https://drive.google.com/file/d/1vKIiRKf-dOTIq_gFiHulLfI3G0RVt28d/view?usp=sharing

In [1]:
```
!gdown 1vKIiRKf-dOTIq_gFiHulLfI3G0RVt28d
```

```
Downloading...
From: https://drive.google.com/uc?id=1vKIiRKf-dOTIq_gFiHulLfI3G0RVt28d
To: C:\Users\91944\ola_driver_scaler.CSV

  0%|          | 0.00/1.13M [00:00<?, ?B/s]
 46%|####6     | 524k/1.13M [00:00<00:00, 1.59MB/s]
100%|##########| 1.13M/1.13M [00:00<00:00, 2.83MB/s]
```

Load Dataset

In [4]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
df  = pd.read_csv("ola_driver_scaler.CSV")
df1 = df.copy(deep=True)
```

In [9]:
```
df.head()
```

Out[9]:

	Unnamed: 0	MMM-YY	Driver_ID	Age	Gender	City	Education_Level	Income	Dateofjoining	LastW
0	0	01/01/19	1	28.0	0.0	C23	2	57387	24/12/18	
1	1	02/01/19	1	28.0	0.0	C23	2	57387	24/12/18	
2	2	03/01/19	1	28.0	0.0	C23	2	57387	24/12/18	
3	3	11/01/20	2	31.0	0.0	C7	2	67016	11/06/20	
4	4	12/01/20	2	31.0	0.0	C7	2	67016	11/06/20	

In [5]:
```
df = df.drop("Unnamed: 0",axis = 1)
```

Shape

In [6]:
```
df.shape
```

Out[6]:
```
(19104, 14)
```

Ola Dataset contains 19104 rows and 14 columns.

In [11]:
```
df.describe()
```

Out[11]:

	Driver_ID	Age	Gender	Education_Level	Income	Joining Designation	
count	19104.000000	19043.000000	19052.000000	19104.000000	19104.000000	19104.000000	19104
mean	1415.591133	34.668435	0.418749	1.021671	65652.025126	1.690536	2
std	810.705321	6.257912	0.493367	0.800167	30914.515344	0.836984	1
min	1.000000	21.000000	0.000000	0.000000	10747.000000	1.000000	1
25%	710.000000	30.000000	0.000000	0.000000	42383.000000	1.000000	1
50%	1417.000000	34.000000	0.000000	1.000000	60087.000000	1.000000	2
75%	2137.000000	39.000000	1.000000	2.000000	83969.000000	2.000000	3
max	2788.000000	58.000000	1.000000	2.000000	188418.000000	5.000000	5

In [12]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19104 entries, 0 to 19103
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   MMM-YY                19104 non-null  object
 1   Driver_ID             19104 non-null  int64
 2   Age                   19043 non-null  float64
 3   Gender                19052 non-null  float64
 4   City                  19104 non-null  object
 5   Education_Level       19104 non-null  int64
 6   Income                19104 non-null  int64
 7   Dateofjoining         19104 non-null  object
 8   LastWorkingDate       1616 non-null   object
 9   Joining Designation   19104 non-null  int64
 10  Grade                 19104 non-null  int64
 11  Total Business Value  19104 non-null  int64
 12  Quarterly Rating      19104 non-null  int64
dtypes: float64(2), int64(7), object(4)
memory usage: 1.9+ MB
```

In [ ]:

Datatype Conversion

In [6]:

```
df["MMM-YY"] = pd.to_datetime(df["MMM-YY"])
df["Dateofjoining"] = pd.to_datetime(df["Dateofjoining"])
df["LastWorkingDate"] = pd.to_datetime(df["LastWorkingDate"])
```

In [187…

```
df.isna().sum()
```

Out[187]:
```
MMM-YY                     0
Driver_ID                  0
Age                       61
Gender                    52
City                       0
Education_Level            0
Income                     0
Dateofjoining              0
LastWorkingDate        17488
Joining Designation        0
Grade                      0
Total Business Value       0
Quarterly Rating           0
dtype: int64
```

In [188…

DATA PROCESSING AND FEATURE ENGINEERING

Create a column which tells whether the quarterly rating has increased for that driver - for those whose quarterly rating has increased we assign the value 1

In [7]:
```
Q1 = df.groupby("Driver_ID").agg({"Quarterly Rating":'first'}) ["Quarterly Rating"].re
Q2 = df.groupby("Driver_ID").agg({"Quarterly Rating":'last'}) ["Quarterly Rating"].res
```

In [8]:
```
Q1.isna().sum(),Q2.isna().sum()
```

Out[8]:
```
(Driver_ID           0
 Quarterly Rating    0
 dtype: int64,
 Driver_ID           0
 Quarterly Rating    0
 dtype: int64)
```

In [9]:
```
new_feature = Q1.merge(Q2,on = "Driver_ID")
new_feature
```

Out[9]:

	Driver_ID	Quarterly Rating_x	Quarterly Rating_y
0	1	2	2
1	2	1	1
2	4	1	1
3	5	1	1
4	6	1	2
...	...	...	...
2376	2784	3	4
2377	2785	1	1
2378	2786	2	1
2379	2787	2	1
2380	2788	1	2

2381 rows × 3 columns

In [10]:
```
new_feature["Promotion"] = np.where(new_feature["Quarterly Rating_x"] == new_feature['
```

Target variable creation: Create a column called target which tells whether the driver has left the company- driver whose last working day is present will have the value 1

In [11]:
```
target_creation = df.groupby("Driver_ID").agg({"LastWorkingDate":'last'}) ["LastWorkir
```

In [12]:
```
target_creation["LastWorkingDate"].replace({True : 1 ,False : 0},inplace=True)
```

In [13]:
```
target_creation = target_creation.rename(columns = {"LastWorkingDate": "Target"})
target_creation.head()
```

Out[13]:

	Driver_ID	Target
0	1	0
1	2	1
2	4	0
3	5	0
4	6	1

Create a column which tells whether the monthly income has increased for that driver - for those whose monthly income has increased we assign the value 1.

In [14]:
```
Inc_1 = df.groupby("Driver_ID").agg({"Income":'first'}) ["Income"].reset_index()
Inc_2 = df.groupby("Driver_ID").agg({"Income":'last'})  ["Income"].reset_index()
```

In [15]:
```
Inc_raise = Inc_1.merge(Inc_2 , on ="Driver_ID")
Inc_raise
```

Out[15]:

	Driver_ID	Income_x	Income_y
0	1	57387	57387
1	2	67016	67016
2	4	65603	65603
3	5	46368	46368
4	6	78728	78728
...	...	...	...
2376	2784	82815	82815
2377	2785	12105	12105
2378	2786	35370	35370
2379	2787	69498	69498
2380	2788	70254	70254

2381 rows × 3 columns

In [16]:
```
Inc_raise["Raise"] = np.where(Inc_raise["Income_x"] == Inc_raise["Income_y"],0,1)
```

In [17]:
```
Inc_raise.head()
```

Out[17]:

	Driver_ID	Income_x	Income_y	Raise
0	1	57387	57387	0
1	2	67016	67016	0
2	4	65603	65603	0
3	5	46368	46368	0
4	6	78728	78728	0

In [18]:
```
final1 = new_feature.merge(target_creation , on = "Driver_ID")
```

In [19]:
```
final = final1.merge(Inc_raise , on = "Driver_ID")
```

In [241…
```
final.head()
```

Out[241]:

	Driver_ID	Quarterly Rating_x	Quarterly Rating_y	Promotion	Target	Income_x	Income_y	Raise
0	1	2	2	0	0	57387	57387	0
1	2	1	1	0	1	67016	67016	0
2	4	1	1	0	0	65603	65603	0
3	5	1	1	0	0	46368	46368	0
4	6	1	2	1	1	78728	78728	0

Aggregating by Driver_id

In [242...
```
df.head()
```

Out[242]:

	MMM-YY	Driver_ID	Age	Gender	City	Education_Level	Income	Dateofjoining	LastWorkingDate
0	2019-01-01	1	28.0	0.0	C23	2	57387	2018-12-24	NaT
1	2019-02-01	1	28.0	0.0	C23	2	57387	2018-12-24	NaT
2	2019-03-01	1	28.0	0.0	C23	2	57387	2018-12-24	2019-03-11
3	2020-11-01	2	31.0	0.0	C7	2	67016	2020-11-06	NaT
4	2020-12-01	2	31.0	0.0	C7	2	67016	2020-11-06	NaT

In [44]:
```
data = df.groupby(["Driver_ID"]).agg({
    "MMM-YY":"count",
    "Age": "max",
    "Gender": "last",
    "City": "last",
    "Education_Level": "last",
    'Dateofjoining':'first',
    'LastWorkingDate':'last',
    "Income": "last",
    "Joining Designation": "max",
    "Grade": "last",
    "Total Business Value": "sum",
    "Quarterly Rating": "max"
}).reset_index()
```

In [45]:
```
data["month"] = data["Dateofjoining"].dt.month
data["year"]  = data["Dateofjoining"].dt.year
```

In [46]:
```
data = data.rename(columns = {"MMM-YY":"rides_count"})
```

In [23]:
```
final = final[["Driver_ID","Promotion","Target","Raise"]]
```

In [261...
```
final
```

Out[261]:

	Driver_ID	Promotion	Target	Raise
0	1	0	0	0
1	2	0	1	0
2	4	0	0	0
3	5	0	0	0
4	6	1	1	0
...	...	...	...	...
2376	2784	1	1	0
2377	2785	0	0	0
2378	2786	1	0	0
2379	2787	1	0	0
2380	2788	1	1	0

2381 rows × 4 columns

In [47]:
```
data = data.merge(final,on="Driver_ID")
```

In [26]:
```
data.shape
```

Out[26]:
```
(2381, 16)
```

In [49]:
```
data.drop(columns = ["Dateofjoining","LastWorkingDate"],axis=1,inplace=True)
```

In [50]:
```
data.head()
```

Out[50]:

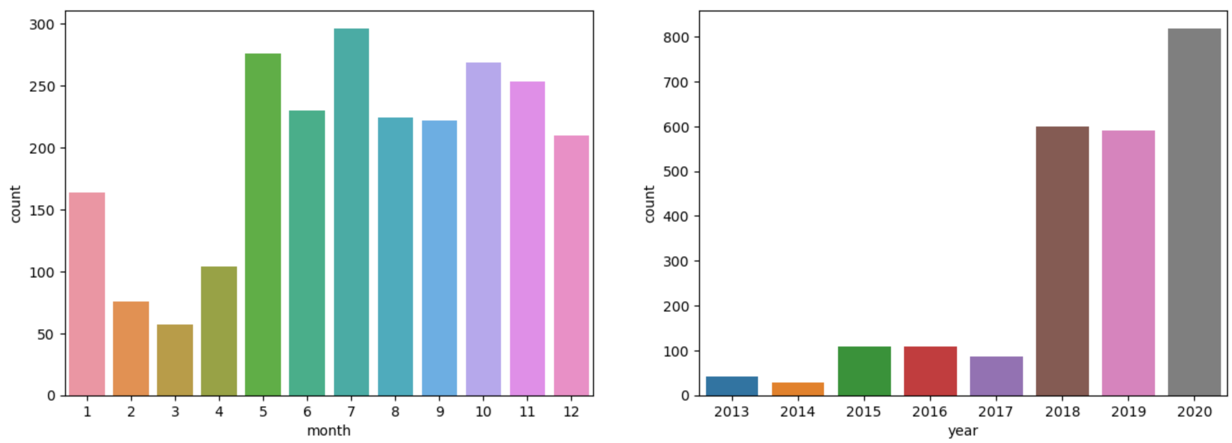
	Driver_ID	rides_count	Age	Gender	City	Education_Level	Income	Joining Designation	Grade	Total Business Value
0	1	3	28.0	0.0	C23	2	57387	1	1	1715580
1	2	2	31.0	0.0	C7	2	67016	2	2	0
2	4	5	43.0	0.0	C13	2	65603	2	2	350000
3	5	3	29.0	0.0	C9	0	46368	1	1	120360
4	6	5	31.0	1.0	C11	1	78728	3	3	1265000

In [ ]:
```

```

Univariate Analysis

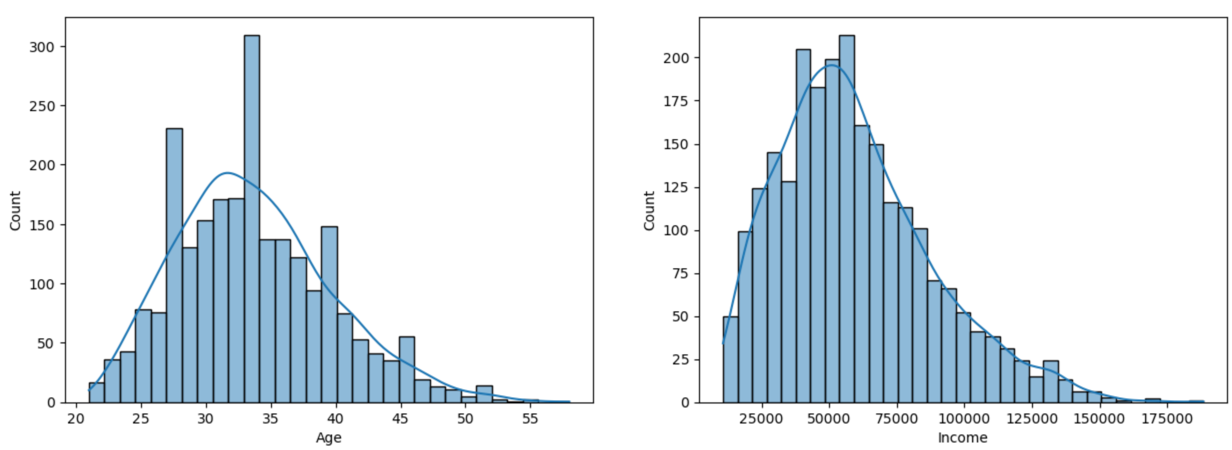
In [28]:
```
fig,axs=plt.subplots(nrows=1,ncols=2,figsize=(15,5))
```

```
cols=["month","year"]
count=0

for i in range(1):
    for j in range(2):
            sns.countplot(data=data,x=cols[count],ax=axs[count])

            count +=1

plt.show()
```



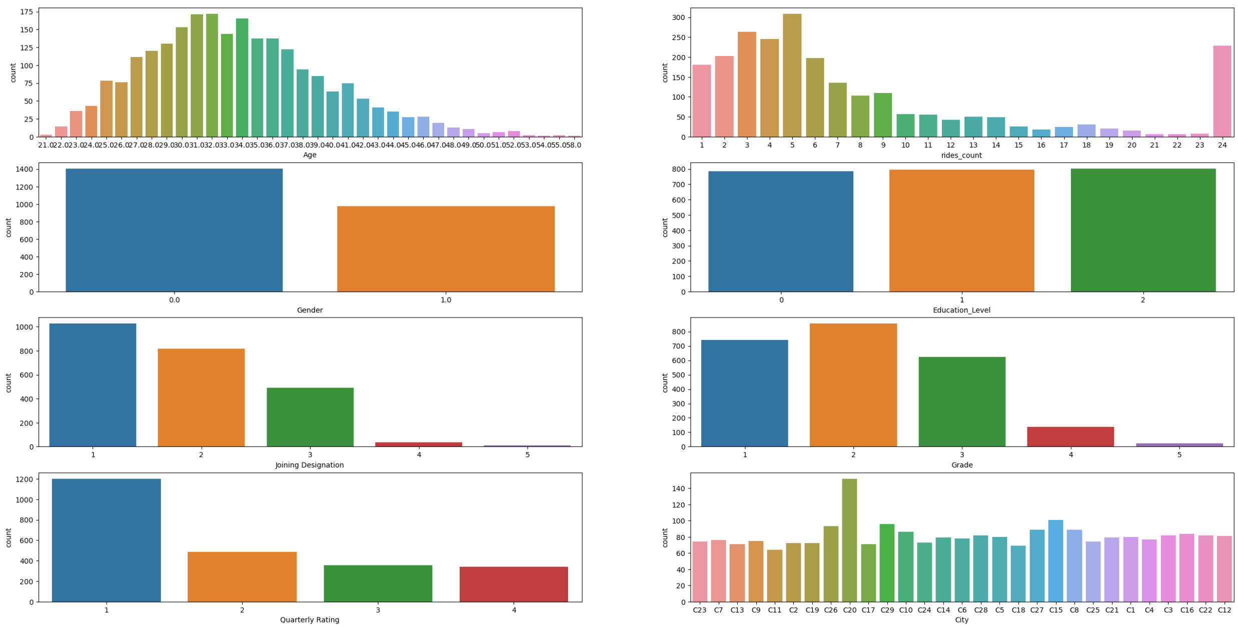
In [ ]:
```
- July received the maximum number of drivers.
- February and March receives the least number of Drivers joining OLA.
- Joining of Drivers receives a boost of about 500% after 2017 and reached its maximun
```

In [289…
```
fig,axs=plt.subplots(nrows=1,ncols=2,figsize=(15,5))

cols=["Age","Income"]
count=0

for i in range(1):
    for j in range(2):
            sns.histplot(data=data,x=cols[count],kde=True,ax=axs[count])

            count +=1

plt.show()
```



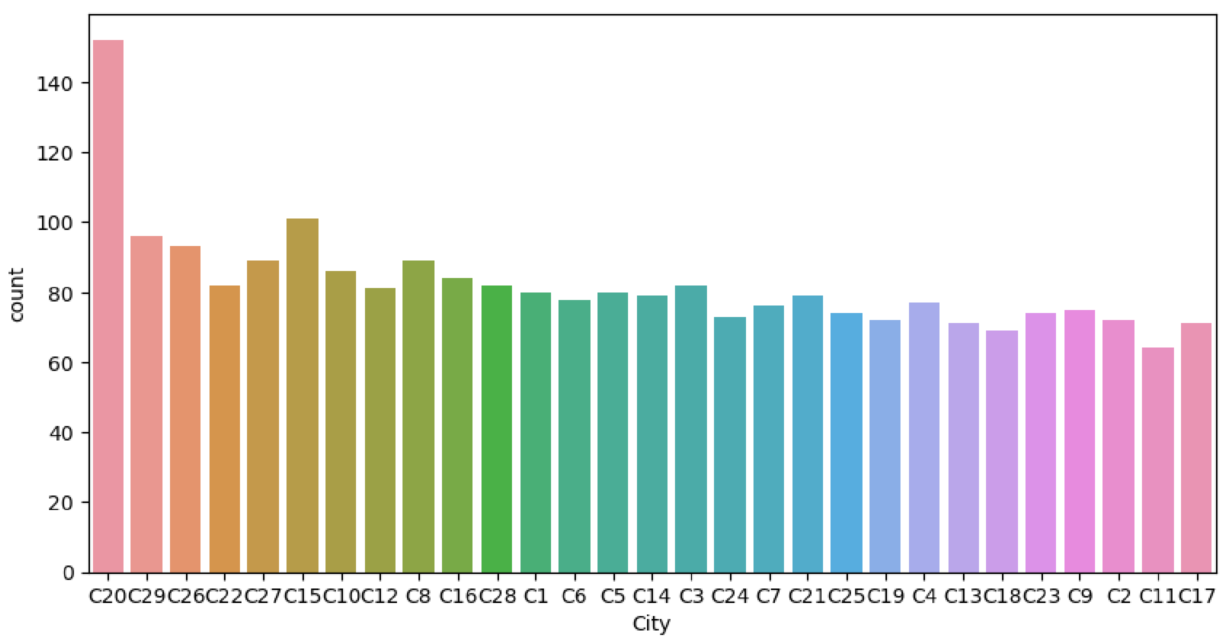
In [298…
```
fig,axs=plt.subplots(nrows=4,ncols=2,figsize=(30,15))
```

```
cols=["Age","rides_count","Gender","Education_Level","Joining Designation","Grade","Qu
count=0

for i in range(4):
    for j in range(2):
            sns.countplot(data=data,x=cols[count],ax=axs[i,j])
            count +=1

plt.show()
```



In [292…
```
fig = plt.subplots(figsize = (10,5))

sns.countplot(data=data,x="City",order = df["City"].value_counts().index)
```

Out[292]: <Axes: xlabel='City', ylabel='count'>



Distribution plot of Age and Income shows that drivers aging 25 - 40 are more and drivers whose income ranging from 50000 - 100000 is more.

Likewise Male Drivers are more than female drivers but their difference is only 400 which is comparable.

Education level – 0 for 10+ ,1 for 12+ ,2 for graduate ,Number of Ola drivers in each of this education level category is equal.

Joining Designation shows majority of drivers belongs to designation 1 and then 2.

Grade 2 & 1 drivers are more i.e., drivers with mid and low experience are more.

Quarterly Rating is 1 for most of drivers which means more cancellation or reschduled the rides.

City C20 received highest ride requests followed by C29 .so more drivers are recommended in that area.

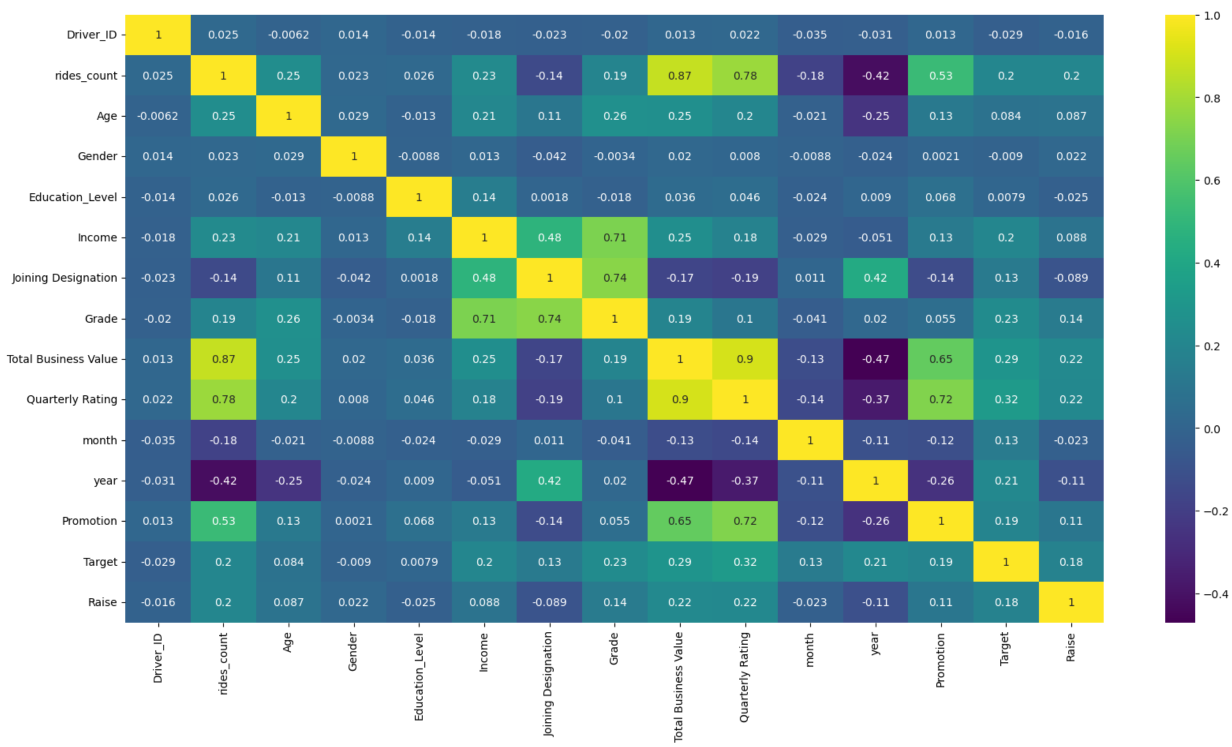
Bivariate Analysis

In [299…

```
df_int = data.select_dtypes(exclude = "object")
```

In [301…

```
plt.figure(figsize=(20,10))
sns.heatmap(df_int.corr(method="spearman"),annot=True,cmap="viridis")
plt.show()
```

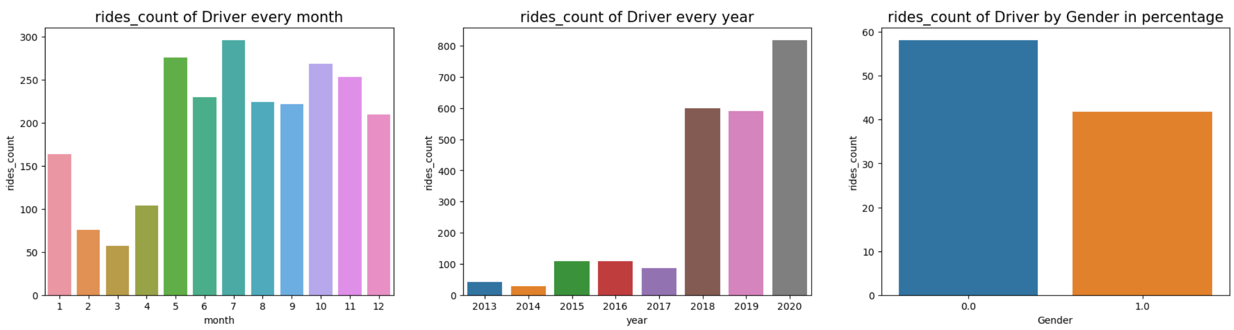


Heatmap shows correlation coefficient of each feature with other .

No of rides count and Total business value are highly positively correlated. No of rides count and Quarterly Rating are highly positively correlated. Quarterly Rating & Total business value are highly correlated with promotion of drivers. Grade & Income , Grade & Joining Designation are highly correlated. Quarterly Rating & Total business value are highly correlated.

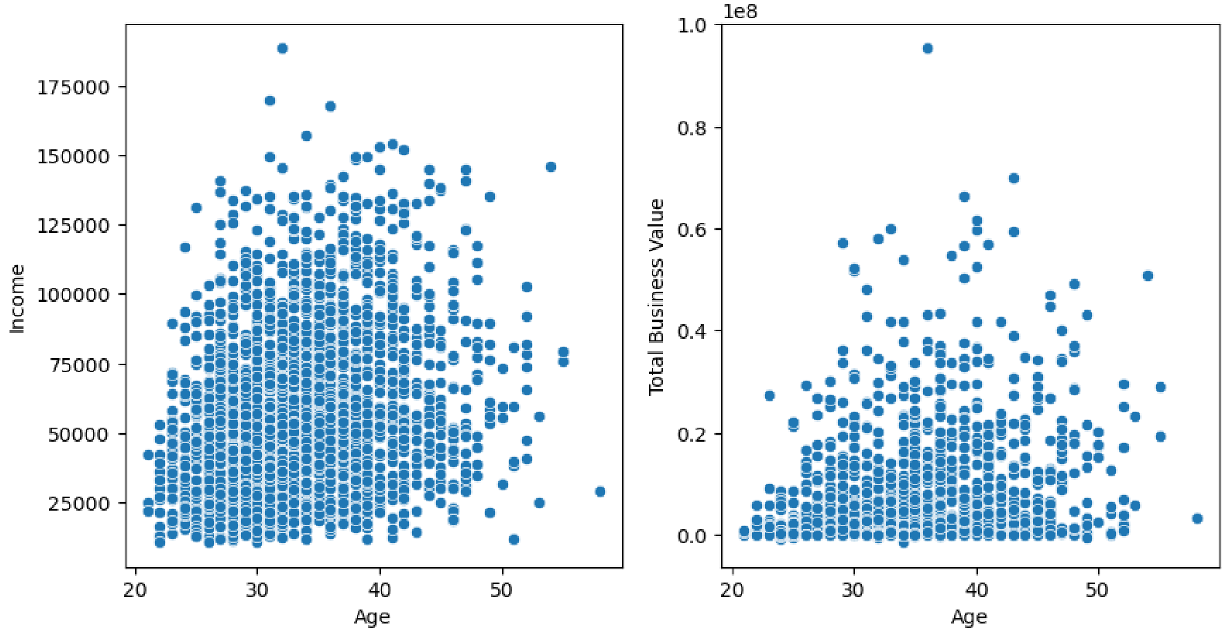
In [317…

```
fig = plt.figure(figsize=(22,5))
ax = fig.add_subplot(1,3,1)
grouped_months = data.groupby(['month'])['rides_count'].count().reset_index()
sns.barplot(data=grouped_months,x='month',y='rides_count')
plt.title('rides_count of Driver every month',fontsize=15)

ax = fig.add_subplot(1,3,2)
grouped_years = data.groupby(['year'])['rides_count'].count().reset_index()
sns.barplot(x='year', y='rides_count', data=grouped_years)
plt.title('rides_count of Driver every year' ,fontsize=15)

ax = fig.add_subplot(1,3,3)
grouped_gender = data.groupby('Gender')['rides_count'].sum().reset_index()
grouped_gender['rides_count'] =(grouped_gender['rides_count']/sum(data.rides_count)*10
sns.barplot(x=grouped_gender['Gender'],y= grouped_gender['rides_count'])
plt.title('rides_count of Driver by Gender in percentage', fontsize=15)
plt.show()
```



In [308…

```
fig,axs=plt.subplots(nrows=1,ncols=2,figsize=(10,5))

cols=["Income","Total Business Value"]
count=0

for i in range(1):
    for j in range(2):
            sns.scatterplot(data=data,x="Age",y=cols[count],ax=axs[count])
            count +=1

plt.show()
```



In [309…

```
data.head()
```

Out[309]:

	Driver_ID	rides_count	Age	Gender	City	Education_Level	Income	Joining Designation	Grade	Total Business Value
0	1	3	28.0	0.0	C23	2	57387	1	1	1715580
1	2	2	31.0	0.0	C7	2	67016	2	2	0
2	4	5	43.0	0.0	C13	2	65603	2	2	350000
3	5	3	29.0	0.0	C9	0	46368	1	1	120360
4	6	5	31.0	1.0	C11	1	78728	3	3	1265000

In [333…

```
data.Gender.value_counts()
```

Out[333]:

```
Gender
0.0    1404
1.0     977
Name: count, dtype: int64
```

In [338…

```
data[data["Target"] == 0]["Gender"].value_counts() / (data.Gender.value_counts()) * 10
```

Out[338]:

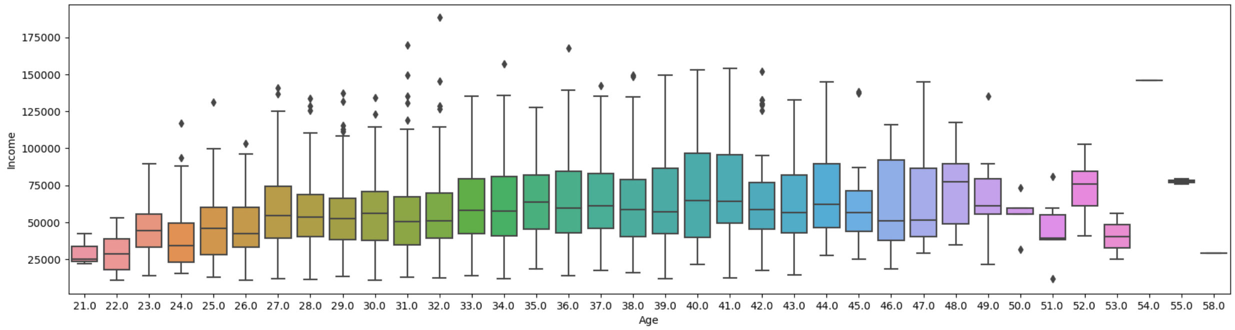
```
Gender
0.0    67.521368
1.0    68.372569
Name: count, dtype: float64
```

In [310…

```
fig = plt.subplots(figsize=(20,5))
sns.boxplot(data=data,x="Age",y = "Income")
```

Out[310]:

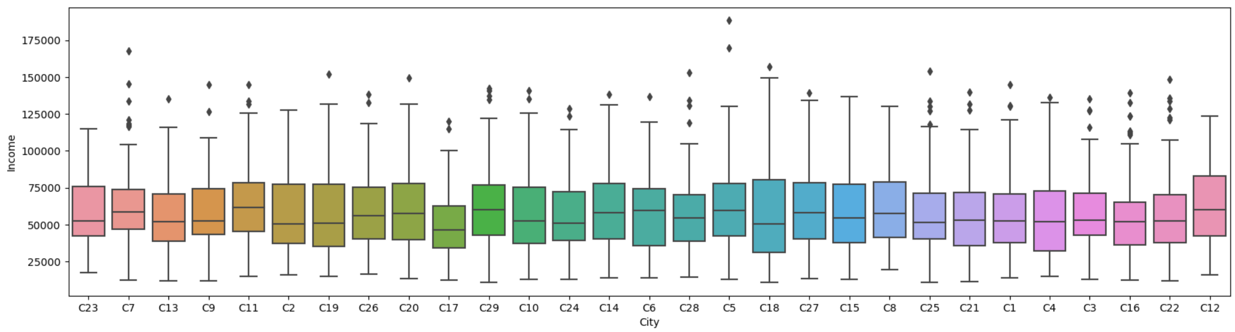
```
<Axes: xlabel='Age', ylabel='Income'>
```



In [154…

```
fig = plt.subplots(figsize=(20,5))
sns.boxplot(data=data,x="City",y = "Income")
```

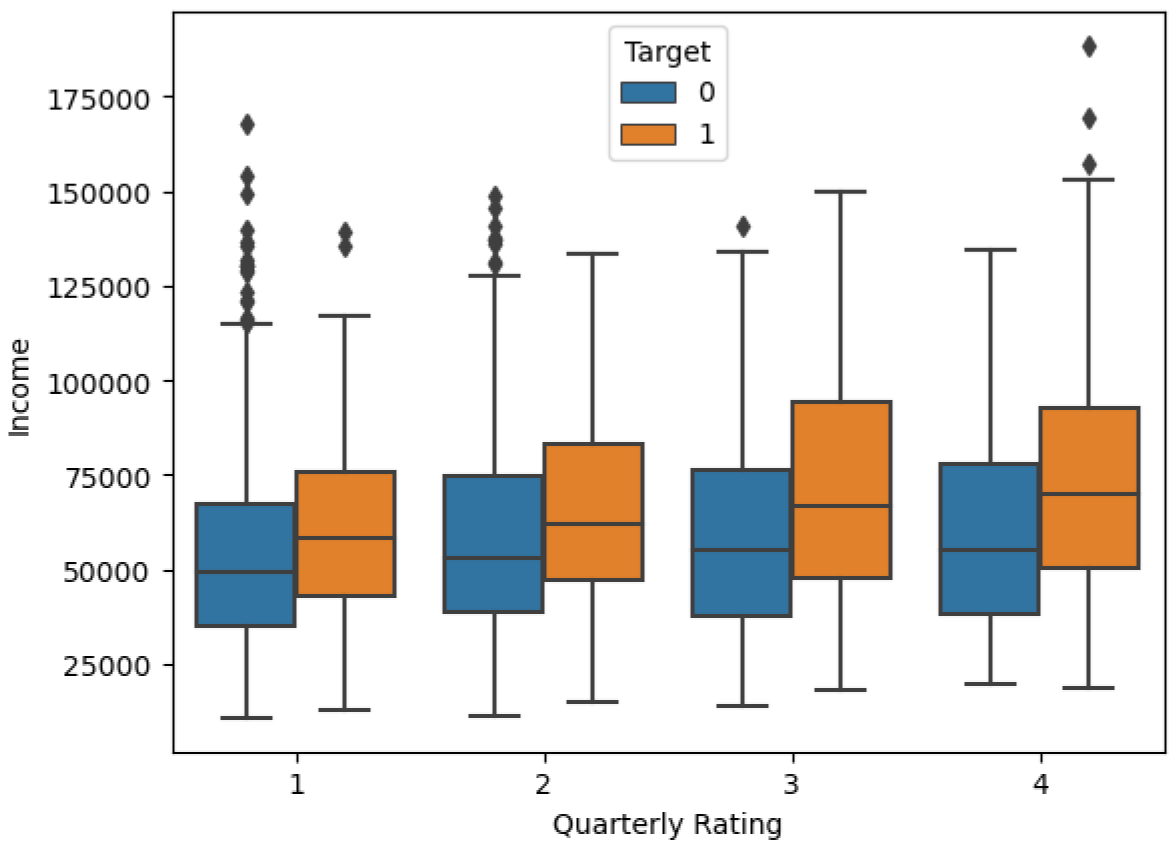
Out[154]: <Axes: xlabel='City', ylabel='Income'>



In [329…

```
sns.boxplot(data=data,x="Quarterly Rating",y = "Income",hue="Target")
```

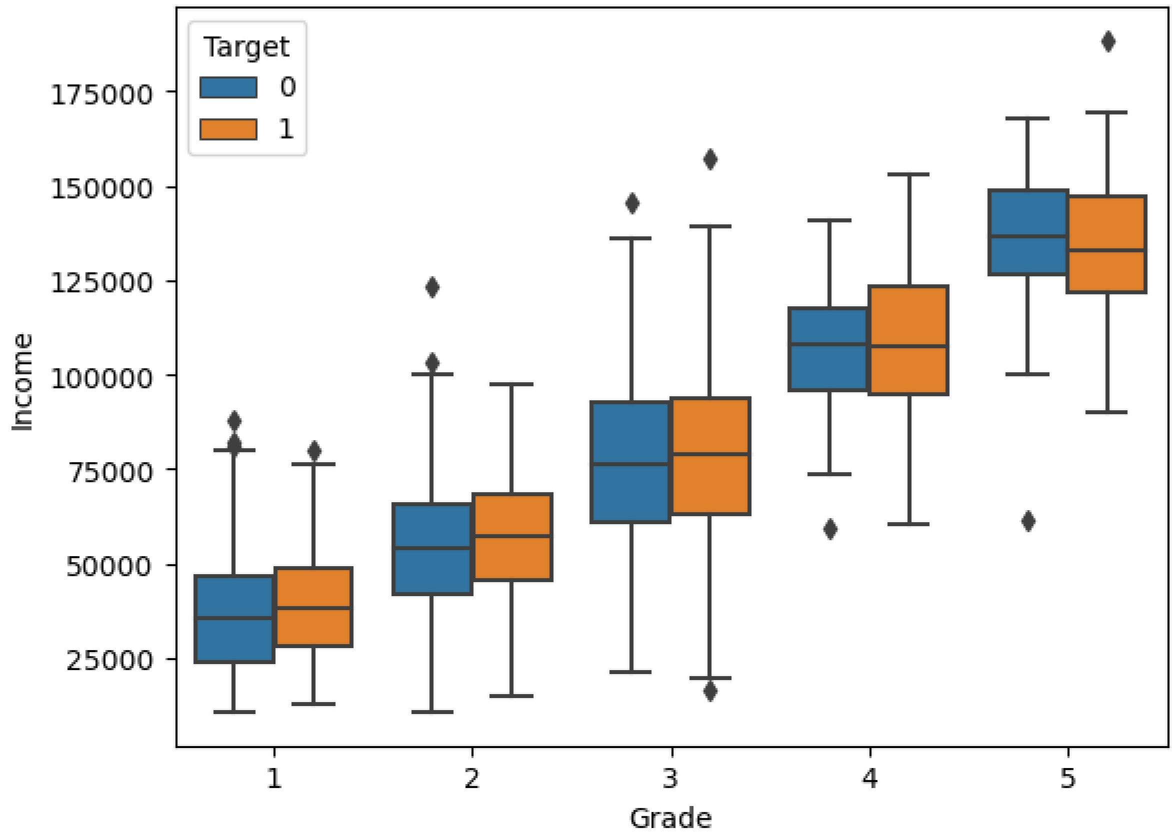
Out[329]: <Axes: xlabel='Quarterly Rating', ylabel='Income'>



In [327…

```
sns.boxplot(data=data,x="Grade",y = "Income",hue="Target")
```

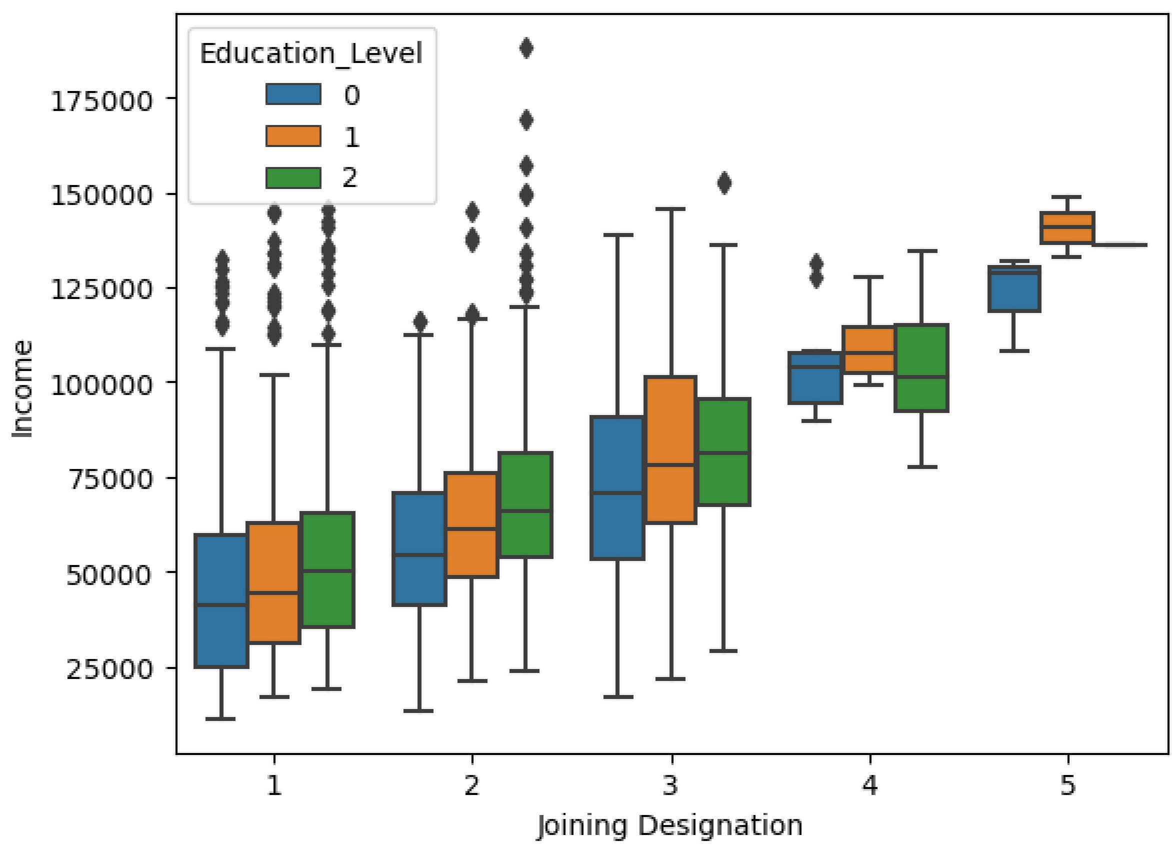
Out[327]: <Axes: xlabel='Grade', ylabel='Income'>



In [127…

```
sns.boxplot(data=data,x="Joining Designation",y = "Income",hue="Education_Level")
```

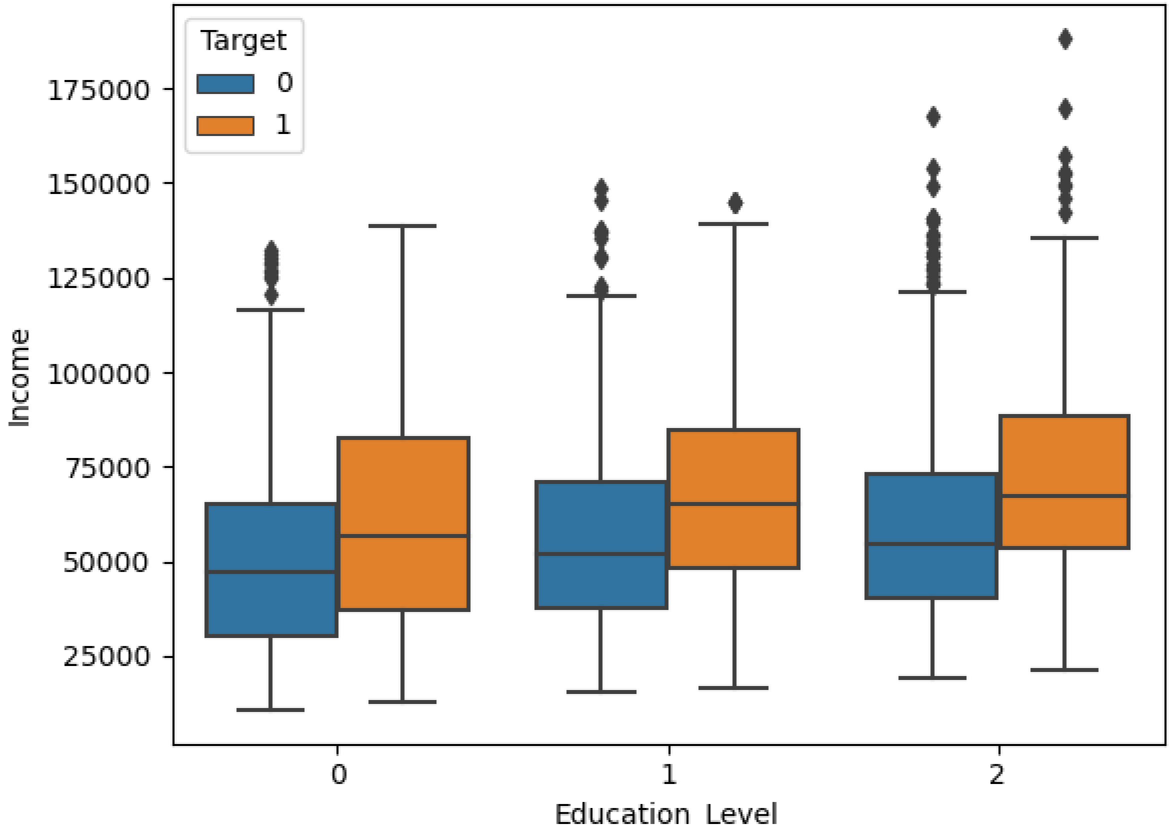
Out[127]: <Axes: xlabel='Joining Designation', ylabel='Income'>



In [328…

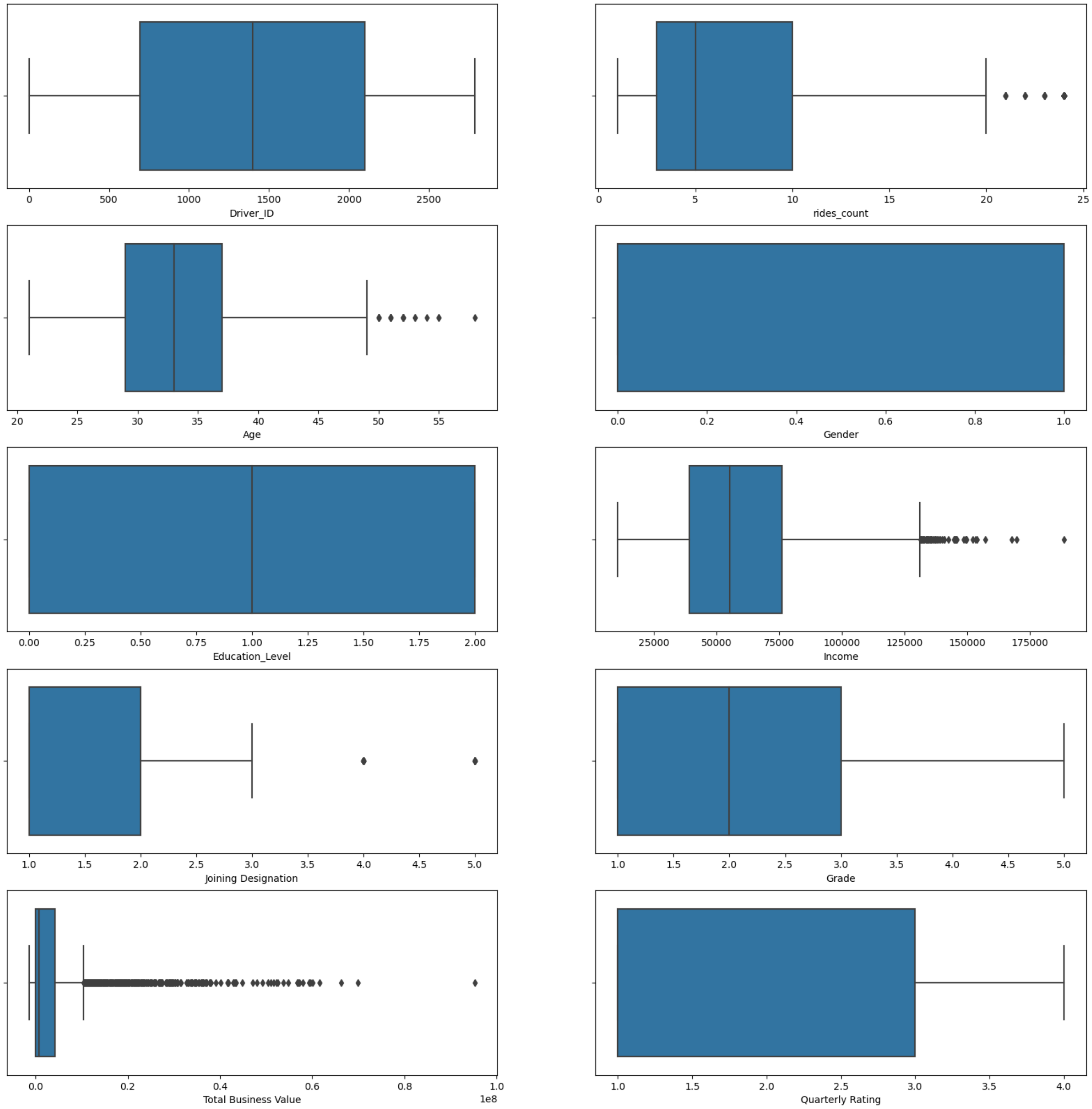
```
sns.boxplot(data=data,x="Education_Level",y = "Income",hue="Target")
```

Out[328]: <Axes: xlabel='Education_Level', ylabel='Income'>



In [323…

```
fig,axs=plt.subplots(nrows=5,ncols=2,figsize=(20,20))

df_num = data.select_dtypes(exclude = "object")
cols = df_num.columns
count=0

for i in range(5):
    for j in range(2):
            sns.boxplot(data=df_num,x=cols[count],ax=axs[i,j])
            count +=1

plt.show()
```



In [332...
```python
data.Raise.value_counts()/data.shape[0] * 100
```

Out[332]:
```
Raise
0    98.194036
1     1.805964
Name: count, dtype: float64
```

In [322...
```python
 data.Gender.value_counts()/data.shape[0]
```

Out[322]:
```
Gender
0.0    58.966821
1.0    41.033179
Name: count, dtype: float64
```

In [340...
```python
data.Promotion.value_counts()/data.shape[0] * 100
```

Out[340]:
```
Promotion
0    65.728685
1    34.271315
Name: count, dtype: float64
```

So we see that there are 59% male employees and 41% female employees. 98.1% of the employees who did not get a raise which is a huge redflag for drivers to churn. Looking at the factors that determine thier Income raise is Quaterly ratings. 68 % females and 6% males who signed up left ola and majority of them are aged between 27-35. Majority of employees joined at lowest designation (1). Promotions highly depends on quaterely ratings and total business value of the drivers only 34% received promotions while rest didnt maybe due to poor ratings. Education level doesnt depend on the income. Joining designation plays major role on high income ranges. The majority of the employees seem to be associated with city C20. Scatter plot of Income shows that Income increases with increase in age but after 45-50, we see a subtle decline. Scatter plot of Total Business Value shows an increase with increase in Age yet we notice a decline after 45. Income decreses with increase in Destination as about 4% of the employees hold higher designations. The median of the Income for employees having higher Grades is greater. Distribution of Income for enployes at different Education level is about a change of 3-5% with level 0. Joining Designation Increases with increase in Grade. About 55% of the ridecounts of the employees has got Quarlerly Rating 1. Number of ridecounts increases with increase in Income as well as Total Business Value.

Outlier Treatment

In [342…
```
data.describe().T
```

Out[342]:

	count	mean	std	min	25%	50%	75%	
Driver_ID	2381.0	1.397559e+03	8.061616e+02	1.0	695.0	1400.0	2100.0	27
rides_count	2381.0	8.023520e+00	6.783590e+00	1.0	3.0	5.0	10.0	
Age	2381.0	3.366317e+01	5.983375e+00	21.0	29.0	33.0	37.0	
Gender	2381.0	4.103318e-01	4.919972e-01	0.0	0.0	0.0	1.0	
Education_Level	2381.0	1.007560e+00	8.162900e-01	0.0	0.0	1.0	2.0	
Income	2381.0	5.933416e+04	2.838367e+04	10747.0	39104.0	55315.0	75986.0	1884
Joining Designation	2381.0	1.820244e+00	8.414334e-01	1.0	1.0	2.0	2.0	
Grade	2381.0	2.096598e+00	9.415218e-01	1.0	1.0	2.0	3.0	
Total Business Value	2381.0	4.586742e+06	9.127115e+06	-1385530.0	0.0	817680.0	4173650.0	953310
Quarterly Rating	2381.0	1.929861e+00	1.104857e+00	1.0	1.0	1.0	3.0	
month	2381.0	7.357413e+00	3.143143e+00	1.0	5.0	7.0	10.0	
year	2381.0	2.018536e+03	1.609597e+00	2013.0	2018.0	2019.0	2020.0	20
Promotion	2381.0	3.427131e-01	4.747162e-01	0.0	0.0	0.0	1.0	
Target	2381.0	3.212936e-01	4.670713e-01	0.0	0.0	0.0	1.0	
Raise	2381.0	1.805964e-02	1.331951e-01	0.0	0.0	0.0	0.0	

In [ ]:

```
Total Business Value has some negative values prone to outliers.
```

In [51]:

```
data = data[data["Total Business Value"] > 1]
```

In [345…

```
data.describe().T
```

Out[345]:

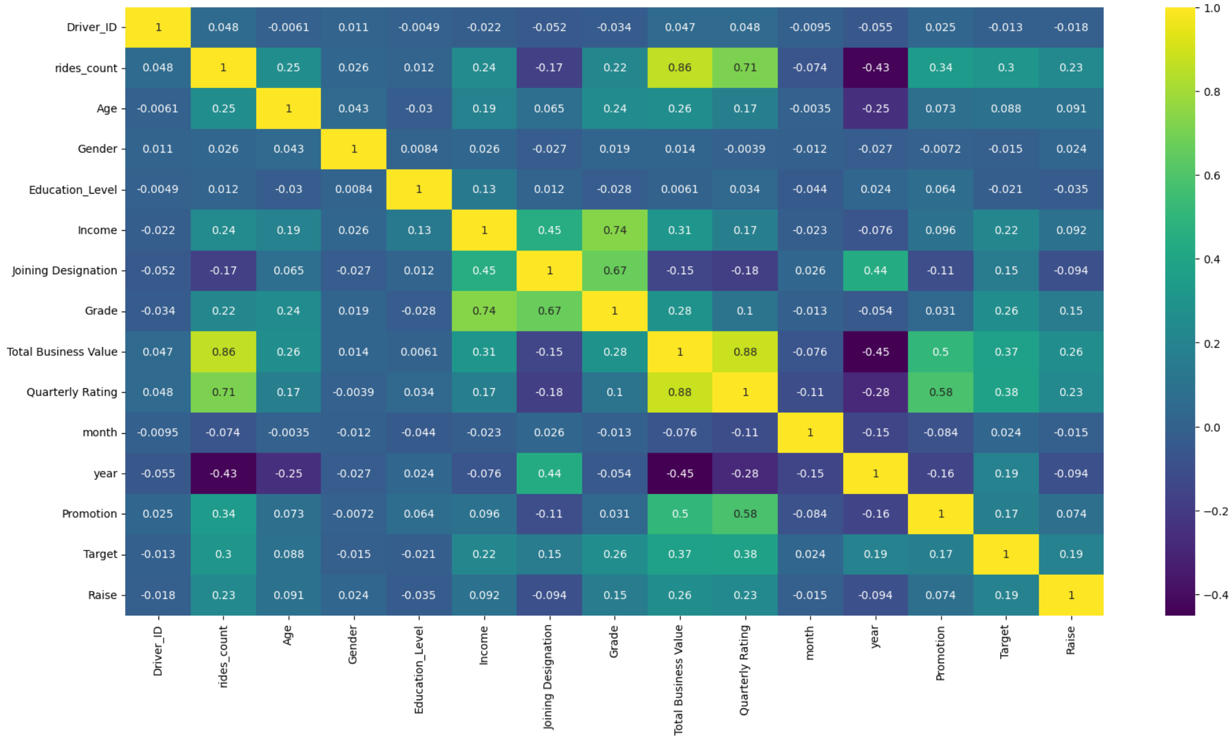
	count	mean	std	min	25%	50%	75%	
Driver_ID	1652.0	1.390315e+03	8.082919e+02	1.0	679.50	1385.0	2097.00	2
rides_count	1652.0	1.026998e+01	6.967589e+00	1.0	5.00	8.0	14.00	
Age	1652.0	3.432385e+01	6.190776e+00	21.0	30.00	34.0	38.00	
Gender	1652.0	4.158596e-01	4.930188e-01	0.0	0.00	0.0	1.00	
Education_Level	1652.0	1.030872e+00	8.093284e-01	0.0	0.00	1.0	2.00	
Income	1652.0	6.174704e+04	2.929270e+04	11068.0	40101.25	57320.5	78768.75	188
Joining Designation	1652.0	1.759685e+00	8.395129e-01	1.0	1.00	2.0	2.00	
Grade	1652.0	2.144068e+00	9.719606e-01	1.0	1.00	2.0	3.00	
Total Business Value	1652.0	6.613094e+06	1.032794e+07	19580.0	663022.50	2242080.0	7418392.50	95331
Quarterly Rating	1652.0	2.339588e+00	1.100215e+00	1.0	1.00	2.0	3.00	
month	1652.0	7.136804e+00	3.067293e+00	1.0	5.00	7.0	10.00	
year	1652.0	2.018208e+03	1.730439e+00	2013.0	2018.00	2018.0	2020.00	2
Promotion	1652.0	4.933414e-01	5.001070e-01	0.0	0.00	0.0	1.00	
Target	1652.0	3.619855e-01	4.807202e-01	0.0	0.00	0.0	1.00	
Raise	1652.0	2.602906e-02	1.592699e-01	0.0	0.00	0.0	0.00	

In [348…

```
df_num = data.select_dtypes(exclude='object')
```

In [349…

```
plt.figure(figsize=(20,10))
sns.heatmap(df_num.corr(method="spearman"),annot=True,cmap="viridis")
plt.show()
```



dependency of promotion to Total Business Value 7 Quarterly Rating has reduced 10-20 %.

In [52]:
```
data.City.unique()
```

Out[52]:
```
array(['C23', 'C13', 'C9', 'C11', 'C19', 'C20', 'C29', 'C10', 'C24',
       'C14', 'C28', 'C5', 'C18', 'C26', 'C15', 'C17', 'C8', 'C25', 'C21',
       'C1', 'C6', 'C27', 'C7', 'C3', 'C16', 'C2', 'C22', 'C12', 'C4'],
      dtype=object)
```

KNN imputer treating missing values

In [354…
```
data.isna().sum()
```

Out[354]:
```
Driver_ID               0
rides_count             0
Age                     0
Gender                  0
City                    0
Education_Level         0
Income                  0
Joining Designation     0
Grade                   0
Total Business Value    0
Quarterly Rating        0
month                   0
year                    0
Promotion               0
Target                  0
Raise                   0
dtype: int64
```

In [30]:
```
from sklearn.impute import KNNImputer
imputer = KNNImputer(n_neighbors= 3)
data_new = imputer.fit_transform(data[["Age","Gender"]])
```

In [359…

```
pip install xgboost
```

```
Collecting xgboostNote: you may need to restart the kernel to use updated packages.

  Obtaining dependency information for xgboost from https://files.pythonhosted.org/pa
ckages/24/ec/ad387100fa3cc2b9b81af0829b5ecfe75ec5bb19dd7c19d4fea06fb81802/xgboost-2.
0.3-py3-none-win_amd64.whl.metadata
  Downloading xgboost-2.0.3-py3-none-win_amd64.whl.metadata (2.0 kB)
Requirement already satisfied: numpy in c:\users\91944\anaconda3\lib\site-packages (f
rom xgboost) (1.24.3)
Requirement already satisfied: scipy in c:\users\91944\anaconda3\lib\site-packages (f
rom xgboost) (1.11.1)
Downloading xgboost-2.0.3-py3-none-win_amd64.whl (99.8 MB)
   ---------------------------------------- 0.0/99.8 MB ? eta -:--:--
   ---------------------------------------- 0.0/99.8 MB ? eta -:--:--
   ---------------------------------------- 0.0/99.8 MB ? eta -:--:--
   ---------------------------------------- 0.0/99.8 MB 393.8 kB/s eta 0:04:14
   ---------------------------------------- 0.3/99.8 MB 2.3 MB/s eta 0:00:45
   ---------------------------------------- 0.8/99.8 MB 4.4 MB/s eta 0:00:23
    --------------------------------------- 1.4/99.8 MB 6.5 MB/s eta 0:00:16
    --------------------------------------- 2.2/99.8 MB 8.2 MB/s eta 0:00:12
   - -------------------------------------- 3.0/99.8 MB 10.1 MB/s eta 0:00:10
   - -------------------------------------- 3.8/99.8 MB 10.9 MB/s eta 0:00:09
   - -------------------------------------- 4.8/99.8 MB 12.2 MB/s eta 0:00:08
   -- ------------------------------------- 5.4/99.8 MB 12.4 MB/s eta 0:00:08
   -- ------------------------------------- 6.0/99.8 MB 12.7 MB/s eta 0:00:08
   -- ------------------------------------- 6.5/99.8 MB 13.0 MB/s eta 0:00:08
   -- ------------------------------------- 7.0/99.8 MB 12.8 MB/s eta 0:00:08
   --- ------------------------------------ 7.6/99.8 MB 12.7 MB/s eta 0:00:08
   --- ------------------------------------ 8.2/99.8 MB 12.8 MB/s eta 0:00:08
   --- ------------------------------------ 8.7/99.8 MB 12.7 MB/s eta 0:00:08
   --- ------------------------------------ 9.4/99.8 MB 12.7 MB/s eta 0:00:08
   --- ------------------------------------ 9.9/99.8 MB 12.7 MB/s eta 0:00:08
   ---- ----------------------------------- 10.3/99.8 MB 14.2 MB/s eta 0:00:07
   ---- ----------------------------------- 11.0/99.8 MB 14.9 MB/s eta 0:00:06
   ---- ----------------------------------- 11.5/99.8 MB 14.5 MB/s eta 0:00:07
   ---- ----------------------------------- 12.2/99.8 MB 14.9 MB/s eta 0:00:06
   ----- ---------------------------------- 12.7/99.8 MB 14.6 MB/s eta 0:00:06
   ----- ---------------------------------- 13.2/99.8 MB 13.9 MB/s eta 0:00:07
   ----- ---------------------------------- 13.7/99.8 MB 13.9 MB/s eta 0:00:07
   ----- ---------------------------------- 14.2/99.8 MB 13.6 MB/s eta 0:00:07
   ----- ---------------------------------- 14.8/99.8 MB 13.1 MB/s eta 0:00:07
   ------ --------------------------------- 15.3/99.8 MB 13.4 MB/s eta 0:00:07
   ------ --------------------------------- 15.8/99.8 MB 12.8 MB/s eta 0:00:07
   ------ --------------------------------- 16.4/99.8 MB 13.1 MB/s eta 0:00:07
   ------ --------------------------------- 16.9/99.8 MB 13.1 MB/s eta 0:00:07
   ------ --------------------------------- 17.4/99.8 MB 12.8 MB/s eta 0:00:07
   ------- -------------------------------- 18.0/99.8 MB 13.1 MB/s eta 0:00:07
   ------- -------------------------------- 18.5/99.8 MB 13.1 MB/s eta 0:00:07
   ------- -------------------------------- 19.0/99.8 MB 12.8 MB/s eta 0:00:07
   ------- -------------------------------- 19.5/99.8 MB 12.8 MB/s eta 0:00:07
   -------- ------------------------------- 20.0/99.8 MB 12.8 MB/s eta 0:00:07
   -------- ------------------------------- 20.6/99.8 MB 13.1 MB/s eta 0:00:07
   -------- ------------------------------- 21.0/99.8 MB 12.8 MB/s eta 0:00:07
   -------- ------------------------------- 21.6/99.8 MB 12.8 MB/s eta 0:00:07
   -------- ------------------------------- 22.0/99.8 MB 12.8 MB/s eta 0:00:07
   --------- ------------------------------ 22.6/99.8 MB 12.8 MB/s eta 0:00:07
   --------- ------------------------------ 23.1/99.8 MB 12.8 MB/s eta 0:00:06
   --------- ------------------------------ 23.7/99.8 MB 12.8 MB/s eta 0:00:06
   --------- ------------------------------ 24.2/99.8 MB 12.8 MB/s eta 0:00:06
   --------- ------------------------------ 24.7/99.8 MB 12.6 MB/s eta 0:00:06
   ---------- ----------------------------- 25.3/99.8 MB 12.8 MB/s eta 0:00:06
```

```
---------- ----------------------------- 25.8/99.8 MB 12.8 MB/s eta 0:00:06
---------- ----------------------------- 26.3/99.8 MB 12.6 MB/s eta 0:00:06
---------- ----------------------------- 26.9/99.8 MB 12.8 MB/s eta 0:00:06
---------- ----------------------------- 27.4/99.8 MB 12.6 MB/s eta 0:00:06
----------- ---------------------------- 27.6/99.8 MB 12.6 MB/s eta 0:00:06
----------- ---------------------------- 27.6/99.8 MB 12.6 MB/s eta 0:00:06
----------- ---------------------------- 29.1/99.8 MB 12.6 MB/s eta 0:00:06
----------- ---------------------------- 29.6/99.8 MB 12.8 MB/s eta 0:00:06
------------ --------------------------- 30.2/99.8 MB 12.8 MB/s eta 0:00:06
------------ --------------------------- 30.7/99.8 MB 12.6 MB/s eta 0:00:06
------------ --------------------------- 31.1/99.8 MB 12.6 MB/s eta 0:00:06
------------ --------------------------- 31.7/99.8 MB 12.6 MB/s eta 0:00:06
------------ --------------------------- 32.2/99.8 MB 12.6 MB/s eta 0:00:06
------------- -------------------------- 32.7/99.8 MB 12.6 MB/s eta 0:00:06
------------- -------------------------- 33.2/99.8 MB 12.6 MB/s eta 0:00:06
------------- -------------------------- 33.6/99.8 MB 12.3 MB/s eta 0:00:06
------------- -------------------------- 34.2/99.8 MB 12.6 MB/s eta 0:00:06
------------- -------------------------- 34.5/99.8 MB 12.4 MB/s eta 0:00:06
-------------- ------------------------- 35.2/99.8 MB 12.3 MB/s eta 0:00:06
-------------- ------------------------- 35.7/99.8 MB 12.4 MB/s eta 0:00:06
-------------- ------------------------- 36.2/99.8 MB 12.4 MB/s eta 0:00:06
-------------- ------------------------- 36.7/99.8 MB 12.4 MB/s eta 0:00:06
-------------- ------------------------- 37.3/99.8 MB 12.4 MB/s eta 0:00:06
--------------- ------------------------ 37.9/99.8 MB 13.6 MB/s eta 0:00:05
--------------- ------------------------ 38.4/99.8 MB 13.1 MB/s eta 0:00:05
--------------- ------------------------ 38.9/99.8 MB 12.6 MB/s eta 0:00:05
--------------- ------------------------ 39.5/99.8 MB 12.4 MB/s eta 0:00:05
---------------- ----------------------- 40.1/99.8 MB 12.6 MB/s eta 0:00:05
---------------- ----------------------- 40.6/99.8 MB 12.4 MB/s eta 0:00:05
---------------- ----------------------- 41.2/99.8 MB 12.4 MB/s eta 0:00:05
---------------- ----------------------- 41.7/99.8 MB 12.4 MB/s eta 0:00:05
---------------- ----------------------- 42.2/99.8 MB 12.6 MB/s eta 0:00:05
----------------- ---------------------- 42.8/99.8 MB 12.6 MB/s eta 0:00:05
----------------- ---------------------- 43.3/99.8 MB 12.6 MB/s eta 0:00:05
----------------- ---------------------- 43.8/99.8 MB 12.9 MB/s eta 0:00:05
----------------- ---------------------- 44.2/99.8 MB 12.6 MB/s eta 0:00:05
----------------- ---------------------- 44.7/99.8 MB 12.6 MB/s eta 0:00:05
------------------ --------------------- 45.3/99.8 MB 12.9 MB/s eta 0:00:05
------------------ --------------------- 45.7/99.8 MB 12.4 MB/s eta 0:00:05
------------------ --------------------- 46.3/99.8 MB 12.6 MB/s eta 0:00:05
------------------ --------------------- 46.9/99.8 MB 12.6 MB/s eta 0:00:05
------------------- -------------------- 47.4/99.8 MB 12.6 MB/s eta 0:00:05
------------------- -------------------- 48.1/99.8 MB 12.6 MB/s eta 0:00:05
------------------- -------------------- 48.7/99.8 MB 12.8 MB/s eta 0:00:04
------------------- -------------------- 49.2/99.8 MB 12.6 MB/s eta 0:00:05
------------------- -------------------- 49.8/99.8 MB 12.8 MB/s eta 0:00:04
-------------------- ------------------- 50.3/99.8 MB 12.6 MB/s eta 0:00:04
-------------------- ------------------- 50.7/99.8 MB 12.6 MB/s eta 0:00:04
-------------------- ------------------- 51.4/99.8 MB 12.8 MB/s eta 0:00:04
-------------------- ------------------- 51.9/99.8 MB 12.6 MB/s eta 0:00:04
--------------------- ------------------ 52.5/99.8 MB 12.8 MB/s eta 0:00:04
--------------------- ------------------ 53.0/99.8 MB 12.8 MB/s eta 0:00:04
--------------------- ------------------ 53.5/99.8 MB 12.6 MB/s eta 0:00:04
--------------------- ------------------ 53.9/99.8 MB 12.8 MB/s eta 0:00:04
--------------------- ------------------ 54.7/99.8 MB 12.8 MB/s eta 0:00:04
---------------------- ----------------- 55.2/99.8 MB 12.8 MB/s eta 0:00:04
---------------------- ----------------- 55.7/99.8 MB 13.1 MB/s eta 0:00:04
---------------------- ----------------- 56.2/99.8 MB 13.1 MB/s eta 0:00:04
---------------------- ----------------- 56.7/99.8 MB 12.8 MB/s eta 0:00:04
---------------------- ----------------- 57.3/99.8 MB 13.1 MB/s eta 0:00:04
```

```
----------------------- ---------------- 57.8/99.8 MB 13.1 MB/s eta 0:00:04
----------------------- ---------------- 58.2/99.8 MB 12.8 MB/s eta 0:00:04
----------------------- ---------------- 58.6/99.8 MB 12.4 MB/s eta 0:00:04
----------------------- ---------------- 58.8/99.8 MB 12.1 MB/s eta 0:00:04
----------------------- ---------------- 59.4/99.8 MB 11.9 MB/s eta 0:00:04
------------------------ --------------- 60.0/99.8 MB 12.1 MB/s eta 0:00:04
------------------------ --------------- 60.8/99.8 MB 12.4 MB/s eta 0:00:04
------------------------ --------------- 61.3/99.8 MB 12.6 MB/s eta 0:00:04
------------------------ --------------- 61.8/99.8 MB 12.4 MB/s eta 0:00:04
------------------------ --------------- 62.3/99.8 MB 12.4 MB/s eta 0:00:04
------------------------- -------------- 62.8/99.8 MB 12.4 MB/s eta 0:00:03
------------------------- -------------- 63.3/99.8 MB 12.6 MB/s eta 0:00:03
------------------------- -------------- 63.8/99.8 MB 12.6 MB/s eta 0:00:03
------------------------- -------------- 64.2/99.8 MB 12.4 MB/s eta 0:00:03
------------------------- -------------- 64.4/99.8 MB 11.9 MB/s eta 0:00:03
------------------------- -------------- 64.7/99.8 MB 11.3 MB/s eta 0:00:04
-------------------------- ------------- 65.7/99.8 MB 11.9 MB/s eta 0:00:03
-------------------------- ------------- 66.5/99.8 MB 12.4 MB/s eta 0:00:03
-------------------------- ------------- 67.0/99.8 MB 12.4 MB/s eta 0:00:03
--------------------------- ------------ 67.6/99.8 MB 12.4 MB/s eta 0:00:03
--------------------------- ------------ 68.2/99.8 MB 12.4 MB/s eta 0:00:03
--------------------------- ------------ 68.7/99.8 MB 13.1 MB/s eta 0:00:03
--------------------------- ------------ 69.2/99.8 MB 13.4 MB/s eta 0:00:03
--------------------------- ------------ 69.7/99.8 MB 13.4 MB/s eta 0:00:03
---------------------------- ----------- 70.3/99.8 MB 13.1 MB/s eta 0:00:03
---------------------------- ----------- 70.8/99.8 MB 12.8 MB/s eta 0:00:03
---------------------------- ----------- 71.3/99.8 MB 12.8 MB/s eta 0:00:03
---------------------------- ----------- 71.9/99.8 MB 12.8 MB/s eta 0:00:03
----------------------------- ---------- 72.4/99.8 MB 12.8 MB/s eta 0:00:03
----------------------------- ---------- 72.9/99.8 MB 13.1 MB/s eta 0:00:03
----------------------------- ---------- 73.5/99.8 MB 12.8 MB/s eta 0:00:03
----------------------------- ---------- 74.1/99.8 MB 13.1 MB/s eta 0:00:02
----------------------------- ---------- 74.6/99.8 MB 13.6 MB/s eta 0:00:02
------------------------------ --------- 75.2/99.8 MB 13.9 MB/s eta 0:00:02
------------------------------ --------- 75.7/99.8 MB 13.6 MB/s eta 0:00:02
------------------------------ --------- 76.3/99.8 MB 13.1 MB/s eta 0:00:02
------------------------------ --------- 76.8/99.8 MB 12.8 MB/s eta 0:00:02
------------------------------- -------- 77.3/99.8 MB 12.8 MB/s eta 0:00:02
------------------------------- -------- 77.8/99.8 MB 12.8 MB/s eta 0:00:02
------------------------------- -------- 78.5/99.8 MB 12.8 MB/s eta 0:00:02
------------------------------- -------- 79.0/99.8 MB 12.8 MB/s eta 0:00:02
------------------------------- -------- 79.6/99.8 MB 12.8 MB/s eta 0:00:02
-------------------------------- ------- 80.1/99.8 MB 13.1 MB/s eta 0:00:02
-------------------------------- ------- 80.4/99.8 MB 12.6 MB/s eta 0:00:02
-------------------------------- ------- 81.0/99.8 MB 12.6 MB/s eta 0:00:02
-------------------------------- ------- 81.5/99.8 MB 12.6 MB/s eta 0:00:02
-------------------------------- ------- 82.0/99.8 MB 12.6 MB/s eta 0:00:02
--------------------------------- ------ 82.5/99.8 MB 12.6 MB/s eta 0:00:02
--------------------------------- ------ 82.7/99.8 MB 12.4 MB/s eta 0:00:02
--------------------------------- ------ 82.7/99.8 MB 12.4 MB/s eta 0:00:02
--------------------------------- ------ 83.9/99.8 MB 12.4 MB/s eta 0:00:02
--------------------------------- ------ 84.4/99.8 MB 12.4 MB/s eta 0:00:02
---------------------------------- ----- 85.0/99.8 MB 12.6 MB/s eta 0:00:02
---------------------------------- ----- 85.5/99.8 MB 12.4 MB/s eta 0:00:02
---------------------------------- ----- 86.0/99.8 MB 12.4 MB/s eta 0:00:02
---------------------------------- ----- 86.5/99.8 MB 12.4 MB/s eta 0:00:02
---------------------------------- ----- 86.9/99.8 MB 12.1 MB/s eta 0:00:02
----------------------------------- ---- 87.5/99.8 MB 12.4 MB/s eta 0:00:01
----------------------------------- ---- 88.0/99.8 MB 12.4 MB/s eta 0:00:01
----------------------------------- ---- 88.3/99.8 MB 12.1 MB/s eta 0:00:01
```

```
   ----------------------------------- ---- 88.3/99.8 MB 12.1 MB/s eta 0:00:01
   ----------------------------------- ---- 88.5/99.8 MB 10.9 MB/s eta 0:00:02
   ----------------------------------- ---- 89.4/99.8 MB 11.3 MB/s eta 0:00:01
   ------------------------------------ --- 89.9/99.8 MB 11.5 MB/s eta 0:00:01
   ------------------------------------ --- 90.5/99.8 MB 11.5 MB/s eta 0:00:01
   ------------------------------------ --- 90.9/99.8 MB 11.5 MB/s eta 0:00:01
   ------------------------------------ --- 91.6/99.8 MB 11.5 MB/s eta 0:00:01
   ------------------------------------ --- 92.1/99.8 MB 11.7 MB/s eta 0:00:01
   ------------------------------------- -- 92.6/99.8 MB 11.7 MB/s eta 0:00:01
   ------------------------------------- -- 93.1/99.8 MB 12.6 MB/s eta 0:00:01
   ------------------------------------- -- 93.7/99.8 MB 12.1 MB/s eta 0:00:01
   ------------------------------------- -- 94.2/99.8 MB 11.7 MB/s eta 0:00:01
   ------------------------------------- -- 94.7/99.8 MB 11.7 MB/s eta 0:00:01
   -------------------------------------- - 95.1/99.8 MB 11.5 MB/s eta 0:00:01
   -------------------------------------- - 95.7/99.8 MB 11.7 MB/s eta 0:00:01
   -------------------------------------- - 96.3/99.8 MB 11.7 MB/s eta 0:00:01
   -------------------------------------- - 96.9/99.8 MB 11.9 MB/s eta 0:00:01
   ---------------------------------------  97.4/99.8 MB 11.7 MB/s eta 0:00:01
   ---------------------------------------  97.9/99.8 MB 11.7 MB/s eta 0:00:01
   ---------------------------------------  98.5/99.8 MB 11.9 MB/s eta 0:00:01
   ---------------------------------------  99.0/99.8 MB 13.1 MB/s eta 0:00:01
   ---------------------------------------  99.6/99.8 MB 12.8 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------  99.7/99.8 MB 12.6 MB/s eta 0:00:01
   ---------------------------------------- 99.8/99.8 MB 7.7 MB/s eta 0:00:00
Installing collected packages: xgboost
Successfully installed xgboost-2.0.3
```

In [31]:

```python
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_auc_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
```

In [70]:

```python
# OHE

cat_cols = ["Education_Level","Joining Designation","Grade"]

df_encod = pd.get_dummies(data,columns = cat_cols,drop_first = True).astype(int)
```

Train - Test Split

In [53]:
```
data["City"] = data["City"].str.split('C').str[1]
```

In [54]:
```
data.City.unique()
```

Out[54]:
```
array(['23', '13', '9', '11', '19', '20', '29', '10', '24', '14', '28',
       '5', '18', '26', '15', '17', '8', '25', '21', '1', '6', '27', '7',
       '3', '16', '2', '22', '12', '4'], dtype=object)
```

In [73]:
```
x = df_encod.drop("Target",axis = 1)
y = df_encod["Target"]

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = 42
```

In [74]:
```
x_train.head()
```

Out[74]:

	Driver_ID	rides_count	Age	Gender	City	Income	Total Business Value	Quarterly Rating	month	year	...	Edu
1851	2175	5	41	0	29	101223	961180	1	7	2020	...	
25	37	6	33	1	14	57375	650020	1	5	2020	...	
594	694	4	32	1	16	82632	999000	2	7	2019	...	
1583	1860	9	35	0	14	37704	234840	1	12	2019	...	
406	476	10	28	1	16	22918	1220340	2	3	2020	...	

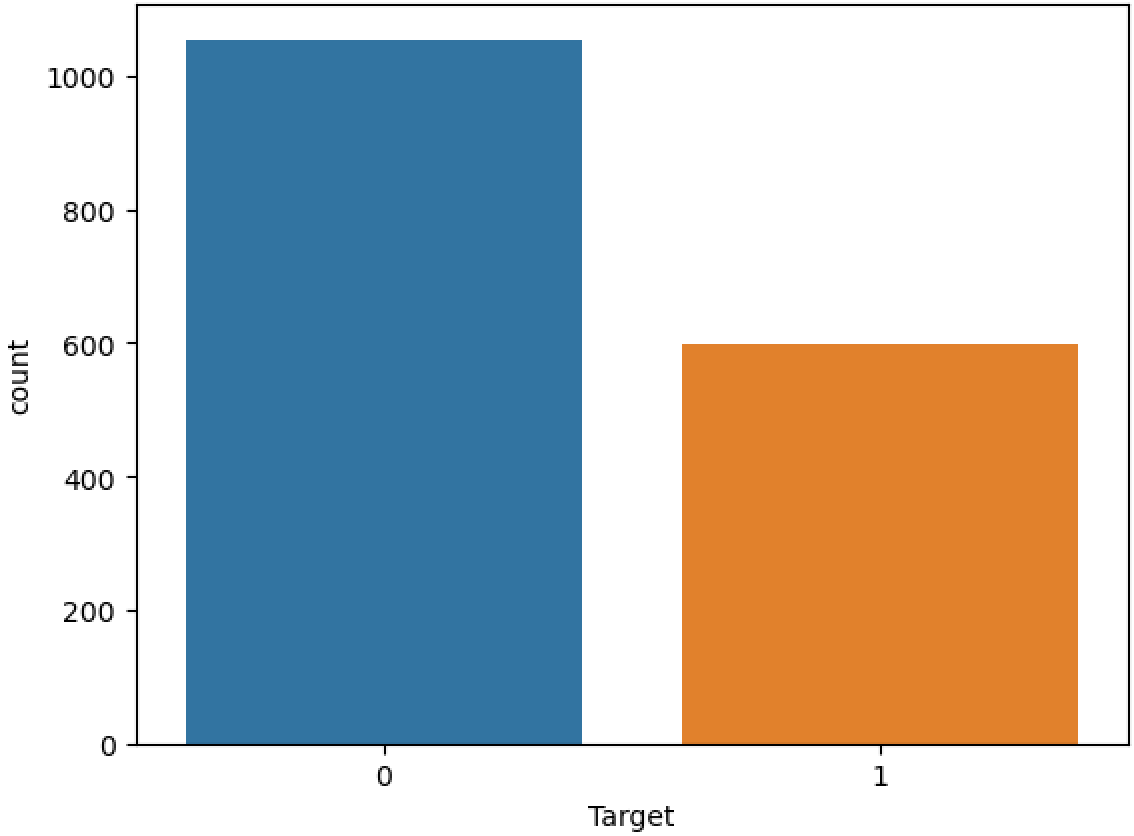
5 rows × 22 columns

In [75]:
```
scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test =  scaler.transform(x_test)
```

Class Imbalance Treatment

In [76]:
```
sns.countplot(data=data,x=data["Target"])
```

Out[76]:
```
<Axes: xlabel='Target', ylabel='count'>
```



SMOTE ANALYSIS

In [77]:
```
from imblearn.over_sampling import SMOTE

smot = SMOTE(random_state=42)
x_train1,y_train1 = smot.fit_resample(x_train,y_train)
```

In [78]:
```
x_train1.shape,y_train1.shape
```

Out[78]:
```
((1504, 22), (1504,))
```

Hyperparameter Tuning

In [84]:
```
param_grid = {
    'n_estimators':list(range(10,20)),
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
}
```

In [86]:
```
clf = RandomForestClassifier()
clf = GridSearchCV(clf,param_grid,cv=10,scoring='recall')
clf.fit(x_train1,y_train1)
```

Out[86]:

```
▸ GridSearchCV
▸ estimator: RandomForestClassifier
  ▸ RandomForestClassifier
```

In [87]:

```
clf.best_params_
```

Out[87]:

```
{'criterion': 'entropy',
 'max_depth': 8,
 'max_features': 'sqrt',
 'n_estimators': 18}
```

RandomForestClassifier

In [102...

```
clf = RandomForestClassifier(max_depth=8,n_estimators= 18)
clf.fit(x_train1,y_train1)
```

Out[102]:

```
▾ RandomForestClassifier
RandomForestClassifier(max_depth=8, n_estimators=18)
```

In [103...

```
y_pred = clf.predict(x_test)
```

In [104...

```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.91      0.88      0.90       302
           1       0.82      0.87      0.85       194

    accuracy                           0.88       496
   macro avg       0.87      0.88      0.87       496
weighted avg       0.88      0.88      0.88       496
```

In [91]:

```
print(confusion_matrix(y_test,y_pred))
```

```
[[265  37]
 [ 22 172]]
```

In [92]:

```
clf.feature_importances_
```

Out[92]:

```
array([2.74835661e-02, 1.68837473e-01, 3.29875694e-02, 7.06831985e-03,
       2.66248571e-02, 3.26731507e-02, 1.51064211e-01, 1.13869024e-01,
       6.23882582e-02, 2.86456086e-01, 2.79714031e-02, 1.00975698e-02,
       4.88153603e-03, 2.66362084e-03, 4.21872552e-03, 1.85060691e-02,
       2.52126523e-04, 2.58705172e-04, 3.19082921e-03, 1.60643131e-02,
       1.27341282e-03, 1.16917340e-03])
```

In [105...

```
from sklearn.metrics import roc_curve, roc_auc_score
```

In [108...

```
probability = clf.predict_proba(x_test)
```

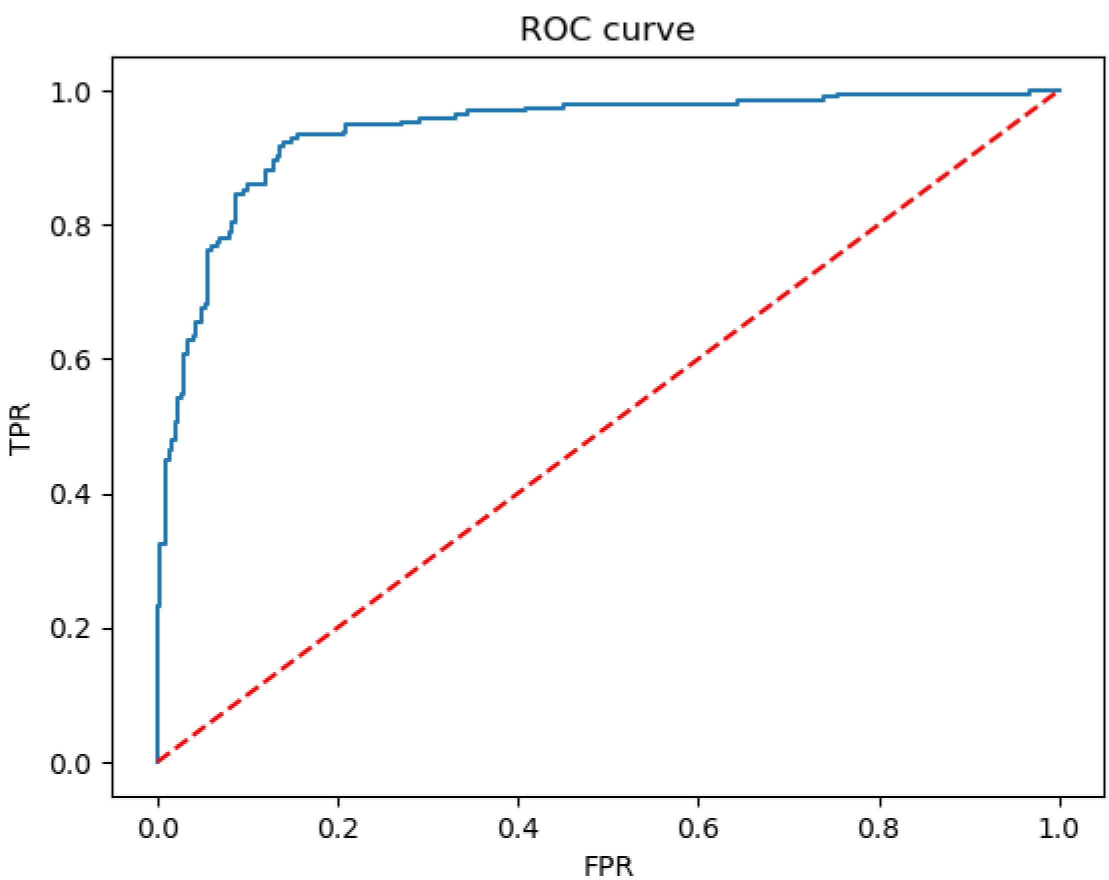
In [107…
```
probabilites = probability[:,1]
```

In [109…
```
fpr, tpr, thr = roc_curve(y_test,probabilites)
```

In [121…
```
roc_auc_score(y_test,probabilites)
```

Out[121]:
```
0.9392537721035025
```

In [110…
```
plt.plot(fpr,tpr)

plt.plot(fpr,fpr,'--',color='red' )
plt.title('ROC curve')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
```



ROC curve shows that the model with 88% accuracy is performing well as it is better than mean model/average model , best model must have high True positive rate and low False positive rate and TPR has peeked at 0.94.

au_roc_score =0.93 > 0.5 , so the model is able to classify more true positve and true negative than false positive and false negative.

XG Boosting Classifier

In [114…
```
xgb = GradientBoostingClassifier()
xgb.fit(x_train, y_train)
y_pred = xgb.predict(x_test)
```

In [115…
```
proba =xgb.predict_proba(x_test)[:, 1]
```

In [116…
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.94      0.93      0.93       302
           1       0.89      0.91      0.90       194

    accuracy                           0.92       496
   macro avg       0.92      0.92      0.92       496
weighted avg       0.92      0.92      0.92       496
```

In [117…
```
print(confusion_matrix(y_test,y_pred))
```

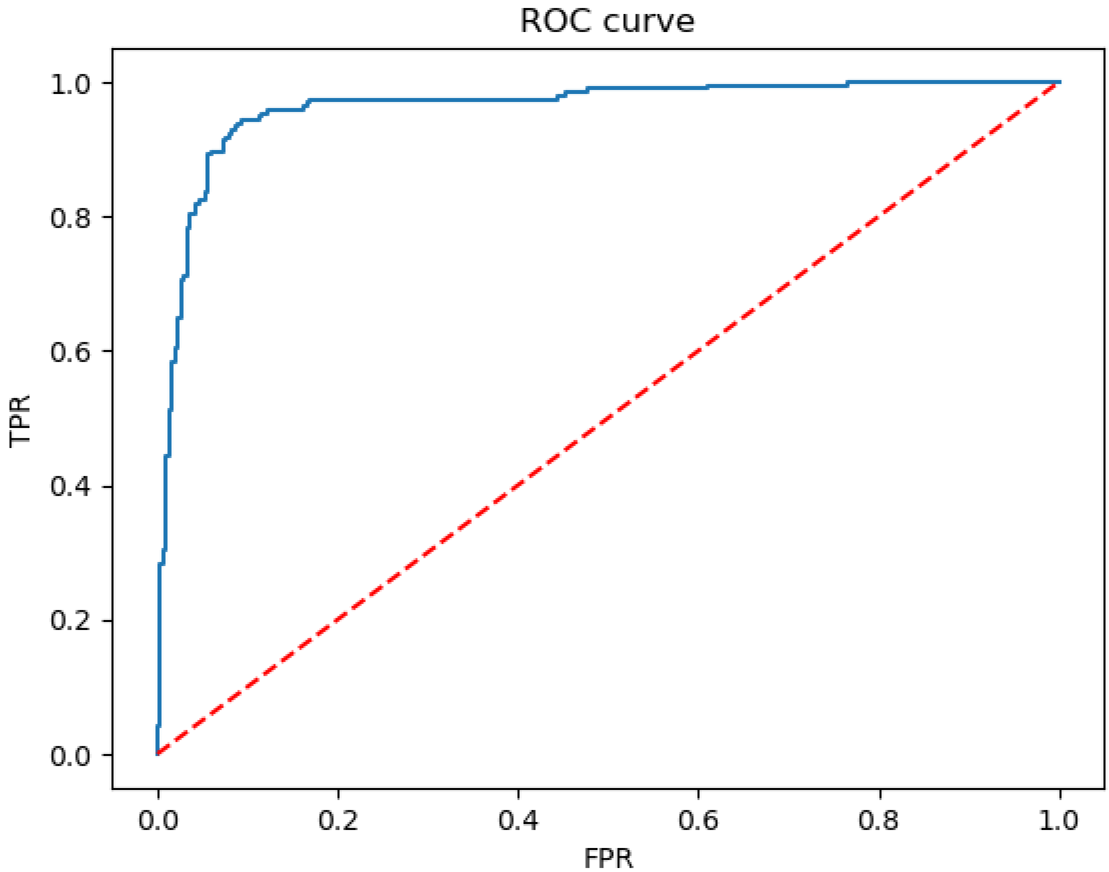
```
[[280  22]
 [ 17 177]]
```

In [118…
```
fpr, tpr, thr = roc_curve(y_test,proba)
```

In [122…
```
roc_auc_score(y_test,proba)
```

Out[122]:
```
0.9620912132177237
```

In [120…
```
plt.plot(fpr,tpr)

plt.plot(fpr,fpr,'--',color='red' )
plt.title('ROC curve')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.show()
```



ROC curve shows that the model with 92% accuracy is performing well as it is better than mean model/average model , best model must have high True positive rate and low False positive rate and TPR has peeked at 0.99.

au_roc_score =0.96 > 0.5 , so the model is able to classify more true positve and true negative than false positive and false negative.

In [ ]:

In [ ]:

In [ ]: