FLIP ROBO

NAME OF THE PROJECT

“Micro-Credit Defaulter Model”

Submitted by:

Leena chatterjee

ACKNOWLEDGMENT:

- I have taken efforts in this project. However, it would not have been possible without the kind support and help of each individual of DATA TRAINED organizations. I would like to extend my sincere thanks to all of them.
- I am highly indebted to all team of Data trained for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

- I would like to express my special gratitude and thanks to MS Swati Rustagi for guiding for this project

Bibliography:

- https://www.searchstartnow.com/web?qo=semQuery&ad=semA&q=github%20for%20beginners&o=1468511&ag=fw4&an=msn_s&adid=79096273683706&agid=1265538600064869&campaignid=416218294&clickid=57fa256a7fcb1776d83445cf499fe6e2&clid=aj-shopnet-it&kwid=kwd-79096564506817%3Aloc-90&rch=intl835&utm_medium=bcpc&utm_source=b
- https://www.kaggle.com/learn

INTRODUCTION

Problem Statement:

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 is very useful in such areas, the implementation of MFS has been uneven with both significant challenges and successes.

We are working client dataset that is in Telecom Industry. They are a fixed wireless telecommunications network provider. They have launched various products and have developed its business and organization based on the budget operator model, offering better products at Lower Prices to all value conscious customers through a strategy of disruptive innovation that focuses on the subscriber.They understand the importance of communication and how it affects a person’s life, thus, focusing on providing their services and products to low income families and poor customers that can help them in the need of hour.

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

The sample data is provided to us from client database. In order to improve the selection of customers for the credit loan of mobile balance , the client wants some predictions that could help them in further investment and improvement in selection of customers. In this dataset target variable is’ label which will represent value as 0 and 1. **Label ‘1’ indicates that the loan has been payed i.e. Non- defaulter, while, Label ‘0’ indicates that the loan has not been payed i.e. defaulter.**

Conceptual Background of the Domain Problem

Data science is the field where we can predict the probability. Here basically we need to analyse who are able to get mobile balance as a microcredit loan which has to be paid back

Basic summery

We can **compare** it is basically analysis of loan sector data. we should analyze data in a way so that it helps client for choosing customer(simple summarization – Assumption --we can compare it with bank sector data like customer will get home loan based on their salary, average balance maintenance in account and many more depending variable , so that bank will get genuine customer)

Review of Literature

AS mentioned here output will be 0 and 1 based on that we will find reliable customer which will paid back money .Here as explained earlier output will depend on some variable for choosing customer

In this dataset those variables are ====➔

- cnt_ma_rech90----- (Number of times main account got recharged in last 30 days) – --So that we can predict that he is a permanent customer who are recharging mobile number more frequently , and here the relationship is positive co-relationship . The number recharge done more getting micro credit loan for mobile recharge probability is high

- sumamnt_ma_rech90 ------(Total amount of recharge in main account over last 90 days (in Indonesian Rupiah)) --Total amount of recharge in main account over last 90 , getting more more getting micro credit loan for mobile recharge probability is high

- sumamnt_ma_rech30-----------(Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)) ------ Total amount of recharge in main account for mobile recharge probability is high

Like below mentioned variable are representing same. One increase getting loan probability will be high

- amnt_loans90 --------Total amount of loans taken by user in last 90 days
- amnt_loans30 ----- --Total amount of loans taken by user in last 30 days
- cnt_loans30 ---- ------Number of loans taken by user in last 30 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 30 days (in Indonesian Rupiah)

Probability of getting micro-credit loan for mobile will be less if below mention variable value will be increased

- medianmarechprebal30 -- Median of main account balance just before recharge in last 30 days at user level (in Indonasian Rupiah)

When we will visualize data we will get graphical representation of above

Motivation for the Problem Undertaken

Objective--This is the micro-credit mobile loan based on some variable we will be deciding where customer is really genuine whether he is able to paid back the loan or not

we study this model so this will help us to analyse any real world loan sector data example – if customer wants to take loan from bank based on records like average balance maintenance in bank account, transaction of every month from the account and income proof will help us to find genuine customer so that anything fraud will not be happening and financial organisation will not run in loss

As we worked with real time data , we have gained knowledge that what are challenges has to face while working with real domain data(heavy data set) , sometimes some information is uncertain so using this experience I believe we can work better on next project and that is being the best motivation behind this project work

Mathematical/ Analytical Modeling of the Problem

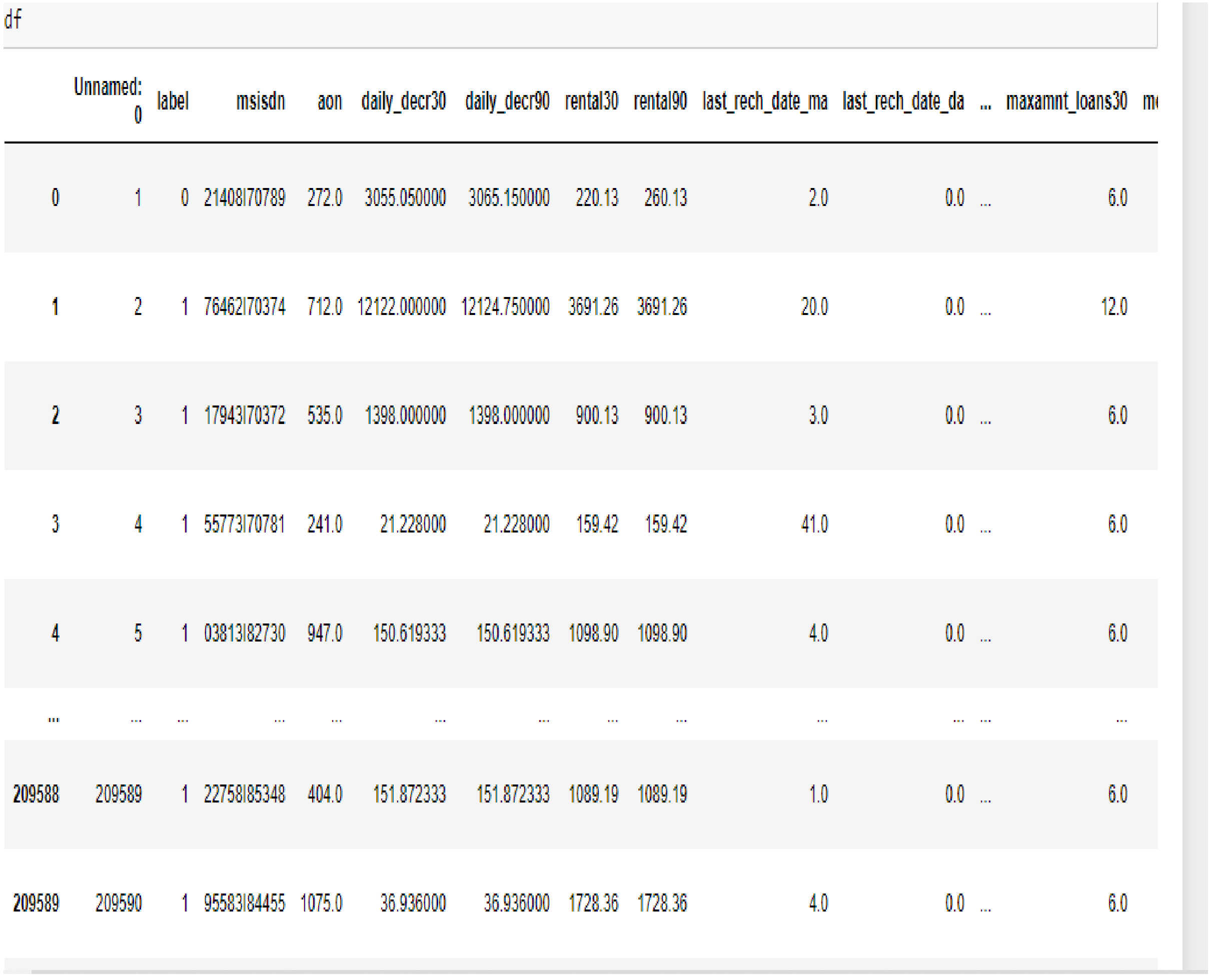
supervised learning uses labelled input and output data Supervised learning (SL) is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.[It infers a function from *labelled training data* consisting of a set of *training examples*.[In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value (also called the *supervisory signal*). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances

Here our dataset consist of Categorical data which is part of supervised learning so we will analyse with classification (Logistic classification)

Classification is a process in which an algorithm is used to analyze an existing data set of known points. The understanding achieved through that analysis is then leveraged as a means of appropriately classifying the data. Classification is a form of machine learning that can be particularly helpful in analyzing very large, complex sets of data to help make more accurate predictions.

Data Sources and their formats

Here data is provided by organisation (It is a real time data which we got from client's database)



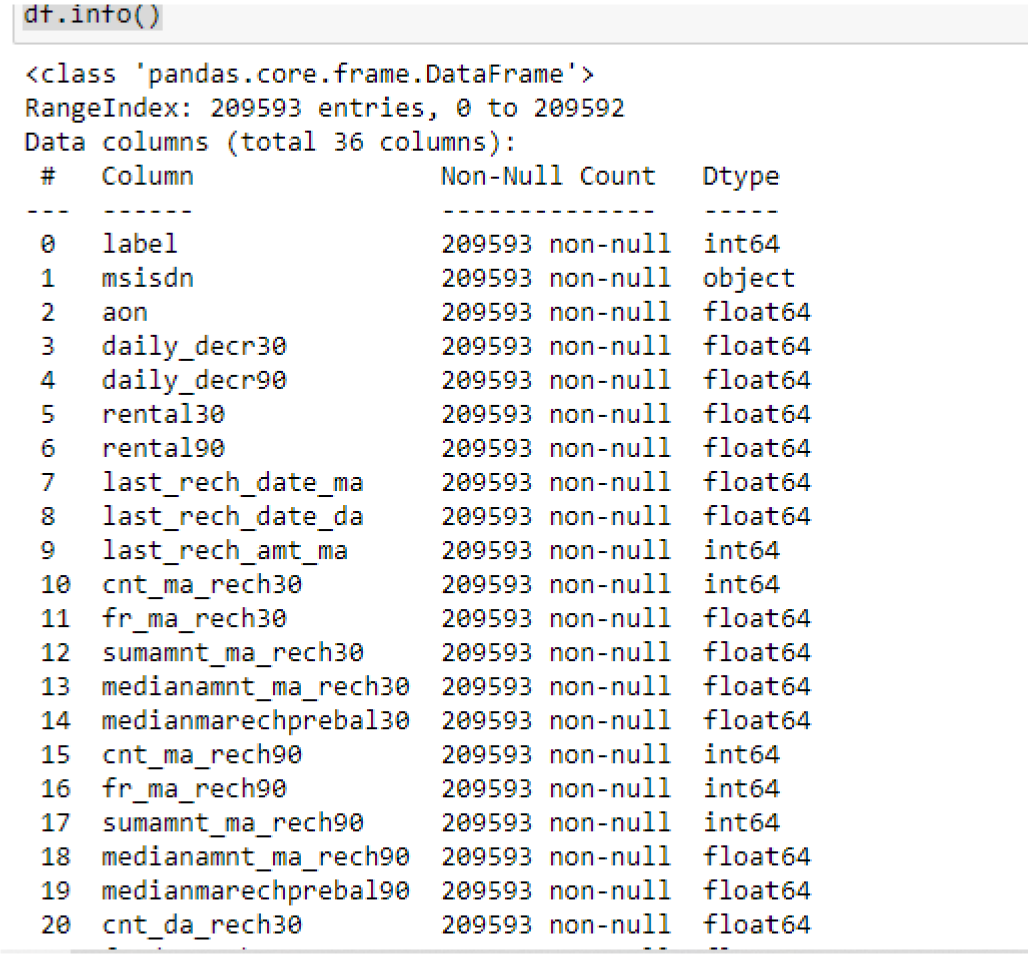
```
df
```

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	m
0	1	0	21408I70789	272.0	3055.050000	3065.150000	220.13	260.13	2.0	0.0	...	6.0	
1	2	1	76462I70374	712.0	12122.000000	12124.750000	3691.26	3691.26	20.0	0.0	...	12.0	
2	3	1	17943I70372	535.0	1398.000000	1398.000000	900.13	900.13	3.0	0.0	...	6.0	
3	4	1	55773I70781	241.0	21.228000	21.228000	159.42	159.42	41.0	0.0	...	6.0	
4	5	1	03813I82730	947.0	150.619333	150.619333	1098.90	1098.90	4.0	0.0	...	6.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
209588	209589	1	22758I85348	404.0	151.872333	151.872333	1089.19	1089.19	1.0	0.0	...	6.0	
209589	209590	1	95583I84455	1075.0	36.936000	36.936000	1728.36	1728.36	4.0	0.0	...	6.0	

Dataset what we have received that is csv file. We saved the file in current working directory of our local system as csv file
After that using panda.read_csv we uploaded to jupyter note book in df variable
[Panda is in built library in jupyter Notebook]

Using below command we got some basic information of data which is mentioned below

df.info()--- it provided object type of each columns .our dataset content of 209593 rows × 36 columns



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209593 entries, 0 to 209592
Data columns (total 36 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   label                 209593 non-null  int64
 1   msisdn                209593 non-null  object
 2   aon                   209593 non-null  float64
 3   daily_decr30          209593 non-null  float64
 4   daily_decr90          209593 non-null  float64
 5   rental30              209593 non-null  float64
 6   rental90              209593 non-null  float64
 7   last_rech_date_ma     209593 non-null  float64
 8   last_rech_date_da     209593 non-null  float64
 9   last_rech_amt_ma      209593 non-null  int64
 10  cnt_ma_rech30         209593 non-null  int64
 11  fr_ma_rech30          209593 non-null  float64
 12  sumamnt_ma_rech30     209593 non-null  float64
 13  medianamnt_ma_rech30  209593 non-null  float64
 14  medianmarechprebal30  209593 non-null  float64
 15  cnt_ma_rech90         209593 non-null  int64
 16  fr_ma_rech90          209593 non-null  int64
 17  sumamnt_ma_rech90     209593 non-null  int64
 18  medianamnt_ma_rech90  209593 non-null  float64
 19  medianmarechprebal90  209593 non-null  float64
 20  cnt_da_rech30         209593 non-null  float64
```

2.df.dypes= its provided info that what the data type belongs to (float , int)

3 df.isnull.sum()--- we found there is no null value

4 df1.head()--- it shows first five columns in the dataset

5 df.columns—it shows total columns of the dataset

6> df1[column name].value_counts()—provide unique value of this particular column

7> df.sample—To take same from dataset

Data Pre-processing

Using label-encoder we converted categorical data to numeric as saved at df1 file We calculated correlation using df.corr () and plot as heat map for checking correlationship

As this is categorical data we cannot find mean so unable to calculate standard deviation , for categorical data that’s being the reason we cannot remove outlier or cannot define skewness and same informed by DATA trained mentor too.

Assumptions to the problem under consideration

We found ‘pcircle’ column having same value of each row so it will not contribute anything to model so we drop this column

While plotting heat map some variable are highly correlated with output variable i assume if I plot those variable with output through scatter plot we will get good graphical representation]

Dataset is too heavy. it is taking time to perform or execute , so for pair plot we use sample of dataset

Hardware and Software Requirements and Tools Used

Hardware

- Good performance PC [Minimum – 8gb RAM +SSD]
- Enough space in hard disk drive

Software requirements

- jupyter note book
- Sometimes you may need Google colab to cross check the output

Package

- Numpy ---import numpy as np (For calculation)
- **P**anda-import pandas as pd (read data frame)
- Imblearn----- For class sampling

Here the list of some other function

- For plotting- 1>import seaborn as sns 2> import matplotlib.pyplot as plt
- For ignore new version warning--- import warnings
  warnings.filterwarnings('ignore')'

- For class balancing----from imblearn.over_sampling import SMOTE
- from sklearn.linear_model import LogisticRegression
- from sklearn.model_selection import train_test_split
- from sklearn.naive_bayes import MultinomialNB
- from sklearn.svm import SVC
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.neighbors import KNeighborsClassifier
- from sklearn.ensemble import AdaBoostClassifier
- from sklearn.ensemble import RandomForestClassifier
- from sklearn.metrics import confusion_matrix, classification_report ,accuracy score

Model/s Development and Evaluation

Problem-solving approaches (methods):----

- As data is numeric we used label encoder and made it numeric data
- Using SMOTE we made both the output class equal

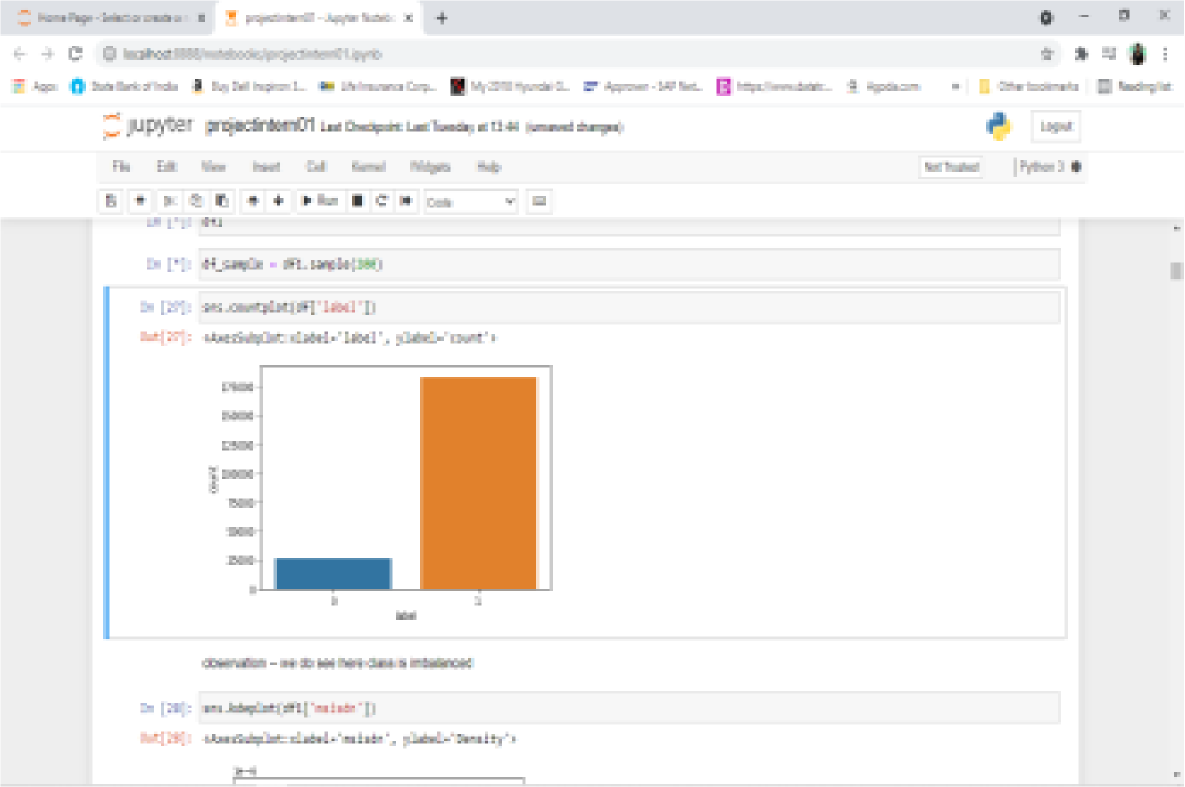
Data visualization:

We visualized data with univariate, bivariate and multivariate plot

Univariate analysis:

Countplot---- Using countplot we found target variable 'label' having class

Imbalance concern .below plot shows the graphical representation of same

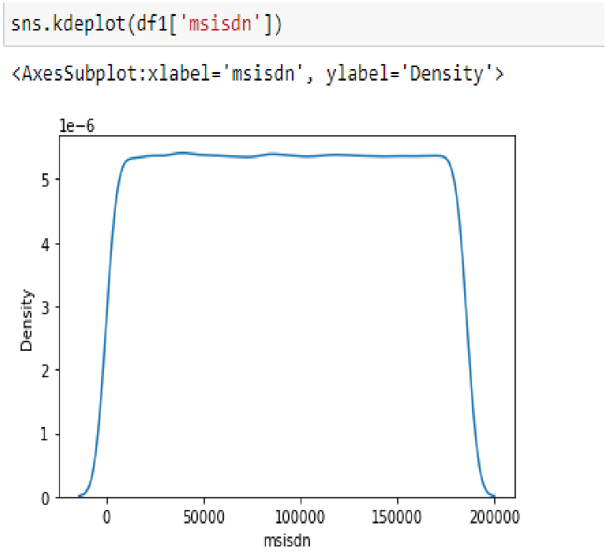


```
In [*]: df1

In [*]: df_sample = df1.sample(100)

In [27]: sns.countplot(df['label'])
Out[27]: <AxesSubplot:xlabel='label', ylabel='count'>
```

observation -- we do see here class is imbalanced

```
In [28]: sns.kdeplot(df1['msisdn'])
Out[28]: <AxesSubplot:xlabel='msisdn', ylabel='Density'>
```

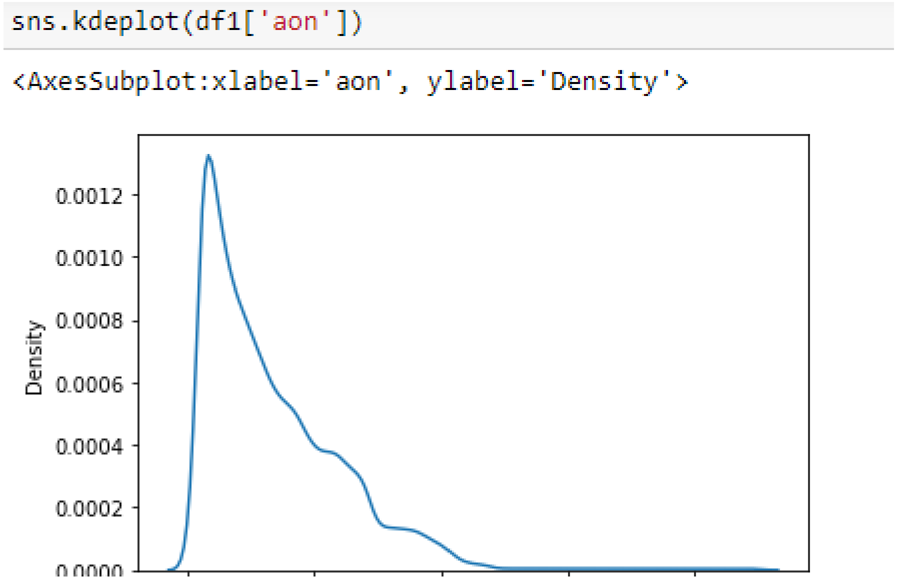
Kdeplot --We used Kde plot of each column where x axis we plot value of column and y axis we plot kernel density . This graph basically represent data distribution



```
sns.kdeplot(df1['msisdn'])
```

```
<AxesSubplot:xlabel='msisdn', ylabel='Density'>
```

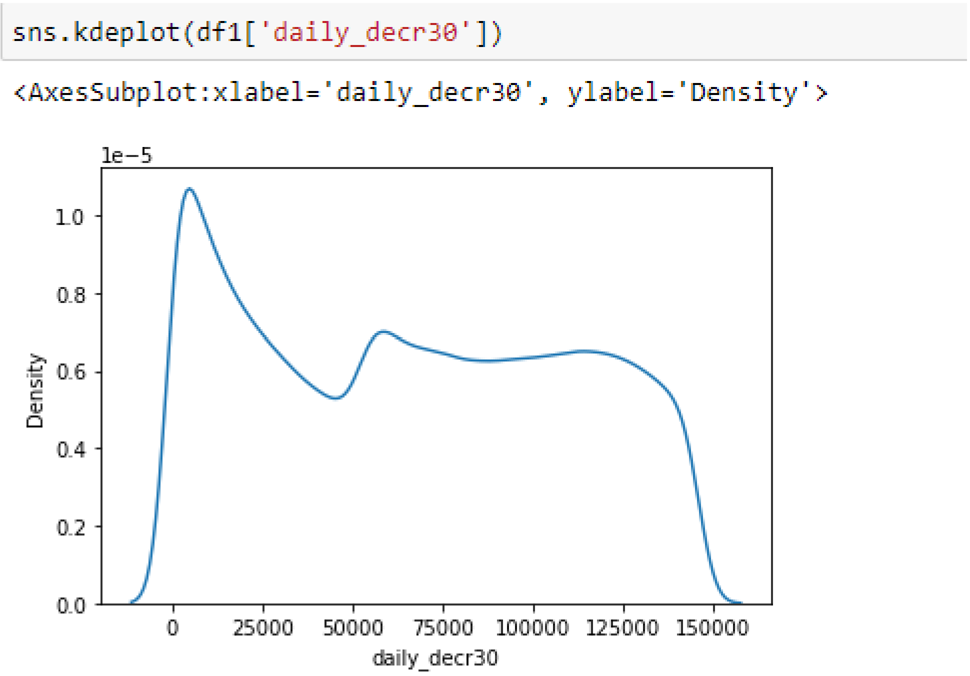
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
sns.kdeplot(df1['aon'])
```

```
<AxesSubplot:xlabel='aon', ylabel='Density'>
```

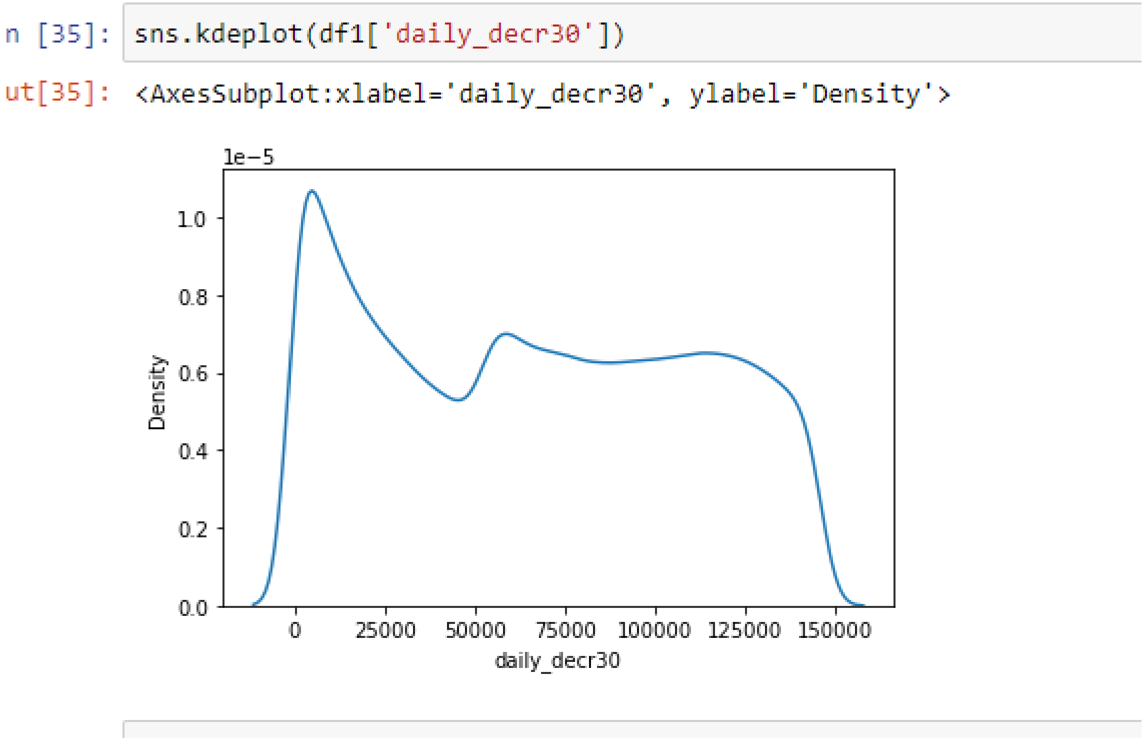
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
sns.kdeplot(df1['daily_decr30'])
```

```
<AxesSubplot:xlabel='daily_decr30', ylabel='Density'>
```

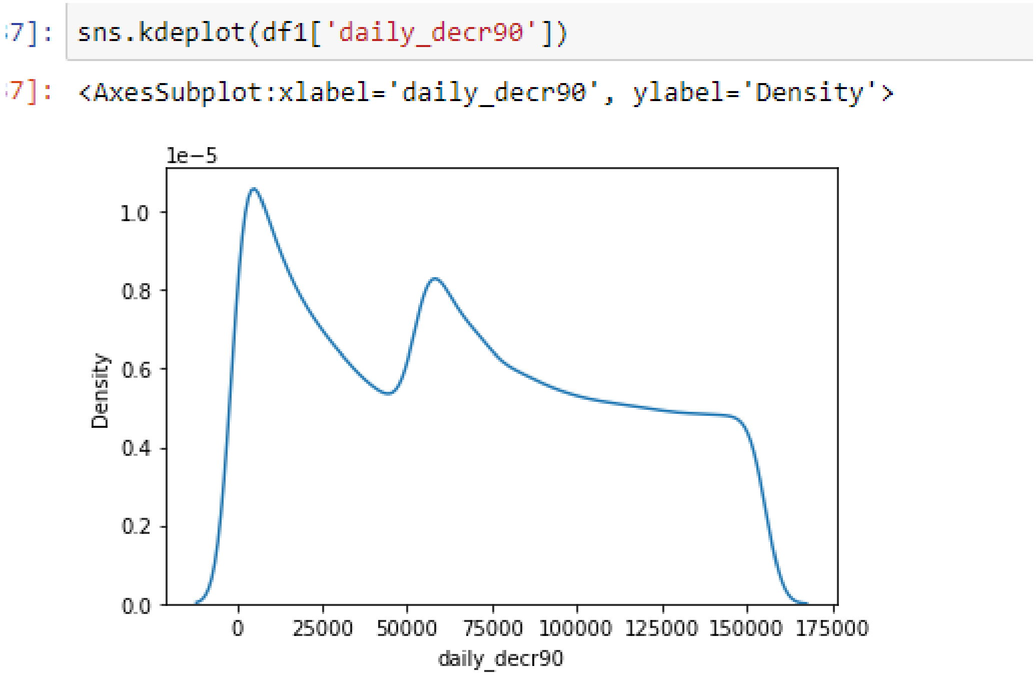
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
n [35]: sns.kdeplot(df1['daily_decr30'])
```

```
ut[35]: <AxesSubplot:xlabel='daily_decr30', ylabel='Density'>
```

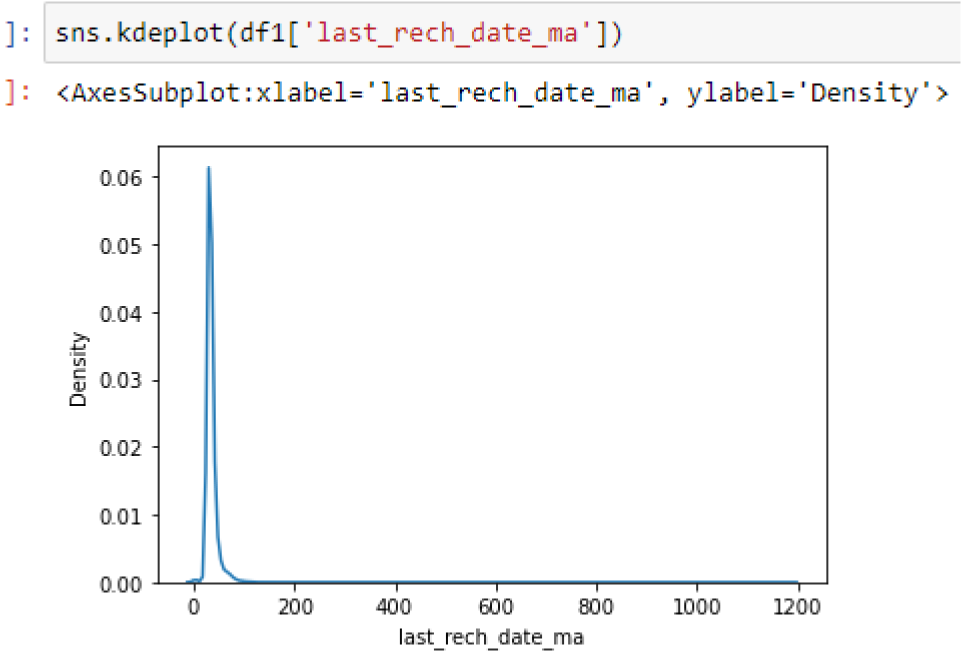
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
7]: sns.kdeplot(df1['daily_decr90'])
```

```
7]: <AxesSubplot:xlabel='daily_decr90', ylabel='Density'>
```

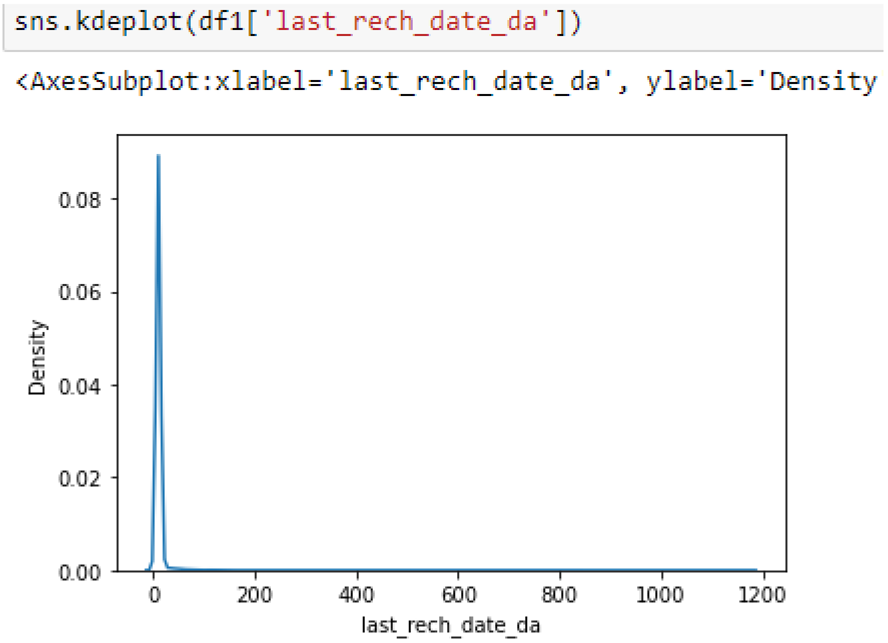
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
]: sns.kdeplot(df1['last_rech_date_ma'])
```

```
]: <AxesSubplot:xlabel='last_rech_date_ma', ylabel='Density'>
```

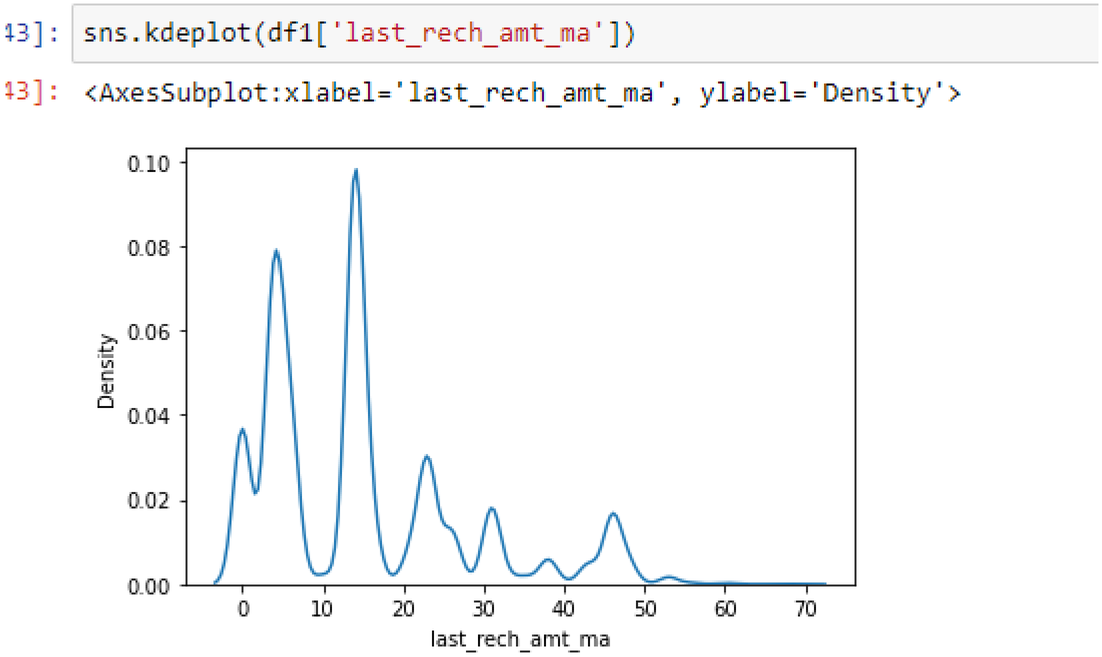
Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution



```
sns.kdeplot(df1['last_rech_date_da'])
```

```
<AxesSubplot:xlabel='last_rech_date_da', ylabel='Density
```

Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution

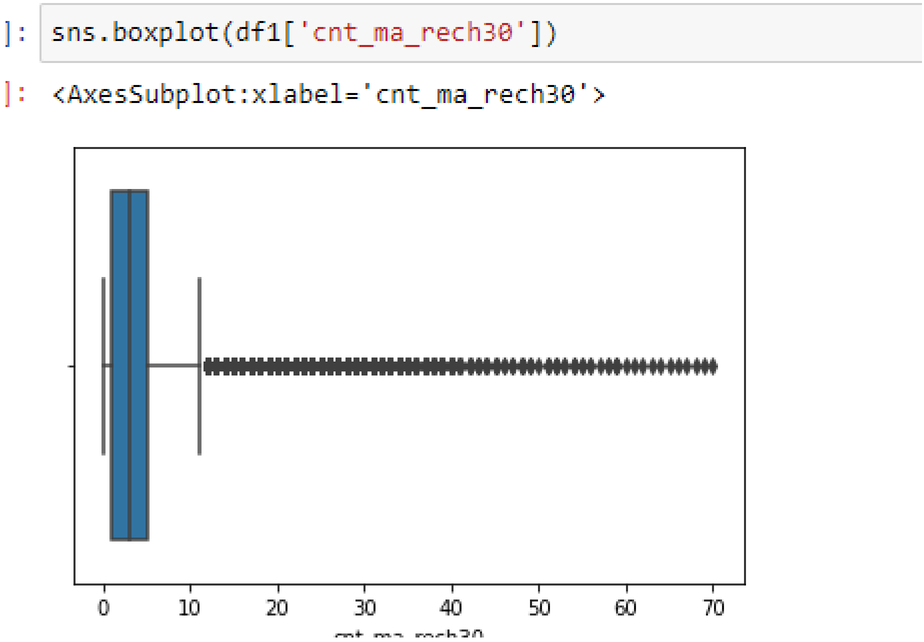


```
43]: sns.kdeplot(df1['last_rech_amt_ma'])
```

```
43]: <AxesSubplot:xlabel='last_rech_amt_ma', ylabel='Density'>
```

Observation--Kde plot of this column where x axis we plot value of column and y axis we plot kernel density we understand data distribution

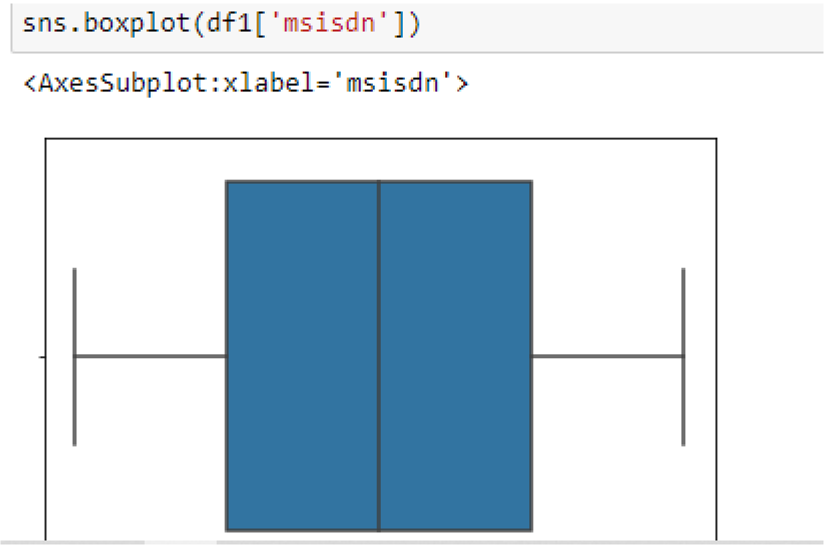
Box plot—Although it is categorical data still we used box plot to get outlier concept   through graphical representation.



```
]: sns.boxplot(df1['cnt_ma_rech30'])
```

```
]: <AxesSubplot:xlabel='cnt_ma_rech30'>
```

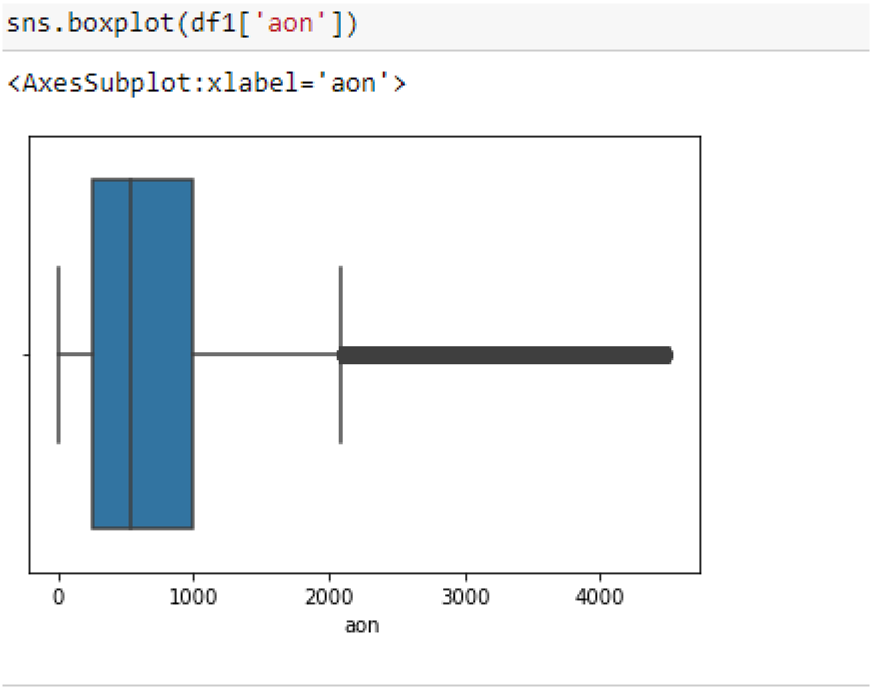
Observation – As per graphical view outlier s present, as some data present above vertices .



```
sns.boxplot(df1['msisdn'])
```

```
<AxesSubplot:xlabel='msisdn'>
```

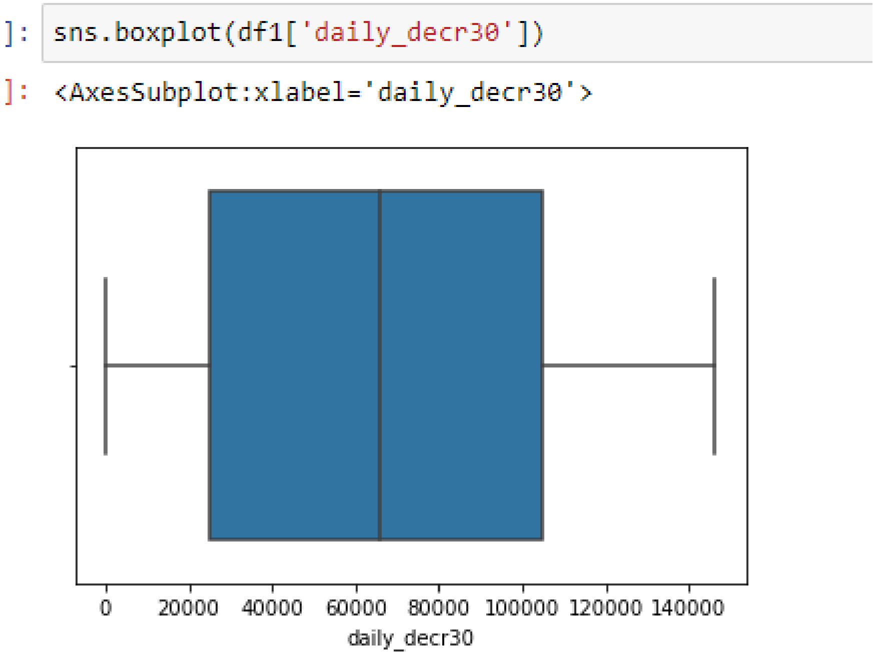
Observation – As per graphical view outlier s not present .



```
sns.boxplot(df1['aon'])
```

```
<AxesSubplot:xlabel='aon'>
```

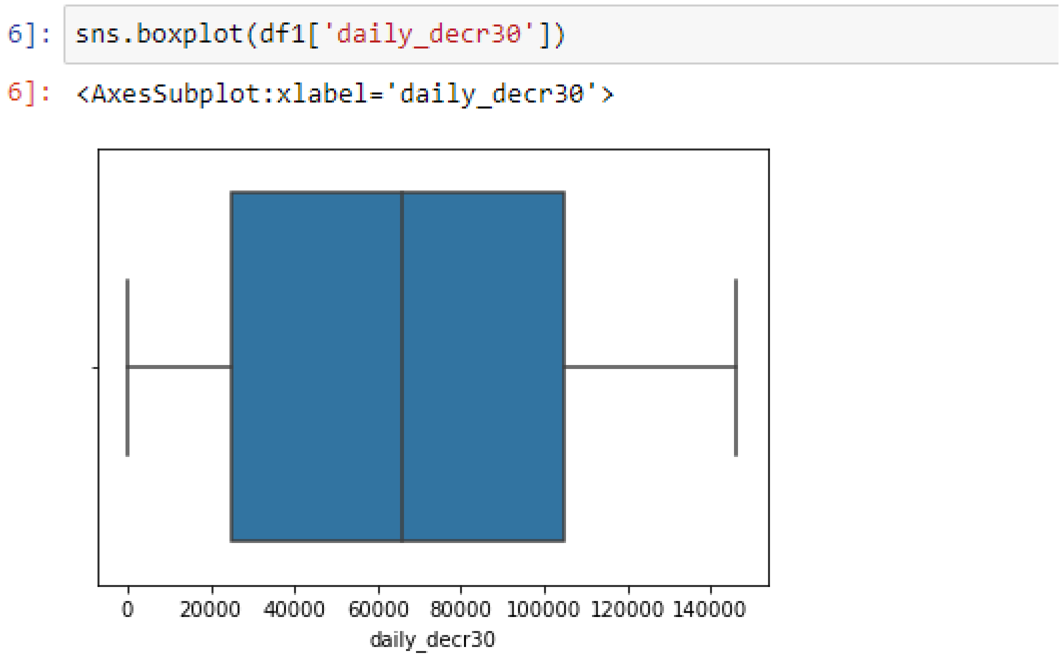
Observation – As per graphical view outlier s not present .



```
]: sns.boxplot(df1['daily_decr30'])
```

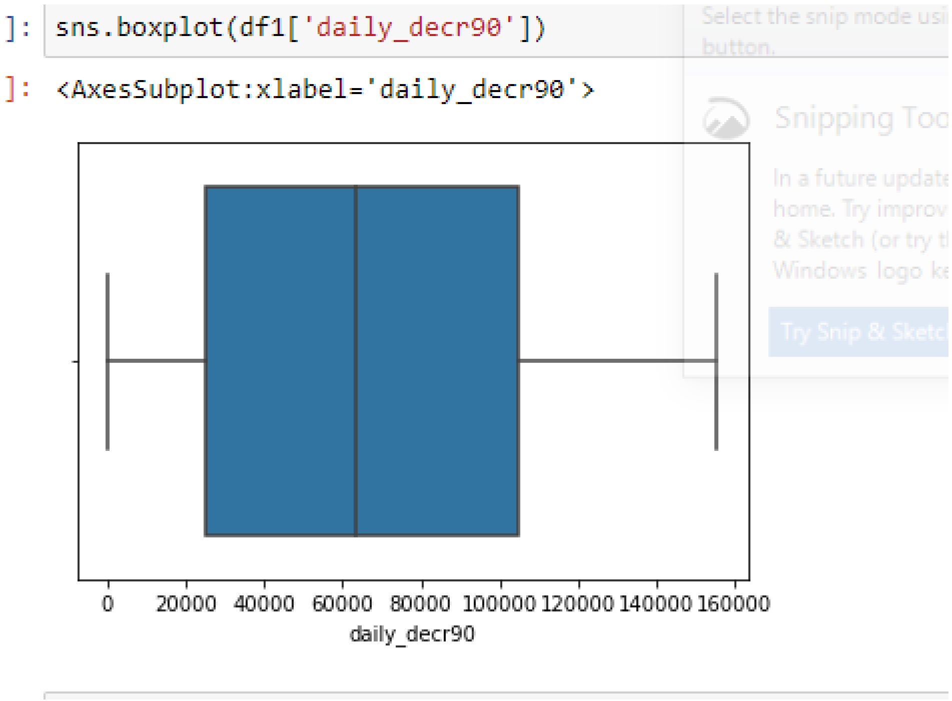
```
]: <AxesSubplot:xlabel='daily_decr30'>
```

Observation – As per graphical view outlier s not present .



```
6]: sns.boxplot(df1['daily_decr30'])
6]: <AxesSubplot:xlabel='daily_decr30'>
```

Observation – As per graphical view outlier s present , as some data present above vertices



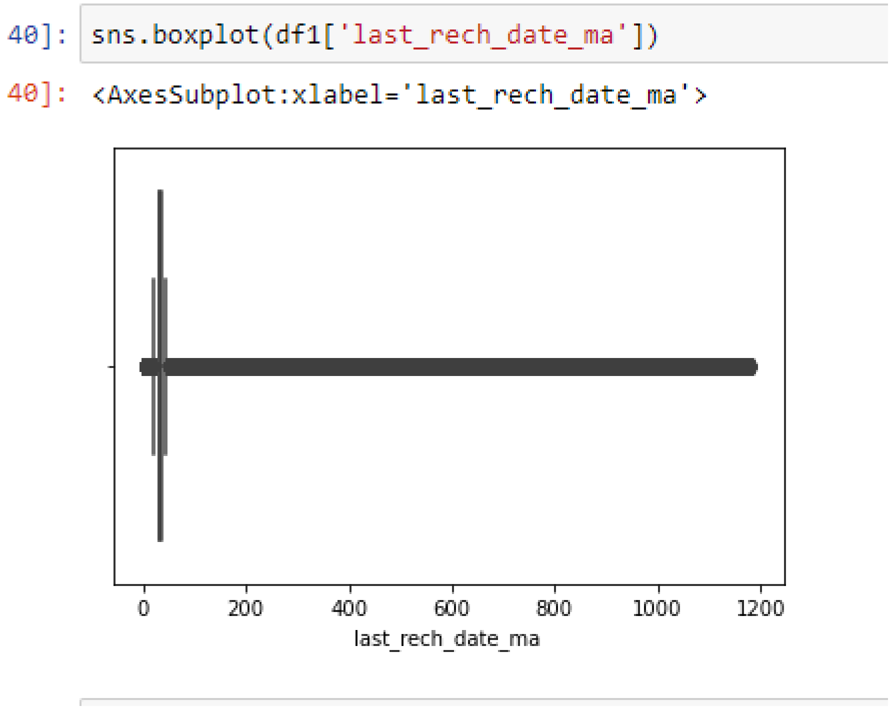
```
]: sns.boxplot(df1['daily_decr90'])
]: <AxesSubplot:xlabel='daily_decr90'>
```

Select the snip mode usi
button.

Snipping Too

In a future update
home. Try improv
& Sketch (or try t
Windows logo ke

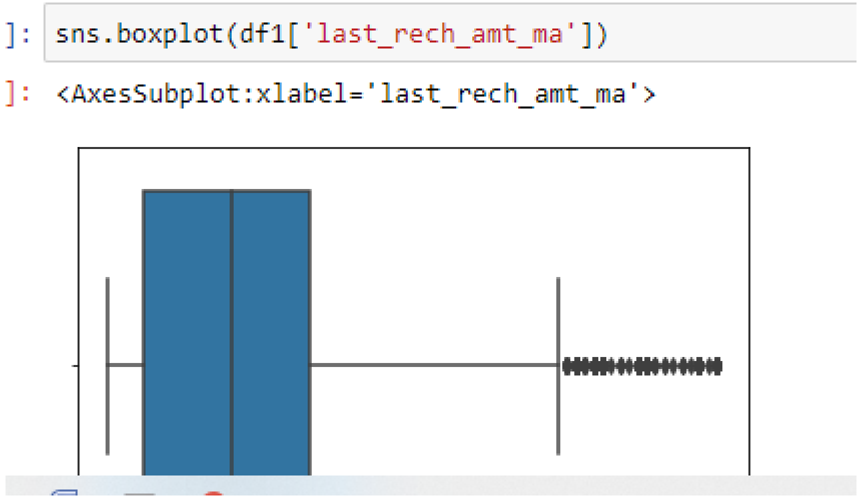
Observation – As per graphical view outlier s not present



```
40]: sns.boxplot(df1['last_rech_date_ma'])
```

```
40]: <AxesSubplot:xlabel='last_rech_date_ma'>
```

Observation – As per graphical view outlier s present , as some data present above vertices .



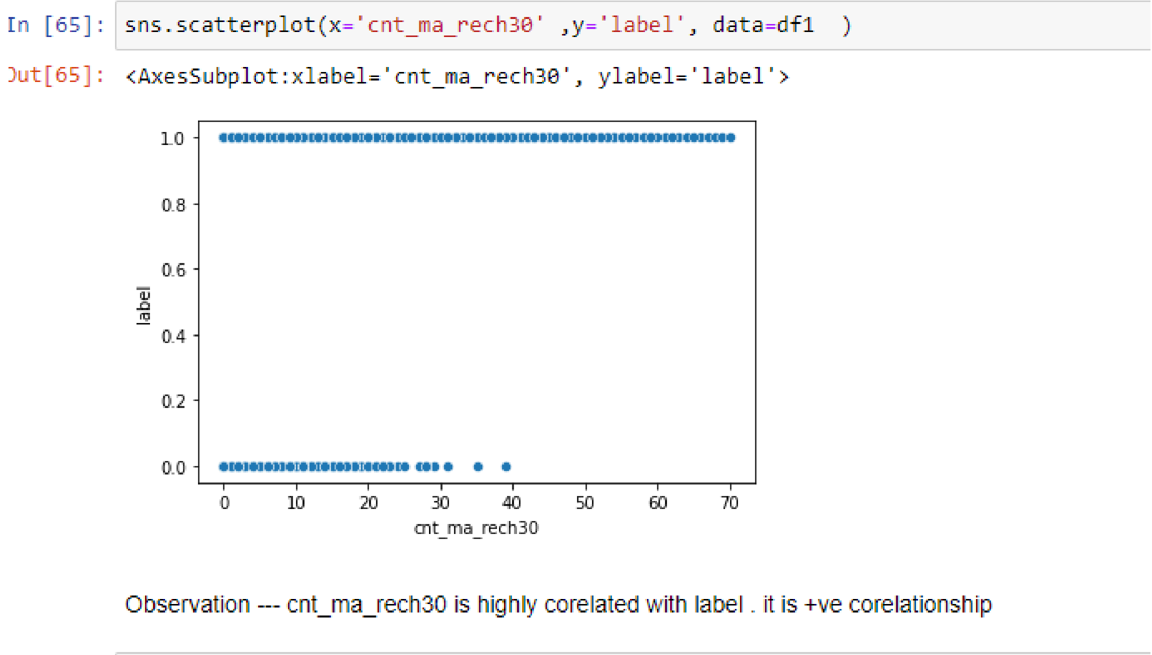
```
]: sns.boxplot(df1['last_rech_amt_ma'])
```

```
]: <AxesSubplot:xlabel='last_rech_amt_ma'>
```

Observation – As per graphical view outlier s not present .

Bivariate---- From df.corr () we get correlationship value, from there we found which variable is highly correlated with each other and which are negatively correlated. Here we plotted graphical representation using scatter plot.

- sns.scatterplot(x='cnt_ma_rech30' ,y='label', data=df1)

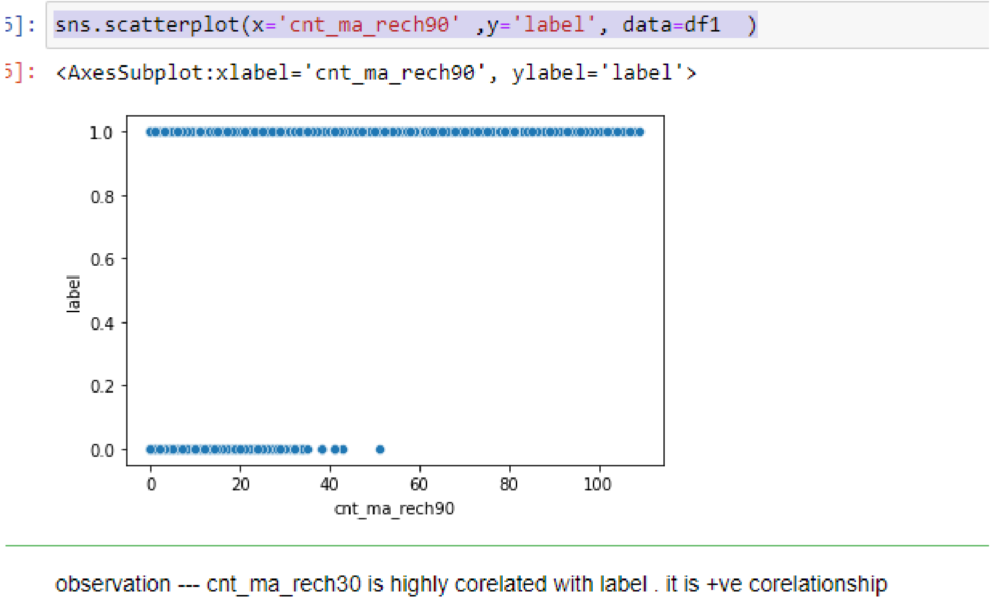


```
In [65]: sns.scatterplot(x='cnt_ma_rech30' ,y='label', data=df1  )
Out[65]: <AxesSubplot:xlabel='cnt_ma_rech30', ylabel='label'>
```

Observation --- cnt_ma_rech30 is highly corelated with label . it is +ve corelationship

Observation --- cnt_ma_rech30 is highly corelated with label . it is +ve correlationship

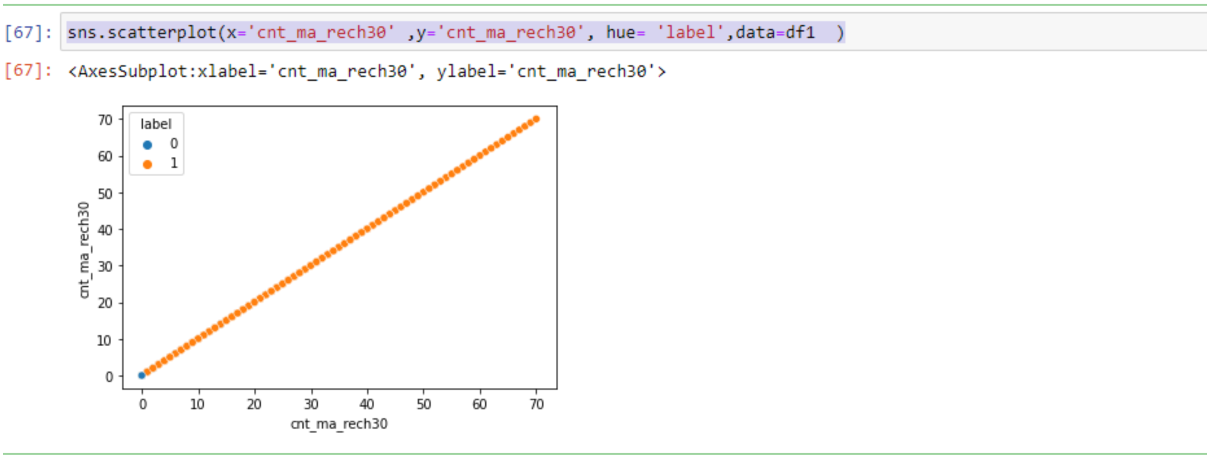
- sns.scatterplot(x='cnt_ma_rech90' ,y='label', data=df1)



```
5]: sns.scatterplot(x='cnt_ma_rech90' ,y='label', data=df1  )
5]: <AxesSubplot:xlabel='cnt_ma_rech90', ylabel='label'>
```

observation --- cnt_ma_rech30 is highly corelated with label . it is +ve corelationship

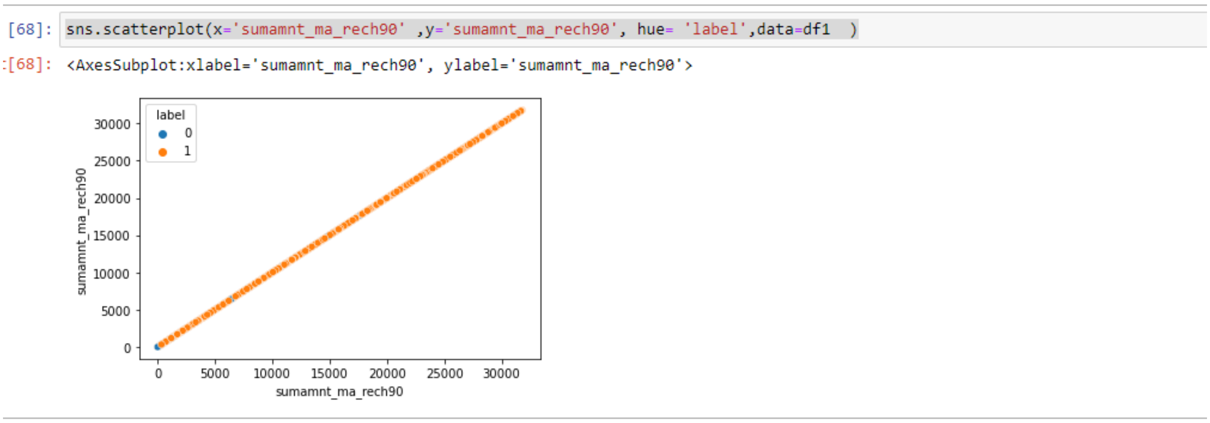
- sns.scatterplot(x='cnt_ma_rech30' ,y='cnt_ma_rech30', hue= 'label',data=df1)



```
[67]: sns.scatterplot(x='cnt_ma_rech30' ,y='cnt_ma_rech30', hue= 'label',data=df1  )
[67]: <AxesSubplot:xlabel='cnt_ma_rech30', ylabel='cnt_ma_rech30'>
```

Observation :x='cnt_ma_rech30' ,y='cnt_ma_rech30' both are having highly co-relatedwith each other , we tried to plot on base of label

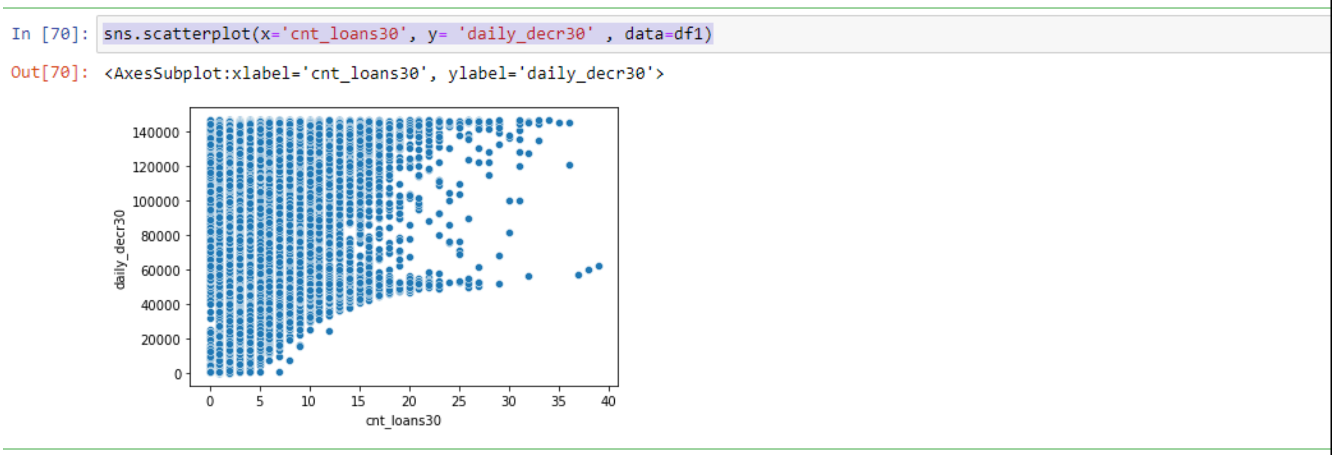
sns.scatterplot(x='sumamnt_ma_rech90' ,y='sumamnt_ma_rech90', hue= 'label',data=df1)



```
[68]: sns.scatterplot(x='sumamnt_ma_rech90' ,y='sumamnt_ma_rech90', hue= 'label',data=df1  )
:[68]: <AxesSubplot:xlabel='sumamnt_ma_rech90', ylabel='sumamnt_ma_rech90'>
```

observation--- (x='sumamnt_ma_rech90' ,y='sumamnt_ma_rech90' there to variable are highly corelated with each other as well as thet are having +ve corelationship with target varibale label also

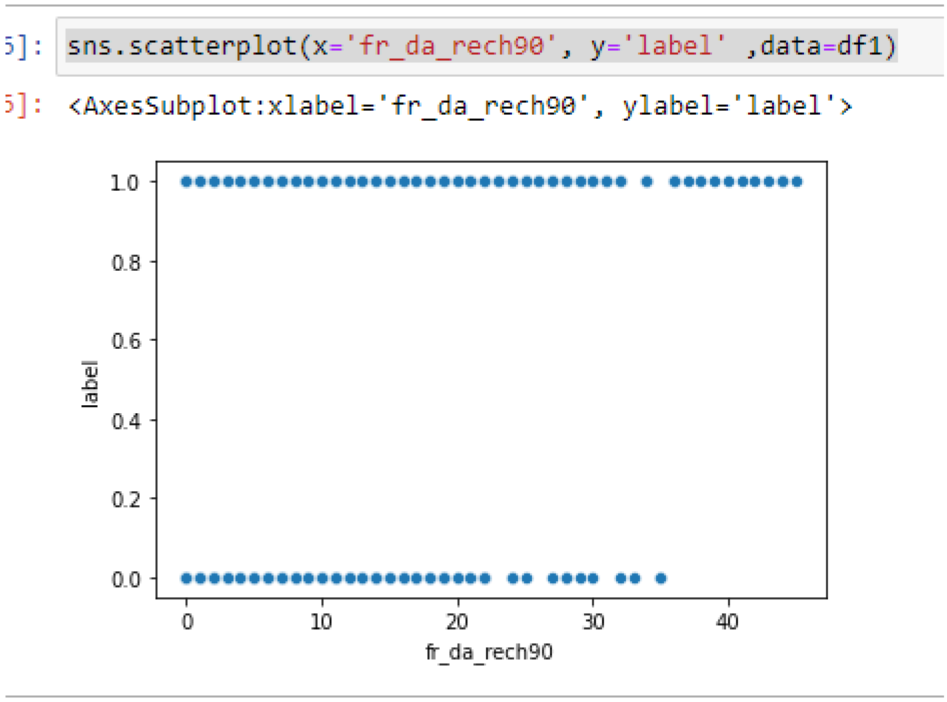
sns.scatterplot(x='cnt_loans30', y= 'daily_decr30' , data=df1)



```
In [70]: sns.scatterplot(x='cnt_loans30', y= 'daily_decr30' , data=df1)
Out[70]: <AxesSubplot:xlabel='cnt_loans30', ylabel='daily_decr30'>
```

Observation----x='cnt_loans30', y= 'daily_decr30' these two are highly corelated with each other , one increase other will get automatically increased

sns.scatterplot(x='fr_da_rech90', y='label' ,data=df1)

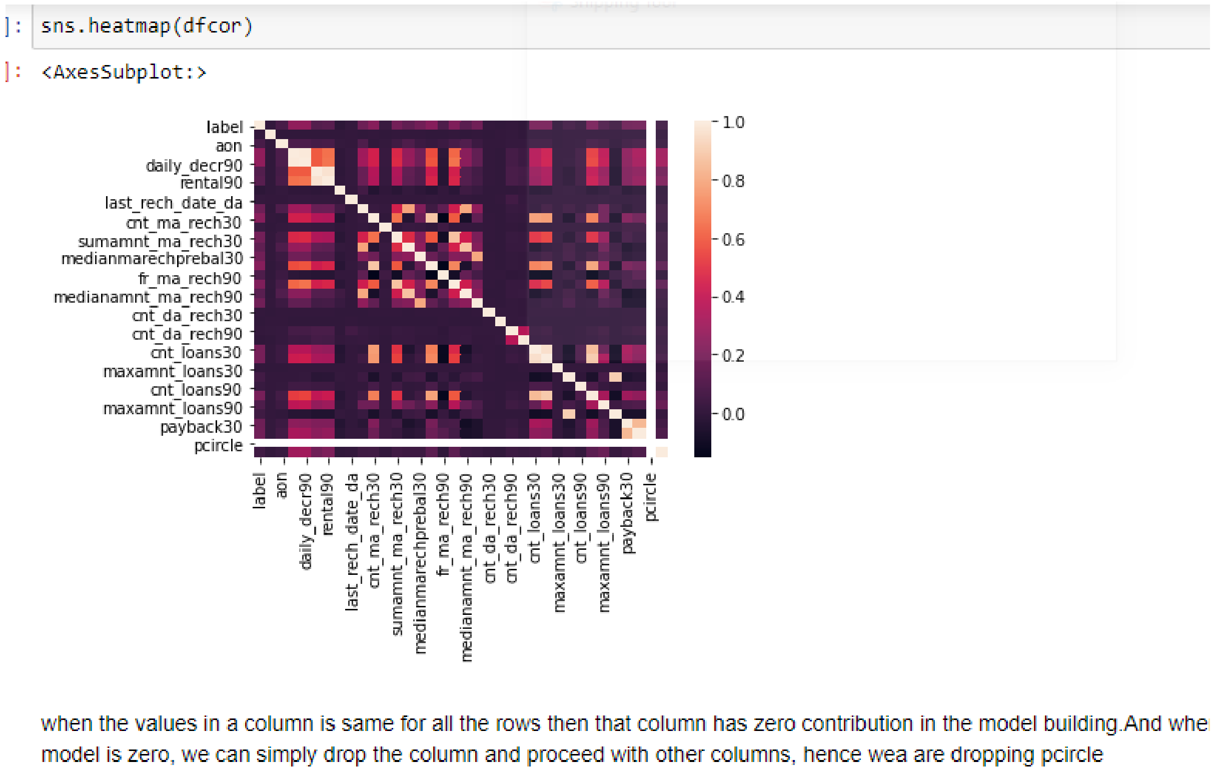


```
5]: sns.scatterplot(x='fr_da_rech90', y='label' ,data=df1)
5]: <AxesSubplot:xlabel='fr_da_rech90', ylabel='label'>
```

Observation-- As per calculation we plotted -ve corelationship

Multivariate analysis—

We plot heatmap and pairplot to get multiplot idea



```
]: sns.heatmap(dfcor)
```

```
]: <AxesSubplot:>
```

when the values in a column is same for all the rows then that column has zero contribution in the model building.And whe
model is zero, we can simply drop the column and proceed with other columns, hence wea are dropping pcircle

.

Testing of Identified Approaches (Algorithms)

We have performed train test where we have send data to model (some data for training and some for testing). We have used 5 model to

- DecissionTree Classifier Model
- Random Forest Model
- Ada-boost Model
- KNeighbour Model
- SVC Model

ALGORITHIM

DecissionTree Classifier Model:

```
dtc=DecisionTreeClassifier()

dtc.fit(x_train,y_train)

preddtc=dtc.predict(x_test)

print ("acccuracy score" , accuracy_score(y_test,preddtc))

print("confusion matrix", confusion matrix(y_test,preddtc))

print("clasification report",classification_report(y_test,preddtc))
```

Output-

```
acccuracy score 0.8969148804361293
```

RandomForestClassifier

```
from sklearn.ensemble  import  RandomForestClassifier


rf=RandomForestClassifier( n_estimators=100, random_state=42)
rf.fit(x_train, y_train)
predrf=rf.predict(x_test)
print(accuracy_score (y_test,predrf))
print(confusion_matrix(y_test, predrf))
print(classification_report(y_test,predrf))
```

```
Output of accuracy score =
0.9449014991946475
```

Ada-boost Model

```
ad=AdaBoostClassifier( n_estimators=50)
ad.fit(x_train, y_train)
adprd=ad.predict(x_test)
print(accuracy_score(y_test,adprd))
print(confusion_matrix(y_test, adprd))
print(classification_report(y_test,adprd))
```

Output of accuracy score-0.8549002601908066

KNeighborsClassifier

```
kmn=KNeighborsClassifier(n_neighbors=5)
kmn.fit(x_train, y_train)
kmnpred=kmn.predict(x_test)
print("accuracy score is",accuracy_score(y_test,kmnpred))
print("confusion matrix", confusion_matrix(y_test,kmnpred))
print("classification report",classification_report(y_test,kmnpred))
```

Output of accuracy score is -0.8723949944244828

SVC model

```
from sklearn.svm import LinearSVC

clf = LinearSVC(random_state=0, tol=1e-5)

clf.fit(x_train, y_train.ravel())
predsvc=sv.predict(x_test)
print(sv.score(x_train,y_train.ravel()))
print("acccuracy score" , accuracy_score(y_test,predsvc))
print("confusion matrix", confusion_matrix(y_test,predsvc))
print("clasification report",classification_report(y_test,predsvc))
```

```
acccuracy score 0.8863636363636364
```

Predictions on multiple metrics ROC-AUC curve

The ROC curve shows the trade-off between sensitivity (or TPR) and specificity (1 – FPR). Classifiers that give curves closer to the top-left corner indicate a better performance. As a baseline, a random classifier is expected to give points lying along the diagonal (FPR = TPR).

AUC ROC score:

AUC means area under the curve so to speak about ROC AUC score we need to define ROC curve first. It is a chart that visualizes the trade-off between true positive rate (TPR) and false positive rate (FPR). ... The more top-left your curve is the higher the area and hence higher ROC AUC score

Using” model. Predict .proba function we analyse all the model roc_auc score and curve   we got Random foreset classifier having good roc_auc score and the more top-left your curve is the higher the area

RFC Model

```
y_pred_proba= rf.predict_proba(x_test)[:,1]
fpr,tpr,thresholds=roc_curve(y_test, y_pred_proba)
plt.plot([0,1],[0,1],'k--')
plt.plot(fpr,tpr,label='RFclassification')
plt.xlabel('false positive rate')
plt.ylabel('true positive rate')
plt.title('RF')
plt.show()
auc_score=roc_auc_score(y_test,rf.predict(x_test))
 print(auc_score)
```

Best model selection

We have calculated cross validation score of each model. Cross-validation is a statistical method used to estimate the skill of machine learning models and we found RandomForestClassifier has having less difference between accuracy and cross validation score .So as per logic RFC is our best model

Conclusion

- we have transform categorical data to numeric using Label Encoder
- We have plotted graphical view of each column to understand data distribution , kernel density, as well as for finding outlier concept
- As this is categorical data we cannot remove outlier as mean concept not there in categorical data , same confirmed by Data Trained mentor

- we drop ‘pcircle’ as all the value of this column is same so it don’t contribute any information to the dataset for analysis
- we divided data x and y as a data and target
- we analysis all the model and found only RFC is having less difference between accuracy and cross_val_score
- same confirmed by plotting roc_curve and determine roc_auc score
- We optimize model using hyper tuning parameter (hyper parameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. These measures are called hyperparameters, and have to be tuned so that the model can optimally solve the machine learning problem)
- We got our final model
- We saved out final model in as .pkl file as per client requirement . It is basically Binary format of output

Limitation:-

- The data could be incomplete. even the lack of a section or a substantial part of the data, could limit its usability.
- We don’t get always accurate information as data might be not completed .
- As it is real time data , it is complex data, took long time to execute