MICRO CREDIT PROJECT

Submitted by:

VISHWA NARAYAN SAINI

INTERNSHIP-33

ACKNOWLEDGEMENT

I would like to extend my sincere gratitude to [FLIP ROBO TECHNOLOGY] for their invaluable contributions to this project. Their expertise and support were instrumental in the completion of this project. I would also like to thank Mr. Shwetank Mishra (SME FOR BATCH-33) for providing us with the necessary resources to complete this project, Further, some more analysis is done through the research papers to know the insights of these institutions, and they are as listed below:

1) ***Competition and microfinance institutions' performance: evidence from India***

- Hailu Abebe Wondirad

2) ***GROWTH OF MICRO-CREDIT IN INDIA: AN EVALUATION***

Prof. Mohammad Tarique

3) Research Study about the Role of Microfinance Institutions in the Development of Entrepreneurs

-Sowmyan Jegatheesan, Sakthi Ganesh, and Praveen Kumar S

INTRODUCTION

- **Business Problem Framing**

Micro-financial institutions, such as small-scale banking organizations, cooperatives, and credit unions, can be framed as a business problem by looking at the various aspects of the financial services they offer. By understanding the objectives, clients, and strategies required to deliver financial services to their target market, micro-financial institutions can develop an effective business model.

The objectives of these institutions should include providing the right type of financial services, such as savings accounts, lending, and insurance, to consumers in their target market. Additionally, they need to ensure that they are cost-effectively offering these services while maintaining compliance with relevant regulations.

To deliver these services, micro-financial institutions need to identify potential clients and understand their needs. This includes understanding their financial backgrounds, such as income levels, credit scores, and other financial indicators, as well as their preferences for financial services. Additionally, micro-financial institutions need to develop relationships with potential clients and provide support to existing clients in the form of education and financial advice.

Finally, micro-financial institutions need to have a strategy in place for acquiring and managing clients, as well as for managing their operations. This includes developing effective marketing and sales strategies, as well as designing and implementing appropriate policies and procedures. Additionally, they need to ensure that they are keeping up with the changing business environment and leveraging technology to remain competitive.

By framing micro-financial institutions as a business problem and looking at the various aspects of financial services they offer, they can develop effective strategies and business models to ensure their long-term success.

- **Conceptual Background of the Domain Problem**

1) MFIs broadly focus on the groups which are less privileged in terms of conventional financial services like loans, insurance, fund transfers, and many more.
2) Increasing the reach of such a portion of people can effectively reduce the cost and increase the efficiency of lending institutions.
3) Analysing the behavior of loanees of such institutions is among the major part of the problem in our case study.
4) Based on the analysis the MFIs can make new plans and be able to predict if a loanee is going to be a defaulter or not.
5) Data like the Average number of days a person takes to refund the loan, the average account balances a person had, the frequency of loans taken in 30 and 90 days, and many more will be in the dataset.

- **Review of Literature**

According to the literature we used for understanding this problem, we can estimate this as a business problem on various factors:

Microfinance institutions have traditionally served as a source of financial access for small businesses and individuals throughout the world. The sector has experienced tremendous growth in recent years, particularly in developing countries, and continues to have a major impact on the global economy.

However, despite the positive aspects of microfinance, it is not without its challenges. A major issue facing the sector is the high degree of competition in the marketplace, with a proliferation of types of financial services being offered by a myriad of providers. This can lead to a "race to the bottom" in terms of prices and services offered, making it difficult for providers to remain profitable. In addition, the sustainability of the sector is questionable due to a lack of long-term, low-cost capital, as well as inadequate monitoring and assessment of the sector.

In this review of literature, the focus will be on the business challenges facing microfinance institutions, including competition in the market, lack of capital, and monitoring and assessment. Research from both developed and developing countries will be explored, as well as potential solutions for these issues. The goal is to provide a comprehensive overview of the challenges and potential solutions within the microfinance sector and to suggest strategies to address these issues in the future.

- **Motivation for the Problem Undertaken**

    - As we can see the trends in the field of MFI sectors and the reach they have been continuously increasing over the years, especially in developing countries, is one of the biggest motivations for choosing this problem.
    - The scope in this field is wide and can be a good help to the Self Help Groups, NGOs, and various institutions who are aiming to enter this domain.
    - Betterment of society especially the weaker sections who lacks the basic financial advantages which can directly impact the growth of the nation.
    - Creation of employment among these regions thus solving a major issue further motivates to take this problem

Analytical Problem Framing

Analytical and mathematical modeling can be used to frame business problems for micro-financial institutions. This includes assessing risk levels and creating mathematically sound models to evaluate the performance of microfinance institutions. To start, it is important to identify the variables that are relevant to the problem and develop a hypothesis. It is also important to consider the economic, legal, and social implications of microfinance institutions.

Next, an appropriate model needs to be designed that can capture the complexities and interactions of the variables. This can include using regression models, decision trees, or other machine-learning techniques. These models can then be used to test the hypothesis and evaluate potential solutions. Finally, the results of the modeling need to be interpreted and used to develop strategies and policies for microfinance institutions.

- Data Sources and their formats

Data sources for microfinance projects can come from a variety of sources, including the Microfinance Information Exchange (MIX) Market database [1], which contains self-reported data on financial service providers (FSPs) and microfinance institutions (MFIs). This includes data on financial statements, operations, financial products, end clients, and social performance. Additionally, trade repositories and third-party data sources such as institutional and commercial data can be utilized [2], as well as Big Data from the Internet [3]. Micro-databases have also been used, offering data on loans, credit, products, resources, and services in the microfinance industry [2]. By opening access to this high-value repository of essential information,

underserved and low-income communities can benefit from increased transparency in the microfinance industry.

DATA DESCRIPTION!!

]:

	Variable	Definition	Comment
0	label	Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}	NaN
1	msisdn	mobile number of user	NaN
2	aon	age on cellular network in days	NaN
3	daily_decr30	Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)	NaN
4	daily_decr90	Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)	NaN
5	rental30	Average main account balance over last 30 days	Unsure of given definition
6	rental90	Average main account balance over last 90 days	Unsure of given definition
7	last_rech_date_ma	Number of days till last recharge of main account	NaN
8	last_rech_date_da	Number of days till last recharge of data account	NaN
9	last_rech_amt_ma	Amount of last recharge of main account (in Indonesian Rupiah)	NaN
10	cnt_ma_rech30	Number of times main account got recharged in last 30 days	NaN
11	fr_ma_rech30	Frequency of main account recharged in last 30 days	Unsure of given definition
12	sumamnt_ma_rech30	Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)	NaN
13	medianamnt_ma_rech30	Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)	NaN
14	medianmarechprebal30	Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)	NaN
15	cnt_ma_rech90	Number of times main account got recharged in last 90 days	NaN
16	fr_ma_rech90	Frequency of main account recharged in last 90 days	Unsure of given definition
17	sumamnt_ma_rech90	Total amount of recharge in main account over last 90 days (in Indonasian Rupiah)	NaN
18	medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)	NaN
		Median of main account balance just before recharge in last 90 days at user level	

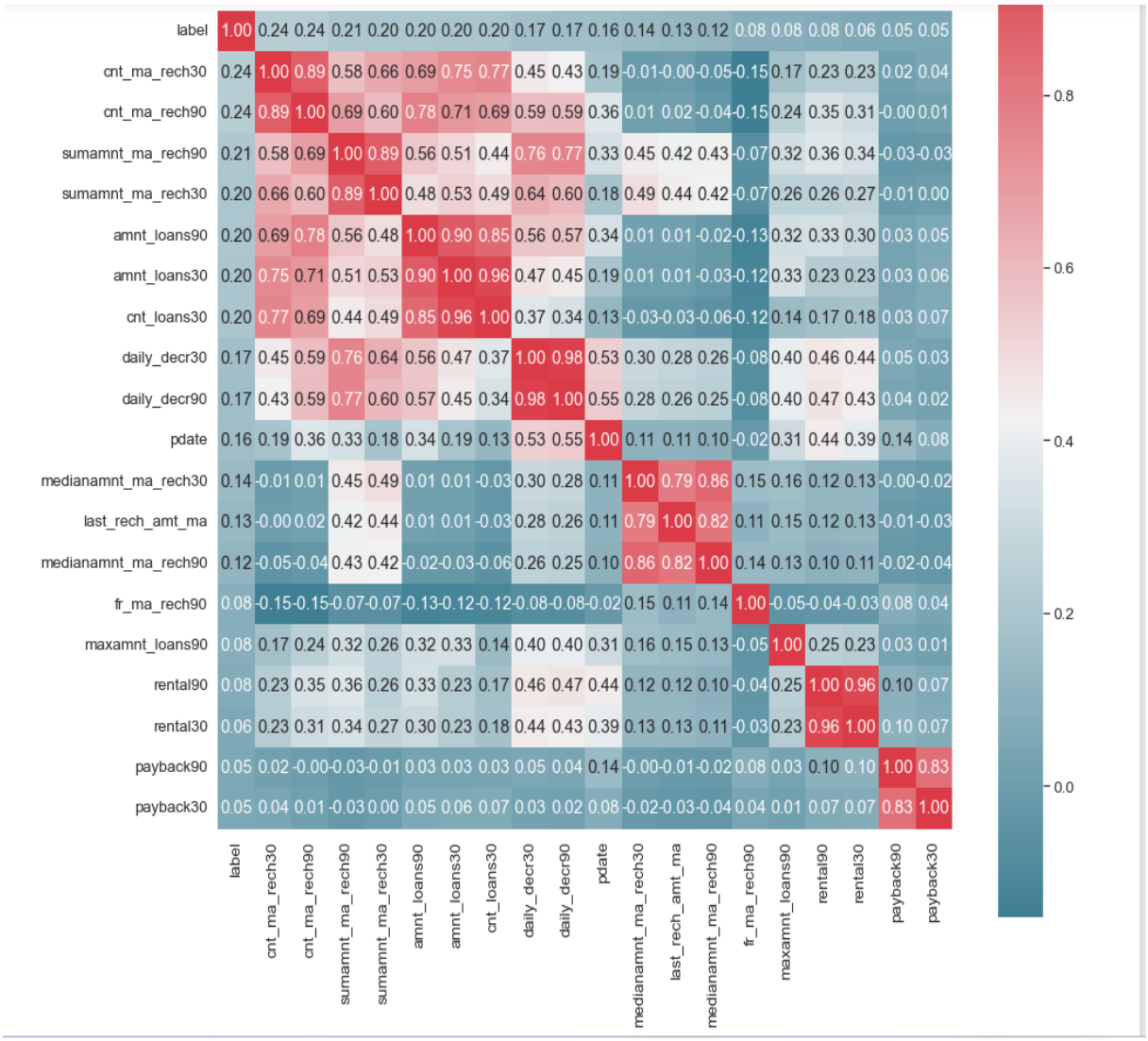
18	medianamnt_ma_rech90	Median of amount of recharges done in main account over last 90 days at user level (in Indonasian Rupiah)	NaN
19	medianmarechprebal90	Median of main account balance just before recharge in last 90 days at user level (in Indonasian Rupiah)	NaN
20	cnt_da_rech30	Number of times data account got recharged in last 30 days	NaN
21	fr_da_rech30	Frequency of data account recharged in last 30 days	NaN
22	cnt_da_rech90	Number of times data account got recharged in last 90 days	NaN
23	fr_da_rech90	Frequency of data account recharged in last 90 days	NaN
24	cnt_loans30	Number of loans taken by user in last 30 days	NaN
25	amnt_loans30	Total amount of loans taken by user in last 30 days	NaN
26	maxamnt_loans30	maximum amount of loan taken by the user in last 30 days	There are only two options: 5 & 10 Rs., for which the user needs to pay back 6 & 12 Rs. respectively
27	medianamnt_loans30	Median of amounts of loan taken by the user in last 30 days	NaN
28	cnt_loans90	Number of loans taken by user in last 90 days	NaN
29	amnt_loans90	Total amount of loans taken by user in last 90 days	NaN
30	maxamnt_loans90	maximum amount of loan taken by the user in last 90 days	NaN
31	medianamnt_loans90	Median of amounts of loan taken by the user in last 90 days	NaN
32	payback30	Average payback time in days over last 30 days	NaN
33	payback90	Average payback time in days over last 90 days	NaN
34	pcircle	telecom circle	NaN
35	pdate	date	NaN

- # Data Pre-processing Done

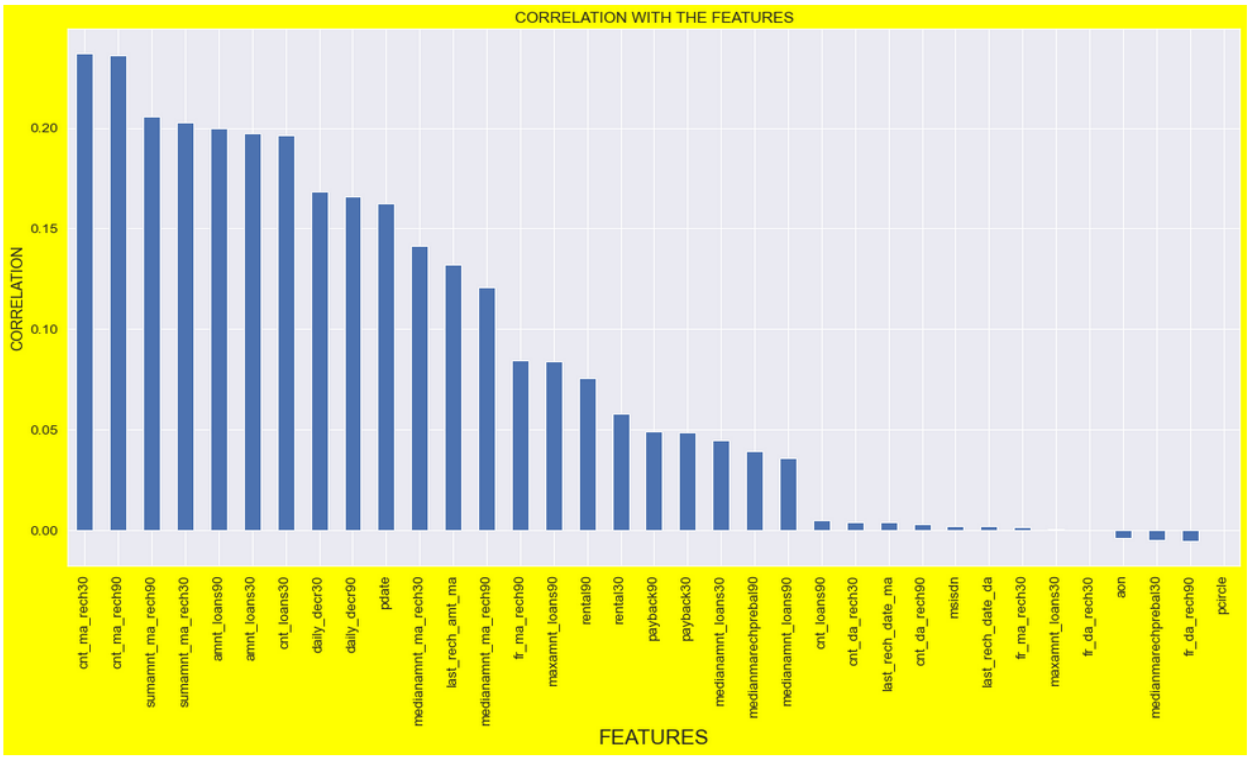
  As we can see there are no nulls present in the dataset. We do not have to engage much in cleaning the dataset.
  Sorting the date using the sorted function, after sorting the date is encoded ordinally.
  Using a label encoder to encode the pcircle and msisdn.

- ## Data Inputs- Logic- Output Relationships

ANALYSIS

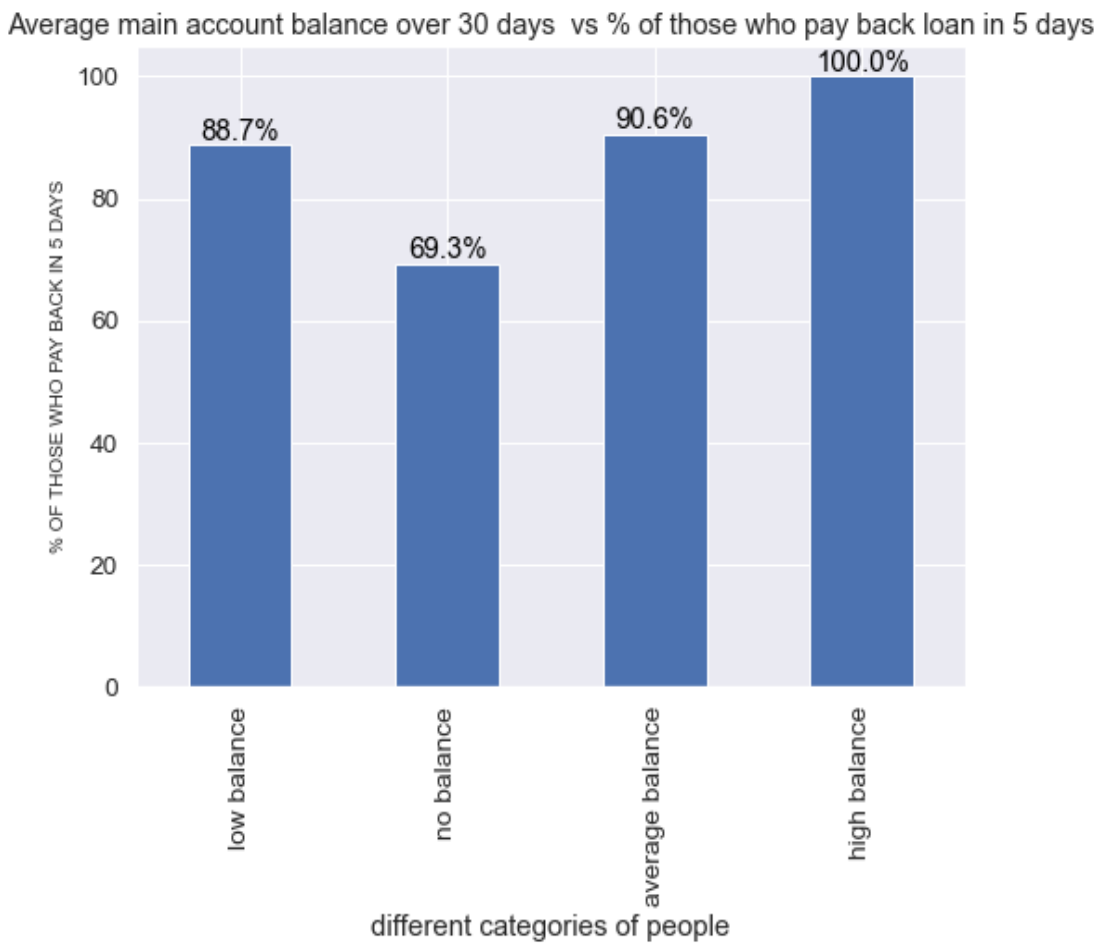


Above is the correlation chart for the best 20 features among all the features.
The correlation coefficient of the top 20 features can be depicted in the above plot.

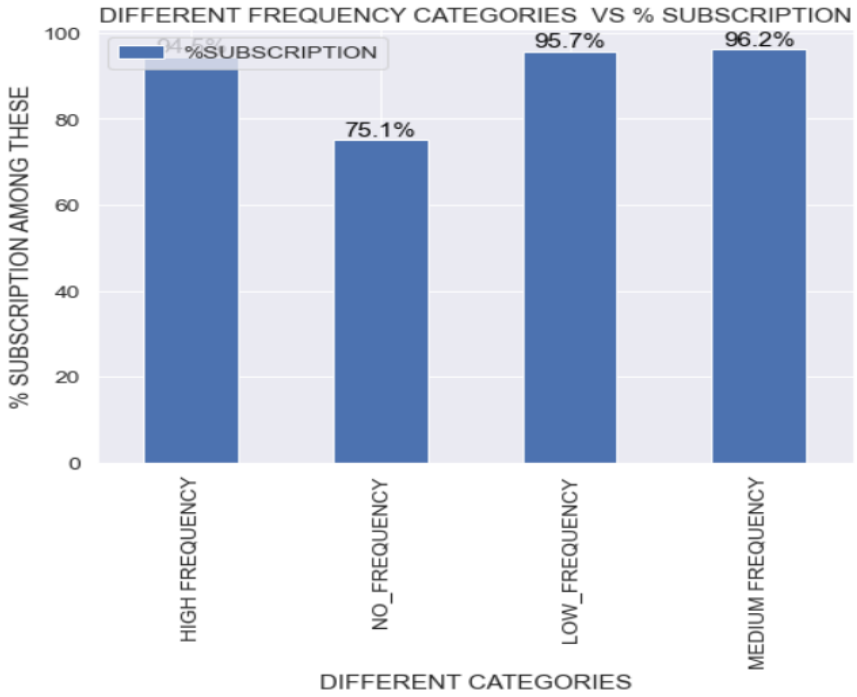


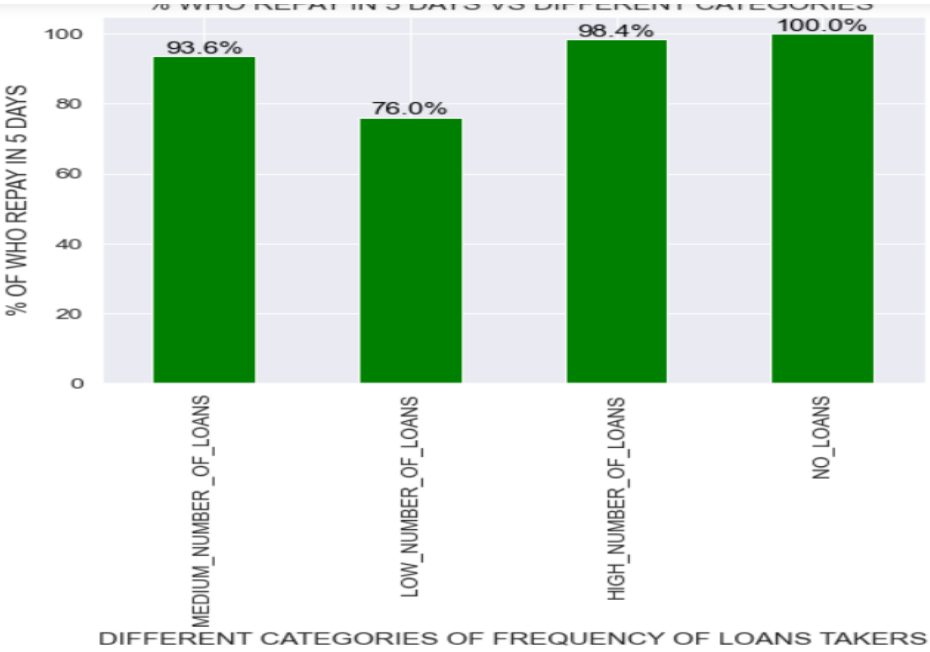
Correlation with the label “label” which describes whether the customer was able to repay the loans or not.

The count of the maximum number of recharges in the period of 30 days can be seen as having the maximum correlation with the label.



Different categories of people with different percentages can be seen in the above plot.
Those having a high balance are seen as having a repayment rate of 100%.
Those who have no balance are most likely to repay the loan with a probability of 0.70.
Strategies that can pull the defaulters into the non-defaulters' categories are the most important in these times.

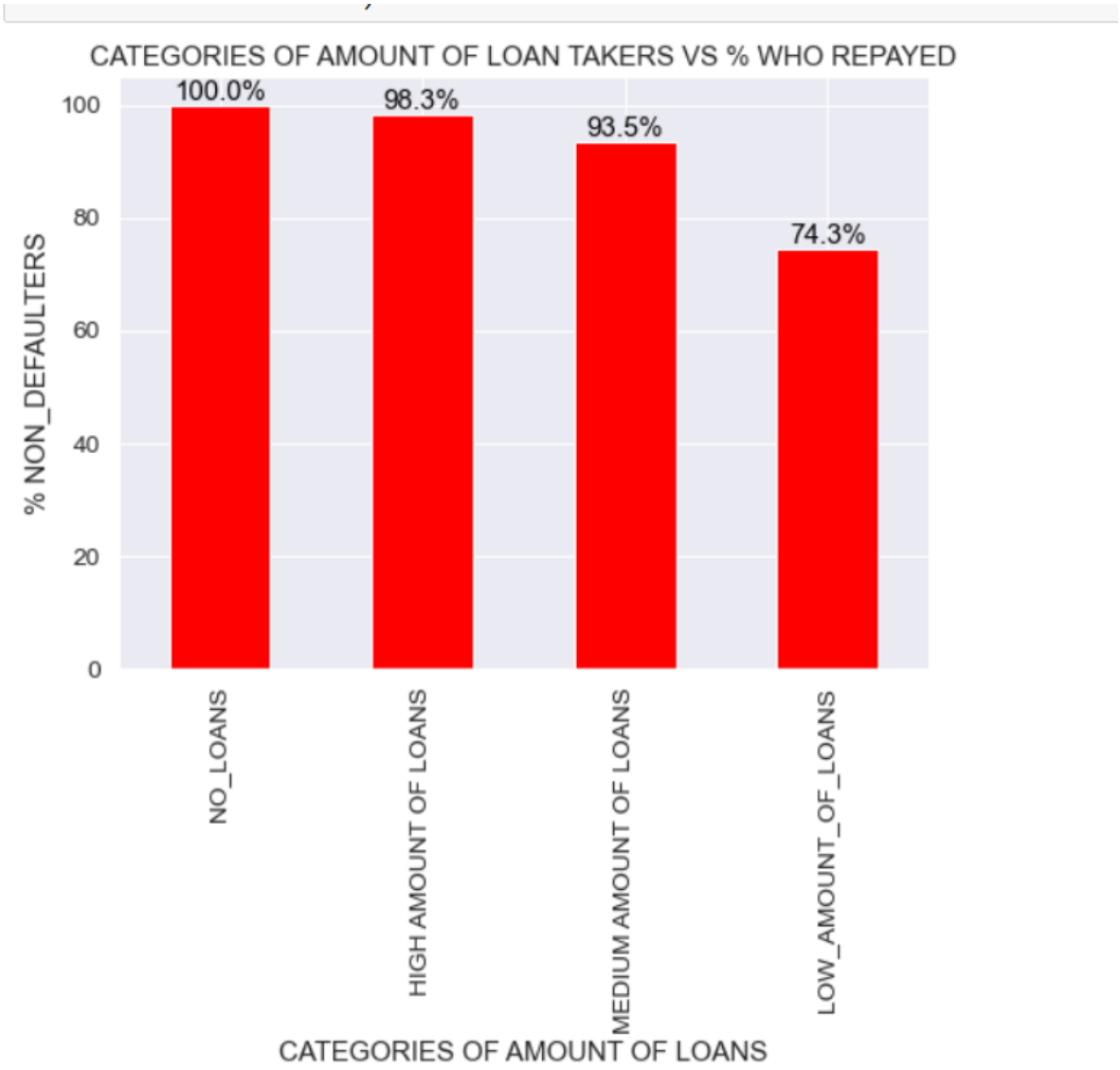




OF THOSE WHO HAVE HIGH AMOUNTS OF LOANS AMONG THEM 98.3% OF TEND TO REPAY THEIR LOANS IN 5 DAYS!!
THOSE IN THE CATEGORY OF LOW_AMOUNT_OF_LOANS HAVE RATES OF ONLY 74.3% REPAYING THE LOANS IN 5 DAYS!!

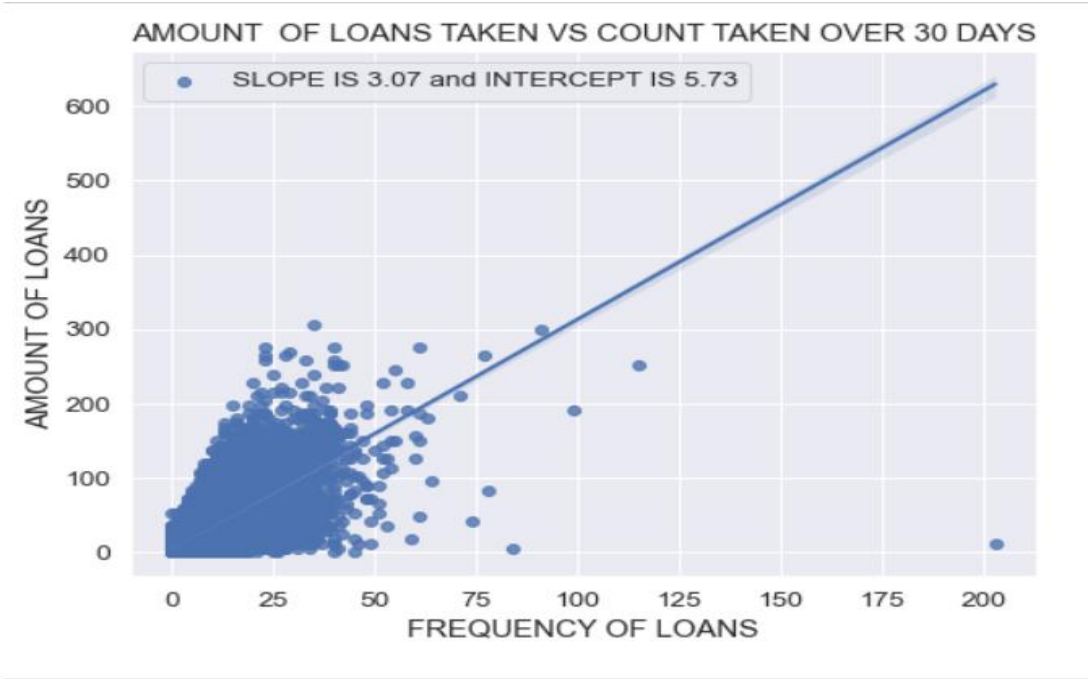
NEW STRATEGIES MUST BE INTRODUCED TO ENCOURAGE PEOPLE TO REPAY LOANS ON TIME!!

SMS, EMAILS, AND COLD CALLS CAN BE SOME OF THE WAYS TO GET IN TOUCH WITH THE CUSTOMERS!

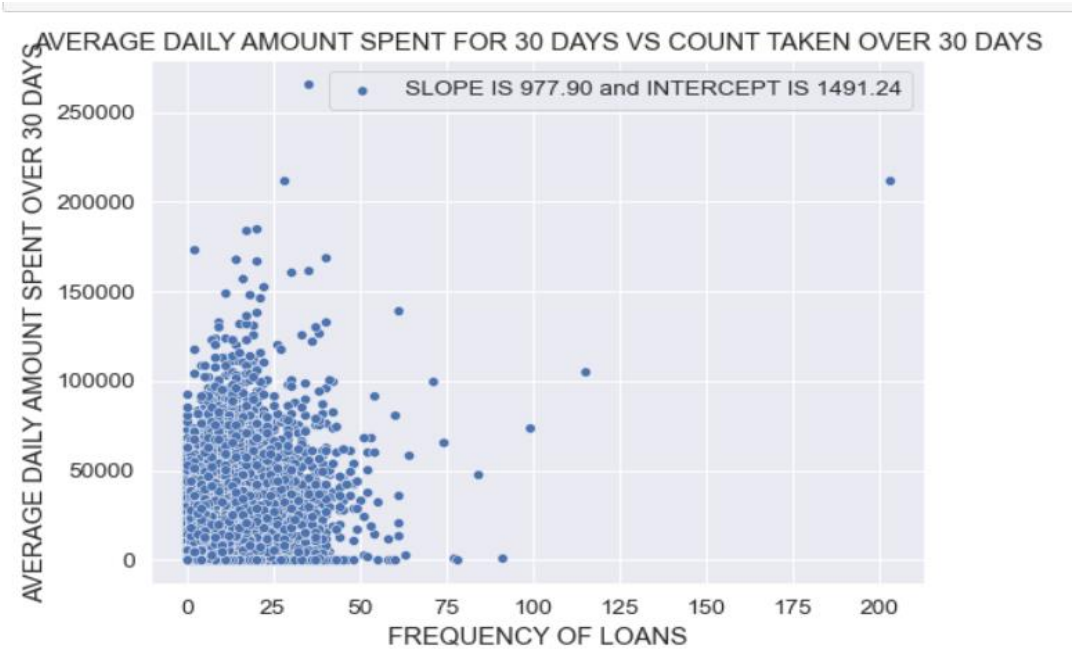


Customers who are taking many loans are seen paying it back with a probability of 0.98 as compared to those who take a medium amount of loans have a repayment probability of 0.93.
New ways should be encouraged which motivate the loanees to repay it within 5 days i.e.
Having a reward system for those who were able to repay the loans within the limit of 5 days the frequent number of times.

Increased limit of number and amount of loans the customer can take as a reward for those who repay it within 5 days.
Making customers aware of the benefits and rewards they can access if they follow all the rules strictly.



It can be seen that between the relation of frequency and amount of loans, a considerable relationship can be seen from the above plot.
With a slope and intercept of 3.07 and 5.73 respectively the plot can be judged above.



The relation between the frequency and the average daily amount spent over 30 days can be seen from the above plot. Though no strong and significant relationship can be seen between the two we can overrule the chances of a connection between the two.

- **Hardware and Software Requirements and Tools Used**

Hardware requirements for a machine learning model typically include a computer with a powerful CPU and GPU, as well as sufficient memory and storage capacity. The specific requirements will depend on the complexity of the model and the size of the dataset. Software requirements usually include a programming language such as Python or R, as well as various libraries and frameworks such as TensorFlow, PyTorch, or scikit-learn for implementing the model.

Model/s Development and Evaluation

- **Identification of possible problem-solving approaches (methods)**

Several analytical and statistical approaches can be used to solve a flight price prediction machine learning problem.

1. Linear Regression: Linear regression is a simple and widely used statistical method for predicting the relationship between a dependent variable and one or more independent variables.
2. Decision Tree: A decision tree is a popular algorithm for both classification and regression problems. It can handle both categorical and numerical data.
3. Random Forest: Random Forest is an ensemble method that combines multiple decision trees to improve the overall performance of the model.
4. Gradient Boosting: Gradient Boosting is also an ensemble method that combines multiple weak models to form a stronger model.
5. Neural Networks: Neural networks are a class of machine learning models that are inspired by the structure and function of the human brain.
6. Time Series Analysis: Time Series Analysis is used to analyze and predict time-related data, such as flight prices. Techniques such as ARIMA, Exponential Smoothing, and LSTM can be used to analyze time series data.

The choice of the analytical and statistical approach will depend on the specific characteristics of the dataset and the problem.

- **Testing of Identified Approaches (Algorithms)**

1) LogisticRegression
2) KNN
3) Decision Tree
4) RandomForestClassifier

5) AdaBoostClassifier
6) BaggingClassifier
7) GradientBoostingClassifier

- **Key Metrics for success in solving the problem under consideration**

- Classification problems are common in Machine Learning and require the selection of an appropriate model to solve them. The selection criteria will depend on the type of problem and the desired outcome. Generally speaking, the selection criteria for classifying problems are based on metrics such as accuracy, precision, recall, F1 score, and others.
- Accuracy is the ratio of correctly classified to all instances and demonstrates how well a model can predict correct categories. Precision is the ratio of true positives and overall positive predictions and is useful for measuring how accurate the model is for each category. The recall is the ratio of true positives overall actual positive cases and helps to identify the model's ability to identify all positive cases. The F1 score is a combination of precision and recall and is a measure of the model's overall performance.
- In addition to these metrics, the selection criteria will also depend on other aspects, such as the type of data and the complexity of the model. For example, a more complex model may have a higher accuracy score, but if it is too computationally expensive, it may not be suitable for the task. Therefore, in addition to evaluating the metrics, the selection criteria should consider the cost and complexity of the model.

- **Defining a function for evaluating the model**

```
def score(model,x_train,y_train,x_test,y_test,train):
    if(train==True):
        print(f"\nFOR MODEL {model}----------------------------------------------------")
        model.fit(x_train,y_train)
        y_pred=model.predict(x_train)
        accuracy=accuracy_score(y_train,y_pred)
        print(f"\n\nACCURACY ON TRAINING DATASET IS {round(accuracy*100,2)}")
    if(train==False):
        y_pred=model.predict(x_test)
        accuracy=accuracy_score(y_test,y_pred)
        print(f"\n\nACCURACY ON TESTING DATASET IS {round(accuracy*100,2)}")
        print('\n\n',classification_report(y_test,y_pred))
        print('\n-------------------------------------------------------------------------------')
```

- **Run and evaluate selected models**

1)Logistic Regression

MODEL 1) LOGISTIC REGRESSION!!

In [112]:
```
from sklearn.model_selection import train_test_split,KFold,cross_val_score
```

In [113]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_curve,roc_auc_score,classification_report,accuracy_score
```

In [114]:
```
for i in range(0,100):
    lr=LogisticRegression()
    x_train,x_test,y_train,y_test=train_test_split(data,df['label'],test_size=0.25,random_state=i)
    lr.fit(x_train,y_train)
    train_pred=lr.predict(x_train)
    train_score=accuracy_score(y_train,train_pred)
    print(f"\nTHE ACCURACY SCORE(TRAINING) FOR RANDOM STATE {i} IS {round(train_score,2)*100}%")
    test_pred=lr.predict(x_test)
    test_score=accuracy_score(y_test,test_pred)
    print(f"\n THE ACCURACY SCORE(TESTING) FOR RANDOM STATE {i} IS {round(test_score,2)*100}%")
    print('\n\n----------------------------------------------------------------------------------------\n\n')
```

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


THE ACCURACY SCORE(TRAINING) FOR RANDOM STATE 4 IS 88.0%

 THE ACCURACY SCORE(TESTING) FOR RANDOM STATE 4 IS 88.0%


----------------------------------------------------------------------------------------


THE ACCURACY SCORE(TRAINING) FOR RANDOM STATE 5 IS 88.0%

 THE ACCURACY SCORE(TESTING) FOR RANDOM STATE 5 IS 88.0%


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

SINCE FOR ALL THE RANDOM STATES WE ARE GETTING A ACCURACY SCORE OF 88% WE WILL BE CONTINUING WITH ANY RANDOM STATE SELECTED AT RANDOM!!

In [115]:
```
x_train,x_test,y_train,y_test=train_test_split(data,df['label'],test_size=0.25,random_state=0)
lr=LogisticRegression()
lr.fit(x_train,y_train)
test_pred=lr.predict(x_test)
```

CROSS VAL SCORES TO CHECK THE MODEL DOES NOT OVER FIT!!

In [116]:
```
for i in range(2,10):
    kfold=KFold(n_splits=i,shuffle=True,random_state=0)
    cv_score=cross_val_score(lr,data,df['label'],cv=kfold).mean()
    print(f"\n\nFOR N_SPLITS=={i} the cross_val_score is {round(cv_score,2)*100}")
```

```
FOR N_SPLITS==2 the cross_val_score is 88.0


FOR N_SPLITS==3 the cross_val_score is 88.0


FOR N_SPLITS==4 the cross_val_score is 88.0


FOR N_SPLITS==5 the cross_val_score is 88.0


FOR N_SPLITS==6 the cross_val_score is 88.0


FOR N_SPLITS==7 the cross_val_score is 88.0


FOR N_SPLITS==8 the cross_val_score is 88.0


FOR N_SPLITS==9 the cross_val_score is 88.0
```

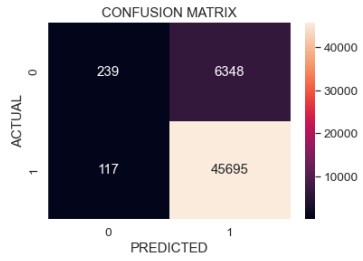
SINCE FOR DIFFERENT N_SPLITS WE ARE GETTING THE SAME CROSS VAL SCORES ITS CLEAR THAT OUR MODEL DOES NOT OVERFIT!!

SINCE FOR DIFFERENT N_SPLITS WE ARE GETTING THE SAME CROSS VAL SCORES ITS CLEAR THAT OUR MODEL DOES NOT OVERFIT!!

In [117]:
```
from sklearn.metrics import confusion_matrix
```

In [118]:
```
cm=confusion_matrix(y_test,test_pred)
```

In [119]:
```
sns.heatmap(cm,annot=True,fmt='d')
plt.title("CONFUSION MATRIX")
plt.ylabel("ACTUAL")
plt.xlabel("PREDICTED")
plt.show()
```



In [120]:
```
tp=cm[1,1]
tn=cm[0,0]
fp=cm[0,1]
fn=cm[1,0]
```

In [121]:
```
accuracy=(tp+tn)/(tp+tn+fp+fn)
precision=(tp)/(tp+fp)
recall=(tp)/(tp+fn)
f1_score=(2*precision*recall)/(accuracy+precision)
```

In [122]:
```
print(f"\nTHE ACCURACY SCORE IS {round(accuracy,4)}")
print(f"\nTHE PRECISION IS {round(precision,4)}")
print(f"\nTHE RECALL IS {round(recall,4)}")
print(f"\nTHE F1_SCORES IS {round(f1_score,4)}")
```

```
THE ACCURACY SCORE IS 0.8766

THE PRECISION IS 0.878

THE RECALL IS 0.9974

THE F1_SCORES IS 0.9982
```

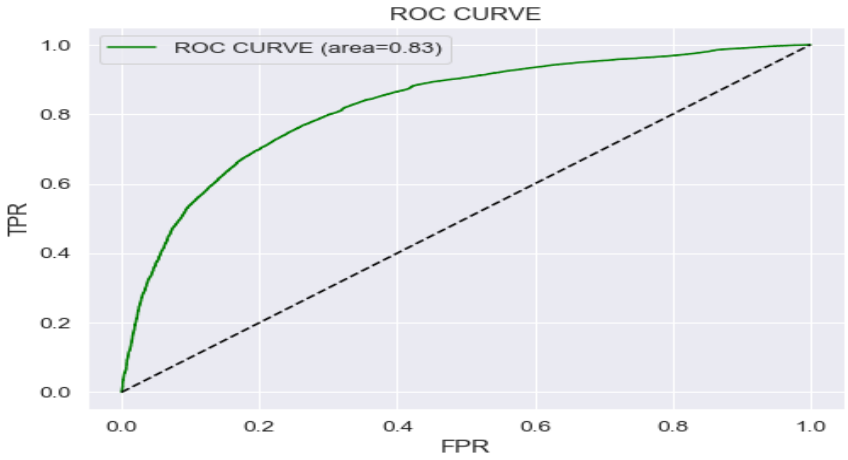
In [123]:
```
y_pred=lr.predict_proba(x_test)
```

In [124]:
```
fpr,tpr,threshold=roc_curve(y_test,y_pred[:,1])
```

In [125]:
```
roc_auc=roc_auc_score(y_test,y_pred[:,1])
```

In [126]:
```
print(f"THE ROC_AUC_SCORE IS {round(roc_auc,4)}")
plt.figure(figsize=(8,6))
plt.plot(fpr,tpr,color='Green',label="ROC CURVE (area=%0.2f)"%roc_auc)
plt.plot([0,1],[0,1],color='black',linestyle='--')
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("ROC CURVE")
plt.legend(loc='best')
plt.show()
```

```
THE ROC_AUC_SCORE IS 0.825
```



2) KNeighborsClassifier

In [133]:
```
score(knn,x_train,y_train,x_test,y_test,train=True)
score(knn,x_train,y_train,x_test,y_test,train=False)
```

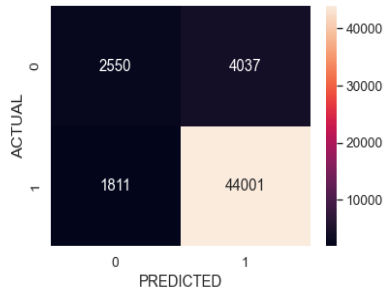
```
FOR MODEL KNeighborsClassifier(n_neighbors=18)----------------------------------------------------

ACCURACY ON TRAINING DATASET IS 89.71

ACCURACY ON TESTING DATASET IS 88.84

              precision    recall  f1-score   support

           0       0.58      0.39      0.47      6587
           1       0.92      0.96      0.94     45812

    accuracy                           0.89     52399
   macro avg       0.75      0.67      0.70     52399
weighted avg       0.87      0.89      0.88     52399


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

In [134]:
```
y_pred=knn.predict(x_test)
```

In [135]:
```
from sklearn.metrics import confusion_matrix
```

In [136]:
```
cm=confusion_matrix(y_test,y_pred)
```

In [137]:
```
sns.heatmap(cm,annot=True,fmt='d')
plt.ylabel("ACTUAL")
plt.xlabel("PREDICTED")
plt.show()
```



In [138]:
```
y_pred=knn.predict_proba(x_test)
```

In [139]:
```
from sklearn.metrics import roc_auc_score,roc_curve
```

In [140]:
```
roc_auc=roc_auc_score(y_test,y_pred[:,1])
```

In [141]:
```
print(f"THE ROC AUC SCORE is {round(roc_auc,2)}")
fpr,tpr,thresholds=roc_curve(y_test,y_pred[:,1])
plt.plot([0,1],[0,1],linestyle='--',color='Black')
plt.plot(fpr,tpr,color='Green')
plt.xlabel("FPR")
plt.ylabel("TPR")
plt.title("TRUE POSITIVITY RATE VS FALSE POSITIVITY RATE CURVE!!",size=10)
plt.show()
```

```
THE ROC AUC SCORE is 0.85
```

3) DecisionTreeClassifier

In [142]:
```
dt=DecisionTreeClassifier()
```

In [143]:
```
score(dt,x_train,y_train,x_test,y_test,train=True)
score(dt,x_train,y_train,x_test,y_test,train=False)
```

```
FOR MODEL DecisionTreeClassifier()----------------------------------------------------


ACCURACY ON TRAINING DATASET IS 100.0


ACCURACY ON TESTING DATASET IS 84.84


                precision    recall  f1-score   support

           0       0.40      0.42      0.41      6587
           1       0.92      0.91      0.91     45812

    accuracy                           0.85     52399
   macro avg       0.66      0.67      0.66     52399
weighted avg       0.85      0.85      0.85     52399


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

USING RANDOMIZED SEARCH CV

In [144]:
```
from sklearn.model_selection import RandomizedSearchCV
```

In [145]:
```
dt.get_params().keys()
```

Out[145]:
```
dict_keys(['ccp_alpha', 'class_weight', 'criterion', 'max_depth', 'max_features', 'max_leaf_nodes', 'min_impurity_decrease
', 'min_samples_leaf', 'min_samples_split', 'min_weight_fraction_leaf', 'random_state', 'splitter'])
```

In [146]:
```
param={"criterion":["gini","entropy","log_loss"],
       "max_depth":np.arange(20),
       "min_samples_split":[1,2,3,4,5,6],
       "max_features":["auto","sqrt","log2"]}
```

In [147]:
```
dt_pscv=RandomizedSearchCV(dt,param_distributions=param,cv=3,scoring='f1_weighted')
dt_pscv.fit(x_train,y_train)
```

Out[147]:
```
RandomizedSearchCV(cv=3, estimator=DecisionTreeClassifier(),
                   param_distributions={'criterion': ['gini', 'entropy',
                                                      'log_loss'],
                                        'max_depth': array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 1
5, 16,
       17, 18, 19]),
                                        'max_features': ['auto', 'sqrt',
                                                         'log2'],
                                        'min_samples_split': [1, 2, 3, 4, 5,
                                                              6]},
                   scoring='f1_weighted')
```

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 [148]:
```
dt_pscv.best_params_
```

Out[148]:
```
{'min_samples_split': 4,
 'max_features': 'sqrt',
 'max_depth': 11,
 'criterion': 'gini'}
```

In [149]:
```
dt=dt_pscv.best_estimator_
```

In [150]:
```
score(dt,x_train,y_train,x_test,y_test,train=True)
score(dt,x_train,y_train,x_test,y_test,train=False)
```

```
FOR MODEL DecisionTreeClassifier(max_depth=11, max_features='sqrt', min_samples_split=
4)----------------------------------------------------


ACCURACY ON TRAINING DATASET IS 89.82


ACCURACY ON TESTING DATASET IS 88.78


              precision    recall  f1-score   support

           0       0.58      0.37      0.45      6587
           1       0.91      0.96      0.94     45812

    accuracy                           0.89     52399
   macro avg       0.75      0.67      0.70     52399
weighted avg       0.87      0.89      0.88     52399


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

4) Random Forest Classifier

In [151]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [152]:
```
rfc=RandomForestClassifier()
```

In [153]:
```
score(rfc,x_train,y_train,x_test,y_test,train=True)
score(rfc,x_train,y_train,x_test,y_test,train=False)
```

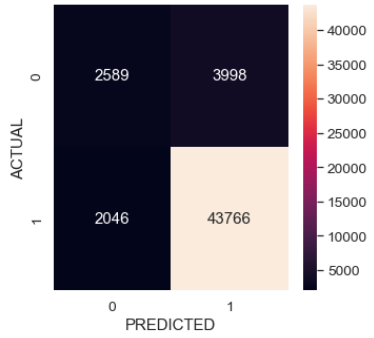
```
FOR MODEL RandomForestClassifier()----------------------------------------------------


ACCURACY ON TRAINING DATASET IS 100.0


ACCURACY ON TESTING DATASET IS 89.58


              precision    recall  f1-score   support

           0       0.65      0.38      0.48      6587
           1       0.92      0.97      0.94     45812

    accuracy                           0.90     52399
   macro avg       0.78      0.67      0.71     52399
weighted avg       0.88      0.90      0.88     52399


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

In [154]:
```
rfc.get_params().keys()
```

Out[154]:
```
dict_keys(['bootstrap', 'ccp_alpha', 'class_weight', 'criterion', 'max_depth', 'max_features', 'max_leaf_nodes', 'max_samp
les', 'min_impurity_decrease', 'min_samples_leaf', 'min_samples_split', 'min_weight_fraction_leaf', 'n_estimators', 'n_job
s', 'oob_score', 'random_state', 'verbose', 'warm_start'])
```

In [155]:
```
rf_var=[]
for val in range(1,50):
    rfc=RandomForestClassifier(n_estimators=val,criterion='gini',random_state=0)
    kfold=KFold(shuffle=True,random_state=0,n_splits=3)
    cross_score=cross_val_score(rfc,x_train,y_train,cv=kfold,n_jobs=-1)
    rf_var.append(np.var(cross_score,ddof=1))
```

In [163]:
```
plt.figure(figsize=(5,5))
sns.heatmap(cm,annot=True,fmt='d')
plt.ylabel("ACTUAL")
plt.xlabel("PREDICTED")
plt.show()
```



In [164]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.56      0.39      0.46      6587
           1       0.92      0.96      0.94     45812

    accuracy                           0.88     52399
   macro avg       0.74      0.67      0.70     52399
weighted avg       0.87      0.88      0.88     52399
```

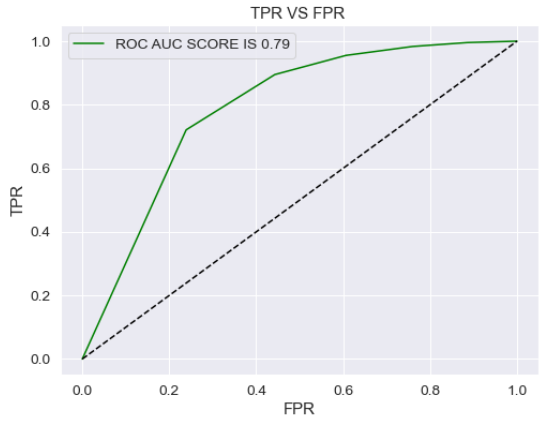
In [165]:
```
y_pred=rfc.predict_proba(x_test)
```

In [166]:
```
roc_auc=roc_auc_score(y_test,y_pred[:,1])
```

In [167]:
```
print(f"ROC AUC SCORE IS {round(roc_auc,2)}")
```

```
ROC AUC SCORE IS 0.79
```

In [168]:
```
plt.figure(figsize=(8,6))
fpr,tpr,thresholds=roc_curve(y_test,y_pred[:,1])
plt.plot(fpr,tpr,color='Green')
plt.plot([0,1],[0,1],color='Black',linestyle='--')
plt.ylabel("TPR")
plt.xlabel("FPR")
plt.title("TPR VS FPR ")
plt.legend(['ROC AUC SCORE IS {:.2f}'.format(roc_auc)])
plt.show()
```



5) ADA BOOST CLASSIFIER

MODEL 5) ADA BOOST REGRESSOR

In [169]:
```
from sklearn.ensemble import AdaBoostClassifier
```

In [170]:
```
adc=AdaBoostClassifier()
adc.fit(x_train,y_train)
```

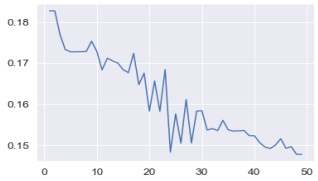
Out[170]:
```
AdaBoostClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [171]:
```
score(adc,x_train,y_train,x_test,y_test,train=True)
score(adc,x_train,y_train,x_test,y_test,train=False)
```

```
FOR MODEL AdaBoostClassifier()----------------------------------------------------


ACCURACY ON TRAINING DATASET IS 88.1


ACCURACY ON TESTING DATASET IS 88.01


              precision    recall  f1-score   support

           0       0.59      0.15      0.24      6587
           1       0.89      0.99      0.93     45812

    accuracy                           0.88     52399
   macro avg       0.74      0.57      0.59     52399
weighted avg       0.85      0.88      0.85     52399


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

In [172]:
```
adc_var=[]
for i in range(1,50):
    adc=AdaBoostClassifier(random_state=0,n_estimators=i)
    kfold=KFold(shuffle=True,random_state=0,n_splits=3)
    cross_score=cross_val_score(adc,x_train,y_train,cv=kfold,n_jobs=-1,scoring='f1_weighted')
    adc_var.append(1-np.mean(cross_score))
```

In [173]:
```
x_axis=np.arange(1,50)
plt.plot(x_axis,adc_var)
plt.show()
```



In [174]:
```
np.argmin(adc_var)
```

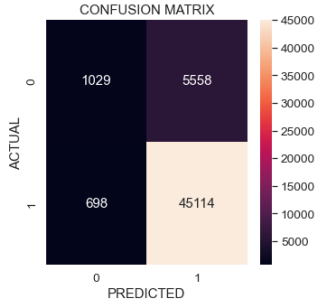
Out[174]:
```
48
```

In [175]:
```
adc=AdaBoostClassifier(n_estimators=47,random_state=0)
```

In [176]:
```
score(adc,x_train,y_train,x_test,y_test,train=True)
score(adc,x_train,y_train,x_test,y_test,train=False)
```

```
FOR MODEL AdaBoostClassifier(n_estimators=47, random_state=0)----------------------------------------------------


ACCURACY ON TRAINING DATASET IS 88.14


ACCURACY ON TESTING DATASET IS 88.06


              precision    recall  f1-score   support

           0       0.60      0.16      0.25      6587
           1       0.89      0.98      0.94     45812

    accuracy                           0.88     52399
   macro avg       0.74      0.57      0.59     52399
weighted avg       0.85      0.88      0.85     52399
```

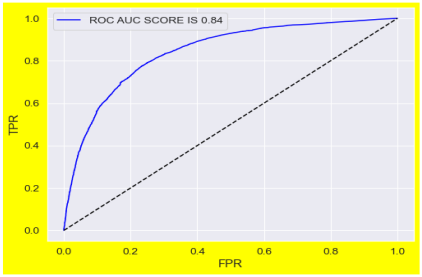
```
In [179]: plt.figure(figsize=(5,5))
          sns.heatmap(cm,annot=True,fmt='d')
          plt.ylabel("ACTUAL")
          plt.xlabel("PREDICTED")
          plt.title("CONFUSION MATRIX")
          plt.show()
```




```
In [180]: print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.60      0.16      0.25      6587
           1       0.89      0.98      0.94     45812

    accuracy                           0.88     52399
   macro avg       0.74      0.57      0.59     52399
weighted avg       0.85      0.88      0.85     52399
```

```
In [181]: from sklearn.metrics import roc_auc_score,roc_curve

In [182]: y_pred=adc.predict_proba(x_test)

In [183]: roc_auc=roc_auc_score(y_test,y_pred[:,1])

In [184]: print(f"ROC AUC SCORE IS {round(roc_auc,2)}")
          ROC AUC SCORE IS 0.84

In [185]: plt.figure(figsize=(8,6),facecolor='Yellow')
          fpr,tpr,thresholds=roc_curve(y_test,y_pred[:,1])
          plt.plot(fpr,tpr,color='Blue')
          plt.plot([0,1],[0,1],linestyle='--',color='Black')
          plt.xlabel("FPR")
          plt.ylabel("TPR")
          plt.legend(["ROC AUC SCORE IS {:.2f}".format(round(roc_auc,2))])
          plt.show()
```

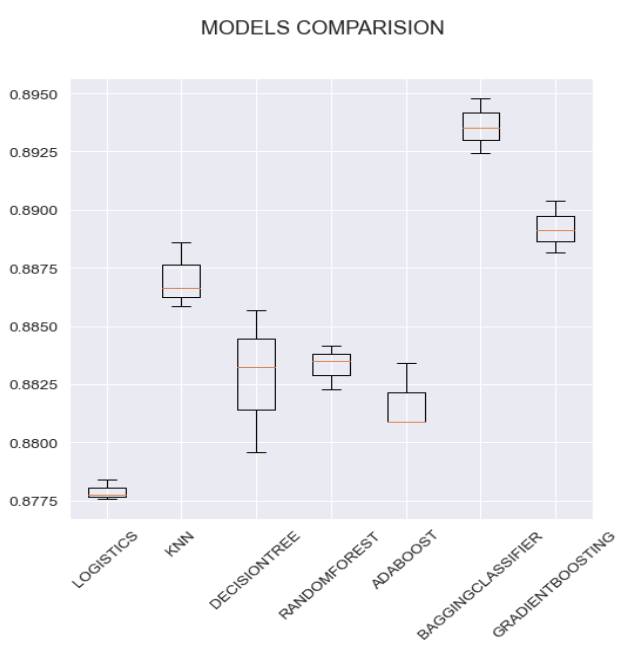


MODELS COMPARISON

Models are going to be compared on the cross-val score and variance present
A model with the highest cross-val scores and low bias and variance error will be used as the final model for this problem.

In [245]:
```
for name,model in models:
    kfold=KFold(shuffle=True,random_state=0,n_splits=3)
    cross_score=cross_val_score(model,x_train,y_train,cv=kfold)
    results.append(cross_score)
    names.append(name)
    print(f"{name}---{np.mean(cross_score)}---{(np.var(cross_score,ddof=1))}")
fig=plt.figure(figsize=(8,8))
fig.suptitle("MODELS COMPARISION")
ax=fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names,rotation=45)
plt.show()
```

```
LOGISTICS---0.8778961029046911---1.9049003771263458e-07
KNN---0.8870313116276702---1.967668604977262e-06
DECISIONTREE---0.8828263165260761---9.324663388453912e-06
RANDOMFOREST---0.8833034339733069---8.822760382779209e-07
ADABOOST---0.8817130424825375---2.1154228279830982e-06
BAGGINGCLASSIFIER---0.8935900861356032---1.403118780016082e-06
GRADIENTBOOSTING---0.8892133287530058---1.2533006114136266e-06
```



CONCLUSIONS!! BaggingClassifier is the best algorith for this model because of low bias error and variance error,The test is done with base estimator as decision tree with number of estimators as 35 and random state as 0.

Conclusion:

It can be seen that

- ✓ BaggingClassifier has the highest cross-val score and low variance.

- ✓ Having a high f-1score gives it an edge over the other models in the comparison.
- ✓ Having a high roc_auc_score with the above-added conclusions is also a significant reason for selecting this model.