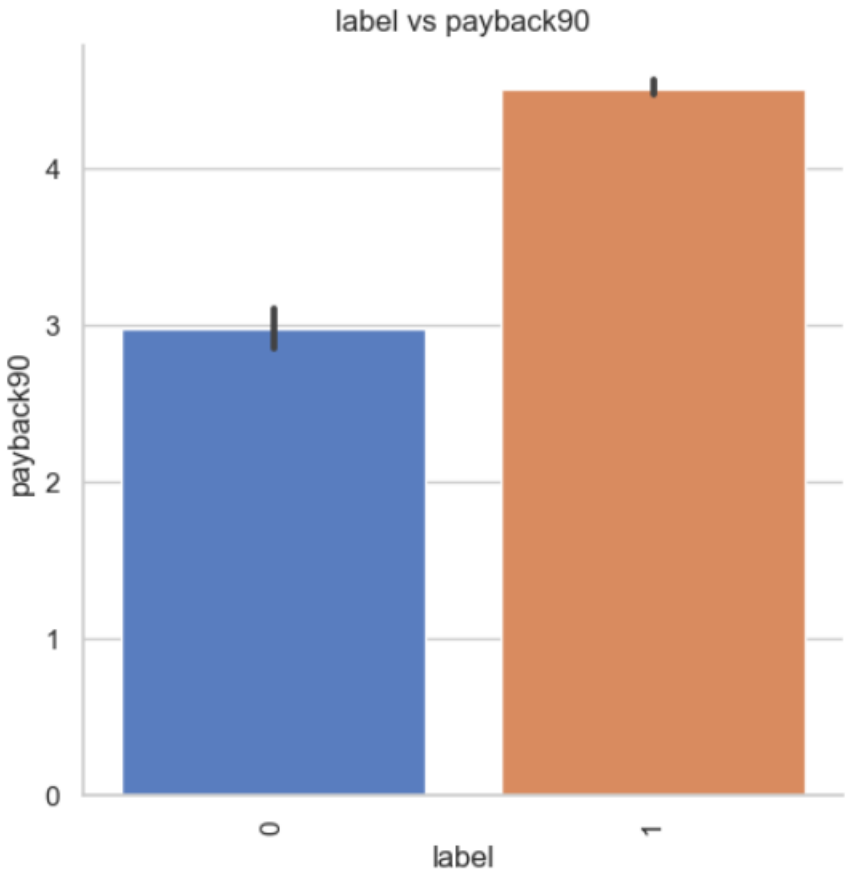
As the maximum amount of loan taken by the user in last 90 days increases, the chances of default decreases.

Factor plot of label vs payback30:



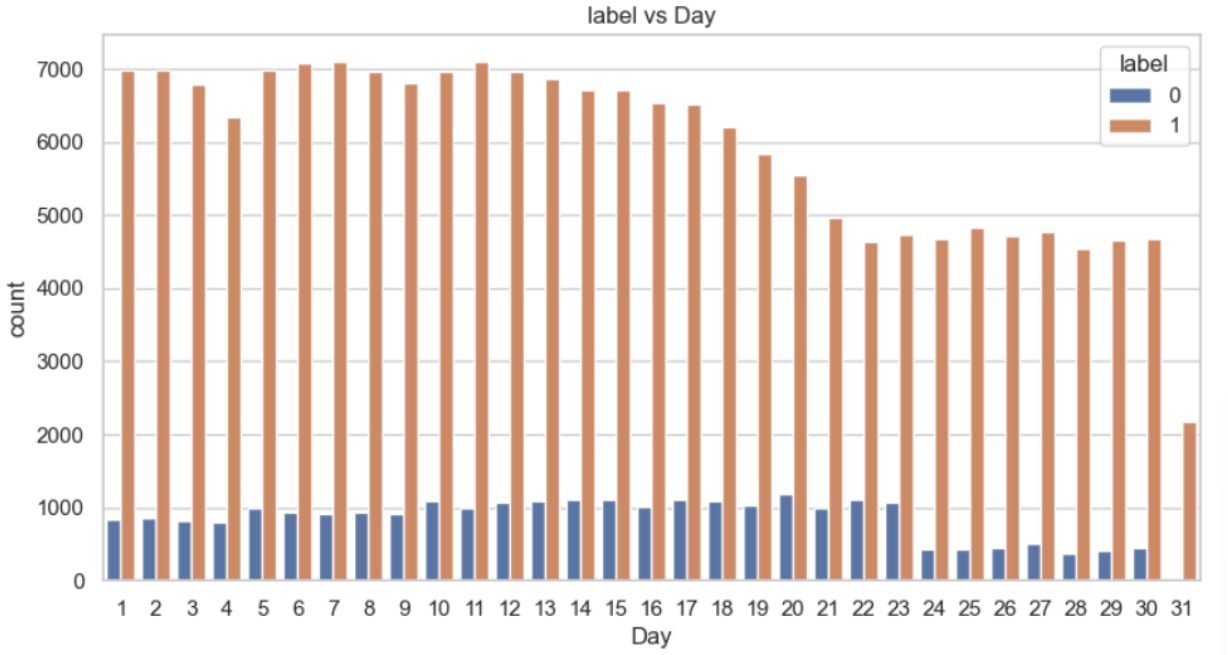
As the average payback time in days over last 30 days increases, the chances of default decreases.

Factor plot of label vs payback90:



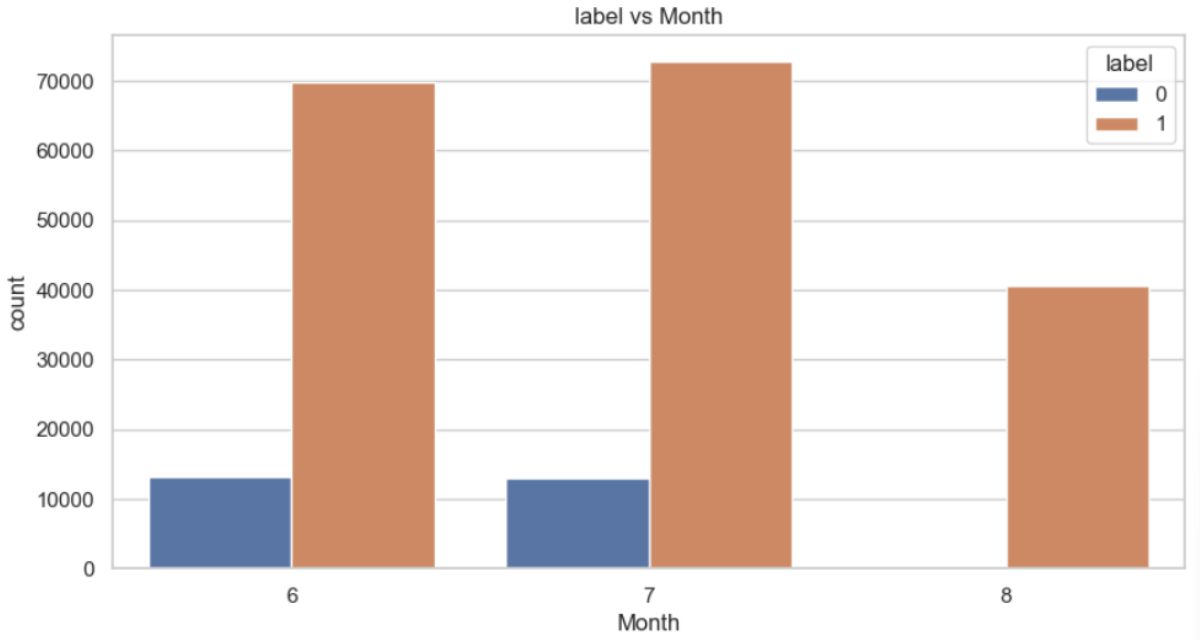
As the average payback time in days over last 90 days increases, the chances of default decreases.

Count plot of label vs Day:



As we can see, no significant observation is found from the above plot.

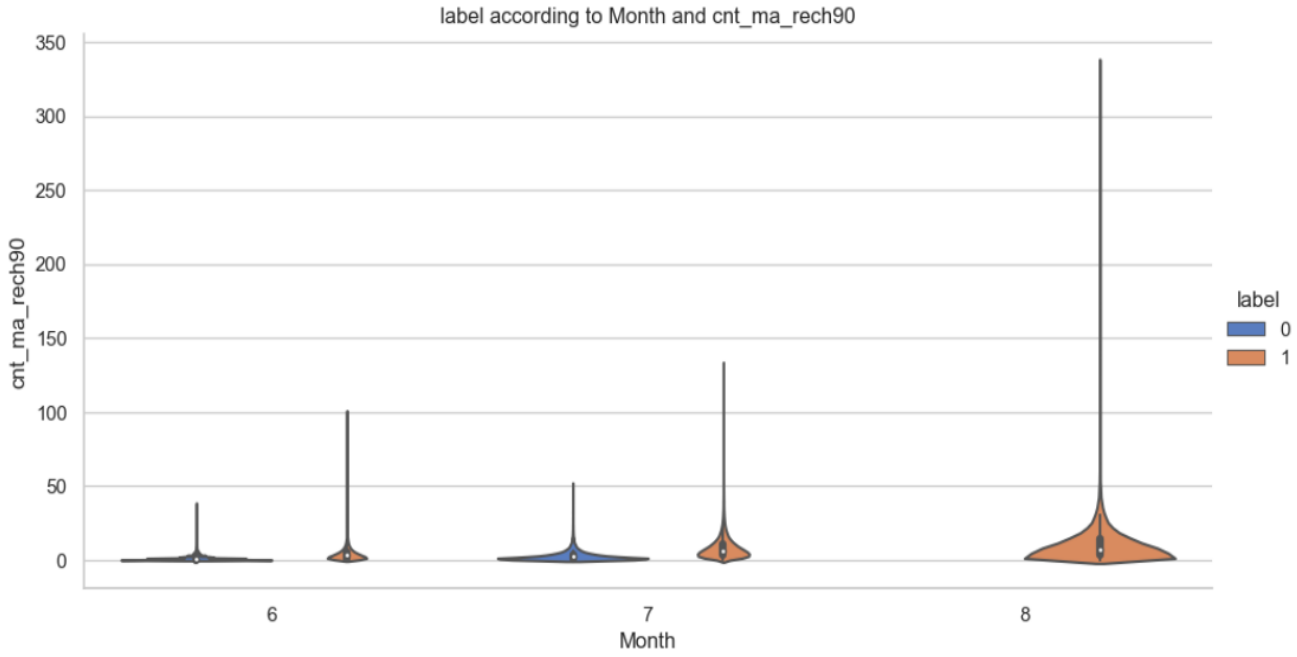
Count plot of label vs Month:



As we can see, the chances of default decreases, if the loan is taken in the 7th month. Also, there are defaulters in the 8th month.

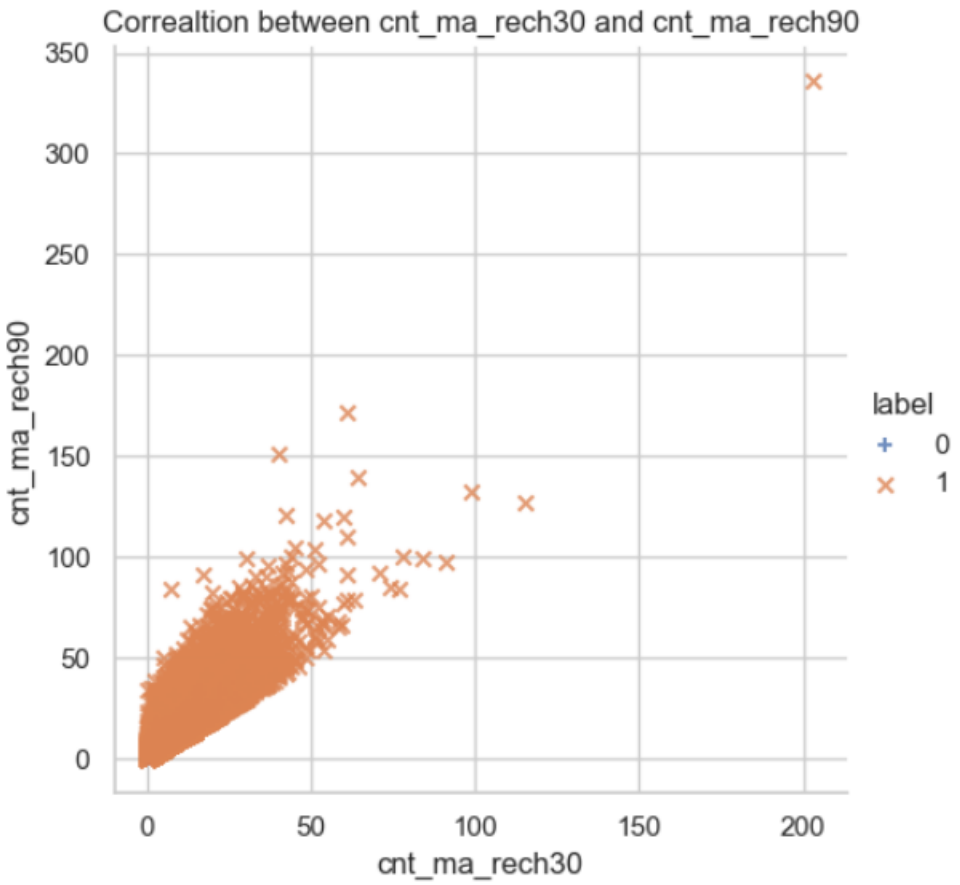
Multivariate Analysis:

Checking 'Month' and 'cnt_ma_rech90' with respect to 'label':



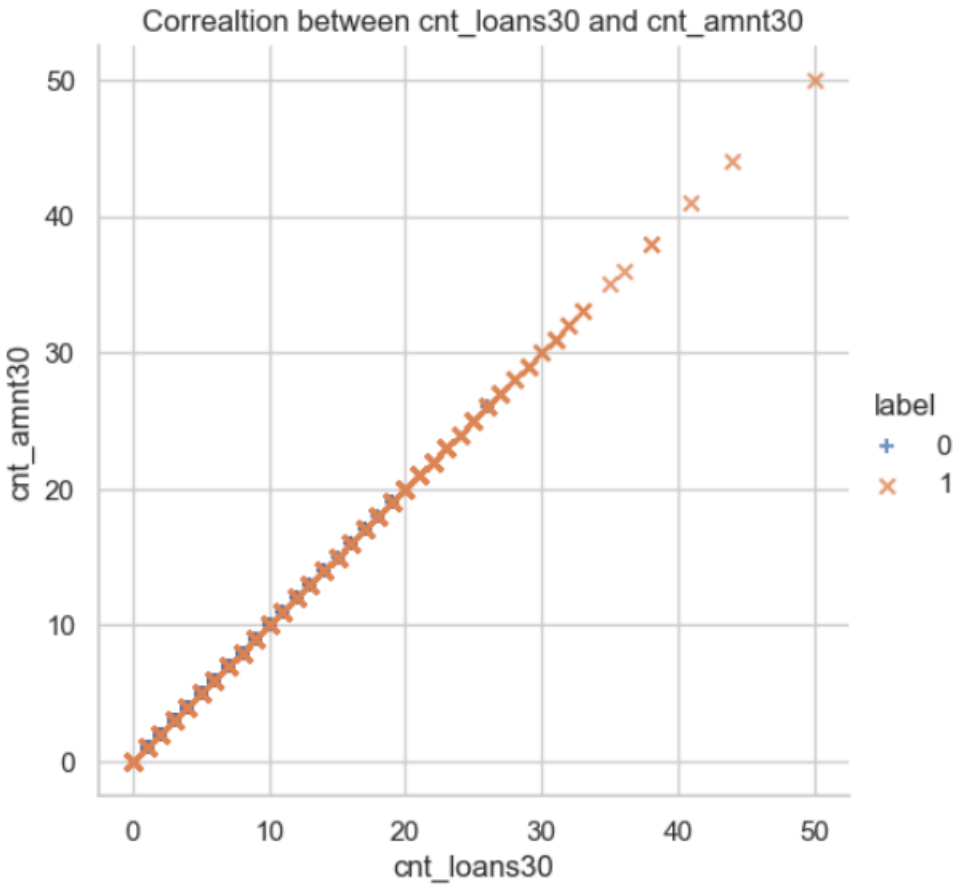
As 'cnt_ma_rech90' increases in the 8th month, the number of non-defaulters also increases.

Scatter plot between 'cnt_ma_rech_30' and 'cnt_ma_rech90' with respect to 'label':



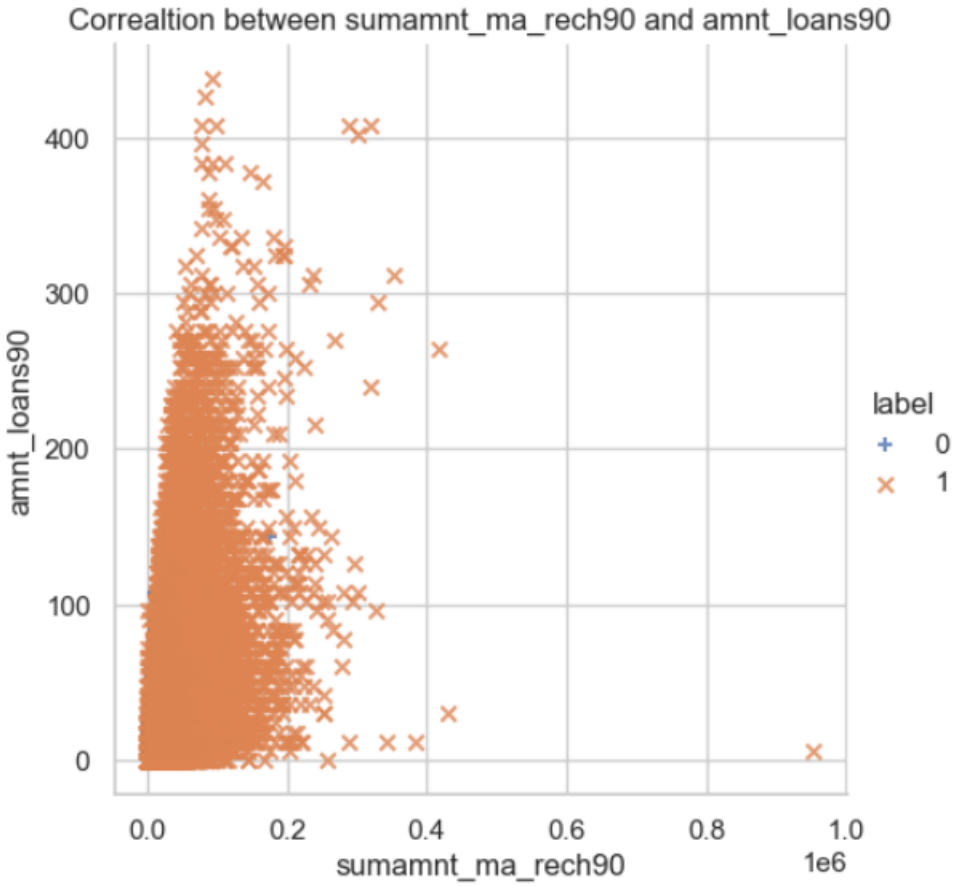
As 'cnt_ma_rech30' and 'cnt_ma_rech90' are increasing the number of non-defaulters are also increasing.

Scatter plot between 'cnt_loans30' and 'cnt_amnt30' with respect to label:



As 'cnt_loans30' and 'cnt_amnt30' are increasing the number of non-defaulters are also increasing.

Scatter plot between 'sumamnt_ma_rech90' and 'amnt_loans90' with respect to 'label':



As 'sumamnt_rech90' and 'amnt_loans30' are increasing, the number of non-defaulters are also increasing.

3.5 Run and Evaluate selected models

1. Finding best random state

In [121]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
maxAccu=0
maxRS=0
for i in range(1,200):
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=.30, random_state =i)
    DTC = DecisionTreeClassifier()
    DTC.fit(x_train, y_train)
    pred = DTC.predict(x_test)
    acc=accuracy_score(y_test, pred)
    if acc>maxAccu:
        maxAccu=acc
        maxRS=i
print("Best accuracy is ",maxAccu," on Random_state ",maxRS)
```

```
Best accuracy is  0.9166447087471266  on Random_state  194
```

Train-Test Split:

In [122]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=.30,random_state=maxRS)
```

2. Model Building

In [125]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,ExtraTreesClassifier
from sklearn.ensemble import GradientBoostingClassifier, BaggingClassifier
from xgboost import XGBClassifier as xgb

from sklearn.metrics import classification_report, confusion_matrix, roc_curve, accuracy_score, roc_auc_score
from sklearn.model_selection import cross_val_score
```

1) Decision Tree Classifier:

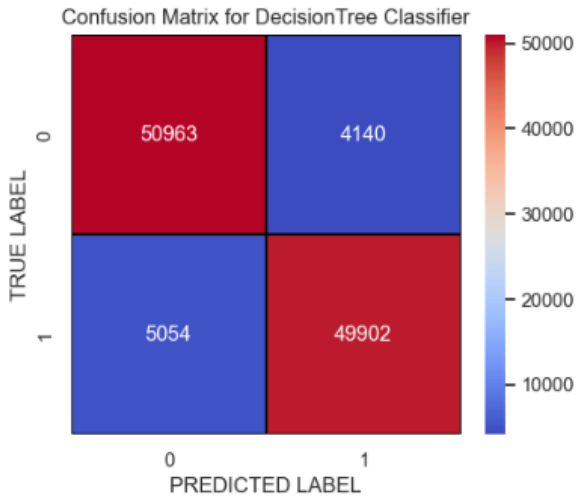
In [126]:

```
# Checking Accuracy and evaluation metrics for Decision Tree Classifier
DTC = DecisionTreeClassifier()

# Training the model
DTC.fit(x_train,y_train)

#Predicting y_test
predDTC = DTC.predict(x_test)

# Accuracy Score
DTC_score = accuracy_score(y_test, predDTC)*100
print("Accuracy Score:", DTC_score)

# ROC AUC Score
from sklearn.metrics import roc_auc_score
roc_auc_score = roc_auc_score(y_test,predDTC)*100
print("\nroc_auc_score:", roc_auc_score)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predDTC)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predDTC)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score = (cross_val_score(DTC, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score)

# Result of accuracy minus cv scores
Result = DTC_score - cv_score
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 91.64629880336909

roc_auc_score: 91.6451746922497

Confusion Matrix:
 [[50963  4140]
 [ 5054 49902]]

Classification Report:
                precision    recall  f1-score   support

           0       0.91      0.92      0.92     55103
           1       0.92      0.91      0.92     54956

    accuracy                           0.92    110059
   macro avg       0.92      0.92      0.92    110059
weighted avg       0.92      0.92      0.92    110059

Cross Validation Score: 90.98518046179122

Accuracy Score - Cross Validation Score is 0.6611183415778754
```

In [127]:

```
# Lets plot confusion matrix for DecisionTree Classifier

cm = confusion_matrix(y_test,predDTC)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for DecisionTree Classifier')
plt.show()
```



2) Random Forest Classifier:

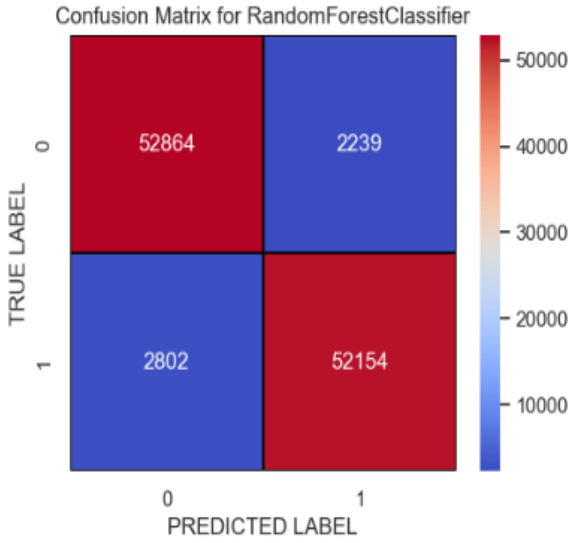
In [128]:

```
# Checking Accuracy and evaluation metrics for Random Forest Classifier
RFC = RandomForestClassifier()

# Training the model
RFC.fit(x_train,y_train)

#Predicting y_test
predRFC = RFC.predict(x_test)

# Accuracy Score
RFC_score = accuracy_score(y_test, predRFC)*100
print("Accuracy Score:", RFC_score)

#ROC AUC Score
from sklearn.metrics import roc_auc_score
roc_auc_score2 = roc_auc_score(y_test,predRFC)*100
print("\nroc_auc_score:", roc_auc_score2)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predRFC)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predRFC)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score2 = (cross_val_score(RFC, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score2)

# Result of accuracy minus cv scores
Result = RFC_score - cv_score2
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 95.4197294178577

roc_auc_score: 95.41903800355658

Confusion Matrix:
 [[52864  2239]
 [ 2802 52154]]

Classification Report:
                precision    recall  f1-score   support

           0       0.95      0.96      0.95     55103
           1       0.96      0.95      0.95     54956

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

Cross Validation Score: 95.04910656513876

Accuracy Score - Cross Validation Score is 0.37062285271893813
```

In [129]:

```
# Lets plot confusion matrix for RandomForestClassifier

cm = confusion_matrix(y_test,predRFC)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for RandomForestClassifier')
plt.show()
```



3) Extra Trees Classifier:

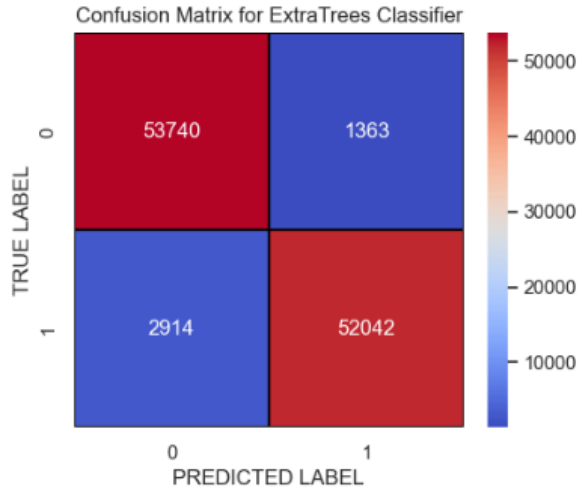
In [130]:

```
# Checking Accuracy and evaluation metrics for ExtraTrees Classifier
XT = ExtraTreesClassifier()

# Training the model
XT.fit(x_train,y_train)

#Predicting y_test
predXT = XT.predict(x_test)

# Accuracy Score
XT_score = accuracy_score(y_test, predXT)*100
print("Accuracy Score:", XT_score)

# ROC AUC Score
roc_auc_score3 = roc_auc_score(y_test,predXT)*100
print("\nroc_auc_score:", roc_auc_score3)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predXT)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predXT)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score3 = (cross_val_score(XT, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score3)

# Result of accuracy minus cv scores
Result = XT_score - cv_score3
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 96.11390254318138

roc_auc_score: 96.11201335415215

Confusion Matrix:
 [[53740  1363]
 [ 2914 52042]]

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.98      0.96     55103
           1       0.97      0.95      0.96     54956

    accuracy                           0.96    110059
   macro avg       0.96      0.96      0.96    110059
weighted avg       0.96      0.96      0.96    110059

Cross Validation Score: 96.40246422644371

Accuracy Score - Cross Validation Score is -0.2885616832623299
```

In [131]:

```
# Lets plot confusion matrix for ExtraTrees Classifier

cm = confusion_matrix(y_test,predXT)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for ExtraTrees Classifier')
plt.show()
```



4) Gradient Boosting Classifier:

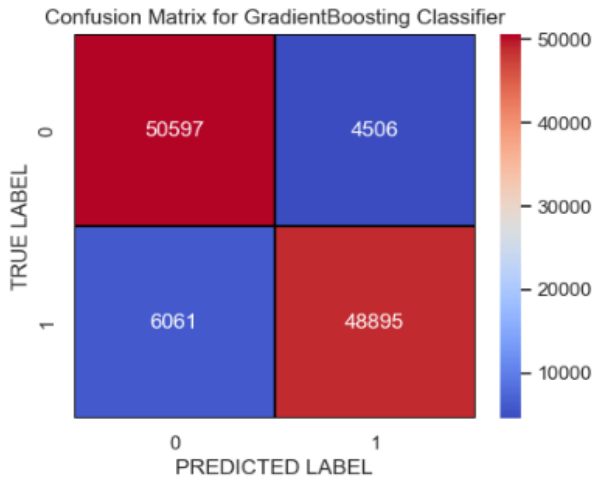
In [132]:

```
# Checking Accuracy and evaluation metrics for GradientBoosting Classifier
GB = GradientBoostingClassifier()

# Training the model
GB.fit(x_train,y_train)

#Predicting y_test
predGB = GB.predict(x_test)

# Accuracy Score
GB_score = accuracy_score(y_test, predGB)*100
print("Accuracy Score:", GB_score)

# ROC AUC Score
roc_auc_score4 = roc_auc_score(y_test,predGB)*100
print("\nroc_auc_score:", roc_auc_score4)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predGB)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predGB)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score4 = (cross_val_score(GB, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score4)

# Result of accuracy minus cv scores
Result = GB_score - cv_score4
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 90.39878610563426

roc_auc_score: 90.39688186678046

Confusion Matrix:
 [[50597  4506]
 [ 6061 48895]]

Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.92      0.91     55103
           1       0.92      0.89      0.90     54956

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

Cross Validation Score: 89.79072190798

Accuracy Score - Cross Validation Score is 0.608064197654258
```

In [133]:

```
# Lets plot confusion matrix for GradientBoosting Classifier

cm = confusion_matrix(y_test,predGB)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for GradientBoosting Classifier')
plt.show()
```



5) Extreme Gradient Boosting Classifier (XGB Classifier):

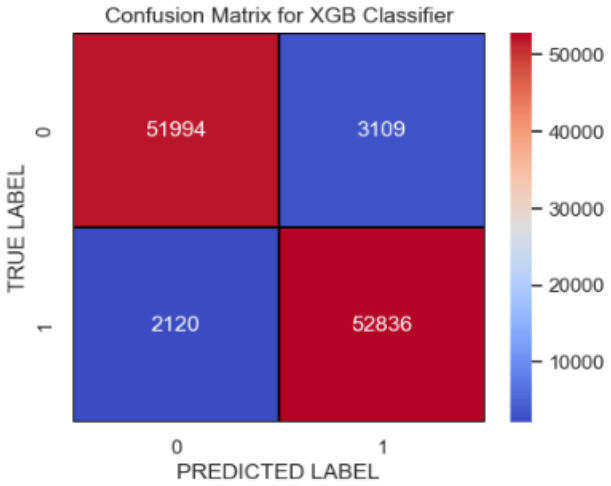
In [134]:

```
# Checking Accuracy and evaluation metrics for XGB Classifier
XGB = xgb(verbosity=0)

# Training the model
XGB.fit(x_train,y_train)

#Predicting y_test
predXGB = XGB.predict(x_test)

# Accuracy Score
XGB_score = accuracy_score(y_test, predXGB)*100
print("Accuracy Score:", XGB_score)

# ROC AUC Score
roc_auc_score5 = roc_auc_score(y_test,predXGB)*100
print("\nroc_auc_score:", roc_auc_score5)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predXGB)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predXGB)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score5 = (cross_val_score(XGB, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score5)

# Result of accuracy minus cv scores
Result = XGB_score - cv_score5
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 95.24891194722831

roc_auc_score: 95.2501036981717

Confusion Matrix:
 [[51994  3109]
 [ 2120 52836]]

Classification Report:
                precision    recall  f1-score   support

           0       0.96      0.94      0.95     55103
           1       0.94      0.96      0.95     54956

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

Cross Validation Score: 93.6905740032923

Accuracy Score - Cross Validation Score is 1.55833794393601
```

In [135]:

```
# Lets plot confusion matrix for XGB Classifier

cm = confusion_matrix(y_test,predXGB)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for XGB Classifier')
plt.show()
```



6) Bagging Classifier:

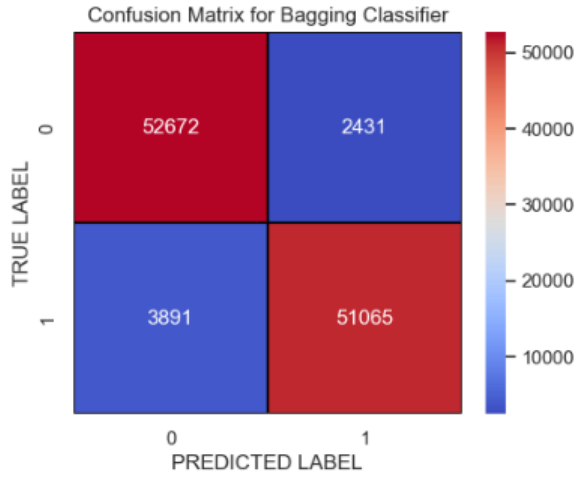
In [136]:

```
# Checking Accuracy and evaluation metrics for Bagging Classifier
BC = BaggingClassifier()

# Training the model
BC.fit(x_train,y_train)

#Predicting y_test
predBC = BC.predict(x_test)

# Accuracy Score
BC_score = accuracy_score(y_test, predBC)*100
print("Accuracy Score:", BC_score)

# ROC AUC Score
roc_auc_score6 = roc_auc_score(y_test,predBC)*100
print("\nroc_auc_score:", roc_auc_score6)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, predBC)
print("\nConfusion Matrix:\n",conf_matrix)

# Classification Report
class_report = classification_report(y_test,predBC)
print("\nClassification Report:\n", class_report)

# Cross Validation Score
cv_score6 = (cross_val_score(BC, x, y, cv=5).mean())*100
print("Cross Validation Score:", cv_score6)

# Result of accuracy minus cv scores
Result = BC_score - cv_score6
print("\nAccuracy Score - Cross Validation Score is", Result)
```

```
Accuracy Score: 94.25580824830318

roc_auc_score: 94.2540261799315

Confusion Matrix:
 [[52672  2431]
 [ 3891 51065]]

Classification Report:
                precision    recall  f1-score   support

           0       0.93      0.96      0.94     55103
           1       0.95      0.93      0.94     54956

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

Cross Validation Score: 93.7608829788175

Accuracy Score - Cross Validation Score is 0.4949252694856767
```

In [137]:

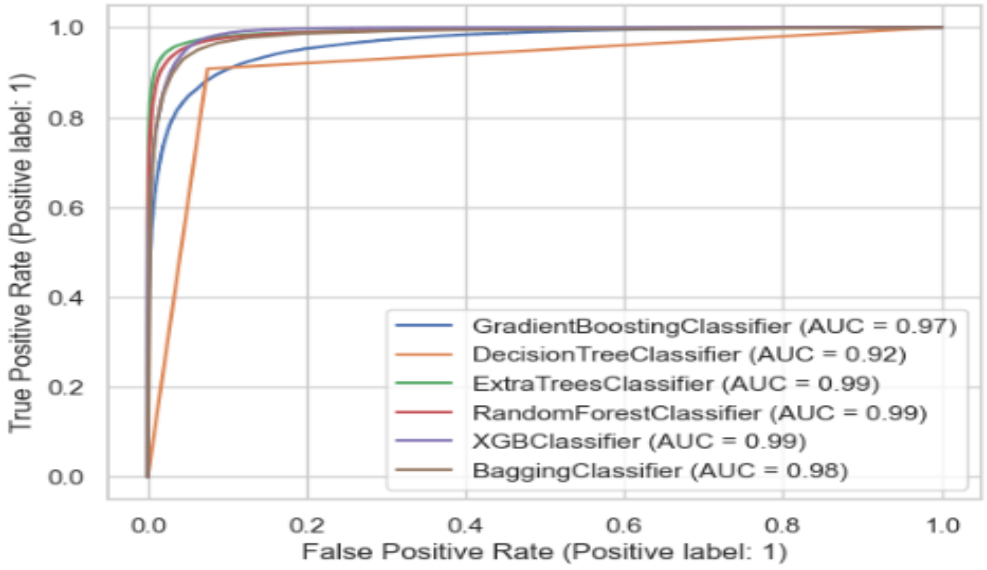
```
# Lets plot confusion matrix for Bagging Classifier

cm = confusion_matrix(y_test,predBC)
x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(5,4))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black",fmt = ".0f", ax=ax, cmap="coolwarm",xticklabels=x_axis_labels,

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for Bagging Classifier')
plt.show()
```



Plotting ROC and Comparing AUC for all the models used:

In [138]:

```
# Plotting for all the models used here

from sklearn import datasets
from sklearn import metrics
from sklearn import model_selection
from sklearn.metrics import plot_roc_curve

disp = plot_roc_curve(GB,x_test,y_test)      # ax_=Axes with confusion matrix
plot_roc_curve(DTC, x_test, y_test, ax=disp.ax_)
plot_roc_curve(XT, x_test, y_test, ax=disp.ax_)
plot_roc_curve(RFC, x_test, y_test, ax=disp.ax_)
plot_roc_curve(XGB, x_test, y_test, ax=disp.ax_)
plot_roc_curve(BC, x_test, y_test, ax=disp.ax_)

plt.legend(prop={'size':11}, loc='lower right')
plt.show()
```



Hyperparameter Tuning:

In [140]:
```
from sklearn.model_selection import RandomizedSearchCV

# GradientBoosting Classifier
parameters = {'max_depth': [3,6,9],
              'max_features':['auto', 'sqrt', 'log2'],
              'learning_rate':[0.1,0.25,0.5],
              'min_samples_leaf': [1,50,100]}
```

In [141]:
```
RCV = RandomizedSearchCV(GradientBoostingClassifier(), parameters, cv=5, n_iter =10)
```

In [142]:
```
RCV.fit(x_train,y_train)
```

Out[142]:
```
RandomizedSearchCV(cv=5, estimator=GradientBoostingClassifier(),
                   param_distributions={'learning_rate': [0.1, 0.25, 0.5],
                                        'max_depth': [3, 6, 9],
                                        'max_features': ['auto', 'sqrt',
                                                         'log2'],
                                        'min_samples_leaf': [1, 50, 100]})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

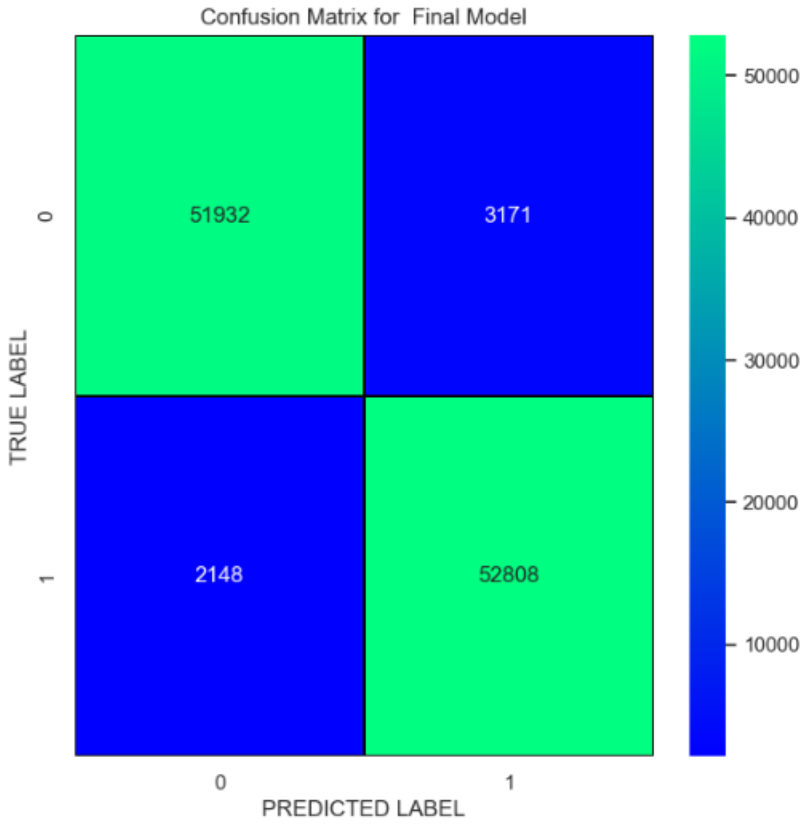
In [143]:
```
RCV.best_params_
```

Out[143]:
```
{'min_samples_leaf': 100,
 'max_features': 'auto',
 'max_depth': 6,
 'learning_rate': 0.25}
```

In [145]:
```
# Creating the final model

Micro_Credit_Model = GradientBoostingClassifier(min_samples_leaf = 100, max_features = "auto", max_depth = 6,learning_rate =
Micro_Credit_Model.fit(x_train, y_train)
pred = Micro_Credit_Model.predict(x_test)
acc_score = accuracy_score(y_test,pred)
print("Accuracy score for the Best Model is:", acc_score*100)
```

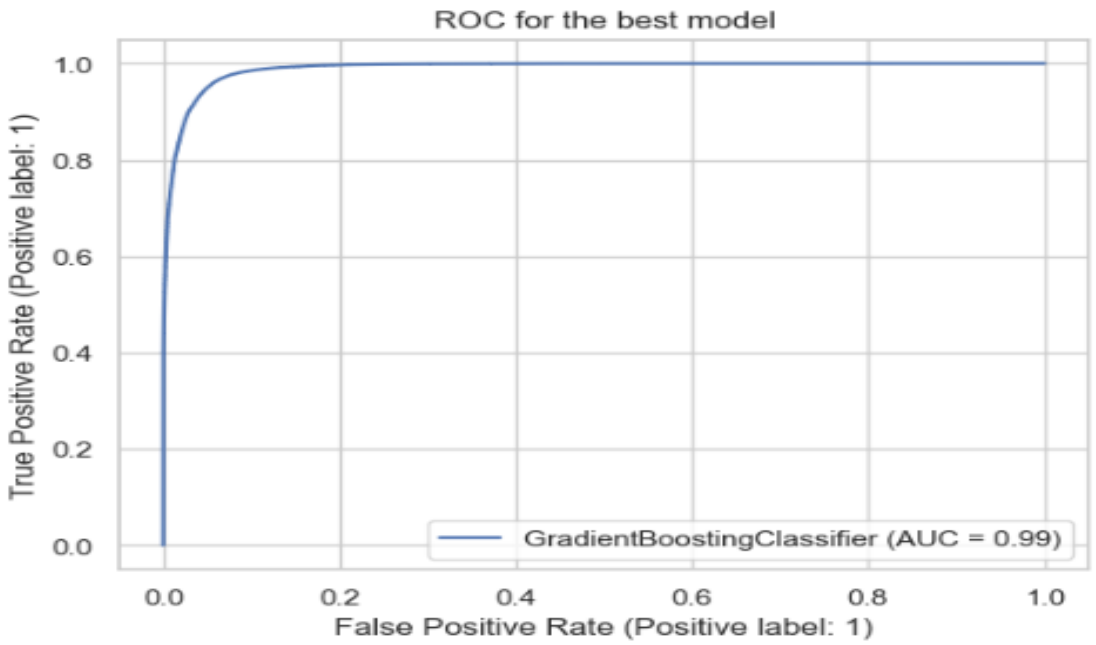
```
Accuracy score for the Best Model is: 95.16713762618232
```

In [146]:
```
# Lets plot confusion matrix for the final model

cm = confusion_matrix(y_test,pred)

x_axis_labels = ["0","1"]
y_axis_labels = ["0","1"]

f , ax = plt.subplots(figsize=(7,7))
sns.heatmap(cm, annot = True,linewidths=.2, linecolor="black", fmt = ".0f", ax=ax, cmap="winter",xticklabels=x_axis_labels,y

plt.xlabel("PREDICTED LABEL")
plt.ylabel("TRUE LABEL")
plt.title('Confusion Matrix for  Final Model')
plt.show()
```


Confusion Matrix for Final Model
51932
3171
2148
52808
0
1
TRUE LABEL
0
1
PREDICTED LABEL
50000
40000
30000
20000
10000

Plotting ROC and Comparing AUC for the Final model:

In [147]:
```
# Let's check the Auc for the best model after hyper parameter tuning

plot_roc_curve(Micro_Credit_Model, x_test, y_test)
plt.title("ROC for the best model")
plt.show()
```



Saving the model:

In [148]:
```
# Saving the model using .pkl

import joblib
joblib.dump(Micro_Credit_Model,"Micro_Credit_Loan_Defaulter.pkl")
```

Loading and Predicting the saved model:

In [149]:
```
# Loading the saved model
model=joblib.load("Micro_Credit_Loan_Defaulter.pkl")

#Prediction
prediction = model.predict(x_test)
prediction
```

Out[149]:
```
array([0, 0, 1, ..., 0, 1, 1], dtype=int64)
```

In [150]:
```
# Creating dataframe for predicted results

pd.DataFrame([model.predict(x_test)[:],y_test[:]], index=["Predicted","Original"])
```

Out[150]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
Predicted	0	0	1	1	0	1	1	0	1	1	0	0	0	0	1	0	1	0	0	1	1	1	1	1	0	1	1	1	1	0	0	1	1	1	1	1	1	0	0
Original	0	1	1	1	0	1	1	0	1	1	0	0	0	0	1	0	1	0	0	1	0	1	1	1	0	1	1	1	1	0	0	1	0	1	1	1	1	0	0

3.6 Interpretation of the Results

Visualizations: I have performed univariate, bivariate and multivariate analysis to find out the correlation between the label and the features. The heat map and bar plot helped me to understand the correlation between dependent and independent features. Also, heat map helped to detect the multicollinearity problem and feature importance. Detected outliers and skewness with the help of box plots and distribution plots respectively. And I found some of the features are skewed to right.

Pre-processing: The dataset should be cleaned and scaled to build the ML models to get good predictions. I have performed few processing steps which I have already mentioned in the pre-processing steps where all the important features are present in the dataset and ready for model building.

Model building: After cleaning and processing data, I performed train-test split to build the model. I have built multiple classification models to get the accurate accuracy score, and evaluation metrics like precision, recall, confusion matrix, f1 score. I got Gradient Boosting Classifier as best model which gives 90% accuracy score. I checked the cross-validation score ensuring there will be no overfitting.

After tuning the best model Gradient Boosting Classifier, I got 95% accuracy score and also got increment in AUC-ROC curve. Finally, I saved my final model and got good prediction results for defaulters.

4. Conclusion

4.1 Key Findings and Conclusions of the Study

This case study aims to give an idea of applying EDA in a real business scenario. In this case study, apart from applying the techniques that we have learnt in the EDA module, we will also develop a basic understanding of risk analytics in banking and financial services and understand how data is used to minimize the risk of losing money while lending to customers. From this dataset we were able to understand that the selection of customers for the credit to know whether they are defaulters or non-defaulters are done on the basis of different features.

In this study, we have used multiple machine learning models to predict the micro credit defaulters' rate. We have gone through the data analysis by performing feature engineering, finding the relation between features and label through visualizations. And got the important feature and we used these features to predict the defaulters' rate by building ML models. After training the model we checked CV score to overcome with the overfitting issue. Performed hyper parameter tuning, on the best model and the best model accuracy increased by 5% and the accuracy score was 95%. We have also got good prediction results.

From the whole study we found that the MFIs have provided loan to the user who have no recharge or balance in their account which needs to be stopped. Also, the frequency of main account recharged in last 30 days & 90 days we have seen the users with low frequency are causing huge losses, company should implement some kind of strategies to reduce like sending SMS alerts for notification. We found the defaulting rate is higher in old customers list. We found outliers and removed them and couldn't remove all the outliers since the data is expensive so, proceeded the data with remaining outliers. Further, removed skewness. Looking at the heat map, I could see there were few features which were correlated with each other, yet I haven't removed them based on their correlation thinking multicollinearity will not affect prediction. Other insight from this study is the impact of SMOTE on the model performance as well as how the number of variables included in the models.

4.2 Learning Outcomes of the Study in respect of Data Science

While working on this project I learned many things about the micro credit loan banks and organizations and how the machine learning models have helped to predict the defaulters' rate which provides greater understanding into the many causes of loan defaults in Microfinance Banks. I found that the project was quite interesting as the dataset contains several types of data. I used several types of plotting to visualize the relation between target and features. This graphical representation helped me to understand which features are important and how these features describe defaulter and non-defaulter rate in the banks. Data cleaning was one of the important and crucial things in this project where I dealt with features having zero values, negative statistical summary and time variables.

One of the challenges I faced while data cleaning is outlier removal, in most of the scenario's Z-score will be used as outlier removal technique since it performs quite well with less data loss. In our data set, Z-score has caused 19% data loss. Then I tried another technique called Inter Quartile Range it caused around 63% data loss. So, I used percentile method to handle the outliers.

Finally, our aim is achieved by predicting the defaulters' rate at the organization that could help the clients in further investment and improvement in selection of customers.

4.3 Limitations of this work and Scope for Future Work

The dataset contains the data of only 2016 year belonging to telecom industry, if we get the data of other years along with other telecom companies then dataset would be quite more interesting to handle and predict on varied scenario. In the dataset our data is not properly distributed in some of the columns many of the values in the columns are 0's and negative values which are not realistic. Because I have seen in some of the columns even the person didn't take loan but the label says that he paid back the loan amount, which is not correct. So, because of that data our models may not make the right patterns and the performance of the model also reduces. So that issues need to be taken care. Due to the presence of huge outliers, we unsure that our model is going to perform well on the dataset. Due to the class imbalance we had to balance the class defaulter (0). This might also have some effect on model.

The potential future work for this project will be a further development of the model by deepening analysis on variables used in the models as well as creating new variables in order to make better predictions. As a recommendation, the Microfinance Institutions (MFI's) should adopt the group loan policy as the main mode through which microcredit may be issued to suitable applicants. Regular Credit risk assessment and analysis should be undertaken by the MFIs. Microfinance Bank officials should personally visit the group clients, either in their shops or houses to ascertain that the group is genuinely formed and that all members are serious business people and not just members by name. This will also avoid customers giving fake addresses with the intention to run away with the Bank money.

5. Reference

- https://www.google.com/
- https://scikit-learn.org/stable/index.html
- https://www.researchgate.net/
- https://journal-archieves31.webs.com
- https://github.com/
- https://towardsdatascience.com/
- https://www.analyticsvidhya.com/