SVKM'S-NMIMS (Deemed-to-be-University),
Indore Campus

School of Technology Management &

Engineering [2022-23]

Project Report on

Exploratory Data Analysis on Loan Defaulting

Under the guidance of

Dr. Prachi Gharpure and Dr. Aaquil Bunglowala

Submitted In partial fulfillment of requirements for the degree of

MBA.Tech. (CE) Program

In ENGINEERING & TECHNOLOGY

Submitted By -

Bhavya Sharma
Ishica Thukral
Purva Patel

SVKM'S-NMIMS (Deemed-to-be-University), Indore Campus

School of Technology Management & Engineering

Super Corridor Rd, Gandhi Nagar, Indore, Madhya Pradesh 452005

Project Phase — MBA. Tech

Submitted in Partial fulfillment of the requirements for Project Phase MBA Tech.

Name of the Student: Bhavya Sharma , Ishica Thukral , Purva Patel

Roll No. & Batch: N310 , N313 , N318;

2019-24

Academic Year: 2022-23

THIS IS TO CERTIFY THAT

Mr. Bhavya Sharma, Ms. Ishica Thukral, Ms. Purva Patel Exam Seat No. N310 , N313 , N318 Satisfactorily Completed the Project Work, submitted the project report, and appeared for the Presentation as required.

Date:

Place: NMIMS Indore

Chairperson/Dean

Seal of the University

DECLARATION

We, Bhavya Sharma, Ishica Thukral, Purva Patel, Roll No. N310, N313, N318 MBATech (Computer Engineering), VII semester understand that plagiarism is defined as anyone or combination of the following:

1. Un-credited verbatim copying of individual sentences, paragraphs or illustration (such as graphs, diagrams, etc.) from any source, published or unpublished, including the internet.

2. Un-credited improper paraphrasing of pages paragraphs (changing a few words phrases, or rearranging the original sentence order)

3. Credited verbatim copying of a major portion of a paper (or thesis chapter) without clear delineation of who did write what. (Source: IEEE, The institute, Dec. 2004)

4. I have made sure that all the ideas, expressions, graphs, diagrams, etc., that are not a result of my work, are properly credited. Long phrases or sentences that had to be used verbatim from published literature have been clearly identified using quotation marks.

5. I affirm that no portion of my work can be considered as plagiarism and I take full responsibility if such a complaint occurs. I understand fully well that the guide of the seminar/ project report may not be in a position to check for the possibility of such incidences of plagiarism in this body of work.

Signature of the Student:

Name:

Roll No.

Place:

Date:

CERTIFICATE

This is to certify that the project entitled

"Exploratory Data Analysis on Loan Defaulting" is the Bonafede work carried out by Bhavya Sharma, Ishica Thukral and Purva Patel of MBA.Tech (Computer Engineering), MPSTME (NMIMS), Indore, during the VIIth semester of the academic year 2022-23, in partial fulfillment of the requirements for the award of the Degree of Bachelors of Engineering as per the norms prescribed by NMIMS. The project work has been assessed and found to be satisfactory.

Dr. Prachi Gharpure

Internal Mentor

Dr. Aaquil Bunglowala

Internal Mentor

_____ _____

Examiner 1 Examiner 2

_____

HOD

Table of Contents

List of Figures

37	Figure 37-birth year	33
38	Figure 38-if he/she changed their number	34
39	Figure 39-YEARS_ID_PUBLISH attribute	35
40	Figure 40-Number of years they have been employed	36
41	Figure 41-years of registration	37

List of Tables

Abstract-

This project tries to find trends that show whether a client has trouble making their payments, which may be used to decide whether to grant the loan, reduce its size, charge riskier applicants a higher interest rate, etc. By doing this, it will be ensured that only borrowers who can repay the loan will be accepted. The objective of this project is to identify such applications using EDA.

Data analysis utilizing visual methods is called exploratory data analysis (EDA). With the use of statistical summaries and graphical representations, it is used to identify trends, and patterns, or to verify assumptions.

We have two datasets, first is the current information of a client called an “application dataset" and the second dataset contains past information of a client which is the “previous dataset”.

An EDA has been performed on the dataset, We are extracting features/attributes from the 122 attributes which will help us to know if the person will be a loan defaulter or not.

We have applied all the EDA steps containing-

1. Importing Libraries
2. Reading Data
3. Descriptive Statistics
4. Missing value imputation
5. Graphical Representation
6. Univariate analysis
7. Bivariate analysis
8. Multivariate analysis

We were able to comprehend which consumer attributes and loan attributes influence the tendency of default and to find which customers are more likely to default on their loan payment.

Chapter 01 – Introduction

1.1 Problem definition:

To bring financial inclusion banks have been encouraged to give easy loans so that more and more people get financially dependent. Giving out loans always comes with the risk of defaulters and this makes giving loans difficult for banks to decide who might default and accordingly give loans. Due to weak or nonexistent credit histories, loan providers find it challenging to grant loans to individuals. Because of this, some customers take advantage of it by defaulting. Imagine you work for a consumer finance company that specializes in providing urban customers with several kinds of loans. To analyze the patterns found in the data, you must employ EDA. This will prevent the applicants from being turned down based on their ability to repay the loan.
Our purpose of the exploratory analysis mainly consists of

- To comprehend which consumer attributes and loan attributes influence the tendency of default.
- To find which customers are more likely to default on their loan payments.
- Find the key features for making the decision

To understand the characteristics that are reliable predictors of loan default, also known as the driving factors (or driver variables) behind loan default. This information can be used by the business in portfolio management and risk analysis.

1.2 Domain Introduction:

Credit risk analysis will help the banks to make a decision for loan approval based on the applicant's profile. Which controls the loss of business to the company and avoids financial loss for the company.
When a loan application is received, the business must evaluate whether to approve the loan based on the applicant's profile. The bank's choice is subject to two different kinds of risks:

If the borrower is likely to repay the loan, refusing to grant it results in the company losing business. If the borrower is not likely to pay back the loan, or is likely to default, then approving the loan may result in a loss of revenue for the business. The information regarding the loan application at the time of applying for the loan is contained in the data presented below. It has two different kinds of scenarios:

- The client experiencing payment difficulties: He/she was more than X days overdue on at least one of the loan's first Y payments in our sample,
- All other situations: Every situation in which the payment is made on time.

Chapter 02 – Environment and Datasets

2.1 Environment:

Colab- Colaboratory, sometimes known as "Colab," is a Google Research product. Colab is particularly well suited to machine learning, data analysis, and education. It enables anyone to create and execute arbitrary Python code through the browser.
We specifically used this platform because colab files are collaborative, which makes it easy to work with other cohorts.

Numpy- Large, multi-dimensional arrays and matrices are supported by NumPy, a library for the Python programming language, along with a substantial number of high-level mathematical operations that may be performed on these arrays.
NumPy objects are primarily used to create arrays or matrices that can be applied to DL or ML models which helped us to define the lengths of arrays used in our project.

Pandas- For the purpose of manipulating and analyzing data, the Python programming language has a software package called pandas. It includes specific data structures and procedures for working with time series and mathematical tables.

Matplotlib- For the Python programming language and its NumPy numerical mathematics extension, Matplotlib is a graphing library. We used this library to plot all the graphs which were necessary to extract the features for detecting loan defaulters.

Seaborn- A package called Seaborn uses Matplotlib as its foundation to plot graphs. In order to see random distributions, it will be used.

Scipy- SciPy is a Python library that is available for free and open source and is used for technical and scientific computing. SciPy includes modules for a variety of common tasks in science and engineering, including optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, and ODE solvers.

EDA- Data analysis utilizing visual methods is called exploratory data analysis (EDA). With the use of statistical summaries and graphical representations, it is used to identify trends, patterns, or to verify assumptions.

2.2 About Dataset:

We used two datasets to draw results application_data and previous_application, these are available on Kaggle.com

'application_data': credit history of clients with 122 numerical and categorical attributes about the customer information.

'previous_application': whether the customer will be able to pay loan does depend on a previous application, so we will be using this data with merging it to applicant data, to get more clear about which Applicants are more likely to default on a loan.

The first dataset "application_dataset" had 122 attributes and it tells us the information of clients, It had a 'TARGET' variable which was boolean where, 0 meant that the person has never been a defaulter and 1 meant that the person has been a loan defaulter before. Another is "previous_application" dataset which had 37 attributes containing the past information of the clients, which was not enough and did not contain those features which could be necessary to know if the person will loan default or not.

After data cleaning in the "application_dataset" we narrowed our 122 attributes to 25 attributes and it helped us to extract the necessary features to predict whether the client will be a loan defaulter or not, and to know if it is safe to give a large sum of money to a person and to trust that they can repay it with an interest.

The following are the 25 parameters that were narrowed down from 122 attributes:

S No.	Column	Description	Reason to choose this attribute
01	EXT_SOURCE_2	Normalized score from external data source	Customer is scored from between 0 to 1, may tell credit worthiness
02	YEARS_BIRTH	Client's age in years at the time of application	Age may be a factor to determine credit allotment. high correlation with TARGET
03	YEARS_LAST_PHONE_CHANGE	How many years before the application did the client change phone	How frequently the client changes phone may determine creditworthiness. high correlation with TARGET
04	YEARS_ID_PUBLISH	How many years before the application did the client change the identity document with which he applied for the loan.	Identity document plays an important role in determining creditworthiness

05	YEARS_EMPLOYED	How many years before the application the person started current employment	For how long the customer is employed can be deciding factor for credit allotment.
06	YEARS_REGISTRATION	How many years before the application did the client change his registration	High correlation value with the TARGET.
07	EXT_SOURCE_3	Normalized score from external data source	Customer is scored from between 0 to 1, may tell credit worthiness
08	REGION_RATING_CLIENT_W_CITY	Our rating of the region where client lives with taking city into account (1,2,3)	Score of where the client lives(city) may determine credit worthiness
09	REGION_RATING_CLIENT	Our rating of the region where client lives (1,2,3)	Score of where the client lives(region) may determine credit worthiness
10	REG_CITY_NOT_WORK_CITY	Flag if client's permanent address does not match work address (1=different, 0=same, at city level)	If permanent add doesn't match work add may affect loan payment. high correlation with TARGET.
11	REG_CITY_NOT_LIVE_CITY	Flag if client's permanent address does not match contact address (1=different, 0=same, at city level)	If permanent add doesn't match contact add may affect loan payment.
12	FLAG_EMP_PHONE	Did client provide work phone (1=YES, 0=NO)	If the client has a work phone it may affect creditworthiness.
13	FLAG_DOCUMENT_3	Did client provide document 3	it had the least missing values and high correlation with TARGET.
14	NAME_CONTRACT_TYPE	Identification if the loan is cash or revolving	Nature of the loan may affect the creditworthiness.
15	CODE_GENDER	Gender of the client	Least missing values and high correlation with TARGET

16	FLAG_OWN_CAR	Flag if the client owns a car	Owning a car can be a deciding factor for loan approval. High correlation with TARGET
17	FLAG_OWN_REALTY	Flag if the client owns a house or flat	Shows property owned by the client, may affect loan repayment abilities of the client.
18	NAME_TYPE_SUITE	Who was accompanying client when he was applying for the loan	high correlation with TARGET
19	NAME_INCOME_TYPE	Clients income type (businessman, working, maternity leave,…)	Income type affects loan repayment ability of the client
20	NAME_EDUCATION_TYPE	Level of highest education the client achieved	People with more education seem to have less default history.
21	NAME_FAMILY_STATUS	Family status of the client	Married, single, divorced, etc may help with creditworthiness.
22	NAME_HOUSING_TYPE	What is the housing situation of the client (renting, living with parents, ...)	house / apartment, with parents , municipal apartment, rented apartment, office apartment, co-op apartment factors affect the creditworthiness.
23	OCCUPATION_TYPE	What kind of occupation does the client have	Labourers, sales staff, core staff, managers, drivers etc factors may be considered for creditworthiness.
24	WEEKDAY_APPR_PROCESS_START	On which day of the week did the client apply for the loan	Less missing values and high correlation with TARGET
25	ORGANIZATION_TYPE	Type of organization where the client works	Business entity, XNA, self employed etc, high correlation with TARGET

Table 1: application_dataset important attributes

In “previous_application” dataset we narrowed down to 15 attributes from 37 attributes, which were necessary for our predictive analysis, the attributes are-

S no.	**Column**	**Description**	**Reason to choose this attribute**
01	NAME_CONTRACT_TYPE	Contract product type (Cash loan, consumer loan [POS],...) of the previous application	Defaulter has more consumer loans than cash loans hence it is a good feature.
02	WEEKDAY_APPR_PROCESS_START	On which day of the week did the client apply for previous application	Column seems balanced and no small category with a high correlation TARGET.
03	NAME_CONTRACT_STATUS	Contract status (approved, canceled, ...) of previous application	Defaulters have more “refused” and “canceled” hence it's a valid feature.
04	NAME_PAYMENT_TYPE	Payment method that client chose to pay for the previous application	Payment type may affect loan repayment ability of the client
05	CODE_REJECT_REASON	Why was the previous application rejected	high correlation with TARGET
06	NAME_CLIENT_TYPE	Was the client old or new client when applying for the previous application	May affect loan repayment capability.
07	NAME_GOODS_CATEGORY	What kind of goods did the client apply for in the previous application	high correlation with TARGET
08	NAME_PRODUCT_TYPE	Was the previous application x-sell or walk-in	May affect loan repayment capability.
09	CHANNEL_TYPE	Through which channel we acquired the client on the previous application	high correlation with TARGET
10	NAME_SELLER_INDUSTRY	The industry of the seller	Less missing value with high correlation with TARGET value.

11	NAME_YIELD_GROUP	Grouped interest rate into small medium and high of the previous application	high correlation with TARGET
12	PRODUCT_COMBINATION	Detailed product combination of the previous application	Less missing value with high correlation with TARGET value.
13	HOUR_APPR_PROCESS_START	Approximately at what hour did the client apply for the loan	high correlation with TARGET
14	DAYS_DECISION	Relative to current application when was the decision about previous application made	Less missing value with high correlation with TARGET value.
15	CNT_PAYMENT	Term of previous credit at the application of the previous application	High correlation with TARGET value. Defaulter and non-defaulter have peaks at the same values but for Non-Defaulter it is high.

Table 2: previous_application dataset important attributes

Note: Correlation shows the strength of a relationship between two variables and is expressed numerically by the correlation coefficient. The correlation coefficient's values range between -1.0 and 1.0. A perfect positive correlation means that the correlation coefficient is exactly 1.

Chapter 03. Literature Review

To comprehend the traits, sometimes referred to as the driving factors (or driver variables), that are accurate predictors of loan default. We had to study the background regarding creditworthiness.

Analyzing the difficulties of efficiency of commercial banks' credit operations,the author has paid special attention to such issues like the process of management and assessment of borrowers' creditworthiness, as well as formation of credit portfolio in unsteady conditions of economics.The main risk of formation of commercial bank's credit portfolio is made up of the credit risk. (A. Chaplinska)

EDA is a fundamental early step after data collection and preprocessing, where the data is simply visualized, plotted, manipulated, without any assumptions, in order to help assess the quality of the data and build models. "Most EDA techniques are graphical in nature with a few quantitative techniques. The reason for the heavy reliance on graphics is that by its very nature the main role of EDA is to explore, and graphics gives the analysts unparalleled power to do so, while being ready to gain insight into the data. There Are many ways to categorize the many EDA techniques" (Matthieu Komorowski, Dominic C. Marshall, Justin D. Salciccioliand Yves Crutain).

The effect of financial institution type and firm-related characteristics on loan amounts advanced. The results show that the preferred credit provider matters, with the sensitivity level varying among the three institutional types. Additionally, the collateralization value, the owner's equity proportion of fixed assets, and any existing credit facility correlate positively with the outcome variable. There is an inverse relationship between the largest shareholder's ownership and the loan amount. (Dr. Edmund Bwire)

Chapter 04. System Analysis and Design

4.1 Use Case:

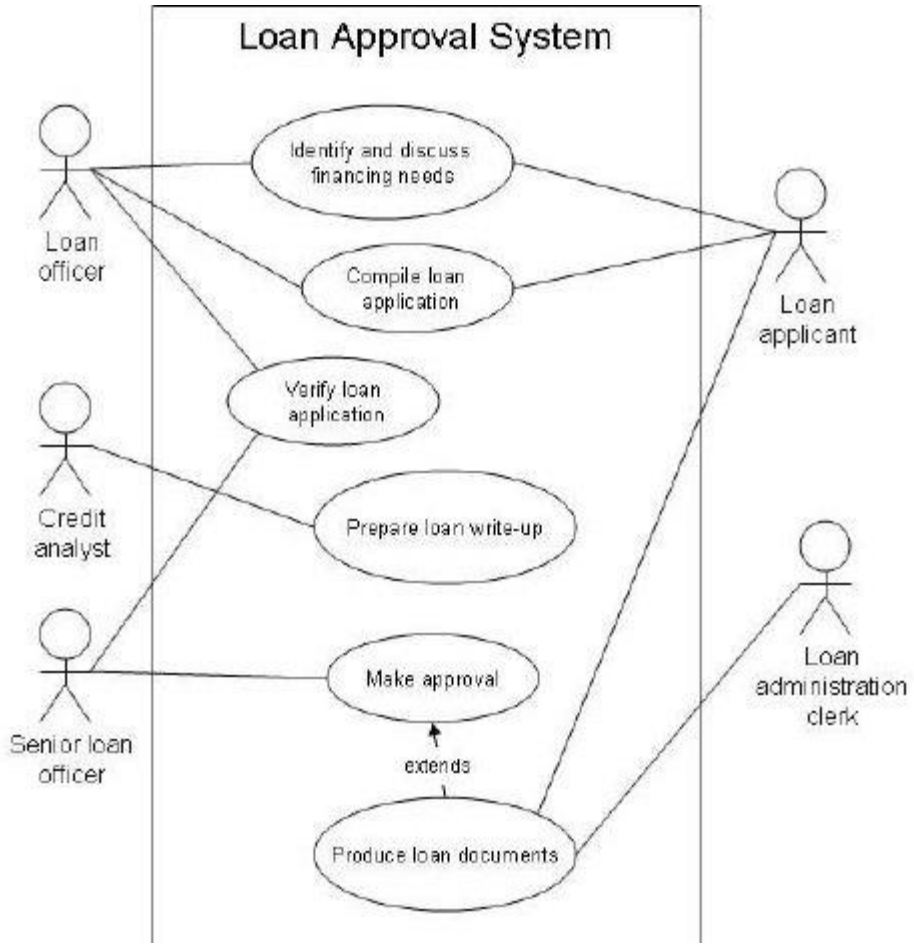


Figure 1- use case diag

First,a Loan applicant applies for a loan application, then the loan officer identifies the financial needs of the applicant and suggests what type of loan he/she may be interested in.
Then the loan officer compiles the loan application of the applicant and then it is verified by the senior loan officer.
Credit analyst comes into place and prepares a loan write-up, he/she ensures whether the loan applicant is eligible for getting the loan desired by them by checking their credit score.
After this, The loan administration clerk produces loan documents with the help of the loan applicant by getting their details.
These loan documents are then approved by the Senior loan officer.

4.2 Process diagram:

4.2.1 For application_data:

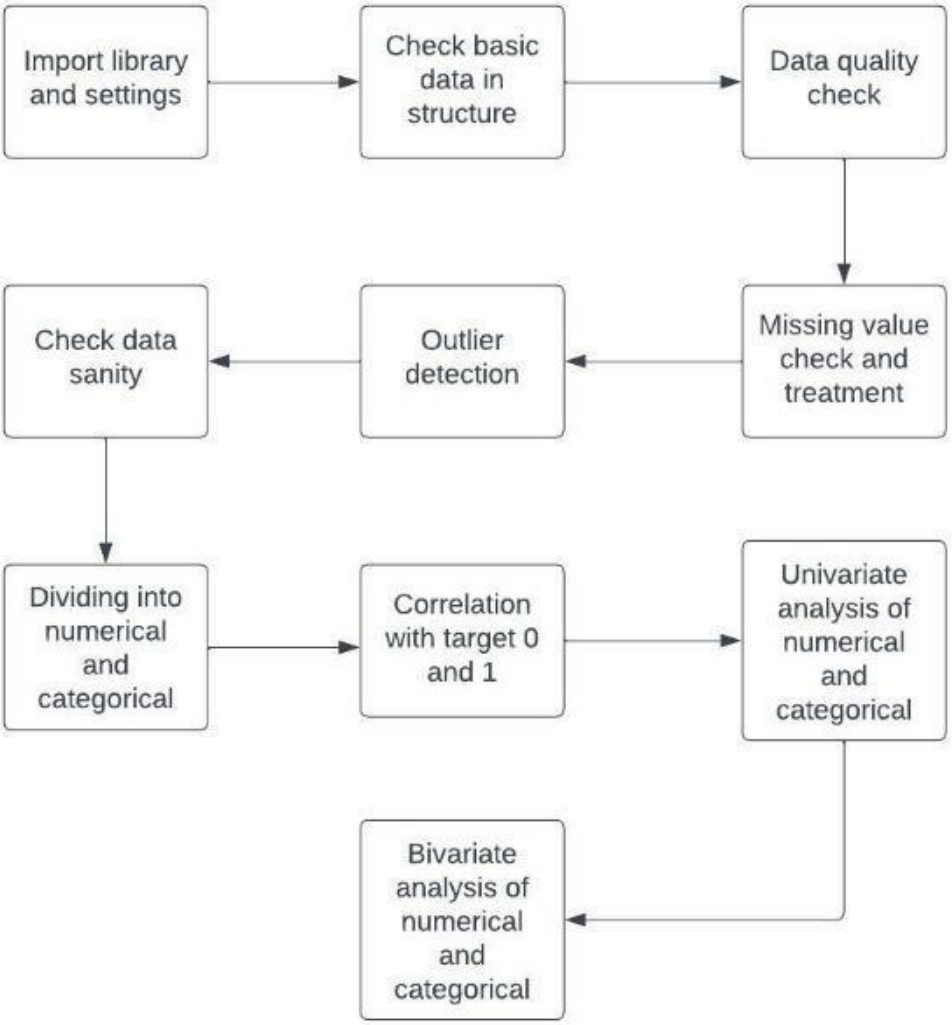


Figure 2- Process flow diagram for applicartion_dataset

4.2.2 For previous_application:

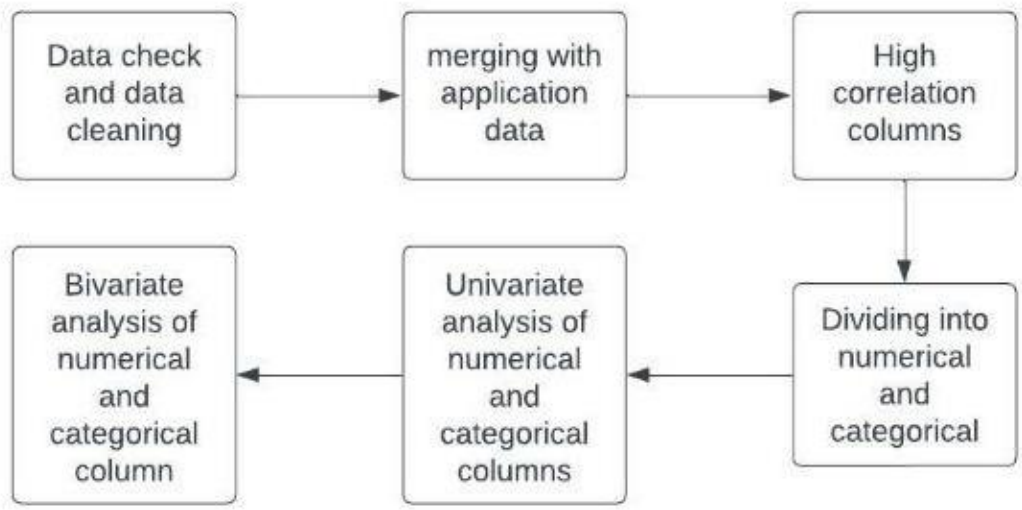


Figure 3: process flow diagram for previous_application

Chapter 05 – System Implementation, Result, and Discussion

5.1 Implementation and code

We can divide our analysis into the following parts:

1) Importing necessary libraries and checking basic data structure of the data. Here we imported numpy, pandas, matplotlib, seaborn libraries which are necessary for analysis as discussed in chapter 2.1 Environment. By checking basic data structure we get a knowledge of size, data type, columns, and non-null values in the dataset. (appendix 1: fig 21, 22, 23)
2) Data Quality check: we started with missing value treatment, we considered 45% missing values as threshold, as we are assuming columns having 45% missing value won't provide much value to our analysis. (appendix 1: fig 24). To treat missing values we plotted boxplot so that we can choose between mean or median(appendix 1: fig 25). As outliers were present we imputed null values by median. Median is less affected by outliers and we will check if there are any quasi-constant features in our dataset(Quasi-constant features are those attributes which have a similar value throughout and hence will not be able to help us in the predictive analysis) (appendix 1: fig 26)

   Then we have divided the columns in numerical and categorical data manually, we concatenated object column and numerical categorical column. After this, we found if the columns are discrete or continuous.(appendix 1: fig 27)

3) Data Sanity: We checked the first five columns and noticed that some values in the “days” attribute are negative which is not possible so we used an absolute function to rectify this problem.(appendix 1: fig 28)
4) Outlier presence in numerical columns: We made a custom function to check whether the attributes have outliers or not,(appendix 1: fig 29) after this, we plotted the boxplots of the attributes.(appendix 1: fig 30), For further analysis we took examples of 5 columns to see the effect of outlier and distribution of those variables the attributes were, AMT_INCOME_TOTAL','DAYS_EMPLOYED','OBS_30_CNT_SOCIAL_CIRCLE','DEF_60_CNT_SOCIAL_CIRCLE','AMT_REQ_CREDIT_BUREAU_QRT.In some of these variables, we had rare cases and we solved them by flooring and capping.
5) We found top 10 correlation attributes with respect to TARGET 0 or 1.(appendix 1: fig 31).
6) Relevance of columns in predicting TARGET: Here, we used only those columns that have high correlation with the Target variable. So We can focus more on the features that are more relevant to TARGET. But as Target is also a categorical variable we used pointbiserialr scipy.stats function (Point Biserial Correlation) The point biserial correlation is used to measure the relationship between a binary variable, x, and a continuous variable, y,(appendix 1: fig 32)

7) Converted days columns to years: For better understanding and easier calculations, We converted “days” columns to “years” by a custom function.(appendix 1: fig 33).
8) Grouped Small categories in one collective category: We had many small categories then those categories won't be much helpful in analysis and hence we made an “other” attribute in which we grouped them. We did this for eight attributes.(appendix 1: fig 34).
9) We separated categorical and numerical variables, then checked the TARGET variable by changing it to percentage and plotted a countplot to check whether there was an imbalance between TARGET values(appendix 1: fig 35).
10) Univariate analysis for numerical variables: We performed univariate analysis for the numerical columns, they were EXT_SOURCE_2','YEARS_BIRTH', 'YEARS_LAST_PHONE_CHANGE', 'YEARS_ID_PUBLISH', 'YEARS_EMPLOYED' , 'YEARS_REGISTRATION', 'EXT_SOURCE_3. We used the distplot function and plotted box plot of all these variables.(appendix 1: fig 36)
11) Univariate analysis for categorical variables: We plotted bar graphs of nineteen attributes with respect to TARGET value 0 or 1.(appendix 1: fig 37)
12) We performed Bivariate and Multivariate analysis on these attributes: Numerical Columns : ['EXT_SOURCE_3', 'EXT_SOURCE_2', 'YEARS_BIRTH']Categorical Columns : ['ORGANIZATION_TYPE', 'NAME_FAMILY_STATUS','NAME_EDUCATION_TYPE','CODE_GENDER']

    For numerical columns, we used heatmap to check the correlation between themselves, and for categorical columns, we used pivot table to check relationship between all numerical columns (among themselves too) and target.

13) We worked on “previos_application” dataset by first checking data quality and treated missing values in our dataset, a 49% value threshold was set to drop the attribute which contained missing values.(appendix 1: fig 38)
14) We merged the two datasets by 'SK_ID_CURR','TARGET’ because TARGET variable was available in “application_dataset”(appendix 1: fig 39)
15) Univariate Analysis of Categorical Columns in “previous_application” dataset and plotted the bar graphs for categorical columns.(appendix 1: fig 40) and then we again plotted bar graphs for numerical columns in “previous_application” dataset.(appendix 1: fig 41) then we performed Bi-variate analysis.

5.2 Results:

1. EXT_SOURCE_2

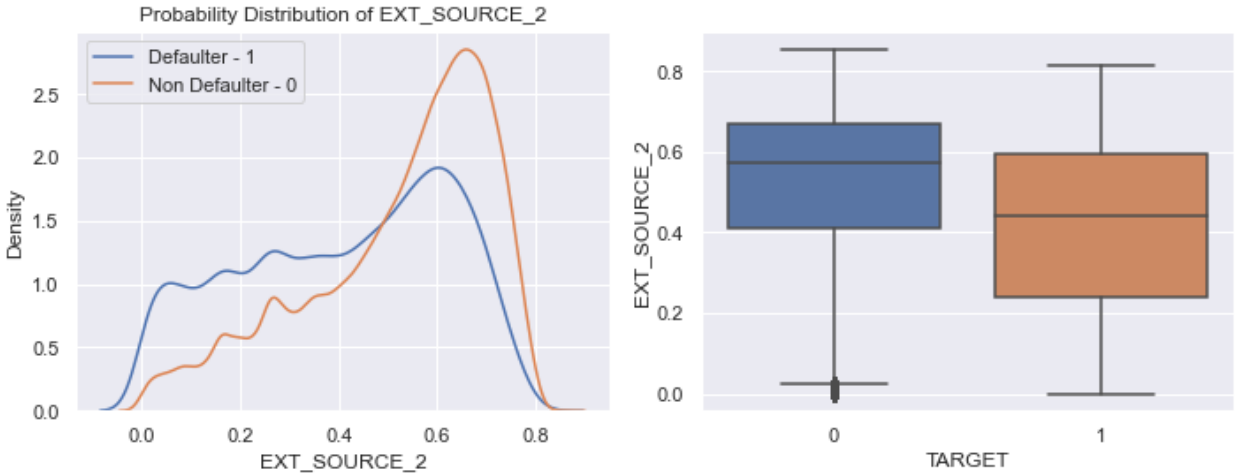


Figure 4: difference between defaulter and non defaulter wrt EXT_SOURCE_2

A. With EXT_SOURCE_2 we can easily distinguish Defaulter and Non Defaulter.
B. Non Defaulters have higher EXT_SOURCE_2 and the spread of values are also small in comparison to Defaulter. But Non Defaulters have lower values until 0.5 (approx). We can see peak for both Targets at 0.6(approx) but Non-Defaulter has at higher side

2. YEARS_BIRTH

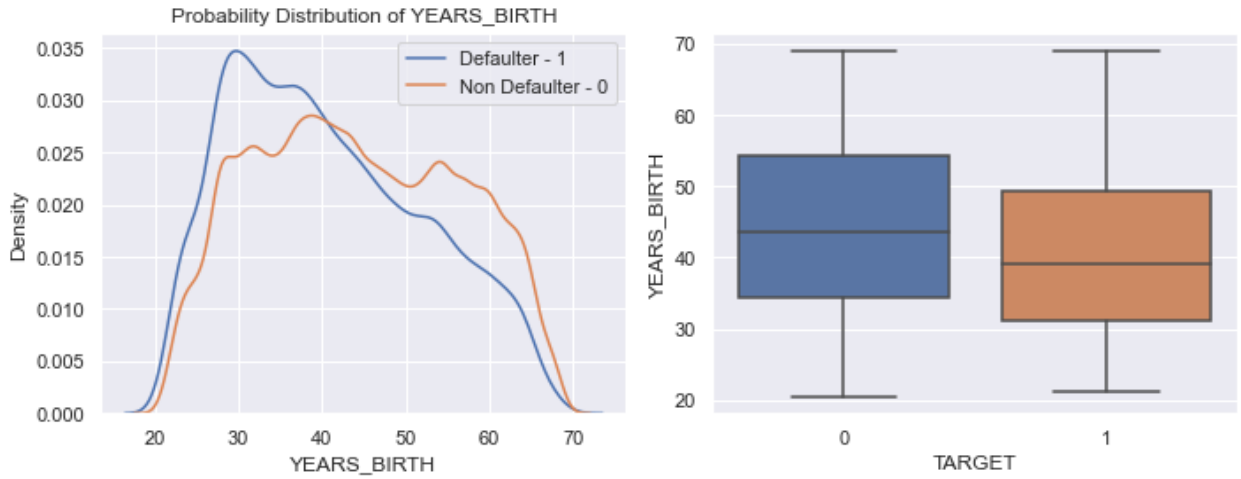


Figure 5: difference between defaulter and non defaulter wrt YEARS_BIRTH

A. People whose age is more than 40 years of age are less likely to default compared to people having age less than 40 years.
B. Non Defaulter seems to be in a higher age range also.

3. EXT_SOURCE_3

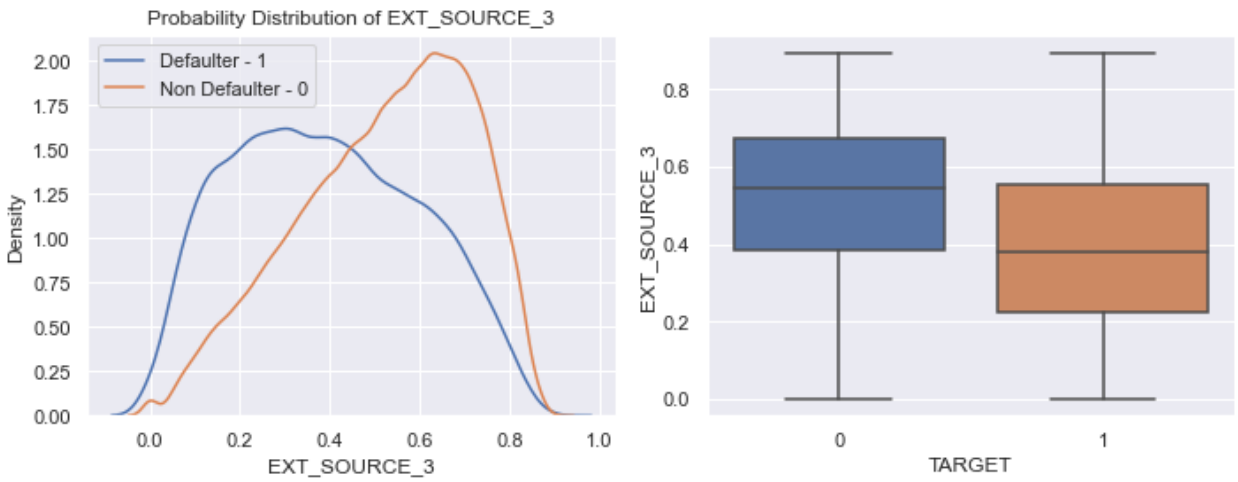


Figure 6: difference between defaulter and non defaulter wrt EXT_SOURCE_3

A. With this feature we can easily differentiate between Defaulter and Non-Defaulter
B. Before 0.45 Defaulter have higher number Applicants,after 0.45 Non-Defaulter have more applicants

4. FLAG_EMP_PHONE

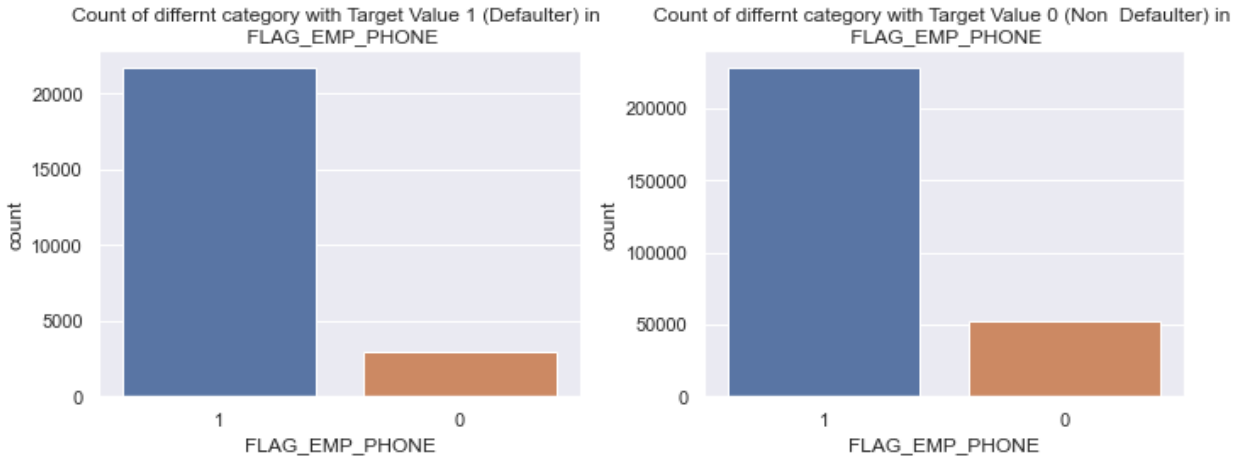


Figure 7: difference between defaulter and non defaulter wrt FLAG_EMP_PHONE

A. When we compared both categories in FLAG_EMP_PHONE category 0 where Applicant has employee phone, has higher number in Non Defaulter than Defaulter.
B. If Applicant does not have an Employee phone there are a little bit higher chances of Not Defaulting.

5. CODE_GENDER

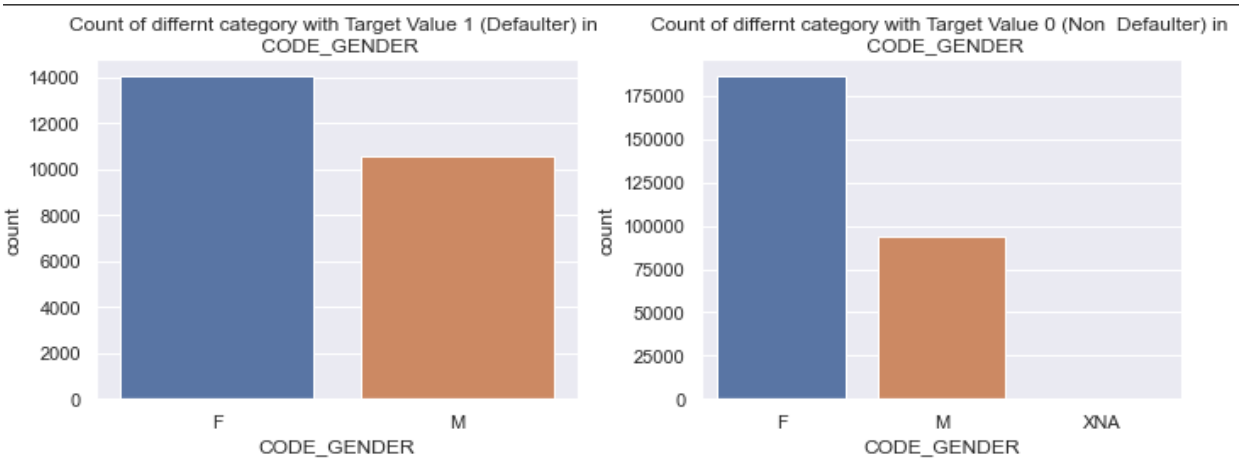


Figure 8: difference between defaulter and non defaulter wrt CODE_GENDER

A. Male are more likely to default on a loan.
B. XNA is only chosen by Non Defaulter Applicants.

6. NAME_EDUCATION_TYPE

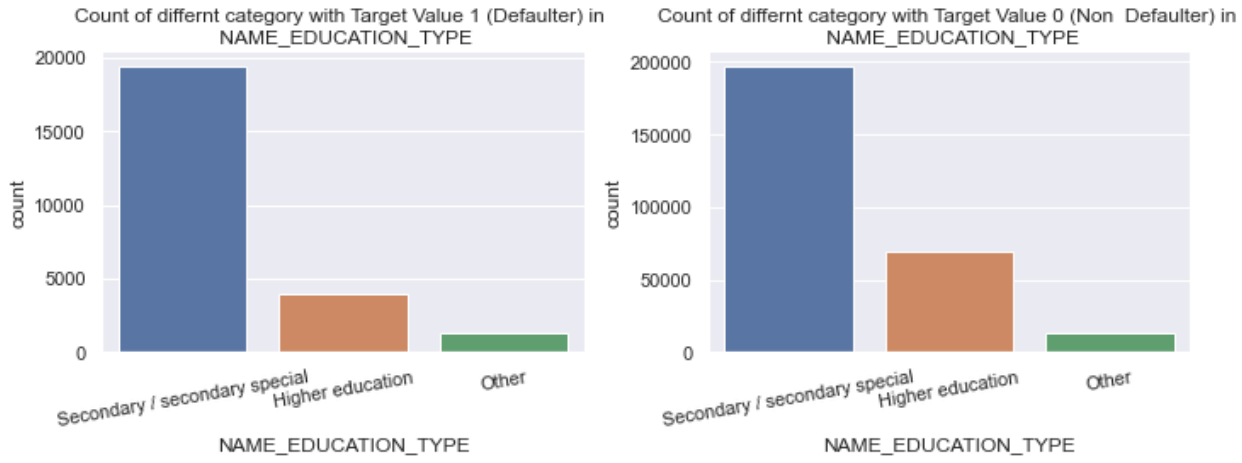


Figure 9: difference between defaulter and non defaulter wrt FLAG_EDUCATION_TYPE

A. Most applicant has only Secondary Education
B. People who have higher education are less likely to default.

7.NAME_FAMILY_STATUS

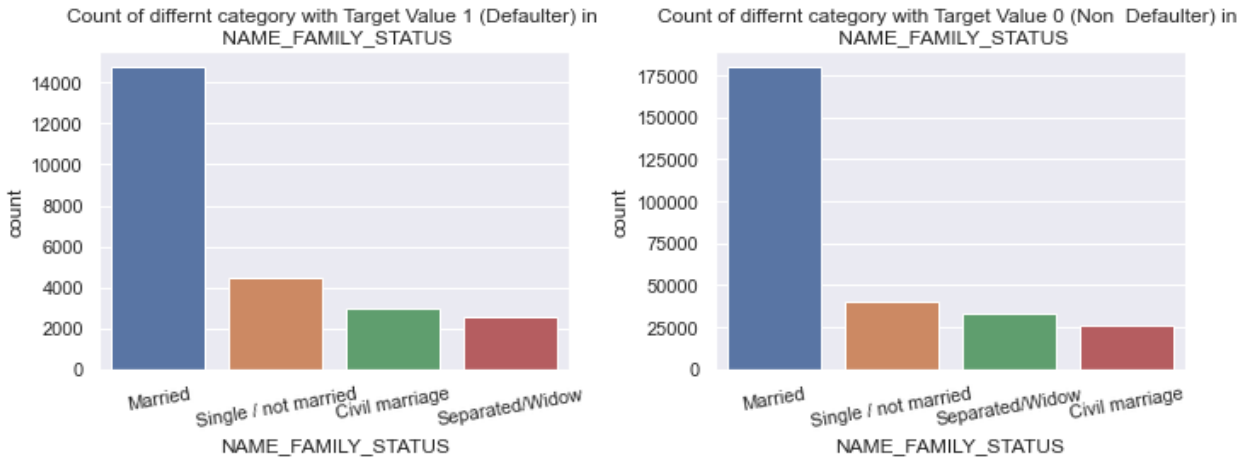


Figure 10: difference between defaulter and non defaulter wrt NAME_FAMILY_STATUS

A. In Non Defaulter and Defaulter, there are more applicants in the Separated /Widow section in Non Defaulter while in Defaulter in Civil Marriage.
B. Most Applicants are in the Married Category.

8. OCCUPATION_TYPE

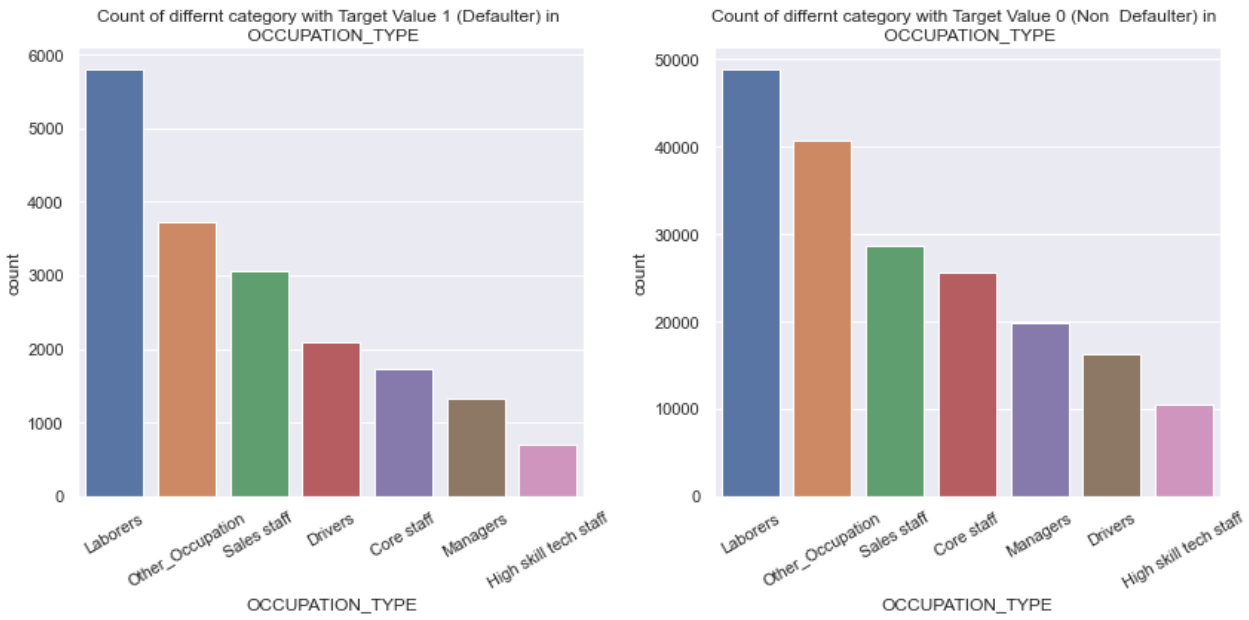


Figure 11: difference between defaulter and non defaulter wrt OCCUPATION_TYPE

A. In both Target values we have a similar trend for all occupations except Driver.
B. Drivers are more likely to Default on a loan.

9. ORGANIZATION_TYPE

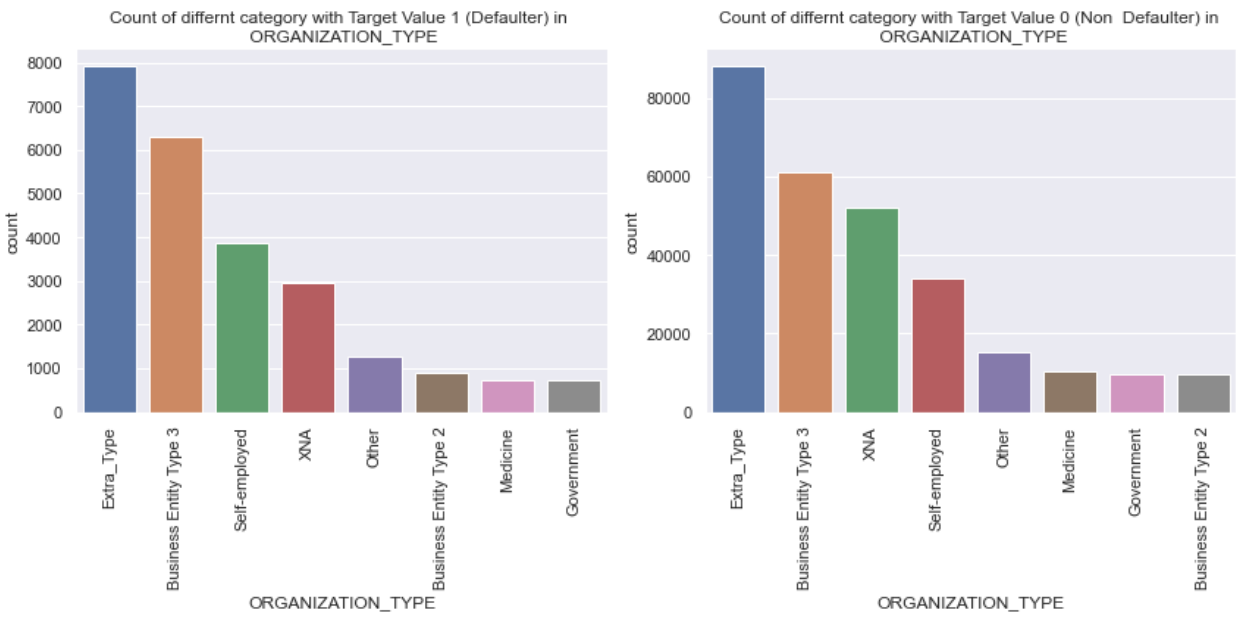


Figure 12: difference between defaulter and non defaulter wrt ORGANIZATION_TYPE

All Organization types of Application have the same trend in Defaulter and Non Defaulter. Except people who have XNA are less likely to Default than those who have filled Self Employed.

10. Correlation among Numerical Features

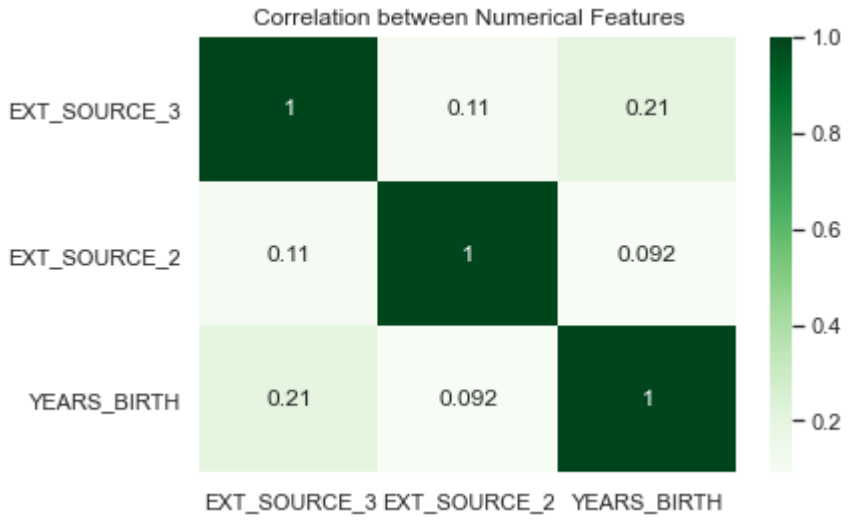


Figure 13: Heatmap of numerical features

A. As all three have very low correlation among themselves and also positive correlation.
B. For a feature to be a good predictor of target it should have high correlation to Target and low correlation among themselves.

11. Relationship in between all numerical columns (among themselves) and target

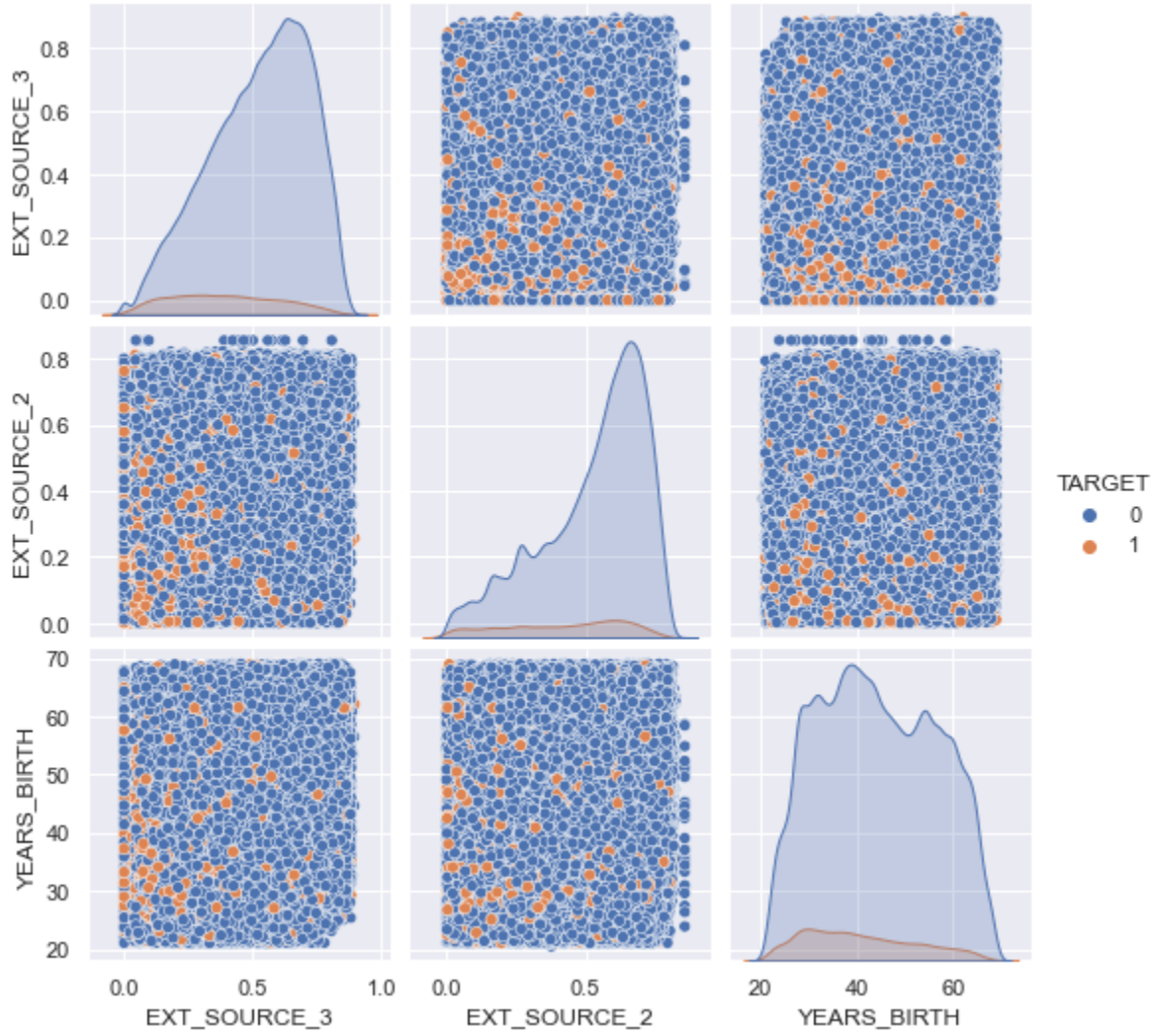


Figure 14: show relationship in between all numerical columns

A. From this graph we can see that defaulter(generally) have low values of EXT_SOURCE2 and EXT_SOURCE3 Score and have age in between 20 to 40.

12. ORGANIZATION_TYPE & CODE_GENDER

ORGANIZATION_TYPE	Business Entity Type 2	Business Entity Type 3	Extra_Type	Government	Medicine	Other	Self-employed	XNA
CODE_GENDER								
F	0.070435	0.080121	0.072836	0.063098	0.065706	0.068753	0.089543	0.049706
M	0.102804	0.110507	0.097726	0.088205	0.068458	0.091251	0.127158	0.073789
XNA	NaN	NaN	0.000000	NaN	0.000000	NaN	NaN	NaN

Figure 15: pivot table of ORGANIZATION_TYPE and CODE_GENDER

With this we can see as male have a higher chance of Default in that for ORGANIZATION_TYPE as self_employed it is much higher.

13. NAME_FAMILY_STATUS & NAME_EDUCATION_TYPE

NAME_EDUCATION_TYPE	Higher education	Other	Secondary / secondary special
NAME_FAMILY_STATUS			
Civil marriage	0.066778	0.104353	0.108373
Married	0.049875	0.084670	0.084315
Separated/Widow	0.054273	0.083639	0.075543
Single / not married	0.062611	0.101224	0.113829

Figure 16: pivot table of NAME_FAMILY_STATUS & NAME_EDUCATION_TYPE

With this we can see as male have a higher chance of Default in that for ORGANIZATION_TYPE as self_employed it is much higher.

14. CNT_PAYMENT (from previous_application)

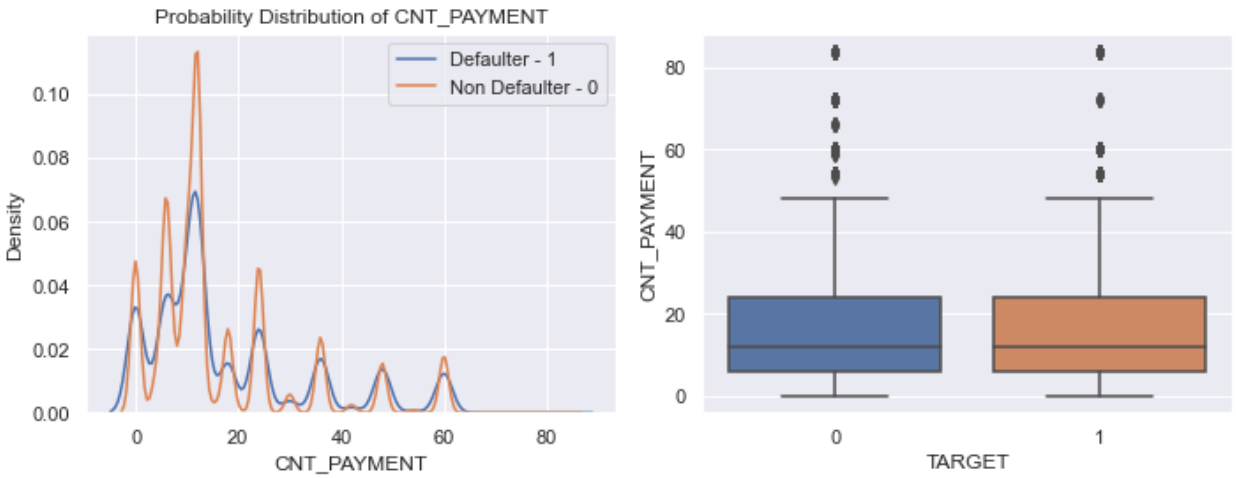


Figure 17: difference between defaulter and non defaulter wrt CNT_PAYMENT

For cnt_payment we can see Defaulter and Non Defaulter seem to peak at the same values but for Non Defaulter it is high.

15. NAME_CONTRACT_STATUS

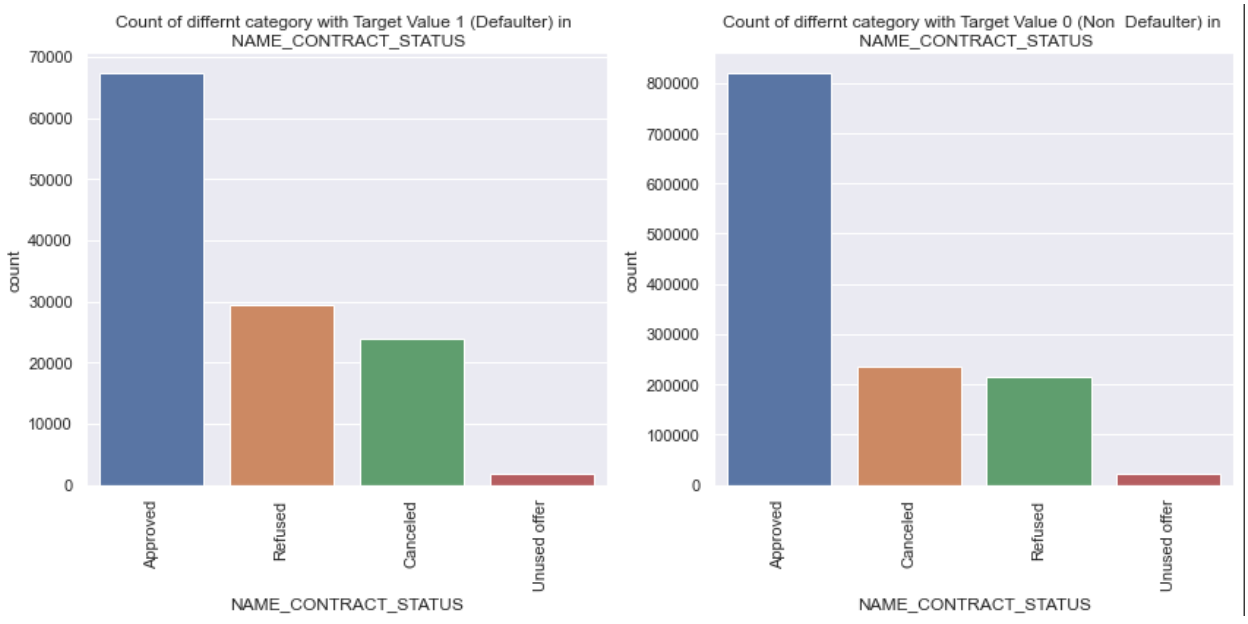


Figure 18: difference between defaulter and non defaulter wrt NAME_CONTRACT_STATUS

For Contact Status, the defaulter applicant seems to have more Refused and canceled than non-defaulter.

16. NAME_CONTRACT_TYPE

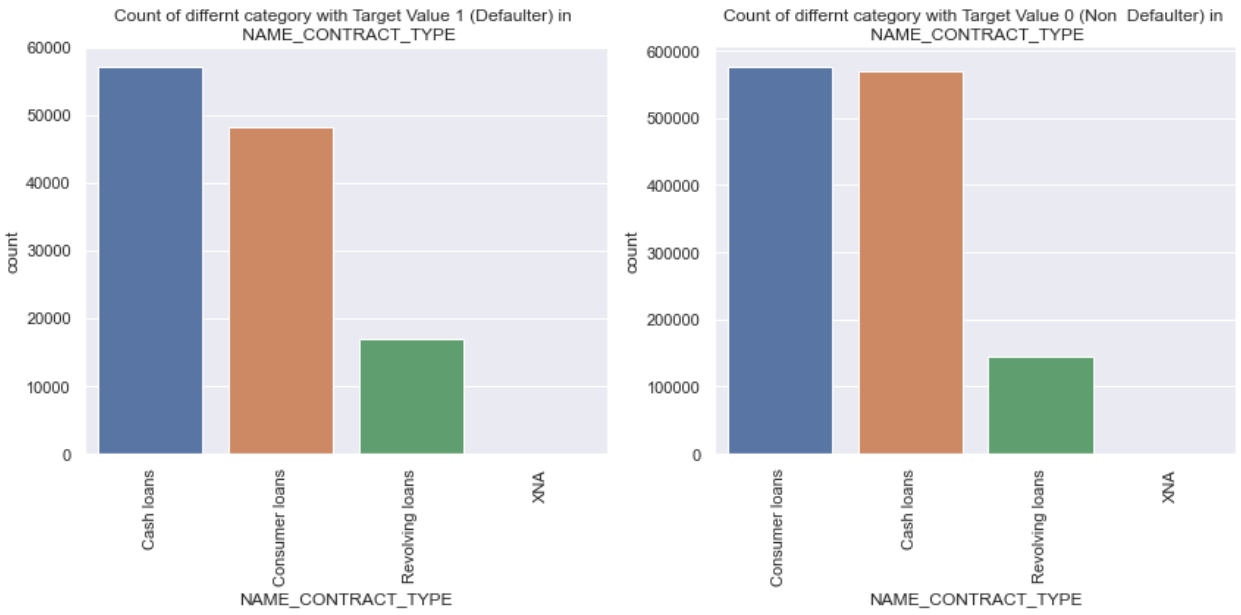


Figure 19: difference between defaulter and non defaulter wrt NAME_CONTRACT_TYPE

Defaulter seems to have more consumer loans than Non Defaulter, while for cash loans even the The difference is little; it is the opposite of consumer loans.

17. NAME_CONTRACT_TYPE & NAME_CONTRACT_STATUS

NAME_CONTRACT_STATUS	Approved	Canceled	Refused	Unused offer
NAME_CONTRACT_TYPE				
Cash loans	0.075516	0.088401	0.125810	0.092593
Consumer loans	0.073853	0.128668	0.101350	0.082337
Revolving loans	0.090343	0.109254	0.129050	0.000000
XNA	NaN	0.197183	0.241379	NaN

Figure 20: pivot table of NAME_CONTRACT_TYPE & NAME_CONTRACT_STATUS

Here we can see Applicants that has refused from loan and of XNA Contract type are more likely to default on their loan

Chapter 06 – Findings and Conclusion

6.1 Findings:

This report covers all the necessary attributes that are needed to check whether the person will default or not in the near future.

IN APPLICATION DATASET

- In Univariate analysis we used:
  Numerical columns: 'EXT_SOURCE_2','YEARS_BIRTH',
  'YEARS_LAST_PHONE_CHANGE',
  'YEARS_ID_PUBLISH', 'YEARS_EMPLOYED', 'YEARS_REGISTRATION',
  'EXT_SOURCE_3'
  Categorical Columns : 'REGION_RATING_CLIENT_W_CITY',
  'REGION_RATING_CLIENT','REG_CITY_NOT_WORK_CITY','FLAG_EMP_PHONE','REG_CITY_NOT_LIVE_CITY‘,'FLAG_DOCUMENT_3',
  'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR',
  'FLAG_OWN_REALTY', 'NAME_TYPE_SUITE‘ ,'NAME_INCOME_TYPE',
  'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS',
  'NAME_HOUSING_TYPE','OCCUPATION_TYPE','WEEKDAY_APPR_PROCESS_START', 'ORGANIZATION_TYPE’
- With EXT_SOURCE_2 we can easily distinguish Defaulter and Non Defaulter. Non Defaulters have higher EXT_SOURCE_2 and the spread of values are also small in comparison to Defaulter. But Non Defaulter have lower values until 0.5 (approx). We can see peaks for both Targets at 0.6(approx) but Non-Defaulter has at higher sides.
- YEARS_BIRTH; People more than 40 years of age are less likely to default compared to people less than 40 years. Non Defaulter seems to be in a higher age range also.
- EXT_SOURCE_3: With this feature we can easily differentiate between Defaulter and Non-Defaulter Before 0.45 Defaulter have higher number Applicants, after 0.45 Non-Defaulter have more applicants
- FLAG_EMP_PHONE: When we compare both categories in FLAG_EMP_PHONE category 0 where Applicant has employee phone, has a higher number in Non-Defaulter than Defaulter. If Applicant does not have an employee phone there are a few higher chances of Not Defaulting.
- CODE_GENDER: Male are more likely to default on a loan. XNA is only chosen by Non Defaulter Applicants.
- NAME_EDUCATION_TYPE: Most applicants have only Secondary Education People who have higher education are less likely to default.
- NAME_FAMILY_STATUS: In Non-Defaulter and Defaulter There is check in order as there are more applicants in Separated /Widow section in Non-Defaulter while in Defaulter in Civil Marriage. Most Applicants are in the married Category.

- OCCUPATION_TYPE: In both Target values we have a similar trend for all occupations except Driver. Drivers are more likely to Default on a loan.
- ORGANIZATION_TYPE: All Organization type of Application have the same trend in Defaulter and Non-Defaulter. Except people who have XNA are less likely to Default than those who have filled Self Employed.
- Correlation among Numerical Features as all three have very low correlation among themselves and also positive correlation. For a feature to be a good predictor of target it should have high correlation to Target and low correlation among themselves.
- ORGANIZATION_TYPE & CODE_GENDER: With bivariate analysis we can see as males have a higher chance of Default in that for ORGANIZATION_TYPE as self_employed it is much higher.

IN PREVIOUS APPLICATION DATASET

- Defaulter seems to have more consumer loans than Non Defaulter, while for cash loans even the difference is little it is opposite of consumer loans
- For week day application process start it is same for both Defaulter or Non Defaulter
- For Name cash Purpose also there isn't any visible difference between both targets.
- For Contact Status, for defaulter applicant seems to have more Refused and canceled than non-defaulter
- For payment type, all category are similar for both Defaulter and Non-Defaulter
- Reject Reason code also seems to have a similar count % for both defaulter and non defaulter.
- Client type also seems to have similar count % for both defaulter and non defaulter.
- Goods_Category, Name_Type, Name_Portfolio, Product_Type, Channel_Type, Seller_industry, Name_yield have similar count % for both defaulter and non-defaulter. In my opinion they should not give any good indication of Target Variable
- Hour Approx process start is too random, and does not provide any clear indication about target variable.
- Day's decision has a similar distribution for both categories.
- For cnt_payment we can see Defaulter and Non-Defaulter seem to peak at the same values but for non-Defaulter it is high.

6.2 Conclusion:

- Application External Source score 2 & 3 matter a lot in deciding between Defaulter and Non-Defaulter. They also have the highest correlation (Linear relation with the target variable). Banks should focus more on these scores of client while providing loan.
- Clients having education Secondary or Secondary Special are more likey to apply for the loan. Clients having education Secondary or Secondary Special have higher risk to default. Other education types have minimal risk.
- Years of Birth (Person's Age) can also be one of the decisive factors for defaulting on a loan. Clients with age more than 40 years are less likely to default on loan payments.
- Female clients with an Academic degree and high-income type have a higher risk of default. Male clients with Secondary/Secondary Special Education having all types of salaries have a higher risk of default.
- From previous_application we can find cnt payment, Type of contract, and Contract's status on their previous application as deciding factors defaulting on their current application. Banks should focus more on these factors for providing loans.

7. References-

1. "Evaluation of the borrower's creditworthiness as an important condition for enhancing the effectiveness of lending operations" A. Chaplinska, SHS Web of Conferences 2, 00009 (2012).
2. "Exploratory Data Analysis", Matthieu Komorowski, Dominic C. Marshall, Justin D. Salciccioliand Yves Crutain.
3. "Financial Institution Type and Firm-Related Attributes as Determinants of Loan Amounts",Dr. Edmund Bwire
4. https://github.com/
5. https://www.youtube.com/watch?v=-o3AxdVcUtQ&ab_channel=edureka%21
6. https://www.youtube.com/watch?v=xhB-dmKmzRk&ab_channel=KrishNaik
7. https://www.upgrad.com/
8. https://www.geeksforgeeks.org
9. https://docs.python.org/3/library/
10. https://in.coursera.org/
11. https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15
12. https://www.simplilearn.com/tutorials/data-analytics-tutorial/exploratory-data-analysis
13. https://www.w3schools.com/python/matplotlib_pyplot.asp
14. https://www.w3schools.com/python/pandas/default.asp
15. https://www.kaggle.com/learn/pandas

8. Appendix:

8.1 appendix 1: figures and tables

```
[ ]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     sns.set()

[ ]  # for ignoring unnecessary warning
     import warnings
     warnings.filterwarnings("ignore")

[ ]  # for seeing all columns, for large dataset with large number of columns
     pd.options.display.max_columns = 200

[ ]  # loading application data
     app_data = pd.read_csv('application_data.csv')
```

Figure 21- Importing libraries loading data

Check basic structure of data

```
[ ]  # check first 3 rows of data to see what does data look like
     app_data.head(3)
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	NAME_TYPE_SUITE	NAME_
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5	24700.5	351000.0	Unaccompanied	
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	35698.5	1129500.0	Family	
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0	6750.0	135000.0	Unaccompanied	

```
# Describe numerical data columns
app_data.describe()
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVE	DAYS_BIRTH	DAYS_EMPLOYED	DAYS_REGIST
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307499.000000	3.072330e+05	307511.000000	307511.000000	307511.000000	307511.0
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27108.573909	5.383962e+05	0.020868	-16036.995067	63815.045904	-4986.1
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14493.737315	3.694465e+05	0.013831	4363.988632	141275.766519	3522.8
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1615.500000	4.050000e+04	0.000290	-25229.000000	-17912.000000	-24672.0
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16524.000000	2.385000e+05	0.010006	-19682.000000	-2760.000000	-7479.5
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24903.000000	4.500000e+05	0.018850	-15750.000000	-1213.000000	-4504.0
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.000000	6.795000e+05	0.028663	-12413.000000	-289.000000	-2010.0

Figure 22- Initial columns of dataset and description

```
[ ] # Shape data frame
    app_data.shape

    (307511, 122)

    # data type, columns and non-null values in dataset
    app_data.info(verbose = True, show_counts=True)

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 307511 entries, 0 to 307510
    Data columns (total 122 columns):
     #    Column                       Non-Null Count   Dtype
    ---   ------                       --------------   -----
     0    SK_ID_CURR                   307511 non-null  int64
     1    TARGET                       307511 non-null  int64
     2    NAME_CONTRACT_TYPE           307511 non-null  object
     3    CODE_GENDER                  307511 non-null  object
     4    FLAG_OWN_CAR                 307511 non-null  object
     5    FLAG_OWN_REALTY              307511 non-null  object
     6    CNT_CHILDREN                 307511 non-null  int64
     7    AMT_INCOME_TOTAL             307511 non-null  float64
     8    AMT_CREDIT                   307511 non-null  float64
     9    AMT_ANNUITY                  307499 non-null  float64
     10   AMT_GOODS_PRICE              307233 non-null  float64
     11   NAME_TYPE_SUITE              306219 non-null  object
     12   NAME_INCOME_TYPE             307511 non-null  object
     13   NAME_EDUCATION_TYPE          307511 non-null  object
     14   NAME_FAMILY_STATUS           307511 non-null  object
     15   NAME_HOUSING_TYPE            307511 non-null  object
     16   REGION_POPULATION_RELATIVE   307511 non-null  float64
     17   DAYS_BIRTH                   307511 non-null  int64
     18   DAYS_EMPLOYED                307511 non-null  int64
     19   DAYS_REGISTRATION            307511 non-null  float64
     20   DAYS_ID_PUBLISH              307511 non-null  int64
     21   OWN_CAR_AGE                  104582 non-null  float64
```

Figure 23- Shape of data and data description

We will take 45% missing values as threshold, as we are assuming columns having 45% missing value won't provide much value to our analysis

```
[ ] # Storing columns that has missing more than 45 %
    drop_cols_45 = list(mis_col_per[mis_col_per['Missing_per']>45]['Column_Names'])

[ ] # Dropping all the columns that has more than 45% missing values
    app_data.drop(labels=drop_cols_45, axis = 1, inplace=True)

[ ] # finding index of columns with less than 1% missing values
    def find_mis_idx_1per(data):
        isna_ser = data.isnull().mean()*100
        col_1per = isna_ser[(isna_ser<1) & (isna_ser>0)].index
        idx_list = []
        for i in col_1per:
            idx_sub_list = list(data[data.loc[:,i].isnull()].index)
            idx_list += idx_sub_list
        return set(idx_list)

[ ] # indexes that has less than 1% missing values we can remove those rows
    idx_mis = find_mis_idx_1per(app_data)

[ ] # removing rows with missing values from idx_mis list
    app_data.drop(labels=idx_mis, axis = 0, inplace=True)
```

Figure 24- Checking missing values with a threshold of more than 45%

```
# plotting distribution of all the variables from imp_cols_less15, so we can choose between mean or median for imputation.
for i in imp_cols_less15:
    sns.boxplot(y = i, data = app_data)
    plt.show()
```

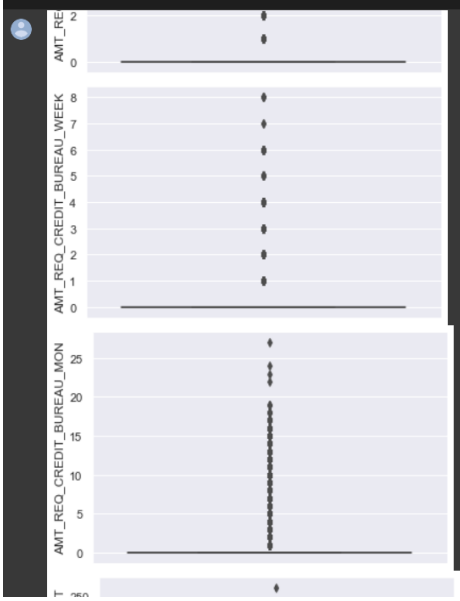


Figure 25: Outlier detection

As we can for all the columns outlier is present so we should impute null values by median. As median is less affected by outliers

```
[ ] # Imputing columns by median value
    for i in imp_cols_less15:
        app_data[i].fillna(value = app_data[i].median(), axis = 0, inplace = True)
```

```
[ ] # verifying if values had imputed or not
    app_data[imp_cols_less15].isna().sum()
```

```
AMT_REQ_CREDIT_BUREAU_HOUR    0
AMT_REQ_CREDIT_BUREAU_DAY     0
AMT_REQ_CREDIT_BUREAU_WEEK    0
AMT_REQ_CREDIT_BUREAU_MON     0
AMT_REQ_CREDIT_BUREAU_QRT     0
AMT_REQ_CREDIT_BUREAU_YEAR    0
dtype: int64
```

Check Constant or Quasi Constant Features and remove those Feature

Constant : 100% of valuse in a column are single value Quasi Constant : Almost all the values in a column are a single value

```
[ ] # We will be removing quasi constant columns threshold will be taken as 98%
    quasi_const_cols = []
    for i in app_data.columns:
        if (app_data[i].value_counts(normalize=True).iloc[0]>0.98):
            quasi_const_cols.append(i)
```

```
[ ] app_data.drop(labels = quasi_const_cols, axis = 1, inplace = True)
```

Figure 26: imputing null values with media and removing quasi-constant features

Dividing columns in Numerical and Categorical

```
[ ]  # Dividing Numerical and Object dtype columns
     num_col = list(app_data.select_dtypes(exclude='object').columns)
     obj_col = list(app_data.select_dtypes(include='object').columns)
     id_col = ['SK_ID_CURR']
```

```
[ ]  # I had to divide these columns manually will add these columns to obj columns to creating categorical column list

     num_cat_col = ['FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_PHONE', 'FLAG_EMAIL', 'REGION_RATING_CLIENT',
                    'REGION_RATING_CLIENT_W_CITY', 'REG_REGION_NOT_WORK_REGION', 'LIVE_REGION_NOT_WORK_REGION',
                   'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY',  'FLAG_DOCUMENT_3','TARGET']
```

```
[ ]  # Final Categorical columns
     cat_col = obj_col + num_cat_col
```

```
[ ]  # Final numerical columns that are either continuous or discrete
     numerical_col = []
     for i in num_col:
         if (i not in num_cat_col) and (i not in id_col):
             numerical_col.append(i)
```

```
[ ]  len(cat_col), len(numerical_col)

     (25, 25)
```

Figure 27: Dividing in numerical and categorical columns

```
[ ]  # check for numrical columns
     app_data[numerical_col].head()
```

	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVE	DAYS_BIRTH	DAYS_EMPLOYED	DAYS_REGISTRATION	DAYS_ID_PUBLISH	CNT_FA
0	0	202500.0	406597.5	24700.5	351000.0	0.018801	-9461	-637	-3648.0	-2120	
1	0	270000.0	1293502.5	35698.5	1129500.0	0.003541	-16765	-1188	-1186.0	-291	
2	0	67500.0	135000.0	6750.0	135000.0	0.010032	-19046	-225	-4260.0	-2531	
3	0	135000.0	312682.5	29686.5	297000.0	0.008019	-19005	-3039	-9833.0	-2437	
4	0	121500.0	513000.0	21865.5	513000.0	0.028663	-19932	-3038	-4311.0	-3458	

As we can see all the columns that has days in them should be positive but they are negative. we need to correct that

```
[ ]  # find the columns with days in them. because only they has positive negative issue
     neg_cols = []
     for i in numerical_col:
         if "DAYS" in i:
             neg_cols.append(i)
```

```
[ ]  # apply absolute function on neg_cols
     app_data[neg_cols] = app_data[neg_cols].apply(abs)
```

```
     # check if results are reflecting
     app_data[neg_cols].head()
```

Figure 28-checking data sanity and positive negative values

Now we need to check for outlier presence in numerical columns

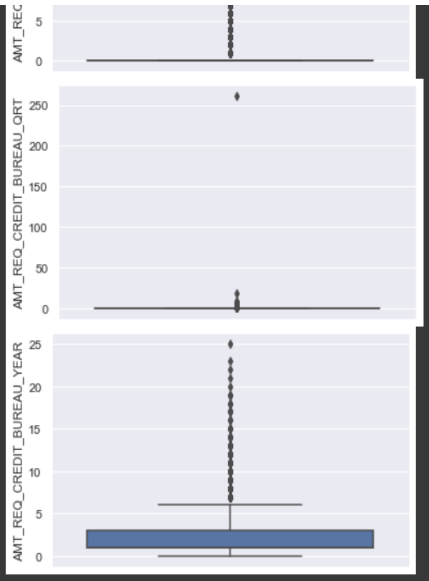
```
# create a function to get a list of function that has outliers
def find_outCol(data):
    out_list = []
    for col in data.columns:
        IQR = data[col].describe()['75%'] - data[col].describe()['25%']
        lower_bound = data[col].describe()['25%'] - 1.5*IQR
        upper_bound = data[col].describe()['75%'] + 1.5*IQR

        for val in data[col]:
            if val < lower_bound :
                out_list.append(col)
                break
            elif val > upper_bound :
                out_list.append(col)
                break
            else :
                continue
    return out_list
```

```
[ ] outlier_col = find_outCol(app_data[numerical_col])
```

```
[ ] for i in outlier_col:
        sns.boxplot(y = i, data = app_data)
        plt.show()
```

Figure 29-Function to get outliers



From both fucntion and boxplot. We can find outliers in numerical columns, But still in those few columns are there that has continuous values even in outliers.For example AMT_CREDIT,AMT_ANNUITY, AMT_GOODS_PRICE, DAYS_REGISTRATION,DAYS_LAST_PHONE_CHANGE

Figure 30-Outliers box plot

```
[ ]  app_target1 = app_data[app_data['TARGET']==1]
     app_target0 = app_data[app_data['TARGET']==0]
```

```
[ ]  # we are finding correlation of datafrme where target is 1
     data1_corr = abs(app_target1.corr()).unstack()
```

```
[ ]  # TOp 10 correlation in dataset with target = 1
     data1_corr[data1_corr != 1].sort_values(ascending = False)[0:20:2]
```

```
FLAG_EMP_PHONE               DAYS_EMPLOYED                0.999703
OBS_30_CNT_SOCIAL_CIRCLE     OBS_60_CNT_SOCIAL_CIRCLE     0.998286
AMT_CREDIT                   AMT_GOODS_PRICE              0.983065
REGION_RATING_CLIENT_W_CITY  REGION_RATING_CLIENT         0.956477
CNT_CHILDREN                 CNT_FAM_MEMBERS              0.885556
DEF_60_CNT_SOCIAL_CIRCLE     DEF_30_CNT_SOCIAL_CIRCLE     0.869761
LIVE_REGION_NOT_WORK_REGION  REG_REGION_NOT_WORK_REGION   0.847260
REG_CITY_NOT_WORK_CITY       LIVE_CITY_NOT_WORK_CITY      0.778110
AMT_ANNUITY                  AMT_GOODS_PRICE              0.752206
AMT_CREDIT                   AMT_ANNUITY                  0.751400
dtype: float64
```

Now move to Target = 0

```
[ ]  # we are finding correlation of datafrme where target is 0
     data0_corr = abs(app_target0.corr()).unstack()
```

```
[ ]  # Top 10 correlation in dataset with target = 0
     data0_corr[data0_corr != 1].sort_values(ascending = False)[0:20:2]
```

```
DAYS_EMPLOYED                FLAG_EMP_PHONE               0.999755
OBS_60_CNT_SOCIAL_CIRCLE     OBS_30_CNT_SOCIAL_CIRCLE     0.998513
AMT_CREDIT                   AMT_GOODS_PRICE              0.987260
REGION_RATING_CLIENT         REGION_RATING_CLIENT_W_CITY  0.949905
CNT_FAM_MEMBERS              CNT_CHILDREN                 0.878681
```

Figure 31- dividing TARGET 0 and 1 and finding highest correlation values

Relevence of column in predicting TARGET

Here I want to use only those column that have high correlation with Target variable. So We can more focus on the feature are more relevant to TARGET. But as Target is also a categorical variable we should use pointbiserialr scipy.stats function (Point Biserial Correlation)

```
[ ] # pointbiserialr does not work with columns with null values we will keep this column
    pcorr_col = [i for i in num_col if i not in 'EXT_SOURCE_3']
```

```
# calculating correlation of all the columns with TARGET variable.
from scipy.stats import pointbiserialr
corr_dic = {}
for i in pcorr_col:
    corr_dic[i] = pointbiserialr(app_data[i],app_data['TARGET'])[0]
pcorr = abs(pd.Series(corr_dic)).sort_values(ascending = False)
```

```
[ ] cols_to_keep = list(pcorr[pcorr>0.04].index)
```

```
[ ] # here we would would add EXT_SOURCE_3 back to cols_to_keep
    cols_to_keep = cols_to_keep + ['EXT_SOURCE_3']
```

Lets check if for any continuous column do we need to do binning

```
[ ] app_data[['DAYS_BIRTH','DAYS_LAST_PHONE_CHANGE','DAYS_ID_PUBLISH','DAYS_EMPLOYED','DAYS_REGISTRATION','EXT_SOURCE_2','EXT_SOURCE_3']].describe()
```

	DAYS_BIRTH	DAYS_LAST_PHONE_CHANGE	DAYS_ID_PUBLISH	DAYS_EMPLOYED	DAYS_REGISTRATION	EXT_SOURCE_2	EXT_SOURCE_3
count	304531.000000	304531.000000	304531.000000	304531.000000	304531.000000	3.045310e+05	244280.000000

Figure 32: Relevance of column in predicting TARGET

Convert days columns to years

```
[ ] days_col = ['DAYS_BIRTH','DAYS_LAST_PHONE_CHANGE','DAYS_ID_PUBLISH','DAYS_EMPLOYED','DAYS_REGISTRATION']
    app_data[days_col] = app_data[days_col].apply(lambda x : round(x/365,2))
    app_data.rename({'DAYS_BIRTH':'YEARS_BIRTH'
                    ,'DAYS_LAST_PHONE_CHANGE':'YEARS_LAST_PHONE_CHANGE'
                    ,'DAYS_ID_PUBLISH':'YEARS_ID_PUBLISH'
                    ,'DAYS_EMPLOYED':'YEARS_EMPLOYED'
                    ,'DAYS_REGISTRATION':'YEARS_REGISTRATION'},axis = 1, inplace =True)
```

```
[ ] def get_catInfo(data):
        cat_info = pd.DataFrame(columns = ['column', 'values',  'values_count'])
        temp = pd.DataFrame()

        for c in data.columns:
            temp['column'] = [c]
            temp['values'] = [data[c].unique()]
            temp['values_count'] = int(data[c].nunique())
            cat_info = cat_info.append(temp)
        return cat_info
```

```
app_cat_info = get_catInfo(app_data[obj_col])
app_cat_info
```

	column	values	values_count
0	NAME_CONTRACT_TYPE	[Cash loans, Revolving loans]	2
0	CODE_GENDER	[M, F, XNA]	3
0	FLAG_OWN_CAR	[N, Y]	2
0	FLAG_OWN_REALTY	[Y, N]	2
0	NAME_TYPE_SUITE	[Unaccompanied, Family, Spouse, partner, Child...	7
0	NAME_INCOME_TYPE	[Working, State servant, Commercial associate,...	8

Figure 33-Converting days to columns

NAME_FAMILY_STATUS Column

```
[ ]  # checking % of each category
     (app_data['NAME_FAMILY_STATUS'].value_counts(normalize=True)*100)
```

```
Married                 63.921243
Single / not married    14.736102
Civil marriage           9.681116
Separated                6.432186
Widow                    5.229353
Name: NAME_FAMILY_STATUS, dtype: float64
```

```
[ ]  # Here value % is not that much low but we can club ['Separated', 'Widow'] tp 'Separated/Widow'
     app_data['NAME_FAMILY_STATUS'] = app_data['NAME_FAMILY_STATUS'].replace(
         to_replace=['Separated', 'Widow'],
         value = 'Separated/Widow')
```

NAME_HOUSING_TYPE Column

```
[ ]  # checking % of each category
     (app_data['NAME_HOUSING_TYPE'].value_counts(normalize=True)*100)
```

```
House / apartment      88.740719
With parents            4.824796
Municipal apartment     3.638053
Rented apartment        1.585717
Office apartment        0.848190
Co-op apartment         0.362525
Name: NAME_HOUSING_TYPE, dtype: float64
```

```
[ ]  # With housing type column we can club ['With parents', 'Municipal apartment', 'Rented apartment', 'Office apartment', 'Co-op apartment'] to
     app_data['NAME_HOUSING_TYPE'] = app_data['NAME_HOUSING_TYPE'].replace(
         to_replace=['With parents', 'Municipal apartment', 'Rented apartment', 'Office apartment', 'Co-op apartment'],
         value = 'Other_Apartment')
```

Figure 34: Grouping small categories into one collective category

Need to again separate Categorical and Numerical variables

```
[ ]  new_cat_col =  ['TARGET','REGION_RATING_CLIENT_W_CITY','REGION_RATING_CLIENT'
                    , 'REG_CITY_NOT_WORK_CITY', 'FLAG_EMP_PHONE', 'REG_CITY_NOT_LIVE_CITY', 'FLAG_DOCUMENT_3']
     new_num_col = [ 'EXT_SOURCE_2','YEARS_BIRTH', 'YEARS_LAST_PHONE_CHANGE', 'YEARS_ID_PUBLISH', 'YEARS_EMPLOYE
                   , 'YEARS_REGISTRATION', 'EXT_SOURCE_3']
```

```
[ ]  len(new_cat_col), len(new_num_col)
```

```
     (19, 7)
```

```
[ ]  new_app_data = app_data[new_cat_col + new_num_col].copy()
```

Check balance of TARGET variable

```
[ ]  # Checking if data is imbalance by value counts %
     (new_app_data['TARGET'].value_counts(normalize=True)*100)
```

```
     0    91.900004
     1     8.099996
     Name: TARGET, dtype: float64
```

```
[ ]  # making countplot of target
     sns.countplot(data = new_app_data, x = 'TARGET')
```

```
     <AxesSubplot:xlabel='TARGET', ylabel='count'>
```

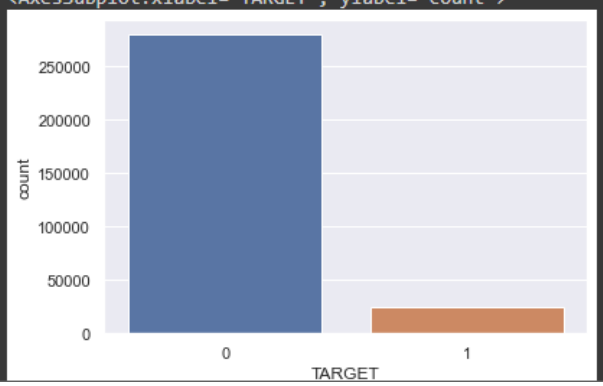


Figure 35-Separate numerical and categorical columns and checking TARGET balance

Univariate Analysis

Numerical Variable

```
[ ]  new_num_col = [ 'EXT_SOURCE_2','YEARS_BIRTH', 'YEARS_LAST_PHONE_CHANGE', 'YEARS_ID_PUBLISH', 'YEARS_EMPLOYED'
                   , 'YEARS_REGISTRATION', 'EXT_SOURCE_3']
```

EXT_SOURCE_2

```
[ ]  # Distribution of 'EXT_SOURCE_2' Feature
     plt.figure(figsize=(10,4))
     plt.subplot(1,2,1)
     plt.title('Probability Distribution of EXT_SOURCE_2')
     sns.distplot(a=app_data1['EXT_SOURCE_2'], label='Defaulter - 1 ', hist=False)
     sns.distplot(a=app_data0['EXT_SOURCE_2'], label='Non Defaulter - 0', hist=False)
     plt.legend()
     plt.subplot(1,2,2)
     sns.boxplot(y = 'EXT_SOURCE_2', x = 'TARGET', data = new_app_data)
     plt.tight_layout()
     plt.show()
```

Figure 36: univariate analysis of numerical columns

Categorical Variables

REGION_RATING_CLIENT_W_CITY

```
[ ]  # Count Distribution of REGION_RATING_CLIENT_W_CITY considering Target Variable
     plt.figure(figsize=(10,4))
     plt.subplot(1,2,1)
     sns.countplot(x = 'REGION_RATING_CLIENT_W_CITY', data = app_data1
                   ,order = app_data1['REGION_RATING_CLIENT_W_CITY'].value_counts().index)
     plt.title("Count of differnt category with Target Value 1 (Defaulter) in \nREGION_RATING_CLIENT_W_CITY")
     plt.subplot(1,2,2)
     sns.countplot(x = 'REGION_RATING_CLIENT_W_CITY', data = app_data0
                   ,order = app_data0['REGION_RATING_CLIENT_W_CITY'].value_counts().index)
     plt.title("Count of differnt category with Target Value 0 (Non-Defaulter) in \nREGION_RATING_CLIENT_W_CITY")
     plt.tight_layout()
     plt.show()
```

Figure 37: univariate analysis of categorical columns

```
[ ]  # get missing values
     prev_miss = (prev_app.isna().mean()*100).reset_index().rename({'index':'Col_Name',0:'Missing_per'},axis = 1)

[ ]  # get columns with missing values greater than 49%
     mis_col = list(prev_miss[prev_miss['Missing_per']>49]['Col_Name'])

[ ]  # Getting Quasi Constant Features
     quasi_const_cols = []
     for i in prev_app.columns:
         if (prev_app[i].value_counts(normalize=True).iloc[0]>0.98):
              quasi_const_cols.append(i)

[ ]  # As we can see 1000 yrs == 365243 in days columns. we need to check those also with missing values
     prev_days = []
     for i in prev_app.columns:
         if 'DAYS' in i:
             prev_days.append(i)

     # Checking missing values
     prev_app[prev_days].isna().mean()

     DAYS_DECISION               0.000000
     DAYS_FIRST_DRAWING          0.402981
     DAYS_FIRST_DUE              0.402981
     DAYS_LAST_DUE_1ST_VERSION   0.402981
     DAYS_LAST_DUE               0.402981
     DAYS_TERMINATION            0.402981
     dtype: float64

Checking columns which has good amount of values as 365243

     # Checking DAYS_FIRST_DRAWING columns pseudo missing values
     prev_app['DAYS_FIRST_DRAWING'].value_counts(normalize = True)*100
```

Figure 38 - Checking data and data quality

```
[ ]  # getting all the needs to be dropped
     col_to_drop = prev_days + ['SK_ID_PREV','SELLERPLACE_AREA'] + quasi_const_cols + mis_col

[ ]  prev_app.drop(labels=col_to_drop, axis = 1,inplace = True)

[ ]  # Getting data from application file, for merging with prev_app data.
     app_data_mer = pd.read_csv('application_data.csv')

[ ]  # taking only relvent data from application file
     app_data_mer = app_data_mer[['SK_ID_CURR','TARGET']]

[ ]  # merging both dataset as previous data on applicant would be helpful for credit worthiness in present.
     merge_app_prev = app_data_mer.merge(prev_app, on='SK_ID_CURR', how = 'inner',suffixes=('_app','_prev'))

[ ]  merge_app_prev = merge_app_prev[list(prev_app.columns)+['TARGET']]
```

Figure 39 - Taking relevant data from application dataset and merging both

Univariate Analysis of Categorical Columns

```
[ ]  # showing count frequency of all columns in prev_app for different Target values (1 & 0).
     for i in prev_cat_col:
         plt.figure(figsize=(12,6))
         plt.subplot(1,2,1)
         sns.countplot(x = i, data = prev_app_data1
                       ,order = prev_app_data1[i].value_counts().index)
         plt.title(f"Count of differnt category with Target Value 1 (Defaulter) in \n {i}")
         plt.xticks(rotation = 90)
         plt.subplot(1,2,2)
         sns.countplot(x = i, data = prev_app_data0
                       ,order = prev_app_data0[i].value_counts().index)
         plt.title(f"Count of differnt category with Target Value 0 (Non  Defaulter) in \n {i}")
         plt.xticks(rotation = 90)
         plt.tight_layout()
         plt.show()
```

Figure 40 - Univariate analysis of categorical columns

Univariate Analysis of Numerical Columns

```
[ ]  prev_num_col = ['HOUR_APPR_PROCESS_START', 'DAYS_DECISION', 'CNT_PAYMENT']
```

```
[ ]  # Showing distribution of numerical feature wit hdifferent target values (1 & 0)
     for i in prev_num_col:
         plt.figure(figsize=(10,4))
         plt.subplot(1,2,1)
         plt.title(f'Probability Distribution of {i}')
         sns.distplot(a=prev_app_data1[i], label='Defaulter - 1 ', hist=False)
         sns.distplot(a=prev_app_data0[i], label='Non Defaulter - 0', hist=False)
         plt.legend()
         plt.subplot(1,2,2)
         sns.boxplot(y = i, x = 'TARGET', data = merge_app_prev)
         plt.tight_layout()
         plt.show()
```

Figure 41 - Univariate analysis of numerical columns

ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

We respect and thank Dr. Prachi Gharpure for providing us an opportunity to do the project work and giving us all support and guidance, which made us complete the project duty. We are extremely thankful to her for providing such nice support and guidance, although she had a busy schedule managing the corporate affairs.

We owe our deepest gratitude to our project guide Dr. Prachi Gharpure and Dr. Aaquil Bunglowala, who took keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staff of STME which helped us in successfully completing our project work. Also, we would like to extend our sincere esteems to all staff in the laboratory for their timely support.

Thankyou

Team members:
Bhavya Sharma 7047191913
Ishica Thukral 70471919020
Purva Patel 70471919009