FLIP ROBO

MICRO CREDIT LOAN CASE STUDY

Submitted by:

Smriti Mathur

Acknowledgement

I would want to express my gratitude to Flip Robo Technologies for providing me with this fantastic chance to work on such an important project. This project taught me a lot about data wrangling and how to use machine learning modules in actual applications.

I would like to use this opportunity to offer my heartfelt appreciation to all of the Flip Robo Technologies employees for their kind assistance, useful information (supplied in the form of a dataset), and helpful advice, which aided me in finishing this assignment at various stages.

I would want to use this occasion to convey my heartfelt thanks and great admiration for my mentor, Ms. Khushboo, for her outstanding supervision, monitoring, and consistent encouragement during this project. Her blessings, assistance, and wisdom from time to time will take me a long way on the life adventure I am about to embark on.

Finally, I want to express my gratitude to the Almighty, my parents, brother, sister, and friends for their unwavering support, without which this task would not have been possible.

References:

1. https://www.researchgate.net/
2. https://en.wikipedia.org/
3. https://corporatefinanceinstitute.com/
4. https://link.springer.com/
5. https://eajournals.org/

INTRODUCTION

- Business Problem Framing

Microfinance Institutions (MFIs) are businesses that provide financial services to low-income people. When addressing unbanked poor families living in rural places with few sources of income, MFS becomes quite effective. Group Loans, Agricultural Loans, Individual Business Loans, and other Microfinance Services (MFS) are some of the Microfinance Services (MFS) provided by MFI.

Many microfinance institutions (MFI), experts, and donors promote the use of mobile financial services (MFS), which they believe are more convenient, efficient, and cost-effective than the traditional high-touch strategy used to deliver microfinance services for a long time. Despite the fact that the MFI industry focuses primarily on low-income households and is particularly valuable in these areas, MFS implementation has been uneven, with both substantial successes and failures.

Microfinance is now widely recognised as a strategy for poverty reduction, with $70 billion in outstanding loans and a global client base of 200 million people.

We are now working with a client in the telecom industry. They are a provider of fixed wireless telecommunications networks. They've released a number of products and built their business and organisation around the budget operator model, which entails providing better products at lower prices to all value-conscious clients via a disruptive innovation strategy that focuses on the subscriber.

They recognise the value of communication and how it influences a person's life, thus they focus on giving low-income families and impoverished consumers with services and products that can assist them in their time of need.

They've teamed up with a microfinance institution to offer micro-credit on mobile balances that must be paid back in five days. If the Consumer deviates from the course of repaying the loaned amount within the time period of 5 days, he is considered a defaulter. The payback amount for a loan of 5 (in Indonesian Rupiah) should be 6 (in Indonesian Rupiah), whereas the payback amount for a loan of 10 (in Indonesian Rupiah) should be 12. (in Indonesian Rupiah).

Here, we need to create a model that can be used to predict if a client would pay back the lent amount within 5 days of loan insurance in terms of probability for each loan transaction. Label '1' shows that the loan has been paid, indicating that it is a non-defaulter, whereas Label '0' indicates that the loan has not been paid, indicating that it is a defaulter.

• Conceptual Background of the Domain Problem

The global expansion of mobile phones, along with financial services industry deregulation, has created new potential for trusted brands.

Organizations with millions of clients and vast distribution channels, such as mobile operators, merchants, and on-line brands, now have the chance to participate in the high margins of financial services that were previously only available to banks and related financial services companies. However, this comes with a high learning curve, challenges with change management, and significant penalties from financial services regulators for negligence.

"How banks can translate the potential of mobile phones into better financial access for disadvantaged people," according to a recent report from the Consultative Group to Assist the Poor (CGAP) (CGAP, 2008b:1). The research draws on a series of papers from donors and industry sources that highlight the potential of mobile phone applications to meet the financial requirements of persons in developing nations who are now unbanked or excluded from conventional financial services (CGAP, 2008a; 2008b; UNCTAD, 2007; World Bank, 2006). Rapid expansion of networks into previously un-served regions and communities in developing countries has fueled belief in the potential of mobile phones to assist address the financial service demands of the poor over the last decade. The impact was most obvious in Sub-Saharan Africa and South Asia's least developed nations (LDCs), where existing fixed-line infrastructure was particularly weak and underdeveloped. Mobile communications are presently the fastest-growing technology in developing countries, and research has already shown that their adoption has had a considerable socioeconomic impact in underprivileged communities (Abraham, 2006; Jensen, 2007; Overa, 2006).

Because mobile phones are increasingly becoming a part of the poor's daily lives, it is argued that they have the potential to become a low-cost, easily accessible 'account' or delivery channel for financial information, services, and transactions (Porteous, 2006), allowing for innovations such as micropayments (m-payments), electronic money (e-money), and a banking channel (m-banking). Existing projects (such as Globe Telecom's GCash in the Philippines, WIZZIT in South Africa, Safaricom's M-PESA in Kenya, and the Grameen Village Phone Program in Bangladesh) have already demonstrated the viability of such services in developing countries. According to research, the impoverished majority needs and is progressively demanding a larger range of micro-financial services, which might be offered via mobile phones or mobile phone carriers.

These are low-cost solutions that can support regular saves, streamline payments, permit monetary transfers (particularly in tiny amounts), and provide micro-credit (Claessens, et.al 2006). However, studies warn that the poor's financial requirements are fungible, implying a complicated set of interactions across a wide range of primarily informal financial service contexts (Ghate, 1992; Matin, Hulme & Rutherford, 2002). As a result, a greater understanding of the interplay between the indicated potential for mobile phone applications and the reality of the impoverished majority's financial service preferences and behaviours is essential.

"Money makes money" . It's frequently simple to get more once you've got a little. Getting that little is the most challenging part." Years ago, when there was a far more unequal distribution of wealth, the problem of loan accessibility was identified. Hundreds of people made bequests to cover revolving loans for ambitious young men, indicating that this problem was well-known. According to Jordan, who studied philanthropy in England during this time period, around 3% of all money accessible to charities between 1480 and 1660 was earmarked for this reason. Micro-credit schemes' poverty-reduction potential is frequently viewed as a promotional process by which poor households 'graduate' out of poverty. This graduation can be simplified as breaking a vicious circle of ‘low investment - low income - and low investment’ by injecting capital in the form of credit to generate productive employment, higher incomes and more investment. However, this model of poverty and the focus on credit as the solution is simplistic, because a range of factors other than investment reproduce poverty and define its qualitative dimensions(Montgomery, 1996).

- Review of Literature

Research on the diffusion of innovations was initiated by independent thinkers from many fields in the early decades, and it was approached in a variety of ways. An innovation, according to Rogers & Shoemaker (1971), is a concept that is seen as novel by a person or a system.

Rogers and Shoemaker (1971) documented a new pattern that began in the mid-60s. The cross-disciplinary indices demonstrated that this new research trend tended toward a more unified viewpoint when they computed an index for each available diffusion publication. Many authors have looked at theories on how people adapt to new technology advancements. The diffusion of innovation idea, proposed by Everett Rogers, is one of the most prominent theories (1983).

In his book Diffusion of Innovation, Rogers (E. Rogers, 1983) explains that diffusion is “the process by which an innovation is communicated through certain channels over time among the members of a social system”.

Because diffusion is a form of communication, it aims to explain how, why, and at what rate a new technology might spread across cultures, as well as why there are varied rates of technological adoption. According to Rogers (1983), the rate at which a new technology may be adopted by a society is determined by various characteristics of innovation:

1. The relative advantage perceived by individuals: the greater is the advantage perceived, faster will be the adoption (Casmar, 2001; Finley, 2003; McKenzie, 2001).
2. The compatibility with the norms and values of a society: the more technological innovation complies with the values of a society, the sooner it will be adapted.

Communication channels, time, and the social structure are also factors in technology diffusion (E. Rogers, 1983). This diffusion strategy, created by Rogers (E. Rogers, 1983), was developed as a common subject in technical research by other writers (Chigona & Licker, 2008; Orlikowski & Iacono, 2001).

According to Lundvall (2009), innovation occurs when technology potential and user requirements collide, and there has been a shift in study focus as Internet usage and diffusion have expanded. Researchers in the field of information systems have been concentrating on identifying elements that may make the integration of technology into business easier (Legris et al., 2003). One of the most powerful forces driving the global economy is digital technologies (Brynjolfsson & McAfee, 2012).

In their analysis of the Saudi banking sector, Al-Jabri & Sohail (2012) employed the diffusion of innovation theory (DIT) as a baseline. They wanted to see if this idea could explain how mobile uptake occurs. The DIT was utilised as a framework in another study to see if the five qualities of innovations have an impact on adoption among the urban poor.

Another model created by Fishbein and Ajzen is the Technology Acceptance Model (TAM). McKechnie, Winklhofer, & Ennew, 2006; Shin & Kim, 2008; Teo, Lee, & Chai, 2007; van Biljon & Kotzé, 2007) used the Technology Acceptance Model in a variety of scenarios and by a number of studies. In terms of usage intentions and perceived usefulness, this paradigm is more focused on the individual's acceptance and adoption of information systems.

Many studies have also been conducted to examine the situation from an economic standpoint. For example, in his study, Baliamoune-Lutz (2003) examined the relationship between ICT diffusion and per capita income using data from developing nations. When examining the importance of IC technology in the workplace, there were significant variances. Because of this, developed countries have an advantage over developing countries (Roztocki & Weistroffer, 2011). Because these disparities are frequently caused by environmental conditions, they can be linked to them.

The problems and factors that impact ICT dissemination differ from country to country. Adoption in developing countries cannot be compared to adoption in developed ones.

Emerging countries is a concept coined by Roztocki and Weistroffer (2011) to denote a region experiencing rapid economic expansion. According to the Roztocki & Weistroffer study, the following are some of the unique environmental factors found in emerging or transitioning economies: 1) laws and regulations, 2) employee characteristics, 3) government control, 4) management style, 5) customer characteristics, and 6) economic condition.

The diffusion of innovation theory includes a wide range of topics, especially when scholars explore innovation dissemination in underdeveloped nations, where there are significant disparities from affluent ones.

- Motivation for the Problem Undertaken

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

2. **Motivation behind this Project:** Mobile financial services has built a set of technology and service capabilities to address these challenges and provide end to end solutions to add value in our business to business delivery model.

Since the early 2000s, the phrase "financial inclusion" has gained traction as research suggests that financial exclusion increases the probability of poverty (Shiimi, 2010). Financial inclusion is defined in this paper as the provision of meaningful and cheap financial services to disadvantaged and low-income persons on a long-term basis (Muzigiti and Schmidt, 2013; World Bank, 2017).

Financial inclusion may have a favourable impact on economic development at the household, firm, and national levels. Increased financial access through efficient financial inclusion programmes can enable households to invest more in their assets, which is linked to productivity and can lead to future increases in household income (DFID, 2004).

Nevertheless, such benefits of financial access are only limited to the developed world, as most developing countries experience a deficiency of access to financial services. According to the World Bank (2012), the provision of financial accounts differs enormously between high-income and developing economies. The poor who live in the rural areas of developing countries tend to have lesser access to financial services due to the lack of infrastructure and poor economic conditions (SantaMaria, 2016).

This will enable us to assess the existing state of our knowledge, research overlaps, and gaps, as well as the emerging research agenda centred around the three keywords.

While Donner and Tellez (2008) and Duncombe and Boateng (2009) provide a comprehensive overview of all three aspects, their findings are outdated and only cover the early stages of MFS development. Shaikh and Karjaluoto (2015), on the other hand, concentrate on mobile banking and provide a limited scope for our goal of examining a variety of MFS, such as mobile payment. We evaluate existing research at the nexus of MFS, financial inclusion, and development to address this vacuum, find missing analytical bridges connecting the three concepts, and draw recommendations for future study.

The expansion of information technology, growing usage of personal computers, improved internet access, and widespread use of mobile phones have all spurred mobile enterprises to offer financial services in the digitalized world. The usage of MPesa in poor countries is a success story. According to Zandi and Singh (2010), electronic payment instruments and economic growth have a strong relationship. Hasan et al. (2012) found that transitioning from cash to electronic payments influenced the whole economy of 27 European countries between 1995 and 2009. According to a Canadian study (Arango & Taylor 2008), a business should consider the costs and benefits before accepting electronic or cash payments.

The advancement of mobile technology has contributed to a significant rise in mobile devices‟ popularity, such as significantly increased data transmission rates and multimedia services and applications (Gruber & Koutroumpis, 2010). According to Yan, G, Paradi, J.C (1998), most financial institutions have limitations on the transactions they perform on customer accounts through the bank, as well as a limit on the amount that can be transferred. M-PESA offers access to financial services and is one of the most famous mobile money platforms (Mbogo, 2010). Customers can use their mobile device to transfer money by using the mobile phone short message services (SMS) and most importantly without the need to have a bank account (Karugu & Mwendwa, 2007).

Analytical Problem Framing

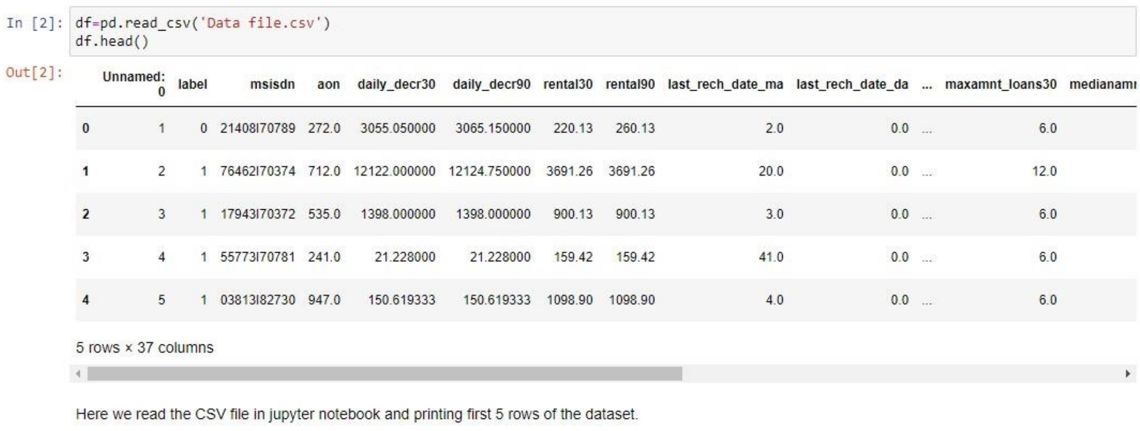
- Data Sources and their formats

The dataset has been given by Flip Robo Technologies in excel format. The dataset has 209593 rows. There are 36 features and 1 target in the dataset. These are as follows:

1. 'Unnamed: 0' : (Integer) It contains the serial number, hence we can drop this column.
2. 'label' : (Integer) Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan{1:success, 0:failure}
3. 'msisdn' : (Object) mobile number of user
4. 'aon' : (Float) age on cellular network in days
5. 'daily_decr30' : (Float) Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah)
6. 'daily_decr90' : (Float) Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah)
7. 'rental30' : (Float) Average main account balance over last 30 days
8. 'rental90' : (Float) Average main account balance over last 90 days
9. 'last_rech_date_ma' : (Float) Number of days till last recharge of main account
10. 'last_rech_date_da' : (Float) Number of days till last recharge of data account
11. 'last_rech_amt_ma' : (Integer) Amount of last recharge of main account (in Indonesian Rupiah)
12. 'cnt_ma_rech30' : (Integer) Number of times main account got recharged in last 30 days
13. 'fr_ma_rech30' : (Float) Frequency of main account recharged in last 30 days
14. 'sumamnt_ma_rech30' : (Float) Total amount of recharge in main account over last 30 days (in Indonesian Rupiah)
15. 'medianamnt_ma_rech30' : (Float) Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah)
16. 'medianmarechprebal30' : (Float) Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah)
17. 'cnt_ma_rech90' : (Integer) Number of times main account got recharged in last 90 days
18. 'fr_ma_rech90' : (Integer) Frequency of main account recharged in last 90 days
19. 'sumamnt_ma_rech90' : (Integer) Total amount of recharge in main account over last 90 days (in Indonesian Rupiah)
20. 'medianamnt_ma_rech90' : (Float) Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah)

21. 'medianmarechprebal90' : (Float) Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupiah)
22. 'cnt_da_rech30' : (Float) Number of times data account got recharged in last 30 days
23. 'fr_da_rech30' : (Float) Frequency of data account recharged in last 30 days
24. 'cnt_da_rech90' : (Integer) Number of times data account got recharged in last 90 days
25. 'fr_da_rech90' : (Integer) Frequency of data account recharged in last 90 days
26. 'cnt_loans30' : (Integer) Number of loans taken by user in last 30 days
27. 'amnt_loans30' : (Integer) Total amount of loans taken by user in last 30 days
28. 'maxamnt_loans30' : (Float) maximum amount of loan taken by the user in last 30 days
29. 'medianamnt_loans30' : (Float) Median of amounts of loan taken by the user in last 30 days
30. 'cnt_loans90' : (Float) Number of loans taken by user in last 90 days
31. 'amnt_loans90' : (Integer) Total amount of loans taken by user in last 90 days
32. 'maxamnt_loans90' : (Integer) maximum amount of loan taken by the user in last 90 days
33. 'medianamnt_loans90' : (Float) Median of amounts of loan taken by the user in last 90 days
34. 'payback30' : (Float) Average payback time in days over last 30 days
35. 'payback90' : (Float) Average payback time in days over last 90 days
36. 'pcircle' : (Object) Telecom circle
37. 'pdate' : (Object) Date

Here, ‘label’ is out target variable. 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. Since the target has been categorized into defaulter and non-defaulter, this becomes a project of Classification Problem. Hence we must use classification algorithms to build machine learning models.



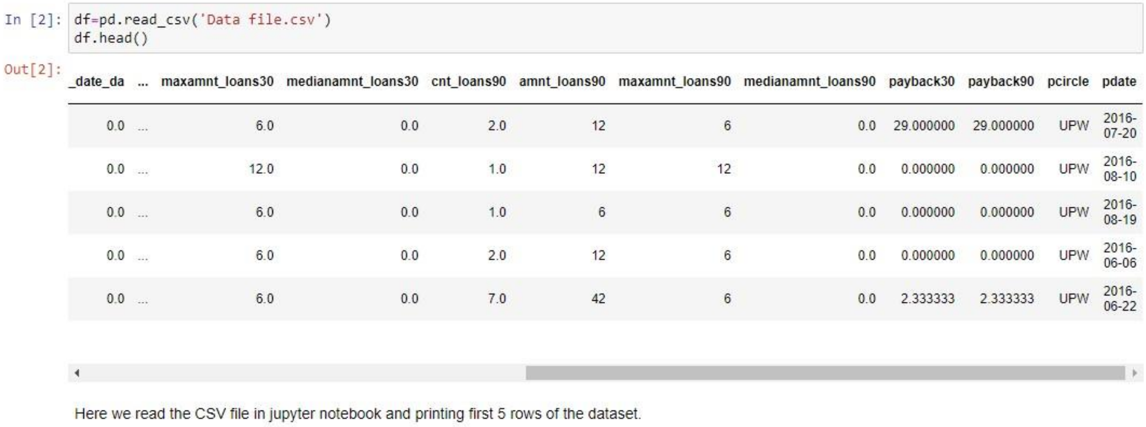
```
In [2]: df=pd.read_csv('Data file.csv')
        df.head()
```

Out[2]:

	Unnamed: 0	label	msisdn	aon	daily_decr30	daily_decr90	rental30	rental90	last_rech_date_ma	last_rech_date_da	...	maxamnt_loans30	medianamı
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	

5 rows × 37 columns

Here we read the CSV file in jupyter notebook and printing first 5 rows of the dataset.



```
In [2]: df=pd.read_csv('Data file.csv')
        df.head()
```

Out[2]:

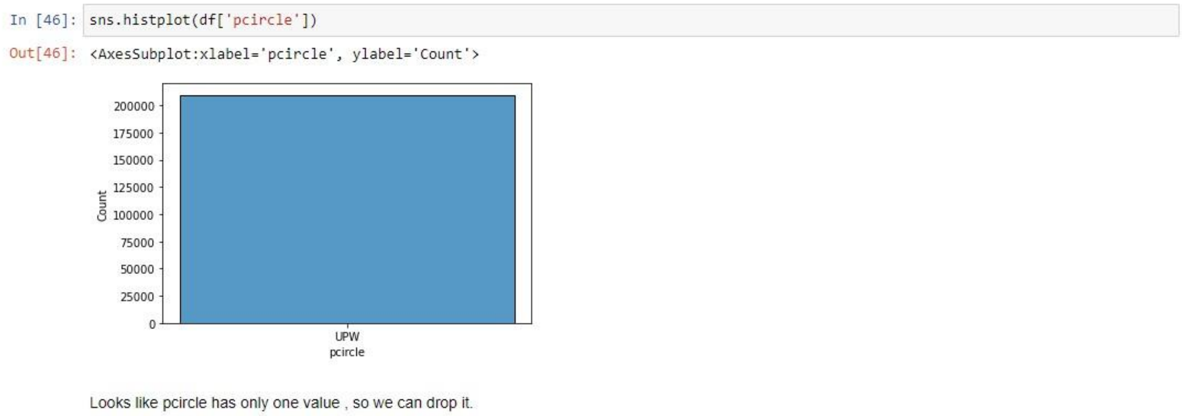
_date_da	...	maxamnt_loans30	medianamnt_loans30	cnt_loans90	amnt_loans90	maxamnt_loans90	medianamnt_loans90	payback30	payback90	pcircle	pdate
0.0	...	6.0	0.0	2.0	12	6	0.0	29.000000	29.000000	UPW	2016-07-20
0.0	...	12.0	0.0	1.0	12	12	0.0	0.000000	0.000000	UPW	2016-08-10
0.0	...	6.0	0.0	1.0	6	6	0.0	0.000000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

Here we read the CSV file in jupyter notebook and printing first 5 rows of the dataset.

Using the describe(), we get the statistical summary of the dataset and through which we interpret that few features are defined over 30 days' period as well as 90 days' period and such features contains almost similar values, which leads to creating multi-collinearity of the features. Hence in order to treat this multi-collinearity, we might have to drop few features defined over different time periods, having the least correlation with the target.

- ## Mathematical/ Analytical Modeling of the Problem

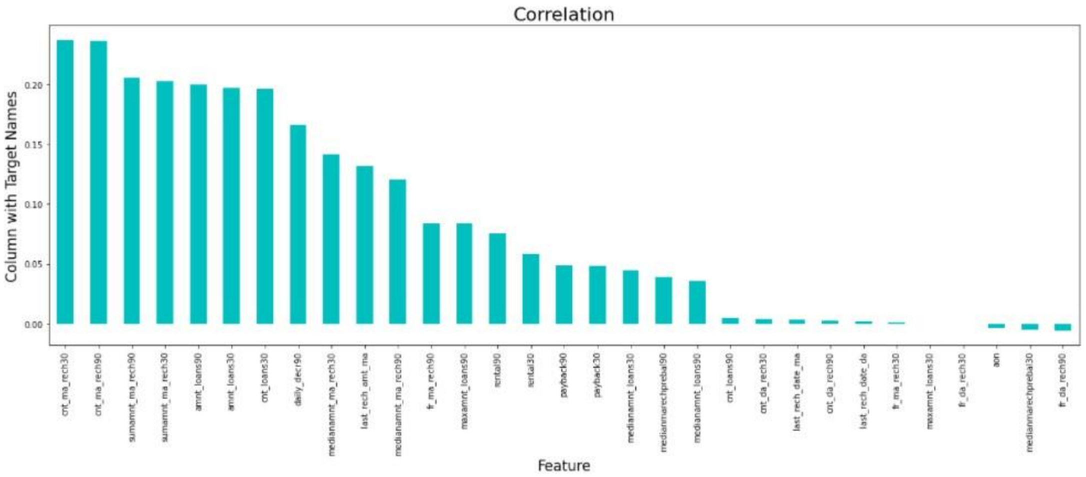
Using dataset.info(), we get all the information about the data like the name of features along with their data types, their non-null value count and total memory in usage. The total memory in use is 59.2 MB. We got to know that there are 21 columns with float data type, 13 columns with integer data typ e and 3 columns with object data type which are 'msisdn', 'pcircle' and 'pdat e'. We have observed that mobile number and date have very wide range of data and which does not add much in prediction of 'Label' , also the column ‘pcircle’ has only one value which does not contribute in ‘Label’ prediction as well.



```
In [46]: sns.histplot(df['pcircle'])

Out[46]: <AxesSubplot:xlabel='pcircle', ylabel='Count'>
```

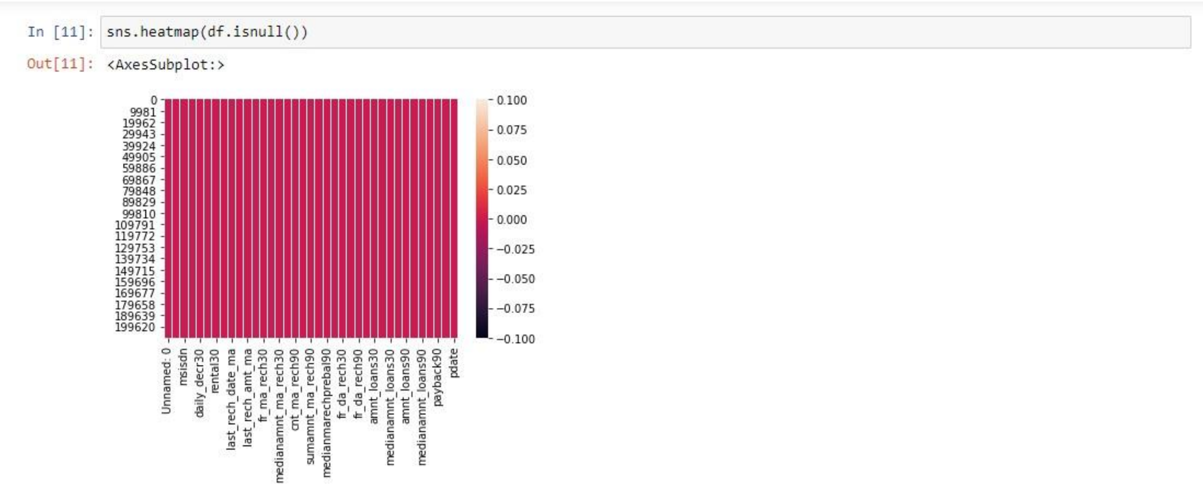
Looks like pcircle has only one value , so we can drop it.

While checking the correlation using Corr(), we found out that, two features have negligible correlation with the target, which we can happily drop to make our model more effective. These features are : ‘maxamnt_loans30’ (maximum amount of loan taken by the user in last 30 days) and ‘fr_da_rech30’ (Frequency of data account recharged in last 30 days).



- ## Data Preprocessing Done

There are missing values in the given dataset which we check by using isnull().sum() then we also plot heatmap to visualize the null values in the dataset.



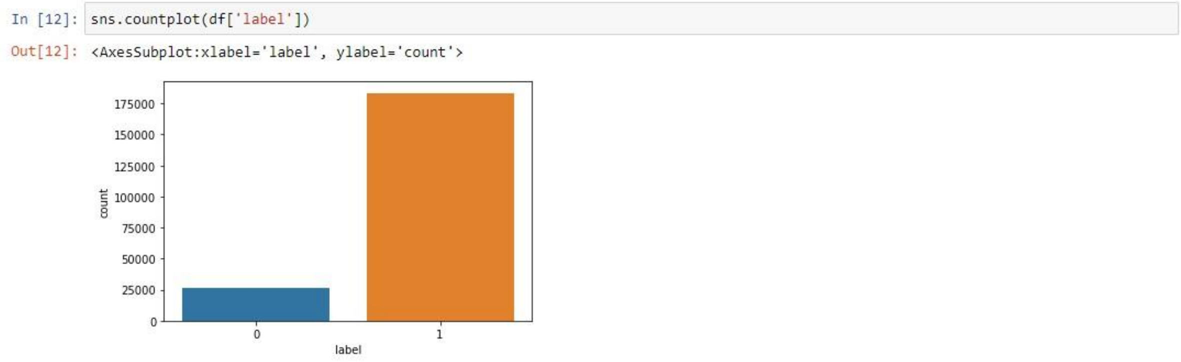
```
In [11]: sns.heatmap(df.isnull())
Out[11]: <AxesSubplot:>
```

Visual representation of having absolutely no null values in the numerical columns .

There are so many outliers in the dataset and if we remove all the outliers, we will lose almost 18% of the data and we can't afford to lose that much of data, hence we will remove the outliers with zscore value more than 5 and in this way, we will lose only 7.5% of the data which is in acceptable range.

We now will check for skewness using skew(). There is skewness in three columns which are 'cnt_da_rech30', 'fr_da_rech90' and 'cnt_loans90'. To treat the skewness we will apply power transformation technique. We then scale the features using StandardScalar().

There is a lot of class imbalance in the target, 'Label' and we will treat this imbalance using Over_Sampling technique.

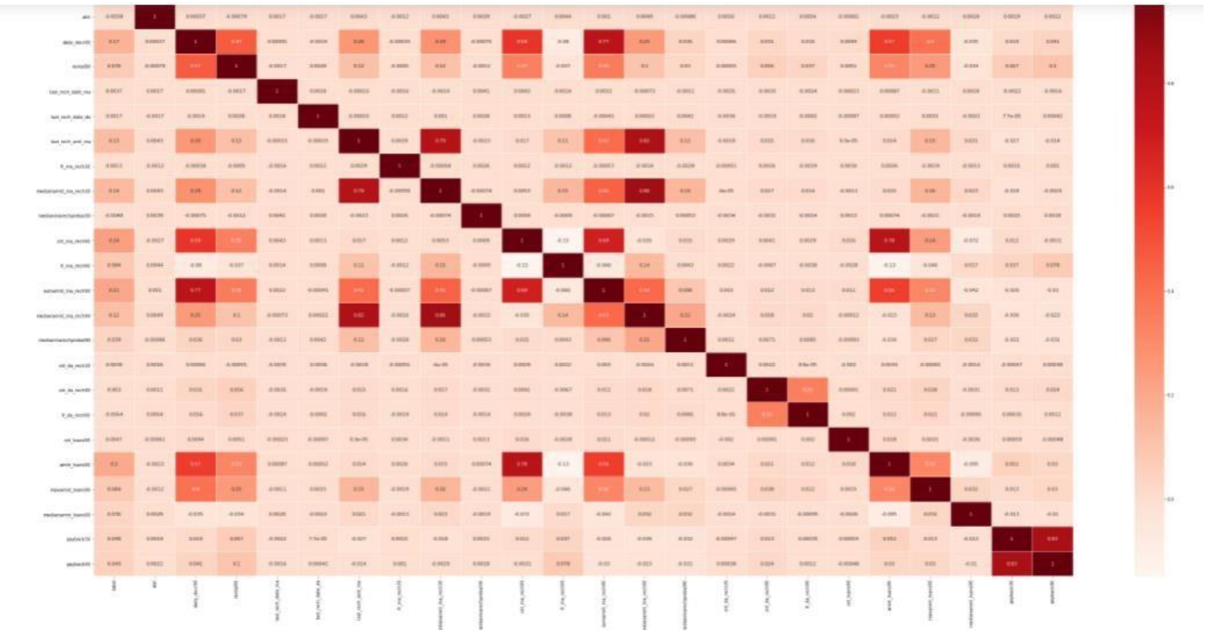


```
In [12]: sns.countplot(df['label'])
Out[12]: <AxesSubplot:xlabel='label', ylabel='count'>
```

There is a lot of class imbalance which we will have to treat to train the model with more accuracy.

- Data Inputs- Logic- Output Relationships

Since there are 36 features in the dataset and by various statistical and graphical analysis we found out that only 24 features are actually adding much of value in predicting the ‘Label’. Hence we first drop the unwanted 12 columns. The correlation of value-adding 24 features is as follows:



From the above heatmap we conclude that ‘cnt_ma_rech90’ has the maximum positive correlation with the target. As the value of ‘cnt_ma_rech90’ increases, there are more chances of a customer being a non-defaulter. ‘sumamnt_ma_rech90’, ‘amnt_loans90’ and ‘cnt_loans30’ have same positive correlation with the target and it’s second highest. Rest all the features have comparatively less correlation with the target. Moreover, ‘aon’ , ‘fr_da_rech90’ and ‘medianmarechprebal30’ are negatively correlated to the data at a very minimum number.

• Hardware and Software Requirements and Tools Used

Hardware Requirements: 64-bit operating system, x64-based processor; minimum of 8.00 GB installed RAM; at least a Core i3 processor, 128 GB SSD hard disk and Windows 10 edition at least.

Software Requirements: This project has been coded in Python language using Anaconda 3 Jupyter Notebook package. The libraries that we have installed and used are as follows:

1. Pandas: Pandas is a free data manipulation and analysis library created for the Python programming language. Pandas are a type of data representation that is incredibly simplified. This makes it easier to examine and comprehend data. For data science projects, simplified data representation facilitates better findings. Pandas aids in the speeding up of data processing. We can devote more attention to data analysis algorithms now that we have saved time. Pandas can save a lot of time by quickly importing massive amounts of data. Here we have loaded the huge volume of dataset in no time using pandas library function. Pandas has a large feature set that you may use on our data to adapt, change, and pivot it according to our own preferences. This makes it easier to get the most out of our data.

2. Numpy: Numerical Python is what it stands for. NumPy aids in the creation of arrays. It is the most important Python package for scientific computing. A multidimensional array and matrix data structures are also included in the NumPy library. We have lists in Python that act as arrays, however they are slow to process. NumPy intends to deliver a 50-fold quicker array object than ordinary Python lists. The array object in NumPy is named ndarray, and it comes with so many support functions to make working with it a breeze. Here we have used numpy to calculate absolute value for detecting outliers using zscore and for converting the list elements into array object.

3. Matplotlib: Matplotlib is a Python data visualisation package. It primarily assists us in the plotting of two-dimensional graphs. It is based on the Numpy and Scipy Python frameworks. It includes such tools for creating publication-standard plots and figures in a range of export formats and contexts. It is a great approach to create high-quality static graphics for publications and professional presentations. It also has interoperability with a variety of additional third-party libraries and packages, such as seaborn, that extend its capability.

4. Seaborn: Seaborn is a Python package that helps to visualise data and make it more understandable to the user. We may plot our data and create a graphical representation of it with the help of the library. Internally, this library makes use of matplotlib; in other words, it is

entirely built on matplotlib. In this project we have used seaborn to plot countplot, histogram, distribution plot, kde plot, strip plot, pie chart, bargraph, pairplot and heatmap.

5. Sklearn: Scikit-learn is an open-source machine learning library that was first released in 2007. Python is the framework's scripting language, and it incorporates numerous machine learning models such as classification, regression, clustering, and dimensionality reduction. Scikit-learn is based on Matplotlib, NumPy, and SciPy, which are all open-source projects. Scikit-learn offers a large number of machine learning methods to users. The framework library focuses on data modelling rather than data loading, summarization, or manipulation.

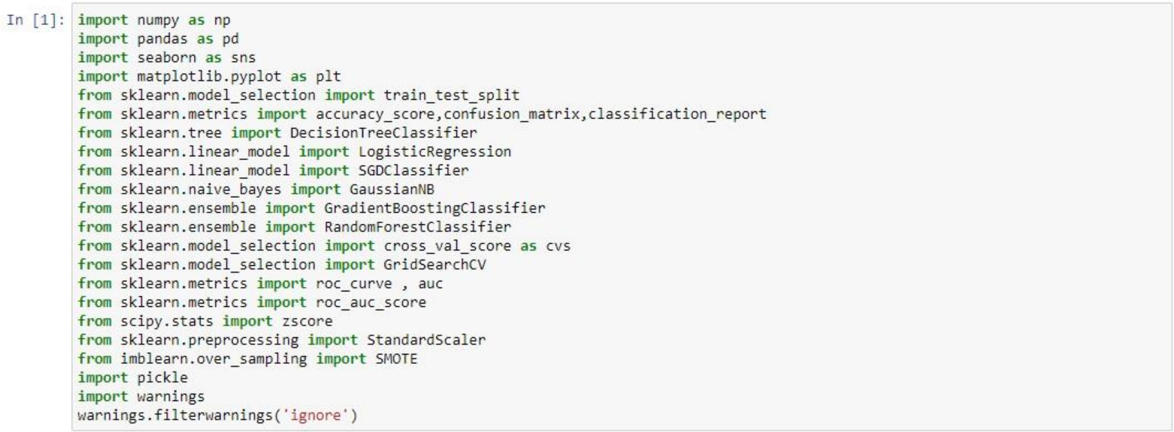
   In Python, Scikit-learn (Sklearn) is the most usable and robust machine learning library. It uses a Python consistency interface to give a set of efficient tools for machine learning and statistical modelling, such as classification, regression, clustering, and dimensionality reduction. NumPy, SciPy, and Matplotlib are the foundations of this library, which is mostly written in Python. It's flexible and works well with other Python libraries like graphing with matplotlib, array vectorization with numpy, and dataframes with pandas. Using sklearn in this project, we've imported toolkit like roc_curve, auc_roc_score, power_transform, standard_scalar, train_test_split (for splitting the dataset into training model and testing model), accuracy_score, confusion_matrix, classification_report, cross_val_score, GridSearchCV (for hyper parameter tuning of the model) and scaled the features using StandardScalar.

6. Scipy: SciPy is a Python-based open-source library for mathematics, scientific computing, engineering, and technical computing. SciPy has a number of sub-packages that aid with the most prevalent problems in Scientific Computation. NumPy is the foundation for SciPy. It gives users the ability to alter and view data using a variety of high-level Python commands. In this project we have used SciPy to measure zscore and eventually remove outliers.

7. Imblearn: The Imblearn library was created to deal with datasets that are unbalanced. To handle and remove the imbalance from the dataset, it offers several approaches such as undersampling and oversampling.

8. Pickle: For serialising and de-serializing a Python object structure, the Python pickle package is used. Pickling an object in Python allows it to be saved on disc. Pickling is the process of converting a Python object hierarchy into a byte stream.

9. Warnings: A warning is usually issued when particular programming pieces, such as a keyword, function, or class, have become obsolete. A warning is not the same as an error in a programme. If an error occurs, the Python application will end instantly. A warning, on the other hand, is not critical. It displays a message, but the software continues to operate. Or in other words we can say that, “It suggests some message, but the application runs”.



```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.linear_model import LogisticRegression
        from sklearn.linear_model import SGDClassifier
        from sklearn.naive_bayes import GaussianNB
        from sklearn.ensemble import GradientBoostingClassifier
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.model_selection import cross_val_score as cvs
        from sklearn.model_selection import GridSearchCV
        from sklearn.metrics import roc_curve , auc
        from sklearn.metrics import roc_auc_score
        from scipy.stats import zscore
        from sklearn.preprocessing import StandardScaler
        from imblearn.over_sampling import SMOTE
        import pickle
        import warnings
        warnings.filterwarnings('ignore')
```

Model/s Development and Evaluation

• Identification of possible problem-solving approaches (methods)

Machine learning is a subset of Artificial Intelligence in which machines learn from their operations and refine themselves to produce better results. The machines improve the computer programmes aligning with the needed output based on the data collected. Traditional programming is not replaced by machine learning. Regular programming techniques are supplemented by machine learning and artificial intelligence (AI).

To begin with we will read and understand the dataset. While reading the dataset we see that the target is already available in the dataset, which makes it a problem of Supervised Machine Learning. Based on the type of target, we will decide what kind of algorithms to be used. Here the target is classified hence we will use Classification algorithm on the dataset. Before applying algorithms we will have to clean and preprocess the data. Then we will have to split the dataset into training model and testing model. Only then we can apply algorithms on it.

Machine Learning Tools are consists of:

1. Preparation and data collection
2. Building models
3. Application deployment and Training

• Testing of Identified Approaches (Algorithms)

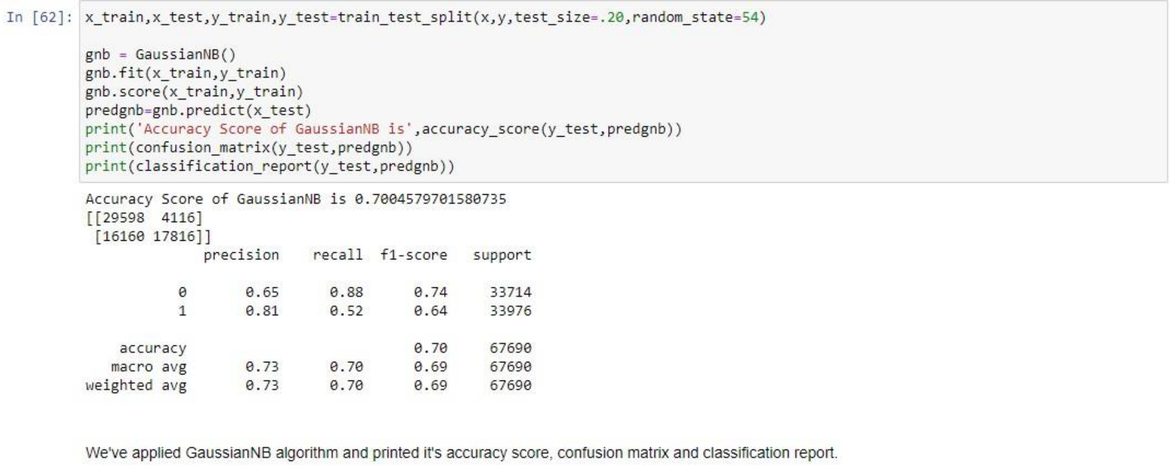
Since the dataset is quite large in size and it’s a classification problem, we will choose following algorithms to apply on our model:

1. Gaussian Naïve Bayes
2. Logistic Regression
3. Stochastic Gradient Descent Classifier
4. Decision Tree Classifier
5. Gradient Boosting Classifier

6. Random Forest Classifier

- Run and Evaluate selected models

1. **GaussianNB** : Gaussian Naive Bayes is a Naive Bayes variation that is based on the Gaussian normal distribution. The Bayes theorem provides the basis for a collection of supervised machine learning classification algorithms known as Naive Bayes. It's a simple categorization technique with a lot of power. Because we'll need to determine the mean and standard deviation for the training data, the Gaussian or normal distribution is the easiest to implement. The dataset was attempted to be modelled as a blend of multiple Gaussian Distributions. This is the model's central concept.

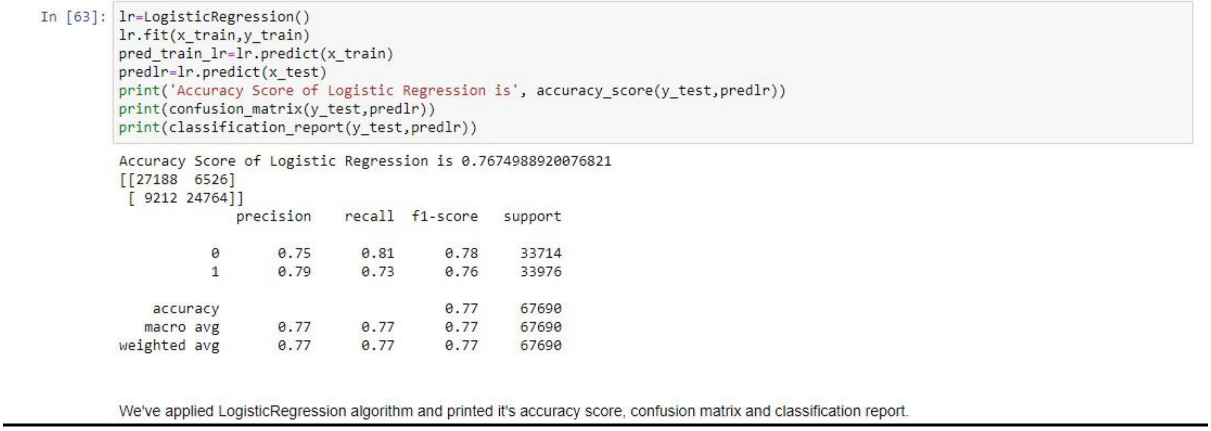


```
In [62]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.20,random_state=54)

         gnb = GaussianNB()
         gnb.fit(x_train,y_train)
         gnb.score(x_train,y_train)
         predgnb=gnb.predict(x_test)
         print('Accuracy Score of GaussianNB is',accuracy_score(y_test,predgnb))
         print(confusion_matrix(y_test,predgnb))
         print(classification_report(y_test,predgnb))
```

```
Accuracy Score of GaussianNB is 0.7004579701580735
[[29598  4116]
 [16160 17816]]
              precision    recall  f1-score   support

           0       0.65      0.88      0.74     33714
           1       0.81      0.52      0.64     33976

    accuracy                           0.70     67690
   macro avg       0.73      0.70      0.69     67690
weighted avg       0.73      0.70      0.69     67690
```

We've applied GaussianNB algorithm and printed it's accuracy score, confusion matrix and classification report.

2. **Logistic Regression**: One of the most useful machine learning techniques in the world of statistics is logistic regression. It can be used to solve problems with binary and multi-class categorization. Because it can generate probabilities and classify new data using both continuous and discrete datasets, logistic regression is a key machine learning approach.



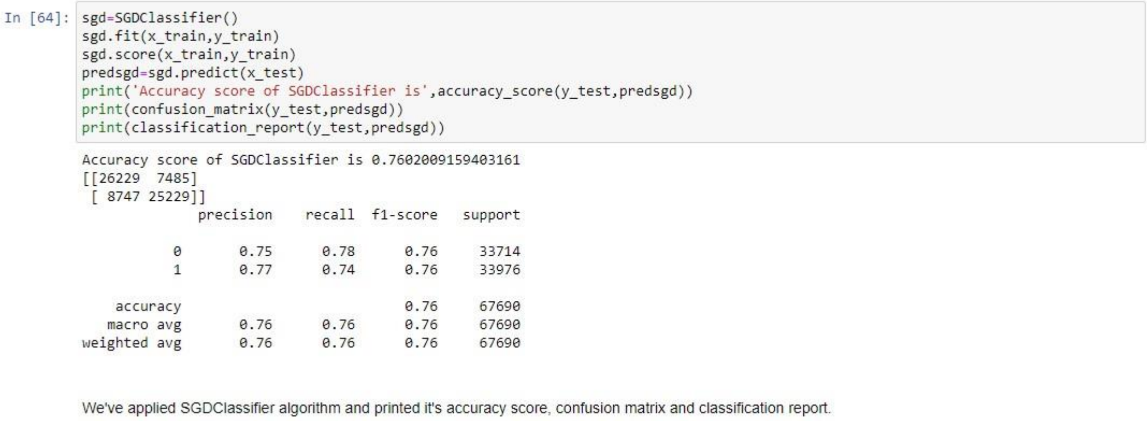
```
In [63]: lr=LogisticRegression()
         lr.fit(x_train,y_train)
         pred_train_lr=lr.predict(x_train)
         predlr=lr.predict(x_test)
         print('Accuracy Score of Logistic Regression is', accuracy_score(y_test,predlr))
         print(confusion_matrix(y_test,predlr))
         print(classification_report(y_test,predlr))
```

```
Accuracy Score of Logistic Regression is 0.7674988920076821
[[27188  6526]
 [ 9212 24764]]
              precision    recall  f1-score   support

           0       0.75      0.81      0.78     33714
           1       0.79      0.73      0.76     33976

    accuracy                           0.77     67690
   macro avg       0.77      0.77      0.77     67690
weighted avg       0.77      0.77      0.77     67690
```

We've applied LogisticRegression algorithm and printed it's accuracy score, confusion matrix and classification report.

3. **Stochastic Gradient Descent Classifier**: Gradient descent is one of the most widely used optimization methods, and it is by far the most frequent method for optimising neural networks. The gradient is computed using stochastic gradient descent (SGD) with a single sample. SGD permits minibatch (online/out-of-core) learning. As a result, SGD makes sense for large-scale situations where it is highly efficient. Because the minimum of the Logistic Regression cost function cannot be calculated directly, we use Stochastic Gradient Descent, also known as Online Gradient Descent, to try to minimise it. For each training observation we encounter, we decrease down the cost function towards its minimum.

   Another reason to use SGD Classifier is that if we can't maintain the record in RAM, SVM or logistic regression won't function. However, SGD Classifier continues to work.

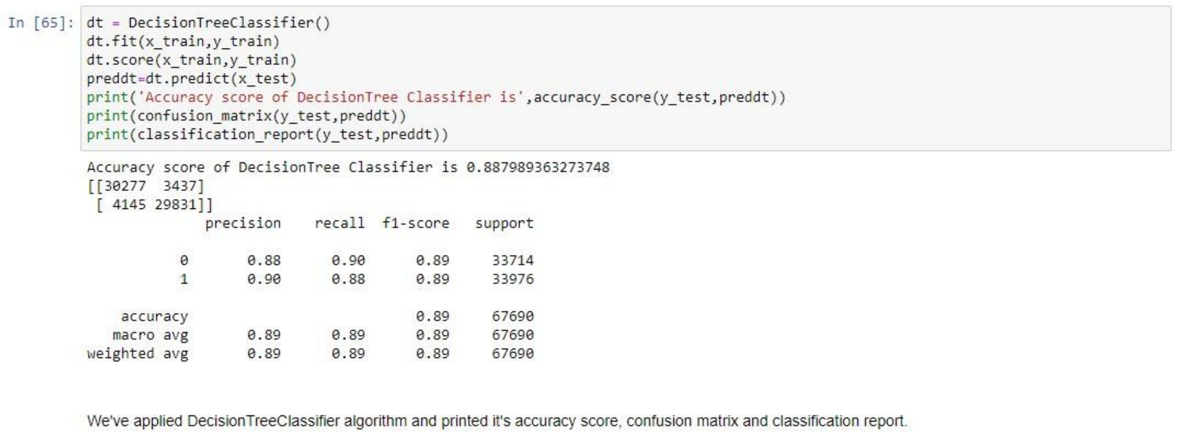


```
In [64]: sgd=SGDClassifier()
         sgd.fit(x_train,y_train)
         sgd.score(x_train,y_train)
         predsgd=sgd.predict(x_test)
         print('Accuracy score of SGDClassifier is',accuracy_score(y_test,predsgd))
         print(confusion_matrix(y_test,predsgd))
         print(classification_report(y_test,predsgd))
```

```
Accuracy score of SGDClassifier is 0.7602009159403161
[[26229  7485]
 [ 8747 25229]]
              precision    recall  f1-score   support

           0       0.75      0.78      0.76     33714
           1       0.77      0.74      0.76     33976

    accuracy                           0.76     67690
   macro avg       0.76      0.76      0.76     67690
weighted avg       0.76      0.76      0.76     67690
```

We've applied SGDClassifier algorithm and printed it's accuracy score, confusion matrix and classification report.

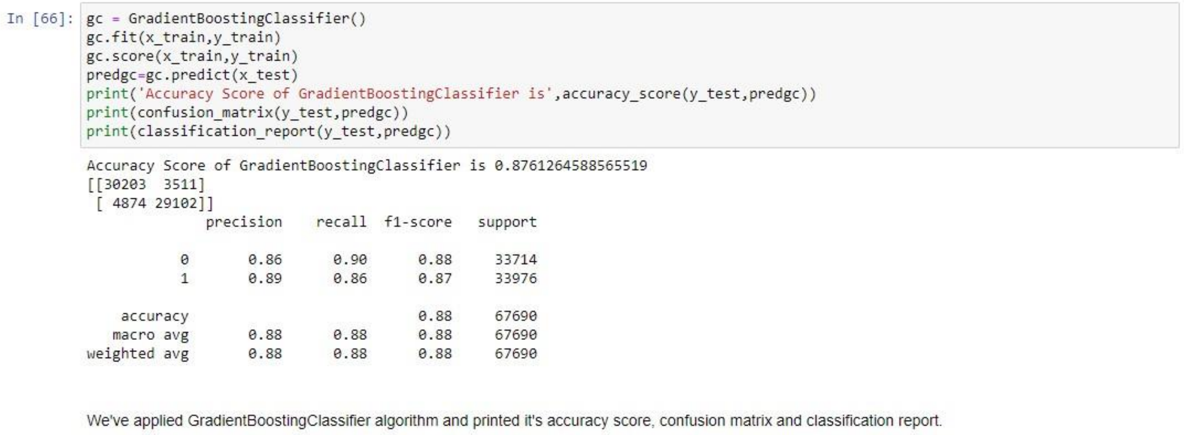
4. **Decision Tree Classifier**: Because it can tackle a wide range of issues, Decision Tree is regarded as one of the most useful Machine Learning algorithms. It is often regarded as the most intelligible and interpretable Machine Learning algorithm. The procedure for determining the class of a given dataset in a decision tree starts at the root node of the tree. This algorithm checks the values of the root attribute with the values of the record (actual dataset) attribute and then follows the branch and jumps to the next node based on the comparison.



```
In [65]: dt = DecisionTreeClassifier()
         dt.fit(x_train,y_train)
         dt.score(x_train,y_train)
         preddt=dt.predict(x_test)
         print('Accuracy score of DecisionTree Classifier is',accuracy_score(y_test,preddt))
         print(confusion_matrix(y_test,preddt))
         print(classification_report(y_test,preddt))

         Accuracy score of DecisionTree Classifier is 0.887989363273748
         [[30277  3437]
          [ 4145 29831]]
                       precision    recall  f1-score   support

                    0       0.88      0.90      0.89     33714
                    1       0.90      0.88      0.89     33976

             accuracy                           0.89     67690
            macro avg       0.89      0.89      0.89     67690
         weighted avg       0.89      0.89      0.89     67690
```

We've applied DecisionTreeClassifier algorithm and printed it's accuracy score, confusion matrix and classification report.

5. **Gradient Boosting Classifier**: Models of this type are popular because of their ability to accurately classify datasets. In most cases, decision trees are used to generate models for gradient boosting classifiers. It offers simple features that help with partitioning data into training and testing sets, as well as training, predicting, and assessing models. The aim behind "gradient boosting" is to take a weak hypothesis or learning algorithm and make a series of modifications to improve the hypothesis's/strength. learner's When doing gradient boosting, decision trees are commonly employed. Gradient boosting models are gaining popularity as a result of their ability to classify difficult information.

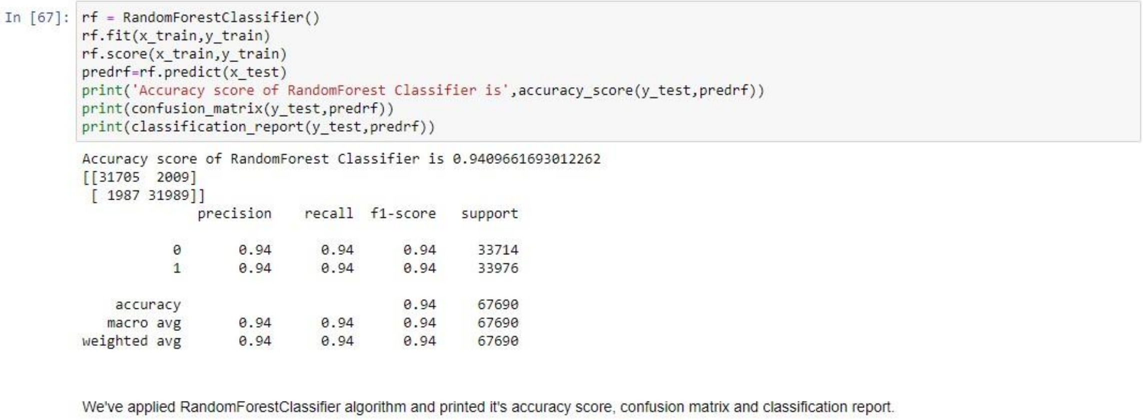


```
In [66]: gc = GradientBoostingClassifier()
         gc.fit(x_train,y_train)
         gc.score(x_train,y_train)
         predgc=gc.predict(x_test)
         print('Accuracy Score of GradientBoostingClassifier is',accuracy_score(y_test,predgc))
         print(confusion_matrix(y_test,predgc))
         print(classification_report(y_test,predgc))

         Accuracy Score of GradientBoostingClassifier is 0.8761264588565519
         [[30203  3511]
          [ 4874 29102]]
                       precision    recall  f1-score   support

                    0       0.86      0.90      0.88     33714
                    1       0.89      0.86      0.87     33976

             accuracy                           0.88     67690
            macro avg       0.88      0.88      0.88     67690
         weighted avg       0.88      0.88      0.88     67690
```

We've applied GradientBoostingClassifier algorithm and printed it's accuracy score, confusion matrix and classification report.

6. **Random Forest Classifier**: The random forest classifier is a supervised learning method that can be used to solve problems involving regression and classification. Due to its high flexibility and ease of implementation, it is one of the most common machine learning algorithms. Overfitting is one of the most common difficulties in machine learning, however

owing to the random forest classifier, this is rarely a problem. The classifier will not overfit the model if there are enough trees in the forest. The biggest drawback of random forest is that having a high number of trees can slow down the algorithm. Bagged decision tree models that split on a subset of features on each split are known as random forests. This is a big word, so let's break it down. First, we'll look at a single decision tree, then we'll talk about bagged decision trees, and then we'll talk about splitting on a random subset of features. Because we are working with subsets of data, random forests works well with high-dimensional data. Because we are only working with a subset of features in our model, it is faster to train than decision trees. We can easily work with hundreds of features. Because we can save created forests for future applications, prediction speed is substantially faster than training speed.



```
In [67]: rf = RandomForestClassifier()
         rf.fit(x_train,y_train)
         rf.score(x_train,y_train)
         predrf=rf.predict(x_test)
         print('Accuracy score of RandomForest Classifier is',accuracy_score(y_test,predrf))
         print(confusion_matrix(y_test,predrf))
         print(classification_report(y_test,predrf))
```

```
Accuracy score of RandomForest Classifier is 0.9409661693012262
[[31705  2009]
 [ 1987 31989]]
              precision    recall  f1-score   support

           0       0.94      0.94      0.94     33714
           1       0.94      0.94      0.94     33976

    accuracy                           0.94     67690
   macro avg       0.94      0.94      0.94     67690
weighted avg       0.94      0.94      0.94     67690
```

We've applied RandomForestClassifier algorithm and printed it's accuracy score, confusion matrix and classification report.

- ## Key Metrics for success in solving problem under consideration

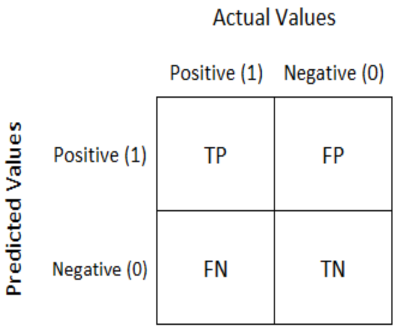
Here, we have used Accuracy, Confusion Matrix, Precision, Recall, F1 Score, Area Under the ROC Curve and ROC Curve (Receiver Operating Characteristic Curve)

1. Accuracy: The percentage of correct predictions made by our model out of all the forecasts is called accuracy. This means we add up the number of forecasts that were accurately forecasted as Positive (TP) or Negative (TN) and divide it by all sorts of predictions, both correct (TP, TN) and incorrect (TP, TN) (FP, FN). The accuracy ranges from 0 to 1. These extreme examples equate to either utterly missing or always correctly making forecasts. For example, if our model is able to forecast flawlessly,

there will be no False Positives or False Negatives, causing the numerator to equal the denominator and the Accuracy to be 1. If our system is constantly off, inaccurately forecasting each time, the number of True Positives and True Negatives will be zero, resulting in a zero divided by a positive equation, resulting in an Accuracy of 0. The maximum accuracy we got is for random forest classifier which is 0.94 .

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

2. Confusion Matrix: The so-called Confusion Matrix, which is just a table arranging the four values, is commonly given in a tabular fashion in the True Positive, True Negative, False Positive, and False Negative.



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

3. Precision: Precision, Recall, and Specificity are commonly used by us to overcome the constraints of Accuracy. Precision indicates what percentage of positive forecasts were right. It accomplishes this by dividing the total positive predictions, right or incorrect, by the number of samples correctly predicted as positive (TP) (TP, FP).

$$Precision = \frac{TP}{(TP + FP)}$$

4. Recall: Recall, like Precision, seeks to determine what percentage of actual positives were accurately identified. It accomplishes this by dividing the number of correctly projected positive samples (TP) by the total number of positives, whether correctly or wrongly forecasted as positive (TP, FN). Even if the accuracy is high, the actual performance is measured by recall. Recall = 0, means low performance of model and Recall = 1, high performance of the model.

   Recall = Sensitivity = True Positive Rate = Hit Rate

$$Recall = \frac{TP}{(TP + FN)}$$

5. F1 Score: The F1 score, which represents the harmonic mean of Precision and Recall, is a lesser-known performance metric. The maximum possible F1 Score is 1, which indicates perfect Precision and Recall, while the lowest possible score is 0, which indicates that either Precision or Recall is zero.

$$F1\ Score = \frac{2TP}{(2TP + FP + FNN)}$$

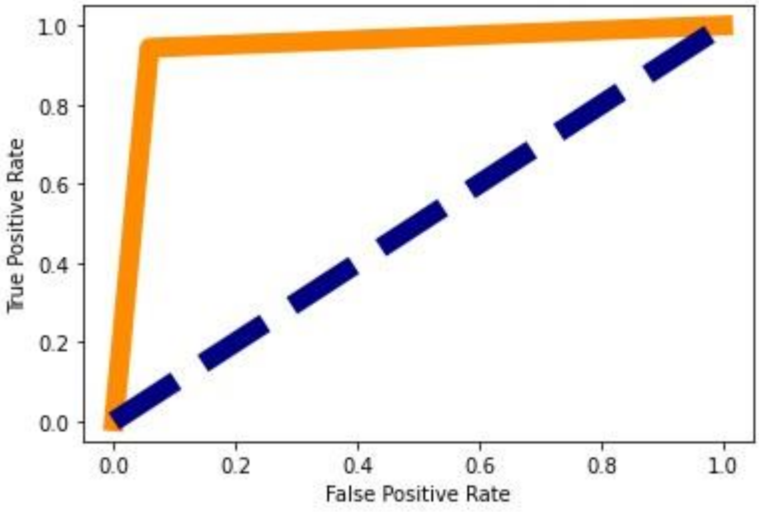
6. Area Under The ROC Curve (AUC): One of the concerns with Accuracy, as we've seen, is that it can lead to unduly inflated performance if the class distribution isn't well balanced. The full two-dimensional area beneath the complete ROC curve is measured by AUC, which stands for "Area under the ROC Curve" .
   It's a total score that takes into account all conceivable classification levels. AUC can also be thought of as the likelihood that the model will score a random positive sample higher than a random negative sample. Even though it can only be used in binary classification scenarios (i.e. not with more than 2 classes as target), AUC is a useful statistic, especially

when dealing with imbalanced classes. It is one of the most commonly used performance indicators in classification.
Instead of measuring absolute values, AUC assesses how well predictions are ranked. Regardless of the classification threshold used, AUC assesses the accuracy of the model's predictions.

7. ROC Curve (Receiver Operating Characteristic Curve): A ROC curve is a graph that shows how well a classification model performs across all categorization levels. The True Positive Rate is on the y-axis, while the False Positive Rate is on the x-axis, and the plot shows the TPR and FPR values as the threshold is changed.

   A 45-degree diagonal line is the worst-case situation (random chance). An angled line, travelling vertically first and then horizontally, is the best-case situation.

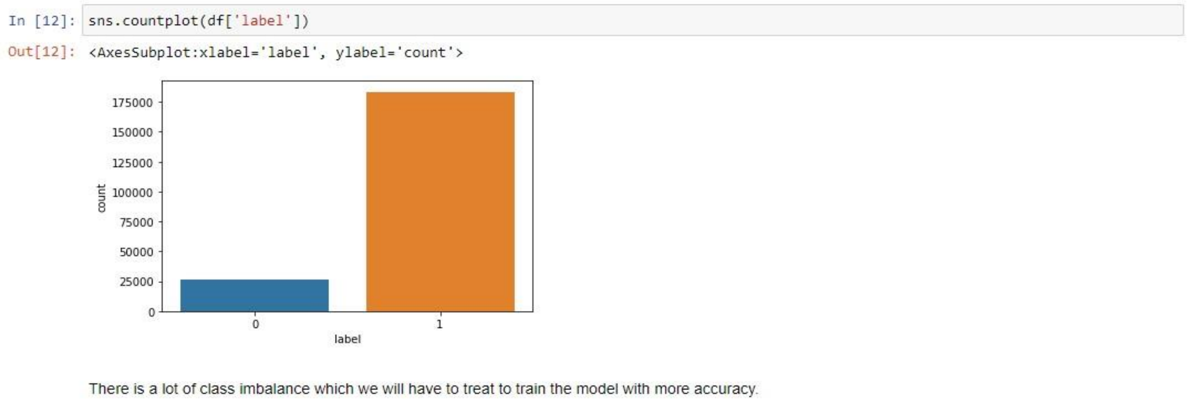
   By lowering the classification threshold, the model is able to classify more items as positive, resulting in a higher number of False Positives and True Positives.



- Visualizations

We have drawn countplot, histogram, distribution plot, kde plot, pie charts, strip plot,box plot, bar graph and heatmap.

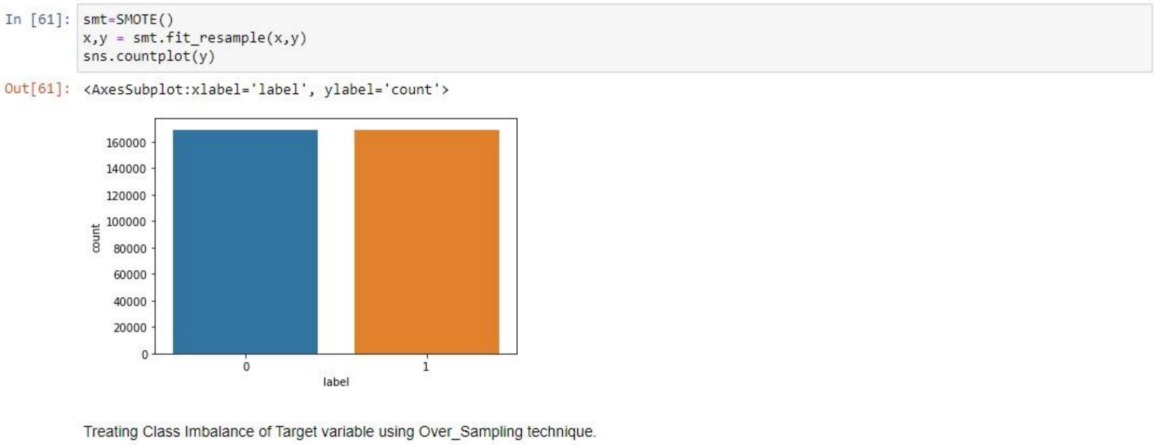
1. Countplot: A countplot counts the categories and returns the number of times they occur. It's one of the seaborn library's more straightforward plots.



```
In [12]: sns.countplot(df['label'])
Out[12]: <AxesSubplot:xlabel='label', ylabel='count'>
```

There is a lot of class imbalance which we will have to treat to train the model with more accuracy.

This is a countplot of ‘Label’, before treating class imbalance.

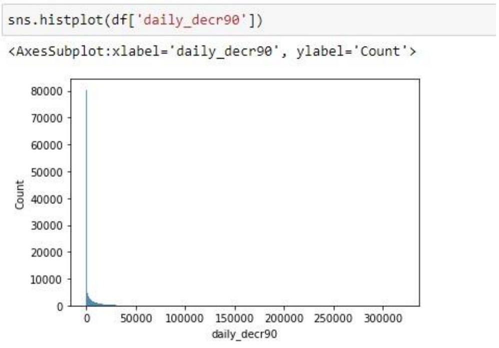


```
In [61]: smt=SMOTE()
         x,y = smt.fit_resample(x,y)
         sns.countplot(y)
Out[61]: <AxesSubplot:xlabel='label', ylabel='count'>
```

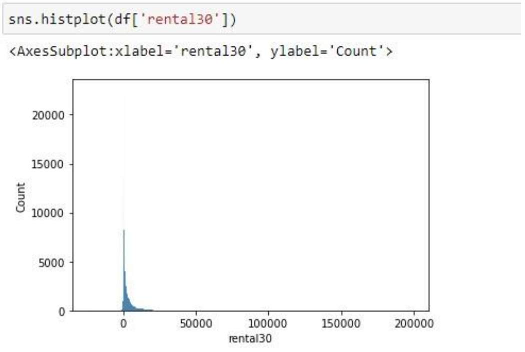
Treating Class Imbalance of Target variable using Over_Sampling technique.

This is a countplot of ‘Label’, after treating class imbalance,

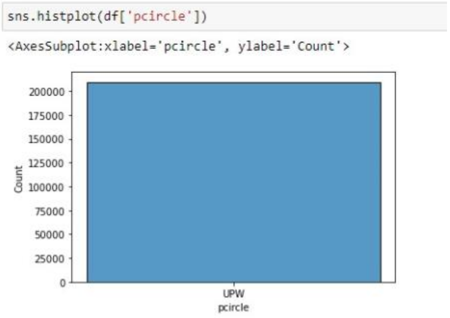
2. Histogram: A histogram is a graphical representation of continuous data in a categorical format. Unlike a bar graph, there are no gaps between the bars in a histogram. The bins are all the same width.



```
sns.histplot(df['daily_decr90'])
<AxesSubplot:xlabel='daily_decr90', ylabel='Count'>
```

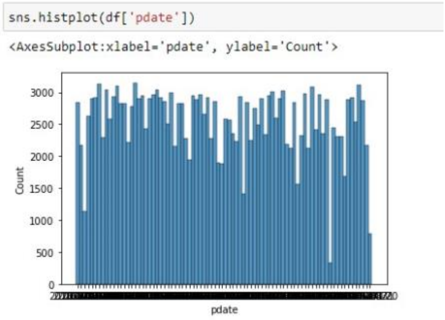


```
sns.histplot(df['rental30'])
<AxesSubplot:xlabel='rental30', ylabel='Count'>
```



```
sns.histplot(df['pcircle'])
```

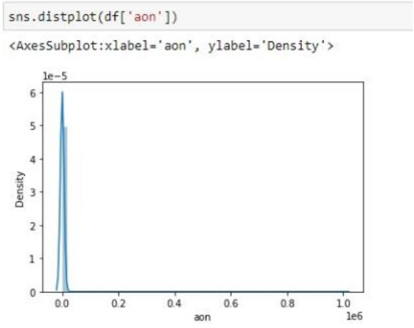
```
<AxesSubplot:xlabel='pcircle', ylabel='Count'>
```



```
sns.histplot(df['pdate'])
```

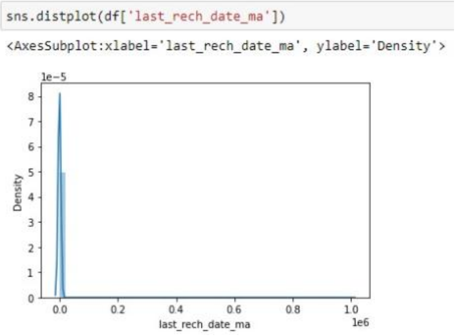
```
<AxesSubplot:xlabel='pdate', ylabel='Count'>
```

3. Distribution Plot: It is mostly used for univariant sets of observations and visualises them using a histogram, i.e. just one observation is used, and hence one column of the dataset is chosen.



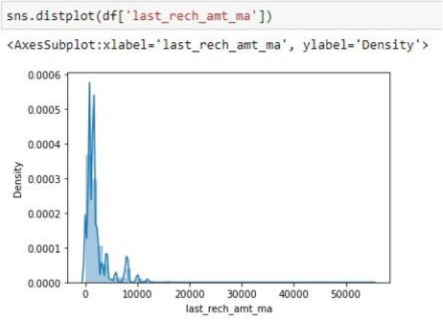
```
sns.distplot(df['aon'])
```

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



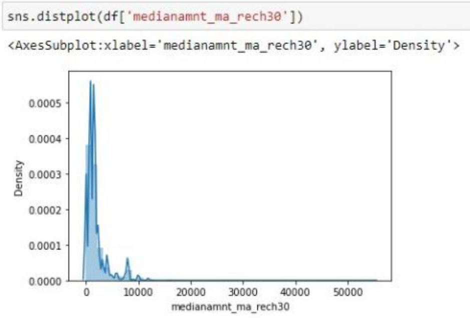
```
sns.distplot(df['last_rech_date_ma'])
```

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



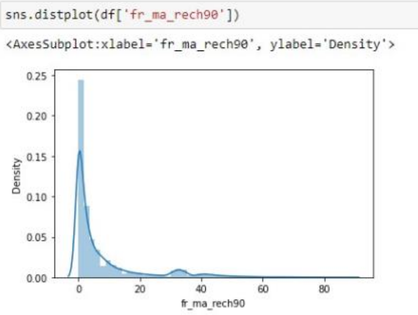
```
sns.distplot(df['last_rech_amt_ma'])
```

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



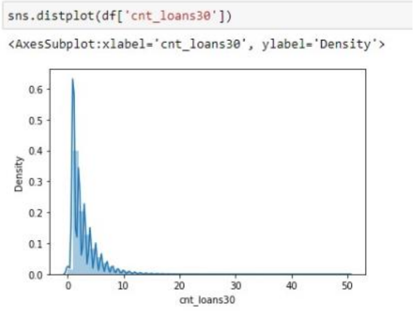
```
sns.distplot(df['medianamnt_ma_rech30'])
```

```
<AxesSubplot:xlabel='medianamnt_ma_rech30', ylabel='Density'>
```



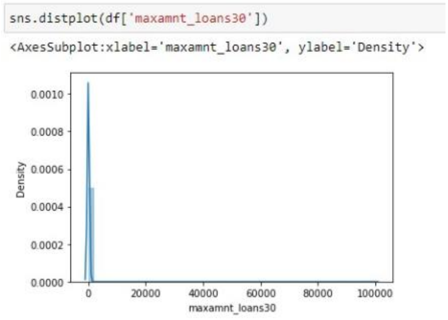
```
sns.distplot(df['fr_ma_rech90'])
```

```
<AxesSubplot:xlabel='fr_ma_rech90', ylabel='Density'>
```



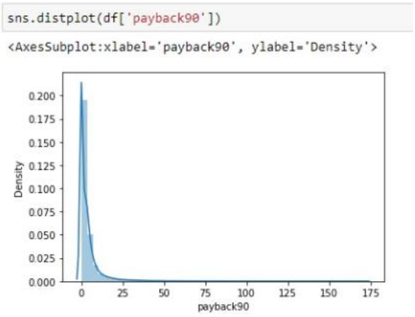
```
sns.distplot(df['cnt_loans30'])
```

```
<AxesSubplot:xlabel='cnt_loans30', ylabel='Density'>
```



```
sns.distplot(df['maxamnt_loans30'])
```

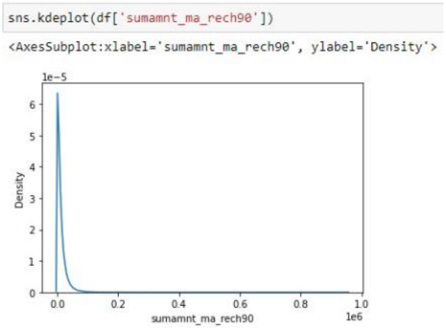
```
<AxesSubplot:xlabel='maxamnt_loans30', ylabel='Density'>
```



```
sns.distplot(df['payback90'])
```

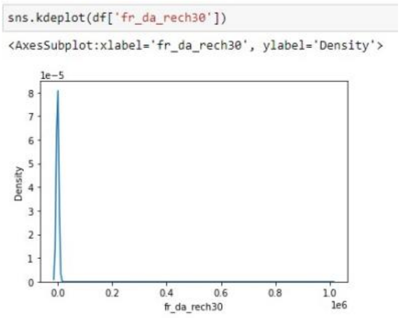
```
<AxesSubplot:xlabel='payback90', ylabel='Density'>
```

4. Kernel Density Estimate Plot: A kernel density estimate (KDE) plot, similar to a histogram, is a method for showing the distribution of observations in a dataset. KDE uses a continuous probability density curve in one or more dimensions to represent the data.



```
sns.kdeplot(df['sumamnt_ma_rech90'])
```

```
<AxesSubplot:xlabel='sumamnt_ma_rech90', ylabel='Density'>
```

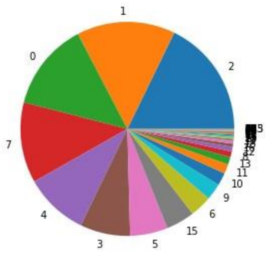


```
sns.kdeplot(df['fr_da_rech30'])
```

```
<AxesSubplot:xlabel='fr_da_rech30', ylabel='Density'>
```

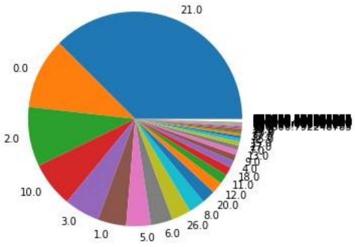
5. Pie Charts: A Pie Chart is a circular statistical layout that can only show one set of data at a time. The overall percentage of the provided data is represented by the chart's area. The proportion of sections of the data is represented by the area of the pie slices. Pie wedges are the pieces of the pie. The length of the wedge's arc determines the area of the wedge. The Python matplotlib pie chart divides the data into slices or wedges, with each slice representing the size of an item. In Python, we need to utilise the pyplot pie function to build a matplotlib pie chart.

```
plt.pie(df['cnt_ma_rech30'].value_counts(), labels=df['cnt_ma_rech30'].unique(),shadow=False)
plt.tight_layout()
plt.show()
```



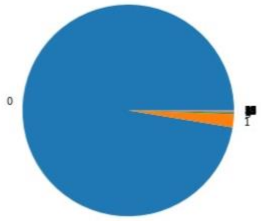
Maximum 2 , 1 , 7 and 0 Number of times main account got recharged in last 30 days

```
plt.pie(df['fr_ma_rech30'].value_counts(), labels=df['fr_ma_rech30'].unique(),shadow=False)
plt.tight_layout()
plt.show()
```

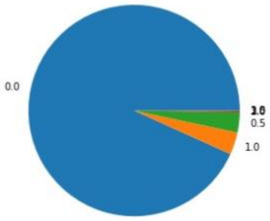


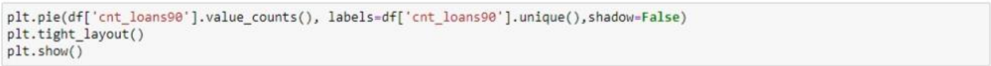
Maximum Frequency of main account recharged in last 30 days is 21.0

```
plt.pie(df['cnt_da_rech90'].value_counts(), labels=df['cnt_da_rech90'].unique(),shadow=False)
plt.tight_layout()
plt.show()
sns.distplot(df['cnt_da_rech90'])
plt.show()
```

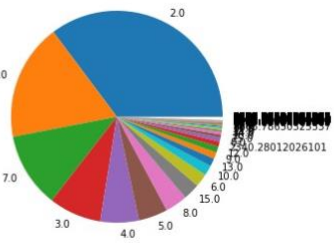


```
plt.pie(df['medianamnt_loans30'].value_counts(), labels=df['medianamnt_loans30'].unique(),shadow=False)
plt.tight_layout()
plt.show()
sns.stripplot(df['medianamnt_loans30'])
plt.show()
```



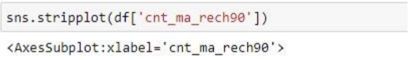


```
plt.pie(df['cnt_loans90'].value_counts(), labels=df['cnt_loans90'].unique(),shadow=False)
plt.tight_layout()
plt.show()
```

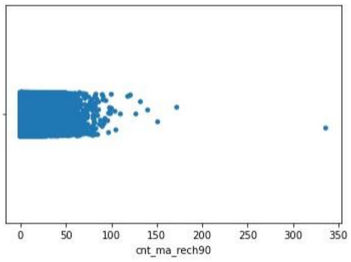


'cnt_loans90' has maximum data distribution for value 2.0

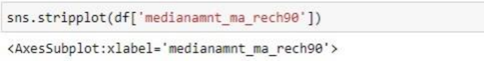
6. Strip Plot: A strip plot is a plot that is drawn on its own. In circumstances when all data are given together with some representation of the underlying distribution, it is a nice complement to a boxplot or violinplot. It's used to generate a scatter plot based on the selected category.



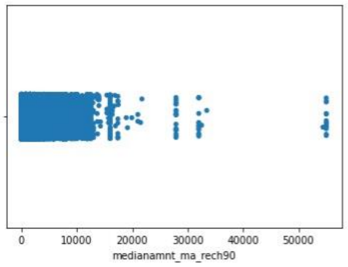
```
sns.stripplot(df['cnt_ma_rech90'])
<AxesSubplot:xlabel='cnt_ma_rech90'>
```



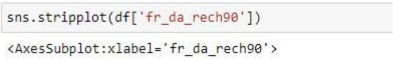
'cnt_ma_rech90' has the data scattered in range 0 to 100



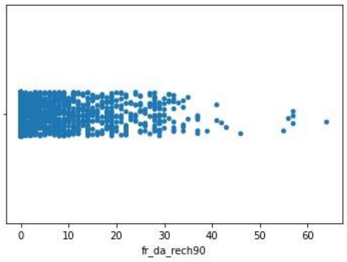
```
sns.stripplot(df['medianamnt_ma_rech90'])
<AxesSubplot:xlabel='medianamnt_ma_rech90'>
```



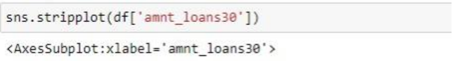
'medianamnt_ma_rech90' has maximum data scattered in range 0 t0 15000



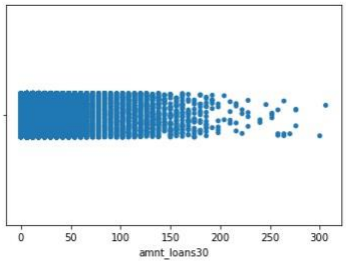
```
sns.stripplot(df['fr_da_rech90'])
<AxesSubplot:xlabel='fr_da_rech90'>
```



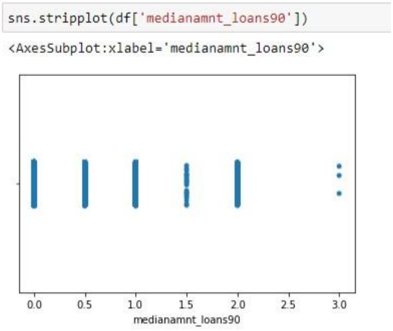
'fr_da_rech90' has maximum data scattered in range 0 to 30



```
sns.stripplot(df['amnt_loans30'])
<AxesSubplot:xlabel='amnt_loans30'>
```

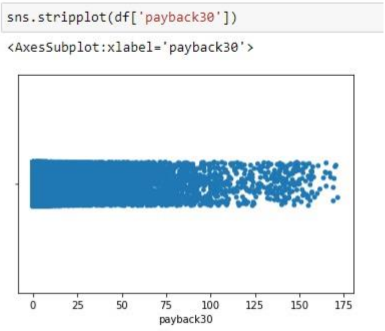


'amnt_loans30' has maximum data scattered over a range of 0 to 200



```
sns.stripplot(df['medianamnt_loans90'])
```

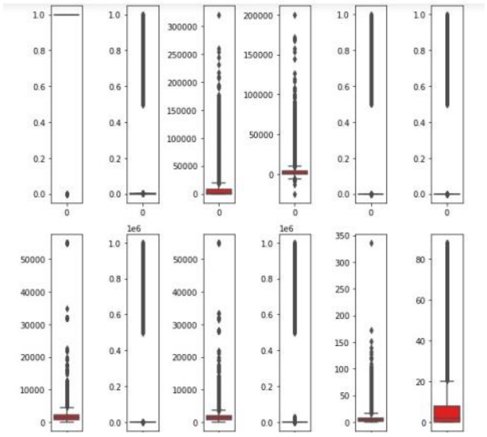
```
<AxesSubplot:xlabel='medianamnt_loans90'>
```

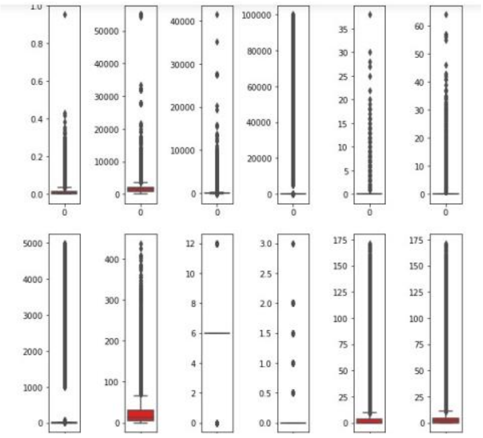


```
sns.stripplot(df['payback30'])
```

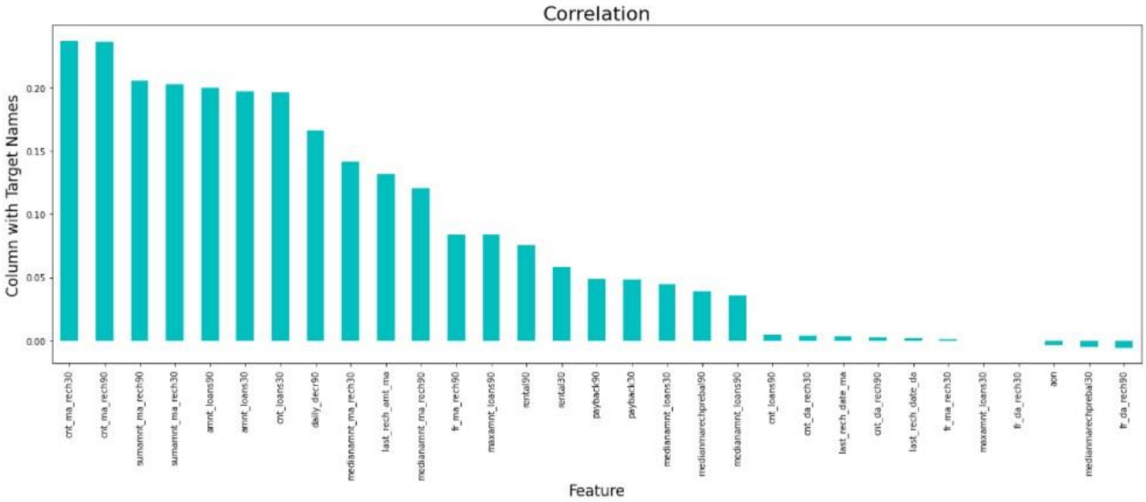
```
<AxesSubplot:xlabel='payback30'>
```

7. Box Plot: A single box plot can be used to illustrate multiple statistics from a vast quantity of data. It uses a number line to show the range and distribution of data. Box plots provide some insight into the symmetry and skewness of the data. Outliers are also visible in box plots.

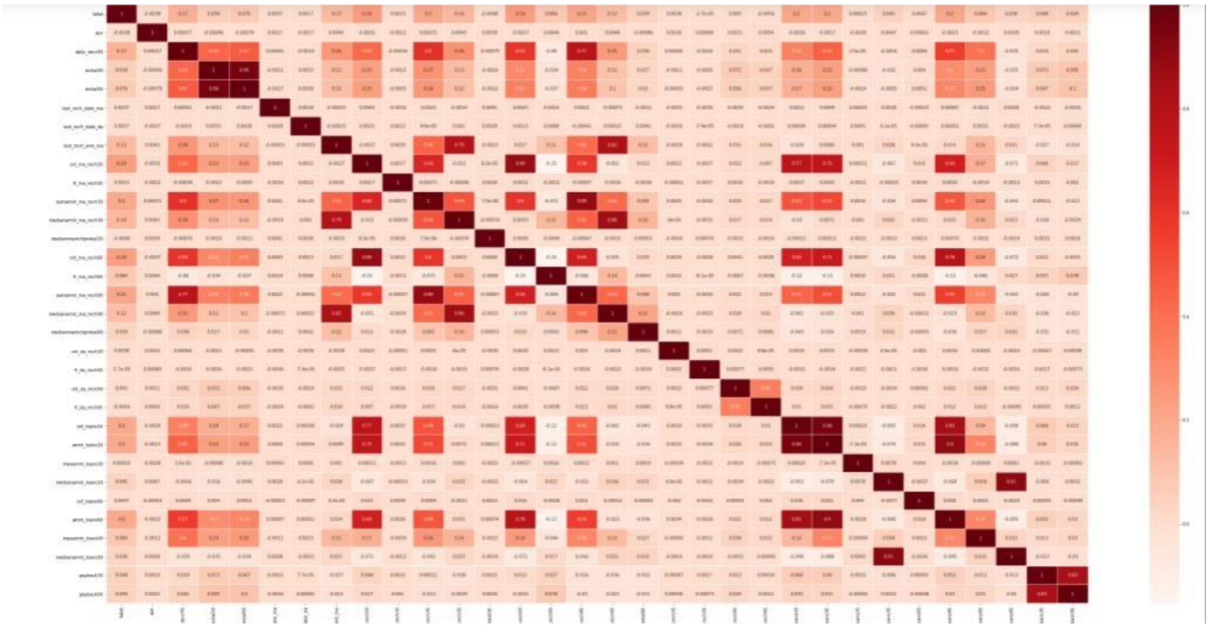




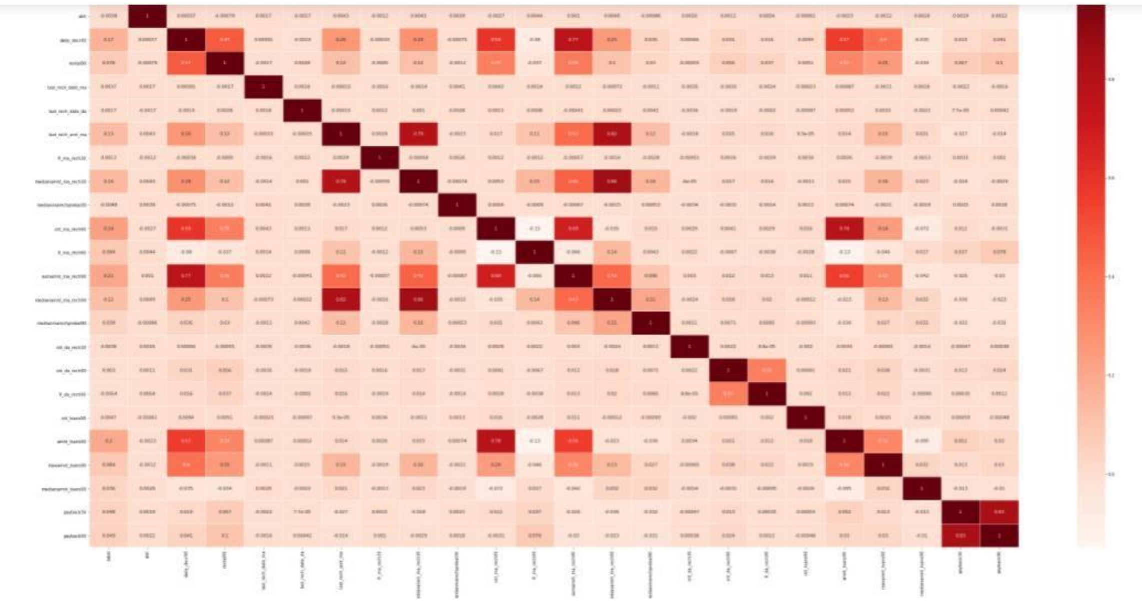
8. Bar Graph: The bar graph makes it simple to compare various sets of data among different groups. It depicts the connection using two axes, with discrete values on one axis and categories on the other. The graph depicts the most significant changes in data over time. A bar graph is a chart that graphically depicts the comparison of several data types. It uses parallel rectangular bars of identical width but varied length to show grouped data. Each rectangular block represents a distinct category, and the length of the bars is determined by the values they represent.



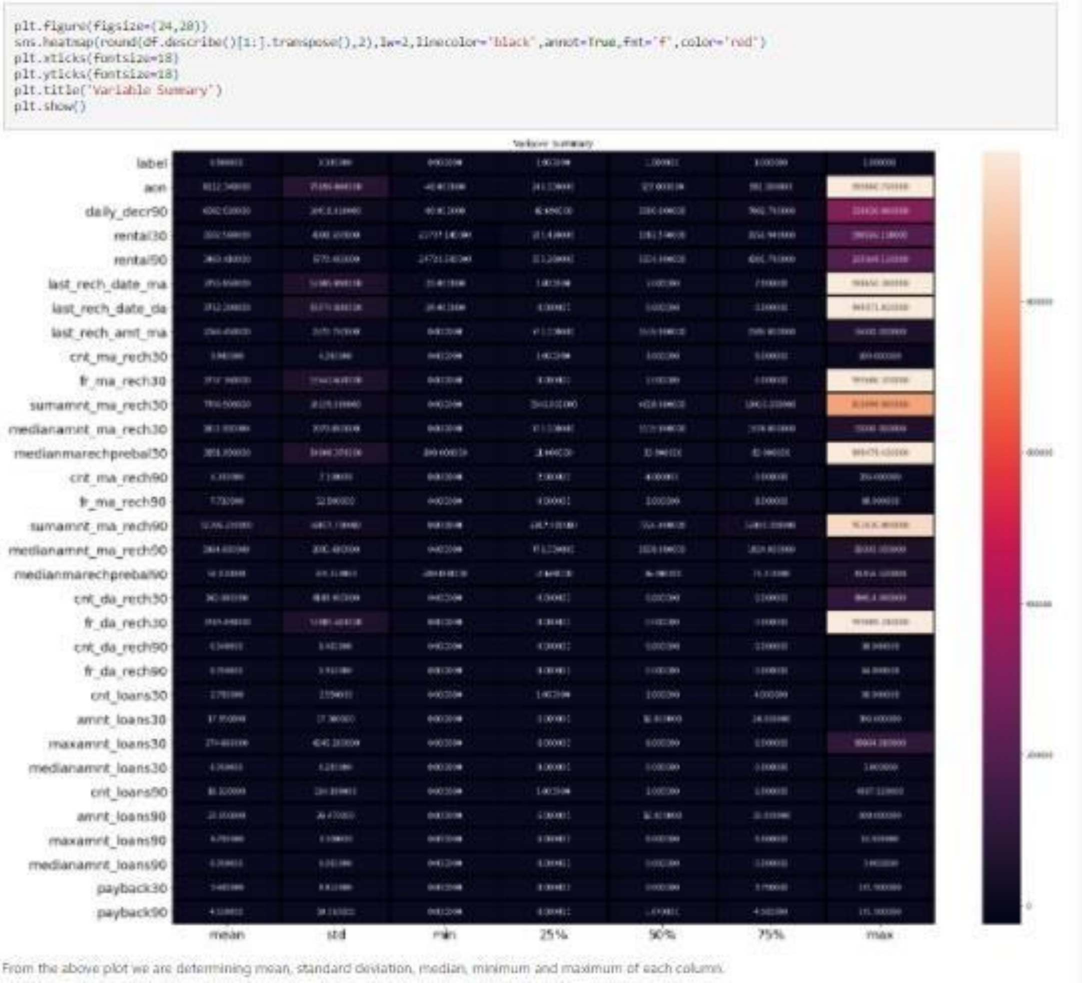
9. Heatmap: A heatmap is a two-dimensional graphical representation of data that uses colours to represent the individual values in a matrix. In a color-coding scheme, each data value is represented by a colour. The bigger the amount, the deeper the shade; the higher the value, the narrower the dispersion, and so on. Perfect for displaying data that has a clear border that is important to the data. The seaborn python package allows users to create annotated heatmaps that may be adjusted using Matplotlib capabilities to meet their needs.



This is the heatmap having a lot multi-collinearity.



This is the heatmap after we have treated multi-collinearity.



```
plt.figure(figsize=(24,28))
sns.heatmap(round(df.describe()[1:].transpose(),2),lw=2,linecolor='black',annot=True,fmt='f',color='red')
plt.xticks(fontsize=18)
plt.yticks(fontsize=18)
plt.title('Variable Summary')
plt.show()
```

From the above plot we are determining mean, standard deviation, median, minimum and maximum of each column.

A heatmap representing statistical details of the dataset.

- Interpretation of the Results

From the visualization we have interpreted that the target has class imbalance which needs to be treated. There are many features with same same data distribution and same data value, just defined over different time periods, which are causing multi-collinearity. Hence we will have to drop few features to treat this multi-collinearity. Also, there are so many outliers in the dataset but if we remove all, we will be left with very less dataset. In order to not to lose a large part of dataset, we will have to treat with zscore value greater than 5. Also the few features have skew-ness and we will have to apply power transformation to treat such skewness. The features are not scaled properly. Hence we will also have to scale them. While reading the heatmap we also find that there are two features which are not related to target at all, hence we will also drop that. Date and Pcirlce does not adds much value to the target prediction, hence we will remove those features as well.

After applying the algorithms, we find out that accuracy, precision, cross validation and F1 score of Random Forest Classifier is quite appreciable, hence we will apply hyper parameter tuning over Random forest classifier and use the hyper parameter tuned model as our final model for ‘Label’ prediction. This hyper parameter tuned final Random Forest Classifier gives us 94.23% accuracy.

CONCLUSION

- ## Key Findings and Conclusions of the Study

  Mobile Money is defined as a service that allows unbanked and lowincome people to access financial services such as payments for goods, services, and bills through mobile device devices.

  The spread of Information and Communication Technologies has been shown to be a complicated process that can also be observed at the country level. Economy, culture, technology, and politics are among the country-level factors. The regulatory environment, existing alternatives, agents' behaviours, the cellular market landscape, and service providers' market share were the primary elements that influenced mobile money diffusion.

  To summarise, this study's theoretical framework and synthetic analysis are concerned with mobile financial services in various economic and social viewpoints and circumstances.

- ## Learning Outcomes of the Study in respect of Data Science

  Identifying faulty components (the features that are repeating themselves and just adding baggage to the dataset), developing solution plans (cleaning and preprocessing the data to effieciently build models), and putting the necessary changes in place without jeopardising the present system's functionality. To arrive at a realistic solution, the individual problem must be abstracted and decomposed. Identifying components that might help to solve a problem. Large-scale informatics efforts require analysis. The well the dataset has been analysed, the well will be the accuracy of the model. Identifying jobs that can be automated, comprehending underlying corporate processes, and assessing business wants.

- ## Limitations of this work and Scope for Future Work

  We recommend learning more about the distinctions between mobile banking and mobile money services.

It's crucial to figure out why one service is more effective in specific nations than the other. It would be fascinating to compare and contrast developed and developing countries.

Another option is to look at the elements that influence the spread of mobile financial services in developing nations, such as country culture, financial literacy, technological progress, and the financial system.

M-Pesa is one of the most popular mobile money systems in developing countries. However, this service did not perform as well as predicted in Eastern Europe. It would be worthwhile to conduct a thorough investigation of the circumstances that contributed to their failure.

This study shows that research on mobile phones and finance in developing countries is rapidly expanding, and there has been no systematic attempt to examine how this research has developed both theoretically and methodologically. This paper aims to bridge that gap. It is hoped that the studies presented here are representative of the discipline, and that the authors' interpretations of those studies appropriately reflect the research carried out.

In response to the growing importance of mobile-based financial inclusion initiatives in practise, this study examines existing academic literature on the three themes of mobile financial services, financial inclusion, and development in order to gain a better understanding of the current research landscape and identify potential gaps.

We classify the articles into three main clusters, namely delivery, environmental factors, and impact, and then analyse the key issues that emerge from the current articles: agent network, interoperability, intention, perception, usage pattern, regulation, sociocultural factors, demographic, financial inclusion impacts, and economic development, using the systematic review.

According to our findings, existing research on MFS, financial inclusion, and development is still in its infancy. Using the Heeks model, we also discover that the important topics covered in the literature are primarily concerned with MFS preparation. Topics relevant to the later phases of the MFS value chain, such as availability, uptake, and impacts, on the other hand, are very briefly explored. More research is needed to address unexplored areas like wellorganized business models, genuine customer requests, and quantitative and qualitative analysis of MFS benefits and hazards. We also discovered that articles on MFS supply prefer to employ qualitative approaches, whereas studies on MFS demand rely primarily on quantitative methods.

Future study should balance out such a disproportionate use of methodologies, and the use of more innovative or blended methods to expand our grasp of the topic is encouraged in the future. From the standpoint of MFS practise, these research gaps also indicate implications. More study into the needs of potential and current consumers, as well as their actual usage patterns, could help MFS providers build new business models.

The breadth of our research is limited by the articles we looked at, which are mostly peer-reviewed academic works written in English. To accomplish our research goal of comprehending the academic environment, we purposefully excluded policy reports and grey literature published by active organisations in the MFS area (e.g., World Bank, IMF, and CGAP, GSMA, etc.).