Content Creator

Portfolio

HIMANSHU JAIN

2023

Professional Background

I am Himanshu Jain, currently in final year in my 3 years M.Sc. Tech. in Applied Geophysics from Indian Institute of Technology (Indian School of Mines), Dhanbad (CGPA 8.17/10). During this course, I got exposure to the subjects like Image Processing and Geographic Information System, Seismological Data analysis and Data Analytics, etc. I have done my graduation with triple major in Computer Science, Physics and Mathematics.

I have worked on several more projects like Medicine Recommendation System, Facies Identification and Employee Attrition Prediction etc. Along which this I have several skills like Machine learning, Data Science, Python, BASH, MATLAB, C language, My-SQL, Web Scrapping and Linux.

I have also worked on research topic ‘Reinforcement learning Based Resolution improvement of Geophysical Data’.

As I am a fresher it would be great to experience the real challenges of the corporate world and understand how things work. Being a fresher, I think I am very flexible and adaptive to learn new things. I have theoretical knowledge. But I am waiting to use my theoretical knowledge in a practical way. And I believe by putting significant efforts I will learn.

Contents

Table of Contents

Module 1: Data Analytics Process

Overview:

We have to analyse the best possible way to get the highest views, Likes, Subscribers lowest dislikes on a YouTube channel. The data for the past 1 year on daily basis is given. The data contains uploaded videos, date and time of uploaded videos, views on a particular single video, category of video (Gaming, roast, vlog etc.), likes and dislikes on a single video, number of comments on video, subscribers at the time video uploaded, number of active subscribers etc. Our aim is to enhance the overall growth of the channel.

Data Analytics Process:

1. Ask
2. Prepare
3. Process
4. Analyse
5. Share
6. Act

1. Ask:

The objective of the project is related to enhance the overall growth of the channel. But this contains multiple aspects. To grow a channel there must be increase in likes, views, subscribers, comments and share, and decrease in dislikes. So, to do that what is our priority. We have to define the order in which these aspects will be treated. Which will we consider in the first and which will be considered in the last. It should also be known that what is current growth of the channel and the features of the channel.

2. Prepare:

The data set needed for this consists of the features views, likes, subscribers, dislikes at the current time and the time when it uploaded to get the growth after uploading each video. It

also needed day to day data for each of the videos (because the videos can be seen after a long time). The data set should have all the possible features which show the growth of the channel.

3. Process:

The data set contains data for all the videos but sometimes it happens, due to YouTube guidelines videos get downvoted or removed from the channels. It should be removed before proceeding to the next step, to avoid bias. Also, the missing information from the data should be treated first. Outliers and errors should also be removed.

4. Analyse:

The next process is to find the patters, correlation and trends in the data and try to extract information as much possible. Find what is the best feature or a trend which will help to grow channel rapidly. Note every outcome of the analysis.

5. Share:

Now communicate the results in the best possible way. Deliver contents in the order in which you find the maximum possibility of the growth of the channel. Try to explain why it is so and other possible ways which have small but significant impact on the growth of the channel. Discuss the outcome and things to implement. Share the next immediate action items that will have the highest impact.

6. Act:

Now work with the peoples of the channel to understand how some actions will help in growth of the channel. Work with different stakeholders to strategize and implement different action items that were highlighted as the outcome of the study.

Module 2: Instagram User Analytics

Project Description:

The project is about the user analytics of Instagram. The project is to analyse the best possible outcomes to some known problems related to marketing and the performance of the Instagram. The data tables given for the project is users, photos, comments, tags, likes, follow, photo_tags. MySQL is used to solve the problems.

Approach:

To start with the project first I understood all the problem statement. Tried to find out what tables will be required to find the best possible result and marked it to use while actual query writing. The queries should be easy understand. I have written each of the primary and foreign key for the tables. So, at the time of writing query I have not to check again and again to get that columns.

Tech-Stack Used:

To solve these problems, I have used MySQL Workbench 8.0 CE. Which is an open software and can be downloaded from https://www.mysql.com/.

Insights:

The project is extremely helpful to understand basics of MySQL. It helped me to learn the structure. It also helped me to learn new keywords like dayofweek etc. I have also learned the concept of JOIN, HAVING, WHERE, IN, NOT IN, GROUP BY, ORDER BY, etc. This project gave me the confidence to work in SQL.

Result:

A) Marketing: The marketing team wants to launch some campaigns. Help them.

1. Rewarding most loyal users: People who have been using the platform for the longest time.

```
1  /* Find the 5 oldest users of the Instagram from the database provided */
2  select distinct username from users order by created_at limit 5;
```

username
Darby_Herzog
Emilio_Bernier52
Elenor88
Nicole71
Jordyn.Jacobson2

2. Remind Inactive Users to Start Posting: By sending them promotional emails to post their 1st photo.

```
1  /* Find the users who have never posted a single photo on Instagram */
2  select * from users where id not in (select user_id from photos);
```

id	username	created_at
5	Aniya_Hackett	2016-12-07 01:04:39
7	Kasandra_Homenick	2016-12-12 06:50:08
14	Jaclyn81	2017-02-06 23:29:16
21	Rocio33	2017-01-23 11:51:15
24	Maxwell.Halvorson	2017-04-18 02:32:44
25	Tierra.Trantow	2016-10-03 12:49:21
34	Pearl7	2016-07-08 21:42:01
36	Ollie_Ledner37	2016-08-04 15:42:20
41	Mckenna17	2016-07-17 17:25:45
45	David.Osinski47	2017-02-05 21:23:37

3. Declaring Contest Winner: The team started a contest and the user who gets the most likes on a single photo will win the contest now they wish to declare the winner.

```
/*  Identify the winner of the contest and provide their details to the team */
select users.id,users.username,photos.image_url,likes.photo_id,
count(likes.user_id) no_of_likes from likes
join photos on likes.photo_id=photos.id join users on users.id=photos.user_id
group by likes.photo_id order by no_of_likes desc limit 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

id	username	image_url	photo_id	no_of_likes
52	Zack_Kemmer93	https://jarret.name	145	48

4. Hashtag Researching: A partner brand wants to know, which hashtags to use in the post to reach the most people on the platform.

```
/*  Identify and suggest the top 5 most commonly used hashtags on the platform */
select tags.id, tags.tag_name, count(photo_id) as no_of_times from tags
join photo_tags on tags.id=photo_tags.tag_id
group by photo_tags.tag_id order by no_of_times desc limit 5
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

id	tag_name	No_of_Times_Used
21	smile	59
20	beach	42
17	party	39
13	fun	38
18	concert	24

5. Launch AD Campaign: The team wants to know, which day would be the best day to launch ADs.

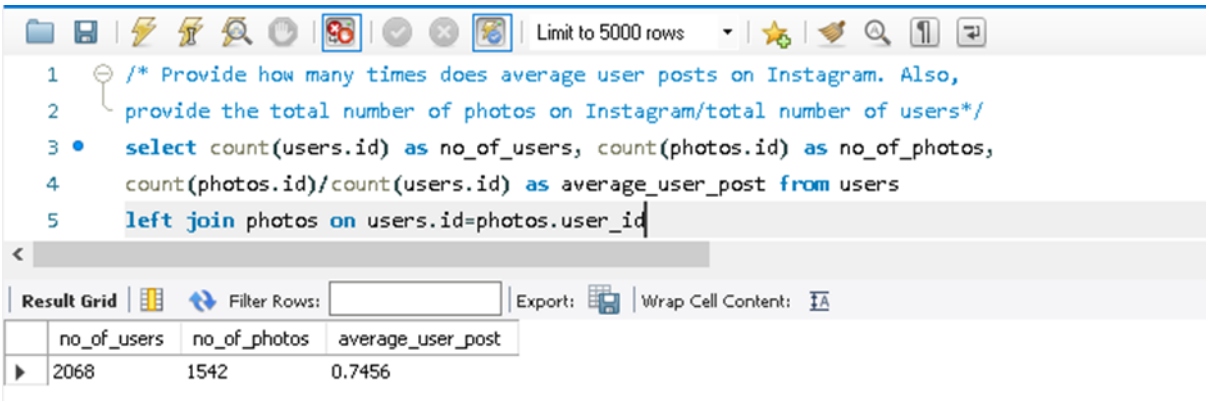
```
/* What day of the week do most users register on?
Provide insights on when to schedule an ad campaign */
select dayofweek(created_at) as day_of_week,count(created_at) as count from users
group by day_of_week order by count desc
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

day_of_week	count
5	96
1	96
6	90
3	84
2	84
4	78
7	72

B) Investor Metrics:

1. User Engagement: Are users still as active and post on Instagram or they are making fewer posts.

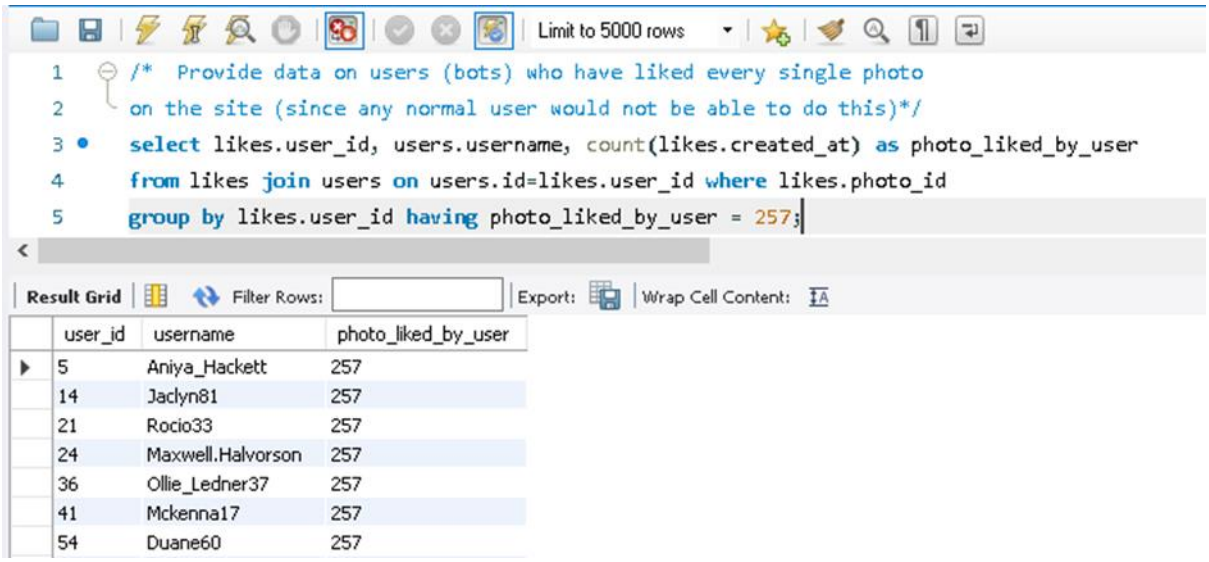


```
/* Provide how many times does average user posts on Instagram. Also,
provide the total number of photos on Instagram/total number of users*/
select count(users.id) as no_of_users, count(photos.id) as no_of_photos,
count(photos.id)/count(users.id) as average_user_post from users
left join photos on users.id=photos.user_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

no_of_users	no_of_photos	average_user_post
2068	1542	0.7456

2. Bots & Fake Accounts: The investors want to know if the platform is crowded with fake and dummy accounts.



```
/*  Provide data on users (bots) who have liked every single photo
on the site (since any normal user would not be able to do this)*/
select likes.user_id, users.username, count(likes.created_at) as photo_liked_by_user
from likes join users on users.id=likes.user_id where likes.photo_id
group by likes.user_id having photo_liked_by_user = 257;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

user_id	username	photo_liked_by_user
5	Aniya_Hackett	257
14	Jaclyn81	257
21	Rocio33	257
24	Maxwell.Halvorson	257
36	Ollie_Ledner37	257
41	Mckenna17	257
54	Duane60	257

Module 3: Operation and Metric Analytics

Description:

Operation analytics is the subset of data analytics that mainly focuses on the improving efficiency of the operations in the company. It shows how currently different operations are working and how those operations can be improved for better profits. In this project, I have answered such questions which are generally asked by the different departments in a company like the ops team, support team, marketing team, etc. which are required to increase efficiency and streamline.

Investigating metric spikes is an important part of operation analytics. The metric spike shows the anomaly in the trends. It gives the answers to questions like- why there is a dip in daily engagement. why have sales taken a dip? etc. These questions are to be answered daily or weekly basis and should be treated seriously. For this task, different data sets are given.

Approach:

To start with the project first I understood all the problem statements. Tried to find out what tables will be required to find the best possible result and marked it to use while actual query writing. The queries should be easy to understand. I have written each of the primary and foreign keys for the tables. So, at the time of writing the query, I did have not to check again and again to get those columns. Studied unknown terms like throughput.

Tech-Stack used:

To solve these problems, I have used MYSQL Workbench 8.0 CE. Which is an open software and can be downloaded from https://www.mysql.com/.

Insights:

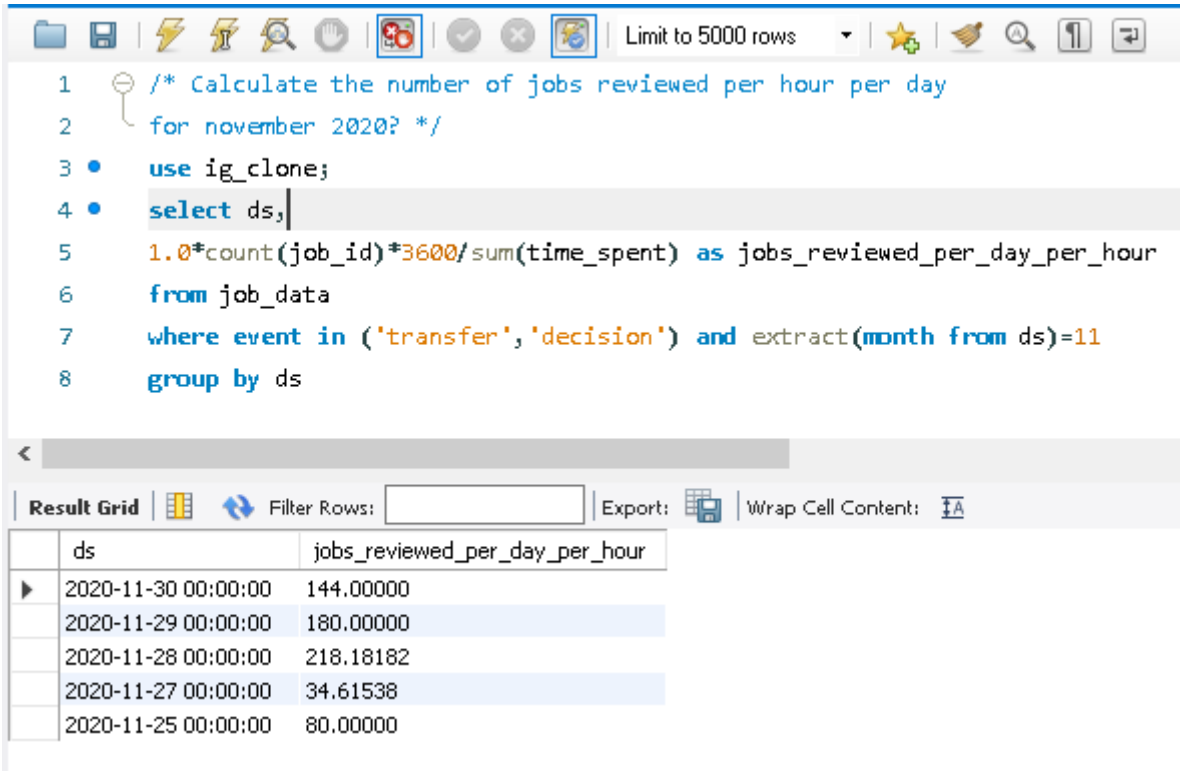
The project is extremely helpful to understand the basics of MySQL. It helped me to learn the structure. It also helped me to learn new keywords like week, day, etc. I have also learned the concept of BETWEEN, GROUP BY, ORDER BY, CASE, Window function, partition by, over, rows between, etc. This project gave me the confidence to work in SQL.

Also, I have learned to import CSV files in the MYSQL workbench. But the files consist of a high number of rows which took a lot of time.

Results:

Case Study 1 (Job Data):

A. **Number of jobs reviewed:** Amount of jobs reviewed over time.



```
/* Calculate the number of jobs reviewed per hour per day
for november 2020? */
use ig_clone;
select ds,
1.0*count(job_id)*3600/sum(time_spent) as jobs_reviewed_per_day_per_hour
from job_data
where event in ('transfer','decision') and extract(month from ds)=11
group by ds
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ds	jobs_reviewed_per_day_per_hour
2020-11-30 00:00:00	144.00000
2020-11-29 00:00:00	180.00000
2020-11-28 00:00:00	218.18182
2020-11-27 00:00:00	34.61538
2020-11-25 00:00:00	80.00000

B) Throughput: It is the no. of events happening per second. Let's say the above metric is called throughput. For throughput, do you prefer daily metric or 7-day rolling, and why?

If the density of the data is larger, we use the daily metric, and if the density is low the 7-day rolling works well. It also depends upon the anomaly in the data set. Because in 7 days metric the view is broader similar to the daily metric view becomes narrower.

```sql
/* Calculate 7 day rolling average of throughtput? */
select ds, round(1*sum(count(job_id))over(order by ds rows between 6
preceding and current row) /(sum(sum(time_spent))
over (order by ds rows between 6 preceding and current row)),2)
as throughtput_7d
from job_data
where event in ('transfer','decision')
group by ds
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ds	throughtput_7d
2020-11-25 00:00:00	0.02
2020-11-27 00:00:00	0.01
2020-11-28 00:00:00	0.02
2020-11-29 00:00:00	0.02
2020-11-30 00:00:00	0.03

C) Percentage share of each language: Share of each language for different contents.

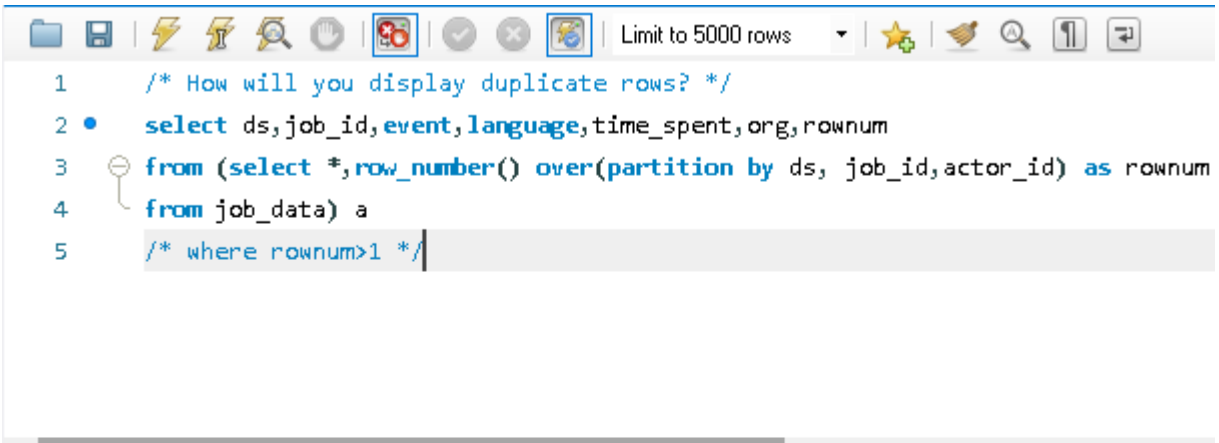
```sql
/* Calculate the percentage share of each language in the lsat 30 days? */
select language, count(job_id) as job_count, count(job_id)*100/total_jobs
as per_share from job_data
cross join (select count(job_id) as total_jobs from job_data) a
group by language
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

language	job_count	per_share
English	1	12.5000
Arabic	1	12.5000
Persian	3	37.5000
Hindi	1	12.5000
French	1	12.5000
Italian	1	12.5000

D) Duplicate rows: Rows that have the same value present in them.

When the row has been duplicated the value of the 'rownum' column in the output will be greater than 1. And the duplicate rows can be extracted by removing the comment in the where clause.



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ds	job_id	event	language	time_spent	org	rownum
2020-11-25 00:00:00	20	transfer	Italian	45	C	1
2020-11-26 00:00:00	23	skip	Persian	56	A	1
2020-11-27 00:00:00	11	decision	French	104	D	1
2020-11-28 00:00:00	23	transfer	Persian	22	D	1
2020-11-28 00:00:00	25	decision	Hindi	11	B	1
2020-11-29 00:00:00	23	decision	Persian	20	C	1
2020-11-30 00:00:00	21	skip	English	15	A	1
2020-11-30 00:00:00	22	transfer	Arabic	25	B	1

Case Study 2 (Investigating Metric Spike):

A) User Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service.

```
1   /* Calculate the weekly user engagement */
2   SELECT week(STR_TO_DATE(occurred_at, '%m/%d/%Y %H:%i')) as week_num,
3   count(distinct user_id) as user_engagement from events
4   where event_type ='engagement' and
5   event_name='login'
6   group by 1
7   order by 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

week_num	user_engagement
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299
32	1225
33	1225
34	1204
35	104

B) User Growth: Amount of users growing over time for a product.

```
/* Calculate user growth for product.*/
select day(created_at) as day,
count(*) as all_users,
count(activated_at) as activated_users
from users u
where created_at>='2013-03-01'
and created_at<'2013-03-31'
group by 1 order by 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

day	all_users	activated_users
1	15	8
2	4	1
3	4	0
4	18	9
5	13	7
6	15	7
7	15	9
8	15	8
9	4	3
10	5	0
11	16	8
12	17	5
13	15	6
14	16	8
15	15	4
16	4	1
17	4	1
18	17	6
19	17	4
20	17	5
21	18	7
22	18	8
23	4	0
24	4	2
25	19	7
26	17	7

C) Weekly Retention: Users get retained weekly after signing up for a product.

```
/* Calculate the weekly retention of user=-sign up cohort. */
select first_week,
sum(case when week_num=1 then 1 else 0 end) as week_0,
sum(case when week_num=2 then 1 else 0 end) as week_1,
sum(case when week_num=3 then 1 else 0 end) as week_2,
sum(case when week_num=4 then 1 else 0 end) as week_3,
sum(case when week_num=5 then 1 else 0 end) as week_4,
sum(case when week_num=6 then 1 else 0 end) as week_5,
sum(case when week_num=7 then 1 else 0 end) as week_6,
sum(case when week_num=8 then 1 else 0 end) as week_7,
sum(case when week_num=9 then 1 else 0 end) as week_8,
sum(case when week_num=10 then 1 else 0 end) as week_9,
sum(case when week_num=11 then 1 else 0 end) as week_10
from (select a.user_id, week, first_week,(week-first_week)
as week_num from
(select user_id, week(STR_TO_DATE(occurred_at, '%m/%d/%Y %H:%i')) as week
from events group by user_id, week) a,
(select user_id, min(week(STR_TO_DATE(occurred_at, '%m/%d/%Y %H:%i')))
as first_week
from events group by user_id) b
where a.user_id=b.user_id) as with_week_number
group by first_week order by first_week;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_week	week_0	week_1	week_2	week_3	week_4	week_5	week_6	week_7	week_8	week_9	week_10
17	472	324	251	205	187	167	146	145	145	136	131
18	362	261	203	168	147	144	127	113	122	106	118
19	284	173	153	114	95	91	81	95	82	68	65
20	223	165	121	91	72	63	67	63	65	67	41
21	187	131	91	74	63	75	72	58	48	45	39
22	224	150	107	87	73	63	60	55	48	41	39
23	219	138	101	90	79	69	61	54	47	35	30
24	205	143	102	81	63	65	61	38	39	29	0
25	218	139	101	75	63	50	46	38	35	2	0
26	181	114	83	73	55	47	43	29	0	0	0
27	199	121	106	68	53	40	36	1	0	0	0
28	194	114	69	46	30	28	3	0	0	0	0
29	186	102	65	47	40	1	0	0	0	0	0
30	202	121	78	53	3	0	0	0	0	0	0
31	145	76	57	1	0	0	0	0	0	0	0
32	188	94	8	0	0	0	0	0	0	0	0
33	202	9	0	0	0	0	0	0	0	0	0

D) Weekly Engagement: To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

```sql
/* Calculate the weekly engagement per device */
select first_week,weekly_number.device,
sum(case when week_num=1 then 1 else 0 end) as week_0,
sum(case when week_num=2 then 1 else 0 end) as week_1,
sum(case when week_num=3 then 1 else 0 end) as week_2,
sum(case when week_num=4 then 1 else 0 end) as week_3,
sum(case when week_num=5 then 1 else 0 end) as week_4,
sum(case when week_num=6 then 1 else 0 end) as week_5,
sum(case when week_num=7 then 1 else 0 end) as week_6,
sum(case when week_num=8 then 1 else 0 end) as week_7,
sum(case when week_num=9 then 1 else 0 end) as week_8,
sum(case when week_num=10 then 1 else 0 end) as week_9,
sum(case when week_num=11 then 1 else 0 end) as week_10
from (select a.user_id,device, week, first_week,(week-first_week)
as week_num from
(select user_id,device, week(STR_TO_DATE(occurred_at, '%m/%d/%Y %H:%i'))
as week
from events where event_type='engagement' group by user_id, week,device) a,
(select user_id, min(week(STR_TO_DATE(occurred_at, '%m/%d/%Y %H:%i')))
as first_week from events group by user_id) b
where a.user_id=b.user_id) as weekly_number
group by 1,2 order by 1,2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_week	device	week_0	week_1	week_2	week_3	week_4	week_5	week_6	week_7	week_8
27	kindle fire	5	3	1	0	0	0	0	0	0
27	lenovo thinkpad	28	20	11	7	9	7	8	0	0
27	mac mini	4	2	2	2	1	0	0	0	0
27	macbook air	31	12	10	8	8	2	6	0	0
27	macbook pro	52	32	20	17	11	9	10	0	0
27	nexus 10	2	2	3	1	2	1	3	0	0
27	nexus 5	13	8	5	4	1	1	1	0	0
27	nexus 7	7	1	5	3	1	2	0	0	0
27	nokia lumia 635	2	2	0	0	0	0	1	0	0
27	samsumg galaxy tablet	3	2	1	1	1	0	1	0	0
27	samsung galaxy note	0	0	0	0	0	0	0	0	0
27	samsung galaxy s4	18	9	11	2	4	2	1	0	0
27	windows surface	3	4	0	3	0	0	0	0	0
28	acer aspire desktop	4	1	1	1	2	1	0	0	0
28	acer aspire notebook	8	4	4	3	0	1	0	0	0
28	amazon fire phone	2	1	0	0	0	0	0	0	0
28	asus chromebook	8	5	3	4	1	1	0	0	0

E) Email Engagement: Users engaging with the email service.

```sql
/* Calculate the email engagement metrics */
select week(occurred_at) as week,
count(case when action='sent_weekly_digest' then user_id else null end)
as weekly_digest,
count(case when action='email_open' then user_id else null end)
as email_open,
count(case when action='email_clickthrough' then user_id else null end)
as email_clickthrough,
count(case when action='sent_reengagement_email' then user_id else null end)
as reengagement_email
from email_events
group by 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

week	weekly_digest	email_open	email_clickthrough	reengagement_email
18	2602	912	430	157
19	2665	972	477	173
20	2733	1004	507	191
21	2822	1014	443	164
22	2911	987	488	192
23	3003	1075	538	197
24	3105	1155	554	226
25	3207	1096	530	196
26	3302	1165	556	219
27	3399	1228	621	213
28	3499	1250	599	213
29	3592	1219	590	213
30	3706	1383	630	231
31	3793	1351	445	222
32	3897	1337	418	200
33	4012	1432	490	264
34	4111	1528	490	261
17	908	310	166	73
35	0	41	38	48

Module 4: Hiring Process Analytics

Project Description:

Hiring process is the fundamental and the most important function of a company. Here, the MNCs get to know about the major underlying trends about the hiring process. Trends such as- number of rejections, number of interviews, types of jobs, vacancies etc. are important for a company to analyse before hiring freshers or any other individual.

The data set given is of a company where details about the people who registered for a particular post in a department of this company.

Approach:

First, to understand the data set I have performed the EDA on the data set. Checked for the null values and the distribution. Now to answer each question I first understood all the questions. What should be outcome for each of the question and what columns to use. Now to get result I checked all the functions which will be required to perform the operations.

Tech-Stack Used:

For this assignment I have used Microsoft Excel (2016).

Insights:

This assignment helps me to understand how a company use hiring data to track the acceptance and rejection for a particular post. I have used pie chart, histogram, and bar plots. I have learned max, min, and average functions.

Result:

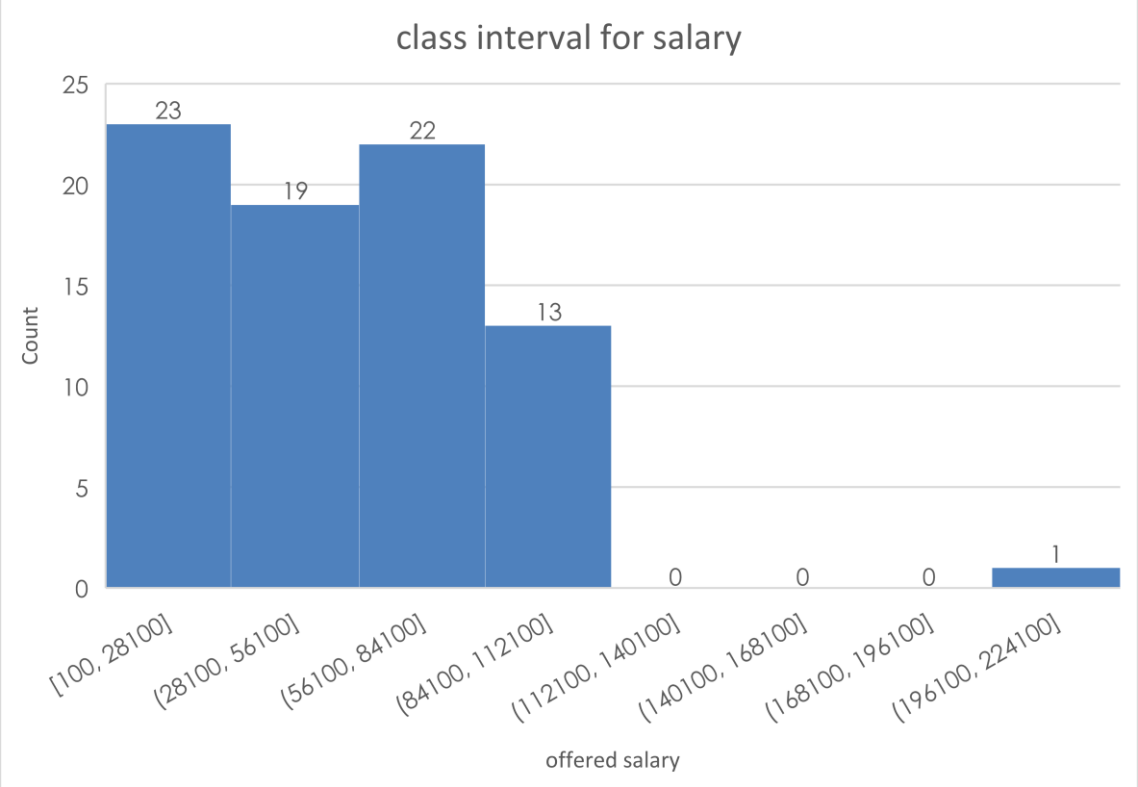
A. **Hiring:** Process of intaking of people into an organization for different kinds of positions.
Your task: How many males and females are Hired?

Male	4085
Female	2675

B. **Average Salary:** Adding all the salaries for a select group of employees and then dividing the sum by the number of employees in the group.
Your task: What is the average salary offered in this company?

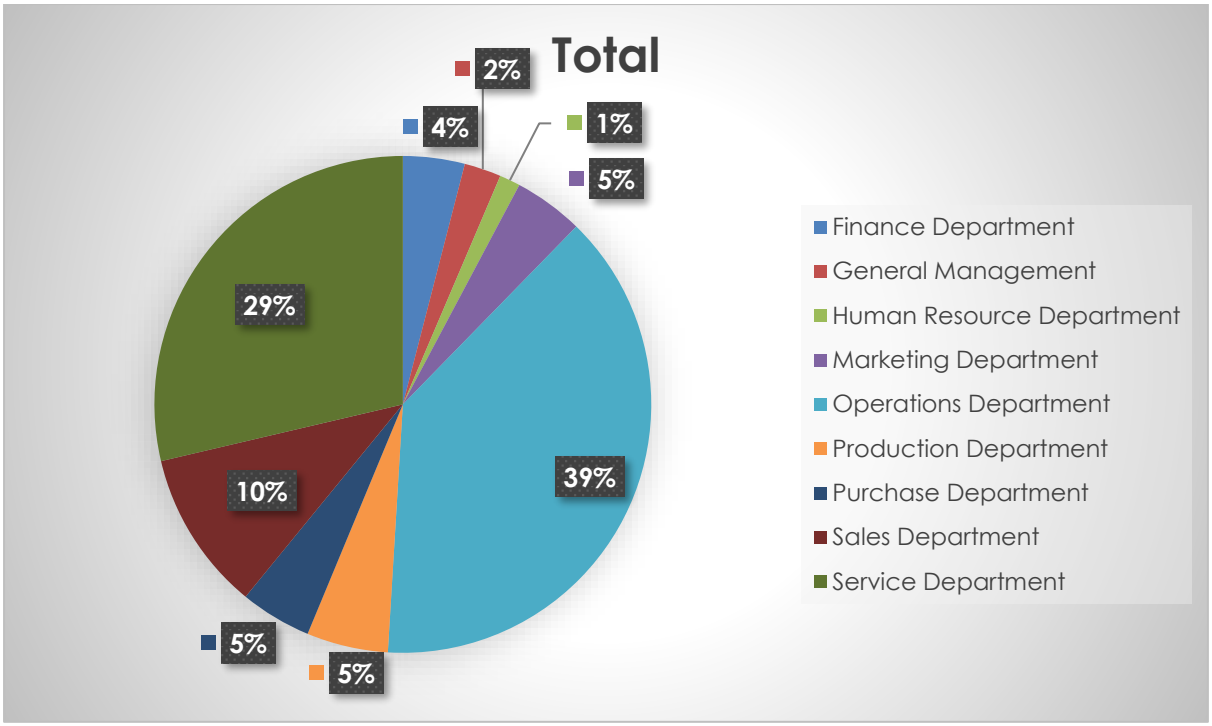
Avg Salary	49983.02902

C. **Class interval:** The class interval is the difference between the upper-class limit and the lower-class limit.
Your task: Draw the class intervals for salary in the company?



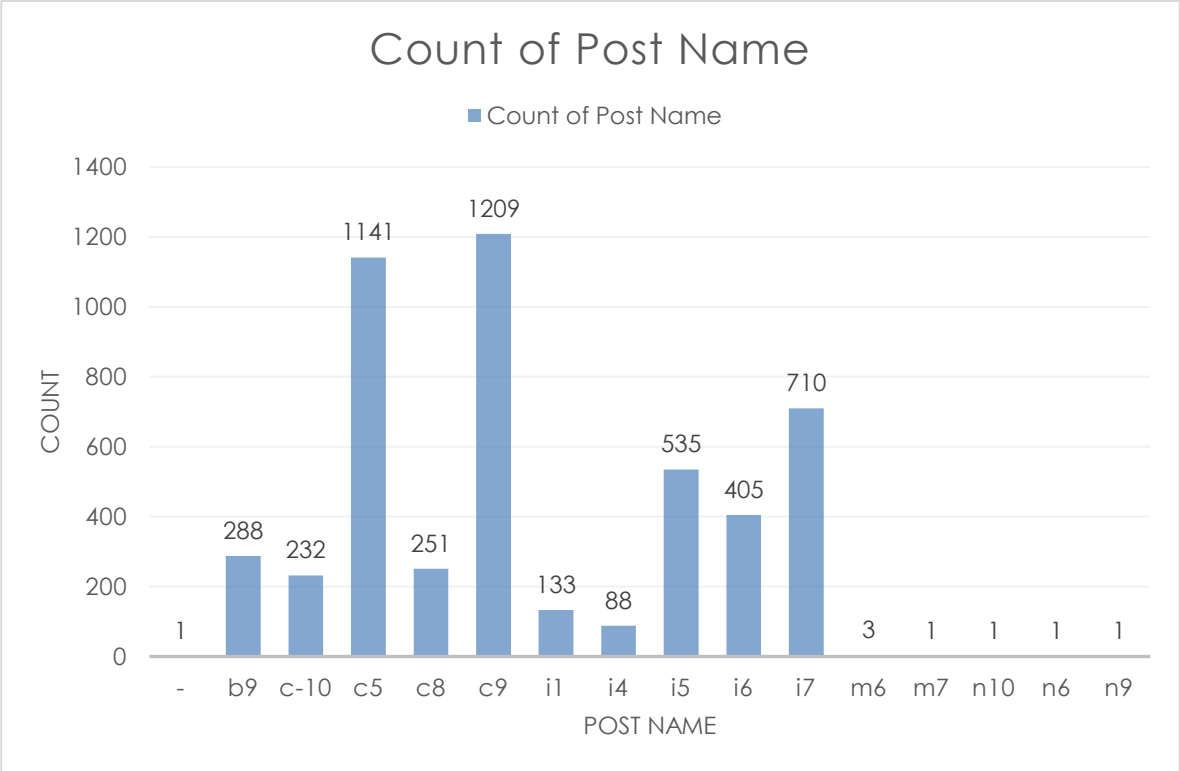
D. **Charts and Plots:** This is one of the most important parts of analysis to visualize the data.
Your task: Draw Pie Chart / Bar Graph (or any other graph) to show proportion of people working different department?

Row Labels	Count of Department
Finance Department	288
General Management	172
Human Resource Department	97
Marketing Department	325
Operations Department	2771
Production Department	380
Purchase Department	333
Sales Department	747
Service Department	2055
Grand Total	**7168**



E. Charts: Use different charts and graphs to perform the task representing the data.

Your task: Represent different post tiers using chart/graph?



Module 5: IMDB Movie Analysis

Project Description:

The project is to analyse the movies trend, which movie is performing better, reason behind that, which movie is more popular, which actor is popular and its root cause. The data set given for this project consist of 28 columns and 5044 data points. Which contains actor names, IMDB rating, movie title, year, genres, director, reviews, language, budget, gross, etc. This also helps us to understand root cause analysis 'Five Whys' approach.

Approach:

First download the data and analyse it by checking for the null values in each row as well as in each column, duplicate rows and in which column data can be interpolated. Identified for each of the question which approach to be best. Which column is most useful to answer the questions and the null values removed from it. For each of the result the chart is the best approach to present it. So, I created chart for each of the result.

Tech-Stack Used:

To perform tasks for the project I have used Microsoft Excel (2016), Microsoft Word (2016).

Insights:

This assignment helps me to understand the functions in MS Excel and the pivot table. This gave me a complete idea of using the excel and the power of excel itself. Also, how a movie performance affect with the time, actor selection, director, reviews and IMDB rating.

Results:

A. **Cleaning the data:** This is one of the most important steps to perform before moving forward with the analysis. Use your knowledge learned till now to do this. (Dropping columns, removing null values, etc.)

Your task: Clean the data

The total number of datapoint before cleaning was 5043.

Step No.	Step	Rows Removed	Rows Left
1	Remove Duplicates	45	4998
2	Removes those rows which have more than 7 columns Null	19	4979
3	Remove rows where gross = Null	859	4120
4	Remove rows where budget = Null	263	3857
5	Replace 'Null' values in language column with 'English'	3857	3857
6	Removes those rows which have more than 3 columns Null	6	3851

Rows left after the cleaning is 3851 which is 76.3% of the original data.

B. Movies with highest profit: Create a new column called profit which contains the difference of the two columns: gross and budget. Sort the column using the profit column as reference. Plot profit (y-axis) vs budget (x- axis) and observe the outliers using the appropriate chart type.

Your task: Find the movies with the highest profit?

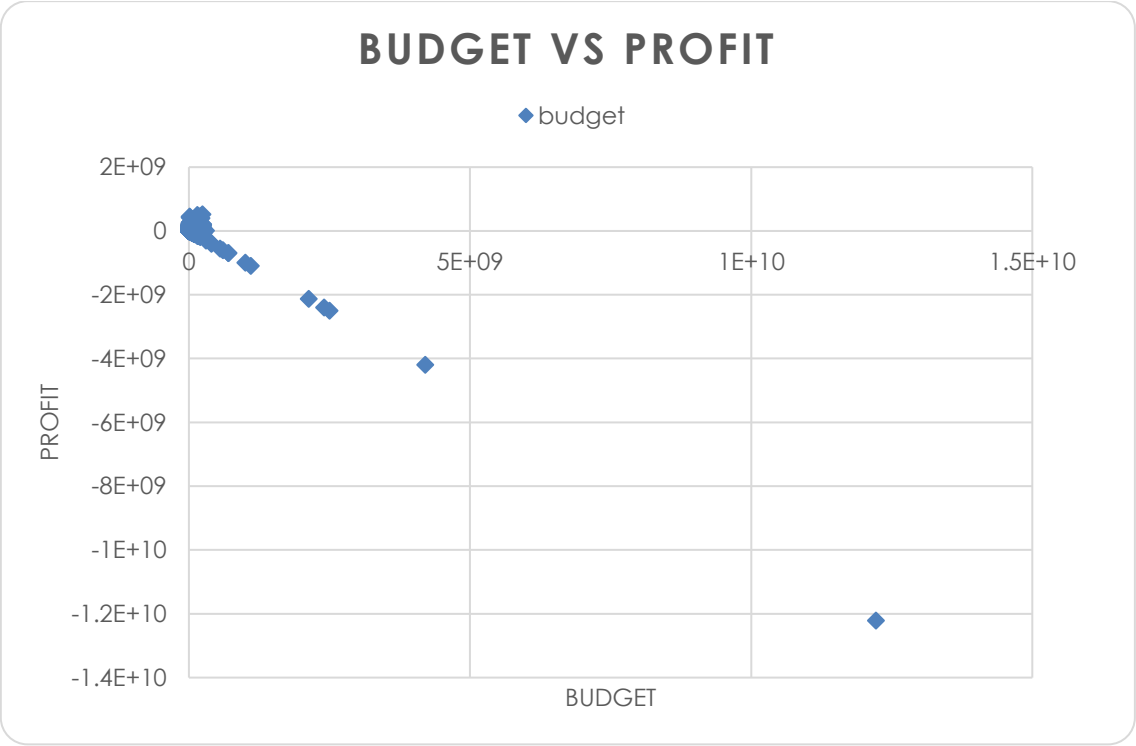


Fig. 1: Budget vs Profit with outliers.

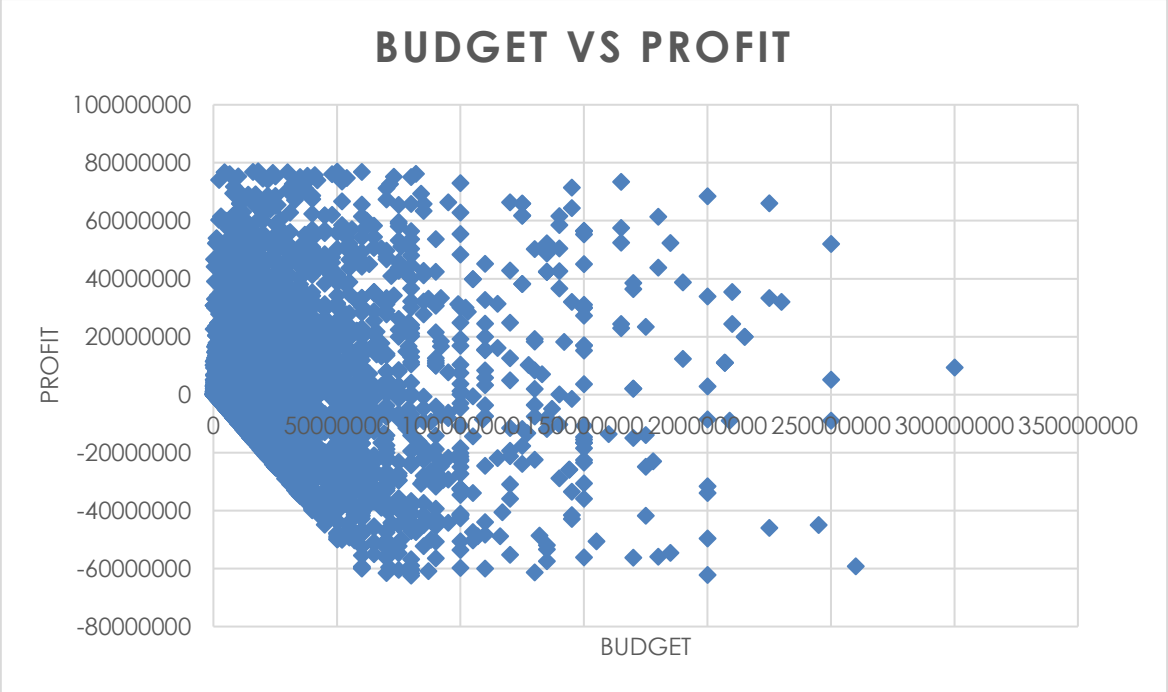


Fig. 2: Budget vs Profit without outliers.

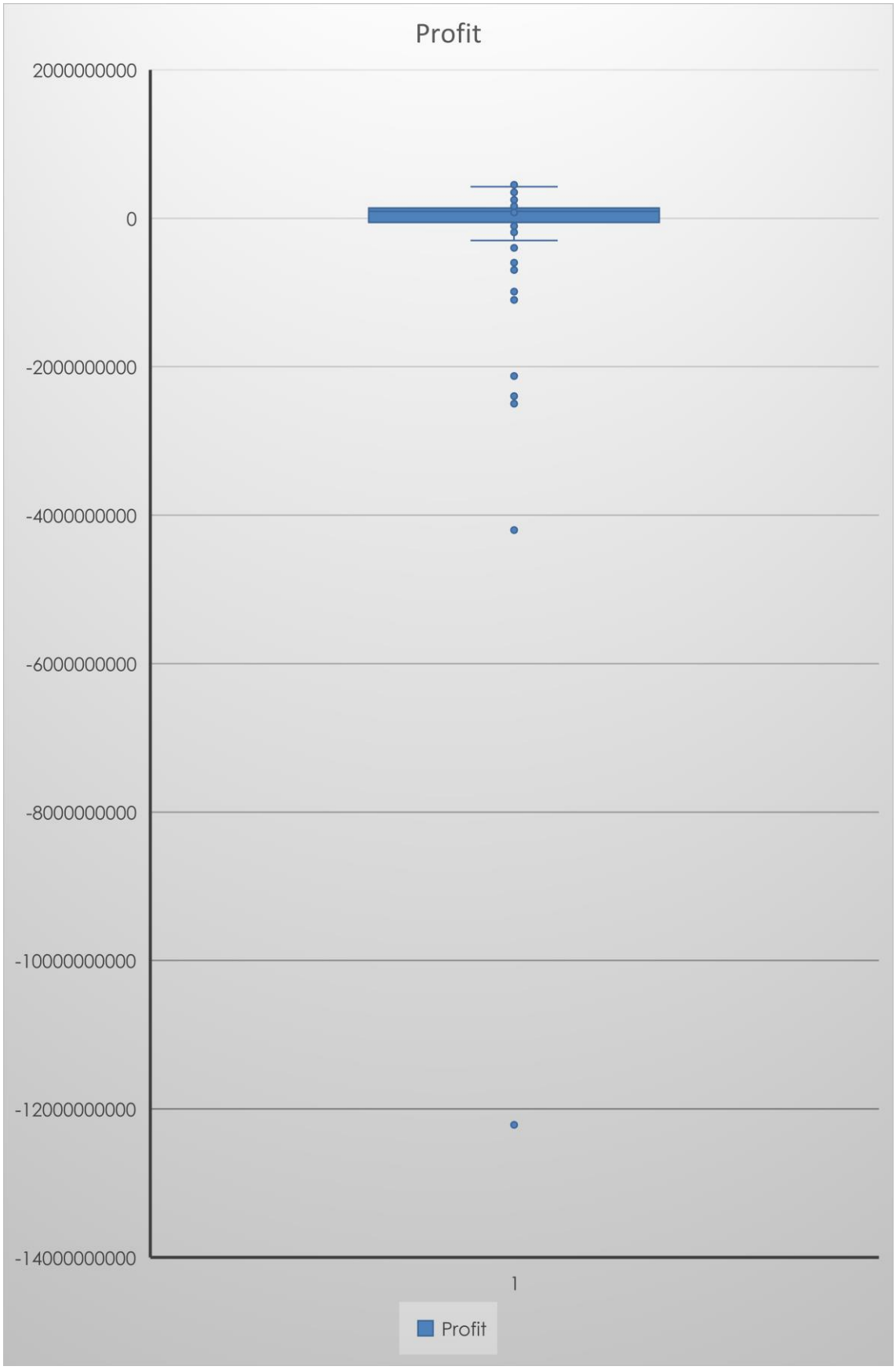


Fig. 3: Box & Whisker Plot with Outliers for profit.

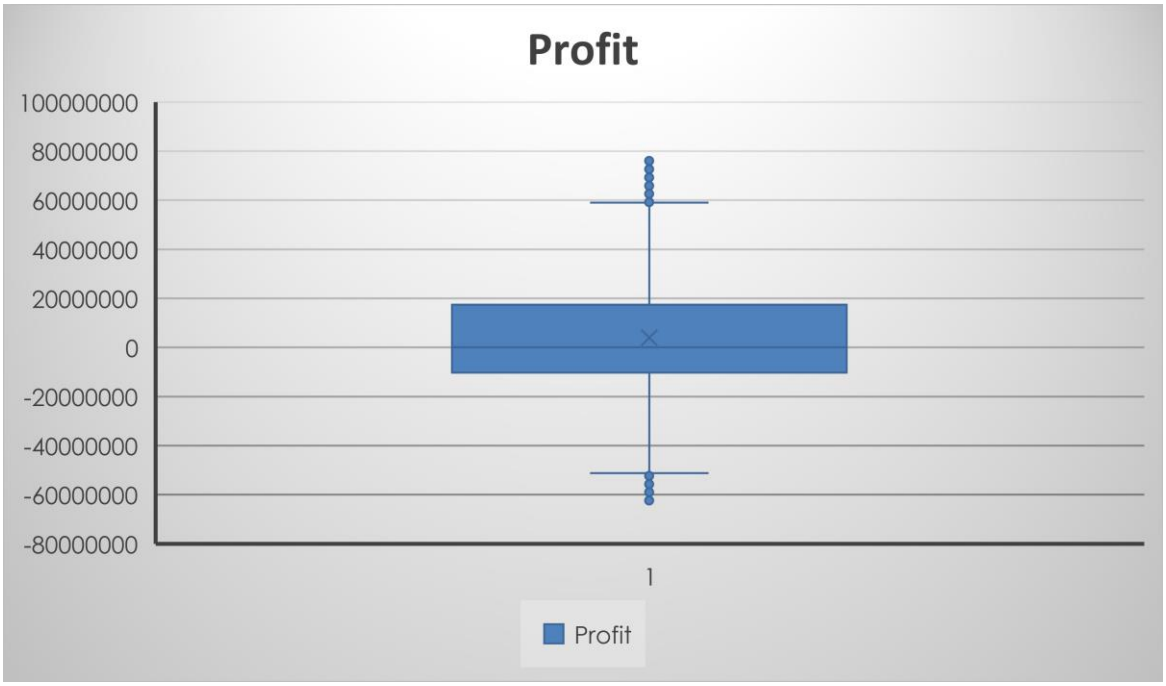


Fig. 4: Box & Whisker plot without Outliers for profit.

C. Top 250: Create a new column IMDb_Top_250 and store the top 250 movies with the highest IMDb Rating (corresponding to the column: imdb_score). Also make sure that for all of these movies, the num_voted_users are greater than 25,000. Also add a Rank column containing the values 1 to 250 indicating the ranks of the corresponding films.

Extract all the movies in the IMDb_Top_250 column which are not in the English language and store them in a new column named Top_Foreign_Lang_Film. You can use your own imagination also.

Your task: Find IMDB Top 25

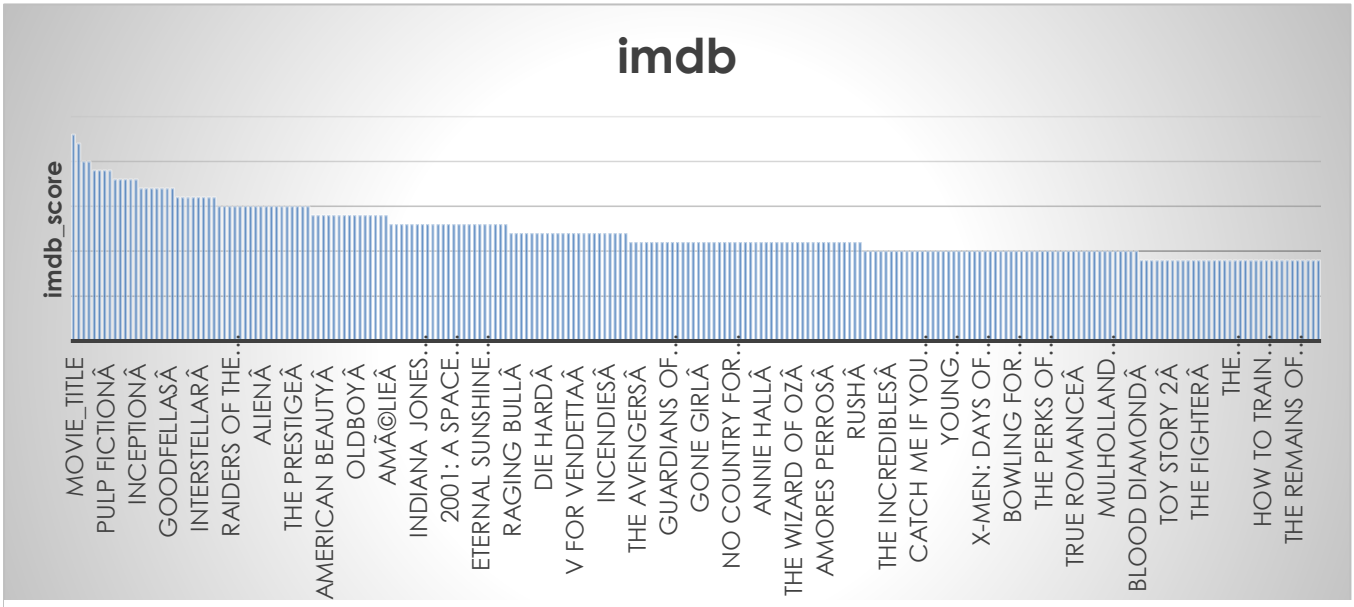


Fig. 5: Top 250 Movies

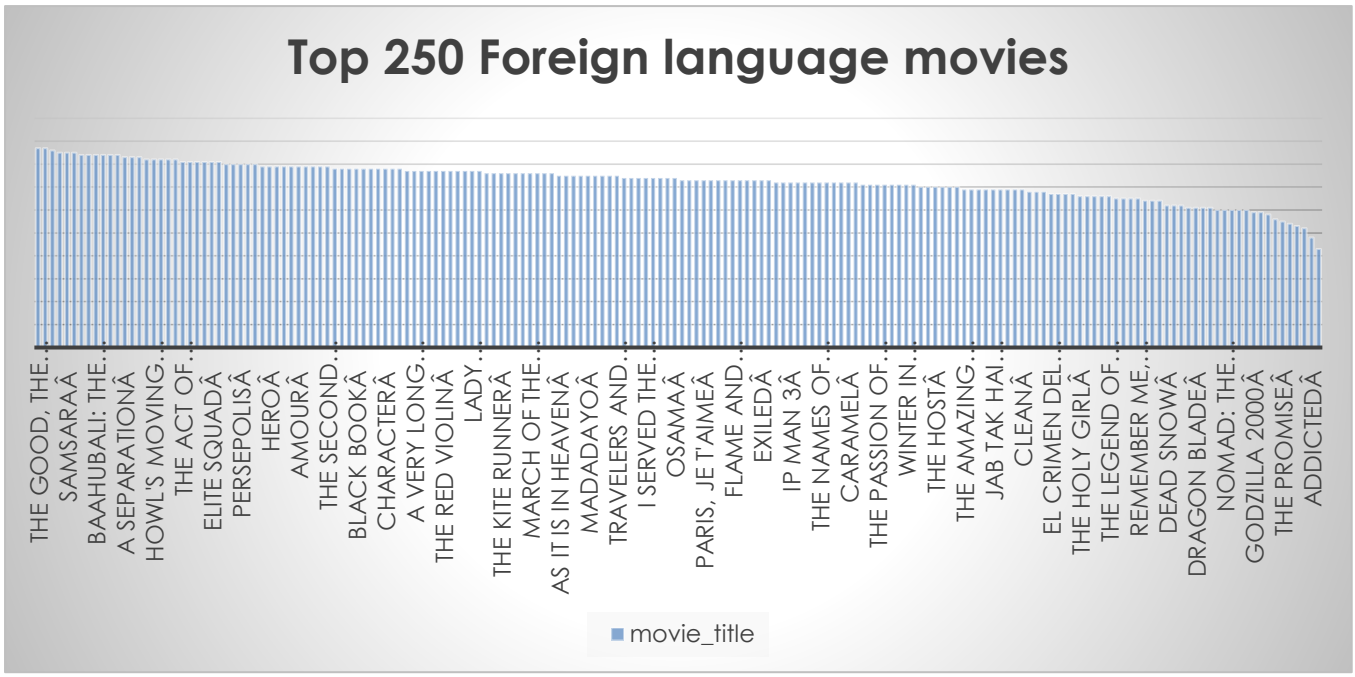


Fig. 6: Top 250 Foreign language movies.

D. Best Directors: TGroup the column using the director_name column.

Find out the top 10 directors for whom the mean of imdb_score is the highest and store them in a new column top10director. In case of a tie in IMDb score between two directors, sort them alphabetically.

Your task: Find the best directors

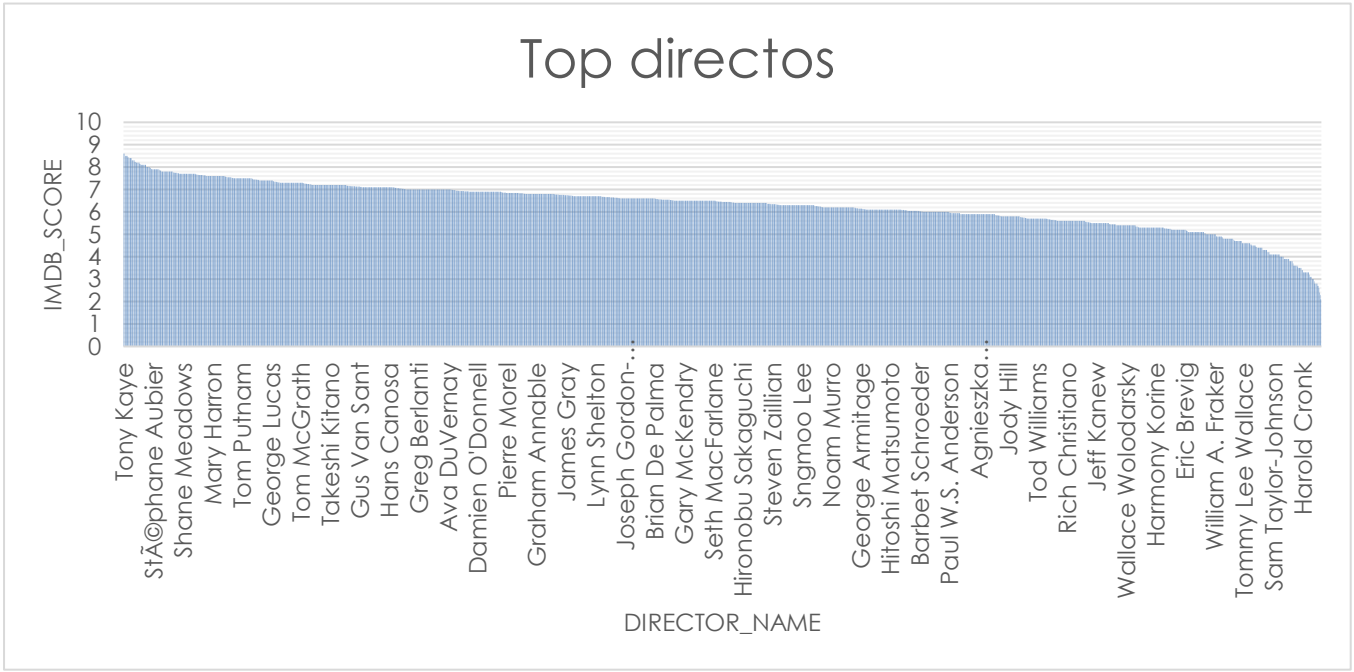


Fig. 7: Top director

E. Popular Genres: Perform this step using the knowledge gained while performing previous steps.

Your task: Find popular genres

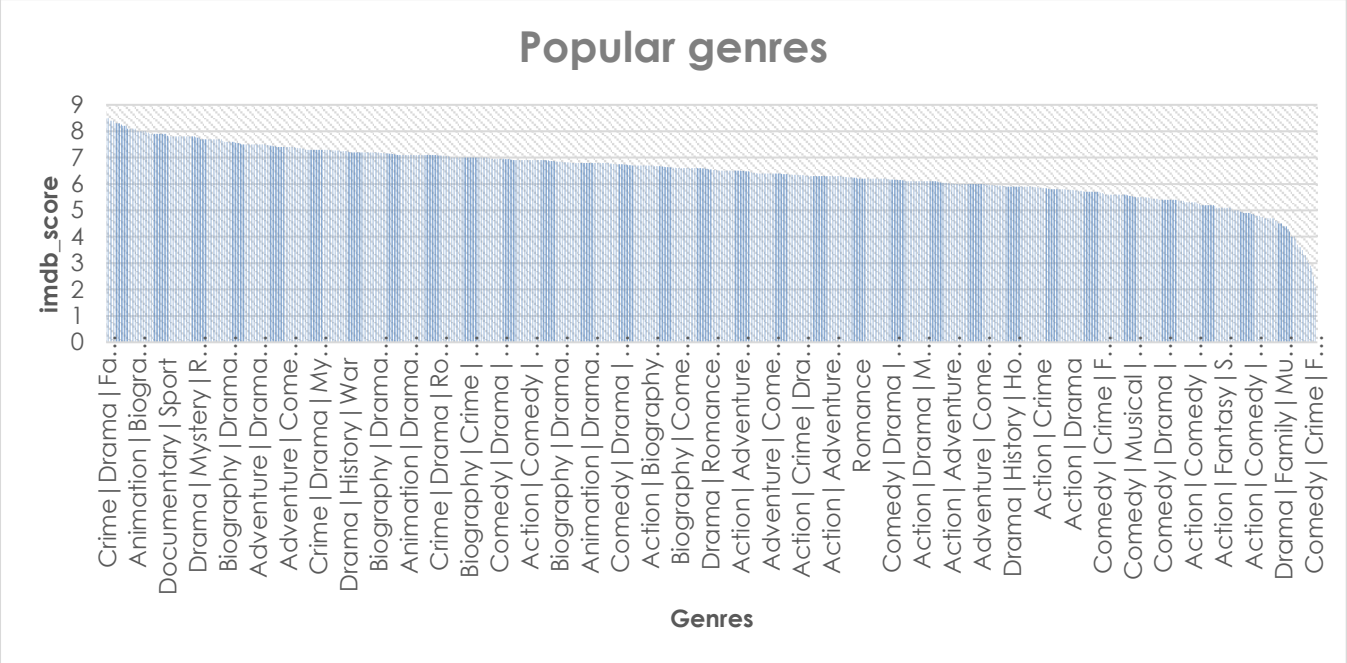


Fig. 8: Popular Genres.

F. Charts: Create three new columns namely, Meryl_Streep, Leo_Caprio, and Brad_Pitt which contain the movies in which the actors: 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' are the lead actors. Use only the actor_1_name column for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.

Append the rows of all these columns and store them in a new column named Combined.

Group the combined column using the actor_1_name column.

Find the mean of the num_critic_for_reviews and num_users_for_review and identify the actors which have the highest mean.

Observe the change in number of voted users over decades using a bar chart. Create a column called decade which represents the decade to which every movie belongs to. For example, the title_year year 1923, 1925 should be stored as 1920s. Sort the column based on the column decade, group it by decade and find the sum of users voted in each decade. Store this in a new data frame called df_by_decade.

Your task: Find the critic-favorite and audience-favorite actors

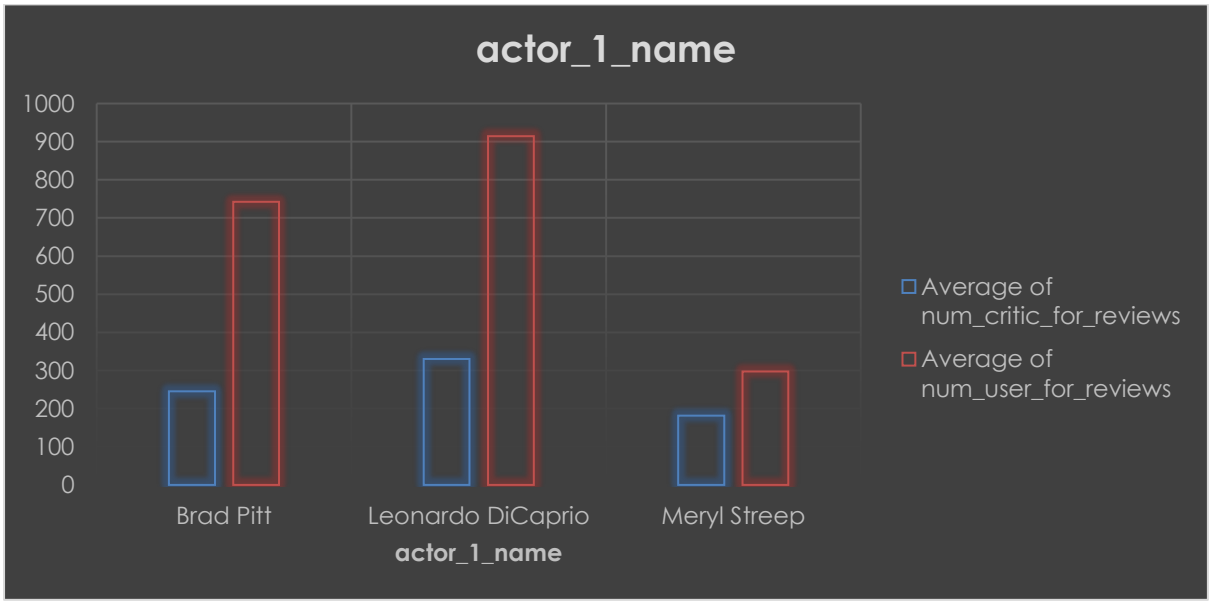


Fig. 9: actor_1_name vs num_critic_for_reviews and num_user_for_reviews.

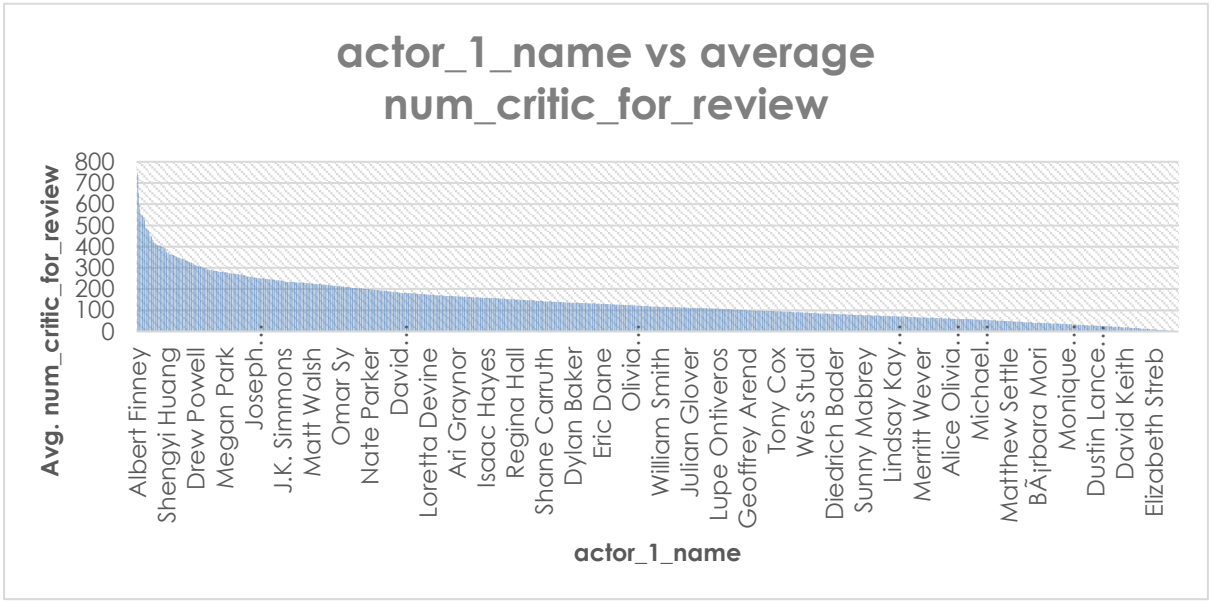


Fig. 10: actor_1_name vs avg, num_critic_for_review.

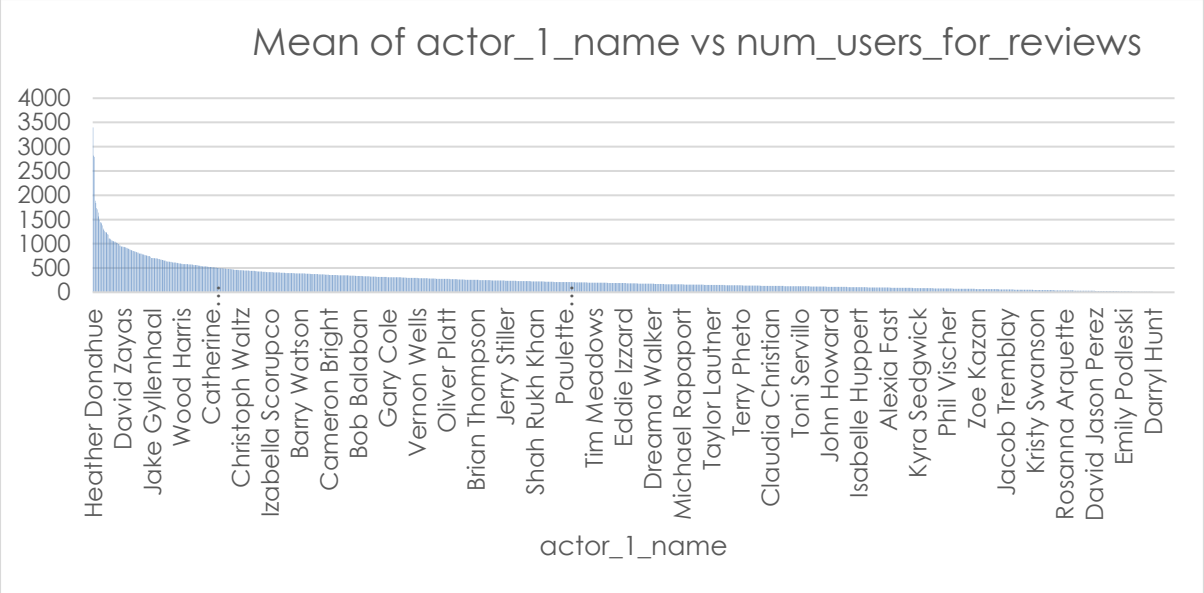


Fig. 11: actor_1_name vs num_critic_for_reviews

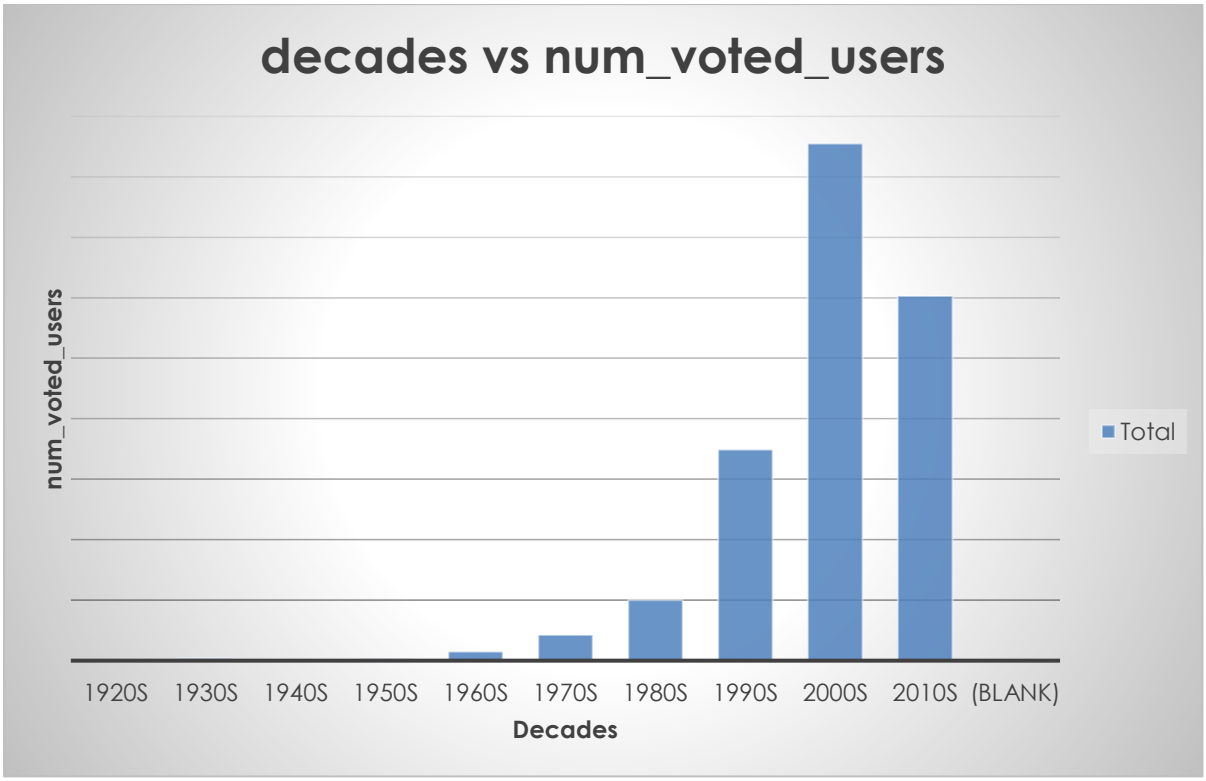


Fig. 12: Decades vs num_voted_users

Module 6: Bank Loan Case Study

Description:

Bank earns their profits by providing loans to the customer. But whenever a person is not able to repay the loan bank suffer from the loss. To avoid this kind of situation bank, try to analyse the past records of the customer which will help bank to understand the chances of a person to repay the loan.

Also, some of the people who be the defaulter apply for the loan again which will cost the bank. Analysing the defaulter in bank is a major work of risk analyst. In this case study I will try to find out the relation between the defaulter and the different features of the data set, so that bank can earn profit.

The loss to the bank can be in two ways:

1. If a person is not likely to repay the loan, then accept ion his loan request will lead to the loss of the bank.
2. If the person is likely to repay the loan but now approving the loan will lead to the loss of the bank.

When a client applies for a loan, there are four types of decisions that could be taken by the client/company:

1. **Approved:** The company has approved loan application
2. **Cancelled:** The client cancelled the application sometime during approval. Either the client changed her/his mind about the loan or in some cases due to a higher risk of the client he received worse pricing which he did not want.
3. **Refused:** The company had rejected the loan (because the client does not meet their requirements etc.).
4. **Unused Offer:** Loan has been cancelled by the client but on different stages of the process.

Approach:

First understanding both the dataset and what is the key connection between them. Check for the null values and plotting some scatter plots and bar plots to understanding the distribution. For each of the result the chart is the best approach to present it. So, I created chart for each of the result.

Tech used:

For the project I have used Google Colab. Google Colab in its free version provides 12 GB of working ram and 90 GB of storage to load and work with the dataset.

Insights:

This project helps me to understand the power of different platforms. I understood that for the large data set where the total number of data point higher than 10 Lac it is better to use Python. Also, with this project I learned how to deal with a lot of features in a data set.

I have also find worked out on the following problems:

- Present the overall approach of the analysis. Mention the problem statement and the analysis approach briefly
- Identify the missing data and use appropriate method to deal with it. (Remove columns/or replace it with an appropriate value) Hint: Note that in EDA, since it is not necessary to replace the missing value, but if you have to replace the missing value, what should be the approach. Clearly mention the approach.
- Identify if there are outliers in the dataset. Also, mention why do you think it is an outlier. Again, remember that for this exercise, it is not necessary to remove any data points.
- Identify if there is data imbalance in the data. Find the ratio of data imbalance. Hint: Since there are a lot of columns, you can run your analysis in loops for the appropriate columns and find the insights.
- Explain the results of univariate, segmented univariate, bivariate analysis, etc. in business terms.
- Find the top 10 correlation for the Client with payment difficulties and all other cases (Target variable). Note that you have to find the top correlation by segmenting the data frame w.r.t to the target variable and then find the top correlation for each of the segmented data and find if any insight is there. Say, there are 5+1(target) variables in a dataset: Var1, Var2, Var3, Var4, Var5, Target. And if you have to find top 3 correlation, it can be: Var1 & Var2, Var2 & Var3, Var1 & Var3. Target variable will not feature in this correlation as it is a categorical variable and not a continuous variable which is increasing or decreasing.
- Include visualizations and summarize the most important results in the presentation. You are free to choose the graphs which explain the numerical/categorical variables. Insights

should explain why the variable is important for differentiating the clients with payment difficulties with all other cases.

Results:

1) Importing libraries:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme(style="whitegrid")
import numpy as np
```

2) Creating function to find the null values in ascending order

```
def count_null_values(df,per=0):
  length = len(df)
  df1 = df.isnull().sum()*100/length
  df2 = df1[df1>=per]
  return df2.sort_values(ascending=False)
```

3) Reading the First dataset (application.csv):

```
# application_data.csv
df1 = pd.read_csv('/content/drive/MyDrive/Trainity/Loan Case Study/application_data.csv')
df1.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0
3	100006	0	Cash loans	F	N	Y	0	135000.0	312682.5
4	100007	0	Cash loans	M	N	Y	0	121500.0	513000.0

5 rows × 122 columns

4) Find the description of the dataset:

```
df1.describe()
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	REGION_POPULATION_RELATIVE
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307499.000000	3.072330e+05	307511.000000
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27108.573909	5.383962e+05	0.020868
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14493.737315	3.694465e+05	0.013831
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1615.500000	4.050000e+04	0.000290
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16524.000000	2.385000e+05	0.010006
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24903.000000	4.500000e+05	0.018850
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.000000	6.795000e+05	0.028663
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06	258025.500000	4.050000e+06	0.072508

8 rows × 106 columns

5) Dealing with the null values:

```
# Check for the null values greater than 50% in each column in df1
null50 = count_null_values(df1,50)
```

```
null50
```

```
['OWN_CAR_AGE',
 'EXT_SOURCE_1',
 'APARTMENTS_AVG',
 'BASEMENTAREA_AVG',
 'YEARS_BUILD_AVG',
 'COMMONAREA_AVG',
 'ELEVATORS_AVG',
 'ENTRANCES_AVG',
 'FLOORSMIN_AVG',
 'LANDAREA_AVG',
 'LIVINGAPARTMENTS_AVG',
 'LIVINGAREA_AVG',
 'NONLIVINGAPARTMENTS_AVG',
 'NONLIVINGAREA_AVG',
 'APARTMENTS_MODE',
 'BASEMENTAREA_MODE',
 'YEARS_BUILD_MODE',
 'COMMONAREA_MODE',
 'ELEVATORS_MODE',
 'ENTRANCES_MODE',
 'FLOORSMIN_MODE',
 'LANDAREA_MODE',
 'LIVINGAPARTMENTS_MODE',
 'LIVINGAREA_MODE',
 'NONLIVINGAPARTMENTS_MODE',
 'NONLIVINGAREA_MODE',
 'APARTMENTS_MEDI',
 'BASEMENTAREA_MEDI',
 'YEARS_BUILD_MEDI',
 'COMMONAREA_MEDI',
 'ELEVATORS_MEDI',
 'ENTRANCES_MEDI',
 'FLOORSMIN_MEDI',
 'LANDAREA_MEDI',
 'LIVINGAPARTMENTS_MEDI',
 'LIVINGAREA_MEDI',
 'NONLIVINGAPARTMENTS_MEDI',
 'NONLIVINGAREA_MEDI',
 'FONDKAPREMONT_MODE',
 'HOUSETYPE_MODE',
 'WALLSMATERIAL_MODE']
```

Inference:

The null columns contain the values most related to the apartment or house, and one column contain data for car age, and one normalized score for external data source. So that can be removed because they cannot be useful for the target.

 # drop values with more than 50% null values

```
df1.drop(null50.index,axis=1,inplace=True)
df1.shape
```

```
(307511, 81)
```

 # Check for the null values greater than 15% in each column in df1

```
print('Percentage (%) null values in each column')
null15 = count_null_values(df1,15)
```

```
Percentage (%) null values in each column
FLOORSMAX_AVG                49.760822
FLOORSMAX_MODE               49.760822
FLOORSMAX_MEDI               49.760822
YEARS_BEGINEXPLUATATION_AVG  48.781019
YEARS_BEGINEXPLUATATION_MODE 48.781019
YEARS_BEGINEXPLUATATION_MEDI 48.781019
TOTALAREA_MODE               48.268517
EMERGENCYSTATE_MODE          47.398304
OCCUPATION_TYPE              31.345545
EXT_SOURCE_3                 19.825307
DAYS_EMPLOYED                18.007161
dtype: float64
```

Inference:

In this the except occupation_type and ext_souorce_3 looks familiar to target null values and those are related to the building. So that can be removed.

 # Remove those columns and drop remaining columns

```
columns_with_null = dict(null15)
del columns_with_null['EXT_SOURCE_3']
del columns_with_null['OCCUPATION_TYPE']
df1.shape
```

```
(307511, 72)
```

6) Feature selection and remove unnecessary columns

 # Treating the columns with the null values

```
null_count = count_null_values(df1)
null_count
```

```
OCCUPATION_TYPE 31.345545
EXT_SOURCE_3 19.825307
AMT_REQ_CREDIT_BUREAU_DAY 13.501631
AMT_REQ_CREDIT_BUREAU_HOUR 13.501631
AMT_REQ_CREDIT_BUREAU_YEAR 13.501631
 ...
REG_REGION_NOT_WORK_REGION 0.000000
LIVE_REGION_NOT_WORK_REGION 0.000000
REG_CITY_NOT_LIVE_CITY 0.000000
TARGET 0.000000
```

```
REG_CITY_NOT_WORK_CITY 0.000000
Length: 72, dtype: float64
```

 # check correlation for the columns EXT_SOURCE_3 and EXT_SOURCE_2 to the target column

```
sns.heatmap(df1[['EXT_SOURCE_3','EXT_SOURCE_2','TARGET']].corr(),annot=True)
```

Inference:

As it is clearly seen that the both columns do not show a good correlation with the Target. So, it can be removed.

```
df1.drop(['EXT_SOURCE_3','EXT_SOURCE_2'],axis=1,inplace=True)
df1.shape
```

```
(307511, 70)
```

 # Now there are some columns named *flag* which contain some true false values

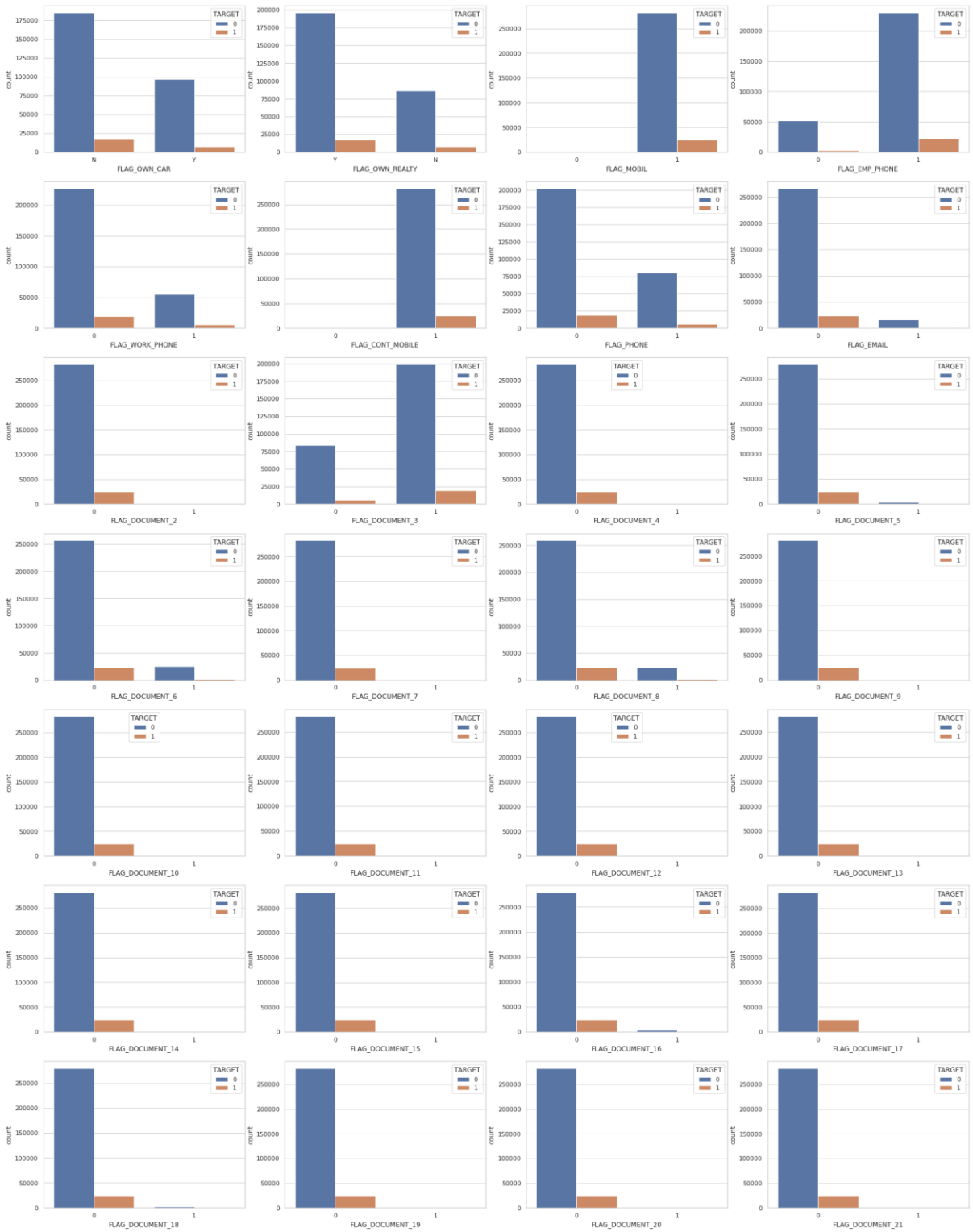
```
flags = []
for i in df1.columns:
  if 'FLAG' in i:
    flags.append(i)
flags
```

['FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'FLAG_MOBIL', 'FLAG_EMP_PHONE',
'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE', 'FLAG_PHONE', 'FLAG_EMAIL',
'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3', 'FLAG_DOCUMENT_4', 'FLAG_DOCUMENT_5',
'FLAG_DOCUMENT_6', 'FLAG_DOCUMENT_7', 'FLAG_DOCUMENT_8', 'FLAG_DOCUMENT_9',
'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_DOCUMENT_12',
'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15',
'FLAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18',
'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21']

 # Plot each flag column vs target column

 # Plot will be of 28 plots with hue = Y/N

```
plt.figure(figsize=(30,40))
for i in range(len(flag_df.columns)-1):
  plt.subplot(7,4,i+1)
  sns.countplot(x=flag_df[flag_df.columns[i]],hue=flag_df[flag_df.columns[-1]])
```



Inference:

All the columns which have sufficient values in both 0 and 1 will be kept, except removed columns to kept FLAG_OWN_CAR, FLAG_OWN_REALTY, FLAG_PHONE, FLAG_WORK_PHONE.

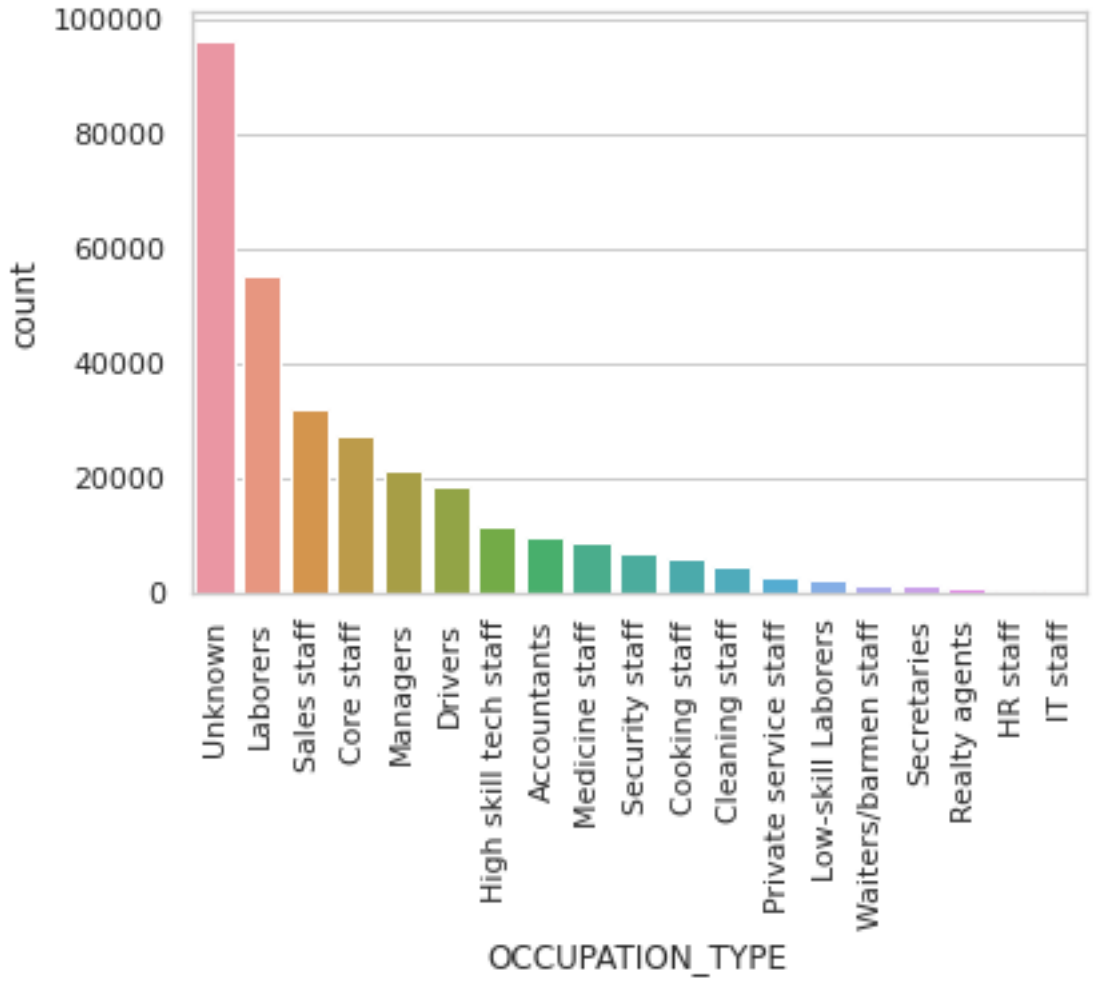
 # Drop unnecessary flag columns:

```
remove_list = ['FLAG_OWN_REALTY','FLAG_OWN_CAR','FLAG_PHONE','FLAG_WORK_PHONE'
]
flags = list(set(flags) - set(remove_list))
# finally remove those columns
df1.drop(flags,axis=1,inplace=True)
# Dataset after removing unnecessary flag values
df1.shape
```
(307511, 46)

7) Filling null values

 # Fill the null values in OCCUPATION_TYPE with the 'Unknown'

```
df1['OCCUPATION_TYPE'].fillna('Unknown',inplace=True)
sns.countplot(x =df1['OCCUPATION_TYPE'],order = df1['OCCUPATION_TYPE'].value_counts().index
)
plt.xticks(rotation=90)
```



 # Filling values of the column name starts with amt_req

```
amt_req = []
for i in df1.columns:
  if "AMT_REQ" in i:
```

```
    amt_req.append(i)
amt_req
```

['AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR']

Inference:

The columns represent the Number of enquiries to Credit Bureau about the client before different time periods. these values should be integer.

Mean

```
df1[amt_req].mean()
```

AMT_REQ_CREDIT_BUREAU_HOUR 0.006402
AMT_REQ_CREDIT_BUREAU_DAY 0.007000
AMT_REQ_CREDIT_BUREAU_WEEK 0.034362
AMT_REQ_CREDIT_BUREAU_MON 0.267395
AMT_REQ_CREDIT_BUREAU_QRT 0.265474
AMT_REQ_CREDIT_BUREAU_YEAR 1.899974
dtype: float64

Mode

```
df1[amt_req].mode()
```

	AMT_REQ_CREDIT_BUREAU_HOUR	AMT_REQ_CREDIT_BUREAU_DAY	AMT_REQ_CREDIT_BUREAU_WEEK	AMT_REQ_CREDIT_BUREAU_MON	AMT_R
0	0.0	0.0	0.0	0.0	

Median

```
df1[amt_req].median()
```

AMT_REQ_CREDIT_BUREAU_HOUR 0.0
AMT_REQ_CREDIT_BUREAU_DAY 0.0
AMT_REQ_CREDIT_BUREAU_WEEK 0.0
AMT_REQ_CREDIT_BUREAU_MON 0.0
AMT_REQ_CREDIT_BUREAU_QRT 0.0
AMT_REQ_CREDIT_BUREAU_YEAR 1.0
dtype: float64

Inference:

For mean the values are real numbers so it cannot bs used, most values in all the columns are 0 but for the column AMT_REQ_CREDIT_BUREAU_YEAR occurrence of all the values is quite comparable so mode will not a good approach for all the columns, but for median it is giving feasible values. So, I will replace all the values by median.

 # Filling the null values by Median

```
try1 = df1[amt_req].fillna(df1[amt_req].median())
```

 # For the column NAME_TYPE_SUITE I will replace the null values by Unknown

```
# replace null by Unknown
df1['NAME_TYPE_SUITE'].fillna('Unknown',inplace=True)
```

Fill the null values for the column social circle

```
social_circle = []
for i in df1.columns:
  if 'SOCIAL_CIRCLE' in i:
    social_circle.append(i)
social_circle
```

['OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',

'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE']

Mean

```
df1[social_circle].mean()
```

OBS_30_CNT_SOCIAL_CIRCLE 1.422245

DEF_30_CNT_SOCIAL_CIRCLE 0.143421

OBS_60_CNT_SOCIAL_CIRCLE 1.405292

DEF_60_CNT_SOCIAL_CIRCLE 0.100049

dtype: float64

Mode

```
df1[social_circle].mode()
```

	OBS_30_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	OBS_60_CNT_SOCIAL_CIRCLE	DEF_60_CNT_SOCIAL_CIRCLE
0	0.0	0.0	0.0	0.0

```
df1[social_circle].median()
```

OBS_30_CNT_SOCIAL_CIRCLE 0.0

DEF_30_CNT_SOCIAL_CIRCLE 0.0

OBS_60_CNT_SOCIAL_CIRCLE 0.0

DEF_60_CNT_SOCIAL_CIRCLE 0.0

dtype: float64

Inference:

This column represents how many observations of client's social surroundings day past due, which is integer. So, I will replace it by median.

```
# Filling values by median
df1[social_circle] = df1[social_circle].fillna(df1[social_circle].median())
```

```
# Fill the null values for AMT_GOODS_PRICE with the mean value
df1['AMT_GOODS_PRICE'].fillna(df1['AMT_GOODS_PRICE'].mean(),inplace=True)
```

```
# Fill the null values for AMT_ANNUITY with the mean value
df1['AMT_ANNUITY'].fillna(df1['AMT_ANNUITY'].mean(),inplace=True)
```

```
# Filling CNT_FAM_MEMBERS with the median
df1['CNT_FAM_MEMBERS'].fillna(df1['CNT_FAM_MEMBERS'].median(),inplace=True)
```

```
# Filling CNT_FAM_MEMBERS with the median
df1['DAYS_LAST_PHONE_CHANGE'].fillna(df1['DAYS_LAST_PHONE_CHANGE'].median(),inplace=True)
```

```
# find the columns with days
```

```
days = []
for i in df1.columns:
  if 'DAYS' in i:
    days.append(i)
days
```

['DAYS_BIRTH', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH',

'DAYS_LAST_PHONE_CHANGE']

```
#There are 5 columns with the days change it in years
```

```
# change days to years
df1[days] = abs(df1[days])/365
# Rename the columns
df1.rename(columns = {'DAYS_BIRTH':'AGE','DAYS_EMPLOYED':'YEARS_EMPLOYED','DAYS_REGISTRATION':'YEARS_REGISTRATION','DAYS_ID_PUBLISH':'YEARS_ID_PUBLISH','DAYS_LAST_PHONE_CHANGE':'YEARS_LAST_PHONE_CHANGE'},inplace=True)
```

8) Reading the dataset previous_applications.csv

```
df = pd.read_csv('/content/drive/MyDrive/Trainity/Loan Case Study/previous_application.csv')
df.info()
```

```
 0   SK_ID_PREV                   1670214 non-null  int64
 1   SK_ID_CURR                   1670214 non-null  int64
 2   NAME_CONTRACT_TYPE           1670214 non-null  object
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_DOWN_PAYMENT             774370 non-null   float64
 7   AMT_GOODS_PRICE              1284699 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object
 9   HOUR_APPR_PROCESS_START      1670214 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64
 12  RATE_DOWN_PAYMENT            774370 non-null   float64
 13  RATE_INTEREST_PRIMARY        5951 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5951 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1670214 non-null  object
 16  NAME_CONTRACT_STATUS         1670214 non-null  object
 17  DAYS_DECISION                1670214 non-null  int64
 18  NAME_PAYMENT_TYPE            1670214 non-null  object
 19  CODE_REJECT_REASON           1670214 non-null  object
 20  NAME_TYPE_SUITE              849809 non-null   object
 21  NAME_CLIENT_TYPE             1670214 non-null  object
 22  NAME_GOODS_CATEGORY          1670214 non-null  object
 23  NAME_PORTFOLIO               1670214 non-null  object
 24  NAME_PRODUCT_TYPE            1670214 non-null  object
 25  CHANNEL_TYPE                 1670214 non-null  object
 26  SELLERPLACE_AREA             1670214 non-null  int64
 27  NAME_SELLER_INDUSTRY         1670214 non-null  object
 28  CNT_PAYMENT                  1297984 non-null  float64
```

Description of the data set

```
df.describe()
```

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_PRICE	HOUR_APPR_PROCESS_START
count	1.670214e+06	1.670214e+06	1.297979e+06	1.670214e+06	1.670213e+06	7.743700e+05	1.284699e+06	1.670214e+06
mean	1.923089e+06	2.783572e+05	1.595512e+04	1.752339e+05	1.961140e+05	6.697402e+03	2.278473e+05	1.248418e+01
std	5.325980e+05	1.028148e+05	1.478214e+04	2.927798e+05	3.185746e+05	2.092150e+04	3.153966e+05	3.334028e+00
min	1.000001e+06	1.000010e+05	0.000000e+00	0.000000e+00	0.000000e+00	-9.000000e-01	0.000000e+00	0.000000e+00
25%	1.461857e+06	1.893290e+05	6.321780e+03	1.872000e+04	2.416050e+04	0.000000e+00	5.084100e+04	1.000000e+01
50%	1.923110e+06	2.787145e+05	1.125000e+04	7.104600e+04	8.054100e+04	1.638000e+03	1.123200e+05	1.200000e+01
75%	2.384280e+06	3.675140e+05	2.065842e+04	1.803600e+05	2.164185e+05	7.740000e+03	2.340000e+05	1.500000e+01
max	2.845382e+06	4.562550e+05	4.180581e+05	6.905160e+06	6.905160e+06	3.060045e+06	6.905160e+06	2.300000e+01

8 rows × 21 columns

9) Dealing with null values

Count null values

```
count_null_values(df)
```

```
RATE_INTEREST_PRIVILEGED       99.643698
RATE_INTEREST_PRIMARY          99.643698
AMT_DOWN_PAYMENT               53.636480
RATE_DOWN_PAYMENT              53.636480
NAME_TYPE_SUITE                49.119754
NFLAG_INSURED_ON_APPROVAL      40.298129
DAYS_TERMINATION               40.298129
DAYS_LAST_DUE                  40.298129
DAYS_LAST_DUE_1ST_VERSION      40.298129
DAYS_FIRST_DUE                 40.298129
DAYS_FIRST_DRAWING             40.298129
AMT_GOODS_PRICE                23.081773
AMT_ANNUITY                    22.286665
CNT_PAYMENT                    22.286366
PRODUCT_COMBINATION             0.020716
AMT_CREDIT                      0.000060
NAME_YIELD_GROUP                0.000000
NAME_PORTFOLIO                  0.000000
NAME_SELLER_INDUSTRY            0.000000
SELLERPLACE_AREA                0.000000
CHANNEL_TYPE                    0.000000
NAME_PRODUCT_TYPE               0.000000
SK_ID_PREV                      0.000000
NAME_GOODS_CATEGORY             0.000000
NAME_CLIENT_TYPE                0.000000
CODE_REJECT_REASON              0.000000
SK_ID_CURR                      0.000000
DAYS_DECISION                   0.000000
NAME_CONTRACT_STATUS            0.000000
NAME_CASH_LOAN_PURPOSE          0.000000
NFLAG_LAST_APPL_IN_DAY          0.000000
FLAG_LAST_APPL_PER_CONTRACT     0.000000
HOUR_APPR_PROCESS_START         0.000000
WEEKDAY_APPR_PROCESS_START      0.000000
AMT_APPLICATION                 0.000000
NAME_CONTRACT_TYPE              0.000000
NAME_PAYMENT_TYPE               0.000000
dtype: float64
```

```
# Shape of the data set
df.shape
```

```
(1670214, 37)
```

Find the columns with more than 40% of null values

```
# find columns with more than 40% of null values
#print('Percentage (%) null values in each column')
null40 = count_null_values(df,40)
# this columns will be removed because of greater than 50% of null values
null40
```

```
RATE_INTEREST_PRIMARY 99.643698

RATE_INTEREST_PRIVILEGED 99.643698

AMT_DOWN_PAYMENT 53.636480

RATE_DOWN_PAYMENT 53.636480
```

NAME_TYPE_SUITE 49.119754

DAYS_FIRST_DRAWING 40.298129

DAYS_FIRST_DUE 40.298129

DAYS_LAST_DUE_1ST_VERSION 40.298129

DAYS_LAST_DUE 40.298129

DAYS_TERMINATION 40.298129

NFLAG_INSURED_ON_APPROVAL 40.298129

dtype: float64

Dop all the values with more than 40% of null values

```
df.drop(null40,axis=1,inplace=True)
df.shape
```

(1670214, 22)

Remove the columns which are unnecessary

```
df.drop(['WEEKDAY_APPR_PROCESS_START','HOUR_APPR_PROCESS_START','FLAG_LAST_APPL_PER_CONTRACT','NFLAG_LAST_APPL_IN_DAY'],axis=1,inplace=True)
df.shape
```

(1670214, 22)

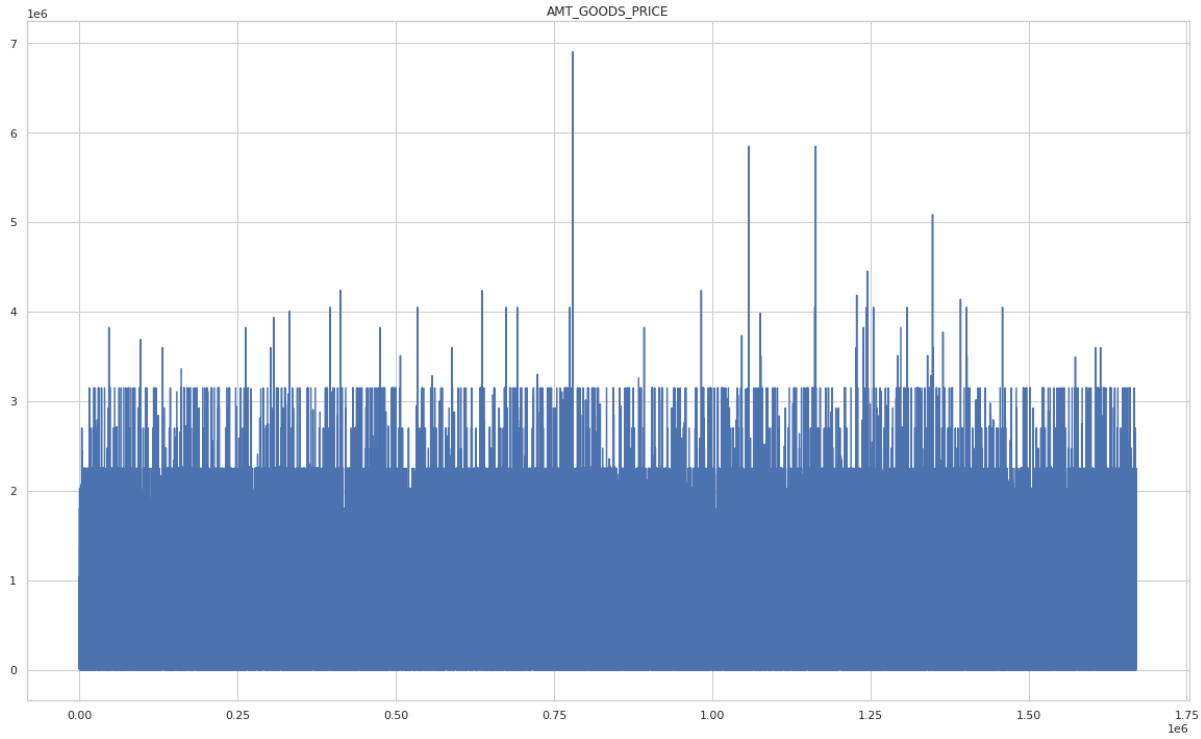
Now find the columns with missing values more than 15%

```
null15 = count_null_values(df,15)
# 10 columns with null values greater than 15%
null15
```

['AMT_ANNUITY', 'AMT_GOODS_PRICE', 'CNT_PAYMENT']

Plot goods price to check distributions

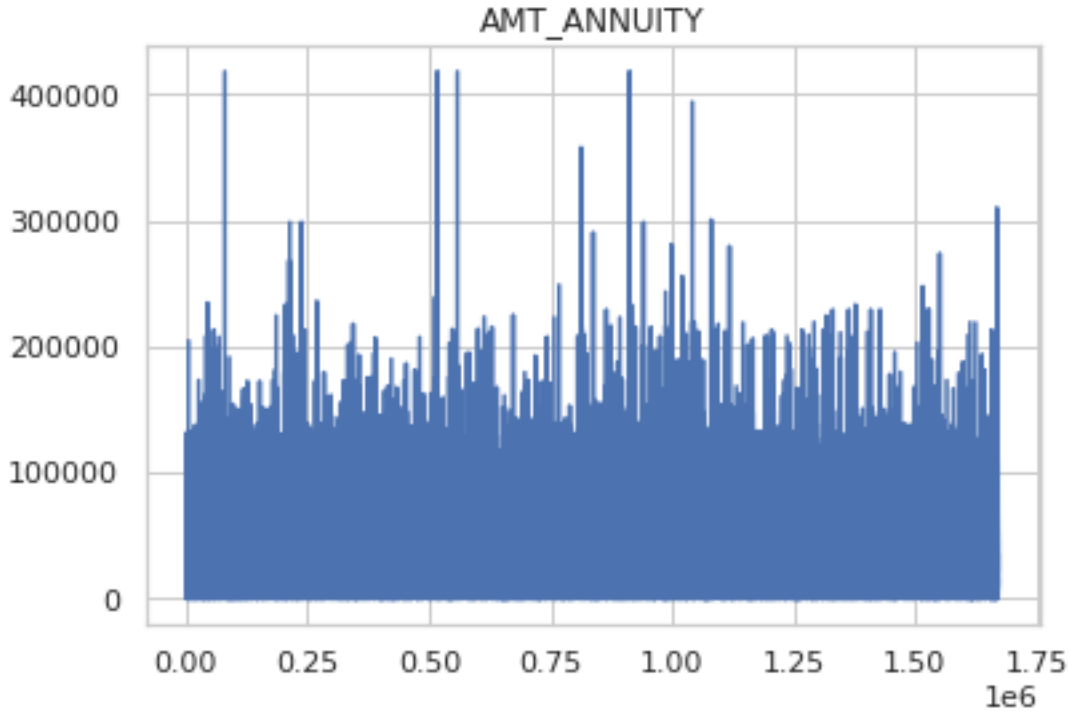
```
plt.figure(figsize=(20,12))
plt.plot(df['AMT_GOODS_PRICE'])
```




```
# Replace null in AMT_GOODS_PRICE by its mean as mean shows good replacement by the plot
df['AMT_GOODS_PRICE'].fillna(df['AMT_GOODS_PRICE'].mean(),inplace=True)
```

Plot amt annuity

```
plt.plot(df['AMT_ANNUITY'])
plt.title('AMT_ANNUITY')
```



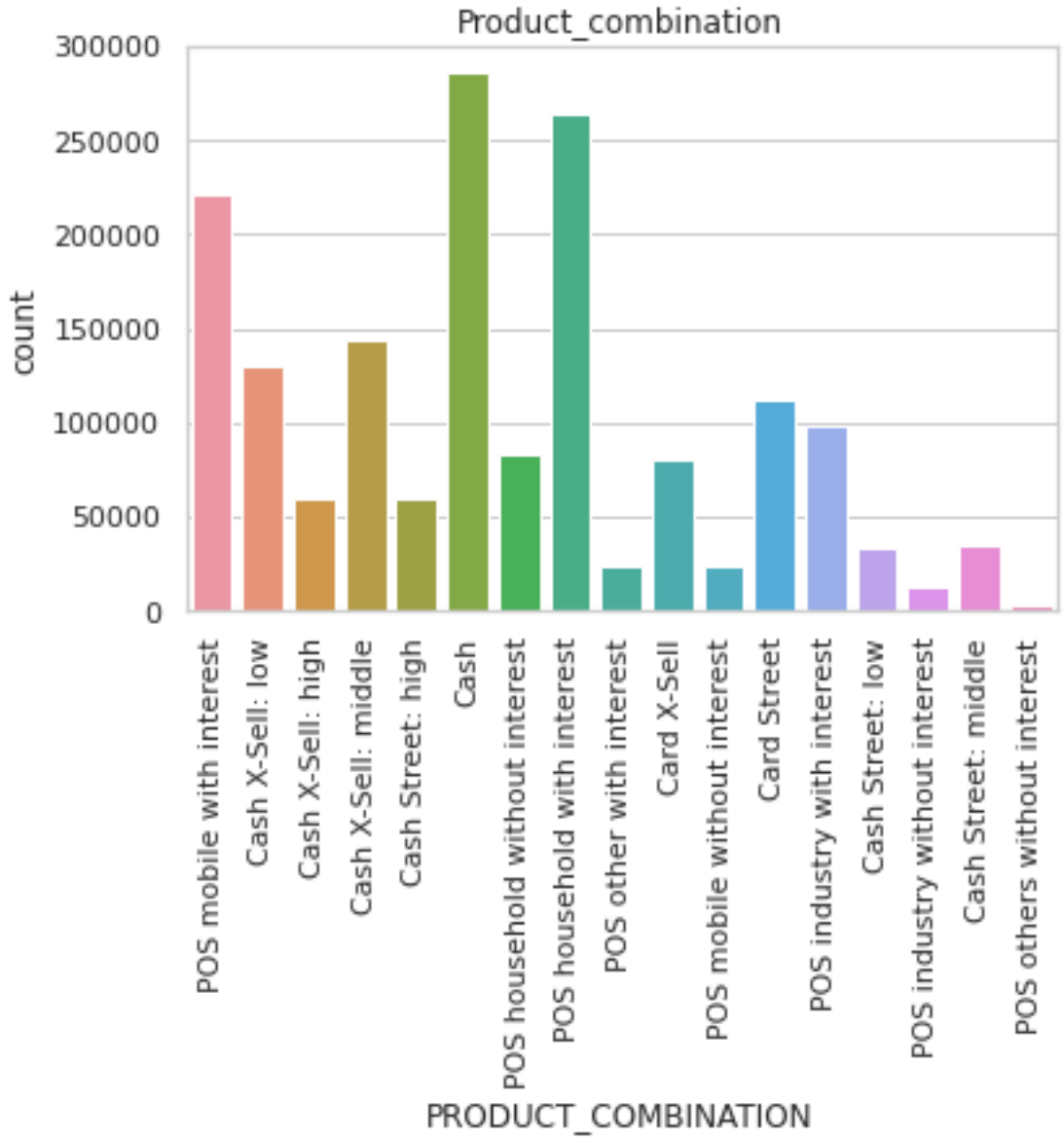

```
# replace null values in ANT_ANNUITY by the mean
```

```
df['AMT_ANNUITY'].fillna(df['AMT_ANNUITY'].mean(),inplace=True)
```

```
# replace null values in CNT_PAYMENT by its median
df['CNT_PAYMENT'].fillna(df['CNT_PAYMENT'].median(),inplace=True)
```

Plot product combination

```
sns.countplot(df['PRODUCT_COMBINATION'])
plt.xticks(rotation=90)
plt.title('Product  combination')
```




```
# replace null with the Unknown in PRODUCT_COMBINATION
df['PRODUCT_COMBINATION'].fillna('Unknown',inplace=True)
```

```
# Replace null in AMT_CREDIT by its mean
df['AMT_CREDIT'].fillna(df['AMT_CREDIT'].mean(),inplace=True)
# convert days to years for DAYS_DECISION column
```

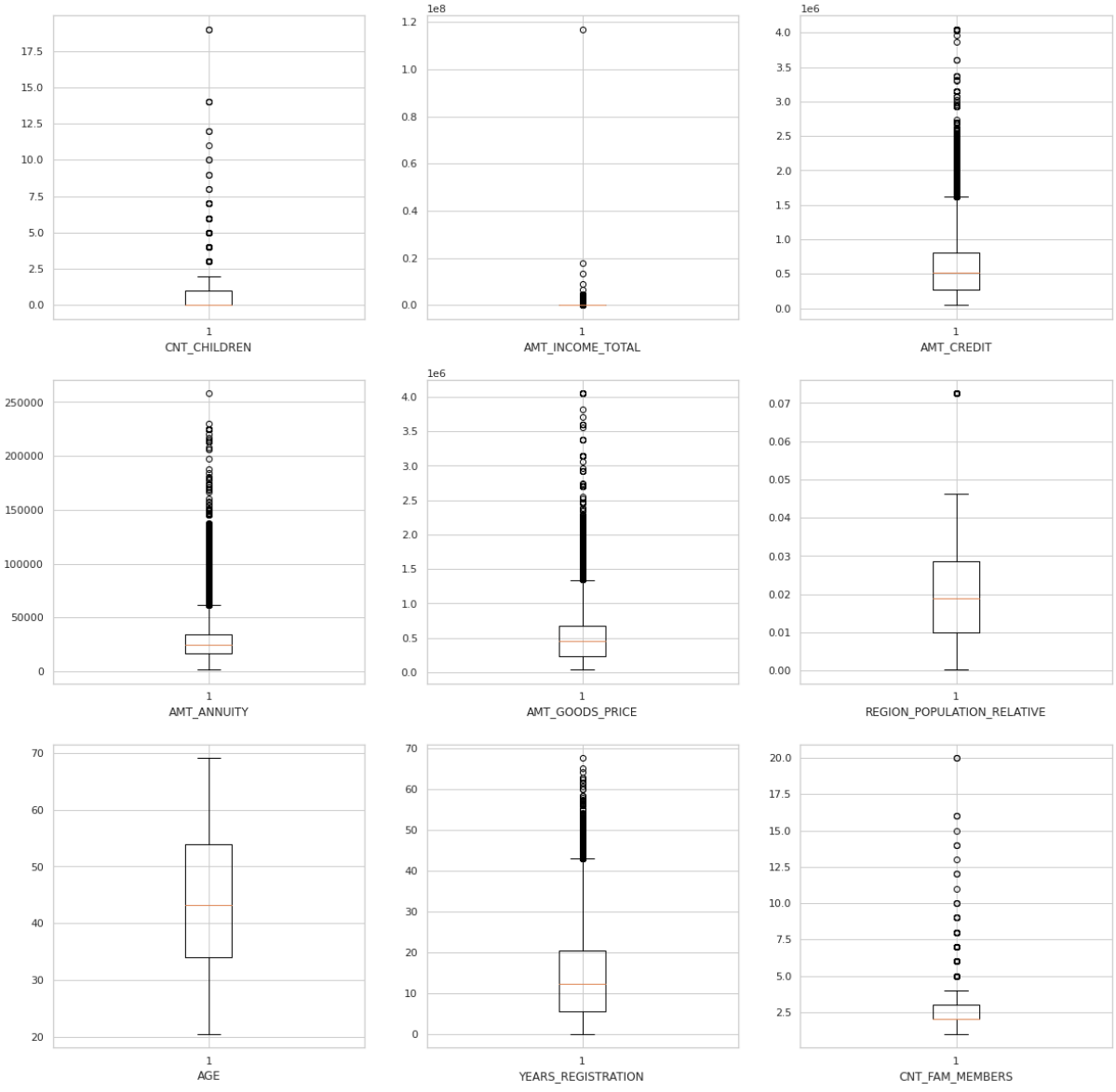
```
df['DAYS_DECISION'] = abs(df['DAYS_DECISION'])/365
```

10) Check Outliers

1) application_data.csv

Find outliers

```
out_check = 'CNT_CHILDREN,AMT_INCOME_TOTAL,AMT_CREDIT,AMT_ANNUITY,AMT_GOODS_PRICE,REGION_POPULATION_RELATIVE,AGE,YEARS_REGISTRATION,CNT_FAM_MEMBERS'.split(',')
plt.figure(figsize=(20,20))
#df1.boxplot(column = out_check, grid=False, rot=90, fontsize=15)
for i in range(len(out_check)):
  plt.subplot(3,3,i+1)
  plt.boxplot(df1[out_check[i]])
  plt.xlabel(out_check[i])
```



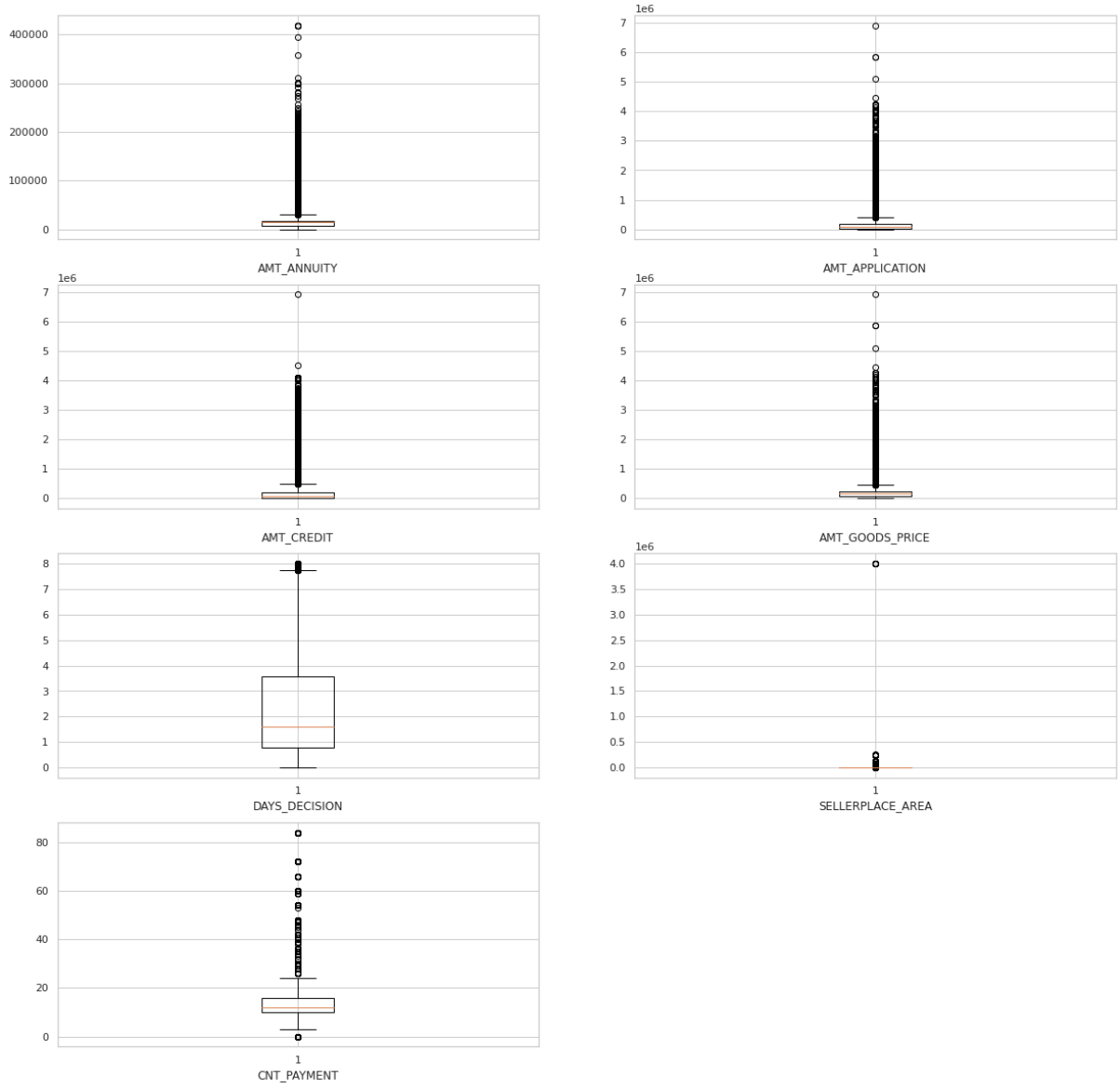
Inference:

There are outliers in each of the column except AGE. In the REGION_POPULATION_RELATIVE there are few outliers.

2) *Previous_application.csv*

Plot for outlier's check

```
out_check2 = 'AMT_ANNUITY,AMT_APPLICATION,AMT_CREDIT,AMT_GOODS_PRICE,DAYS_DECISION,SELLERPLACE_AREA,CNT_PAYMENT'.split(',')
plt.figure(figsize=(20,20))
#df1.boxplot(column = out_check, grid=False, rot=90, fontsize=15)
for i in range(len(out_check2)):
  plt.subplot(4,2,i+1)
  plt.boxplot(df[out_check2[i]])
  plt.xlabel(out_check2[i])
```



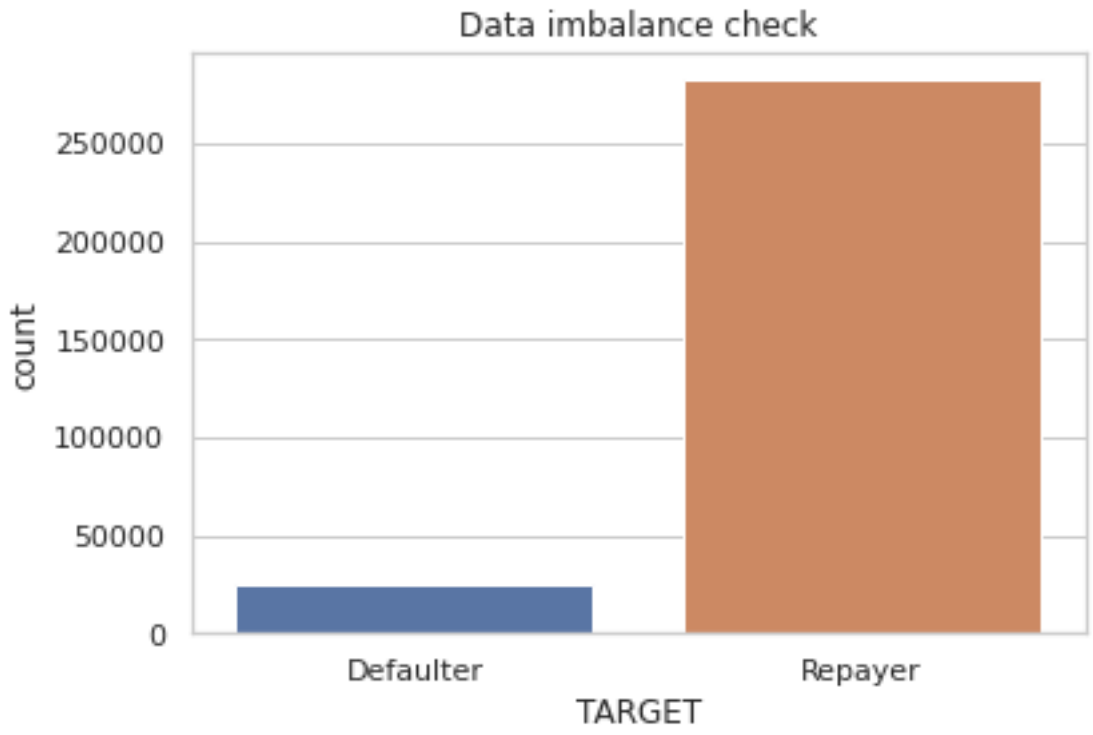
Inference:

Each of the numerical column contain outliers.

11) Data Imbalance

```
# before moving to the data imbalance rename the data values 0 to repayer and 1 to defaulter for better und
erstanding
df1["TARGET"] = df1["TARGET"].replace({1:"Defaulter",0:"Repayer"})
```

 # Plot Target column



 # percentage value count of defaulter and repayer

```
df1['TARGET'].value_counts()*100/len(df1)
```

Repayer 91.927118

Defaulter 8.072882

Name: TARGET, dtype: float64

12) Univariate Analysis for application_data.csv

 # Function to write labels in the bar plot

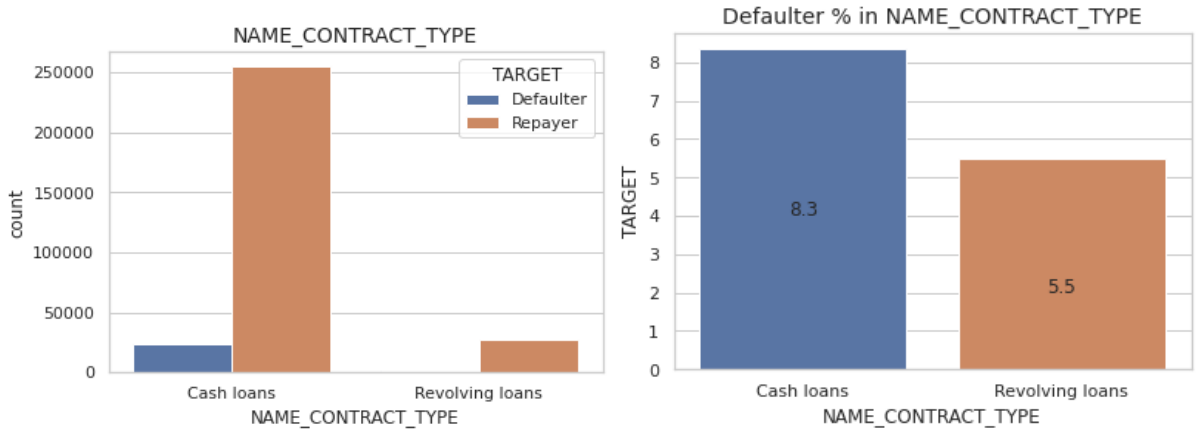
```
def addlabels(x,y):
    for i in range(len(x)):
        plt.text(i, y[i]//2, y[i], ha = 'center')

# function for univariate analysis
```

```
# Cateriogical and numerical
def univariate(df,data,target):
  col = df[data]
  tar = df[target]
  types = col.dtypes

  ex = col.value_counts()
  if len(ex)>8:
    plt.figure(figsize = (20,12))
    sns.countplot(x = col, hue = tar)
    plt.xticks(rotation = 90)
    plt.title(data,fontdict={'fontsize': 20})
  elif len(ex)>4:
    plt.figure(figsize = (15,8))
    sns.countplot(x = col, hue = tar)
    plt.xticks(rotation = 45)
    plt.title(data,fontdict={'fontsize': 20})
  else:
    plt.figure()
    sns.countplot(x = col, hue = tar)
    plt.title(data,fontdict={'fontsize': 14})
  plt.xlabel(data)

  # Calculate defaulter % for the data
  pf = df[[data,target]].value_counts().reset_index()
  percent= []
  for i in pf[data].unique():
    try:
      percent.append(pf[(pf[data]==i)&(pf[target]=='Defaulter')][0].values[0]*100/(pf[(pf[data]==i)&(pf[tar
get]=='Defaulter')][0].values[0]+pf[(pf[data]==i)&(pf[target]=='Repayer')][0].values[0]))
    except:
      percent.append(0)
  if len(percent)>8:
    plt.figure(figsize = (20,12))
    sns.barplot(y = percent, x = pf[data].unique())
    plt.xticks(rotation = 90)
    plt.title('Defaulter % in '+data,fontdict={'fontsize': 20})
  elif len(percent)>4:
    plt.figure(figsize = (15,8))
    sns.barplot(y = percent, x = pf[data].unique())
    plt.xticks(rotation = 45)
    plt.title('Defaulter % in '+data,fontdict={'fontsize': 20})
  else:
    plt.figure()
    sns.barplot(y = percent, x = pf[data].unique())
    plt.title('Defaulter % in '+data,fontdict={'fontsize': 14})
  plt.xlabel(data)
  plt.ylabel(target)
```

```
addlabels(x = pf[data].unique(),y=np.round(percent,1))
```

 # Check for NAME_CONTRACT_TYPE based on loan repay status

```
univariate(df1,'NAME_CONTRACT_TYPE','TARGET')
```

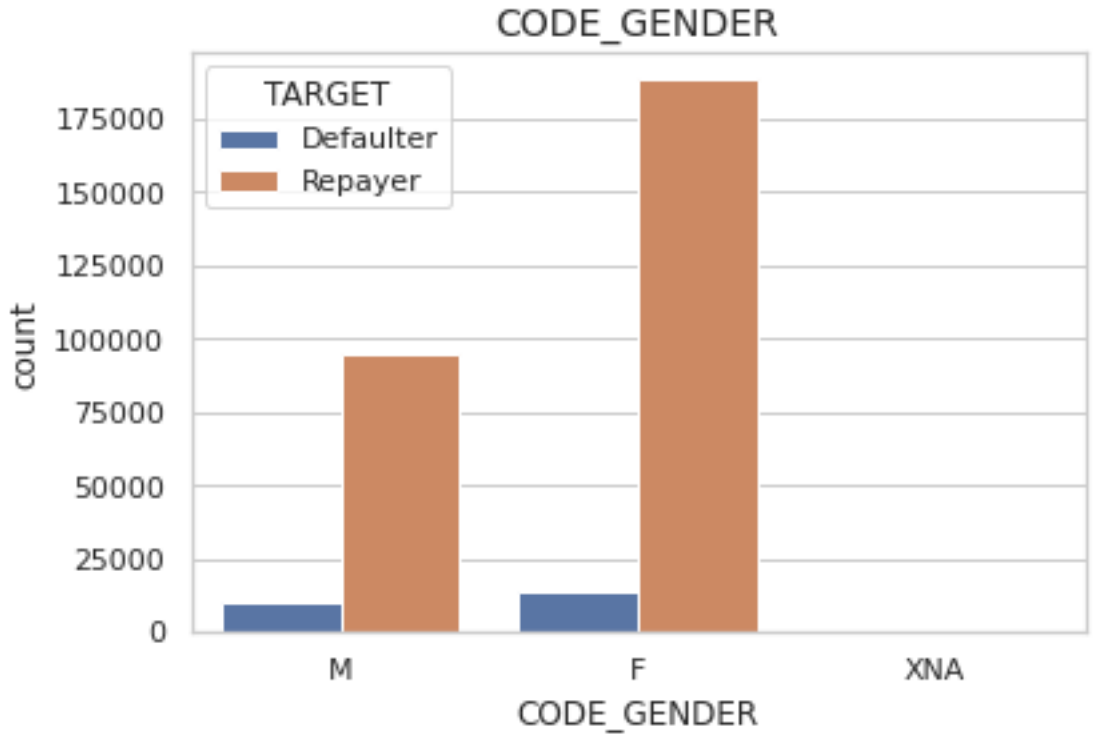


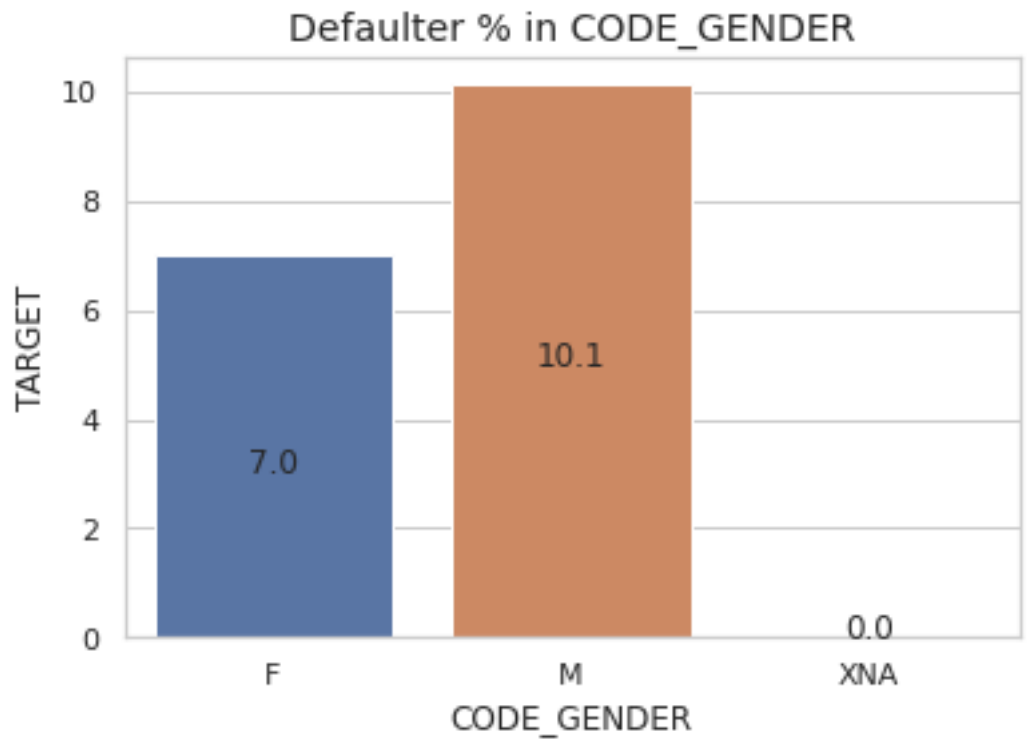
Inference:

- Revolving loans are only 8-12% compared to cash loans.
- Revolving loans have less defaulter%

 # Check for CODE_GENDER based on loan repay status

```
univariate(df1,'CODE_GENDER','TARGET')
```



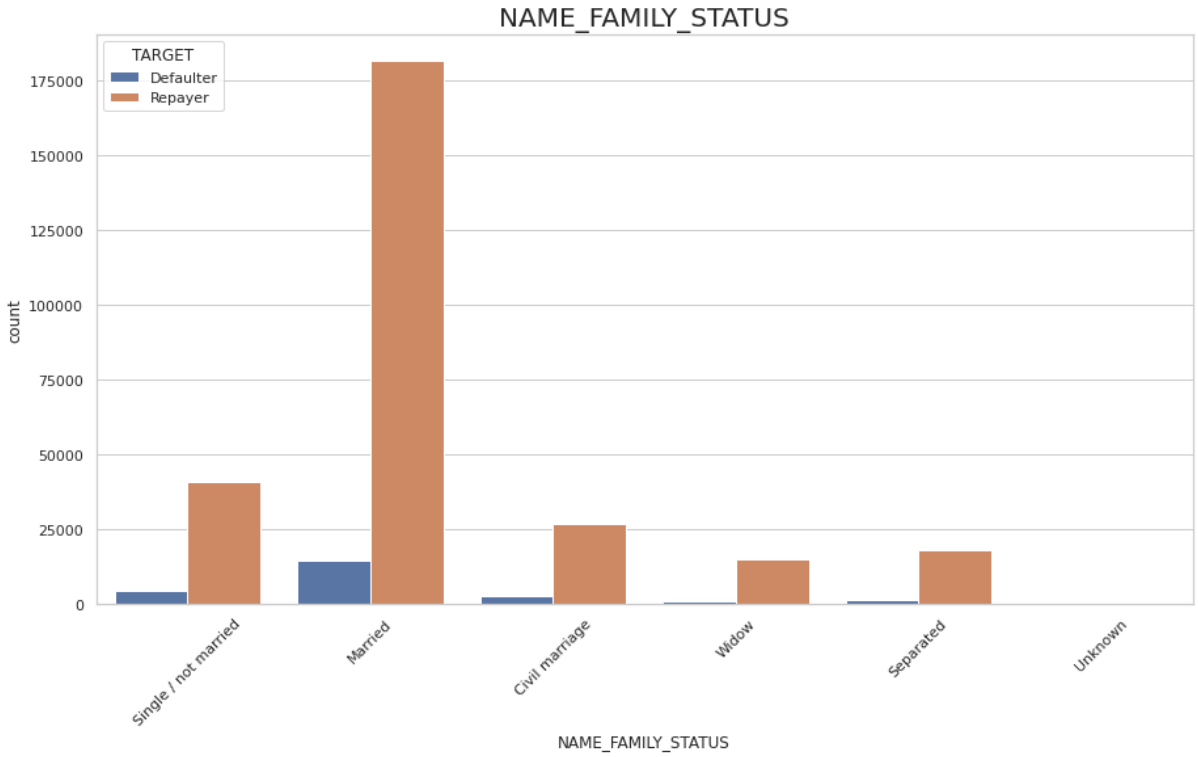


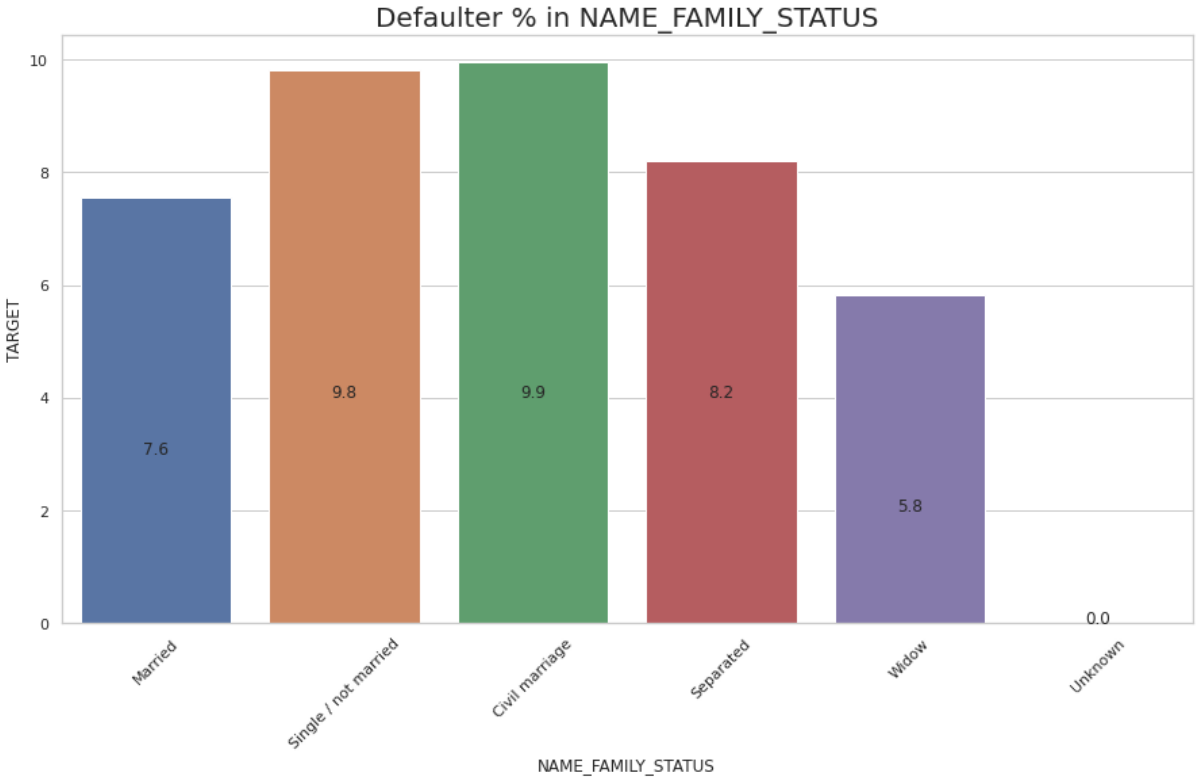
Inference:

- Females are most likely to take loans compared to males almost twice.
- Male have high defaulter than female.
- There are only few cases where people have not shared the gender but they paid all the loan.

 # Check for NAME_FAMILY_STATUS based on loan repay status

```
univariate(df1,'NAME_FAMILY_STATUS','TARGET')
```



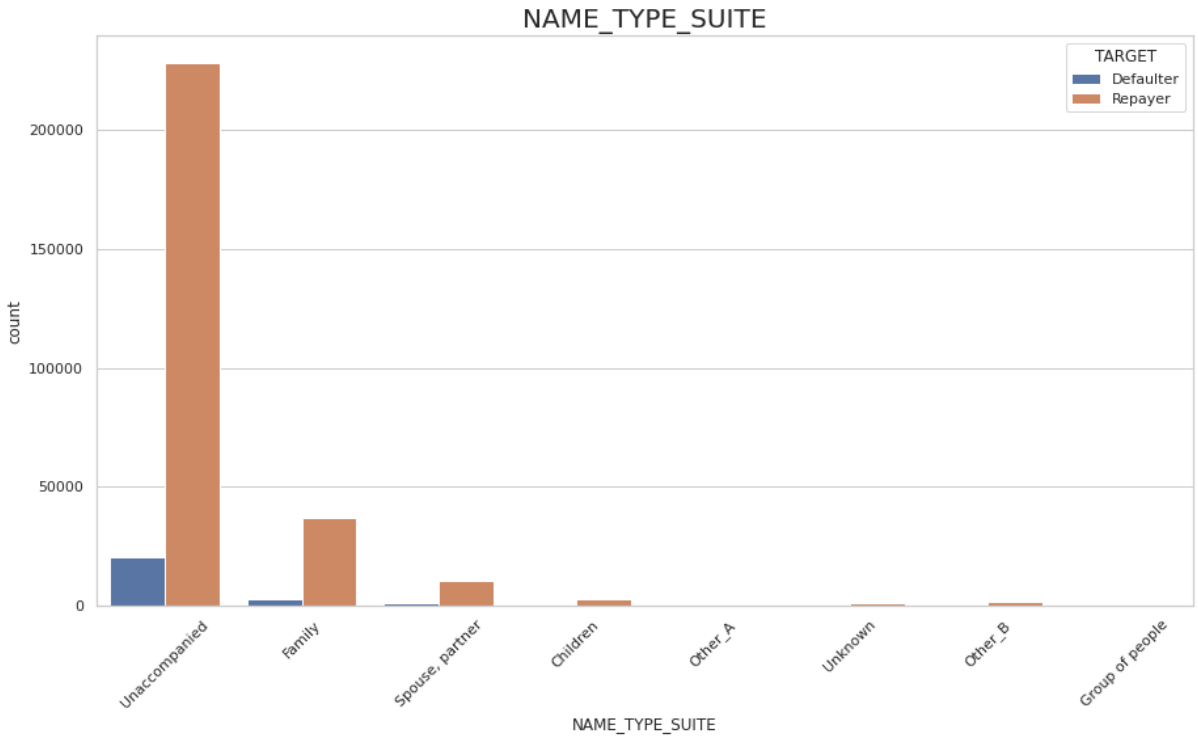


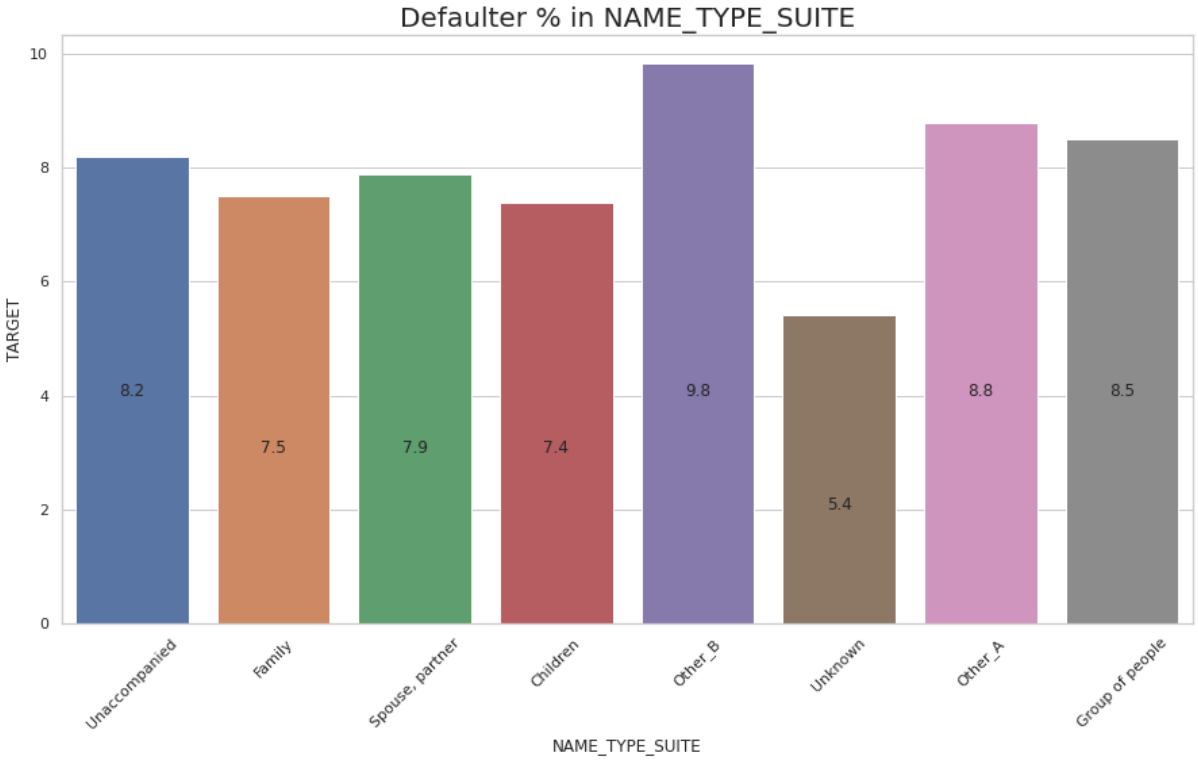
Inference:

- Married Family has taken more loans compared to all the category.
- In all the categories civil marriage have the highest default % followed by single / not married, separated, Married, and widow.
- Widow has the lowest defaulter value.

 # Check for NAME_TYPE_SUITE based on loan repay status

```
univariate(df1,'NAME_TYPE_SUITE','TARGET')
```



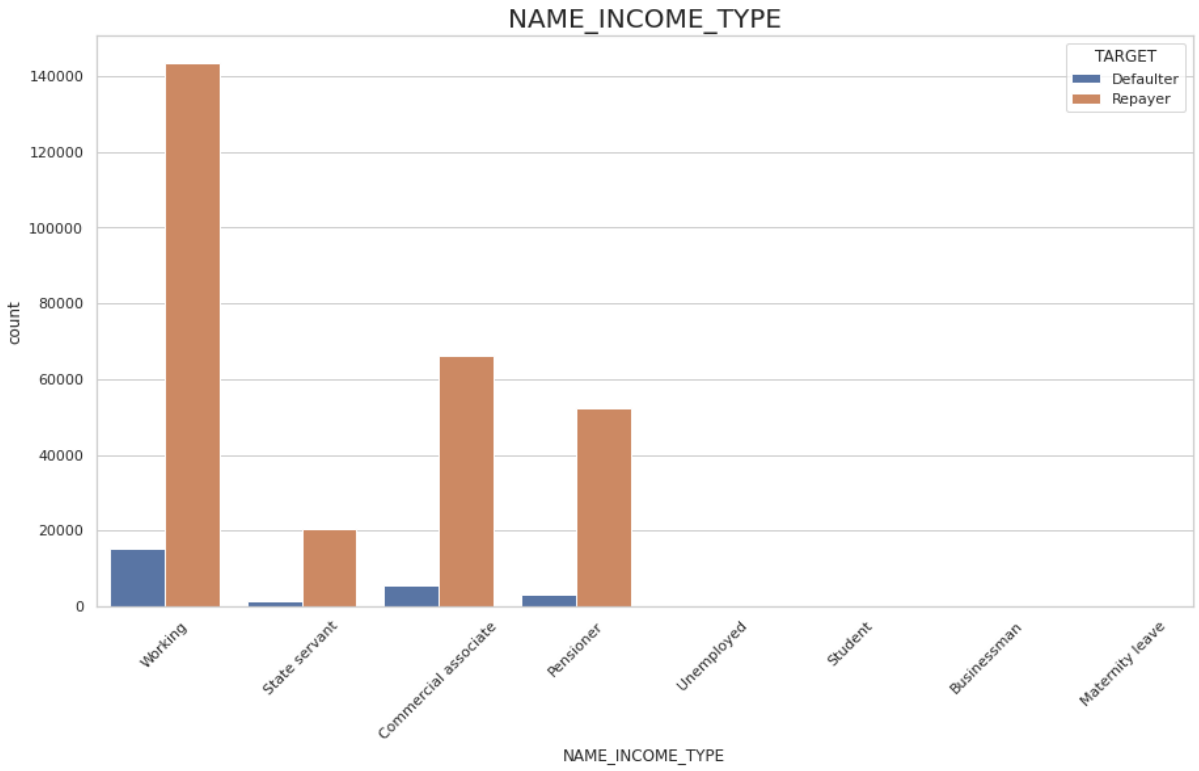


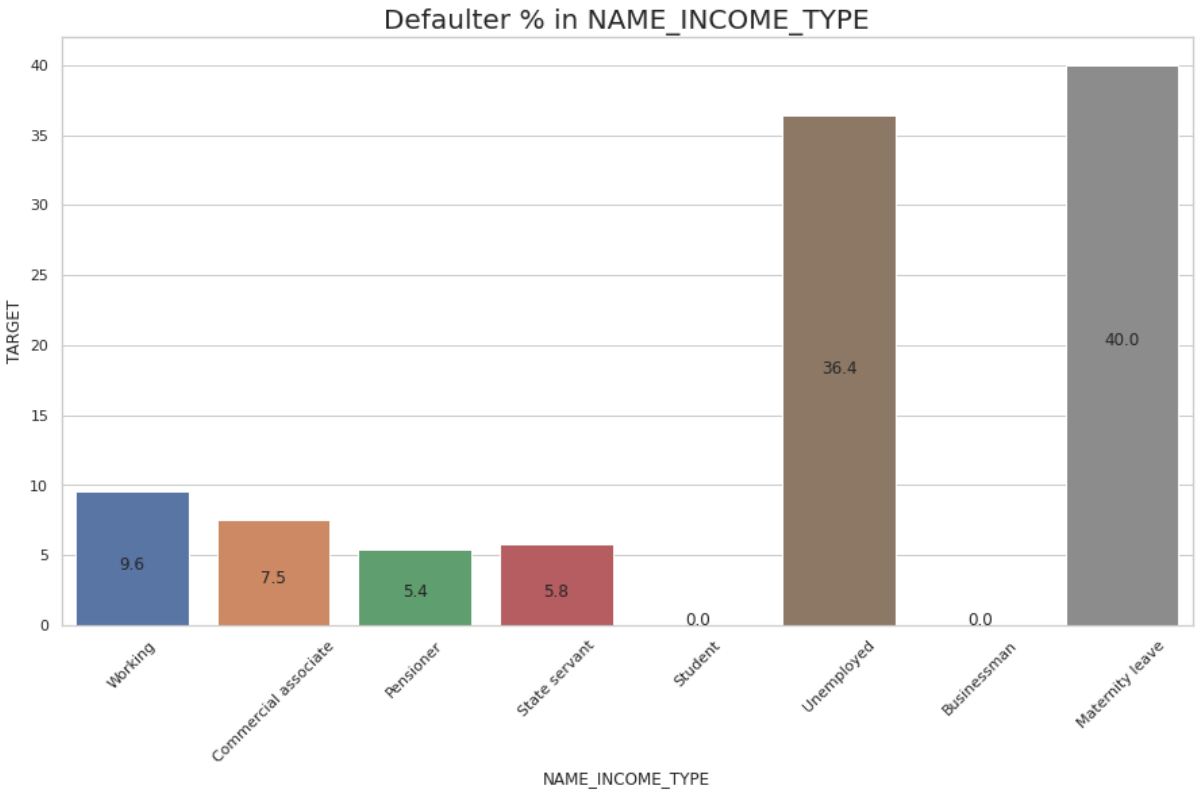
Inference:

- Accompanied category has the highest loans.
- The maximum and minimum defaulter% are Other_B (9.8%) and Unknown (5.4%).

 # Check for NAME_INCOME_TYPE based on loan repay status

```
univariate(df1,'NAME_INCOME_TYPE','TARGET')
```



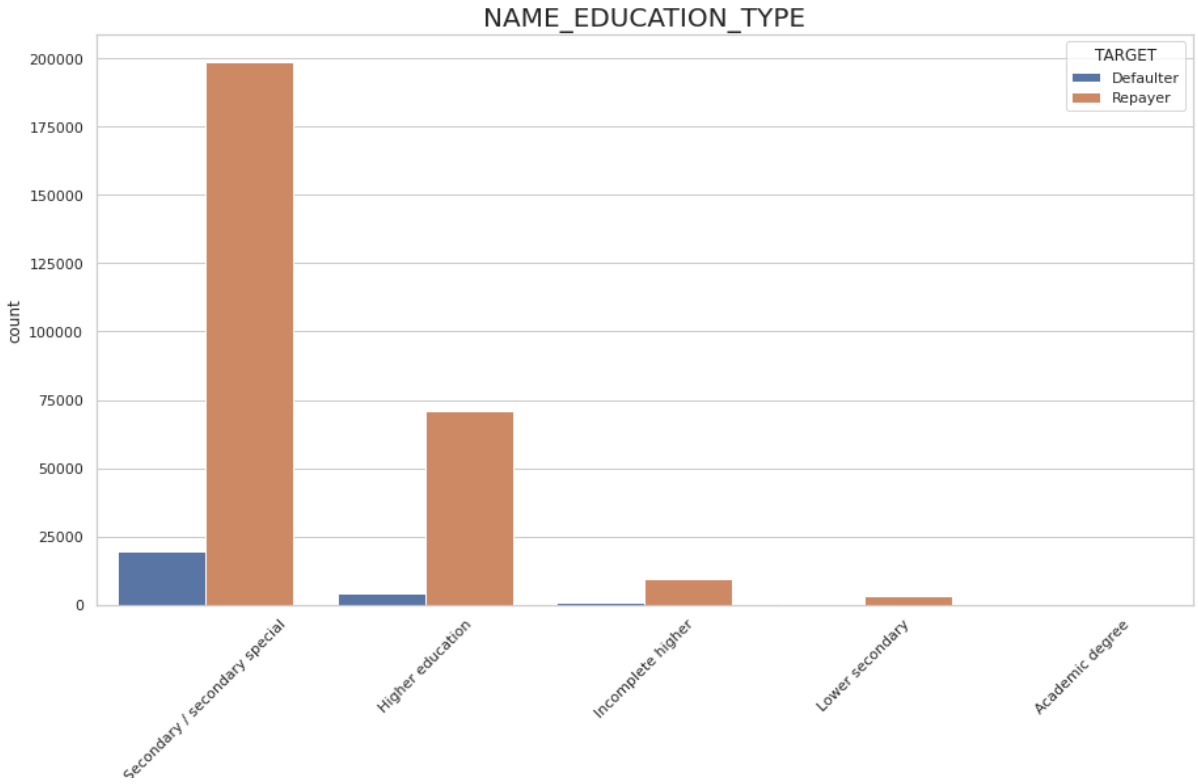


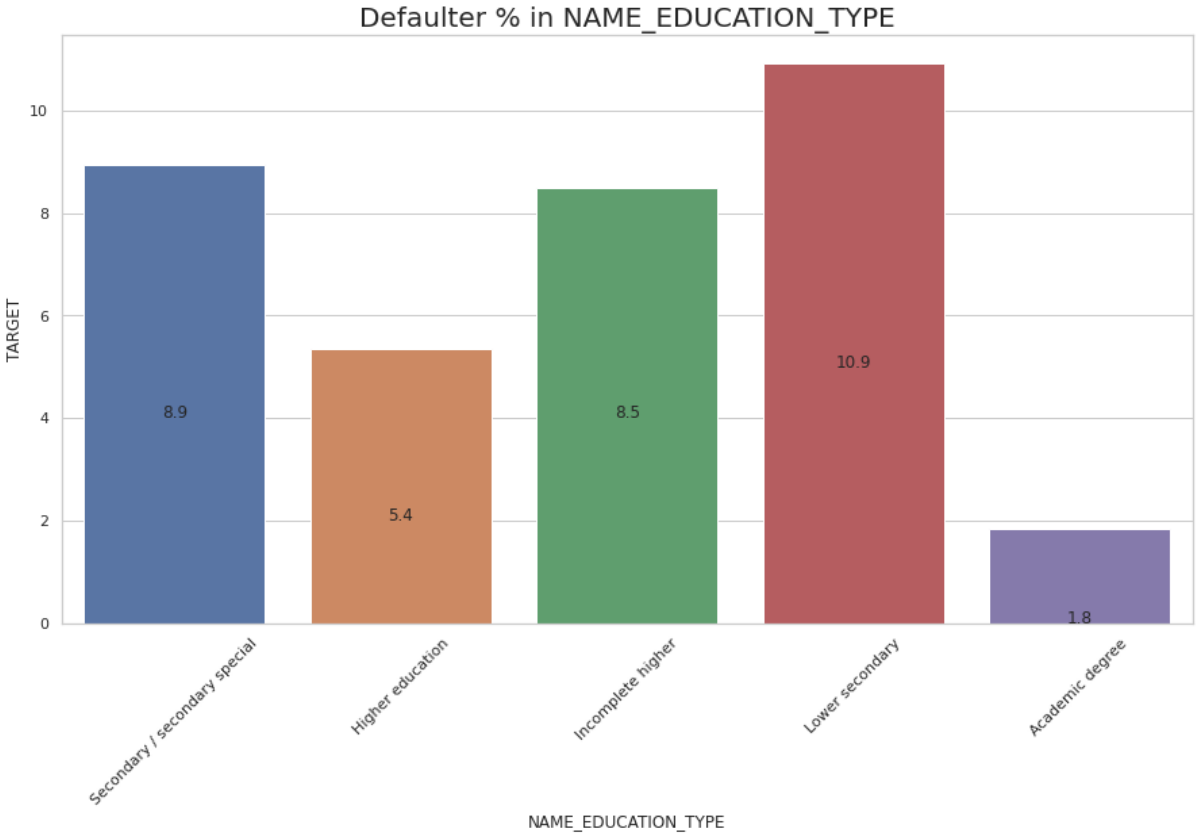
Inference:

- Working is more likely to take a loan, followed by commercial assistant, pensioner, etc.
- People on maternity leave is the riskiest category followed by unemployed, rest lies in less than 10%.
- Student and business are less in number but no records of defaulter hence it is the safest category to provide the loan.

 # Check for NAME_EDUCATION_TYPE based on loan repay status

```
univariate(df1,'NAME_EDUCATION_TYPE','TARGET')
```





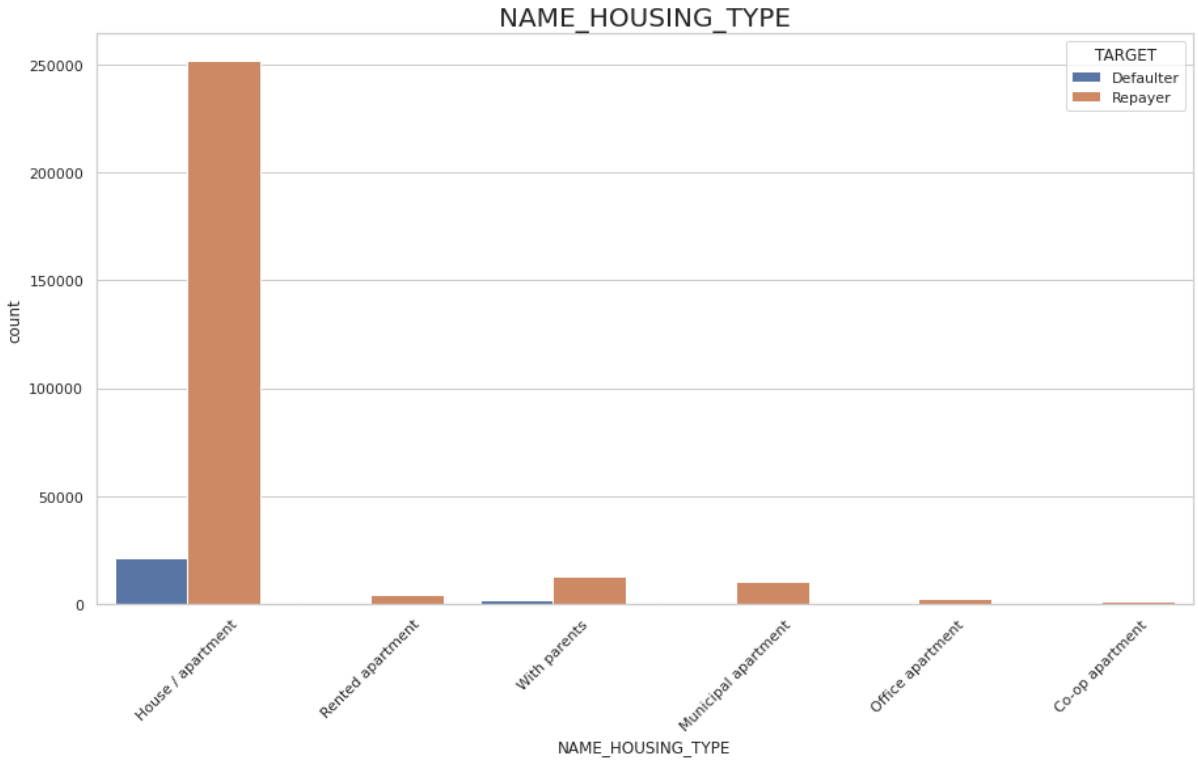
Inference:

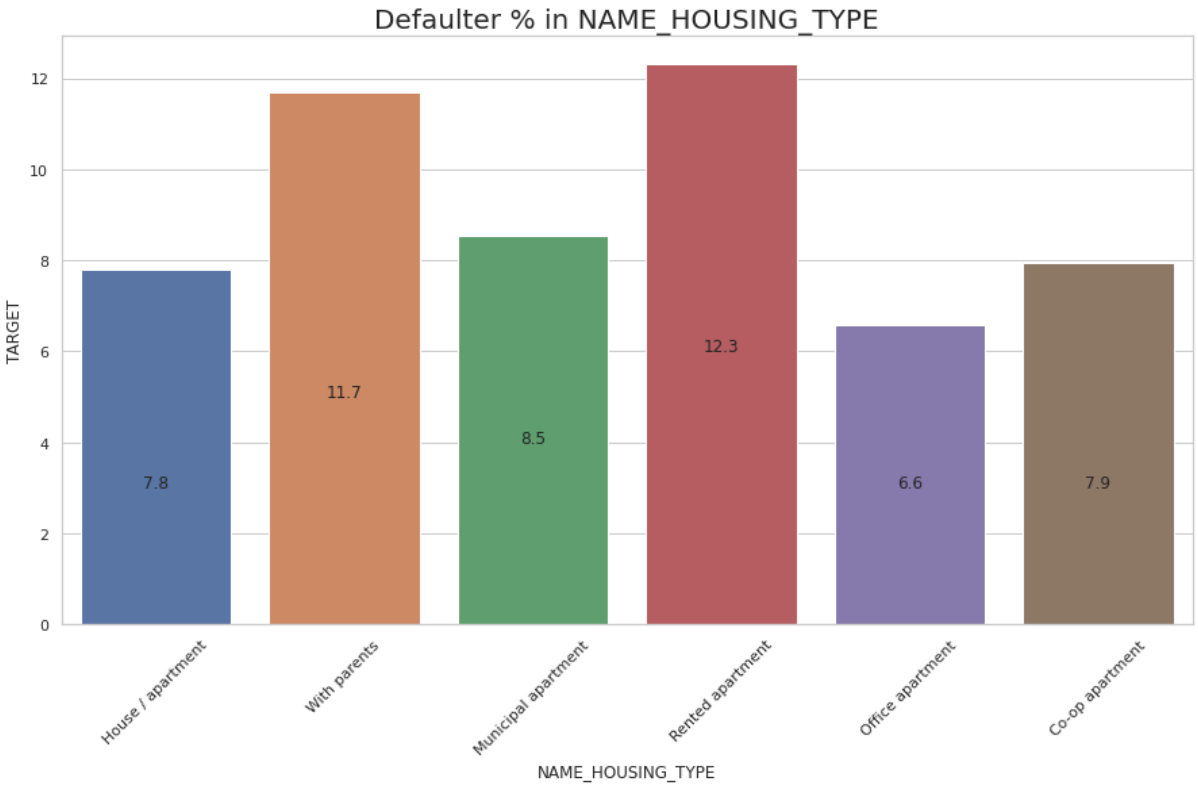
- People with the secondary / secondary special category has the hights loan application.

- lower secondary has the highest defaulter%.
- Academic degree has the lowest defaulter% among all the categories.

 # Check for NAME_HOUSING_TYPE based on loan repay status

```
univariate(df1,'NAME_HOUSING_TYPE','TARGET')
```



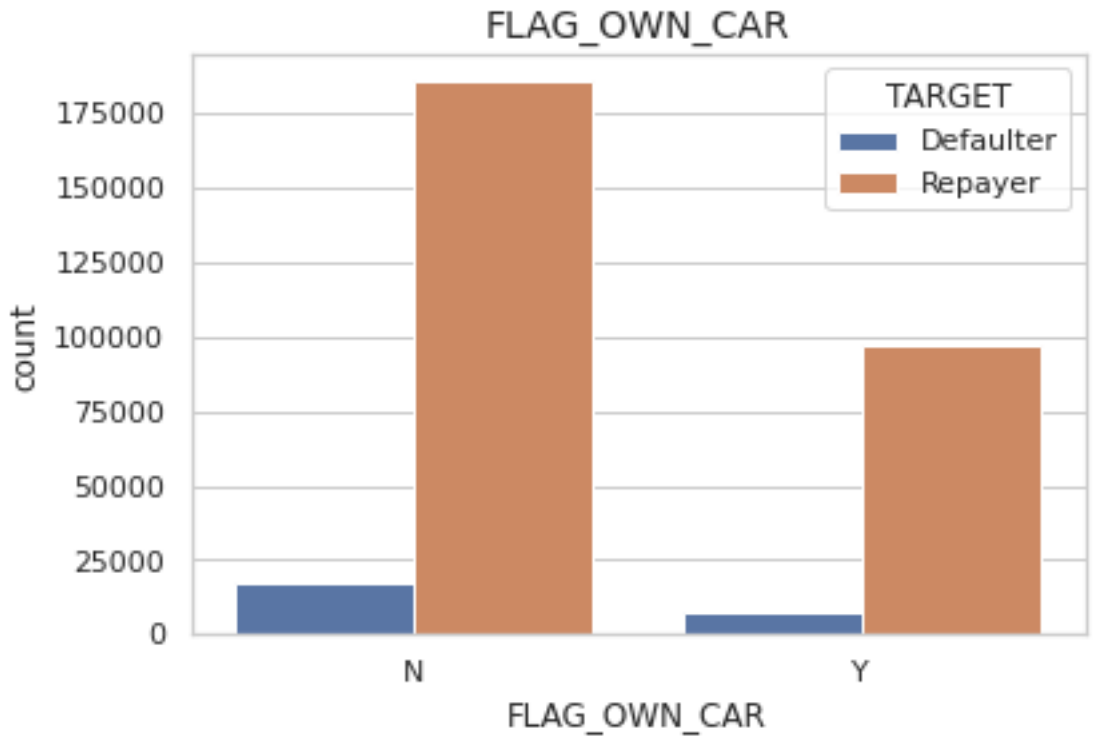


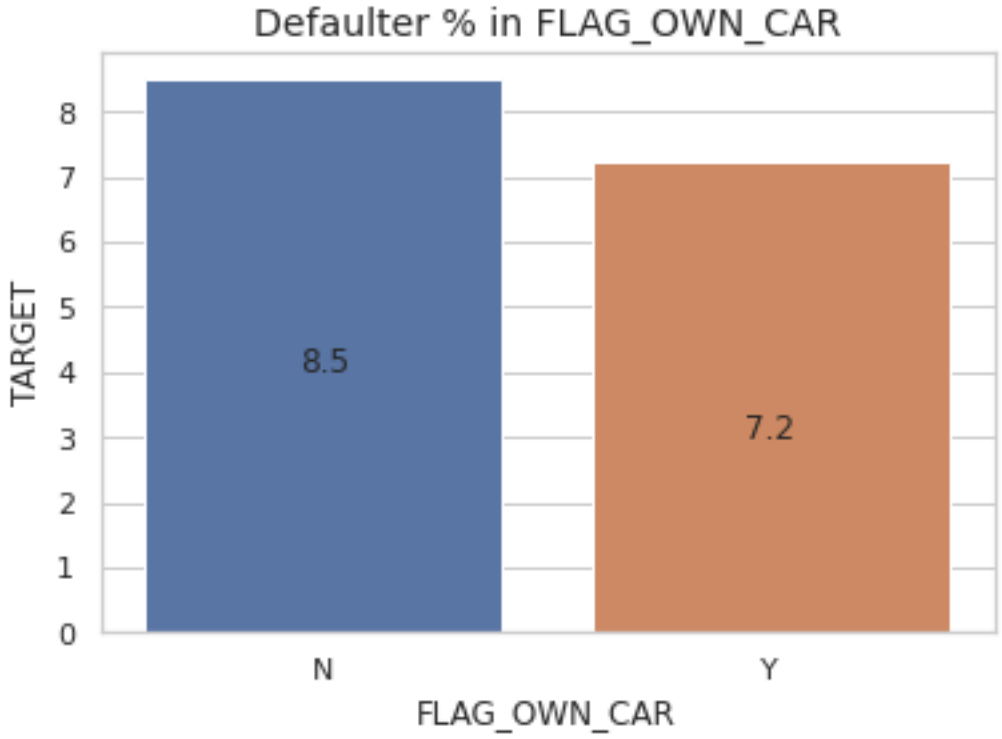
Inference:

- People with the house / apartment have the highest application.
- Rented apartment is the riskiest category among all.

 # Check for FLAG_OWN_CAR based on loan repay status

```
univariate(df1,'FLAG_OWN_CAR','TARGET')
```



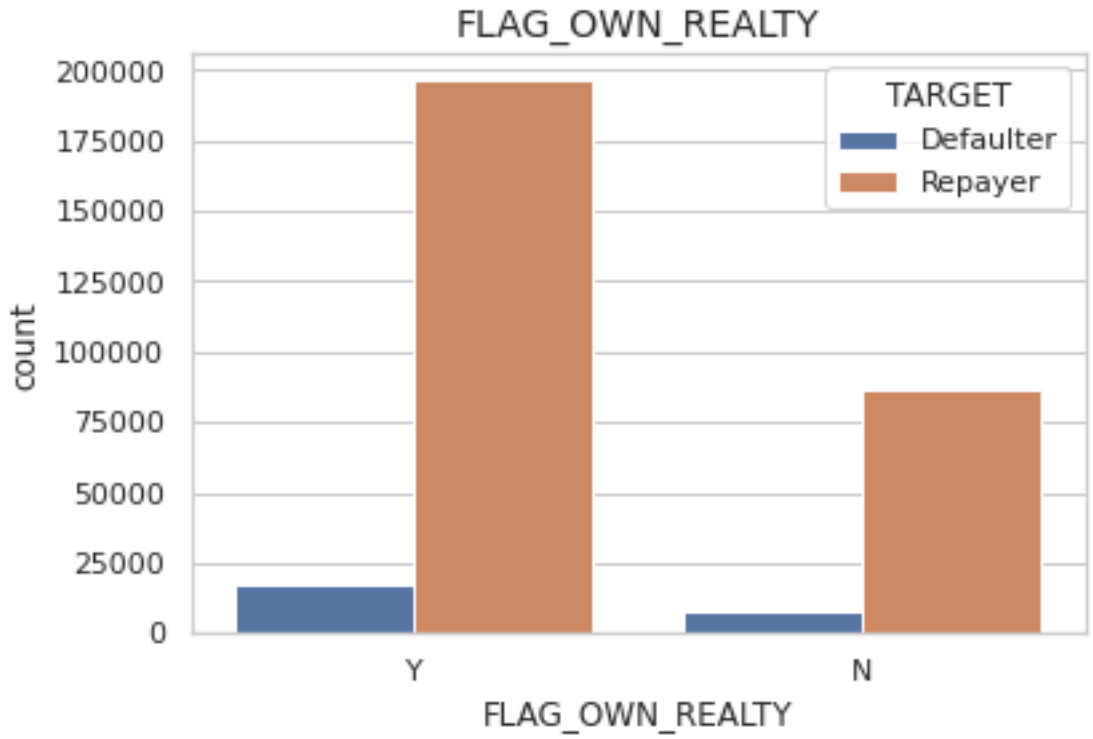


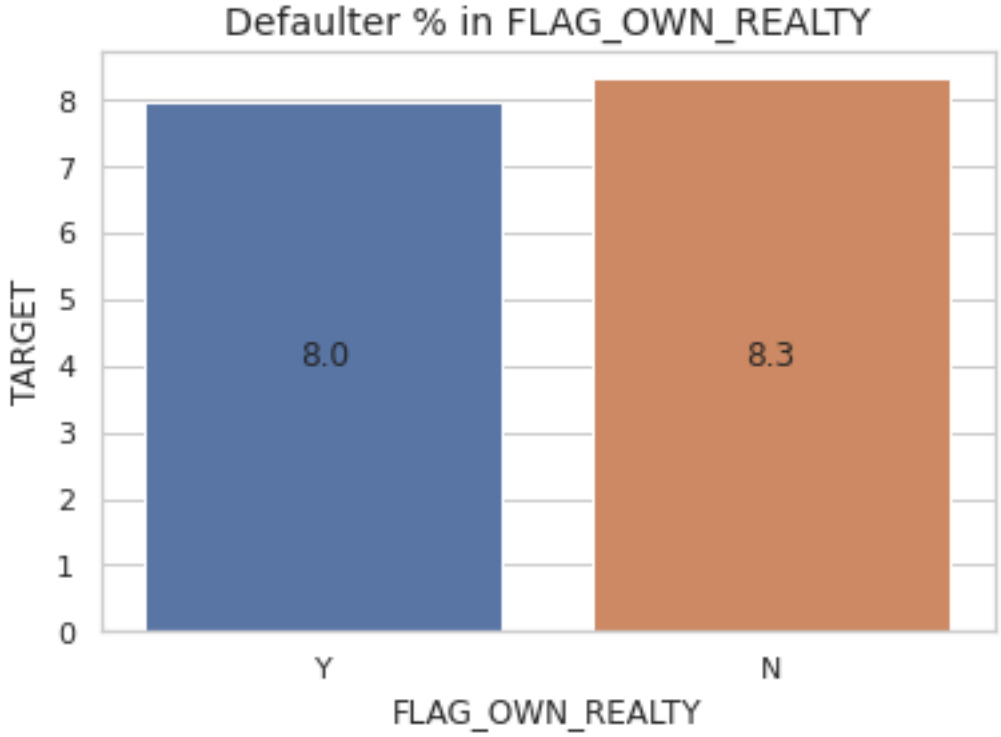
Inference:

- People who do not own car have high loan applications as well as high defaulter%.

 # Check for FLAG_OWN__REALTY type based on loan repay status

```
univariate(df1,'FLAG_OWN_REALTY','TARGET')
```



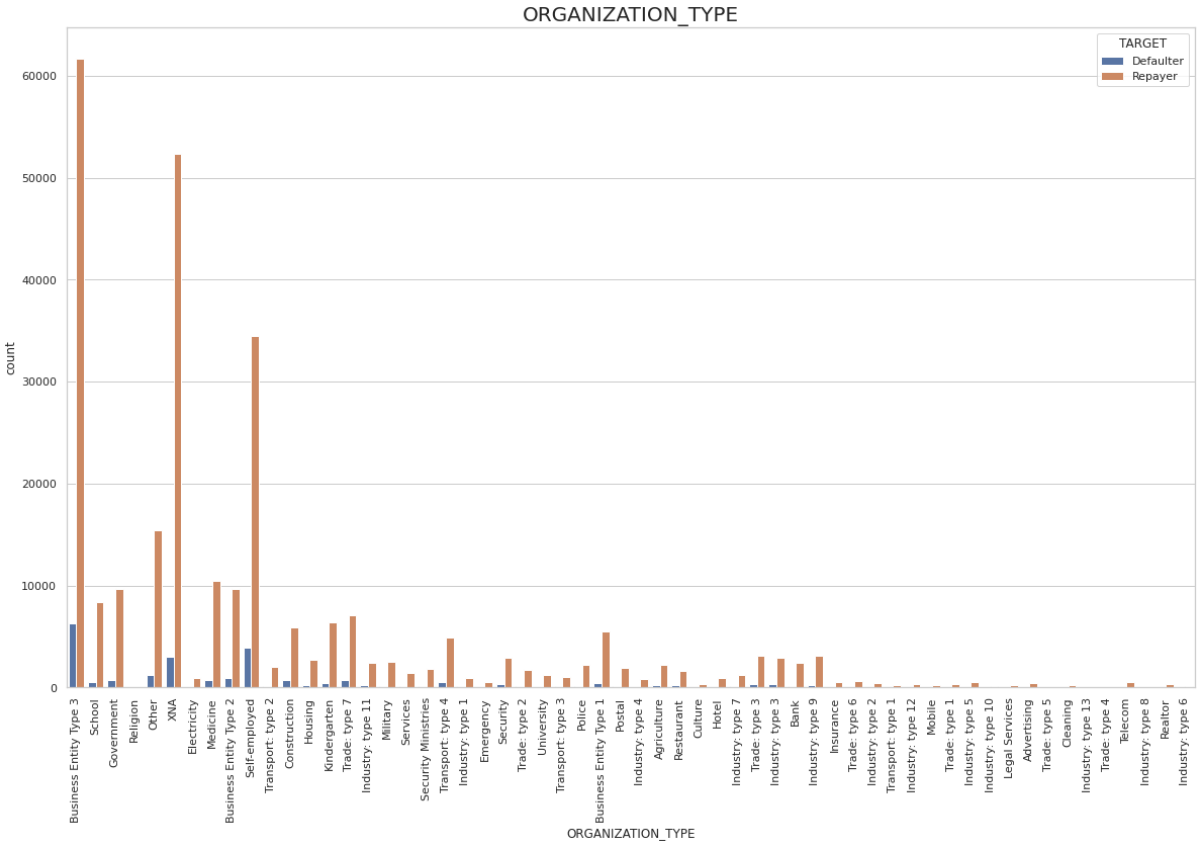


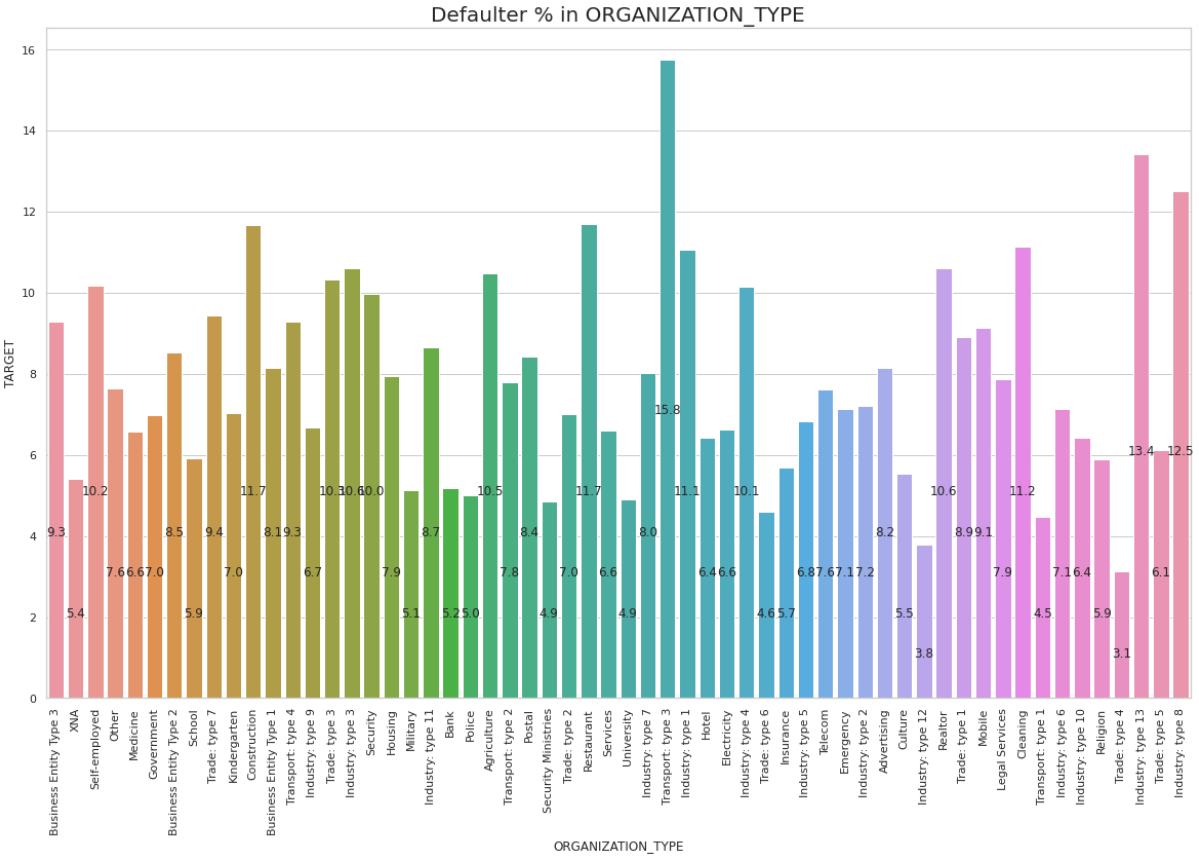
Inference:

- people who own realty have high loan application and low defaulter% than who does not own realty.

Check for ORGANIZATION_TYPE based on loan repay status

```
univariate(df1,'ORGANIZATION_TYPE','TARGET')
```





Inference:

- Business entity type 3, XNA and self employed have the highest loan application others are less than 15%.
- riskiest category is transport: type 3 (15.8%)

 # Convert AMT_INCOME_TOTAL to bins

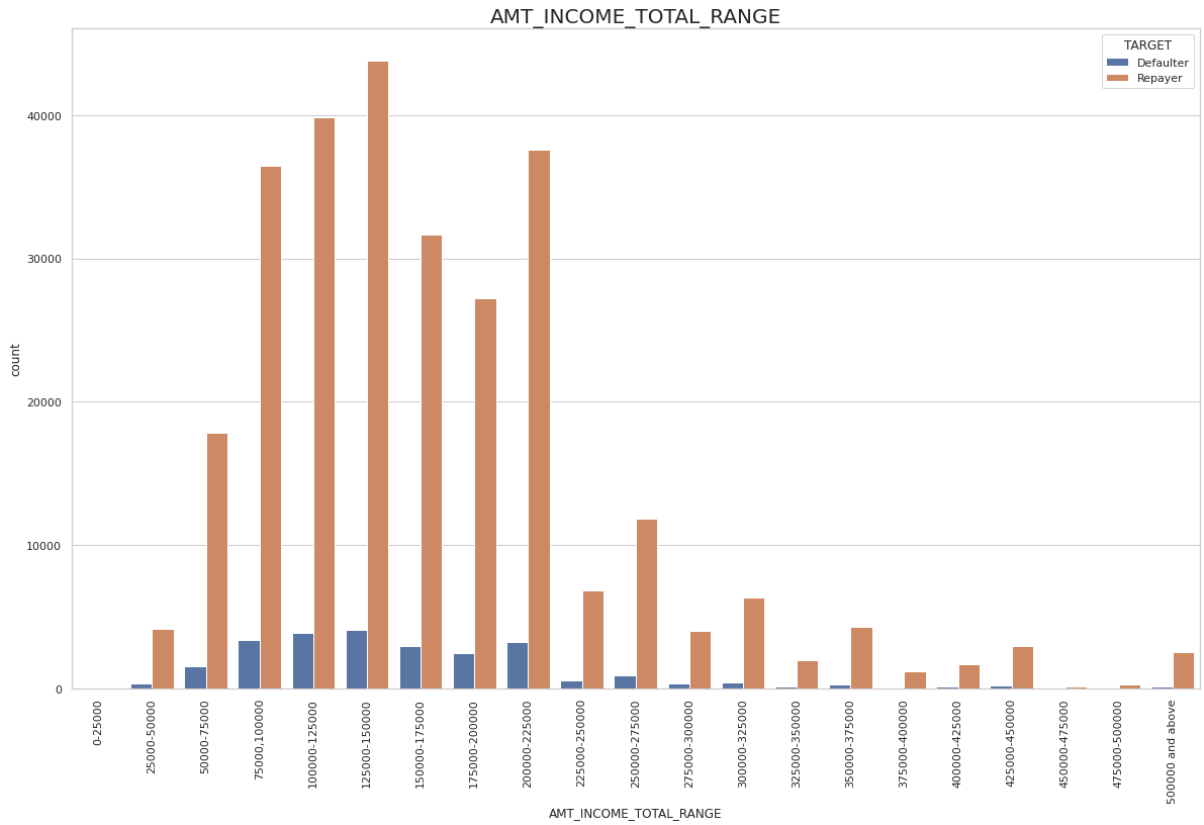
```
incomebins=[0,25000,50000,75000,100000,125000,150000,175000,200000,225000,250000,275000,300000,325000,350000,375000,400000,425000,450000,475000,500000,10000000000]
incomeslots = ['0-25000','25000-50000','50000-75000','75000,100000','100000-125000', '125000-150000', '150000-175000','175000-200000','200000-225000','225000-250000','250000-275000','275000-300000','300000-325000','325000-350000','350000-375000','375000-400000','400000-425000','425000-450000','450000-475000','475000-500000','500000 and above']
df1['AMT_INCOME_TOTAL_RANGE']=pd.cut(df1['AMT_INCOME_TOTAL'],bins=incomebins,labels=incomeslots)
```

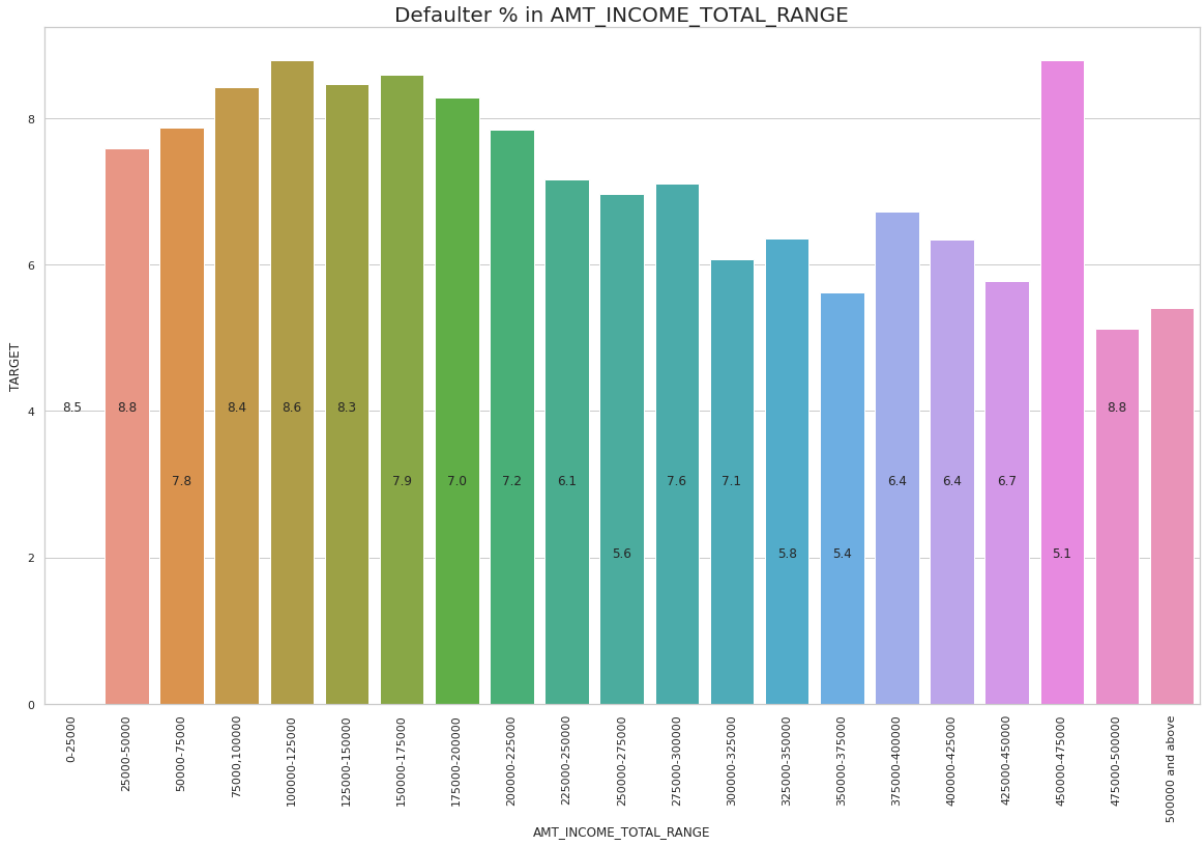
 # convert AMT_CREDIT to bins

```
creditbins = [0,150000,200000,250000,300000,350000,400000,450000,500000,550000,600000,650000,700000,750000,800000,850000,900000,1000000000]
creditslots = ['0-150000', '150000-200000','200000-250000', '250000-300000', '300000-350000', '350000-400000','400000-450000','450000-500000','500000-550000','550000-600000','600000-650000','650000-700000','700000-750000','750000-800000','800000-850000','850000-900000','900000 and above']

df1['AMT_CREDIT_RANGE'] = pd.cut(df1.AMT_CREDIT,bins=creditbins,labels=creditslots)
```

 # univariate analysis for AMT_INCOME_TOTAL_RANGE

```
univariate(df1,'AMT_INCOME_TOTAL_RANGE','TARGET')
```





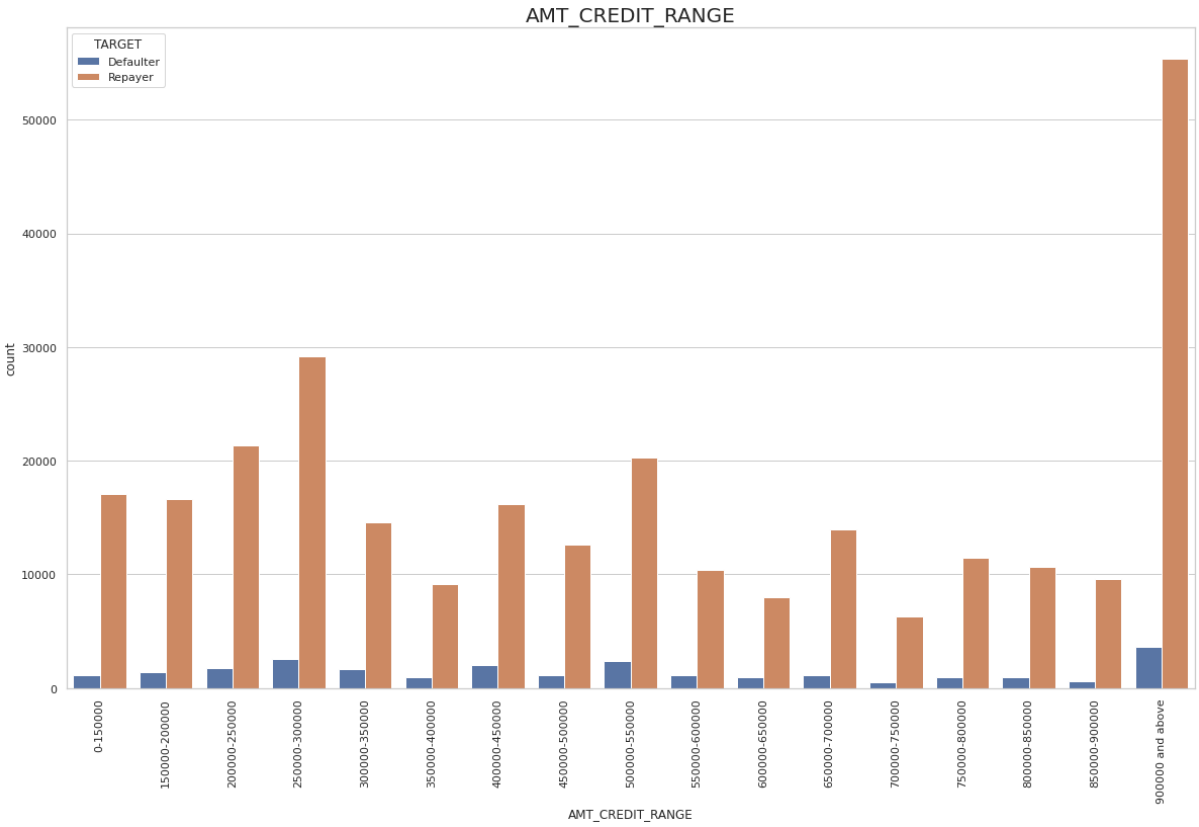
Inference:

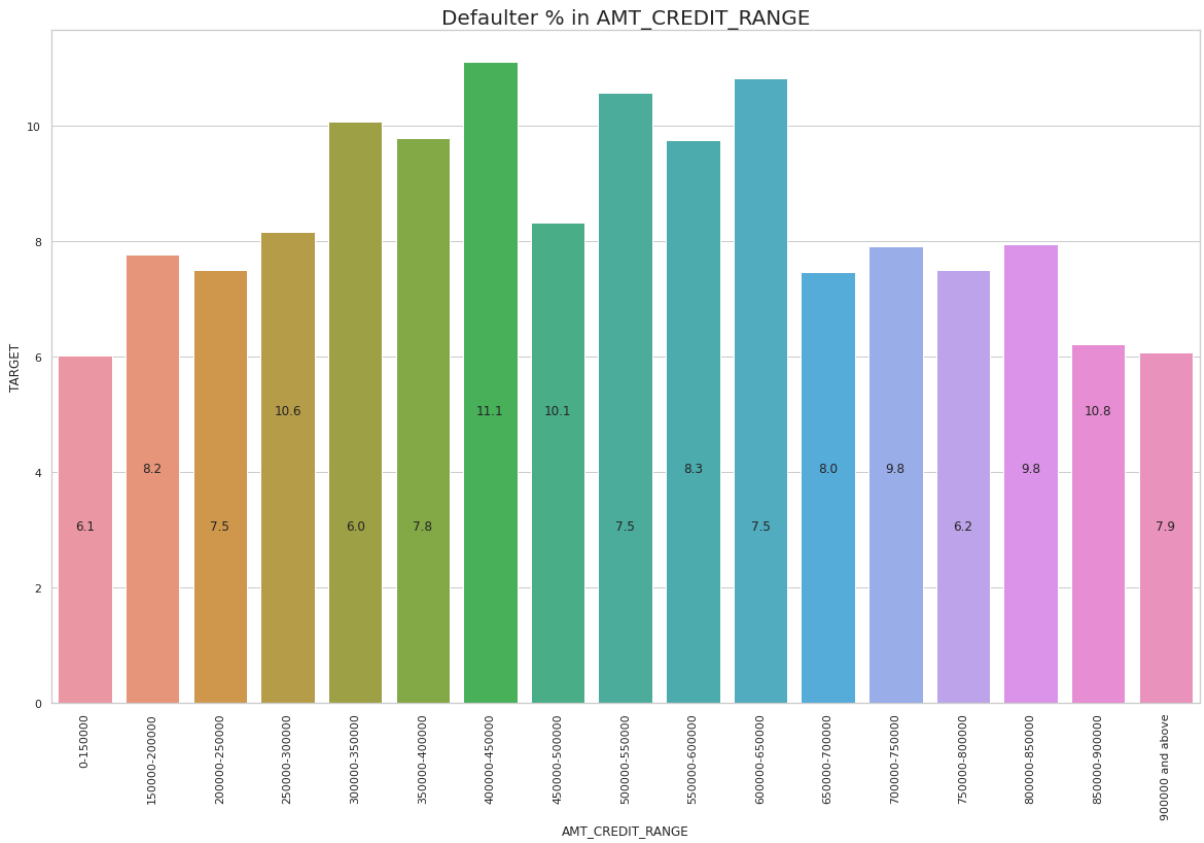
- Majority of the people who apply for the loan lies between 75000 to 225000.

- people with low salary have higher default rate.
- exception to the above point salary from 475000 to 50000 have the higher default rate.

 # Univariate analysis for AMT_CREDIT

```
univariate(df1,'AMT_CREDIT_RANGE','TARGET')
```





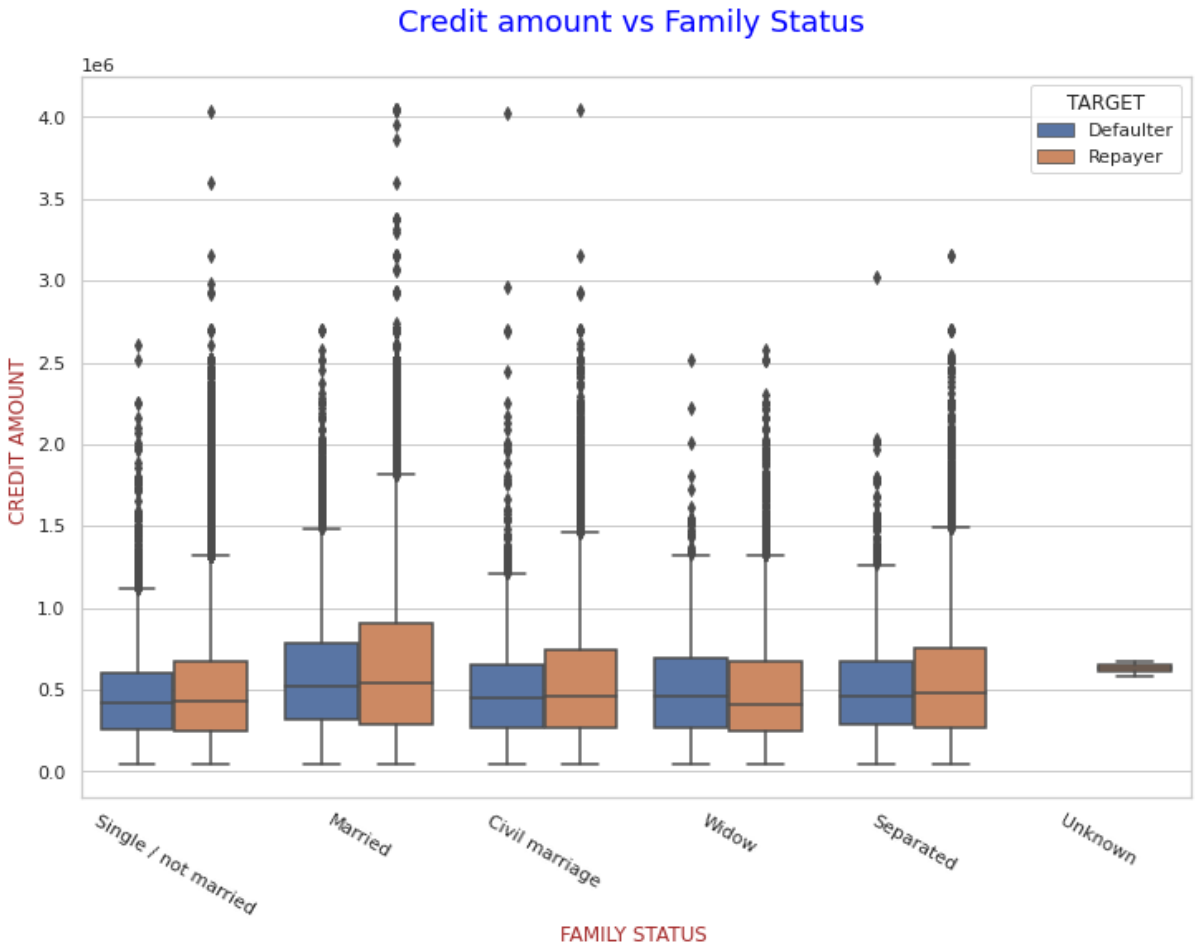
Inference:

- Maximum loans are given in the range of 90000 and above.
- credit loan 30000 to 65000 have the higher default rate.

13) Bivariate analysis for application_data.csv

#Checking for Credit amount provided to the customers based on their Family type and plotted according TO target

```
plt.figure(figsize=(12,8))
scale_factor=5
sns.boxplot(data=df1,x=df1.NAME_FAMILY_STATUS,y=df1.AMT_CREDIT,hue=df1.TARGET)
plt.xticks(rotation=-30)
plt.xlabel("FAMILY STATUS ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.ylabel("CREDIT AMOUNT ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.title('Credit amount vs Family Status \n',fontdict={'fontsize': 18, 'fontweight' : 10, 'color' : 'Blue'})
plt.show()
```

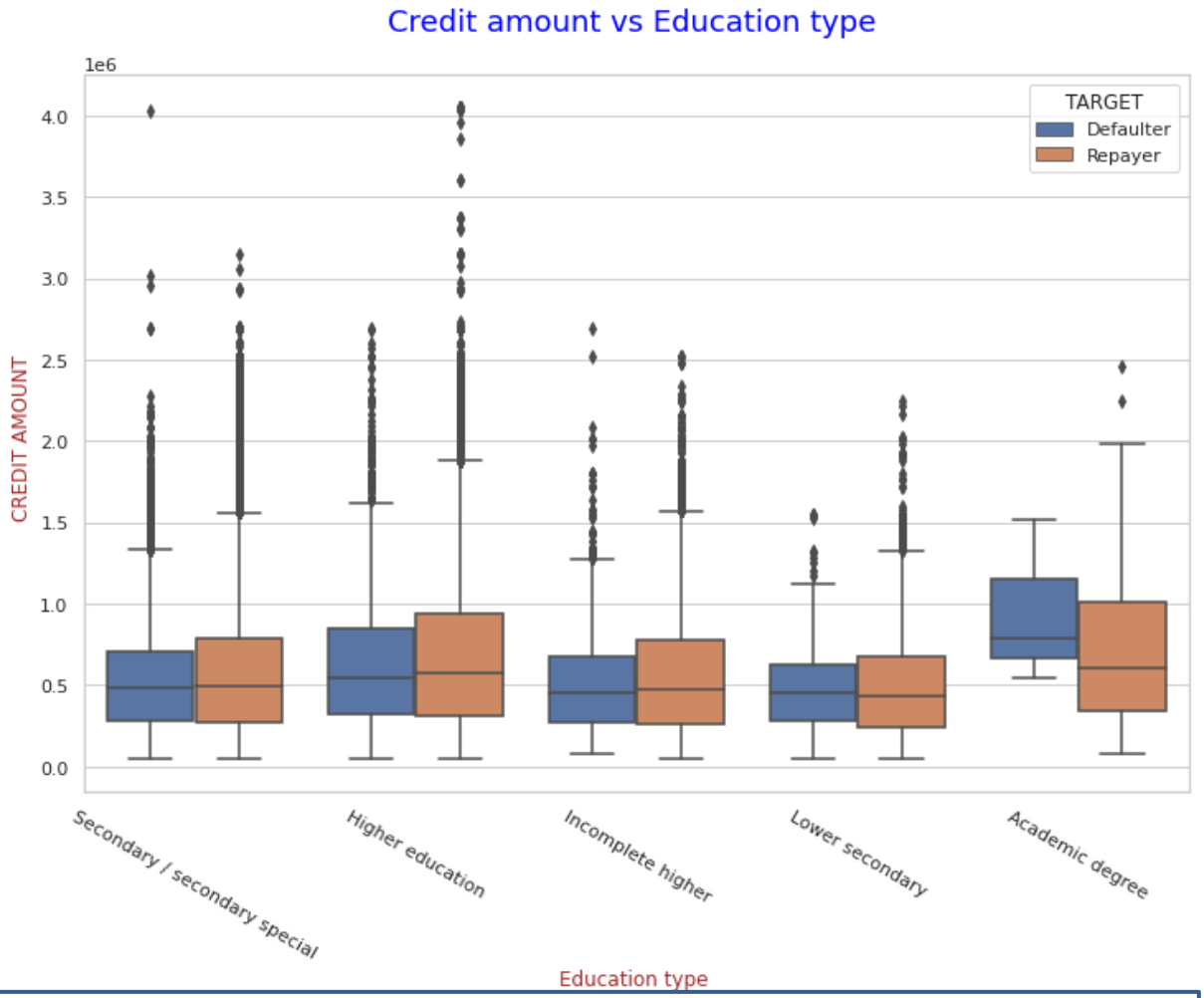


Inference:

- Civil Marriage has the highest defaulter credit amount.
- The credit amount for the repayers is high except in the case of civil marriage.

#Checking for Credit amount provided to the customers based on their Education type and plotted according TO target

```
plt.figure(figsize=(12,8))
scale_factor=5
sns.boxplot(data=df1,x=df1.NAME_EDUCATION_TYPE,y=df1.AMT_CREDIT,hue=df1.TARGET)
plt.xticks(rotation=-30)
plt.xlabel("Education type ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.ylabel("CREDIT AMOUNT ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.title('Credit amount vs Education type \n',fontdict={'fontsize': 18, 'fontweight' : 10, 'color' : 'Blue'})
plt.show()
```

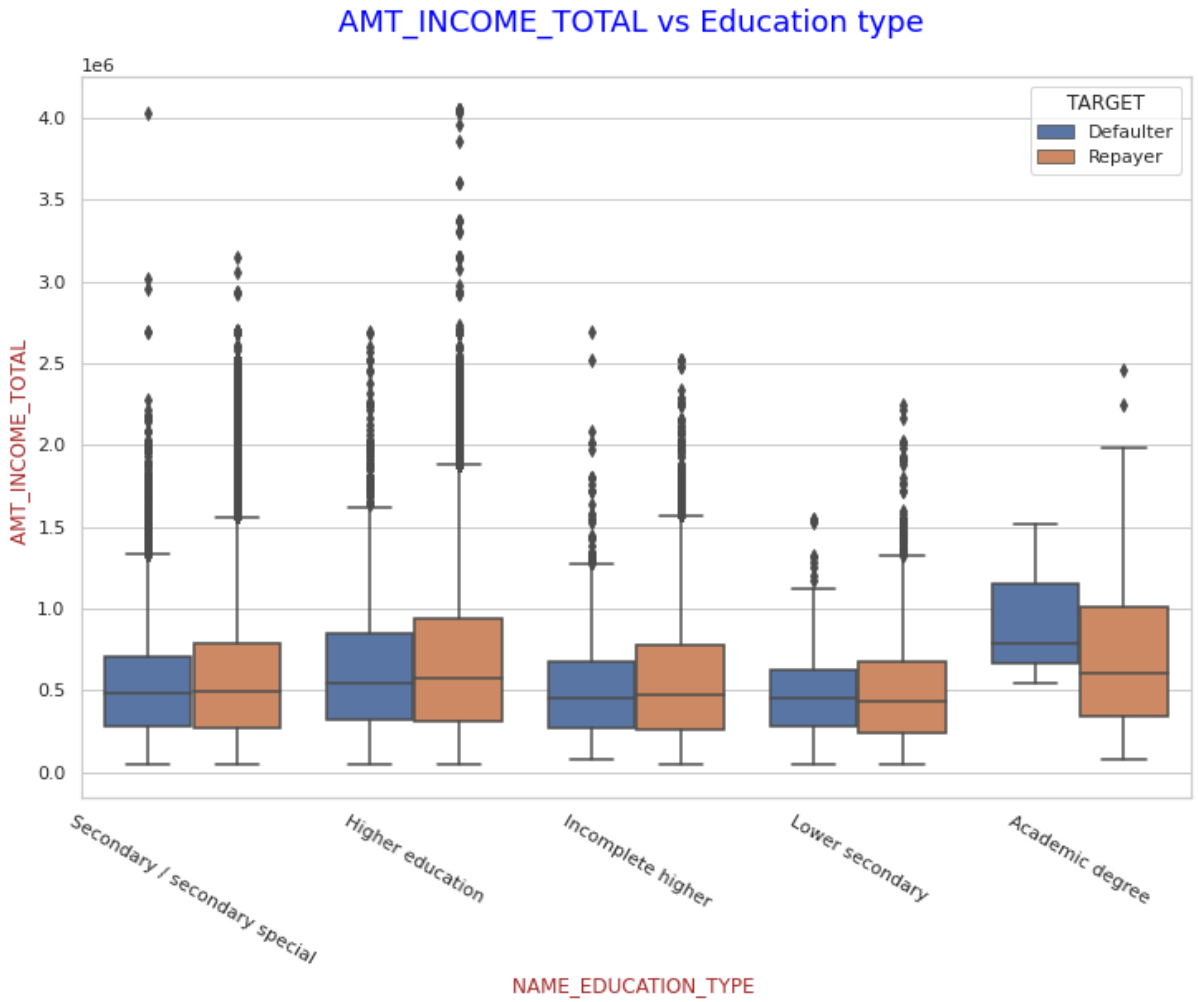


Inference:

- Higher education people are high credit seekers.
- Secondary / secondary special has high defaulter credit amount.

#Checking for AMT_INCOME_TOTAL provided to the customers based on their education type and plotted according to target

```
plt.figure(figsize=(12,8))
scale_factor=5
sns.boxplot(data=df1,x=df1.NAME_EDUCATION_TYPE,y=df1.AMT_CREDIT,hue=df1.TARGET)
plt.xticks(rotation=-30)
plt.xlabel("NAME_EDUCATION_TYPE ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.ylabel("AMT_INCOME_TOTAL ", fontdict={'fontsize': 12, 'fontweight' : 5, 'color' : 'Brown'})
plt.title('AMT_INCOME_TOTAL vs Education type \n',fontdict={'fontsize': 18, 'fontweight' : 10, 'color' : '
Blue'})
plt.show()
```

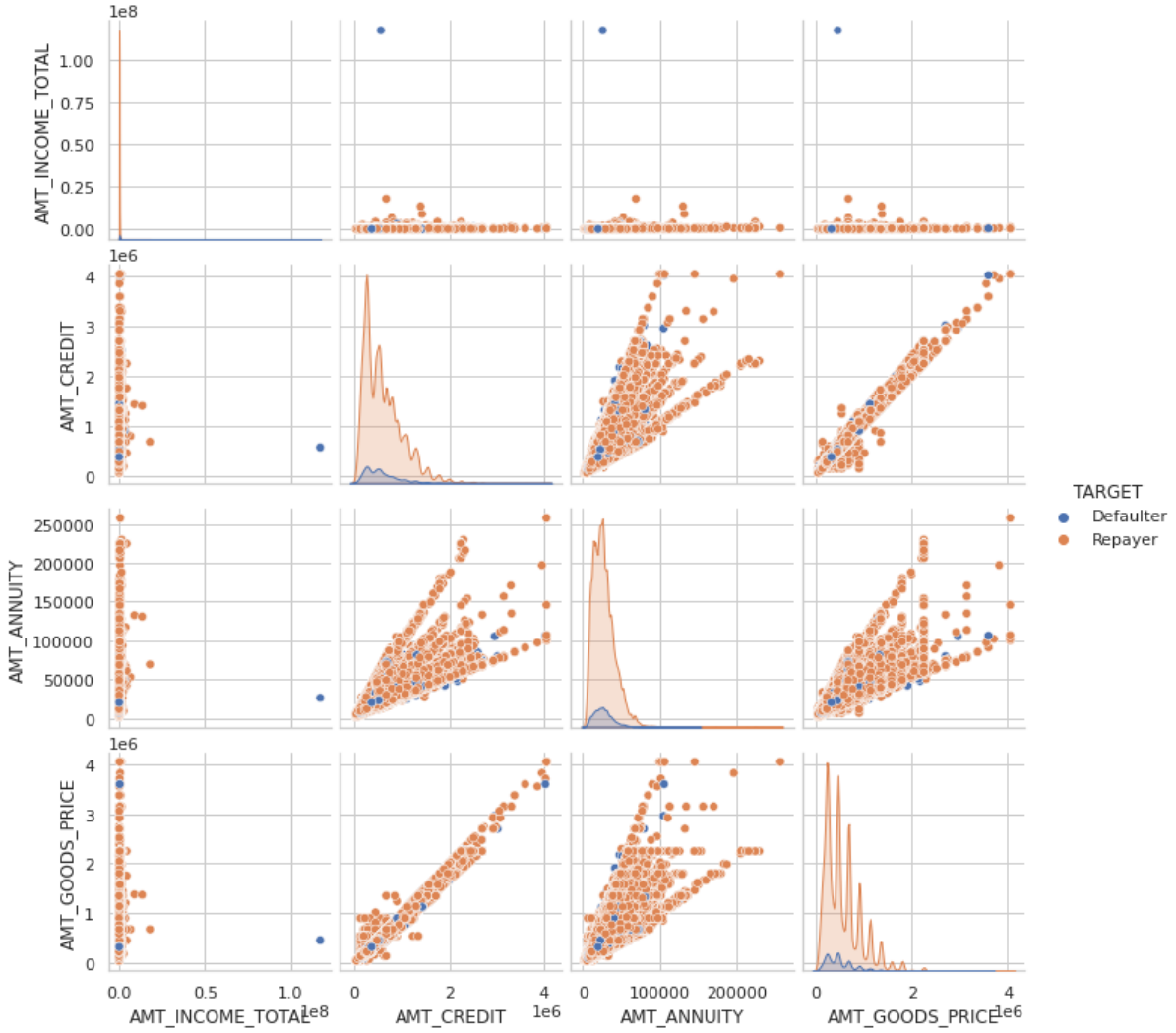


Inference:

- higher education is earning more than any education type.
- secondary / secondary special has the higher defaulter percentage.

 # Pair plots for the numeric column

```
amount = df1[['AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY','AMT_GOODS_PRICE','TARGET']]
sns.pairplot(amount,hue = 'TARGET')
```



Inference:

- When Annuity Amount > 15K and Good Price Amount > 20 Lakhs, there is a lesser chance of defaulters
- Loan Amount (AMT_CREDIT) and Goods price(AMT_GOODS_PRICE) are highly correlated. It forms a linear relation.
- There are very less defaulters for AMT_CREDIT >20 Lakhs

14) Correlation check for the defaulter in application_data.csv

 # Divide the data based on the target column

```
defaulter_df = df1[df1['TARGET']==1]
```

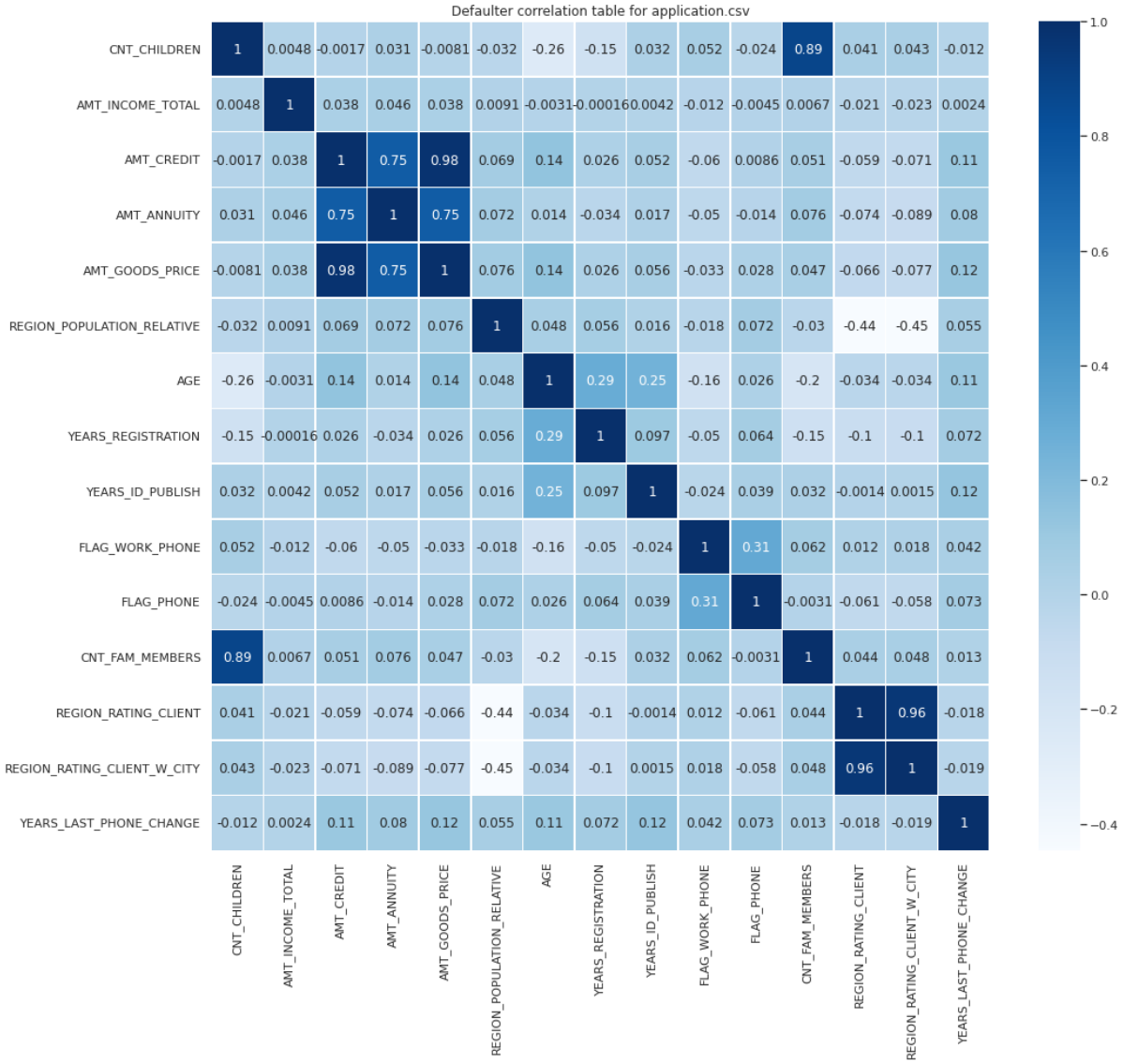
 # Correlation check

```
corr2 = defaulter_df[['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_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS',
    'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE', 'AGE',
```

```
    'YEARS_REGISTRATION', 'YEARS_ID_PUBLISH', 'FLAG_WORK_PHONE',
    'FLAG_PHONE', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBERS',
    'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY','ORGANIZATION_TYPE',
'YEARS_LAST_PHONE_CHANGE']].corr()

plt.figure(figsize=(16,14))
sns.heatmap(corr2,cmap='Blues',annot=True,linewidth=0.5)
plt.title('Defaulter correlation table for application.csv')
corr2 = corr2.where(np.triu(np.ones(corr2.shape), k=1).astype(bool))
corr_mat = corr2.unstack().sort_values(ascending=False).reset_index()
corr_mat.rename(columns={'level_0':'Var1','level_1':'Var2',0:'Correlation'},inplace=True )
print(corr_mat.head(10))
```

Top 10 correlation

	Var1	Var2	Correlation
0	AMT_GOODS_PRICE	AMT_CREDIT	0.982566
1	REGION_RATING_CLIENT_W_CITY	REGION_RATING_CLIENT	0.956637
2	CNT_FAM_MEMBERS	CNT_CHILDREN	0.885484
3	AMT_ANNUITY	AMT_CREDIT	0.752195
4	AMT_GOODS_PRICE	AMT_ANNUITY	0.752022
5	FLAG_PHONE	FLAG_WORK_PHONE	0.311035
6	YEARS_REGISTRATION	AGE	0.289114
7	YEARS_ID_PUBLISH	AGE	0.252863
8	AGE	AMT_GOODS_PRICE	0.135754
9	AGE	AMT_CREDIT	0.135316



15) Univariate analysis of merged data frame

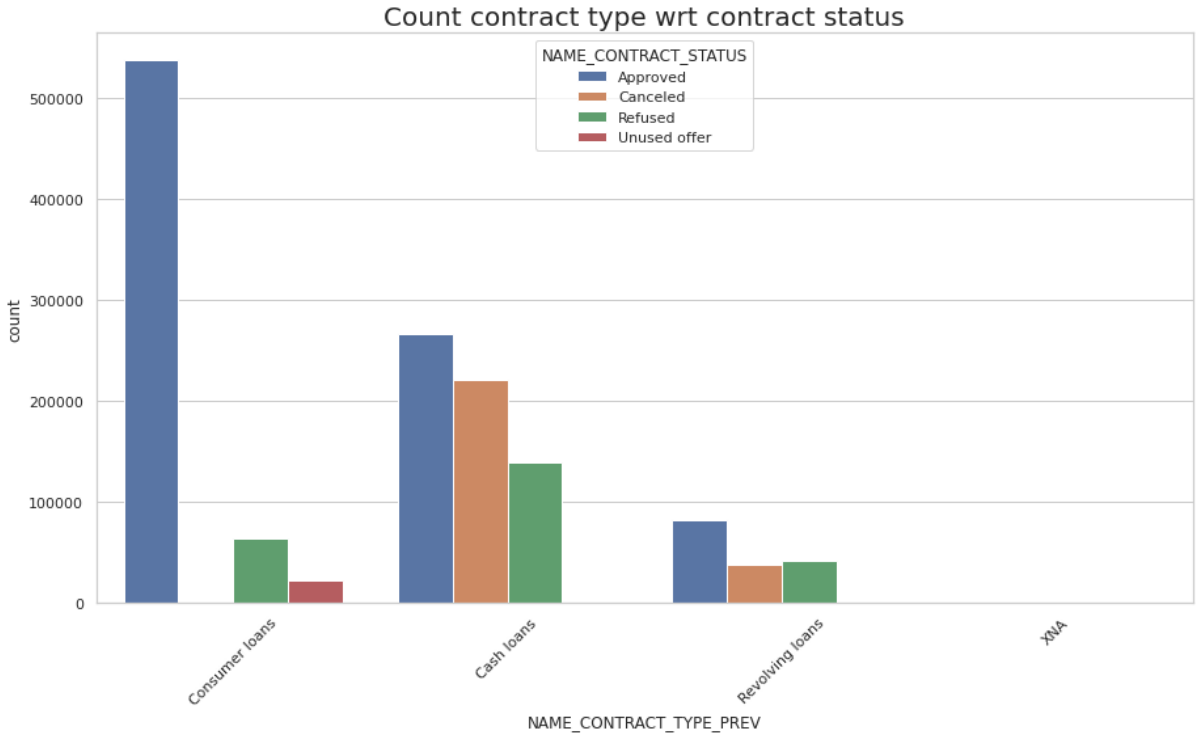
 # Merge the data set on SK_ID_CURR

```
mergedf = df1.merge(df, on='SK_ID_CURR')
```

 # Renaming the columns as convention

```
mergedf = mergedf.rename({'NAME_CONTRACT_TYPE_x' : 'NAME_CONTRACT_TYPE','AMT_CREDIT_x':'AMT_CREDIT','AMT_ANNUITY_x':'AMT_ANNUITY',
                'WEEKDAY_APPR_PROCESS_START_x' : 'WEEKDAY_APPR_PROCESS_START','AMT_GOODS_PRICE_x':'AMT_GOODS_PRICE',
                'HOUR_APPR_PROCESS_START_x':'HOUR_APPR_PROCESS_START','NAME_CONTRACT_TYPE_y':'NAME_CONTRACT_TYPE_PREV',
                'AMT_CREDIT_y':'AMT_CREDIT_PREV','AMT_ANNUITY_y':'AMT_ANNUITY_PREV','AMT_GOODS_PRICE_y':'AMT_GOODS_PRICE_PREV',
                'WEEKDAY_APPR_PROCESS_START_y':'WEEKDAY_APPR_PROCESS_START_PREV',
                'HOUR_APPR_PROCESS_START_y':'HOUR_APPR_PROCESS_START_PREV'}, axis =1)
```

```
plt.figure(figsize = (15,8))
sns.countplot(x = mergedf['NAME_CONTRACT_TYPE_PREV'], hue = mergedf['NAME_CONTRACT_STATUS'])
plt.xticks(rotation = 45)
plt.title('Count contract type wrt contract status',fontdict={'fontsize': 20})
```

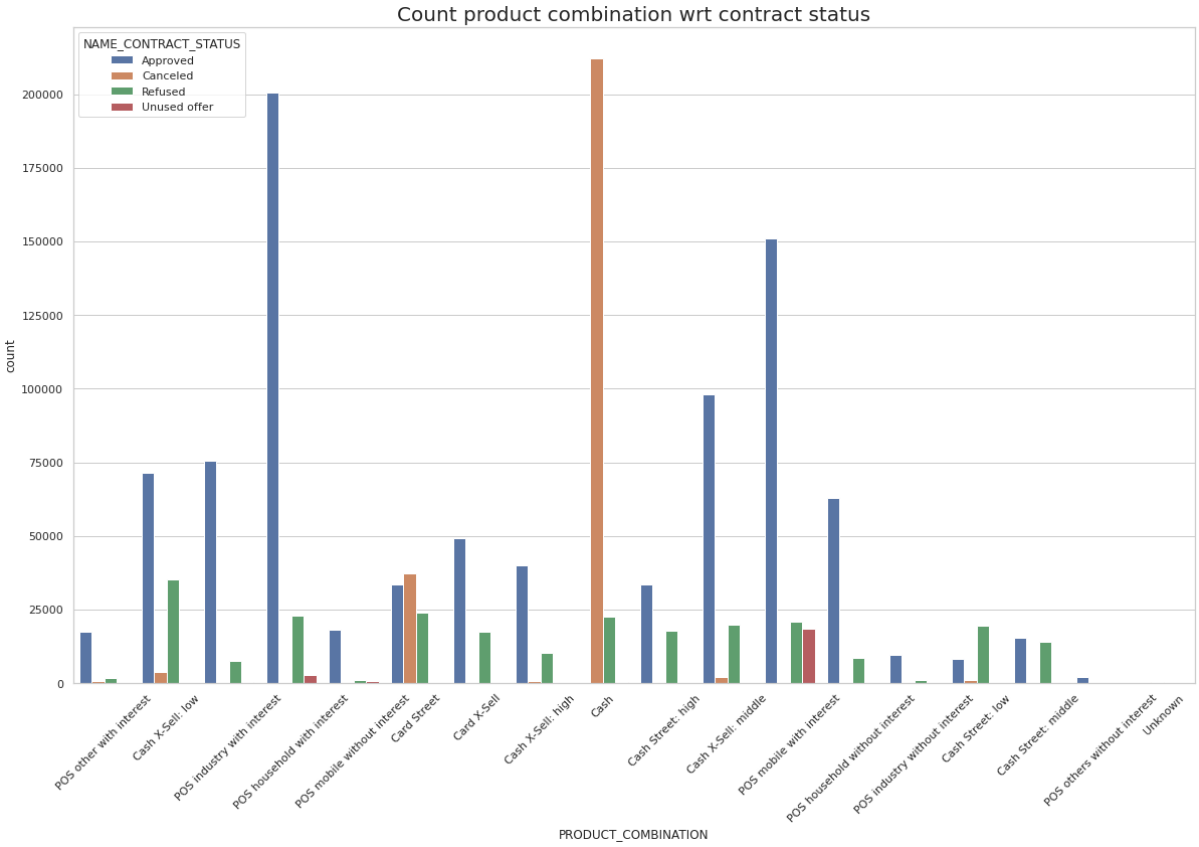


Inference:

- Consumer loans are highly applied and highly approved.
- There are only few cancelled loans in consumer loans.
- Revolving loans are less applied.
- More than 70% of cash loans are cancelled.

 # univariate analysis of PRODUCT_COMBINATION with hue NAME_CONTRACT_STATUS

```
plt.figure(figsize = (20,12))
sns.countplot(mergedf['PRODUCT_COMBINATION'],hue=mergedf['NAME_CONTRACT_STATUS'])
plt.xticks(rotation = 45)
plt.title('Count product combination wrt contract status',fontdict={'fontsize': 20})
```

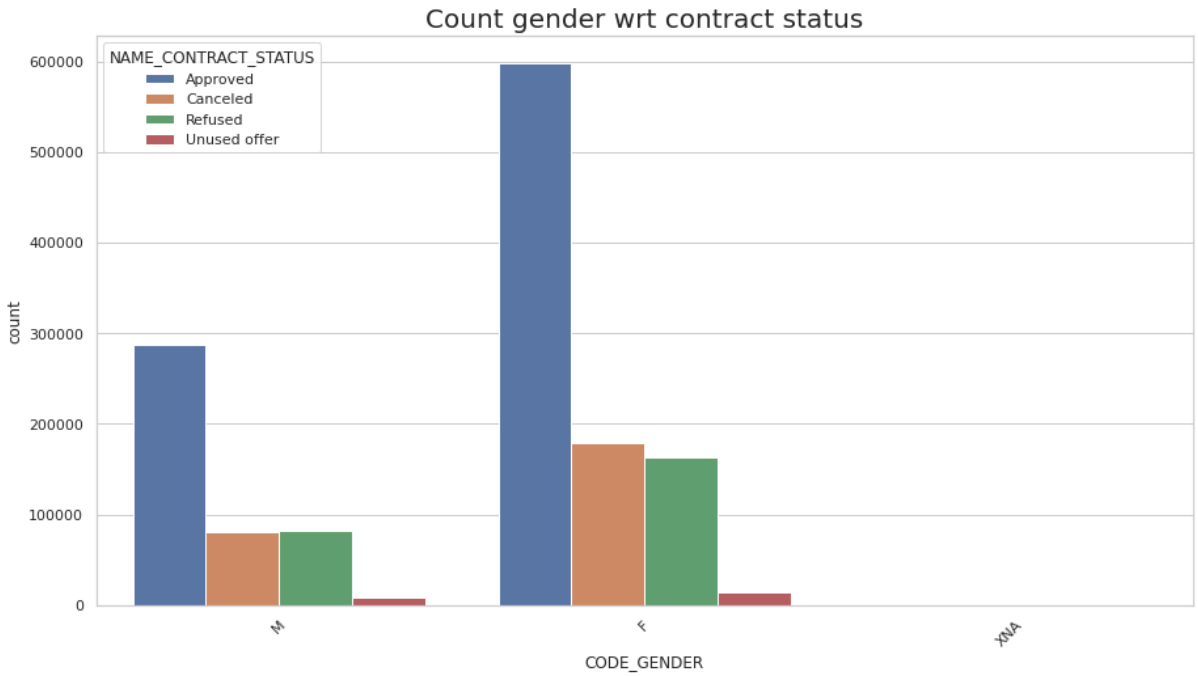


Inference:

- POS industry with interest has the highest approved loans.
- Cash loans has the highest cancelled loans.

 # Univariate analysis of CODE_GENDER with hue NAME_CONTRACT_STATUS

```
plt.figure(figsize = (15,8))
sns.countplot(mergedf['CODE_GENDER'],hue=mergedf['NAME_CONTRACT_STATUS'])
plt.xticks(rotation = 45)
plt.title('Count gender wrt contract status',fontdict={'fontsize': 20})
```



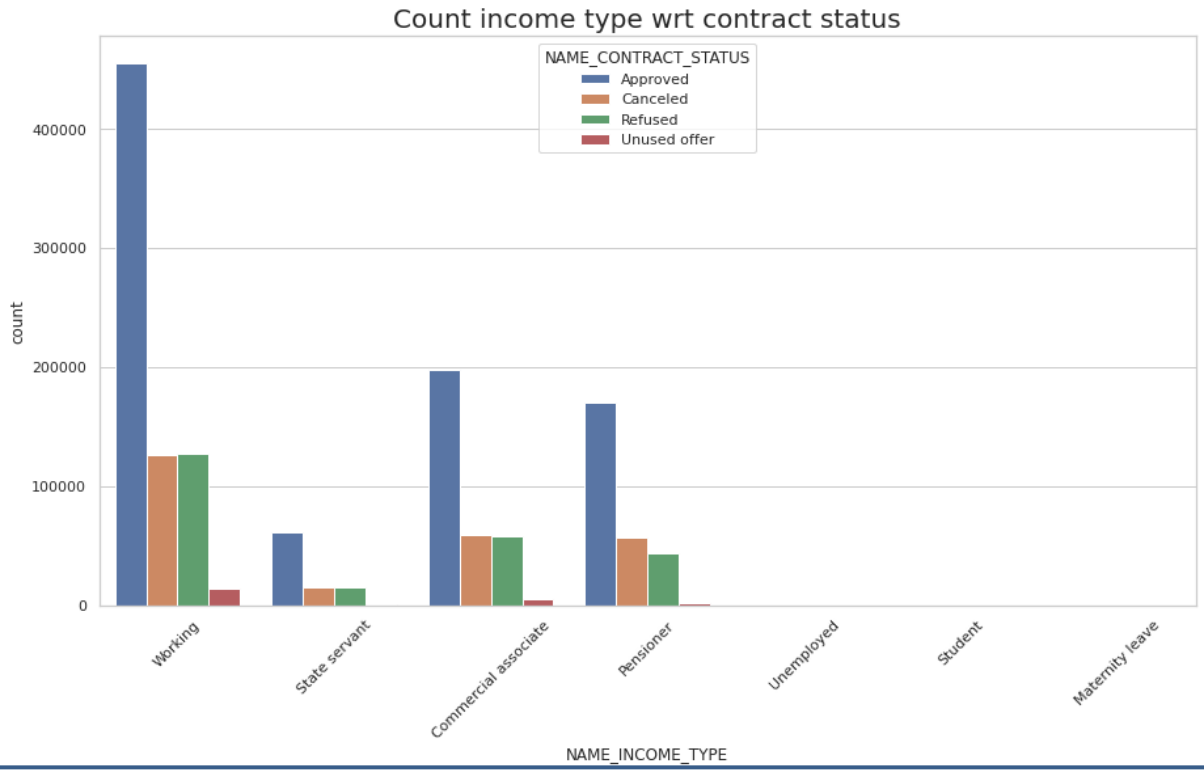
Inference:

- Female loans are Approved more than male.
- Female loans have high unused offers than male.

 # Univariate analysis of NAME_INCOME_TYPE with hue

NAME_CONTRACT_STATUS

```
plt.figure(figsize = (15,8))
sns.countplot(mergedf['NAME_INCOME_TYPE'],hue=mergedf['NAME_CONTRACT_STATUS'])
plt.xticks(rotation = 45)
plt.title('Count income type wrt contract status',fontdict={'fontsize': 20})
```

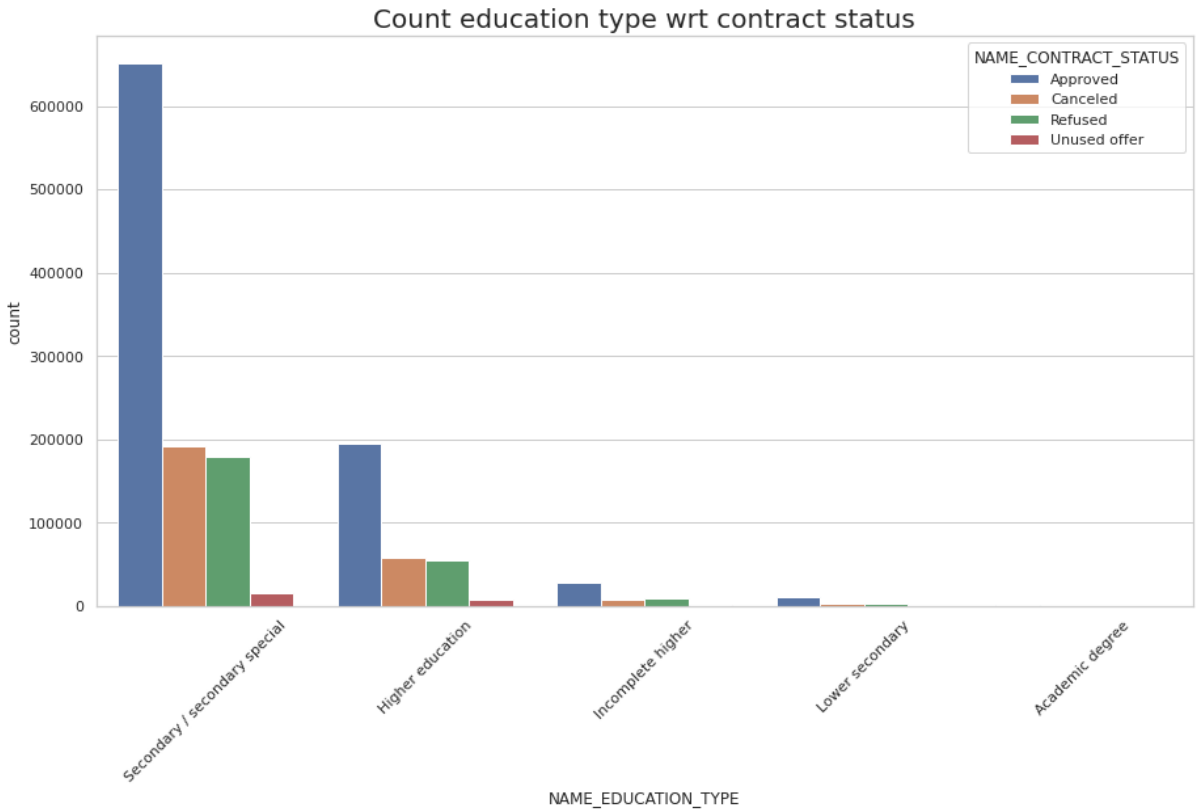


Inference:

- loans for working people are approved more.
- unemployed, student and maternity leave have very few records.

 # Univariate analysis of NAME_EDUCATION_TYPE with hue NAME_CONTRACT_STATUS

```
plt.figure(figsize = (15,8))
sns.countplot(mergedf['NAME_EDUCATION_TYPE'],hue=mergedf['NAME_CONTRACT_STATUS'])
plt.xticks(rotation = 45)
plt.title('Count education type wrt contract status',fontdict={'fontsize': 20})
```



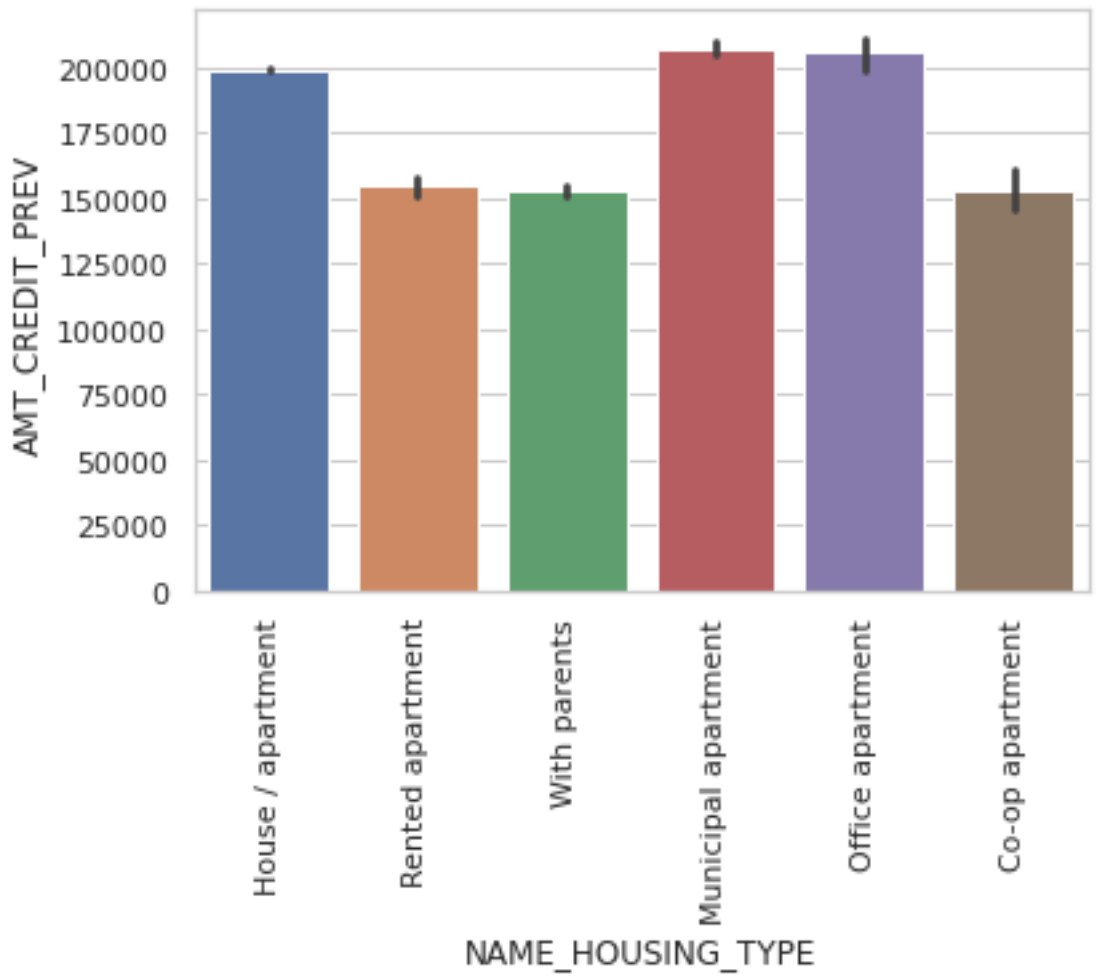
Inference:

- secondary education has the highest loan approved.
- Academic degree has very few records

16) Bivariate analysis of merged data frame

 # Bar plot between NAME_HOUSING_TYPE and AMT_CREDIT_PREV

```
sns.barplot(data=mergedf,x = 'NAME_HOUSING_TYPE',y = 'AMT_CREDIT_PREV' )
plt.xticks(rotation=90)
```



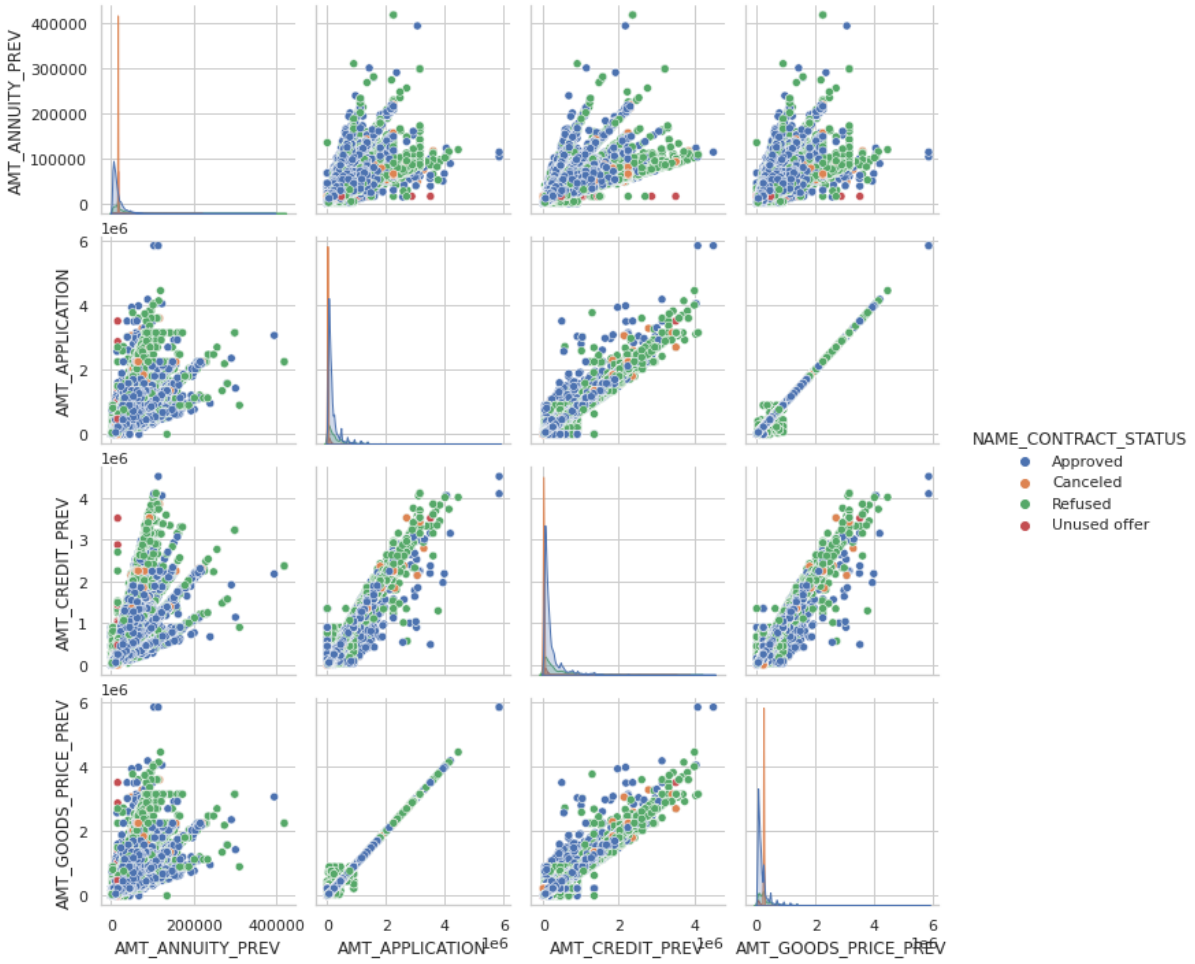
Inference:

- Municipal apartment and credit apartment have the highest amount credit.

17) Correlation check for previous_application.csv

 # Pair plots for the numeric column

```
plt.figure(figsize=(20,14))
amount = mergedf[['AMT_ANNUITY_PREV', 'AMT_APPLICATION',
      'AMT_CREDIT_PREV', 'AMT_GOODS_PRICE_PREV','NAME_CONTRACT_STATUS']]
sns.pairplot(amount,hue = 'NAME_CONTRACT_STATUS')
```



Inference:

- AMT_GOODS_PRICE_PREV and AMT_APPLICATION shows too high linear relation.

 # Extracting columns for the correlation check

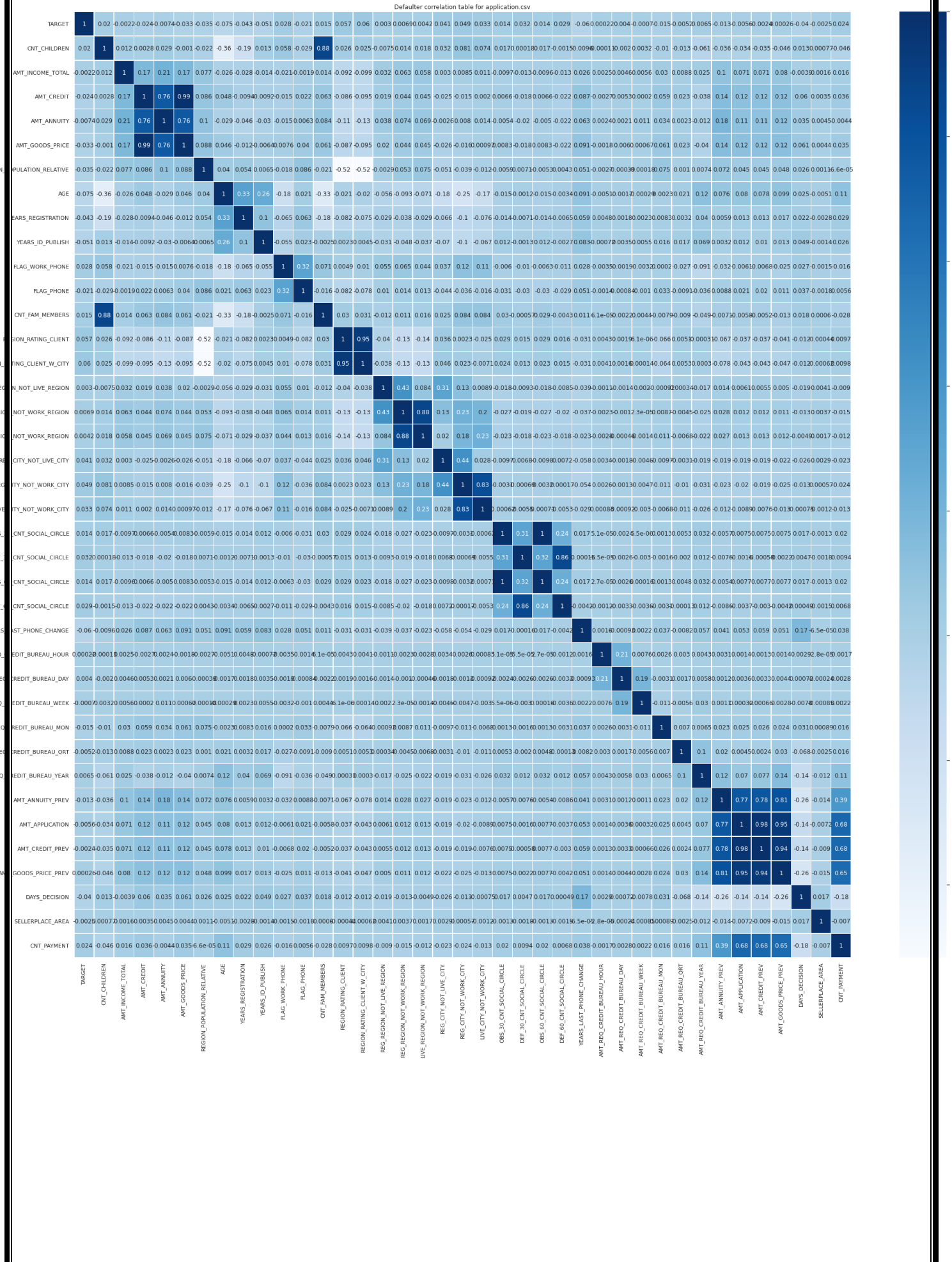
```
corr4 = mergedf[['TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
    'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
    'AMT_CREDIT', 'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE',
    'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS',
    'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE', 'AGE',
    'YEARS_REGISTRATION', 'YEARS_ID_PUBLISH', 'FLAG_WORK_PHONE',
    'FLAG_PHONE', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBERS',
    'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY',
    'REG_REGION_NOT_LIVE_REGION', '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',
    'ORGANIZATION_TYPE', 'OBS_30_CNT_SOCIAL_CIRCLE',
    'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE',
    'DEF_60_CNT_SOCIAL_CIRCLE', 'YEARS_LAST_PHONE_CHANGE',
    'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
    'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
```

```
    'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR',
    'AMT_INCOME_TOTAL_RANGE', 'AMT_CREDIT_RANGE',
    'NAME_CONTRACT_TYPE_PREV', 'AMT_ANNUITY_PREV', 'AMT_APPLICATION',
    'AMT_CREDIT_PREV', 'AMT_GOODS_PRICE_PREV', 'NAME_CASH_LOAN_PURPOSE',
    'NAME_CONTRACT_STATUS', 'DAYS_DECISION', 'NAME_PAYMENT_TYPE',
    'CODE_REJECT_REASON', 'NAME_CLIENT_TYPE', 'NAME_GOODS_CATEGORY',
    'NAME_PORTFOLIO', 'NAME_PRODUCT_TYPE', 'CHANNEL_TYPE',
    'SELLERPLACE_AREA', 'NAME_SELLER_INDUSTRY', 'CNT_PAYMENT',
    'NAME_YIELD_GROUP', 'PRODUCT_COMBINATION']].corr()
```

 # Plot correlation map

```
plt.figure(figsize=(35,35))
sns.heatmap(corr4,cmap='Blues',annot=True,linewidth=0.5)
plt.title('Defaulter correlation table for application.csv')
corr5 = corr4.where(np.triu(np.ones(corr4.shape), k=1).astype(bool))
corr_mat = corr5.unstack().sort_values(ascending=False).reset_index()
corr_mat.rename(columns={'level_0':'Var1','level_1':'Var2',0:'Correlation'},inplace=True )
print(corr_mat.head(10))
```

```
                Var1                    Var2        Correlation
0    OBS_60_CNT_SOCIAL_CIRCLE    OBS_30_CNT_SOCIAL_CIRCLE     0.998561
1           AMT_GOODS_PRICE              AMT_CREDIT     0.985959
2           AMT_CREDIT_PREV           AMT_APPLICATION     0.975683
3       AMT_GOODS_PRICE_PREV           AMT_APPLICATION     0.945759
4  REGION_RATING_CLIENT_W_CITY       REGION_RATING_CLIENT     0.945596
5       AMT_GOODS_PRICE_PREV           AMT_CREDIT_PREV     0.939147
6           CNT_FAM_MEMBERS             CNT_CHILDREN     0.879224
7  LIVE_REGION_NOT_WORK_REGION  REG_REGION_NOT_WORK_REGION     0.875505
8    DEF_60_CNT_SOCIAL_CIRCLE    DEF_30_CNT_SOCIAL_CIRCLE     0.862698
9     LIVE_CITY_NOT_WORK_CITY      REG_CITY_NOT_WORK_CITY     0.831422
```


Defaulter correlation table for application.csv
TARGET
CNT_CHILDREN
AMT_INCOME_TOTAL
AMT_CREDIT
AMT_ANNUITY
AMT_GOODS_PRICE
REGION_POPULATION_RELATIVE
AGE
YEARS_REGISTRATION
YEARS_ID_PUBLISH
FLAG_WORK_PHONE
FLAG_PHONE
CNT_FAM_MEMBERS
REGION_RATING_CLIENT
REGION_RATING_CLIENT_W_CITY
REG_REGION_NOT_LIVE_REGION
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
OBS_30_CNT_SOCIAL_CIRCLE
DEF_30_CNT_SOCIAL_CIRCLE
OBS_60_CNT_SOCIAL_CIRCLE
DEF_60_CNT_SOCIAL_CIRCLE
YEARS_LAST_PHONE_CHANGE
AMT_REQ_CREDIT_BUREAU_HOUR
AMT_REQ_CREDIT_BUREAU_DAY
AMT_REQ_CREDIT_BUREAU_WEEK
AMT_REQ_CREDIT_BUREAU_MON
AMT_REQ_CREDIT_BUREAU_QRT
AMT_REQ_CREDIT_BUREAU_YEAR
AMT_ANNUITY_PREV
AMT_APPLICATION
AMT_CREDIT_PREV
AMT_GOODS_PRICE_PREV
DAYS_DECISION
SELLERPLACE_AREA
CNT_PAYMENT
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

Conclusion:

• The data is highly imbalance with almost 92% repayor and 8% defaulter.

• It is found that revolving loans has less defaulter (5.5%) than cash loans (8.2%).

• Unemployed and Maternity leave, income types have the highest defaulter 36.5% and 40% respectively. Loans giving to this category should be avoided.

• Academic degree in education type has the minimum defaulter rate. Loan distribution to this category can be increased.

• Person who owns car have the less defaulter rate than who does not own car.

• Banks should focus more on contract type ‘Student’, ‘pensioner’ and ‘Businessman’ with housing ‘type other than ‘Co-op apartment’ and 'office apartment' for successful payments.

• OCCUPATION_TYPE: Avoid Low-skill Laborers, Drivers and Waiters/barmen staff, Security staff, Laborers and Cooking staff as their default rate is huge.

• AMT_GOODS_PRICE: When the credit amount goes beyond 3lakhs, there is an increase in defaulters.

Challenges:

- The Dataset contains 122 features in application_data.csv and 32 features in previous_application.csv which makes it complicated to find the relation between the dataset.
- The processing time while making pair plots and correlation plots is quite high because of huge dataset.

Module 7: XYZ Ads Airing Report Analysis

Project Description:

Advertising is a way of marketing your business in order to increase sales or make your audience aware of your products or services. Until a customer deal with you directly and actually buys your products or services, your advertising may help to form their first impressions of your business. Target audience for businesses could be local, regional, national or international or a mixture. So, they use different ways for advertisement. Some of the types of advertisement are: Internet/online directories, Trade and technical press, Radio, Cinema, Outdoor advertising, National papers, magazines and TV. Advertising business is very competitive as a lot of players bid a lot of money in a single segment of business to target the same audience. Here comes the analytical skills of the company to target those audiences from those types of media platforms where they convert them to their customers at a low cost.

The dataset having different TV Airing Brands, their product, their category. Dataset includes the network through which Ads are airing, types of networks like Cable/ Broadcast and the show name also on which Ads got aired. You can also see the data of Dayparts, Time zone and the time & date at which Ads got aired. IT also includes other data like Pod Position (the lesser the valuable), duration for which Ads aired on screen, Equivalent sales &, total amount spent on the Ads aired.

Approach:

Downloading the data set and performing EDA to understand the data set. Checked for the null values and the distribution. Defining the problem and to analyse each task first noted the features to be used. Now to get result I checked all the functions which will be required to perform the operations. Create charts and graphs to define underlying aspects of the data. Finally created a report consisting the description, approach, result, insights, conclusion, etc.

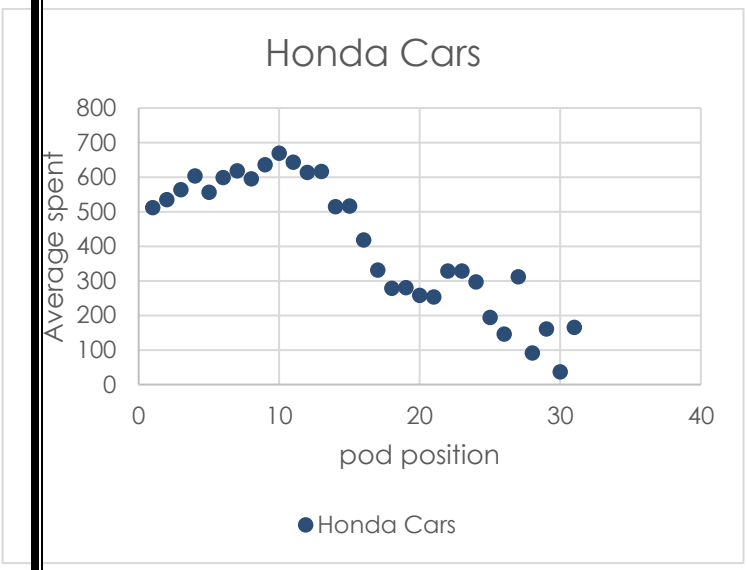
Tech-Stack Used:

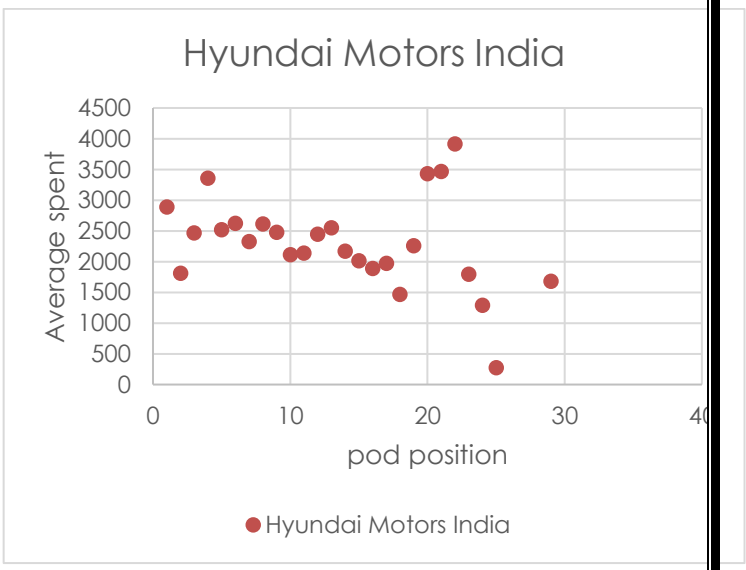
For this assignment I have used Microsoft Excel (2016).

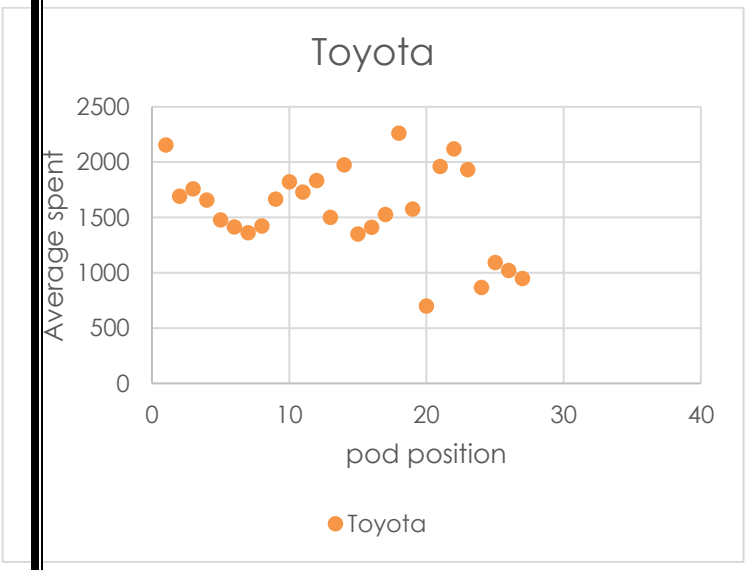
Insights & results:

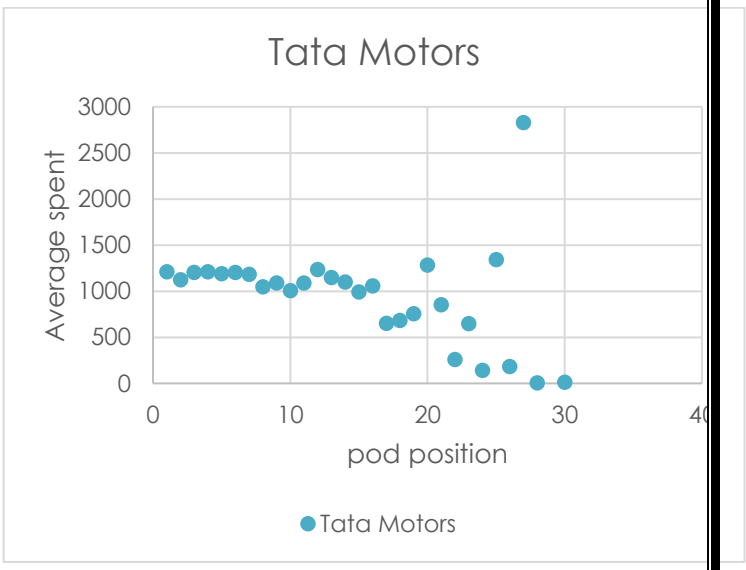
A) What is Pod Position? Does the Pod position number affect the amount spent on Ads for a specific period of time by a company? (Explain in Details with examples from the dataset provided)

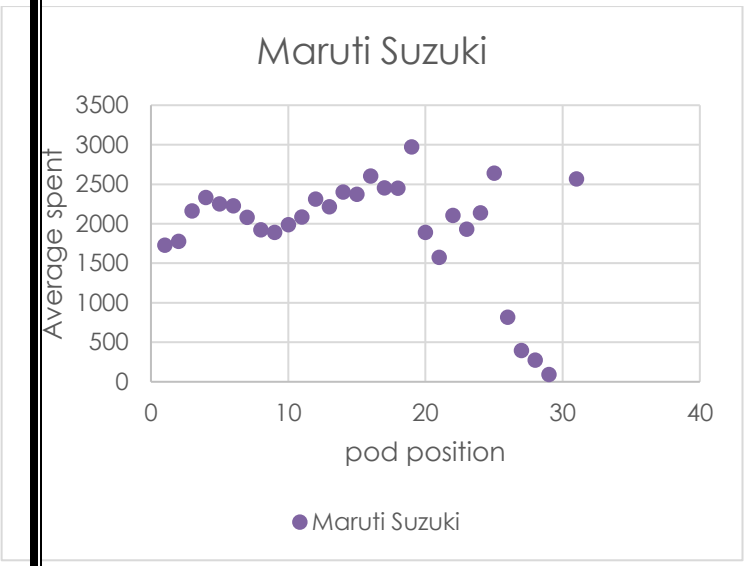
Pod Position: Pod position refers to the order in which the advertisements have shown.

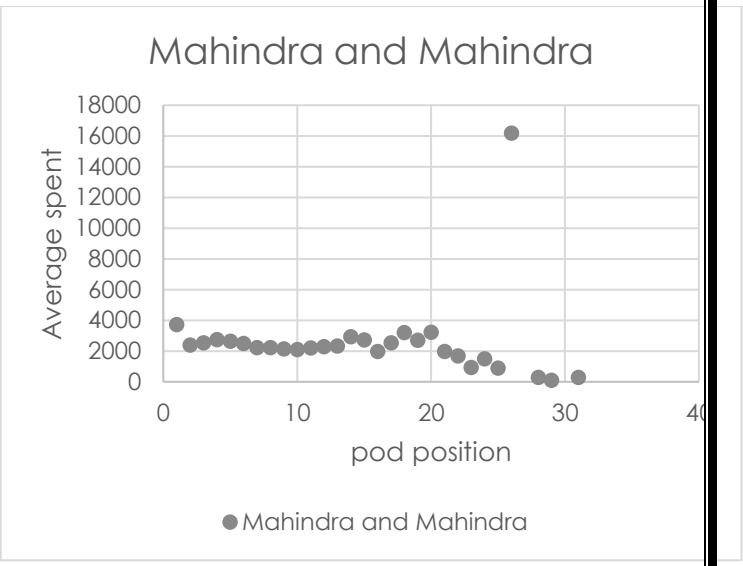










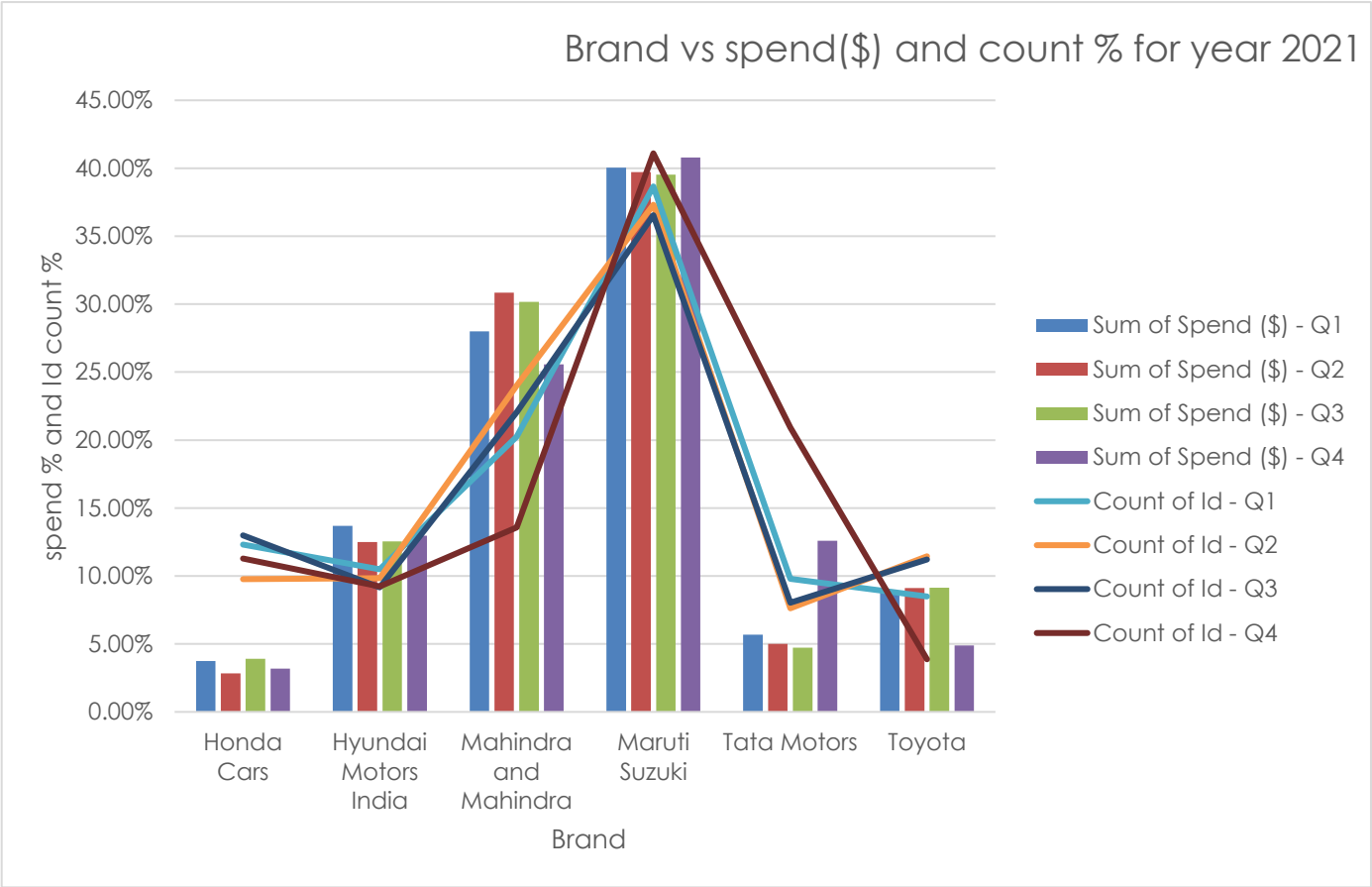


- From the following graphs it can be concluded that the for the smaller value of the pod position the value of average spent is high (Toyota and Hyundai Motors India).
- For some companies like Maruti Suzuki and Honda Cars first few positions have lesser value than the average spent increases.
- For companies like Mahindra and Mahindra and Tata Motors there is no much changes from value 1-15 in pod position.
- In all the plots it is clearly seen that the pod position with high value is less costly than the pod position with low values.

B) What is the share of various brands in TV airings and how has it changed from Q1 to Q4 in 2021?

	Column Labels									
	Sum of Spend ($)				Count of Id				Total Sum of Spend	Total Count of Id
Row Labels	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4		
Honda Cars	3.75%	2.82%	3.90%	3.18%	12.32%	9.77%	12.99%	11.29%	3.45%	11.61%
Hyundai Motors India	13.68%	12.51%	12.55%	12.97%	10.49%	9.84%	9.17%	9.23%	12.99%	9.74%
Mahindra and Mahindra	28.01%	30.84%	30.18%	25.57%	20.25%	24.01%	22.05%	13.57%	28.67%	20.37%
Maruti Suzuki	40.04%	39.71%	39.53%	40.80%	38.66%	37.31%	36.55%	41.10%	40.00%	38.26%
Tata Motors	5.68%	5.01%	4.72%	12.60%	9.79%	7.62%	8.03%	20.93%	6.74%	10.99%
Toyota	8.85%	9.12%	9.13%	4.89%	8.49%	11.45%	11.21%	3.87%	8.15%	9.04%
Grand Total	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**

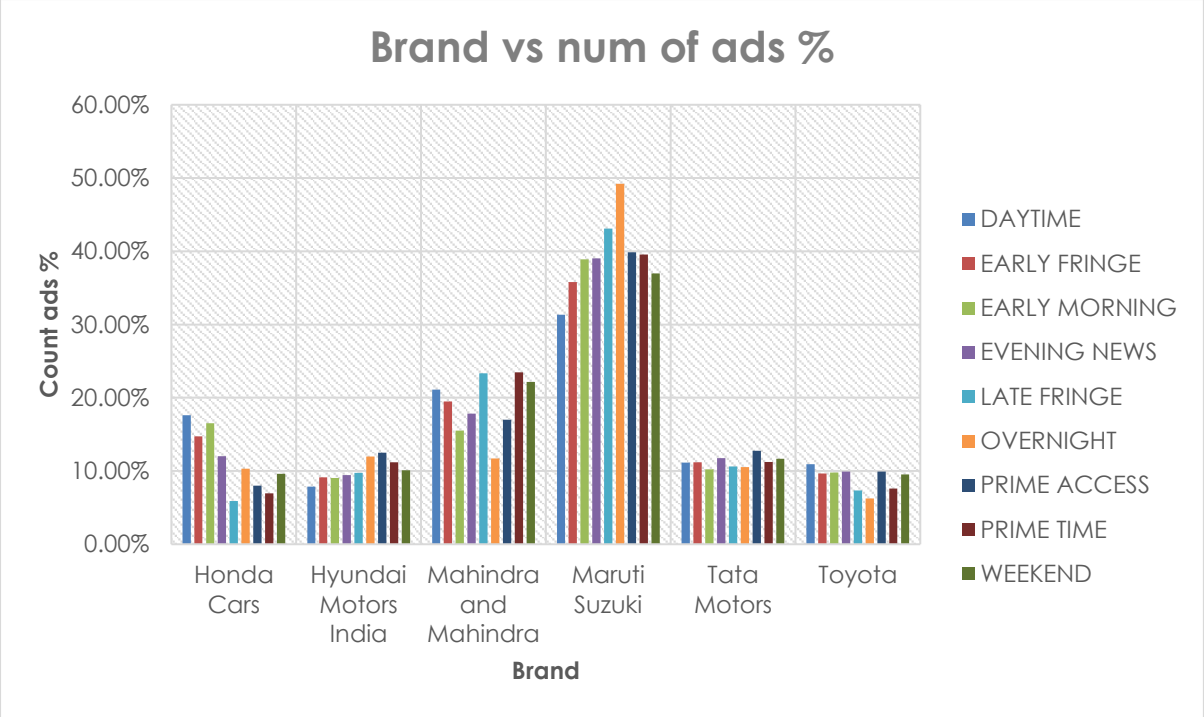
The Table shows the percentage share of each of the companies in advertisement on the basis of investment and number of ads.

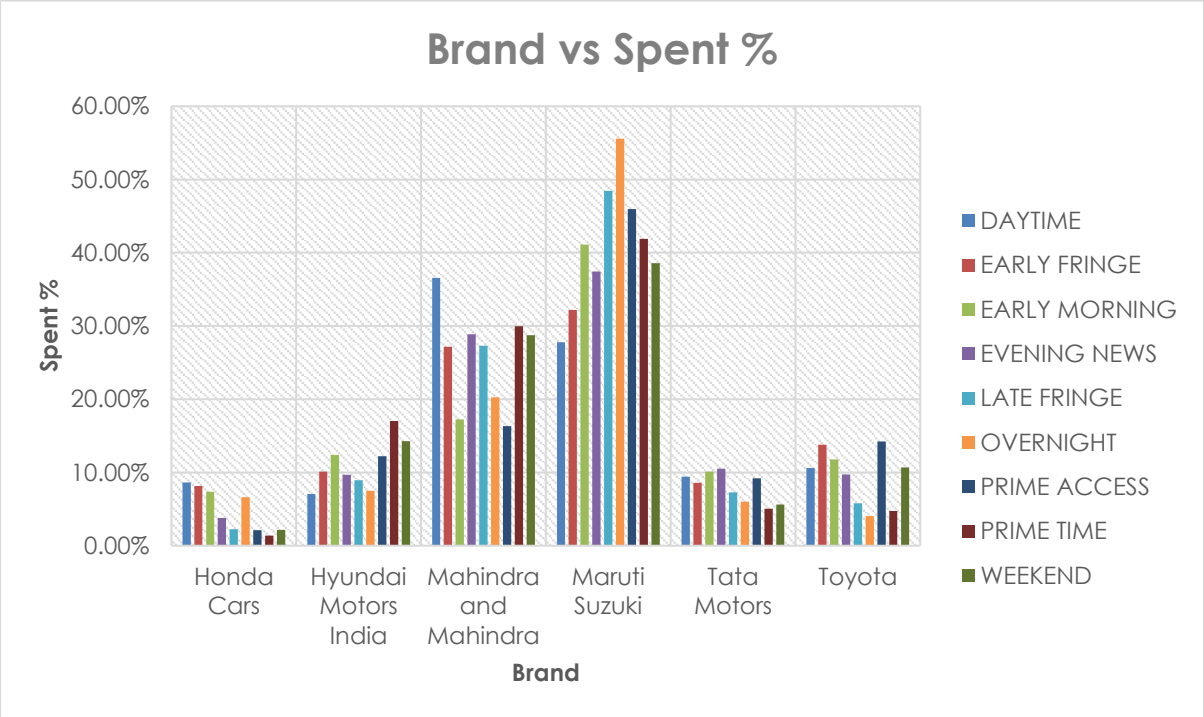


- Bars represent the sum of spend in percentage and lines represents the number of ads in percentage.
- For Maruti Suzuki have the maximum number of ads and contribute maximum share.
- Tata motors has increased its spent share in Q4.
- For Mahindra and Mahindra, the number of ads is less in Q2, it indicates that it invested money on getting lower pod position or ads aired at the prime time.
- Honda car's share is minimum.

C) Conduct a competitive analysis for the brands and define advertisement strategy of different brands and how it differs across the brands.

Company strategy can be of more than one features. Here I have shown the analysis on two features Id and Spent based on the Daypart.

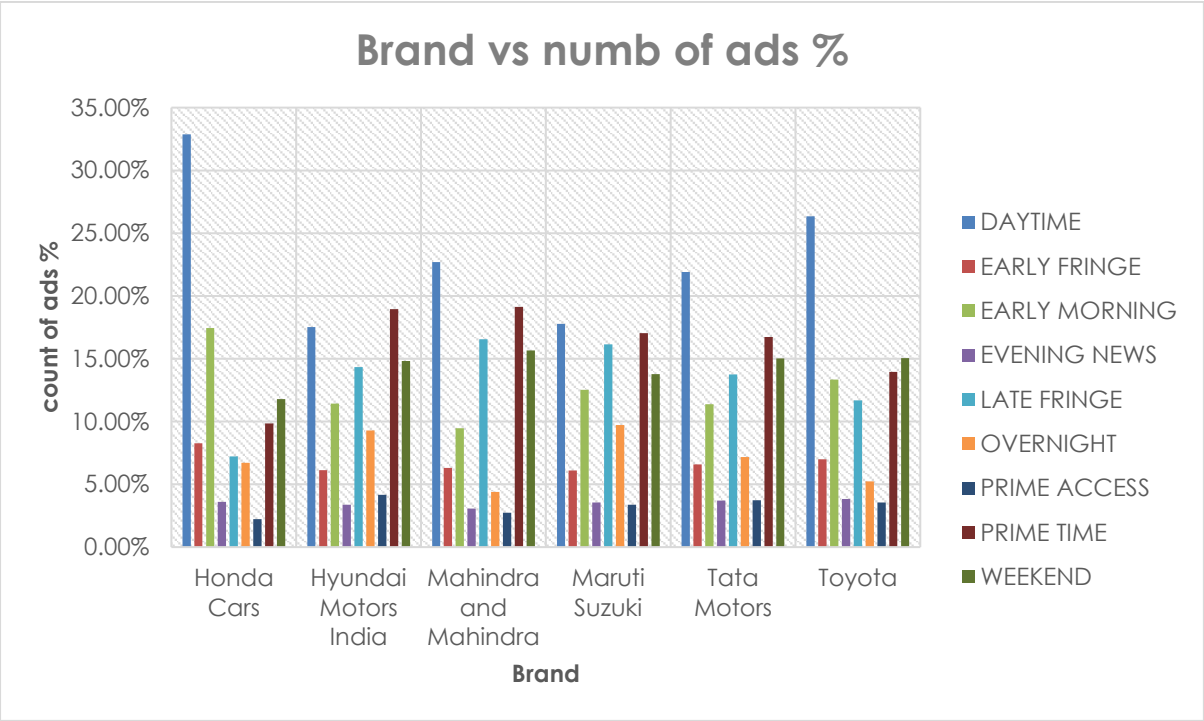




Figures 1: represent the relative spent and number of ads for each company.

- From all the companies Maruti Suzuki which spent maximum of all the companies, also spent maximum in each daypart except Daytime.
- Mahindra and Mahindra Spend maximum at the day time but numb of ads aired is less than Maruti Suzuki.
- Numb of ads aired of Tata Motors in each day part is almost same.

- Honda Cars spent minimum of all the companies.



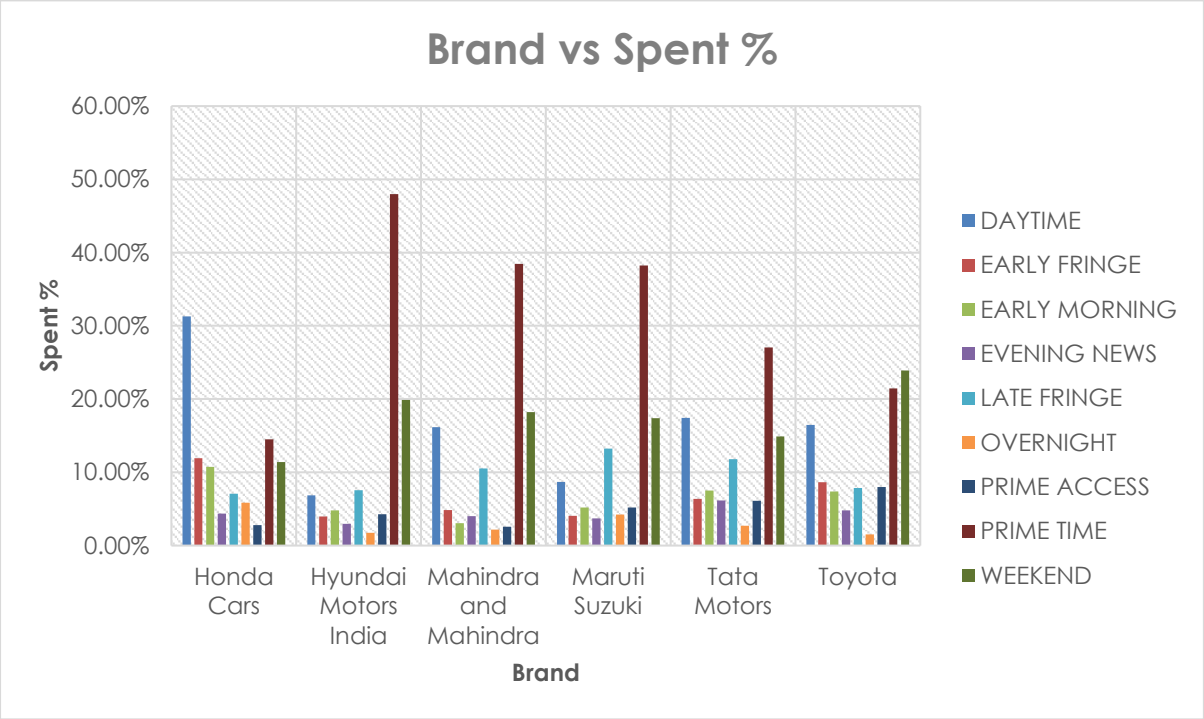


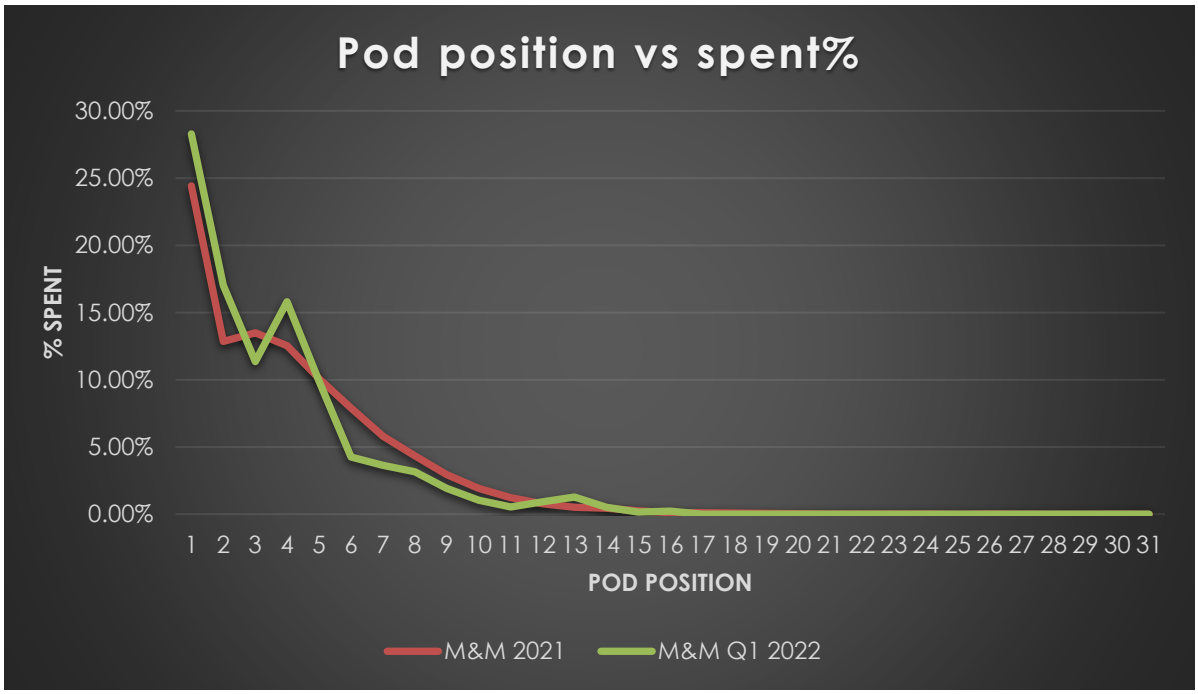
Figure 2: represents the spent% and count of ads% of a particular company at different dayparts.

- All the companies except Honda Cars and Toyota spent maximum for the prime time.
- Honda Cars which spent maximum for the daytime also air its maximum ads at the daytime.

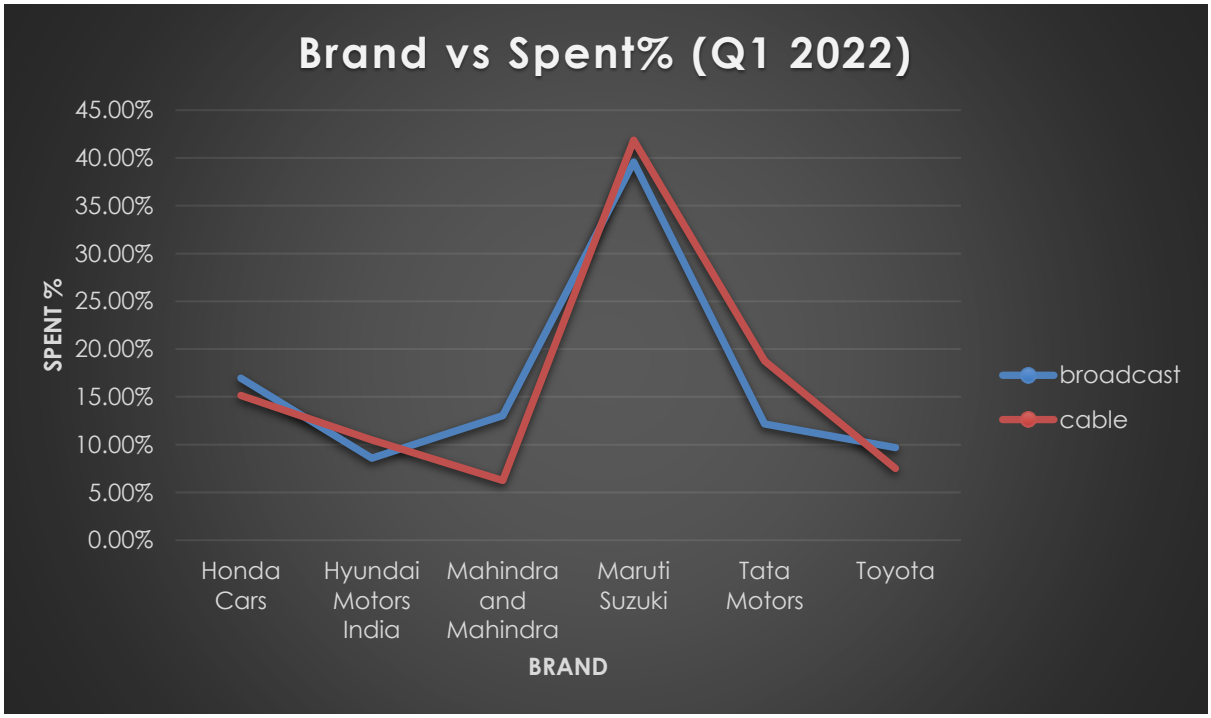
D) Mahindra and Mahindra want to run a digital ad campaign to complement its existing TV ads in Q1 of 2022. Based on the data from 2021, suggest a media plan to the CMO of Mahindra and Mahindra. Which audience should they target? *Assume XYZ Ads has the ad viewership data and TV viewership for the people in India.

There can multiple features:

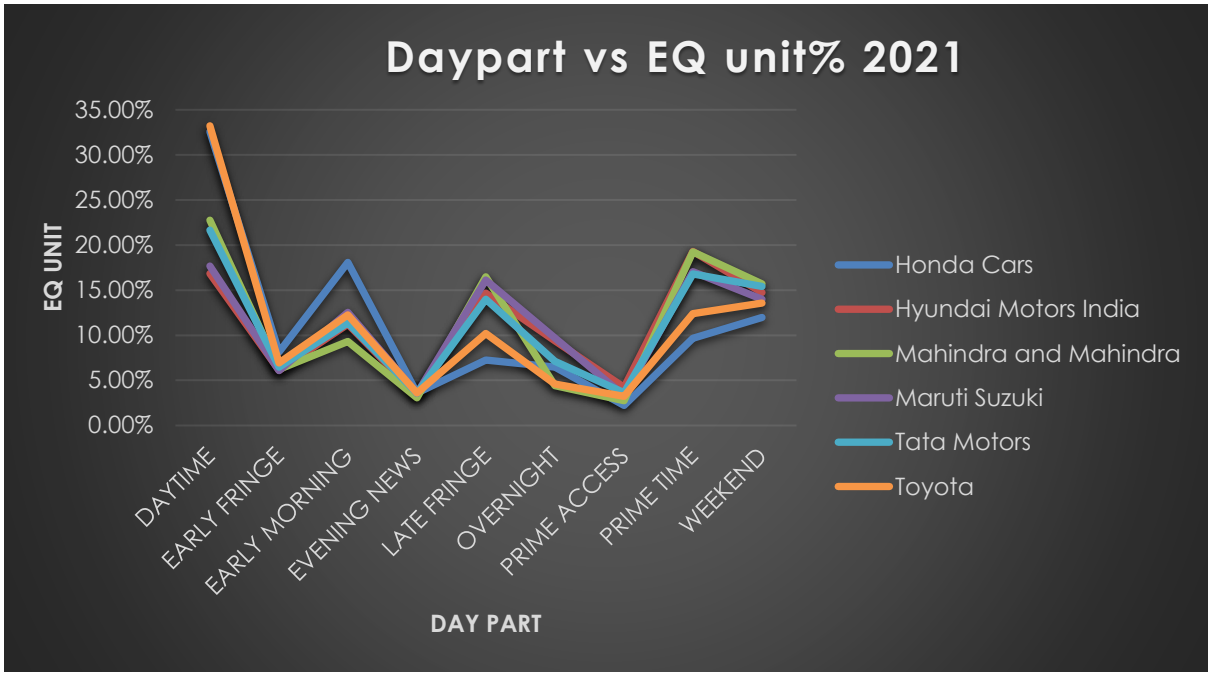
1) **Pod position vs spent% direct comparison:**

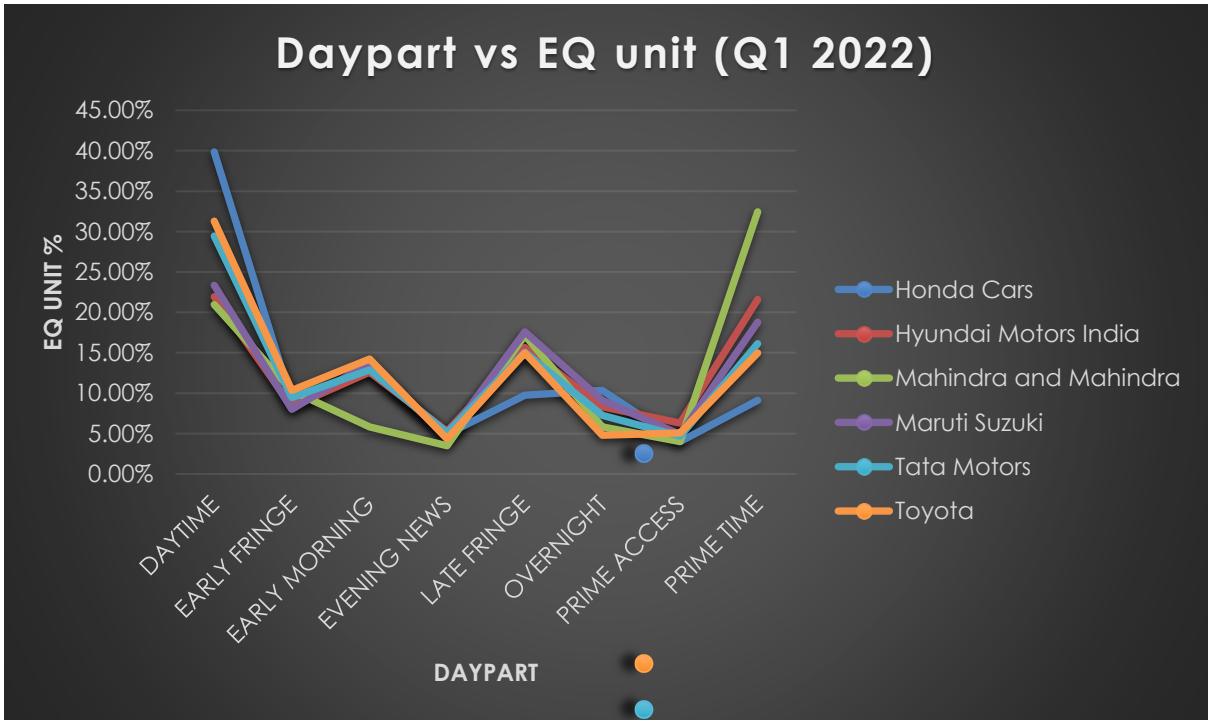


2) **Pod position vs spent analysis of each of the company for year 2021 and 2022 Q1:**

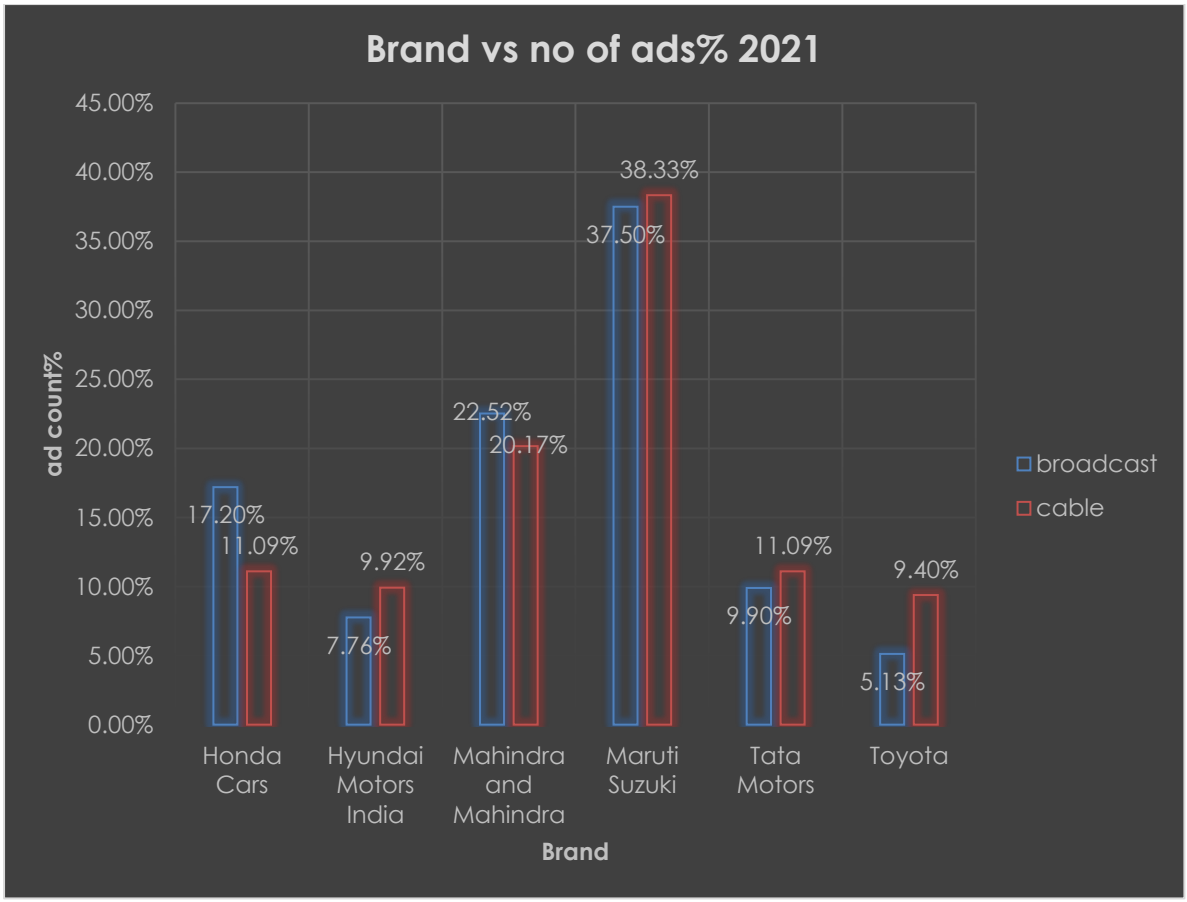


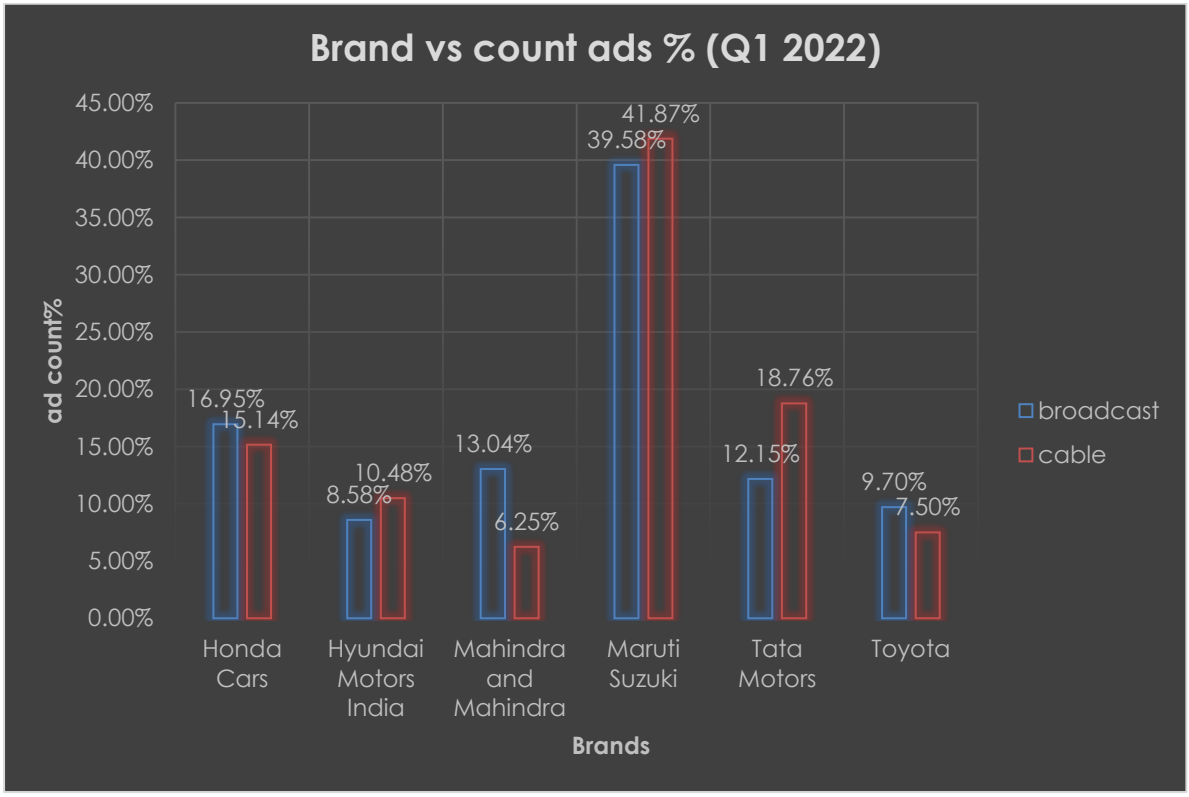
3) **Daypart vs EQ unit% for 2021 and 2022 Q1 for each company.**





4) Brand vs number of ads% of 2021 and 2022 Q1 for each company.


Brand vs no of ads% 2021
45.00%
40.00%
35.00%
30.00%
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
ad count%
17.20%
11.09%
7.76%
9.92%
22.52%
20.17%
37.50%
38.33%
9.90%
11.09%
5.13%
9.40%
Honda Cars
Hyundai Motors India
Mahindra and Mahindra
Maruti Suzuki
Tata Motors
Toyota
Brand
broadcast
cable


Brand vs count ads % (Q1 2022)
45.00%
40.00%
35.00%
30.00%
25.00%
20.00%
15.00%
10.00%
5.00%
0.00%
ad count%
16.95%
15.14%
8.58%
10.48%
13.04%
6.25%
39.58%
41.87%
12.15%
18.76%
9.70%
7.50%
Honda Cars
Hyundai Motors India
Mahindra and Mahindra
Maruti Suzuki
Tata Motors
Toyota
Brands
broadcast
cable

Inferences:

1. M&M spent Maximum in broadcast in Prime-time follower by daytime

2. In cable it also spent maximum at the prime time followed by daytime.

3. The major share of EQ unit is also at the prime time followed by the daytime.

4. M&M showed its ad in first 5 pod positions.

5. M&M's maximum share of EQ unit lies in day time for the year 2021 followed by prime time and late fringe.

6. at Day time companies Toyota and Honda cars having share of EQ unit more than M&M.

7. at early morning, M&M's EQ unit share is minimum of all the companies.

8. Maruti Suzuki holds the maximum EQ unit share followed by M&M.

9. Honda cars spent its maximum at daytime followed by tata motors, Toyota and M&M.

10. Three companies Hyundai motors India, Maruti Suzuki and M&M spent their maximum at the prime time.

11. M&M spent maximum at the day time whereas Maruti Suzuki spent its maximum for overtime, late Frings, prime access and prime time although it shares 40% of the market share.

12. Maruti Suzuki which contribute 40% of the ads market shows maximum of its ads at lower pod position by paying higher followed by M&M.

Challenges:

The project consists of the deep knowledge of the ads airing and how companies' tract their ads and get a balanced spent, numb of ads ratio at every daytime. There were some terms which exclusively used in the field of ads.

Module 8: ABC Call Volume Trend Analysis

Project Description:

A customer experience (CX) team consists of professionals who analyse customer feedback and data, and share insights with the rest of the organization. Typically, these teams fulfil various roles and responsibilities such as: Customer experience programs (CX programs), Digital customer experience, Design and processes, Internal communications, Voice of the customer (VoC), User experiences, Customer experience management, Journey mapping, Nurturing customer interactions, Customer success, Customer support, Handling customer data, Learning about the customer journey.

Advertising is a way of marketing your business in order to increase sales or make your audience aware of your products or services. Until a customer deal with you directly and actually buys your products or services, your advertising may help to form their first impressions of your business. Target audience for businesses could be local, regional, national or international or a mixture. So, they use different ways for advertisement. Some of the types of advertisement are: Internet/online directories, Trade and technical press, Radio, Cinema, Outdoor advertising, National papers, magazines and TV. Advertising business is very competitive as a lot of players bid a lot of money in a single segment of business to target the same audience. Here comes the analytical skills of the company to target those audiences from those types of media platforms where they convert them to their customers at a low cost.

Approach:

Downloading the data set and performing EDA to understand the data set. Checked for the null values and the distribution. Defining the problem and to analyse each task first noted the features to be used. Now to get result I checked all the functions which will be required to perform the operations. Create charts and graphs to define underlying aspects of the data. Finally created a report consisting the description, approach, result, insights, conclusion, etc.

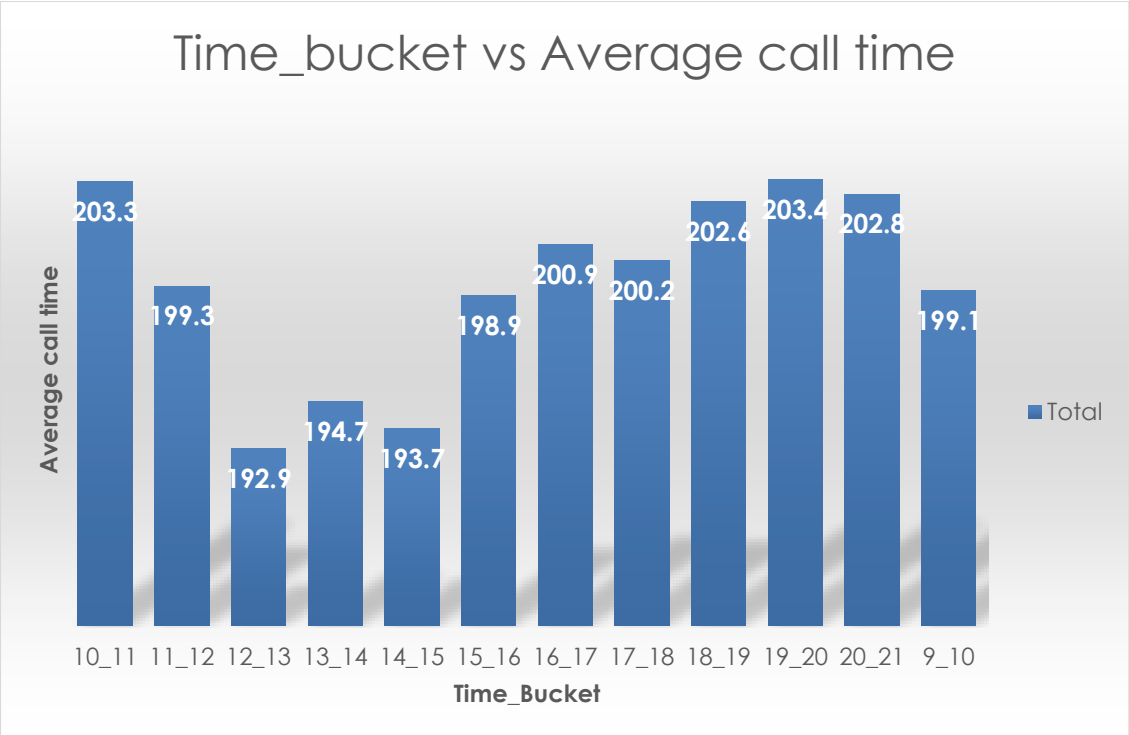
Tech-Stack Used:

For this assignment I have used Microsoft Excel (2016).

Insights & results:

A) Calculate the average call time duration for all incoming calls received by agents (in each Time_Bucket).

Row Labels	Average of Call_Seconds (s)
10_11	203.3
11_12	199.3
12_13	192.9
13_14	194.7
14_15	193.7
15_16	198.9
16_17	200.9
17_18	200.2
18_19	202.6
19_20	203.4
20_21	202.8
9_10	199.1
Grand Total	**198.62**

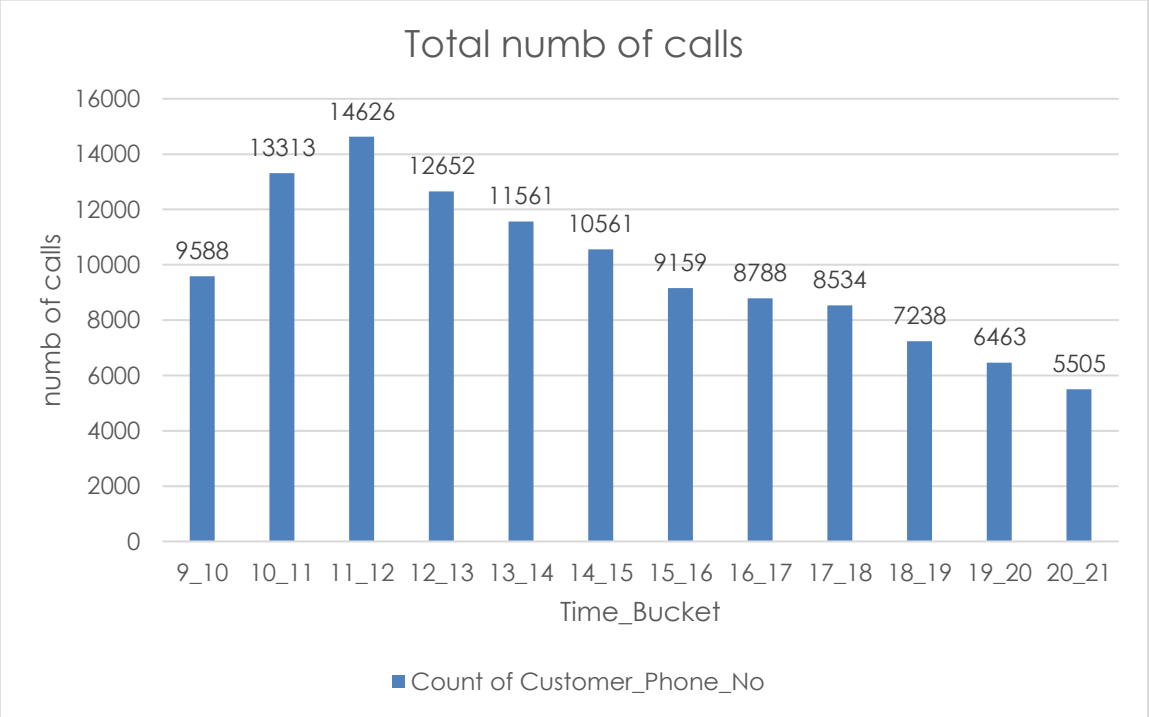


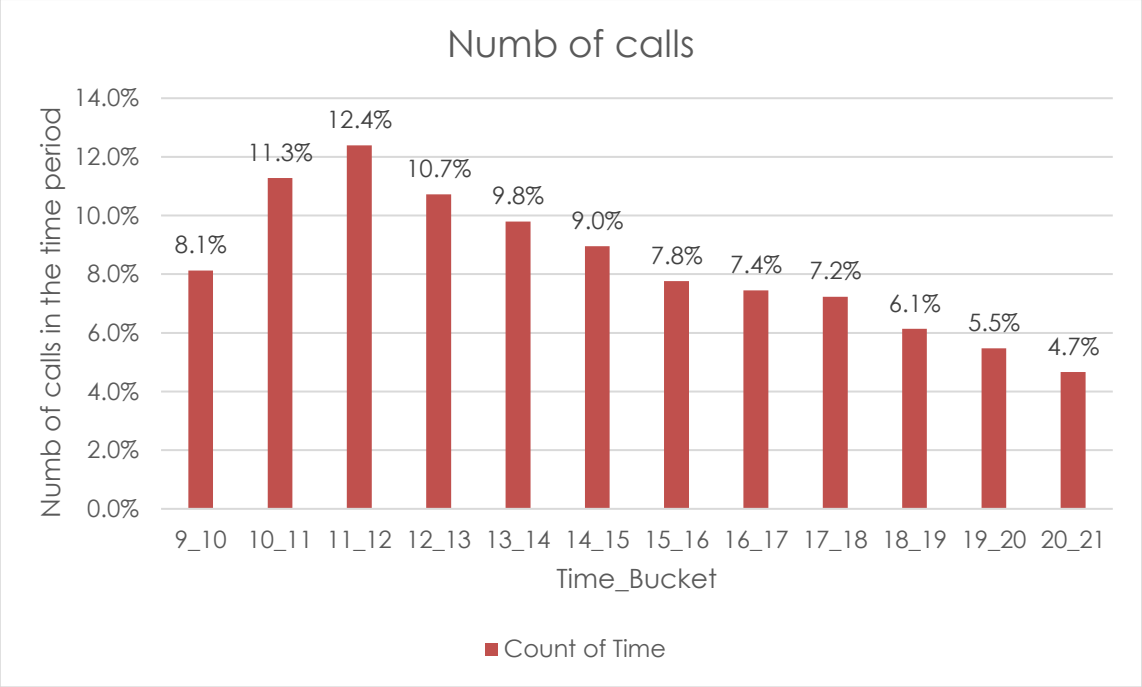
- Minimum average call time is for time bucket 12_13.

- Total average time to answer each call is 198.62 sec.

B) Show the total volume/ number of calls coming in via charts/ graphs [Number of calls v/s Time]. You can select time in a bucket form (i.e., 1-2, 2-3, ….)

Row Labels	Count of Customer_Phone_No	Count of Time
10_11	13313	11.28%
11_12	14626	12.40%
12_13	12652	10.72%
13_14	11561	9.80%
14_15	10561	8.95%
15_16	9159	7.76%
16_17	8788	7.45%
17_18	8534	7.23%
18_19	7238	6.13%
19_20	6463	5.48%
20_21	5505	4.67%
9_10	9588	8.13%
Grand Total	**117988**	**100.00%**





C) As you can see current abandon rate is approximately 30%. Propose a manpower plan required during each time bucket [between 9am to 9pm] to reduce the abandon rate to 10%. (i.e., You have to calculate minimum number of agents required in each time bucket so that at least 90 calls should be answered out of 100.)

Date	abandon	answered	transfer	Grand Total
01-Jan	684	3883	77	4644
02-Jan	356	2935	60	3351
03-Jan	599	4079	111	4789
04-Jan	595	4404	114	5113
05-Jan	536	4140	114	4790
06-Jan	991	3875	85	4951
07-Jan	1319	3587	42	4948
08-Jan	1103	3519	50	4672
09-Jan	962	2628	62	3652
10-Jan	1212	3699	72	4983
11-Jan	856	3695	86	4637
12-Jan	1299	3297	47	4643
13-Jan	738	3326	59	4123
14-Jan	291	2832	32	3155
15-Jan	304	2730	24	3058
16-Jan	1191	3910	41	5142
17-Jan	16636	5706	5	22347

18-Jan	1738	4024	12	5774
19-Jan	974	3717	12	4703
20-Jan	833	3485	4	4322
21-Jan	566	3104	5	3675
22-Jan	239	3045	7	3291
23-Jan	381	2832	12	3225
Grand Total	34403	82452	1133	117988
Percentage total	29%	70%	1%	100%

Numb of calls on daily basis:

Date	abandon	answered	transfer	Grand Total
numb of calls on daily basis	1495.8	3584.9	49.3	5129.9

From the Question A we know that the average call duration is 198.62 sec.

So total time to answer 90% of the calls will be:

$$total\ time = total\ numb\ of\ calls * average\ call\ duration$$

time required to answer 90% calls per day	254.7001826

As one person works for 4.5 hours the total numb of persons required to answer 90% of calls will be:

$$numb\ of\ persons = \frac{tme\ required\ to\ answer\ 90\%\ of\ the\ calls}{hours\ for\ a\ person\ works}$$

numb of working persons	57

So, to answer 90% of the calls we will require 57 agents.

D) Let's say customers also call this ABC insurance company in night but didn't get answer as there are no agents to answer, this creates a bad customer experience for this Insurance company. Suppose every 100 calls that customer made during 9 Am to 9 Pm, customer also made 30 calls in night between interval [9 Pm to 9 Am] and distribution of those 30 calls are as follows:

Distribution of 30 calls coming in night for every 100 calls coming in between 9am - 9pm (i.e. 12 hrs slot)											
9pm- 10pm	10pm - 11pm	11pm- 12am	12am- 1am	1am - 2am	2am - 3am	3am - 4am	4am - 5am	5am - 6am	6am - 7am	7am - 8am	8am - 9am
3	3	2	2	1	1	1	1	3	4	4	5

Now propose a manpower plan required during each time bucket in a day. Maximum Abandon rate assumption would be same 10%.

Assumption: An agent work for 6 days a week; On an average total unplanned leaves per agent is 4 days a month; An agent total working hrs is 9 Hrs out of which 1.5 Hrs goes into lunch and snacks in the office. On average an agent occupied for 60% of his total actual working Hrs (i.e., 60% of 7.5 Hrs) on call with customers/ users. Total days in a month is 30 days.

	Values	Formulas
Daily call volume from (9AM-9PM)	5129.9	Average of the total numb of calls per day
Total calls from (9PM-9AM)	1539.0	Average of the total numb of calls per night
Total night time required	76.4	night time to answer 90% of calls(hours) = $= \frac{\text{numb of calls at night} * \text{average call duration} * 0.9}{3600}$
Additional man power needed	17	Additional man power = $\frac{\text{total night time}}{\text{hours for a agent works}}$
Total man power needed	74	Total man power = agents on day time + agents on night time

Total agents needed to answer night calls is 17.

Challenges:

The project consists of the deep knowledge of the call volume trend and analysis of the company. Also, it includes the prediction of number of agents at day and night time which was a challenging part.

Appendix

The Google drive links of the following projects is given below:

1. Data Analytics Process - Link
2. Instagram User Analytics - Link
3. Operation & Metric Analytics - Link
4. Hiring Process Analytics - Link
5. IMDB Movie Analysis - Link
6. Bank Loan Case Study - Link
7. XYZ Ads Airing Report - Link
8. ABC Call Volume Trend - Link