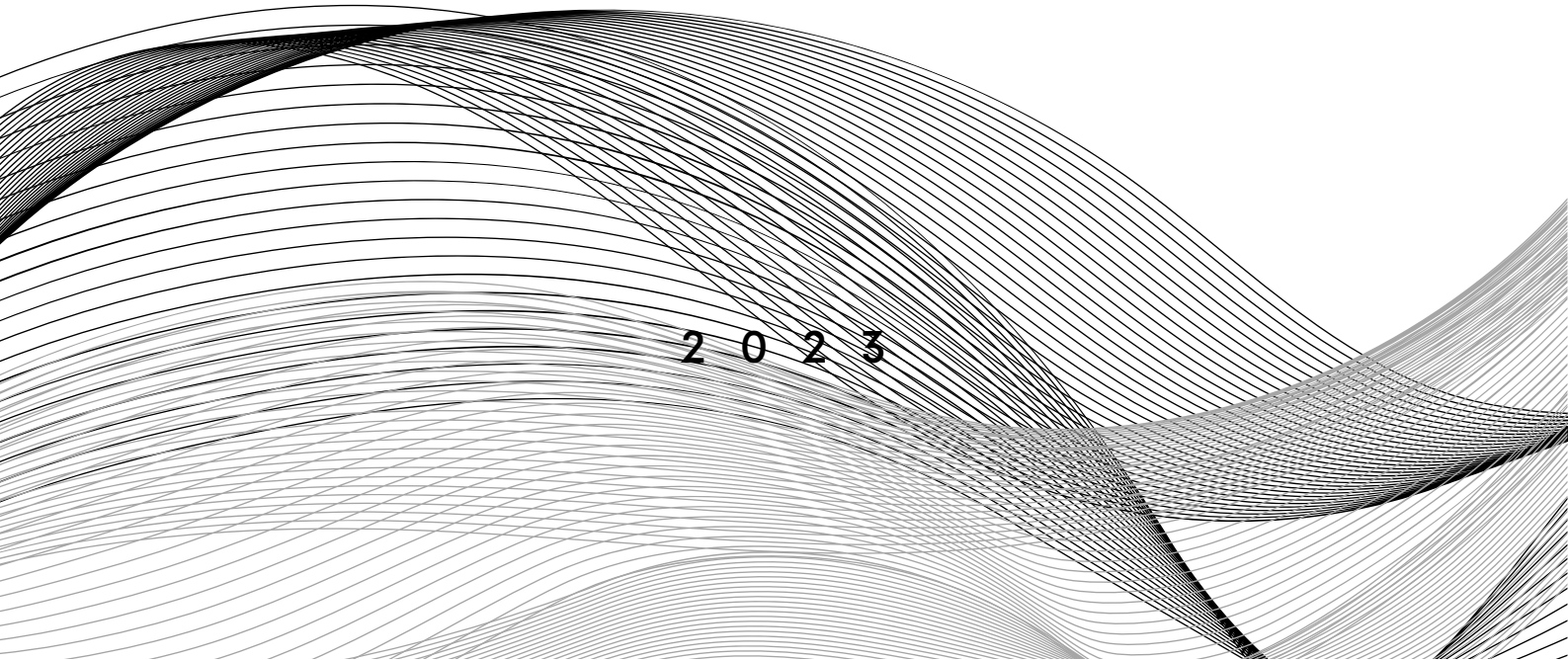
REINE D BHARALI

DATA ANALYST

PORTFOLIO



P R O F E S S I O N A L

B A C K G R O U N D

A dynamic professional with work experience in technical recruitment. With a diverse background in MBA in Human Resource and Marketing and BE in Electronics and Communication, I have a strong foundation in both business and technical disciplines.

As a former Technical Recruiter, I have gained valuable experience in identifying top talent and understanding key technical tools & requirements. I have a keen eye for detail and a deep understanding of the technical skills required for success in various roles.

Driven by my passion for data analysis, I made a successful transition to my current role as a Data Analyst Trainee at Trainity. My creative mindset and analytical skills enable me to provide valuable insights that drive informed business decisions. I have quickly continued to develop my skills in this exciting field.

Overall, I am a versatile and innovative professional with a proven track record of success in both technical recruiting and data analysis. I am committed to ongoing learning and growth and am eager to contribute my skills and expertise to any organization I work with.

TABLE OF CONTENTS

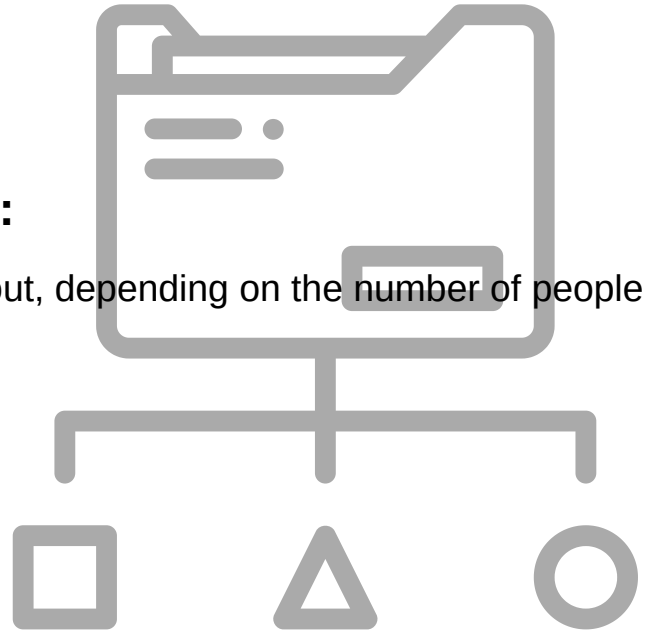
I. Data Analytics Process

Project Description:

This project describes the use of data analytics in everyday life and requires providing a real-life example of how data analytics is used or can be used in a situation and linking it with the data analytics process.

The Problem:

Deciding a Dine out, depending on the number of people.



Design:

By breaking down the stages into these we can create an efficient analysis
Plan - Prepare - Process - Analyze - Share - Act

Conclusion:

Data Lifecycle consists of these key steps which help in Data Analytics to generate insights. However, they can also be applied in everyday life to accomplish personal or professional goals. For example, an individual can use these steps to plan, prepare, process, analyze, share, and act on their job search strategy to plan their daily routines to improve productivity and achieve personal goals.

II. Instagram User Analytics

Project Description:

User analysis tracks how people interact with our digital products, providing insights for marketing and product development teams. These insights help launch marketing campaigns, build new app features, measure user engagement, improve user experience, and grow the business. As part of the Instagram product team, the product manager has asked to provide insights to the management team.

Findings:

5 oldest users of the Instagram from the database provided

id	username	created_at
80	Darby_Herzog	06-05-2016 00:14
67	Emilio_Bernier52	06-05-2016 13:04
63	Elenor88	08-05-2016 01:30
95	Nicole71	09-05-2016 17:30
38	Jordyn.Jacobson2	14-05-2016 07:56

users who have never posted a single photo on Instagram

Here is a list of total 26 users in who haven't posted any single photo yet. They'll be receiving a promotion mail when they upload first post on their feed.

id	username
5	Aniya_Hackett
7	Kasandra_Homenick
14	Jaclyn81
21	Rocio33
24	Maxwell.Halvorson
25	Tierra.Trantow
34	Pearl7
36	Ollie_Ledner37
41	Mckenna17
45	David.Osinski47
49	Morgan.Kassulke
53	Linnea59
54	Duane60
57	Julien_Schmidt
66	Mike.Auer39
68	Franco_Keebler64
71	Nia_Haag
74	Hulda.Macejkovic
75	Leslie67
76	Janelle.Nikolaus81
80	Darby_Herzog
81	Esther.Zulauf61
83	Bartholome.Bernhard
89	Jessyca_West
90	Esmeralda.Mraz57
91	Bethany20

Declaring Contest Winner

USER ID	52
USERNAME	Zack_Kemmer93
LIKES	48

Hashtag researching

Top most commonly used hashtags as per count analysis

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

Launching ad campaign: to suggest the best day to launch

With highest count of 16, the best time to schedule an ad campaign are on Thursday and Sunday**.**

Investors metrics: On an average the frequency of an user posts on Instagram.

ACTIVE USERS	**74 users**
TOTAL USERS	**100 users**
TOTAL POSTS	**257 posts**

On an average no. of 3 to 4 times posts are made.
The total number of photos on instagram per total number of users are =257/100 = 2.57

Data on users (bots): fake and dummy accounts

Identifying accounts who has liked almost all the posts.

id	username	total_likes
5	Aniya_Hackett	257
14	Jadyn81	257
21	Rocio33	257
24	Maxwell.Halvorson	257
36	Ollie_Ledner37	257
41	Mckenna17	257
54	Duane60	257
57	Julien_Schmidt	257
66	Mike.Auer39	257
71	Nia_Haag	257
75	Leslie67	257
76	Janelle.Nikolaus81	257
91	Bethany20	257

Conclusion:

This project has allowed us to gain insights into social media users and their basic metrics, despite the small size of the dataset.

III. Operation and Metric Analytics

Project Description:

Operation Analytics analyzes a company's end-to-end operations to identify areas for improvement. By working closely with various teams, such as operations, support, and marketing, insights can be derived from data collection. This analysis is crucial for predicting a company's overall growth or decline, improving automation, cross-functional team understanding, and effective workflows. Investigating metric spikes, such as dips in daily engagement or sales, is also essential for daily decision-making.

Findings:

Number of jobs reviewed per hour per day for November 2020

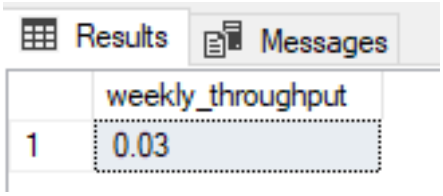
From our SQL query we get the following output:
With 218 being the highest and 35 being the lowest number of times the jobs were reviewed for Nov 2020.

Results | Messages

	dates	Jobs_reviewed
1	2020-11-25 00:00:00.000	80
2	2020-11-26 00:00:00.000	64
3	2020-11-27 00:00:00.000	35
4	2020-11-28 00:00:00.000	218
5	2020-11-29 00:00:00.000	180
6	2020-11-30 00:00:00.000	180

Throughput: *Finding the no. of events happening per second. 7 day rolling average of throughput.*

Dates	Daily Throughput
2020-11-25 00:00:00.000	0.0222222222222222
2020-11-26 00:00:00.000	0.0178571428571429
2020-11-27 00:00:00.000	0.00961538461538462
2020-11-28 00:00:00.000	0.0606060606060606
2020-11-29 00:00:00.000	0.05
2020-11-30 00:00:00.000	0.05



In results it can be seen that the highest Daily throughput is 0.06 on 2020-11-28 and on weekly, throughput is 0.03

Percentage *share of each language in the last 30 days*

Languages	Percentage
English	12.50
Arabic	12.50
Persian	37.50
Hindi	12.50
French	12.50
Italian	12.50

here, percentage share is shown for the nov month with Persian language having highest share of 37.50%

Duplicate rows *displaying duplicates from the table*

In this dataset, 2 duplicate rows are found in actor_id column

Investigating metric spike
User Engagement: *the weekly user engagement*

Week Numbers	Weekly Active Users
18	663
19	1068
20	1113
21	1154
22	1121
23	1186
24	1232
25	1275
26	1264
27	1302
28	1372
29	1365
30	1376
31	1467
32	1299
33	1225
34	1225
35	1204
36	104

User Growth: *Amount of users growing over time for a product*

Months	Users	Growth in %
1	712	NULL
2	685	-3.79
3	765	11.68
4	907	18.56
5	993	9.48
6	1086	9.37
7	1281	17.96
8	1347	5.15
9	330	-75.50
10	390	18.18
11	399	2.31
12	486	21.80

Email Engagement: *Users engaging with the email service metrics*

	user_id	occurred_at	action	user_type
1	0	2014-05-06 09:30:00.0000000	sent_weekly_digest	1

	Week	weekly_digest_rate	Email Open Rate	Email Clickthrough Rate	Reengagement Email Rate
1	18	62.32	21.28	11.39	5.01
2	19	63.45	22.24	10.49	3.83
3	20	62.16	22.67	11.13	4.04
4	21	61.62	22.64	11.43	4.31
5	22	63.52	22.82	9.97	3.69
6	23	63.59	21.56	10.66	4.19
7	24	62.39	22.34	11.18	4.09
8	25	61.61	22.92	10.99	4.48
9	26	63.77	21.79	10.54	3.9
10	27	62.99	22.22	10.61	4.18
11	28	62.24	22.49	11.37	3.9
12	29	62.92	22.48	10.77	3.83
13	30	63.98	21.71	10.51	3.79
14	31	62.29	23.24	10.59	3.88
15	32	65.27	23.25	7.66	3.82
16	33	66.59	22.85	7.14	3.42
17	34	64.73	23.1	7.91	4.26
18	35	64.33	23.91	7.67	4.08
19	36	0	32.28	29.92	37.8

IV. Hiring Process Analytics

Project Description:

Hiring Process Analytics is the collection and analysis of data related to recruitment and selection of employees. It uses a data-driven approach that leads to improved efficiency, better candidate quality, reduced time - to - fill, and reducing recruitment costs. In this project, as a former Lead Data Analyst at Google, solutions will be suggested in reference to previous hiring records.

Findings:

Hiring: *Process of intaking of people into an organization for different kinds of positions.*
Total count of males & females Hired:

Male	2563
Female	1856

Average Salary *offered by company across all departments.*

	Department	Average Offered Salary
1		
2	Finance Department	49628.00694
3	General Management	58722.09302
4	Human Resource Department	49002.27835
5	Marketing Department	48489.93538
6	Operations Department	49151.35438
7	Production Department	49448.48421
8	Purchase Department	52564.77477
9	Sales Department	49310.3807
10	Service Department	50629.88418

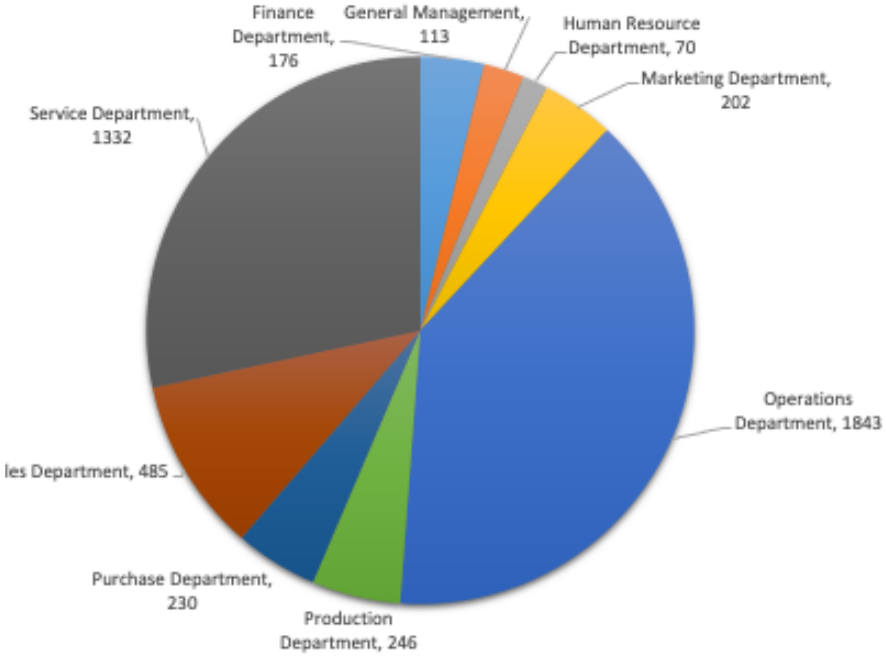
Class Intervals *for salary in the company:*

By using "MAX" & "MIN" function to calculate the intervals and finding difference per bins for classifying the intervals lastly by concatenation, forming sequential intervals

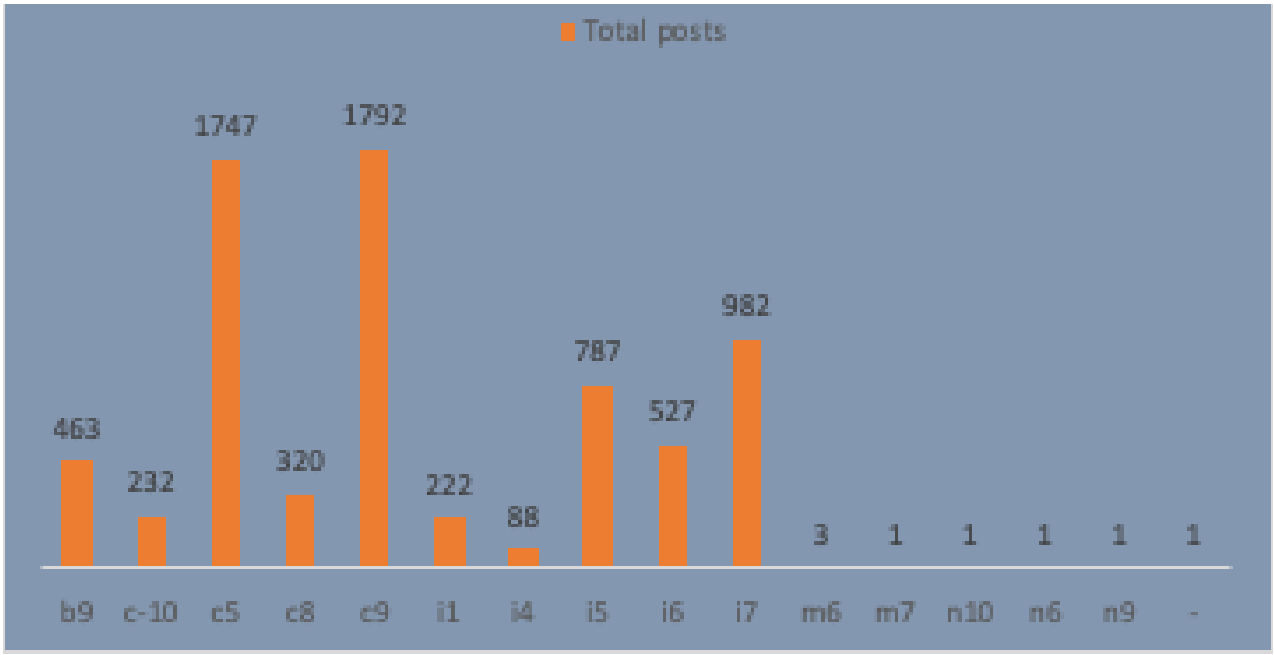
Max salary	400000
Min salary	100
Difference	399900
Bins	5
Range	79980
Class interval	
100-80080	
80081-160060	
160061-240040	
240041-320020	
320021-400000	

Department	Employees
Finance Department	176
General Management	113
Human Resource Departmer	70
Marketing Department	202
Operations Department	1843
Production Department	246
Purchase Department	230
Sales Department	485
Service Department	1332

Charts and Plots: *To visualize the data using Pie Chart / Bar Graph to show proportion of people working in different department.*



Charts: Representing different post tiers using chart/graph



V. IMDb Movie Analysis

Project Description:

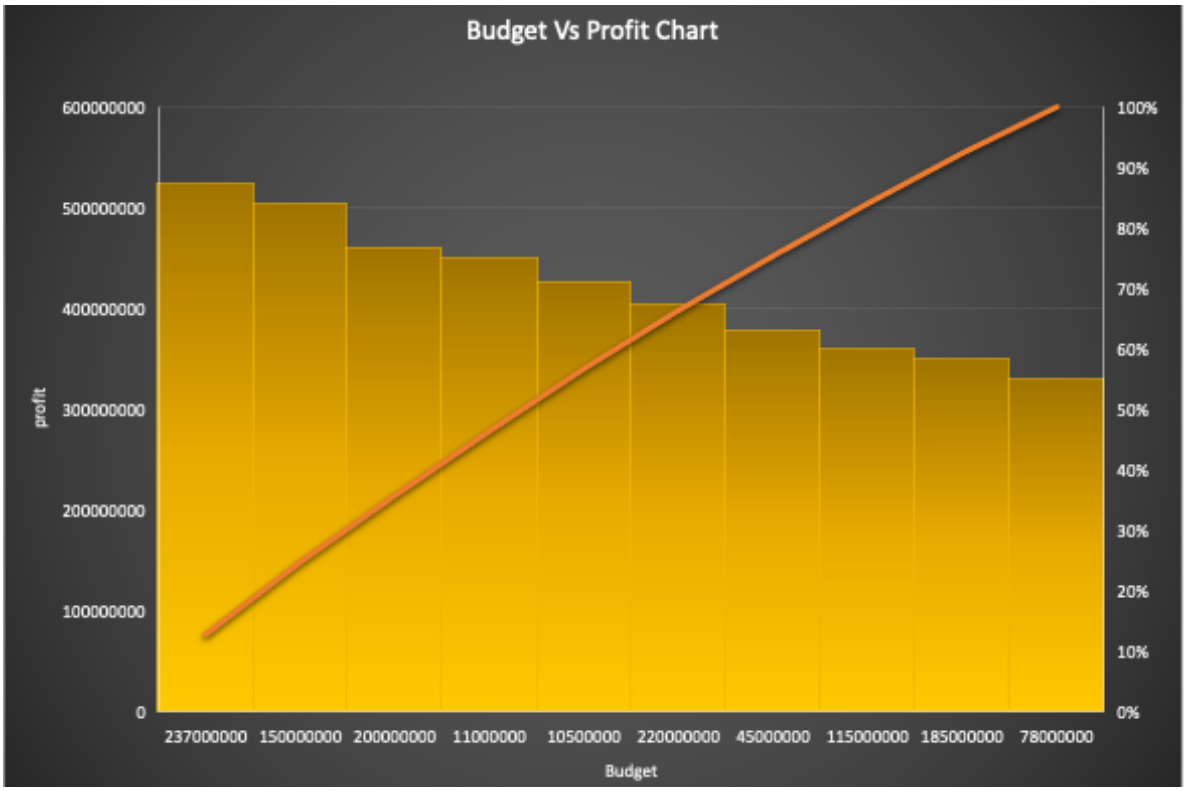
In this final project, we'll be performing Data Cleaning. Initially defining the problem for the given IMDB movie dataset, followed by asking the 5 Why's:
What do we see happening?
What is our hypothesis for the cause of the problem?
What is the impact of the problem on stakeholders?
What is the impact of the problem not being solved?
*Also known as the **Root Cause Analysis**, developed by Sakichi Toyoda, founder of Toyota Industries.*

Findings:

Movies with highest profit

TOP 10 Movies with high profit

movie_title higest profit	budget	Profit
Avatar	237000000	523505847
Jurassic World	150000000	502177271
Titanic	200000000	458672302
Star Wars: Episode IV - A New Hope	11000000	449935665
E.T. the Extra-Terrestrial	10500000	424449459
The Avengers	220000000	403279547
The Lion King	45000000	377783777
Star Wars: Episode I - The Phantom Menace	115000000	359544677
The Dark Knight	185000000	348316061
The Hunger Games	78000000	329999255



Top 250

- Creating a new column "IMDb_Top_250" and storing the top 250 movies with the highest IMDb Rating and **num_voted_users** > 25,000
- By sorting all the movies with the highest imdb scores and num_voted_users>25000 & ordering **imdb_score** in descending order.

Top - 250 movies with Voted_users > 25000 and imdb_score largest to smallest

Rank	movie_title	num_voted_u	imdb_sco	language
1	The Shawshank Reden	1689764	9.3	English
2	The Godfather	1155770	9.2	English
3	The Dark Knight	1676169	9	English
4	The Godfather: Part II	790926	9	English
5	The Lord of the Rings:	1215718	8.9	English
6	Pulp Fiction	1324680	8.9	English
7	Schindler's List	865020	8.9	English
8	The Good, the Bad an	503509	8.9	Italian
9	Forrest Gump	1251222	8.8	English
10	Star Wars: Episode V -	837759	8.8	English

Extracting all the non-English movies in the IMDb_Top_250 and storing them in a new column named Top_Foreign_Lang_Film.

Movies in Foreign languages

Top_Foreign_Lang_Film	num_voted_users	imdb_score	language
The Good, the Bad and the Ugly	503509	8.9	Italian
City of God	533200	8.7	Portuguese
Seven Samurai	229012	8.7	Japanese
Spirited Away	417971	8.6	Japanese
The Lives of Others	259379	8.5	German
Children of Heaven	27882	8.5	Persian
A Separation	151812	8.4	Persian
Oldboy	356181	8.4	Korean
Das Boot	168203	8.4	German
AmÃ©lie	534262	8.4	French
Princess Mononoke	221552	8.4	Japanese
The Hunt	170155	8.3	Danish
Metropolis	111841	8.3	German
Downfall	248354	8.3	German
Pan's Labyrinth	467234	8.2	Spanish
The Secret in Their Eyes	131831	8.2	Spanish
Incendies	80429	8.2	French
Howl's Moving Castle	214091	8.2	Japanese
Amores Perros	173551	8.1	Spanish
The Celebration	65951	8.1	Danish
Elite Squad	81644	8.1	Portuguese
The Sea Inside	64556	8.1	Spanish
Tae Guk Gi: The Brotherhood of War	31943	8.1	Korean

Best Directors

Using the pivot table to calculate mean (Average) by keeping Director_name in row and imdb_score in values. Finally sorting directors in descending order.

TOP 10 Directors	
Director Name	Average of imdb_score
Akira Kurosawa	8.7
Charles Chaplin	8.6
Tony Kaye	8.6
Alfred Hitchcock	8.5
Damien Chazelle	8.5
Majid Majidi	8.5
Ron Fricke	8.5
Sergio Leone	8.433333333
Christopher Nolan	8.425
Asghar Farhadi	8.4
Richard Marquand	8.4

Popular Genres

Using COUNTIF for counting the frequency of watchers and sorting them in descending order.

Most Watched Genre	
Genre	watchers
Drama	1874
Comedy	1451
Thriller	1105
Action	951
Romance	849
Adventure	773
Crime	702
Fantasy	504
Sci-Fi	491
Family	439
Horror	386
Mystery	378
Biography	238
Animation	196
Musical	95
Western	57
Documentary	44

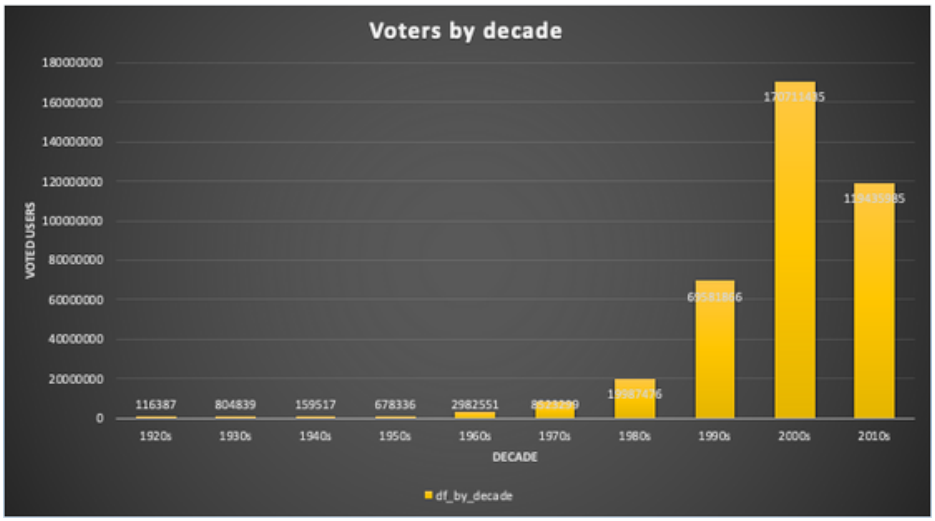
Charts

critic-favorite and audience-favorite actors

Actors with highest mean		
Row Labels	Average of num_user_for_reviews	Average of num_critic_for_reviews
Leonardo DiCaprio	914.4761905	330.1904762
Brad Pitt	742.3529412	245
Meryl Streep	297.1818182	181.4545455
Grand Total	**716.1836735**	**267.244898**

Bar chart

Observe change in number of voted users over decades.



Decade	df_by_decade
1920s	116387
1930s	804839
1940s	159517
1950s	678336
1960s	2982551
1970s	8523299
1980s	19987476
1990s	69581866
2000s	170711435
2010s	119435985

VI. Bank Loan Case Study

Project Description:

This Project aims to identify patterns indicating whether a client will have difficulty paying their installments and determine the driving factors behind loan default. The analysis will inform actions such as denying loans, reducing loan amounts, or lending at higher interest rates.

Findings:

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

In [2]: df_application_data = pd.read_csv('application_data.csv')
        df_previous_application = pd.read_csv('previous_application.csv')

In [3]: #Reading the first 5 Rows of the data
        df_application_data.head()

Out[3]:
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREI
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

5 rows × 122 columns

```
In [4]: df_previous_application = pd.read_csv('previous_application.csv')
        df_previous_application.head()
```

```
In [5]: #Dataset Dimensions
        print("The dimension of Application_data:",df_application_data.shape)
        print("The dimension of Previous_Application:",df_previous_application.shape)

        The dimension of Application_data: (307511, 122)
        The dimension of Previous_Application: (1670214, 37)

In [6]: #Dataset size
        print("The size of Application_data:",df_application_data.size)
        print("The size of Previous_Application:",df_previous_application.size)

        The size of Application_data: 37516342
        The size of Previous_Application: 61797918
```

Cleaning data by firstly counting the total number of null values present in each column of the dataset.

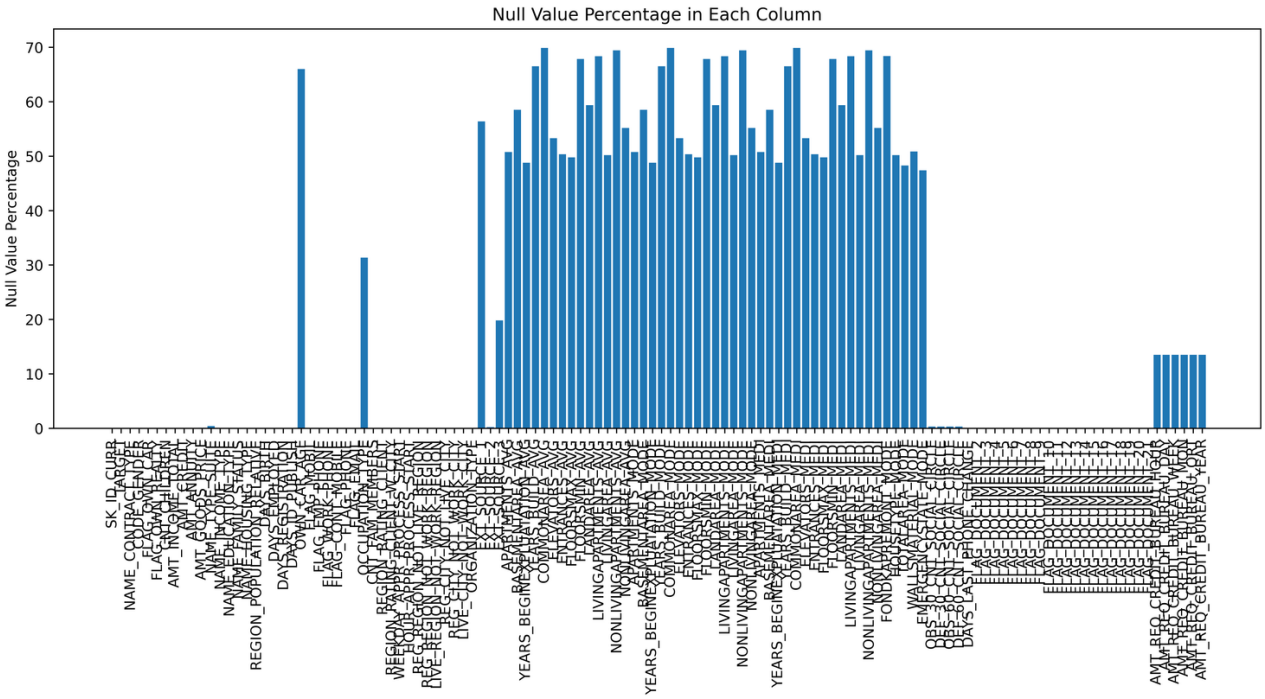
2. Data Cleaning and Manipulation

A. Checking for Missing Values & Dropping them

```
In [14]: #funcion to get null value percentage
         def column_wise_null_percentage(df):
             output = round(df.isnull().sum()/len(df.index)*100,2)
             return output

In [15]: #get missing values of all columns
         null_col = column_wise_null_percentage(df_application_data)
         null_col

Out[15]: SK_ID_CURR                    0.00
         TARGET                       0.00
         NAME_CONTRACT_TYPE           0.00
         CODE_GENDER                  0.00
         FLAG_OWN_CAR                 0.00
         FLAG_OWN_REALTY              0.00
         CNT_CHILDREN                 0.00
         AMT_INCOME_TOTAL             0.00
```

Visualizing the graph between null value percentage in each column Vs Null value percentage

```
In [20]: # Calculate the percentage of null values in each column
         null_percentage = df_application_data.isnull().sum() * 100 / df_application_data.shape[0]
         null_df = pd.DataFrame({'Column Name': null_percentage.index, 'Null Value Percentage': null_percentage.values})

         # Plot the percentage of null values for each column
         plt.figure(figsize=(15, 5), dpi=600)
         plt.bar(x=null_df['Column Name'], height=null_df['Null Value Percentage'])
         plt.xticks(rotation=90)
         plt.title('Null Value Percentage in Each Column')
         plt.ylabel('Null Value Percentage')
         plt.show()
```



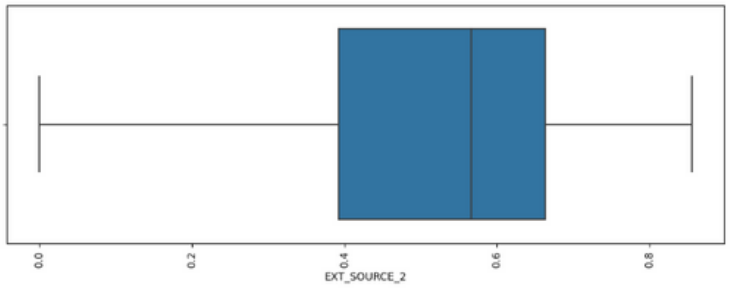
Checking correlation between External source columns and target columns using heatmap for better overall view.

Insights: Most of the columns with high missing values are related to different area sizes on apartment owned/rented by the loan applicant.

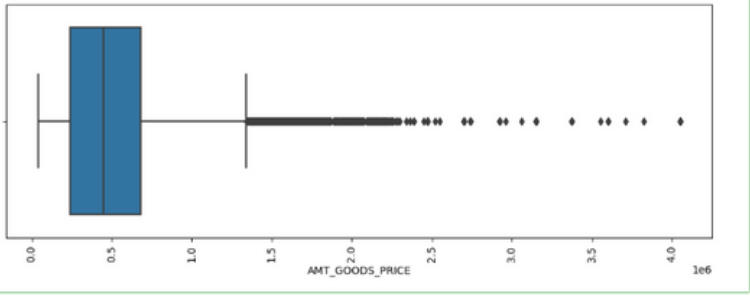
```
In [25]: # Checking correlation of EXT_SOURCE_X columns vs TARGET column
         Source = df_application_data[["EXT_SOURCE_1","EXT_SOURCE_2","EXT_SOURCE_3","TARGET"]]
         source_corr = Source.corr()
         ax = sns.heatmap(source_corr,
                          xticklabels=source_corr.columns,
                          yticklabels=source_corr.columns,
                          annot = True,
                          cmap ="RdYlGn_r")
```

Continuous variables and categorical variables using box plot to visualize the outliers identify potential relationships between variables.

```
For analysis of imputation selected 7 variables.
continuous variables:

> 'EXT_SOURCE_2','AMT_GOODS_PRICE'

Categorical variables:

> 'OBS_30_CNT_SOCIAL_CIRCLE','OBS_60_CNT_SOCIAL_CIRCLE','DEF_60_CNT_SOCIAL_CIRCLE','DEF_30_CNT_SOCIAL_CIRCLE','NAME_TYPE_SUITE'

Continous variables:

In [34]: # Box plot for continuous variable
         plt.figure(figsize=(12,4))
         sns.boxplot(x=df_application_data['EXT_SOURCE_2'], orient='h')
         plt.xticks(rotation=90)
         plt.show()
```




```
In [35]: plt.figure(figsize=(12,4))
         sns.boxplot(x=df_application_data['AMT_GOODS_PRICE'])
         plt.xticks(rotation=90)
         plt.show()
```



Categorical value:

In [29]:
```
# identify maximum frequency values
print('Maximum Frequncy categorical values are:')
print('NAME_TYPE_SUITE: ',df_application_data['NAME_TYPE_SUITE'].mode()[0])
print('OBS_30_CNT_SOCIAL_CIRCLE:', df_application_data['OBS_30_CNT_SOCIAL_CIRCLE'].mode()[0])
print('DEF_30_CNT_SOCIAL_CIRCLE:', df_application_data['DEF_30_CNT_SOCIAL_CIRCLE'].mode()[0])
print('OBS_60_CNT_SOCIAL_CIRCLE:', df_application_data['OBS_60_CNT_SOCIAL_CIRCLE'].mode()[0])
print('DEF_60_CNT_SOCIAL_CIRCLE:', df_application_data['DEF_60_CNT_SOCIAL_CIRCLE'].mode()[0])
```

```
Maximum Frequncy categorical values are:
NAME_TYPE_SUITE:  Unaccompanied
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
```

For categorical variable the value which should be imputed with maximum in frequency. So the value to be imputed are: NAME_TYPE_SUITE: Unaccompanied 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

In [30]:
```
# Remove unwanted columns from application dataset for better analysis.

unwanted=['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE','FLAG_PHONE', 'FLAG_EMAIL',
          'REGION_RATING_CLIENT','REGION_RATING_CLIENT_W_CITY','FLAG_EMAIL','CNT_FAM_MEMBERS', 'REGION_RATING_CLIENT',
          'REGION_RATING_CLIENT_W_CITY','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','EXT_SOURCE_2','EXT_SOURCE_3','YEARS_BEGINEXPLUATATION_AVG','FLOORSMAX_AVG','YEARS_BEGINEXPLUATATION
          'FLOORSMAX_MODE','YEARS_BEGINEXPLUATATION_MEDI','FLOORSMAX_MEDI','TOTALAREA_MODE','EMERGENCYSTATE_MODE']

df_application_data.drop(labels=unwanted,axis=1,inplace=True)
```

Displaying dataset shape after removing unwanted columns

In [31]:
```
df_application_data.shape
```

Out[31]: (307511, 42)

In [32]:
```
df_application_data.head()
```

Out[32]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CRE
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

There are some columns where the value is mentioned as 'XNA' which means 'Not Available'. So we have to find the number of rows and columns.

There are some columns where the value is mentioned as 'XNA' which means 'Not Available'. So we have to find the number of rows and columns.

In [33]:
```
# For Code Gender column

print('CODE_GENDER: ',df_application_data['CODE_GENDER'].unique())
print('No of values: ',df_application_data[df_application_data['CODE_GENDER']=='XNA'].shape[0])

XNA_count = df_application_data[df_application_data['CODE_GENDER']=='XNA'].shape[0]
per_XNA = round(XNA_count/len(df_application_data.index)*100,3)

print('% of XNA Values:',  per_XNA)

print('maximum frequency data :', df_application_data['CODE_GENDER'].describe().top)
```

```
CODE_GENDER:  ['M' 'F' 'XNA']
No of values:  4
% of XNA Values: 0.001
maximum frequency data : F
```

Since, Female is having the majority and only 2 rows are having XNA values, we can impute those with Gender 'F' as there will be no impact on the dataset. Also there will no impact if we drop those rows.

In [34]:
```
# Dropping the XNA value in column 'CODE_GENDER' with "F" for the dataset

df_application_data = df_application_data.drop(df_application_data.loc[df_application_data['CODE_GENDER']=='XNA'].index)
df_application_data[df_application_data['CODE_GENDER']=='XNA'].shape
```

Out[34]: (0, 42)

Checking for the percentage of XNA values and dropping rows having "XNA" as value in the organization type column

In [35]:
```
# For Organization column
print('No of XNA values: ', df_application_data[df_application_data['ORGANIZATION_TYPE']=='XNA'].shape[0])

XNA_count = df_application_data[df_application_data['ORGANIZATION_TYPE']=='XNA'].shape[0]
per_XNA = round(XNA_count/len(df_application_data.index)*100,3)

print('% of XNA Values:',  per_XNA)

df_application_data['ORGANIZATION_TYPE'].describe()
```

```
No of XNA values:  55374
% of XNA Values: 18.007
```

Out[35]:
```
count                   307507
unique                      58
top       Business Entity Type 3
freq                     67992
Name: ORGANIZATION_TYPE, dtype: object
```

In [36]:
```
# # Dropping the rows have 'XNA' values in the organization type column

# df_application_data = df_application_data.drop(df_application_data.loc[df_application_data['ORGANIZATION_TYPE']=='XNA'].index)
# df_application_data[df_application_data['ORGANIZATION_TYPE']=='XNA'].shape
```

2.d. Check the data type of all the columns and changed the data type.¶

In [37]:
```
df_application_data.head()
```

Out[37]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CRE
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

In [38]:
```
# Casting variable into numeric in the dataset

numeric_columns=['TARGET','CNT_CHILDREN','AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY','REGION_POPULATION_RELATIVE',
                 'DAYS_BIRTH','DAYS_EMPLOYED','DAYS_REGISTRATION','DAYS_ID_PUBLISH','HOUR_APPR_PROCESS_START',
                 'LIVE_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY','REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY',
                'DAYS_LAST_PHONE_CHANGE']

df_application_data[numeric_columns]=df_application_data[numeric_columns].apply(pd.to_numeric)
df_application_data.head(5)
```

Out[38]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CRE
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

converting the following age/days columns having -ve value to +ve value.

2.d. Check the data type of all the columns and changed the data type.¶

In [37]:
```
df_application_data.head()
```

Out[37]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CRE
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

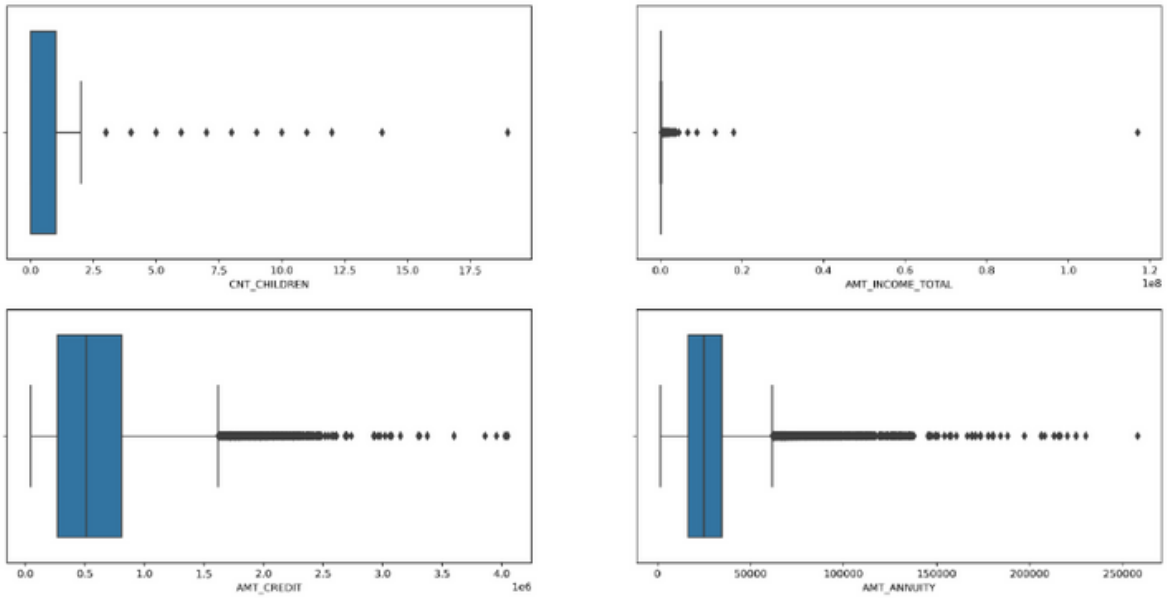
In [38]:
```
# Casting variable into numeric in the dataset

numeric_columns=['TARGET','CNT_CHILDREN','AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY','REGION_POPULATION_RELATIVE',
                 'DAYS_BIRTH','DAYS_EMPLOYED','DAYS_REGISTRATION','DAYS_ID_PUBLISH','HOUR_APPR_PROCESS_START',
                 'LIVE_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY','REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY',
                'DAYS_LAST_PHONE_CHANGE']

df_application_data[numeric_columns]=df_application_data[numeric_columns].apply(pd.to_numeric)
df_application_data.head(5)
```

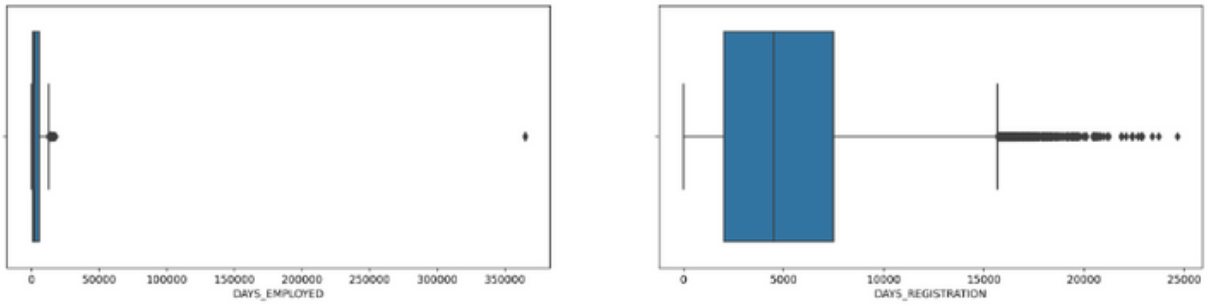
Out[38]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CRE
0	100002	1	Cash loans	M	N	Y	0	202500.0	40659
1	100003	0	Cash loans	F	N	N	0	270000.0	129350
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	13500
3	100006	0	Cash loans	F	N	Y	0	135000.0	31268
4	100007	0	Cash loans	M	N	Y	0	121500.0	51300

Detecting outliers using Box plot for the selected columns.

In [41]:
```
# Box plot for selected columns
features = ['CNT_CHILDREN', 'AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY','DAYS_EMPLOYED', 'DAYS_REGISTRATION']

plt.figure(figsize = (20, 15), dpi=300)
for i in enumerate(features):
    plt.subplot(3, 2, i[0]+1)
    sns.boxplot(x = i[1], data = df_application_data)
plt.show()
```





From the above box plot and descibe analysis we found that following are the numeric columns are having outliers:

~ CNT_CHILDREN, AMT_INCOME_TOTAL,AMT_CREDIT,AMT_ANNUITY,DAYS_EMPLOYED, DAYS_REGISTRATION

The first quartile almost missing for CNT_CHILDREN that means most of the data are present in the first quartile.

There is single high value data point as outlier present in AMT_INCOME_TOTAL and Removal this point will dtrasticaly impact the box plot for further analysis.

The first quartiles is slim compare to third quartile for AMT_CREDIT,AMT_ANNUITY, DAYS_EMPLOYED, DAYS_REGISTRATION. This mean data are skewed towards first quartile.

- *Creating bins for continuous categorical variables so as to reduce the number of unique values in a variables.*

2.f. Bin Creation

Creating bins for continous variable categories column 'AMT_INCOME_TOTAL' and 'AMT_CREDIT'

In [42]:
```
bins = [0,100000,200000,300000,400000,500000,10000000000]
slot = ['<100000', '100000-200000','200000-300000','300000-400000','400000-500000', '500000 and above']

df_application_data['AMT_INCOME_RANGE']=pd.cut(df_application_data['AMT_INCOME_TOTAL'],bins,labels=slot)
```

In [43]:
```
bins = [0,100000,200000,300000,400000,500000,600000,700000,800000,900000,10000000000]
slot = ['<100000', '100000-200000','200000-300000','300000-400000','400000-500000', '500000-600000',
        '600000-700000','700000-800000','850000-900000','900000 and above']

df_application_data['AMT_CREDIT_RANGE']=pd.cut(df_application_data['AMT_CREDIT'],bins,labels=slot)
```

3. TARGET Analysis:

In [44]:
```
# Dividing the dataset into two dataset of  target=1(client with payment difficulties) and target=0(all other)

target0_df=df_application_data.loc[df_application_data["TARGET"]==0]
target1_df=df_application_data.loc[df_application_data["TARGET"]==1]
```

Dividing the targets for analysis into 2 for percentage defaulters of people who did pay and did not pay their loan

In [45]:
```
# insights from number of target values

percentage_defaulters= round(100*len(target1_df)/(len(target0_df)+len(target1_df)),2)

percentage_nondefaulters=round(100*len(target0_df)/(len(target0_df)+len(target1_df)),2)

print('Count of target0_df:', len(target0_df))
print('Count of target1_df:', len(target1_df))


print('Percentage of people who paid their loan are: ', percentage_nondefaulters, '%' )
print('Percentage of people who did not paid their loan are: ', percentage_defaulters, '%' )
```

```
Count of target0_df: 282682
Count of target1_df: 24825
Percentage of people who paid their loan are:  91.93 %
Percentage of people who did not paid their loan are:  8.07 %
```

In [46]:
```
# Calculating Imbalance percentage

# Since the majority is target0 and minority is target1

imb_ratio = round(len(target0_df)/len(target1_df),2)

print('Imbalance Ratio:', imb_ratio)
```

```
Imbalance Ratio: 11.39
```

Univariate Analysis

Categorical Univariate Analysis in logarithmic scale for target=0 (client with no payment difficulties)

In [47]:
```
# Count plotting in Logarithmic scale

def uniplot(df,col,title,hue =None):

    sns.set_style('whitegrid')
    sns.set_context('talk')
    plt.rcParams["axes.labelsize"] = 14
    plt.rcParams['axes.titlesize'] = 16
    plt.rcParams['axes.titlepad'] = 14


    temp = pd.Series(data = hue)
    fig, ax = plt.subplots()
    width = len(df[col].unique()) + 7 + 4*len(temp.unique())
    fig.set_size_inches(width , 8)
    plt.xticks(rotation=45)
    plt.yscale('log')
    plt.title(title)
    ax = sns.countplot(data = df, x= col, order=df[col].value_counts().index,hue = hue)

    plt.show()
```

In [48]:
```
# Categoroical Univariate Analysis in Logarithmic scale

features = ['AMT_INCOME_RANGE', 'AMT_CREDIT_RANGE','NAME_INCOME_TYPE','NAME_CONTRACT_TYPE']
plt.figure(figsize = (20, 15))

for i in enumerate(features):
    plt.subplot(2, 2, i[0]+1)
    plt.subplots_adjust(hspace=0.5)
    sns.countplot(x = i[1], hue = 'TARGET', data = df_application_data)

    plt.rcParams['axes.titlesize'] = 16

    plt.xticks(rotation = 45)
    plt.yscale('log')
```

Insights:

AMT_INCOME_RANGE :

- *The people having 100000-200000 are having higher number of loan and also having higher value in defaulter*
- *The income segment having >500000 are having less defaulter.*

AMT_CREDIT_RANGE:

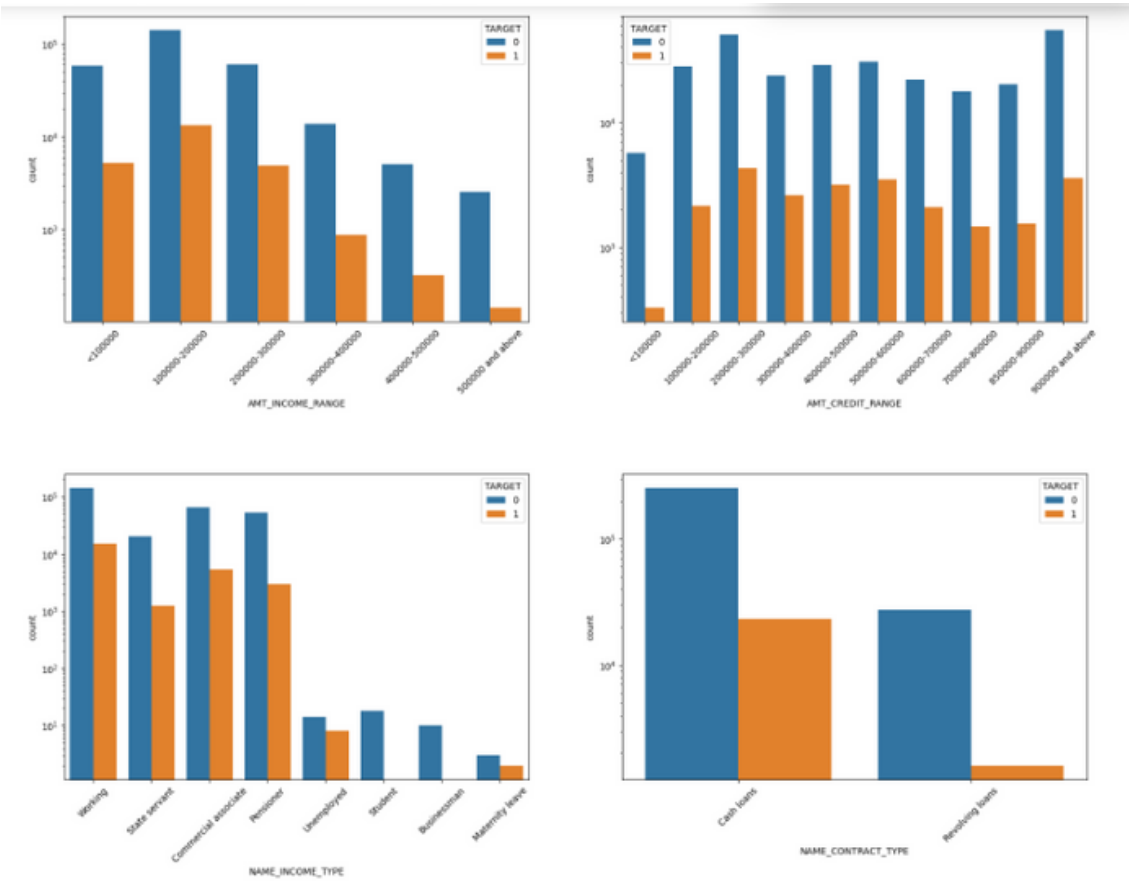
- *The people having <100000 loan are less defaulter.*
- *Income having more than >100000 are almost equal % to loan defaulter*

NAME_INCOME_TYPE:

- *Student pensioner and business have higher percentage of loan repayment.*
- *Working, State servant and Commercial associates have higher default percentage.*
- *Maternity category is significantly higher problem in replacement.*

NAME_CONTRACT_TYPE

- *For contract type 'cash loans' is having higher number of credits than 'Revolving loans' contract type.*
- *From the above graphs we can see that the Revolving loans are small amount compared to Cash loans but the % of non payment for the revolving loans are comapritvely high.*

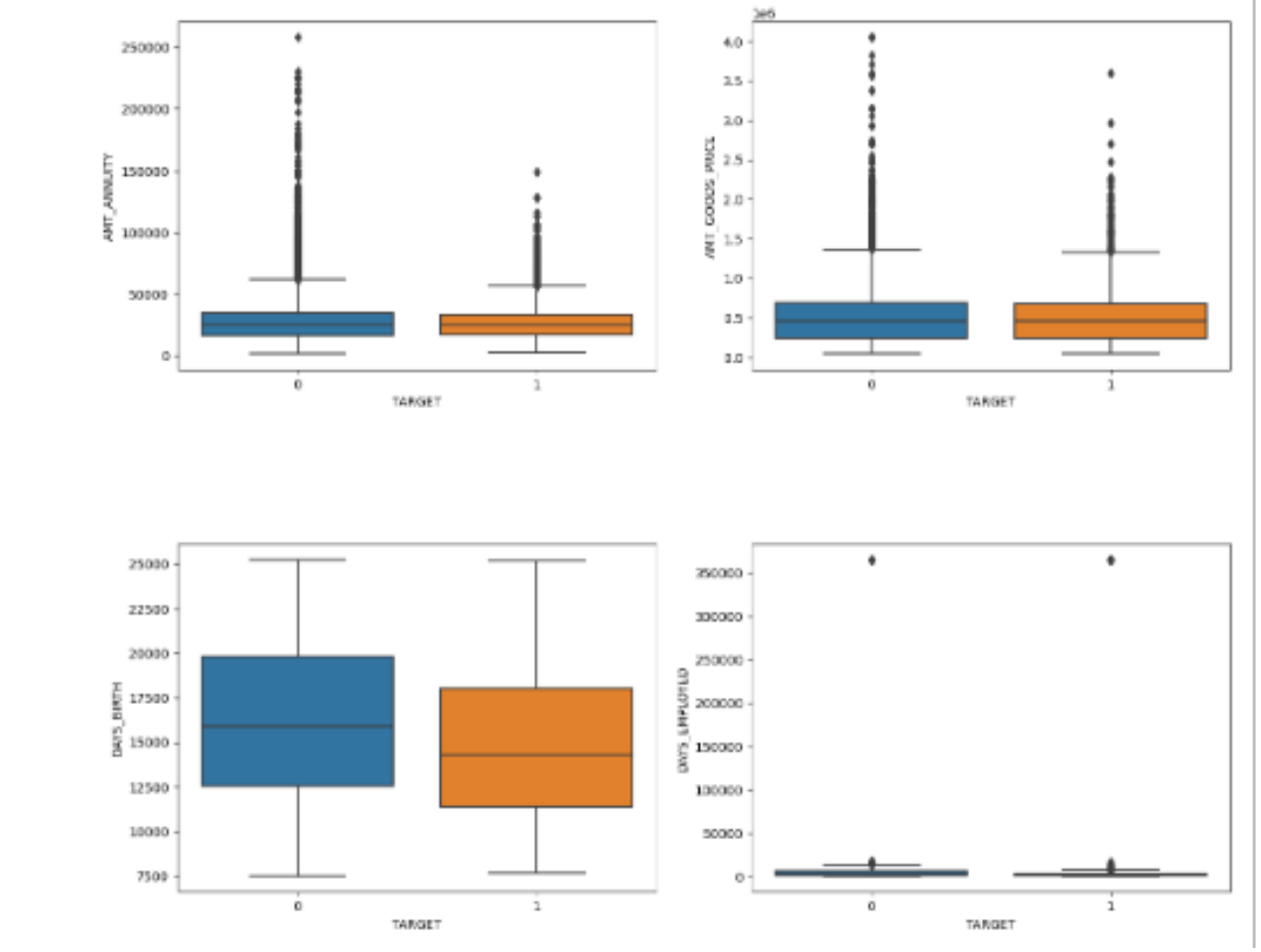


Univariate Analysis for continuous variables

- **Days_Birth**: People with more age has higher probability of repayment. Some outliers are observed in 'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'DAYS_EMPLOYED', DAYS_LAST_PHONE_CHANGE is less than Days_Birth and DAYS_ID_PUBLISH, 1st quartile is smaller than third quartile in In 'AMT_ANNUITY','AMT_GOODS_PRICE', DAYS_LAST_PHONE_CHANGE.
- **DAYS_ID_PUBLISH:** People changing ID in recent days are relatively prone to be default. There is a single high value data point as outlier present in DAYS_EMPLOYED. Removal of this point will drastically impact the box plot in analysis.

Univariate analysis Continuous variables:

```
In [49]: # Univariate Analysis for continous variable

features = ['AMT_ANNUITY','AMT_GOODS_PRICE','DAYS_BIRTH','DAYS_EMPLOYED','DAYS_LAST_PHONE_CHANGE','DAYS_ID_PUBLISH']
plt.figure(figsize = (15, 20))

for i in enumerate(features):
    plt.subplot(3, 2, i[0]+1)
    plt.subplots_adjust(hspace=0.5)
    sns.boxplot(x = 'TARGET', y = i[1], data = df_application_data)
```

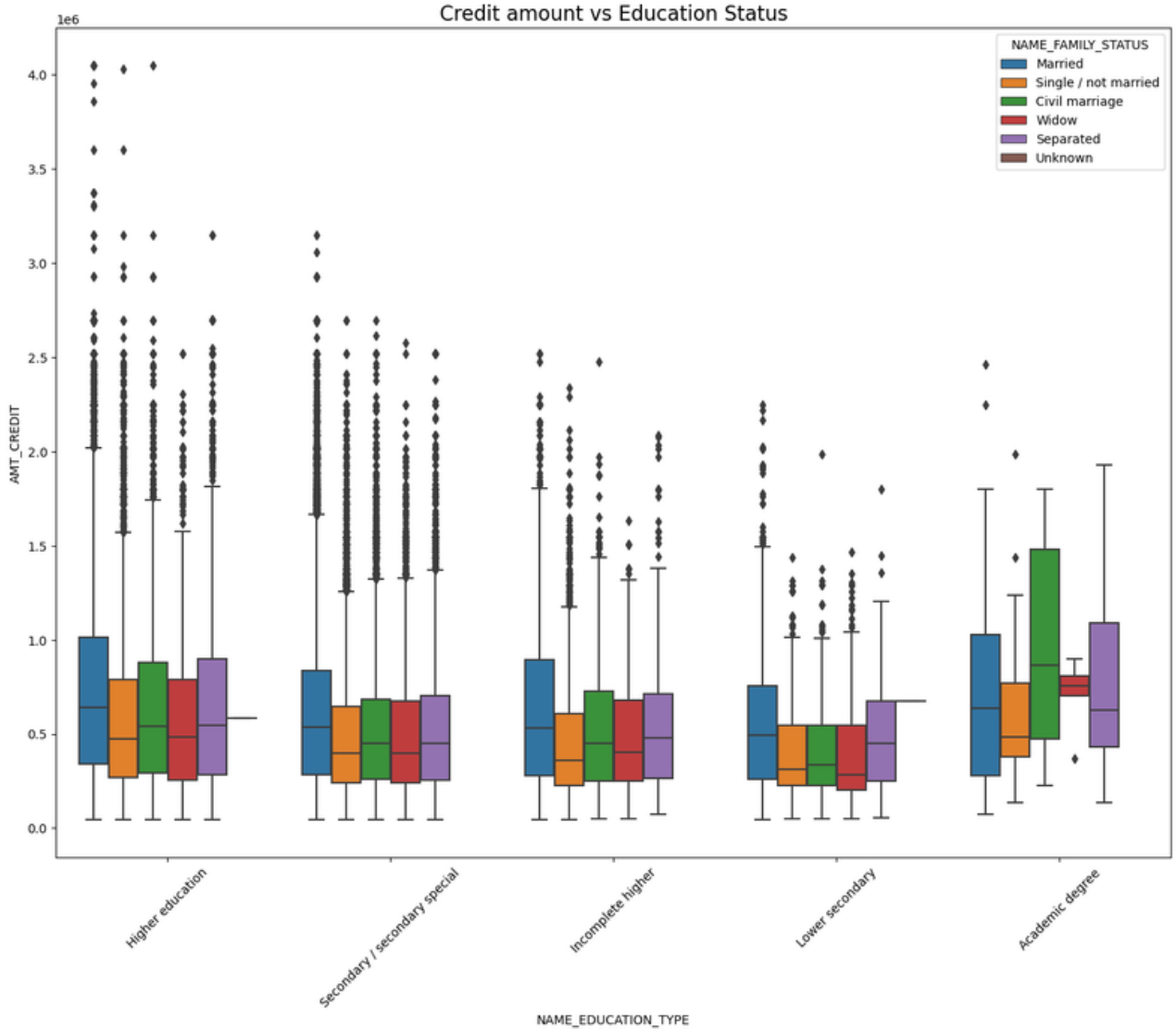


Bivariate Analysis for continuous variables

Family status of 'civil marriage', 'marriage' and 'separated' of Academic degree education are having higher number of credits than others. Also, higher education of family status of 'marriage', 'single' and 'civil marriage' has more outliers. Civil marriage for Academic degree has most of the credits in the third quartile.

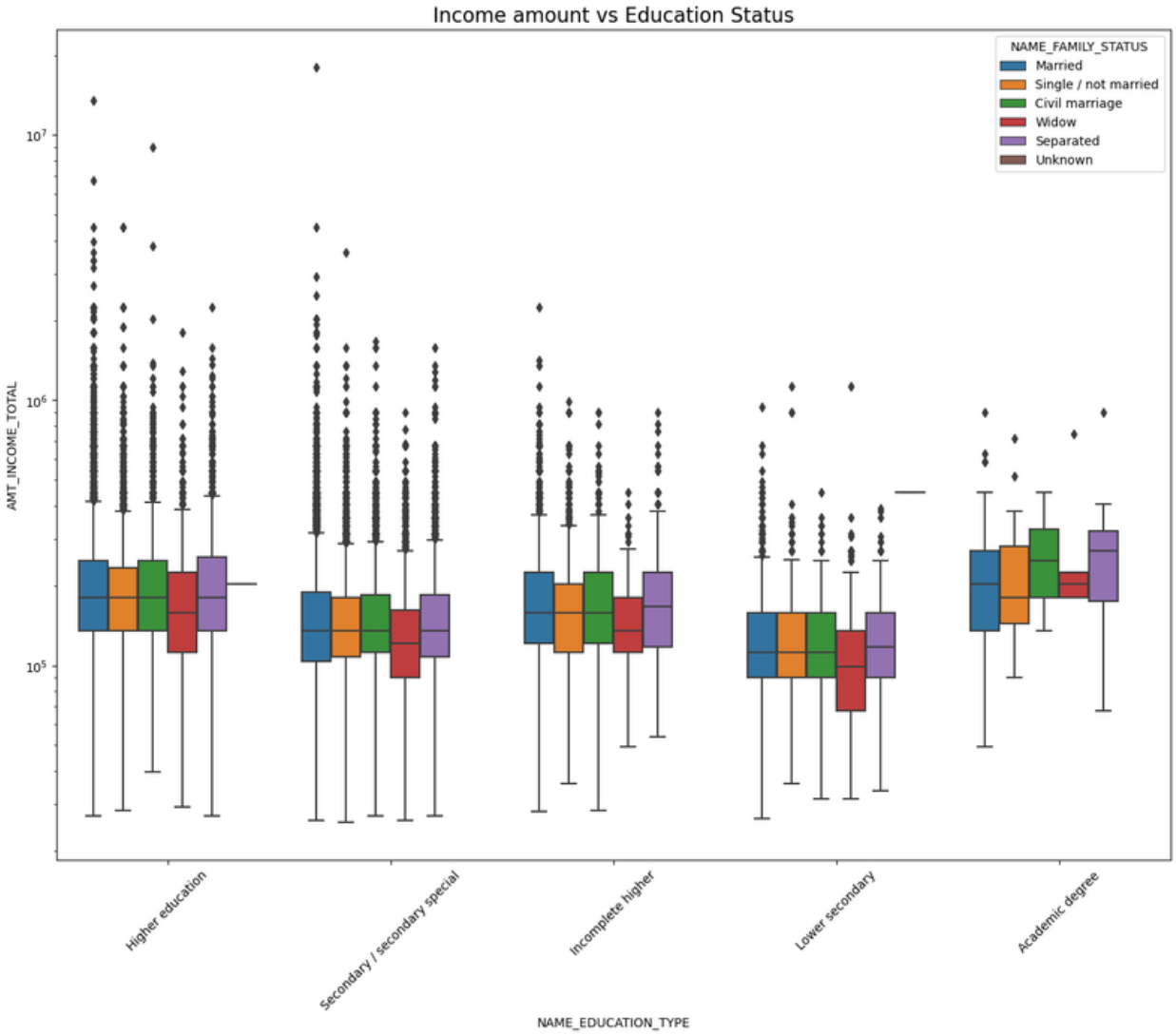
For Target 0

```
In [50]: # Box plotting for Credit amount

         plt.figure(figsize=(16,12))
         plt.xticks(rotation=45)
         sns.boxplot(data =target0_df, x='NAME_EDUCATION_TYPE',
                     y='AMT_CREDIT', hue ='NAME_FAMILY_STATUS',orient='v')
         plt.title('Credit amount vs Education Status')
         plt.show()
```



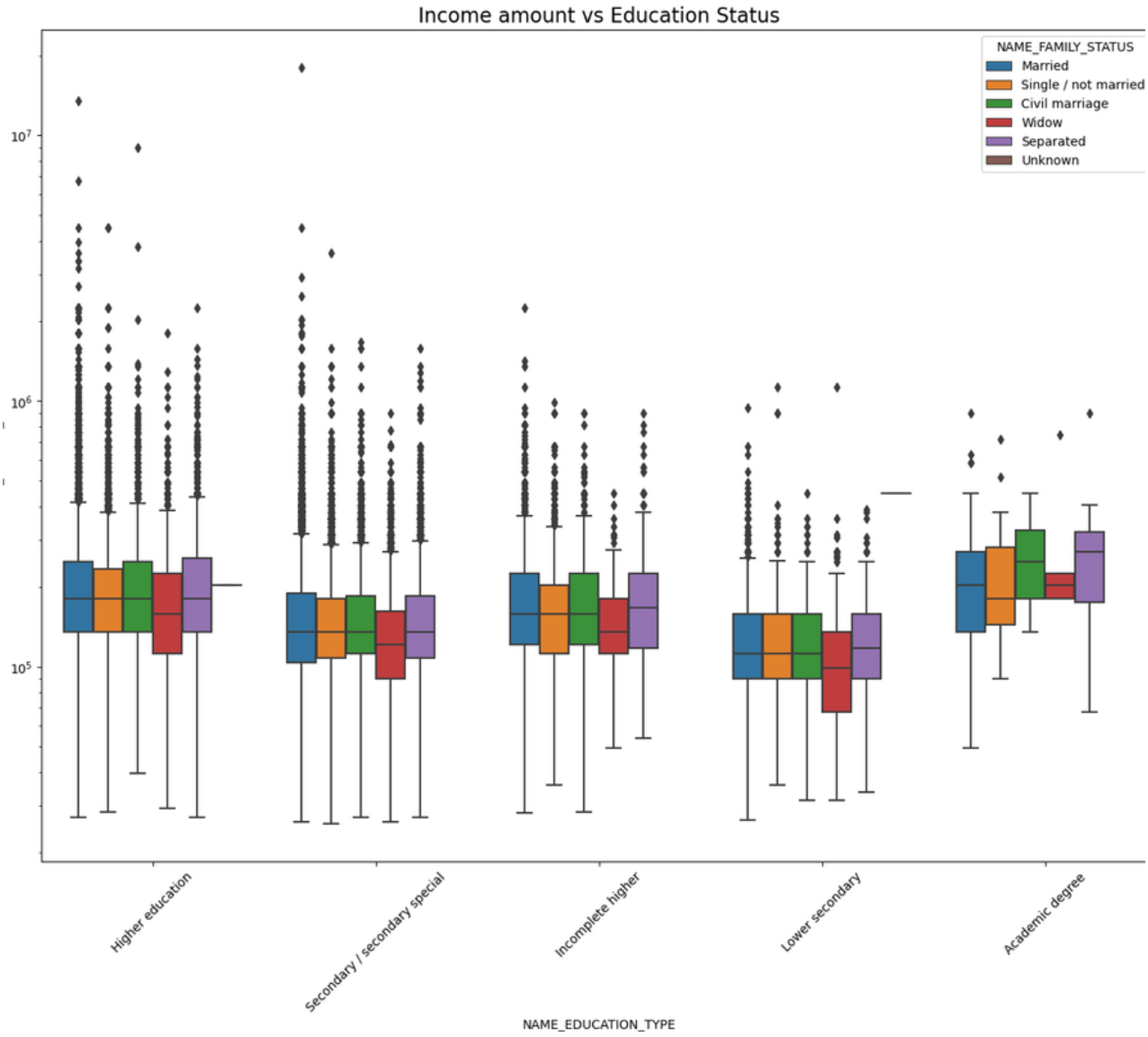
In Education type 'Higher education' the income amount is mostly equal to family status. And contain many outliers. Less outlier are present for Academic degree although their income amount is bit higher than Higher education. Lower secondary of civil marriage family status are having less income amount than others.

In [51]:
```
# Box plotting for Income amount in logarithmic scale

plt.figure(figsize=(16,12))
plt.xticks(rotation=45)
plt.yscale('log')
sns.boxplot(data =target0_df, x='NAME_EDUCATION_TYPE',
            y='AMT_INCOME_TOTAL', hue ='NAME_FAMILY_STATUS',orient='v')
plt.title('Income amount vs Education Status')
plt.show()
```



It can be inferred that they are very similar to Target 0 Family status of 'civil marriage', 'marriage' and 'separated' of Academic degree education as they have higher number of credits than others. Most of the outliers are from Education type 'Higher education' and 'Secondary'. Civil marriage for Academic degree has most of the credits in the third quartile.

```
In [53]: # Box plotting for Income amount in logarithmic scale

         plt.figure(figsize=(16,12))
         plt.xticks(rotation=45)
         plt.yscale('log')
         sns.boxplot(data =target0_df, x='NAME_EDUCATION_TYPE',
                     y='AMT_INCOME_TOTAL', hue ='NAME_FAMILY_STATUS',orient='v')
         plt.title('Income amount vs Education Status')
         plt.show()
```



Correlation

Getting top 10 correlation between variables

In [54]:
```
# Top 10 correlated variables: target 0 dataframe

corr = target0_df.corr(numeric_only=True)
corrdf = corr.where(np.triu(np.ones(corr.shape), k=1).astype(bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

Out[54]:

	Var1	Var2	Correlation
649	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
184	AMT_GOODS_PRICE	AMT_CREDIT	0.99
680	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.86
464	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.86
557	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.83
185	AMT_GOODS_PRICE	AMT_ANNUITY	0.78
154	AMT_ANNUITY	AMT_CREDIT	0.77
278	DAYS_EMPLOYED	DAYS_BIRTH	0.63
433	REG_REGION_NOT_WORK_REGION	REG_REGION_NOT_LIVE_REGION	0.45
526	REG_CITY_NOT_WORK_CITY	REG_CITY_NOT_LIVE_CITY	0.44

In [55]:
```
# Top 10 correlated variables: target 1 dataframe

corr = target1_df.corr(numeric_only=True)
corrdf = corr.where(np.triu(np.ones(corr.shape), k=1).astype(bool))
corrdf = corrdf.unstack().reset_index()
corrdf.columns = ['Var1', 'Var2', 'Correlation']
corrdf.dropna(subset = ['Correlation'], inplace = True)
corrdf['Correlation'] = round(corrdf['Correlation'], 2)
corrdf['Correlation'] = abs(corrdf['Correlation'])
corrdf.sort_values(by = 'Correlation', ascending = False).head(10)
```

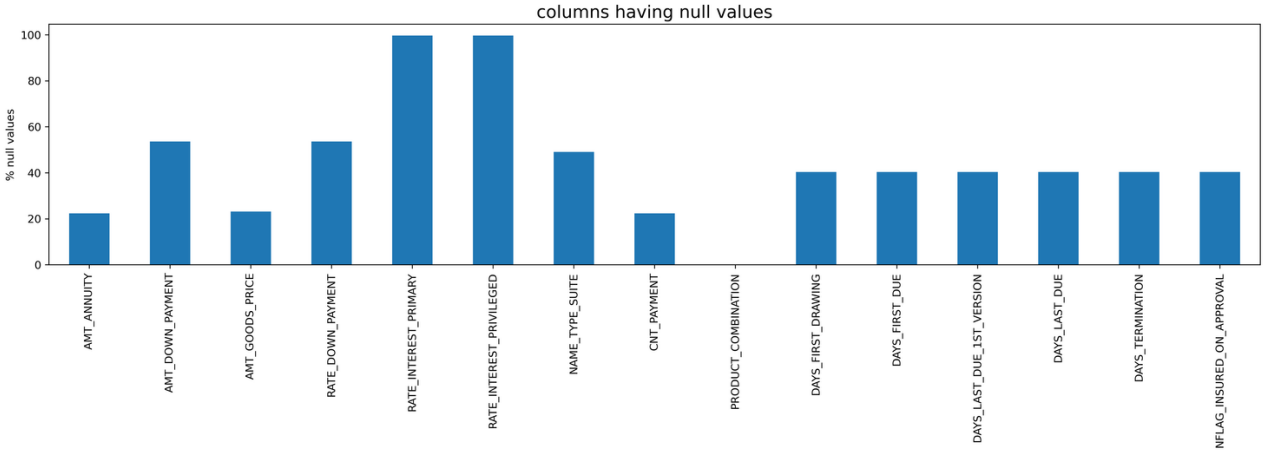
Out[55]:

	Var1	Var2	Correlation
649	OBS_60_CNT_SOCIAL_CIRCLE	OBS_30_CNT_SOCIAL_CIRCLE	1.00
184	AMT_GOODS_PRICE	AMT_CREDIT	0.98
680	DEF_60_CNT_SOCIAL_CIRCLE	DEF_30_CNT_SOCIAL_CIRCLE	0.87
464	LIVE_REGION_NOT_WORK_REGION	REG_REGION_NOT_WORK_REGION	0.85
557	LIVE_CITY_NOT_WORK_CITY	REG_CITY_NOT_WORK_CITY	0.78
185	AMT_GOODS_PRICE	AMT_ANNUITY	0.75
154	AMT_ANNUITY	AMT_CREDIT	0.75
278	DAYS_EMPLOYED	DAYS_BIRTH	0.58
433	REG_REGION_NOT_WORK_REGION	REG_REGION_NOT_LIVE_REGION	0.50
526	REG_CITY_NOT_WORK_CITY	REG_CITY_NOT_LIVE_CITY	0.47

From the above correlation analysis it is infered that the highest corelation (1.0) is between (OBS_60_CNT_SOCIAL_CIRCLE with OBS_30_CNT_SOCIAL_CIRCLE) and (FLOORSMAX_MEDI with FLOORSMAX_AVG) which is same for both the data set.

4. Read Previous Application data and merging with application data

```
In [63]: # graphical representation of columns having % null values
         plt.figure(figsize= (20,4),dpi=300)
         Null_prev.plot(kind = 'bar')
         plt.title ('columns having null values')
         plt.ylabel('% null values')
         plt.show()
```



- Extracting columns with null values over 50%
- Dropping columns over 50%
- Merging both the dataset of both

```
In [64]: # Get the column with null values more than 50%
         Null_prev = Null_prev[Null_prev>50]
         print("Number of columns having null value more than 50% :", len(Null_prev.index))
         print(Null_prev)
```

```
Number of columns having null value more than 50% : 4
AMT_DOWN_PAYMENT            53.64
RATE_DOWN_PAYMENT           53.64
RATE_INTEREST_PRIMARY       99.64
RATE_INTEREST_PRIVILEGED    99.64
dtype: float64
```

Droped all columns from Dataframe for which missing value percentage are more than 50%.

'AMT_DOWN_PAYMENT', 'RATE_DOWN_PAYMENT', 'RATE_INTEREST_PRIMARY','RATE_INTEREST_PRIVILEGED'

```
In [65]: # removed 4 columns having null percentage more than 50%.
         df_previous_application = df_previous_application.drop(Null_prev.index, axis =1)
         df_previous_application.shape
```

```
Out[65]: (1670214, 33)
```

```
In [66]: # Merging the Application dataset with previous appliaction dataset

         df_combine = pd.merge(left=df_application_data, right=df_previous_application, how='inner', on='SK_ID_CURR', suffixes=('_x', '_y
         df_combine.shape
```

```
Out[66]: (1413646, 76)
```

```
In [67]: df_combine.head()
```

Out[67]:

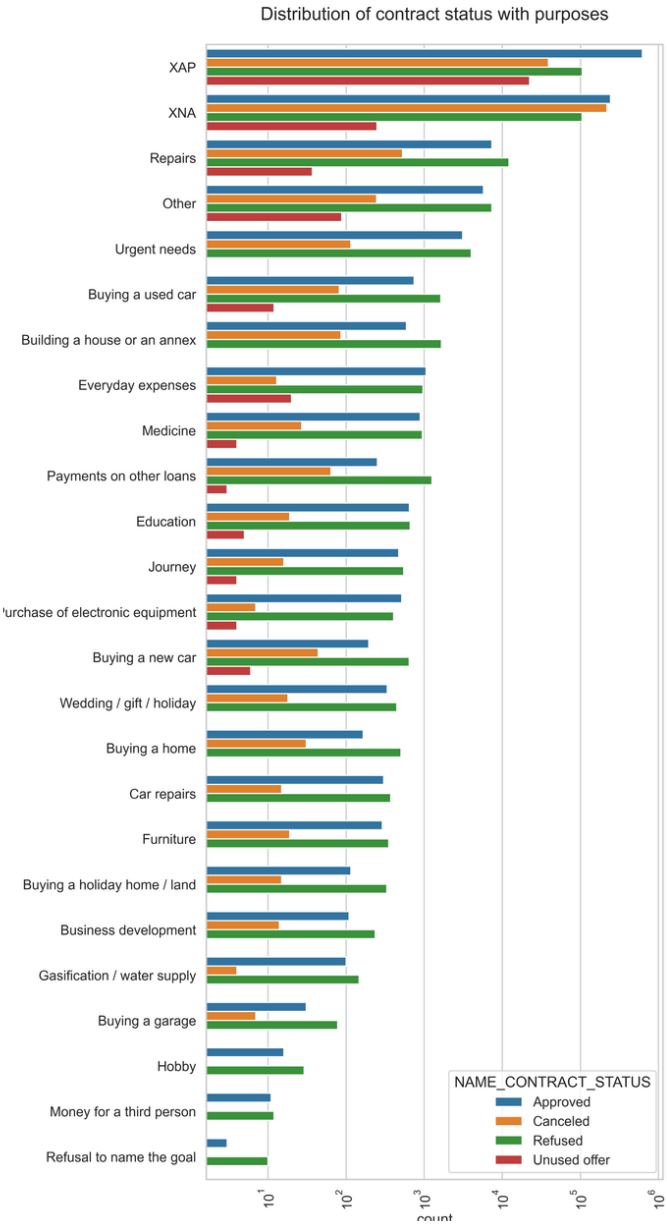
	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CR
0	100002	1	Cash loans	M	N	Y	0	202500.0	4
1	100003	0	Cash loans	F	N	N	0	270000.0	12
2	100003	0	Cash loans	F	N	N	0	270000.0	12

Univariate Analysis

Inferences: Most rejection of loans came from purpose 'repairs'. For education purposes we have equal number of approves and rejection paying other loans and buying a new car is having significant higher rejection than approves.

Performing univariate analysis

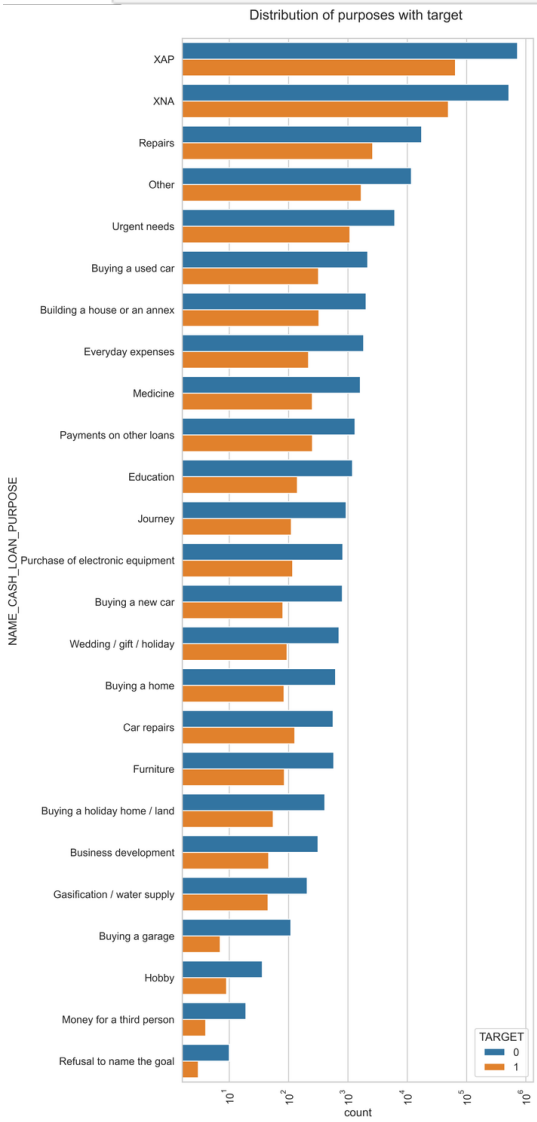
In [70]:
```
# Distribution of contract status in logarithmic scale
# Distribution of contract status in logarithmic scale

sns.set_style('whitegrid')
sns.set_context('talk')

plt.figure(figsize=(10,25),dpi = 300)
plt.rcParams["axes.labelsize"] = 20
plt.rcParams['axes.titlesize'] = 22
plt.rcParams['axes.titlepad'] = 30
plt.xticks(rotation=90)
plt.xscale('log')
plt.title('Distribution of contract status with purposes')
ax = sns.countplot(data = df_combine, y= 'NAME_CASH_LOAN_PURPOSE',
                   order=df_combine['NAME_CASH_LOAN_PURPOSE'].value_counts().index,hue = 'NAME_CONTRACT_STATUS')
```



Univariate Analysis

Inference:
Loan purposes with 'Repairs' are facing more difficulties in payment on time. There are few places where loan payment is significantly higher than facing difficulties. They are 'Buying a garage', 'Business developemt', 'Buying land', 'Buying a new car' and 'Education' Hence we can focus on these for which the client is having for minimal payment difficulties.

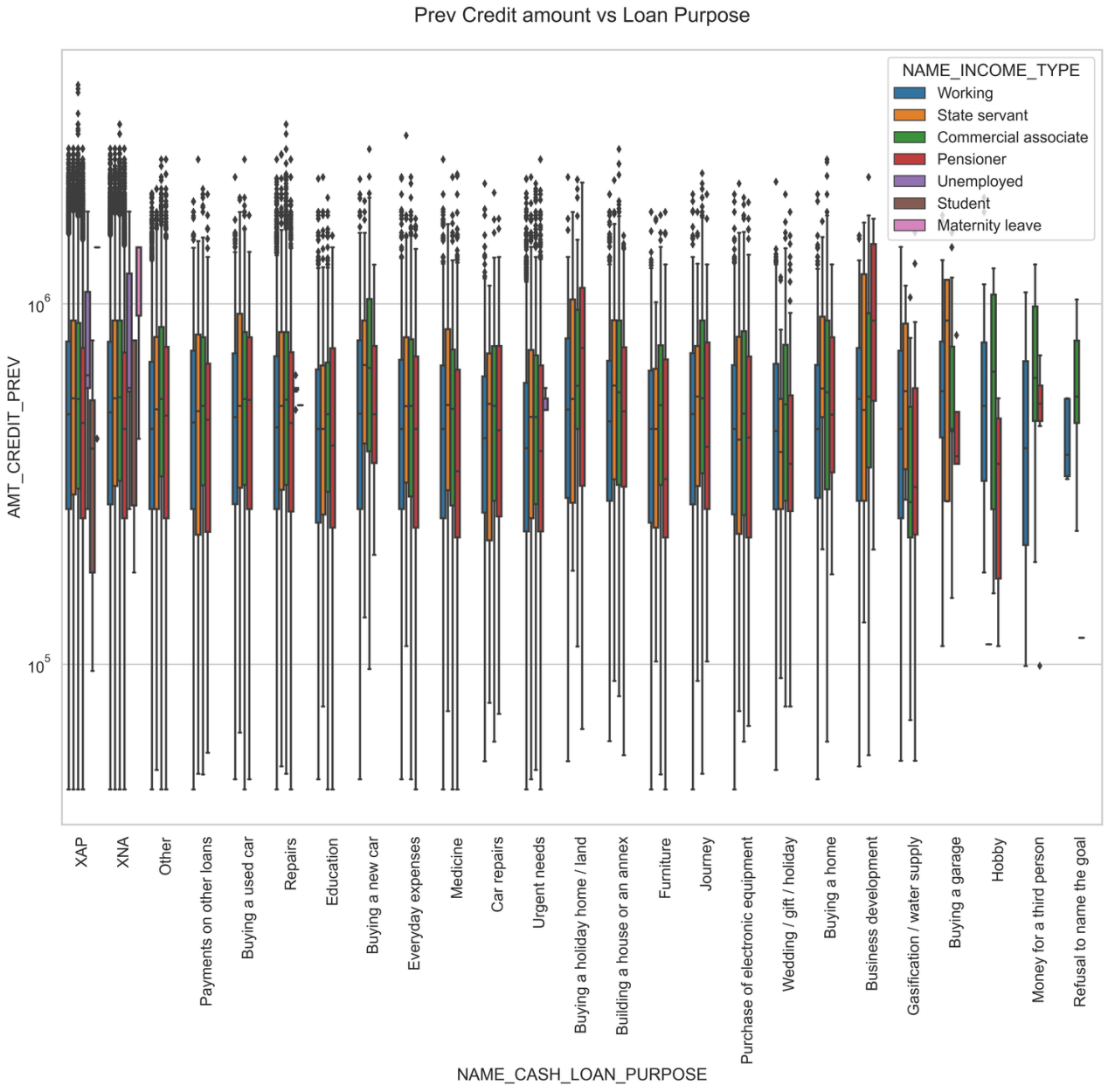
In [71]:
```
# Distribution of contract status

sns.set_style('whitegrid')
sns.set_context('talk')

plt.figure(figsize=(10,30),dpi = 300)
plt.rcParams["axes.labelsize"] = 20
plt.rcParams['axes.titlesize'] = 22
plt.rcParams['axes.titlepad'] = 30
plt.xticks(rotation=90)
plt.xscale('log')
plt.title('Distribution of purposes with target ')
ax = sns.countplot(data = df_combine, y= 'NAME_CASH_LOAN_PURPOSE',
                   order=df_combine['NAME_CASH_LOAN_PURPOSE'].value_counts().index,hue = 'TARGET')
```



Bivariate Analysis

The credit amount of Loan purposes like 'Buying a home', 'Buying a land', 'Buying a new car' and 'Building a house' is higher. Income type of state servants have a significant amount of credit applied Money for third person or a Hobby is having less credits applied for.

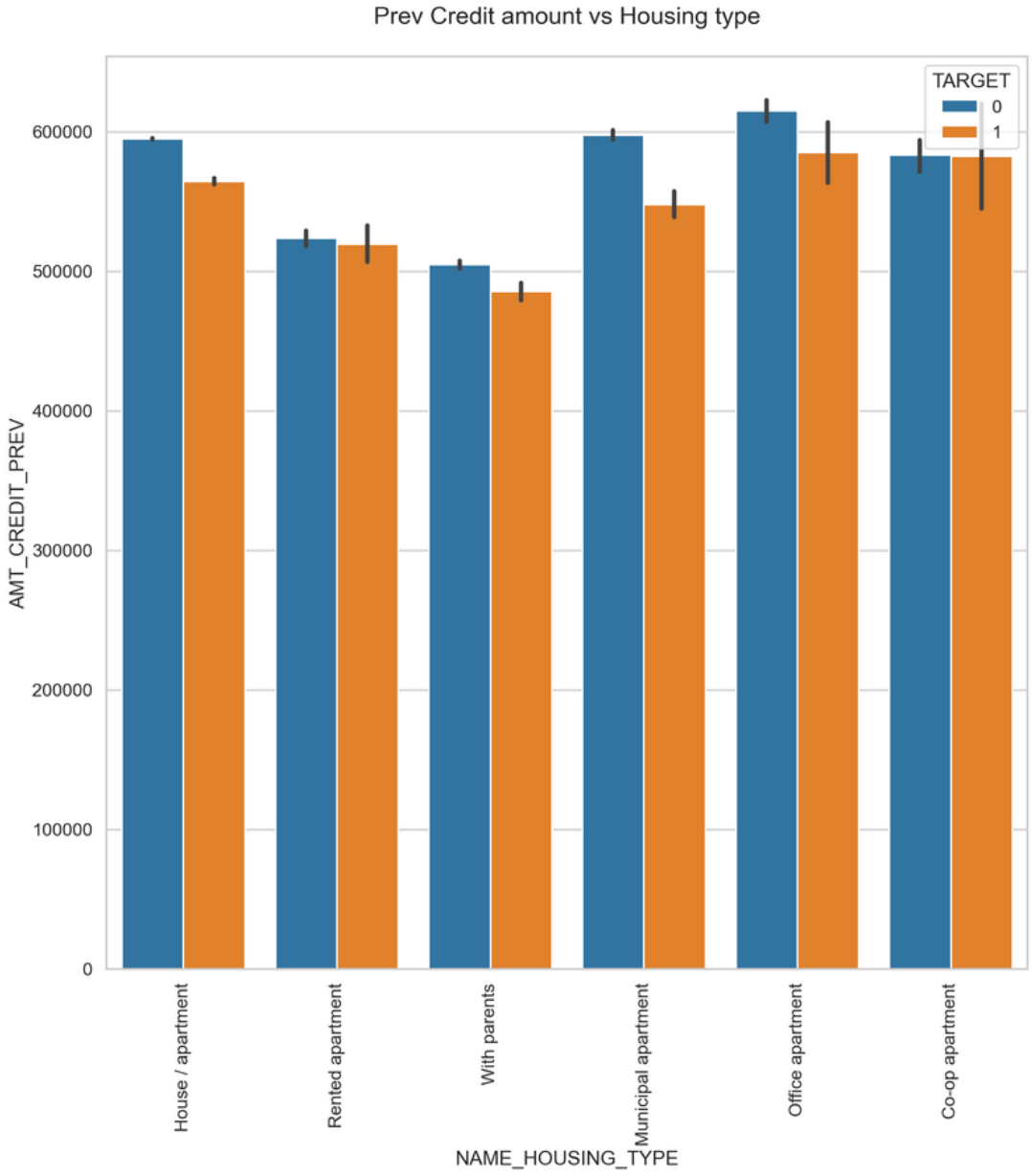
```
In [72]: # Box plotting for Credit amount in logarithmic scale

         plt.figure(figsize=(20,15),dpi = 300)
         plt.xticks(rotation=90)
         plt.yscale('log')
         sns.boxplot(data =df_combine, x='NAME_CASH_LOAN_PURPOSE',
                     hue='NAME_INCOME_TYPE',y='AMT_CREDIT_PREV',orient='v')
         plt.title('Prev Credit amount vs Loan Purpose')
         plt.show()
```



Bivariate Analysis

For Housing type, office apartment has higher credit of target 0 and co-op apartment has higher credit of target 1. So, we can conclude that bank should avoid giving loans to the housing type of co-op apartment as they are facing difficulties in payment. Bank can focus mostly on housing type with parents or House\apartment or municipal apartment for successful payments.

In [73]:
```
# Box plotting for Credit amount prev vs Housing type in logarithmic scale

plt.figure(figsize=(15,15),dpi = 150)
plt.xticks(rotation=90)
sns.barplot(data =df_combine, y='AMT_CREDIT_PREV',hue='TARGET',x='NAME_HOUSING_TYPE')
plt.title('Prev Credit amount vs Housing type')
plt.show()
```



Conclusion

- To ensure successful payments, banks should prioritize contracts with clients who are students, pensioners, and business owners, and who have a housing type other than a co-op apartment.
- Banks should reduce their focus on clients who are categorized as "working" since they have the highest rate of unsuccessful payments.
- Although loan applications for repairs have a high rejection rate, there are also challenges with on-time payment. The bank should exercise caution when approving loans for this purpose, as there are significantly high rates of delayed payments.
- Banks should avoid granting loans for co-op apartments, as these clients have difficulties making payments on time. Instead, banks can focus on clients with housing types such as "living with parents," "house/apartment," and "municipal apartment," which have a higher likelihood of successful payments.

VII. Analyzing the Impact of Car Features on Price and Profitability

Project Description:

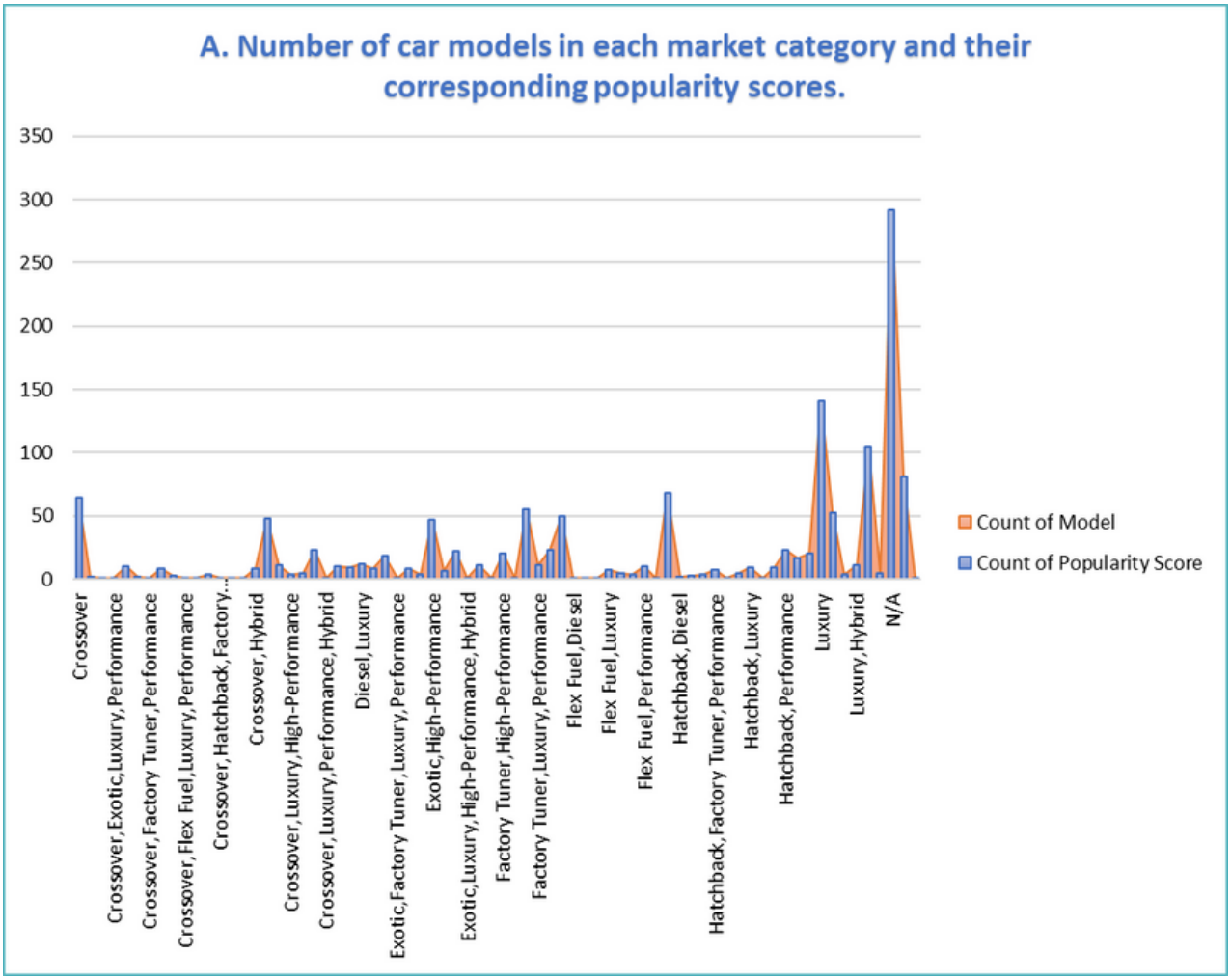
The automotive industry is evolving rapidly with a focus on fuel efficiency, sustainability, and innovation. It's essential to understand the factors that drive consumer demand for cars. The trend towards electric and hybrid vehicles exists, but traditional gasoline-powered cars still dominate the market.

To optimize pricing and product development decisions to maximize profitability while meeting consumer demand. By analyzing a car's features, market category, and pricing, manufacturers can develop a pricing strategy that balances consumer demand with profitability.

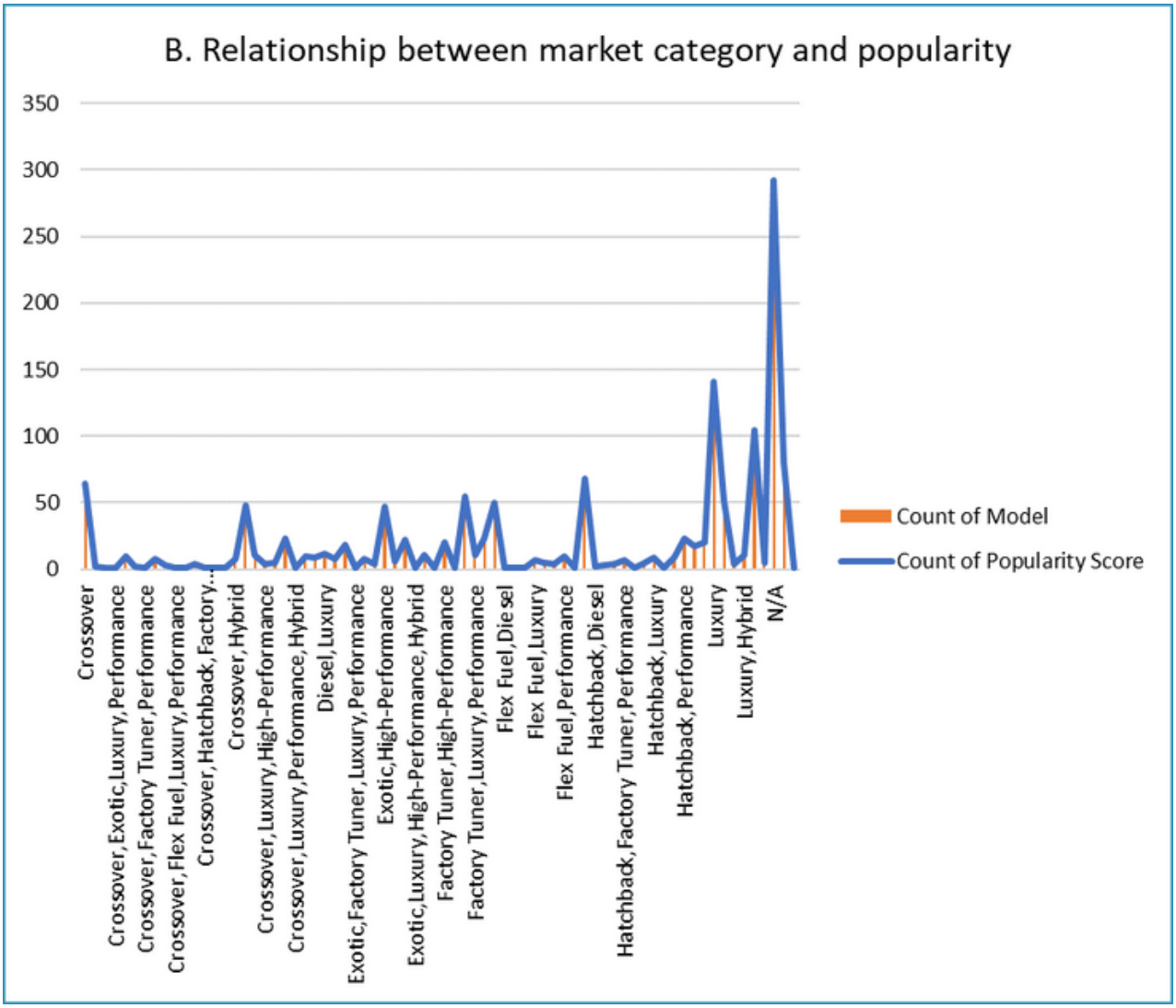
Identifying popular features and categories to focus on for future product development efforts, improve competitiveness, and increase profitability over time.

Findings:

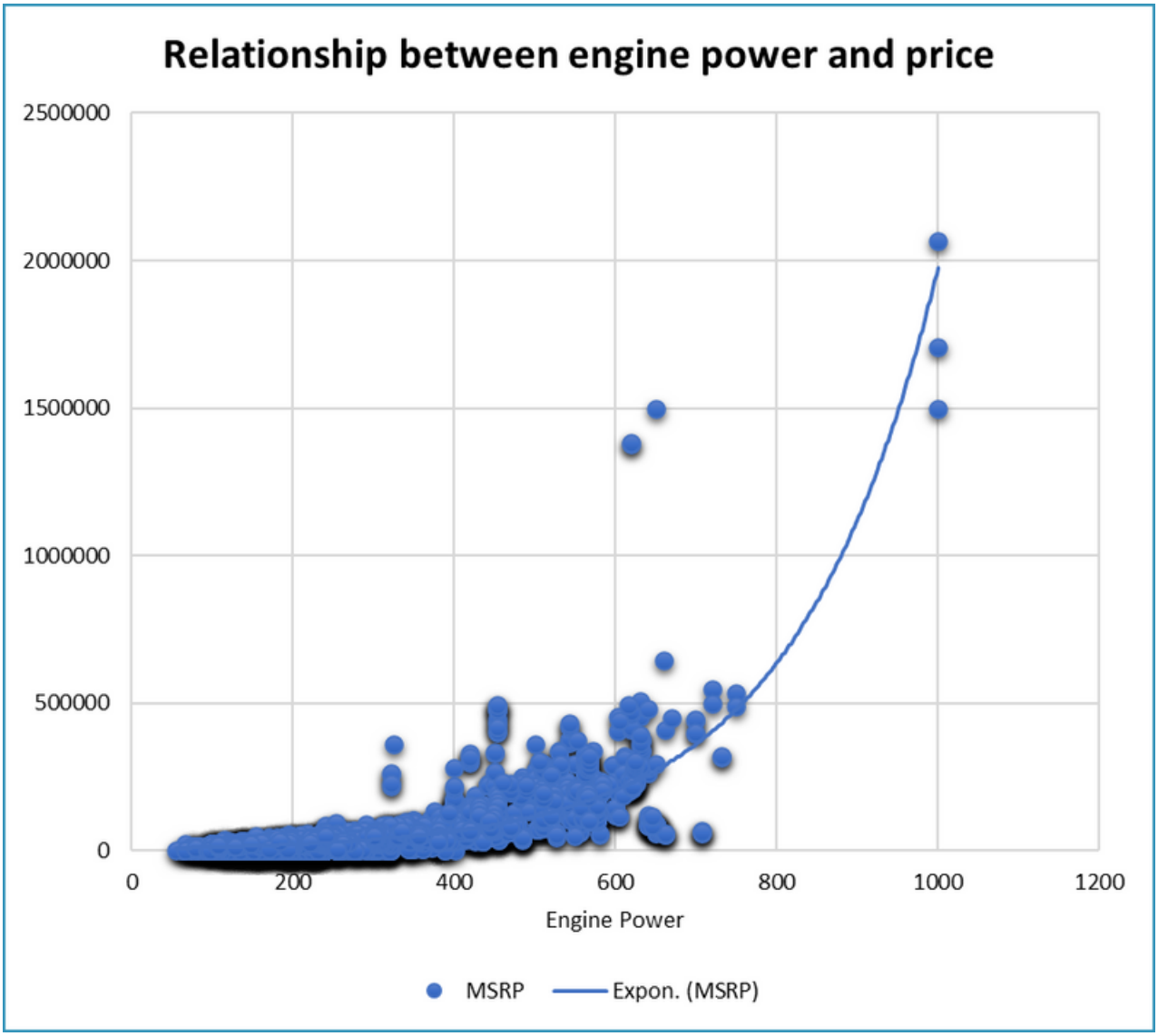
- Understanding the distribution of car models across different market categories.
- Identifying the most popular car models based on their popularity scores, where Luxury stands at highest with 141 count value.
- Gaining insights into the market demand for different car categories and their corresponding popularity scores, where N/A, Luxury, Hybrid, Hatchback, diesel being in the top 5.



- The most popular car models are Ford-Aerostar, Aspire, and Bronco II with a popularity score of 5657, of the market category Hatchback, Hybrid and Factory Tuner.
- The least popular categories are Exotic, High-Performance, with popularity scores of 2.
- Popularity scores for Performance, Diesel, and hatchback of Volkswagen and Volvo are in the middle, with scores of 873.
- Based on the insights, a flat trend line suggests that the data being analyzed is stable and consistent, with no clear trend in any particular direction.
- It is recommended to focus more on the popular categories like Hatchback, Hybrid and Factory Tuner, Performance and try to improve the performance of the less popular categories like Exotic, High-Performance.



- From the chart it can be observed that the relationship between engine power HP and MSRP (price) is positive and the trendline shows that both these variables are directly proportional to each other.
- Majority of the engine power which is between 200 HP to 800 HP are under 5,00,000 price and the highest being at 2M of 1000 HP engine power.



- From the Summary output we can observe that the most dependent variable are Engine HP, Engine Cylinder and MPG of both the highway and city.
- It can be noticed that these variables have the highest coefficient value which denotes that they have a very strong relationship with the independent variable.
- So we can conclude that these variables are the key features that will help determine the car's price

SUMMARY OUTPUT

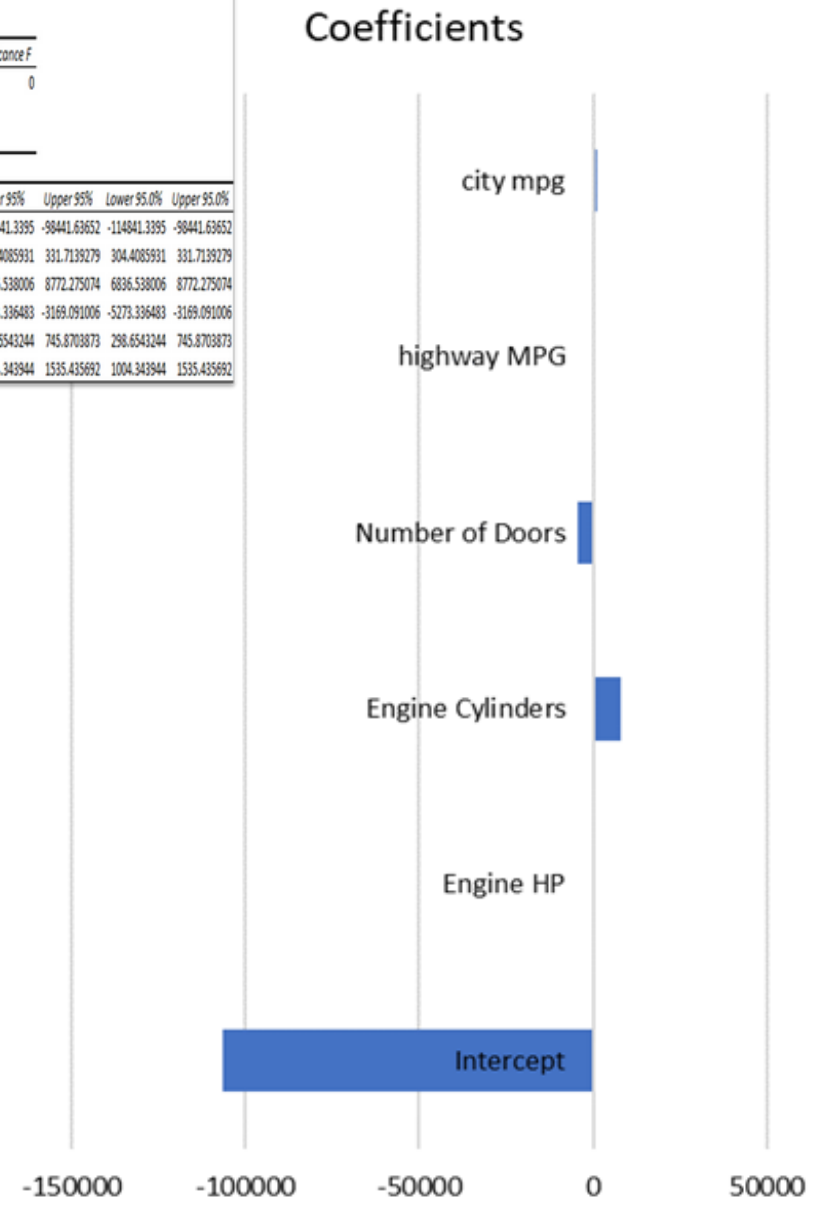
Most important Car features in Determining a car's price

Regression Statistics	
Multiple R	0.666487219
R Square	0.444205213
Adjusted R Squa	0.443930258
Standard Error	46168.11505
Observations	10113

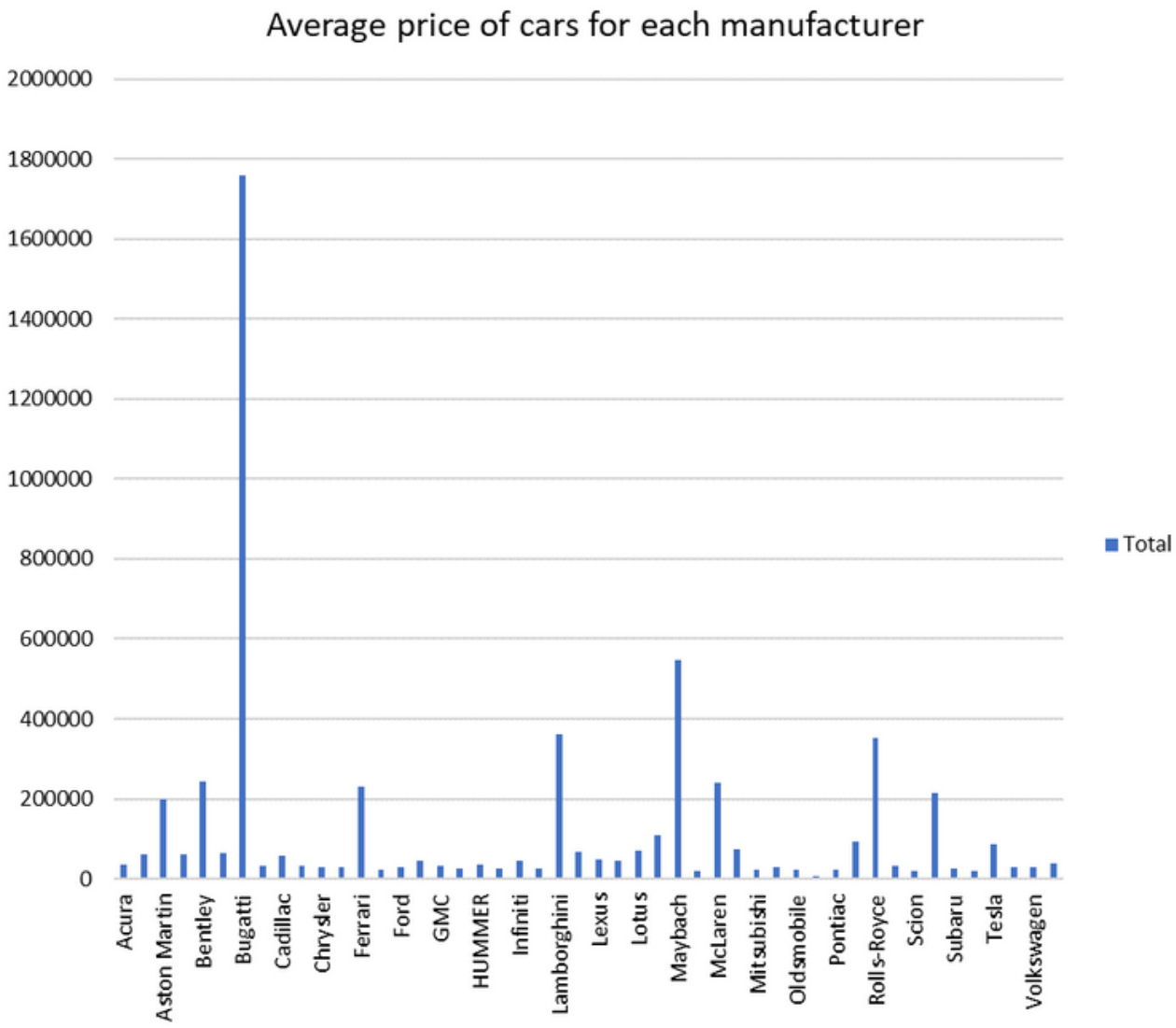
ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	5	1.72177E+13	3.44354E+12	1615.553868	0
Residual	10107	2.1543E+13	2131494847		
Total	10112	3.87607E+13			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-106641.488	4183.173564	-25.49296279	5.4687E-139	-114841.3395	-98441.63652	-114841.3395	-98441.63652
Engine HP	318.0612605	6.964940442	45.66604168	0	304.4085931	331.7139279	304.4085931	331.7139279
Engine Cylinder:	7804.40654	493.7604133	15.80605964	1.3065E-55	6836.538006	8772.275074	6836.538006	8772.275074
Number of Door	-4221.213744	536.7428942	-7.864498608	4.08436E-15	-5273.336483	-3169.091006	-5273.336483	-3169.091006
highway MPG	522.2623558	114.0741641	4.578270284	4.74435E-06	298.6543244	745.8703873	298.6543244	745.8703873
city mpg	1269.889818	135.4688534	9.374035333	8.47746E-21	1004.343944	1535.435692	1004.343944	1535.435692



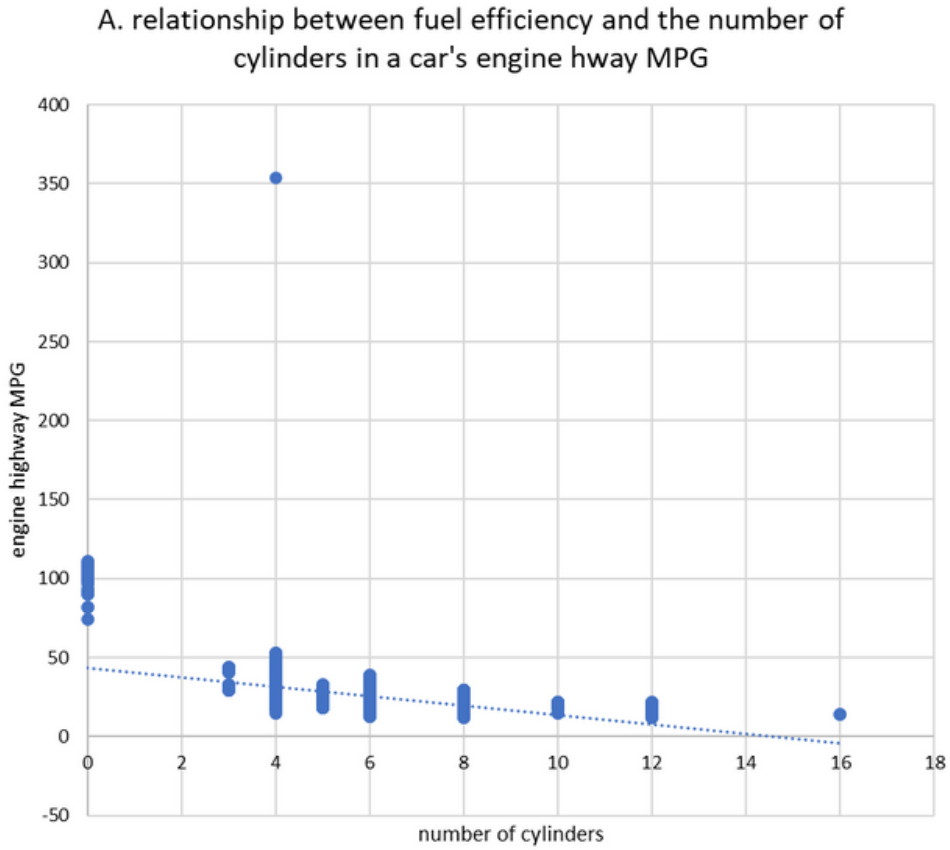
- In this column chart we can observe that Bugatti has significantly higher average car price and the next in line is Maybach, Lamborghini, Rolls-Royce, and Bentley.
- The least average car price manufacturers are Plymouth, Suzuki, Scion, Mazda and Oldsmobile.
- The Average car price manufacturers in between are Genesis, Lexus, Lincoln, Volvo and Hummer



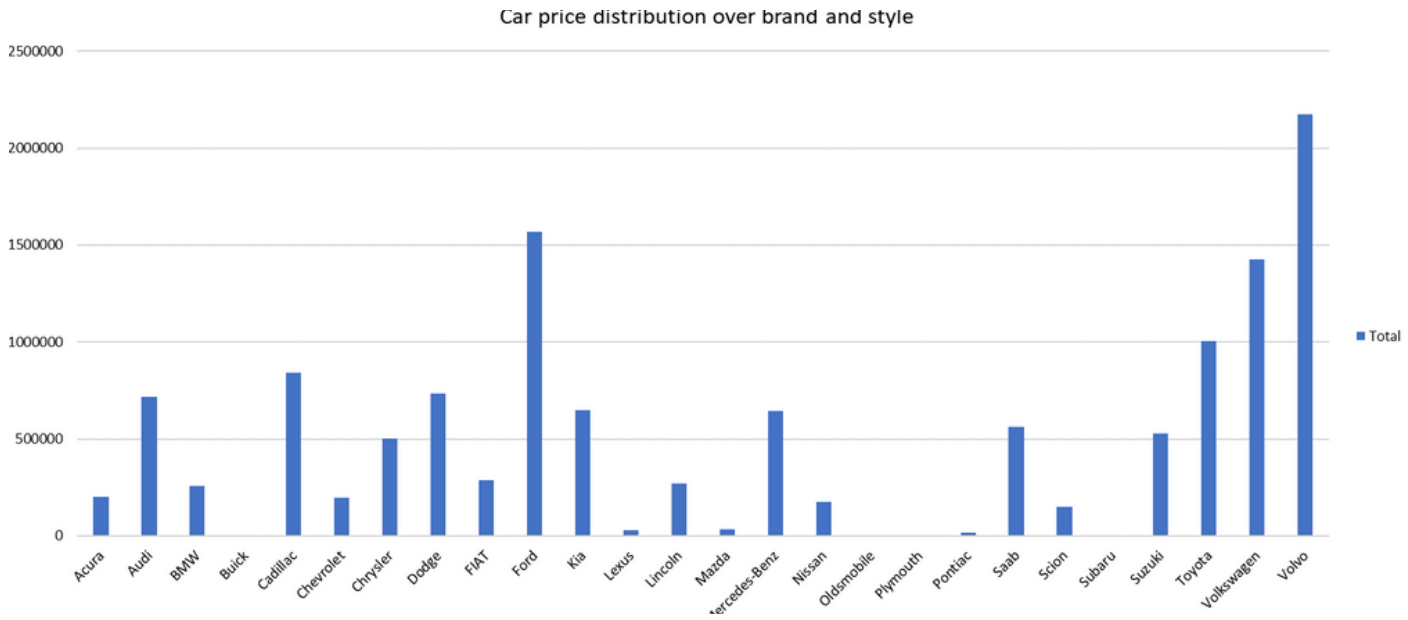
	Engine Cylinders	highway MPG
Engine Cylinders	1	
highway MPG	-0.621605733	1

relationship between fuel efficiency and the number of cylinders in a car's engine

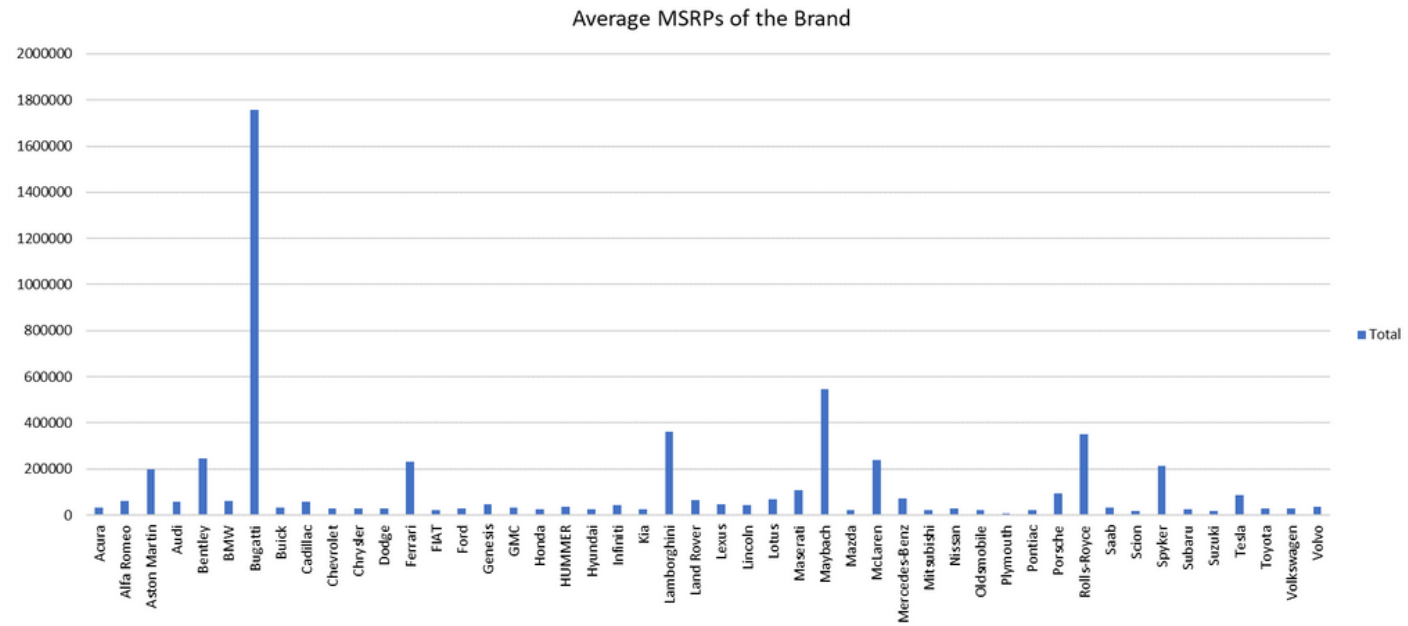
Since the correlation coefficient is negative, it suggests that as the number of cylinders in a car's engine increases, the fuel efficiency (highway MPG) decreases. This could be due to larger engines requiring more fuel to operate and resulting in lower fuel efficiency.



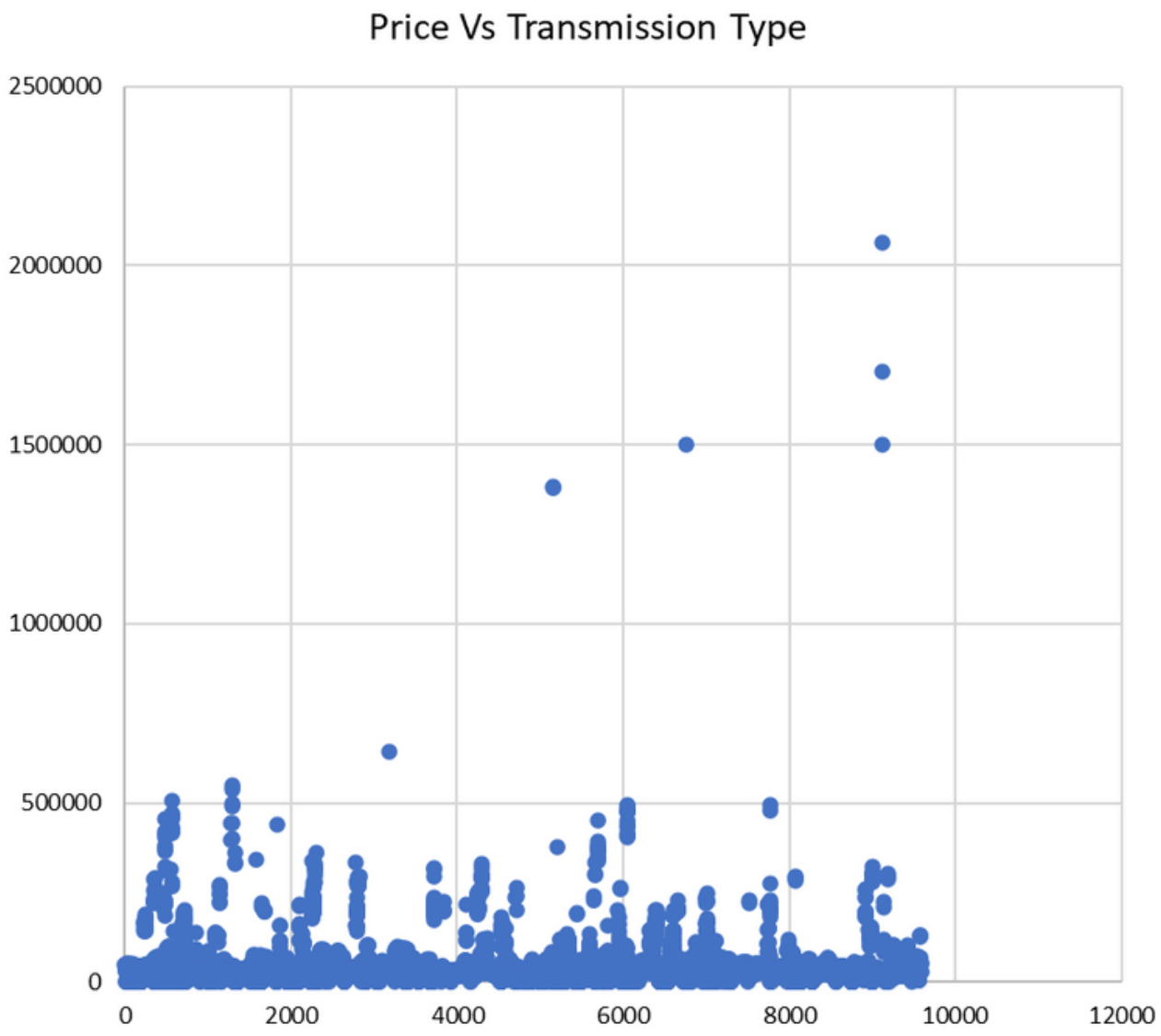
Distribution of car prices by brand and body style



Car brands with the highest and the lowest MSRPs, and how it varies by body style

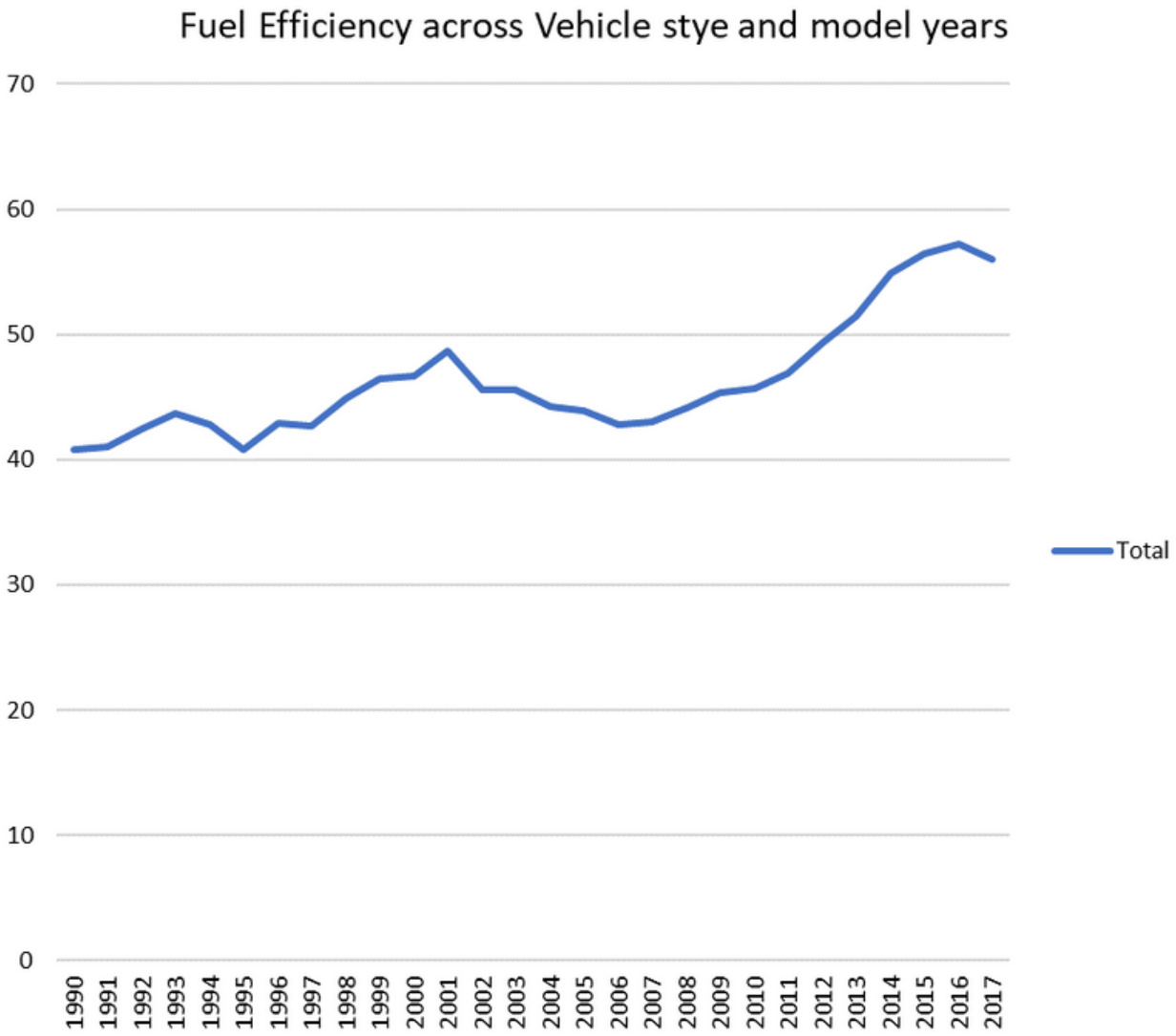


How do the different feature such as transmission type affect the MSRP, and how does this vary by body style?

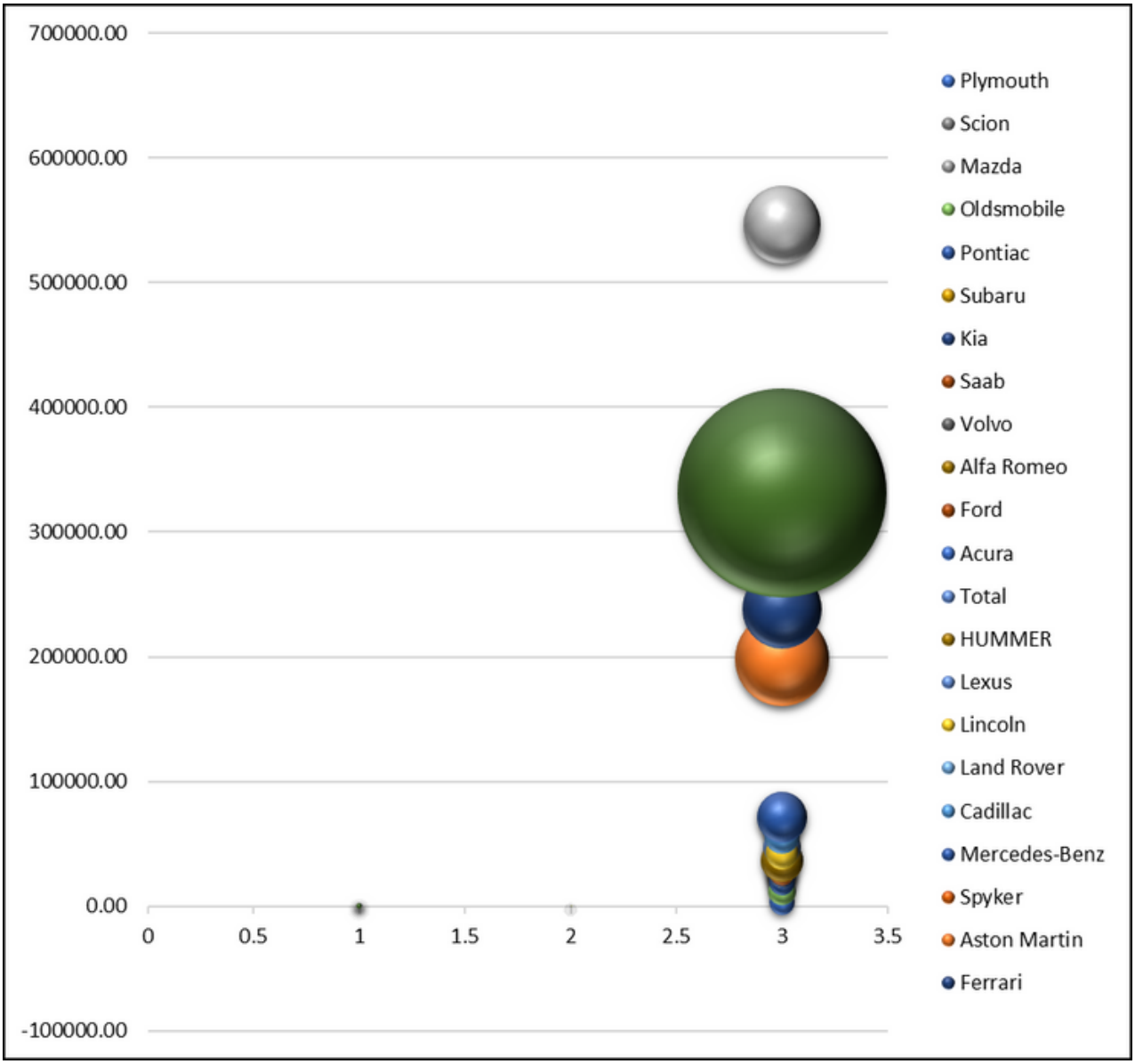


- From the Scatter Plot, it shows that the points are clustered majorly in the bottom section it suggests that cars with a certain transmission type tend to have a higher or lower MSRP.
- The different symbols for each body style make it easier to distinguish between them and identify any patterns in the data.
- The average MSRP for each combination of transmission type and body style is calculated using pivot tables, which helps to further understand the relationship between the MSRP and the transmission type.
- This information can help identify which transmission types and body styles have the highest and lowest average MSRP, and provide insights into how transmission type affects the price of a car.

How does the fuel efficiency of cars vary across different body styles and model years?



- From the line chart, It can be observed that the trend of fuel efficiency (MPG) over time for each body style. The x-axis represents the model years, while the y-axis represents the average MPG. The gradual increase in the line shows that the fuel efficiency of cars has generally improved over time. The slope of each line indicates the rate of increase in fuel efficiency for each body style.
- By comparing the lines on the chart, we can infer which body styles have improved their fuel efficiency at a faster rate than others. Additionally, by calculating the average MPG for each combination of body style and model year using Pivot Tables, we can identify that 2d Hatchback have the highest average fuel efficiency of 50.94 year- 2016 and Sedan, Regular cab and Coupe having lowest average fuel efficiency of 37.93.



- From the 3D Bubble chart, displays a scatter plot with three variables: horsepower, MPG, and MSRP. Each car model is represented by a bubble with the size of the bubble corresponding to the MSRP of the car.
- The position of the bubble on the chart indicates the MPG and horsepower of the car. By assigning different colors to each brand, it makes it easier to visualize how different car brands compare in terms of these variables.
- It shows that certain car brands such as Bugatti tend to have higher horsepower and higher prices, while others may have lower horsepower and lower prices such as Plymouth. The chart can be useful in identifying trends and making comparisons between different car brands

VIII. ABC Call Volume Trend Analysis

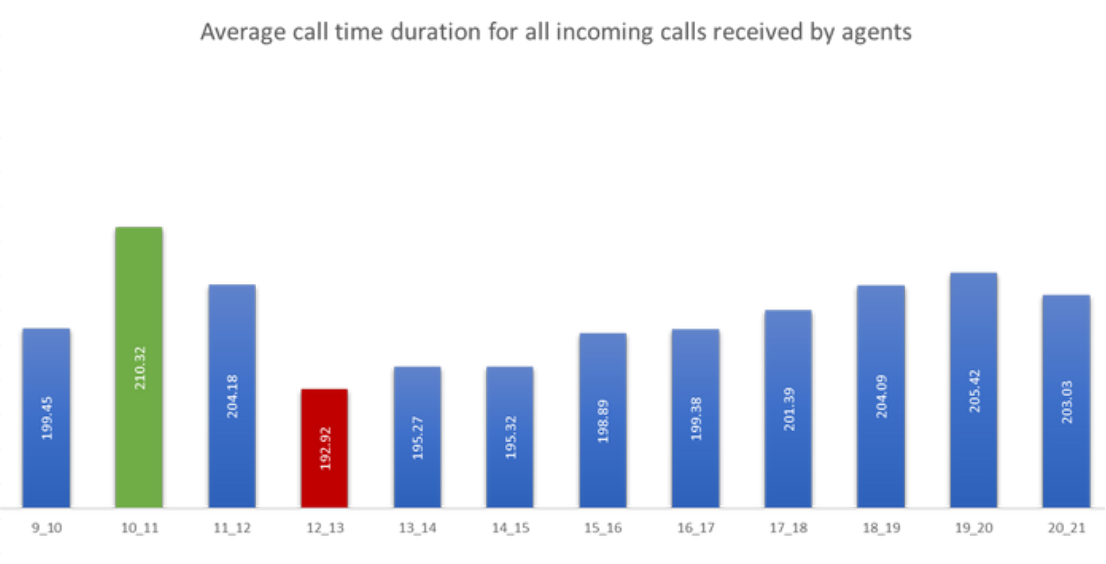
Project Description:

- A Customer Experience (CX) Dataset of Inbound calling team for 23 days, which includes various data points such as Agent_Name, Agent_ID, Queue_Time, Time, Time_Bucket, Duration, Call_Seconds, and call status.
- A CX team analyzes customer feedback and data and performs roles such as CX programs, digital CX, internal communication, VoC, user experience, customer support, and more.
- AI-powered customer experience tools, like IVR, RPA, Predictive Analytics, and Intelligent Routing, have a significant impact. Customer service representatives can work in different roles, such as email, inbound, outbound, or social media support. Inbound support handles customer calls and builds customer loyalty. Advertising is used to increase sales or awareness of a business and can take various forms such as online directories, press, radio, cinema, outdoor advertising, papers, magazines, or TV.

Findings:

*To calculate the **average call time duration** for all incoming calls received by agents (in each Time_Bucket)*

Call_Status	answered
Wrapped _By	Agent
Time_bucket	Average of Call_Seconds (s)
10_11	210.32
11_12	204.18
12_13	192.92
13_14	195.27
14_15	195.32
15_16	198.89
16_17	199.38
17_18	201.39
18_19	204.09
19_20	205.42
20_21	203.03
9_10	199.45
Grand Total	199.81



Insights:

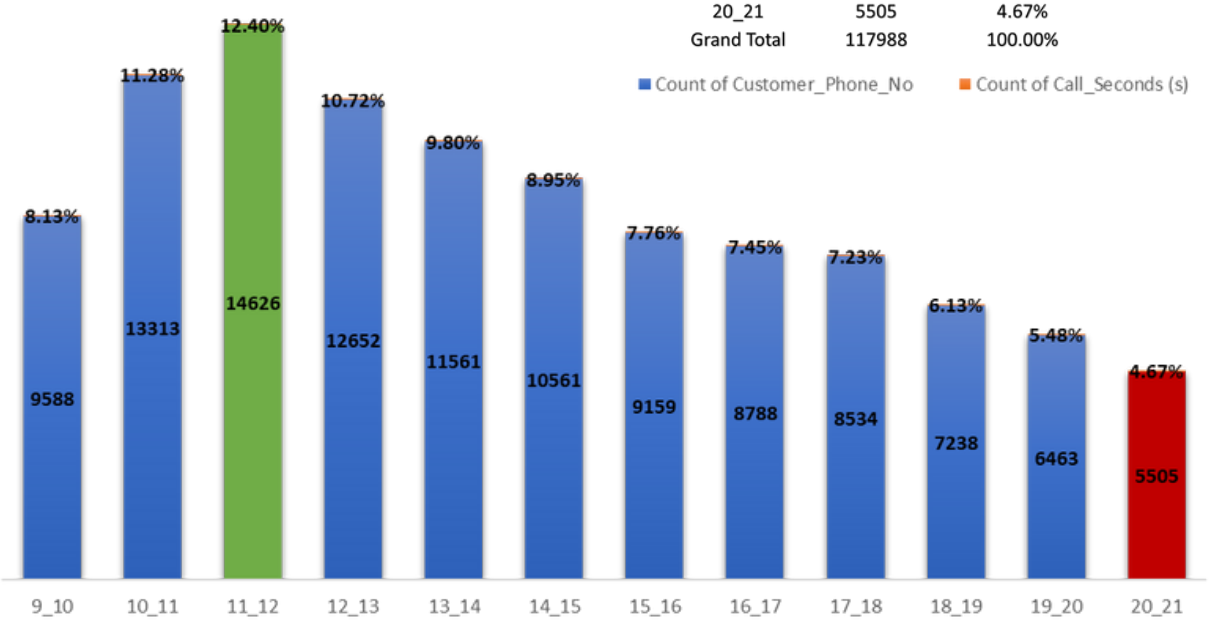
- In the chart it is observed that time_bucket series is applied on the x-axis (row-label) and Average of Call_Seconds (s) in y-axis , which is calculated using pivot table also the Call_status & Wrapped_by has been filtered to “answered” & “agent” respectively.
- In Average of Call_Seconds (s) duration total is 199.81s where the timeframe between 10-11 & 19-20 having the highest number of call duration of 210.32s & 205.42s respectively.
- The timeframe between 12-13 has the lowest call duration of 192.92s.

Total volume/ number of calls coming in via charts/ graphs [Number of calls v/s Time].

Inferences:

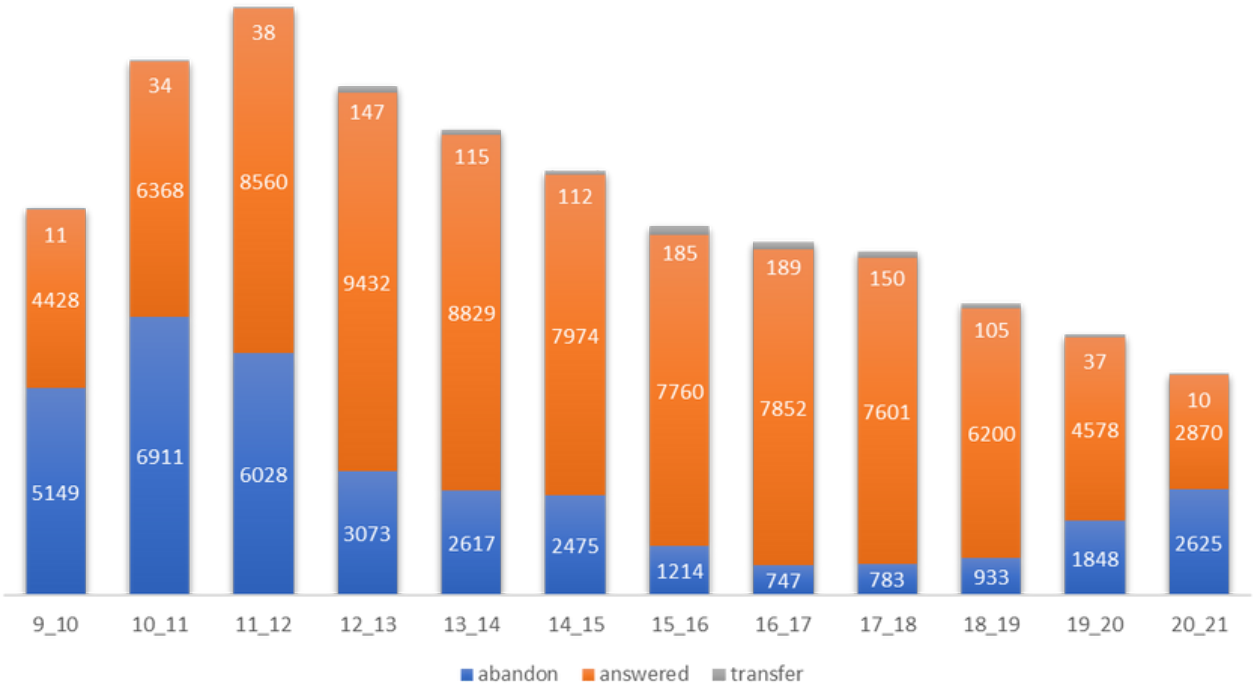
- Timeframe between 11 to 12 has the highest number of calls and 20-21 the least number of calls
- The count is converted to percentage where the total count percent is considered in terms of 100%

Time_bucket	Count of Customer_Phone _No	Count % of Call_Seconds (s)
9_10	9588	8.13%
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%
Grand Total	117988	100.00%



Count of Customer_Phone_No Call_Status

Time_bucket	abandon	answered	transfer	Grand Total
9_10	5149	4428	11	9588
10_11	6911	6368	34	13313
11_12	6028	8560	38	14626
12_13	3073	9432	147	12652
13_14	2617	8829	115	11561
14_15	2475	7974	112	10561
15_16	1214	7760	185	9159
16_17	747	7852	189	8788
17_18	783	7601	150	8534
18_19	933	6200	105	7238
19_20	1848	4578	37	6463
20_21	2625	2870	10	5505
Grand Total	34403	82452	1133	117988



- It can be noticed from the chart that in the beginning and end, abandon calls are slightly higher as compared to answered and transferred calls.
- The calls are typically highest during the day time.

Drop rate

Agent_Name	abandon	answered	transfer
#N/A	34198		
Grand Total	34403	82452	1133

Time Bucket	No of Agents	Abandon Rate	Calls Each Day	Avg Call Duration	Avg Queue Time
09_10	42	53.70%	416.87	92.01	82.86
10_11	51	51.91%	578.83	97.42	83.25
11_12	59	41.21%	635.91	116.78	72.32
12_13	60	24.29%	550.09	144.73	41.66
13_14	58	22.64%	502.65	149.54	41.8
14_15	60	23.44%	459.17	146.97	43.6
15_16	58	13.25%	398.22	169.9	29.88
16_17	58	8.50%	382.09	181.44	23.54
17_18	58	9.18%	371.04	179.72	23.75
18_19	59	12.89%	314.7	174.32	34.09
19_20	52	28.59%	281	144.58	58.69
20_21	27	47.68%	239.35	105.95	75.28
Grand Total	66	29.16%	5,129.91	139.53	52.17

High abandon rates are observed during specific times of the day, while the lowest rates occur at a particular time. Interestingly, even though the number of calls is low during the 9-10 time slot, the drop rate remains very high. To determine the total time call center agents spend talking to customers within a specific time slot, the average call duration for each bucket is available. Using this information, we can estimate the number of agents required to maintain a drop rate below 10%. Additionally, it is evident that time slots with longer average call durations and shorter average queue times tend to have lower drop rates. It is important to incorporate a tolerance level in all calculations to account for errors.

Time_bucket	9 AM shift	10 AM shift	12 PM shift	Agents req	Calculated Agents req
9pm-10pm	50	0	0	50	44.88
10pm-11pm	50	20	0	70	64.04
11pm-12-am	50	20	0	70	73.96
12am-1am	25	10	40	75	62.65
1am-2am	0	20	40	60	58.47
2am-3am	50	0	20	70	53.57
3am-4am	25	20	40	85	48.84
4am-5am	50	20	0	70	47.6
5am-6am	50	10	20	80	46.01
6am-7am	0	20	30	50	39.48
7am-8am	0	0	40	40	34.78
8am-9am	0	0	30	30	26.58

Formula used to calculate the agents required:
Agents Req Calculated = **1.1 * calls Each Day * 2 * (avg call duration + avg queue time)**
where 1.1 and 2 denotes the tolerance of no. of calls and waiting time respectively.
total manpower required = **9 AM shift + 10 AM Shift + 12 PM Shift** = **50 + 20 + 40 = 110**
New Manpower to be added = **req manpower - available employees** = **110 - 66**
= **44**
It is suggested to add at least 44 new employees to reduce the abandon rate from 30% to 10%

Time Bucket	Men Required	Total Men	8 AM Shift	5 PM Shift	2 AM Shift
08_09	42.93	80	60	0	20
09_10	44.88	90	70	0	20
10_11	64.04	70	70	0	0
11_12	73.96	70	70	0	0
12_13	62.65	60	60	0	0
13_14	58.47	60	60	0	0
14_15	53.57	60	60	0	0
15_16	48.84	50	50	0	0
16_17	47.6	50	50	0	0
17_18	46.01	50	0	50	0
18_19	39.48	50	0	50	0
19_20	34.78	50	0	50	0
20_21	26.58	50	0	50	0
21_22	25.76	25	0	25	0
22_23	25.76	25	0	25	0
22-00	17.17	50	0	50	0
00_01	17.17	50	0	50	0
01_02	8.59	70	0	50	20
02_03	8.59	70	0	50	20
03_04	8.59	20	0	0	20
04_05	8.59	10	0	0	10
05_06	8.59	10	0	0	10
06_07	25.76	20	0	0	20
07_08	34.34	20	0	0	20
08_09	34.34	20	0	0	20
02_03	42.93	80	60	0	20
09_10	44.88	90	70	0	20

the total number of manpower needed:
8 AM shift + 5 PM Shift + 2 AM Shift = 70 + 50 + 20 = 140.
140 - 66 = 74 agents,
we can expect the call abandon rates to be lesser than 10 % after hiring minimum 74 agents

Appendix

- ***Project 1 - Data Analytics Process Link***
- ***Project 2 Instagram User Analytics Link***
- ***Project 3 Operation and Metric Analytics Link***
- ***Project 4 Hiring Process Analytics Link***
- ***Project 5 IMDB Movie Analysis Link***
- ***Project 6 Bank Loan Case Study Link***
- ***Project 7 Analyzing Impact of Car Features on Price and Profitability Link***
- ***Project 8 ABC Call Volume Trend Analysis Link***