```
import numpy as np
import pandas as pd
import seaborn as sns
from scipy import stats
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve
from sklearn.metrics import precision_recall_curve
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.preprocessing import MinMaxScaler
from datetime import datetime
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
!wget "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/002/492/original/ola_driver_scaler.csv"
```

```
--2024-03-30 06:23:17--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/002/492/original/ola_driver_scaler.csv
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 108.157.172.176, 108.157.172.173, 108.157.172.183, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|108.157.172.176|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1127673 (1.1M) [text/plain]
Saving to: ‘ola_driver_scaler.csv’

ola_driver_scaler.c 100%[===================>]   1.08M  --.-KB/s    in 0.1s

2024-03-30 06:23:18 (9.03 MB/s) – ‘ola_driver_scaler.csv’ saved [1127673/1127673]
```

```
df = pd.read_csv('ola_driver_scaler.csv')
```

```
df.ndim
```

```
2
```

```
df.shape
```

```
(19104, 14)
```

```
df.columns
```

```
Index(['Unnamed: 0', 'MMM-YY', 'Driver_ID', 'Age', 'Gender', 'City',
       'Education_Level', 'Income', 'Dateofjoining', 'LastWorkingDate',
       'Joining Designation', 'Grade', 'Total Business Value',
       'Quarterly Rating'],
      dtype='object')
```

```
df.head()
```

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

Next steps: Generate code with df   View recommended plots

```
df.describe()
```

	Unnamed: 0	Driver_ID	Age	Gender	Education_Level	Income
count	19104.000000	19104.000000	19043.000000	19052.000000	19104.000000	19104.000000
mean	9551.500000	1415.591133	34.668435	0.418749	1.021671	65652.025126
std	5514.994107	810.705321	6.257912	0.493367	0.800167	30914.515344
min	0.000000	1.000000	21.000000	0.000000	0.000000	10747.000000
25%	4775.750000	710.000000	30.000000	0.000000	0.000000	42383.000000
50%	9551.500000	1417.000000	34.000000	0.000000	1.000000	60087.000000
75%	14327.250000	2137.000000	39.000000	1.000000	2.000000	83969.000000
max	19103.000000	2788.000000	58.000000	1.000000	2.000000	188418.000000

```
df.describe(include='object')
```

	MMM-YY	City	Dateofjoining	LastWorkingDate
count	19104	19104	19104	1616
unique	24	29	869	493
top	01/01/19	C20	23/07/15	29/07/20
freq	1022	1008	192	70

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19104 entries, 0 to 19103
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Unnamed: 0            19104 non-null  int64  
 1   MMM-YY                19104 non-null  object 
 2   Driver_ID             19104 non-null  int64  
 3   Age                   19043 non-null  float64
 4   Gender                19052 non-null  float64
 5   City                  19104 non-null  object 
 6   Education_Level       19104 non-null  int64  
 7   Income                19104 non-null  int64  
 8   Dateofjoining         19104 non-null  object 
 9   LastWorkingDate       1616 non-null   object 
 10  Joining Designation   19104 non-null  int64  
 11  Grade                 19104 non-null  int64  
 12  Total Business Value  19104 non-null  int64  
 13  Quarterly Rating      19104 non-null  int64  
dtypes: float64(2), int64(8), object(4)
memory usage: 2.0+ MB
```

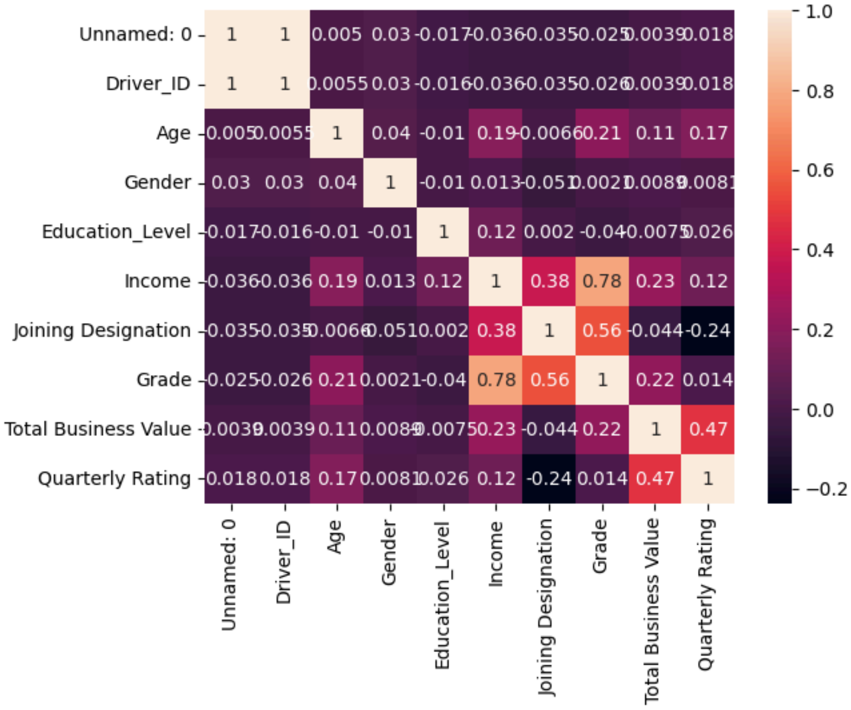
```
df.corr()
```

```
<ipython-input-11-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in
  df.corr()
```

	Unnamed: 0	Driver_ID	Age	Gender	Education_Level	Income	Jo Design
Unnamed: 0	1.000000	0.999887	0.005041	0.030030	-0.016548	-0.035774	-0.0
Driver_ID	0.999887	1.000000	0.005457	0.030349	-0.016132	-0.035767	-0.0
Age	0.005041	0.005457	1.000000	0.040261	-0.010245	0.191112	-0.0
Gender	0.030030	0.030349	0.040261	1.000000	-0.010123	0.013229	-0.0
Education_Level	-0.016548	-0.016132	-0.010245	-0.010123	1.000000	0.115008	0.0
Income	-0.035774	-0.035767	0.191112	0.013229	0.115008	1.000000	0.3
Joining Designation	-0.034988	-0.035166	-0.006641	-0.050878	0.002041	0.380878	1.0
Grade	-0.025225	-0.025712	0.210702	0.002076	-0.039552	0.778383	0.5
Total Business Value	0.003920	0.003896	0.108835	0.008909	-0.007504	0.234044	-0.0
Quarterly Rating	0.017946	0.017917	0.171818	0.008099	0.026064	0.116897	-0.2

```
sns.heatmap(df.corr(),annot = True)
# unnamed and Driver_ID are highly correlated infact same...so we can remove one of them
# Dropping the unnamed column
```

```
<ipython-input-12-7e796203328a>:1: FutureWarning: The default value of numeric_only in
  sns.heatmap(df.corr(),annot = True)
<Axes: >
```



```
df.drop(columns = 'Unnamed: 0',axis = 1,inplace = True)
# The column name is Unnamed: 0 not Unnamed
```

```
df.nunique()
```

```
MMM-YY                     24
Driver_ID                2381
Age                        36
Gender                      2
City                       29
Education_Level             3
Income                   2383
Dateofjoining             869
LastWorkingDate           493
Joining Designation         5
Grade                       5
Total Business Value    10181
Quarterly Rating            4
dtype: int64
```

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

```
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
```

```
df.head()
```

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

Next steps: Generate code with df   View recommended plots

```
df1 = df.copy(deep=True)
```

```
# # 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

first =  (df1.groupby('Driver_ID').agg({'LastWorkingDate':'last'})['LastWorkingDate'].isna()).reset_index()
first['LastWorkingDate'].replace({True:1,False:0},inplace=True)
first.rename(columns={'LastWorkingDate':'target'},inplace=True)
first.head()
```

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

Next steps: Generate code with first View recommended plots

```
# 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
QR1 =  (df1.groupby('Driver_ID').agg({'Quarterly Rating':'first'})['Quarterly Rating']).reset_index()
QR2 = (df1.groupby('Driver_ID').agg({'Quarterly Rating':'last'})['Quarterly Rating']).reset_index()
```

```
QR1.shape,QR2.shape
```

```
((2381, 2), (2381, 2))
```

```
QR1.isna().sum(),QR2.isna().sum()
```

```
(Driver_ID           0
 Quarterly Rating    0
 dtype: int64,
 Driver_ID           0
 Quarterly Rating    0
 dtype: int64)
```

```
first = first.merge(QR1,on='Driver_ID')
first = first.merge(QR2,on='Driver_ID')
```

```
first.head()
```

	Driver_ID	target	Quarterly Rating_x	Quarterly Rating_y
0	1	0	2	2
1	2	1	1	1
2	4	0	1	1
3	5	0	1	1
4	6	1	1	2

Next steps: Generate code with first View recommended plots

```
first['Promotion'] = np.where(first['Quarterly Rating_x'] == first['Quarterly Rating_y'], 0,1)
```

```
# 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
incm1 =  (df1.groupby('Driver_ID').agg({'Income':'first'})['Income']).reset_index()
incm2 = (df1.groupby('Driver_ID').agg({'Income':'last'})['Income']).reset_index()
```

```
incm1.shape,incm2.shape
```

```
((2381, 2), (2381, 2))
```

```
incm1.isna().sum(),incm2.isna().sum()
```

```
(Driver_ID    0
 Income       0
 dtype: int64,
 Driver_ID    0
 Income       0
 dtype: int64)
```

```
first = first.merge(incm1,on='Driver_ID')
first = first.merge(incm2,on='Driver_ID')
```

```
first.head()
```

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

Next steps: Generate code with `first` View recommended plots

```
first['Raise'] = np.where(first['Income_x'] == first['Income_y'], 0,1)
```

```
first = first[['Driver_ID','target','Raise','Promotion']]
```

```
first.head()
```

	Driver_ID	target	Raise	Promotion
0	1	0	0	0
1	2	1	0	0
2	4	0	0	0
3	5	0	0	0
4	6	1	0	1

Next steps: Generate code with `first` View recommended plots

```
functions = {'MMM-YY':'count',
             'Driver_ID':'first',
             'Age':'max',
             'Gender':'last',
            'City':'last',
             'Education_Level':'last',
             'Dateofjoining':'first',
            'LastWorkingDate':'last',
             'Grade':'last',
             'Total Business Value':'sum',
            'Income':'sum',
             'Dateofjoining':'first',
             'LastWorkingDate':'last',
            'Joining Designation':'last',
             'Grade':'last',
             'Quarterly Rating':'first'}
df1 = df1.groupby([df1['Driver_ID']]).aggregate(functions)
df1['month'] = pd.to_datetime(df['Dateofjoining']).dt.month
df1['year'] = pd.DatetimeIndex(df1['Dateofjoining']).year
df1.rename(columns={'MMM-YY':'Reportings'},inplace=True)
```

```
df1.reset_index(drop=True, inplace=True)
df1 = df1.merge(first,on='Driver_ID')
df1.head()
```

	Reportings	Driver_ID	Age	Gender	City	Education_Level	Dateofjoining	LastWorking
0	3	1	28.0	0.0	C23	2	24/12/18	03/
1	2	2	31.0	0.0	C7	2	11/06/20	
2	5	4	43.0	0.0	C13	2	12/07/19	27/
3	3	5	29.0	0.0	C9	0	01/09/19	03/
4	5	6	31.0	1.0	C11	1	31/07/20	

Next steps: Generate code with `df1` View recommended plots

```
import regex
df1['Age'] = df1['Age'].astype('int64')
df1['Cities'] =df1['City'].astype('str').str.extractall('(\d+)').unstack().fillna('').sum(axis=1).astype(int)
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2381 entries, 0 to 2380
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Reportings            2381 non-null   int64
 1   Driver_ID             2381 non-null   int64
 2   Age                   2381 non-null   int64
 3   Gender                2381 non-null   float64
 4   City                  2381 non-null   object
 5   Education_Level       2381 non-null   int64
 6   Dateofjoining         2381 non-null   object
 7   LastWorkingDate       1616 non-null   object
 8   Grade                 2381 non-null   int64
 9   Total Business Value  2381 non-null   int64
 10  Income                2381 non-null   int64
 11  Joining Designation   2381 non-null   int64
 12  Quarterly Rating      2381 non-null   int64
 13  month                 2381 non-null   int64
 14  year                  2381 non-null   int64
 15  target                2381 non-null   int64
 16  Raise                 2381 non-null   int64
 17  Promotion             2381 non-null   int64
 18  Cities                2381 non-null   int64
dtypes: float64(1), int64(15), object(3)
memory usage: 436.6+ KB
```

```
df1.drop(columns=['Dateofjoining','LastWorkingDate','City'],axis=1,inplace=True)
df1['Gender'].replace({'M':0,'F':1},inplace=True)
df1['Gender'] = df1['Gender'].astype('int64')
```

```
sum(df1.isna().sum())
```

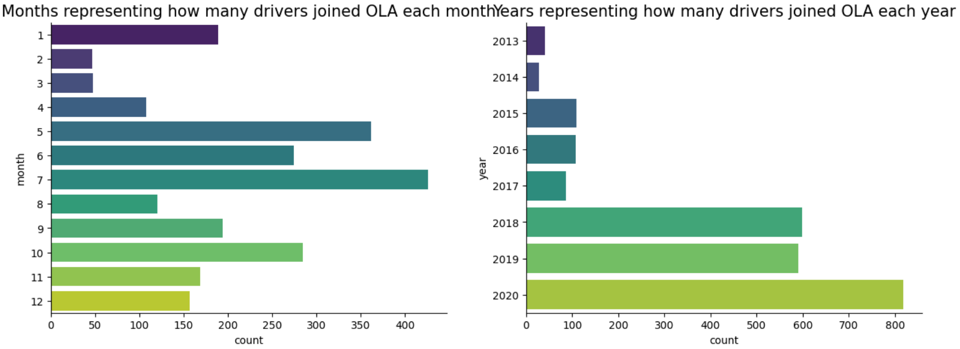
```
0
```

```
df1.describe().T
```

	count	mean	std	min	25%	50%	75%	
Reportings	2381.0	8.023520e+00	6.783590e+00	1.0	3.0	5.0	10.0	
Driver_ID	2381.0	1.397559e+03	8.061616e+02	1.0	695.0	1400.0	2100.0	
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	
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	953
Income	2381.0	5.267603e+05	6.231633e+05	10883.0	139895.0	292980.0	651456.0	45
Joining Designation	2381.0	1.820244e+00	8.414334e-01	1.0	1.0	2.0	2.0	
Quarterly Rating	2381.0	1.486350e+00	8.343483e-01	1.0	1.0	1.0	2.0	
month	2381.0	6.975220e+00	3.007801e+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	
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	
Promotion	2381.0	3.427131e-01	4.747162e-01	0.0	0.0	0.0	1.0	

```
fig = plt.figure(figsize=(15,5))
ax = fig.add_subplot(1,2,1)
sns.countplot(y=df1.month,palette='viridis')
plt.title('Months representing how many drivers joined OLA each month',fontname='Franklin Gothic Medium', fontsize=15)


ax = fig.add_subplot(1,2,2)
sns.countplot(y=df1.year,palette='viridis')
plt.title('Years representing how many drivers joined OLA each year',fontname='Franklin Gothic Medium', fontsize=15)
sns.despine()
plt.show()
```

```
<ipython-input-43-320a29748f03>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(y=df1.month,palette='viridis')
<ipython-input-43-320a29748f03>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(y=df1.year,palette='viridis')
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```

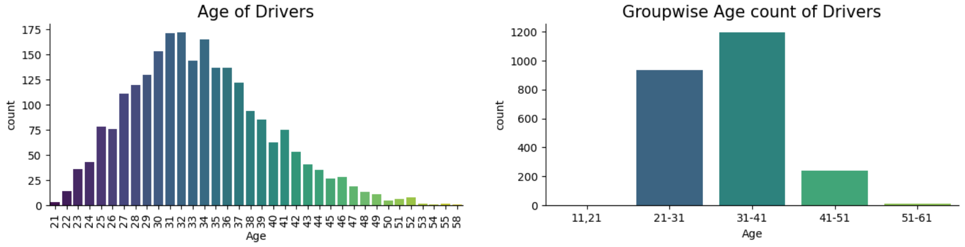

Months representing how many drivers joined OLA each month

Years representing how many drivers joined OLA each year

```
# July received the maximum number of drivers in 8 years.
# February and March receives the least number of Drivers joining OLA.
# Joining of Drivers receives a boost of about 500% after 2017.
```

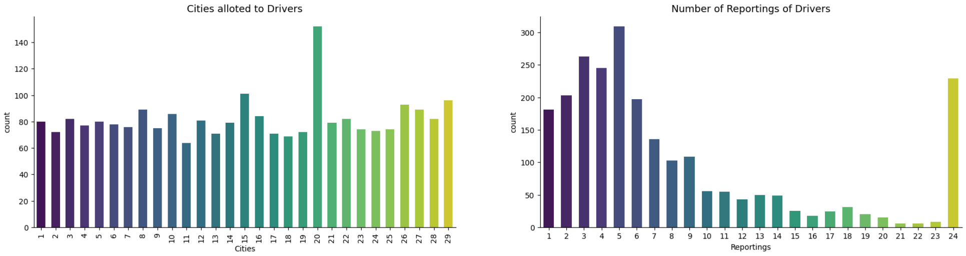
```
fig = plt.figure(figsize=(15,3))
ax = fig.add_subplot(121)
sns.countplot(x=df1.Age,palette='viridis',width=0.8)
plt.title('Age of Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.xticks(rotation=90)

ax = fig.add_subplot(122)
a = pd.cut(df1.Age,bins=[11,21,31,41,51,61],labels=['11,21','21-31','31-41','41-51','51-61'])
sns.countplot(x=a,palette='viridis')
plt.title('Groupwise Age count of Drivers',fontname='Franklin Gothic Medium', fontsize=15)
sns.despine()
plt.show()
```

```
<ipython-input-45-26f4f8796b87>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Age,palette='viridis',width=0.8)
<ipython-input-45-26f4f8796b87>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=a,palette='viridis')
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```



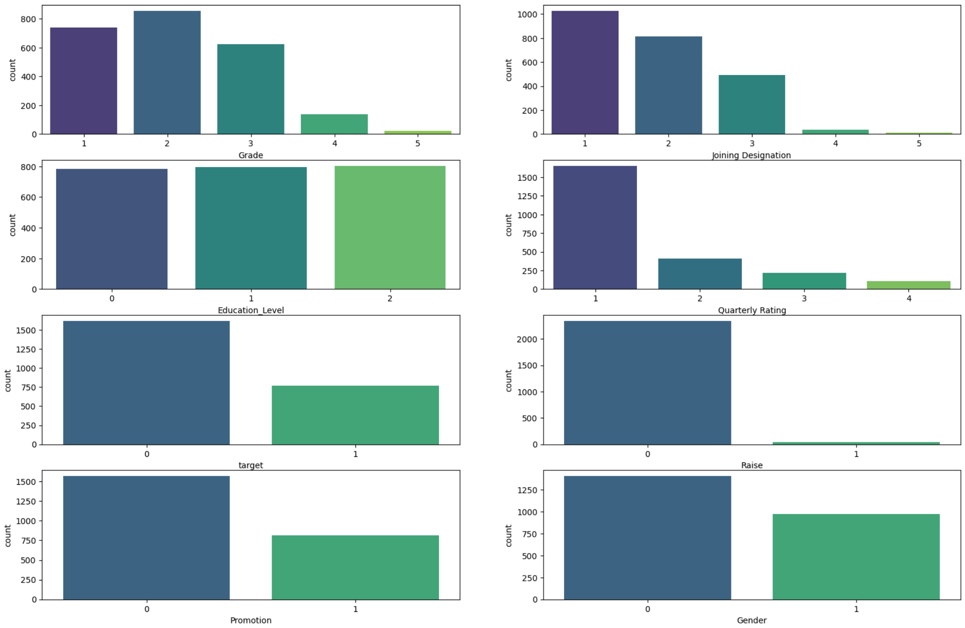
```
fig = plt.figure(figsize=(22,5))
ax = fig.add_subplot(121)
sns.countplot(x=df1.Cities,palette='viridis',width=0.6)
plt.title('Cities alloted to Drivers',fontname='Franklin Gothic Medium', fontsize=13)
plt.xticks(rotation=90)

ax = fig.add_subplot(122)
sns.countplot(x=df1.Reportings,palette='viridis',width=0.6)
plt.title('Number of Reportings of Drivers',fontname='Franklin Gothic Medium', fontsize=13)

sns.despine()
plt.show()
```

```
<ipython-input-46-8b2edd8fc541>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Cities,palette='viridis',width=0.6)
<ipython-input-46-8b2edd8fc541>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Reportings,palette='viridis',width=0.6)
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```



```
plt.figure(figsize=(20,13))
plt.subplot(4,2,1)
sns.countplot(x=df1.Grade,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,2)
sns.countplot(x=df1['Joining Designation'],palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,3)
sns.countplot(x=df1.Education_Level,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,4)
sns.countplot(x=df1['Quarterly Rating'],palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,5)
sns.countplot(x=df1.target,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,6)
sns.countplot(x=df1.Raise,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,7)
sns.countplot(x=df1.Promotion,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(4,2,8)
sns.countplot(x=df1.Gender,palette='viridis')
# plt.title('Grade given to different Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.show()
```

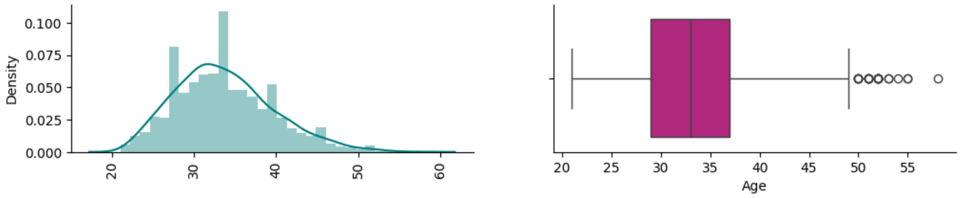
```
<ipython-input-48-e47fdf56ad97>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Grade,palette='viridis')
<ipython-input-48-e47fdf56ad97>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1['Joining Designation'],palette='viridis')
<ipython-input-48-e47fdf56ad97>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Education_Level,palette='viridis')
<ipython-input-48-e47fdf56ad97>:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1['Quarterly Rating'],palette='viridis')
<ipython-input-48-e47fdf56ad97>:15: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.target,palette='viridis')
<ipython-input-48-e47fdf56ad97>:18: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Raise,palette='viridis')
<ipython-input-48-e47fdf56ad97>:21: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Promotion,palette='viridis')
<ipython-input-48-e47fdf56ad97>:24: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=df1.Gender,palette='viridis')
```



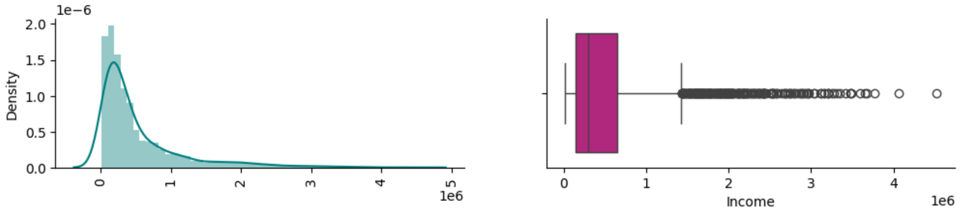
```
# Between 21 years(min age) to 58(max age) years of age, maximum number of drivers are 32 years o
# meanwhile the age group between 31-41 years of age receives the maximum number of drivers.
# 58.9% of the Drivers are male.
# City C20 has been used by the most of the drivers.
# There are 3 Education levels and all of them alomst have the equal distribution of Drivers.
# Grade 2 has been received by most of the Drivers and then the count of grade keeps on falling.
```

```
a = df1[['Age','Income','Total Business Value']]
for i in a:
    plt.figure(figsize=(12,2))
    plt.subplot(121)
    sns.distplot(x=df1[i],color='teal')
    plt.title('')
    plt.xticks(rotation=90)

    plt.subplot(122)
    sns.boxplot(x=df1[i],color='mediumvioletred')
    plt.title('')
    sns.despine()
    plt.show()
```

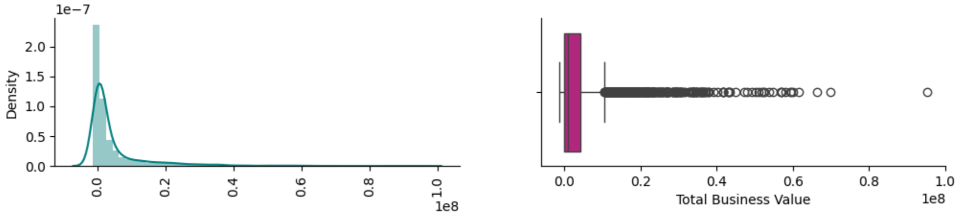
```
<ipython-input-51-c119a5f016d8>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='teal')
```



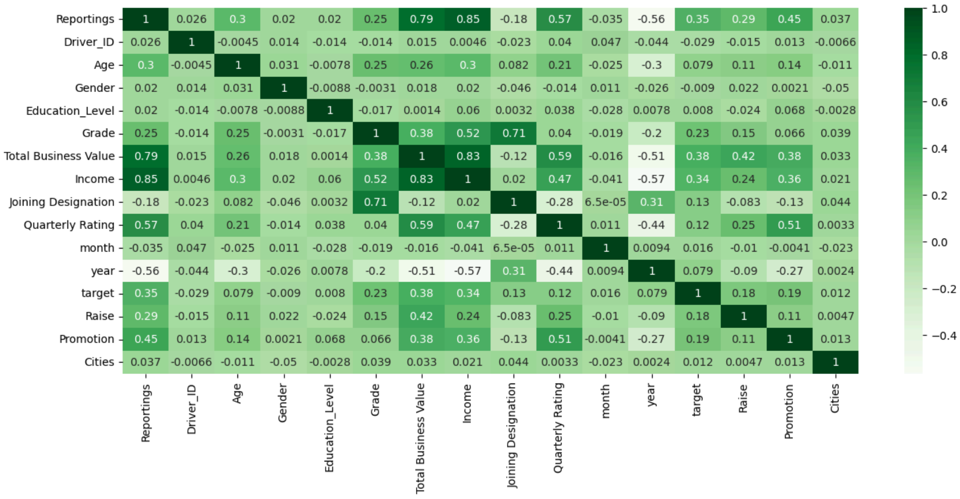
```
<ipython-input-51-c119a5f016d8>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='teal')
```



```
<ipython-input-51-c119a5f016d8>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='teal')
```

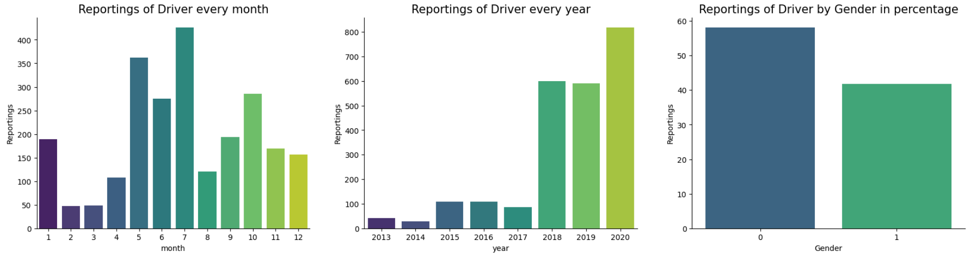


```
corr = df1.corr()
plt.figure(figsize=(15,6))
sns.heatmap(corr,annot=True,cmap='Greens')
plt.show()
```



```
fig = plt.figure(figsize=(22,5))
ax = fig.add_subplot(1,3,1)
grouped_months = df1.groupby(['month'])['Reportings'].count().reset_index()
sns.barplot(data=grouped_months,x='month',y='Reportings',palette='viridis')
plt.title('Reportings of Driver every month',fontname='Franklin Gothic Medium', fontsize=15)

ax = fig.add_subplot(1,3,2)
grouped_years = df1.groupby(['year'])['Reportings'].count().reset_index()
sns.barplot(x='year', y='Reportings', data=grouped_years,palette='viridis')
plt.title('Reportings of Driver every year',fontname='Franklin Gothic Medium', fontsize=15)

ax = fig.add_subplot(1,3,3)
grouped_gender = df1.groupby('Gender')['Reportings'].sum().reset_index()
grouped_gender['Reportings'] =(grouped_gender['Reportings']/sum(df1.Reportings)*100).round(2)
sns.barplot(x=grouped_gender['Gender'],y= grouped_gender['Reportings'],palette='viridis')
plt.title('Reportings of Driver by Gender in percentage',fontname='Franklin Gothic Medium', fontsize=15)
sns.despine()
sns.despine()
plt.show()
```

```
<ipython-input-54-e6a8fed92c02>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(data=grouped_months,x='month',y='Reportings',palette='viridis')
<ipython-input-54-e6a8fed92c02>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x='year', y='Reportings', data=grouped_years,palette='viridis')
<ipython-input-54-e6a8fed92c02>:15: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x=grouped_gender['Gender'],y= grouped_gender['Reportings'],palette='viri
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```



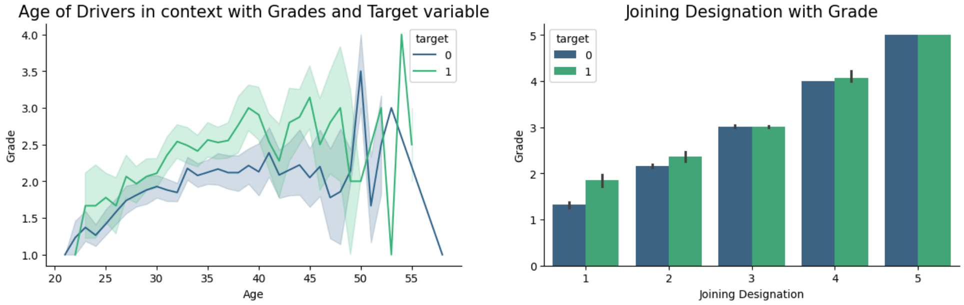
```
grouped_gender
```

	Gender	Reportings
0	0	58.12
1	1	41.88

Next steps: Generate code with `grouped_gender` View recommended plots

```
fig = plt.figure(figsize=(15,4))
ax = fig.add_subplot(1,2,1)
sns.lineplot(x=df1.Age,y=df1.Grade,hue=df1.target,palette='viridis')
plt.title('Age of Drivers in context with Grades and Target variable',fontname='Franklin Gothic Medium', fontsize=15)

ax = fig.add_subplot(1,2,2)
sns.barplot(data=df1, x="Joining Designation", y="Grade",palette='viridis',hue='target')
plt.title('Joining Designation with Grade',fontname='Franklin Gothic Medium', fontsize=15)
sns.despine()
plt.show()
```

```
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```

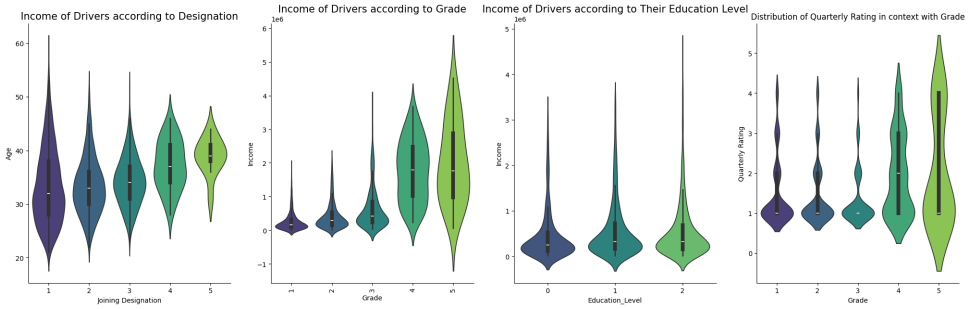


Age of Drivers in context with Grades and Target variable

Joining Designation with Grade

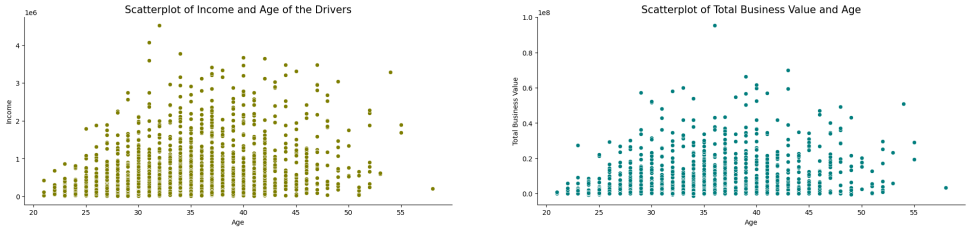
```
plt.figure(figsize=(25,7))
plt.subplot(1,4,1)
sns.violinplot(y=df1.Age,x=df1['Joining Designation'],palette='viridis')
plt.title('Income of Drivers according to Designation',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(1,4,2)
sns.violinplot(x=df1.Grade,y=df1.Income,palette='viridis')
plt.title('Income of Drivers according to Grade',fontname='Franklin Gothic Medium', fontsize=15)
plt.xticks(rotation=90)
plt.subplot(1,4,3)
sns.violinplot(x=df1.Education_Level,y=df1.Income,palette='viridis')
plt.title('Income of Drivers according to Their Education Level',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(1,4,4)
sns.violinplot(x=df1['Grade'],y=df1["Quarterly Rating"],palette='viridis')
plt.title('Distribution of Quarterly Rating in context with Grade')
sns.despine()
sns.despine()
plt.show()
```

```
<ipython-input-57-191e5e762c95>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.violinplot(y=df1.Age,x=df1['Joining Designation'],palette='viridis')
<ipython-input-57-191e5e762c95>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.violinplot(x=df1.Grade,y=df1.Income,palette='viridis')
<ipython-input-57-191e5e762c95>:10: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.violinplot(x=df1.Education_Level,y=df1.Income,palette='viridis')
<ipython-input-57-191e5e762c95>:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.violinplot(x=df1['Grade'],y=df1["Quarterly Rating"],palette='viridis')
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```



```
plt.figure(figsize=(25,5))
plt.subplot(1,2,1)
sns.scatterplot(x=df1.Age,y=df1.Income,color='olive')
plt.title('Scatterplot of Income and Age of the Drivers',fontname='Franklin Gothic Medium', fontsize=15)
plt.subplot(1,2,2)
sns.scatterplot(x=df1.Age,y=df1['Total Business Value'],color='teal')
plt.title('Scatterplot of Total Business Value and Age',fontname='Franklin Gothic Medium', fontsize=15)
sns.despine()
plt.show()
```

```
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
WARNING:matplotlib.font_manager:findfont: Font family 'Franklin Gothic Medium' not fou
```



```
grouped_gender = df1.groupby('Gender')['Income'].sum().reset_index()
grouped_education = df1.groupby('Education_Level')['Income'].sum().reset_index()
grouped_grade = df1.groupby('Grade')['Income'].sum().reset_index()
grouped_desig = df1.groupby('Joining Designation')['Income'].sum().reset_index()
grouped_QR = df1.groupby('Quarterly Rating')['Income'].sum().reset_index()
grouped_target = df1.groupby('target')['Income'].sum().reset_index()
grouped_raise = df1.groupby('Raise')['Income'].sum().reset_index()
grouped_promote = df1.groupby('Promotion')['Income'].sum().reset_index()
```

```
# Observations from plots
# So we see that there are 57% male employees and 43% female employees.
# The percentages of employees with different education levels are almost same for level 1 & 2.
# 97.3% of the employees who did not get a raise.
# Almost 43% of the employees joined at lowest designation (1). 34% joined at level 2, 20% at level 3 and below 2% joined at higher level
# Majority (35%) of the employees currently are at designation level 2, followed by designation level 1 (31%) and 3 (26%). Less than 5% o
# Only 54.6% of the employees received a promotion, while 45.4% did not. However, only 2.6% received a raise in income.
# Number of employees has been increase with increase in year as well as number of reportings.
# 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.
# Max reporting days is 24 days.
# About 55% of the reportings of the employees has got Quarlerly Rating 1.
# Number of reportings increases with increase in Income as well as Total Business Value.
```

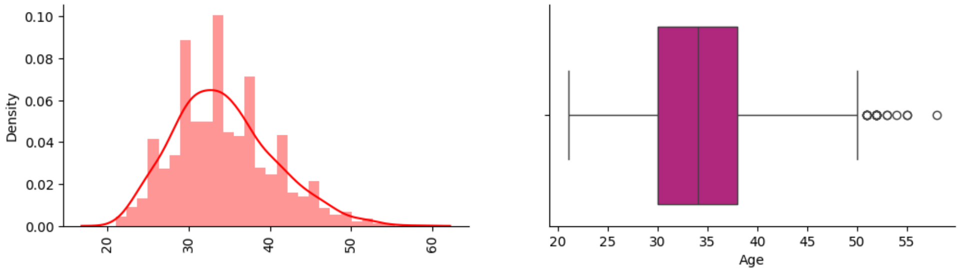
```
len(df1[df1['Total Business Value'] < 1])
# As we can notice Total Business Value column has some values in negative.
# We consider them as outlier which will affect the results of the our machine learning model.
# Considering the parts of datasets that has Total Business Value > 1.
# There are exactly 729 Driver having Total Business Value that less than 1.
```

```
729
```

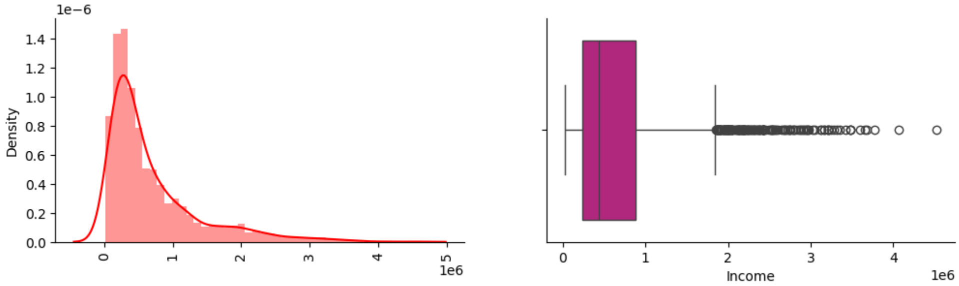
```
df1= df1[df1['Total Business Value'] > 1]
```

```
a =df1[['Age','Income','Total Business Value']]
for i in a:
    plt.figure(figsize=(12,3))
    plt.subplot(121)
    sns.distplot(x=df1[i],color='red')
    plt.xticks(rotation=90)
#     plt.figure(figsize=(9,5))
    plt.subplot(122)
    sns.boxplot(x=df1[i],color='mediumvioletred')
    sns.despine()
    plt.show()
```

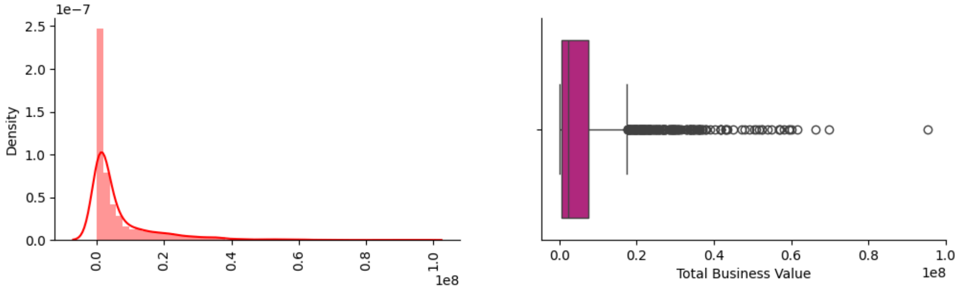
```
<ipython-input-65-706bcf0581b7>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='red')
```



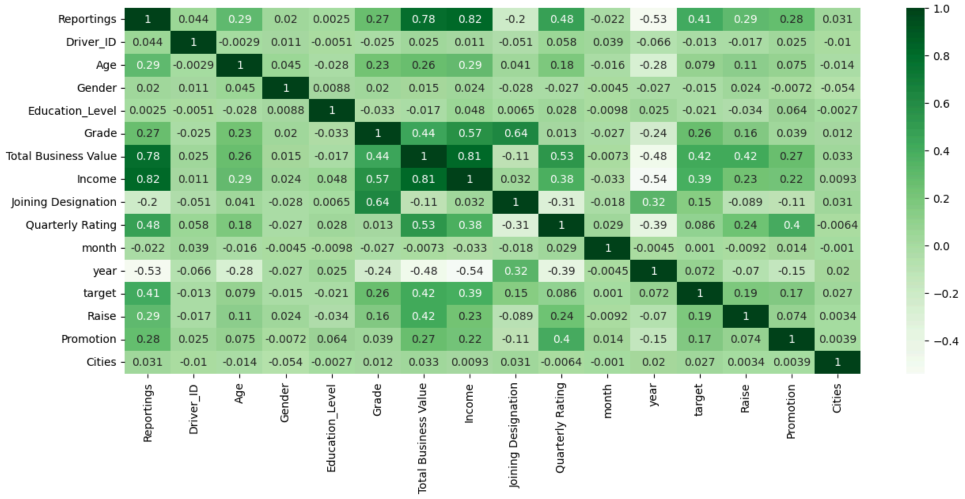
```
<ipython-input-65-706bcf0581b7>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='red')
```



```
<ipython-input-65-706bcf0581b7>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=df1[i],color='red')
```



```
corr = df1.corr()
plt.figure(figsize=(15,6))
sns.heatmap(corr,annot=True,cmap='Greens')
plt.show()
```



```
df1.describe().T
```

	count	mean	std	min	25%	50%	75%	
Reportings	1652.0	1.026998e+01	6.967589e+00	1.0	5.0	8.0	14.0	
Driver_ID	1652.0	1.390315e+03	8.082919e+02	1.0	679.5	1385.0	2097.0	
Age	1652.0	3.432385e+01	6.190776e+00	21.0	30.0	34.0	38.0	
Gender	1652.0	4.158596e-01	4.930188e-01	0.0	0.0	0.0	1.0	
Education_Level	1652.0	1.030872e+00	8.093284e-01	0.0	0.0	1.0	2.0	
Grade	1652.0	2.144068e+00	9.719606e-01	1.0	1.0	2.0	3.0	
Total Business Value	1652.0	6.613094e+06	1.032794e+07	19580.0	663022.5	2242080.0	7418392.5	9533
Income	1652.0	6.864932e+05	6.814522e+05	20886.0	236652.5	428960.0	877151.0	452
Joining Designation	1652.0	1.759685e+00	8.395129e-01	1.0	1.0	2.0	2.0	
Quarterly Rating	1652.0	1.700363e+00	9.237035e-01	1.0	1.0	1.0	2.0	
month	1652.0	6.914044e+00	3.021205e+00	1.0	5.0	7.0	9.0	
year	1652.0	2.018208e+03	1.730439e+00	2013.0	2018.0	2018.0	2020.0	
target	1652.0	3.619855e-01	4.807202e-01	0.0	0.0	0.0	1.0	
Raise	1652.0	2.602906e-02	1.592699e-01	0.0	0.0	0.0	0.0	
Promotion	1652.0	4.933414e-01	5.001070e-01	0.0	0.0	0.0	1.0	

```
from sklearn.preprocessing import StandardScaler
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
```

```
X = df1.drop('target',axis=1)
y = df1['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state= 42)
```

```
from sklearn.model_selection import learning_curve
def plot_learning_curve(estimator, X, Y, title):
    train_sizes, train_scores, test_scores, _, _ = learning_curve(estimator,X,Y,return_times=True)
    fig, axes = plt.subplots(1, 1, figsize = (15, 5))
    axes.set_title(title)
    axes.plot
    axes.set_xlabel("Training examples")
    axes.set_ylabel("Score")
    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)
    # Plot learning curve
#     32
    axes.grid()
    axes.fill_between(
    train_sizes,
    train_scores_mean - train_scores_std,
    train_scores_mean + train_scores_std,
    alpha=0.1,
    color="r",
    )
    axes.fill_between(
    train_sizes,
    test_scores_mean - test_scores_std,
    test_scores_mean + test_scores_std,
    alpha=0.1,
    color="g",
    )
    axes.plot(
    train_sizes, train_scores_mean, "o-", color="r", label="Training score"
    )
    axes.plot(
    train_sizes, test_scores_mean, "o-", color="g", label="Cross-validation score"
    )
    axes.legend(loc="best")
    plt.show()
```

```
X.head()
```

	Reportings	Driver_ID	Age	Gender	Education_Level	Grade	Total Business Value	Income	Joi Designat
0	3	1	28	0	2	1	1715580	172161	
2	5	4	43	0	2	2	350000	328015	
3	3	5	29	0	0	1	120360	139104	
4	5	6	31	1	1	3	1265000	393640	
7	6	12	35	0	2	1	2607180	168696	

Next steps: Generate code with X   View recommended plots

```
ss= StandardScaler()
ss.fit_transform(X_train)
```

```
array([[-0.61446611, -1.09640018,  1.70794584, ..., -0.16737851,
         1.023749  , -0.04979913],
       [ 1.93718866, -1.32951199,  1.54780698, ..., -0.16737851,
        -0.97680193, -0.5247786 ],
       [-0.18919032, -1.0914666 ,  0.26669606, ..., -0.16737851,
         1.023749  ,  1.25639439],
       ...,
       [-0.75622471,  0.03585718, -1.49483144, ..., -0.16737851,
        -0.97680193, -0.88101319],
       [ 0.51960268,  1.32105562, -1.33469258, ..., -0.16737851,
         1.023749  , -1.59348238],
       [-0.33094892,  0.60815284, -0.69413712, ..., -0.16737851,
        -0.97680193, -0.28728886]])
```

```
from sklearn.model_selection import cross_validate
```

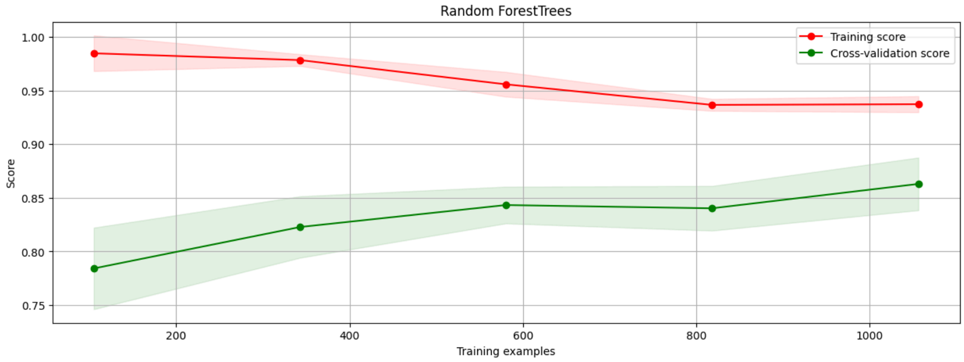
```
valid1 = cross_val_score(LogisticRegression(),X,y,cv=5)
print('Logistic Regression:',valid1.round(2))
print('Mean:',valid1.mean())
valid2 = cross_val_score( DecisionTreeClassifier(),X,y,cv=5)
print('Decision Tree:',valid2.round(3))
print('Mean:',valid2.mean())
valid3 = cross_val_score(RandomForestClassifier(),X,y,cv=5)
print('RandomForestClassifier():',valid3.round(2))
print('Mean:',valid3.mean())
valid4 = cross_val_score(GradientBoostingClassifier(),X,y,cv=5)
print('GradientBoostingClassifier:',valid4.round(3))
print('Mean:',valid4.mean())
valid5 =cross_val_score(XGBClassifier(),X,y,cv=5)
print('XGBoostClassifier:',valid1.round(2))
print('Mean:',valid5.mean())
```

```
Logistic Regression: [0.7  0.75 0.75 0.75 0.76]
Mean: 0.7415453629955141
Decision Tree: [0.861 0.873 0.855 0.845 0.852]
Mean: 0.8571308248649638
RandomForestClassifier(): [0.89 0.91 0.89 0.86 0.9 ]
Mean: 0.8916341664377919
GradientBoostingClassifier: [0.891 0.918 0.879 0.879 0.845]
Mean: 0.8825395953492631
XGBoostClassifier: [0.7  0.75 0.75 0.75 0.76]
Mean: 0.879520278311819
```

```
# Random Forest Classifier
rf_clf1 = RandomForestClassifier(criterion='gini',max_depth=7,max_features='sqrt',n_estimators=10)
rf_clf1.fit(X_train,y_train)
```

```
▾              RandomForestClassifier
RandomForestClassifier(max_depth=7, n_estimators=10)
```

```
plot_learning_curve(rf_clf1, X_train, y_train, "Random ForestTrees")
```



```
y_pred = rf_clf1.predict(X_test)
proba = rf_clf1.predict_proba(X_test)[:,1]
print("Train data accuracy:",rf_clf1.score(X_train, y_train))
print("Test data accuracy:",rf_clf1.score(X_test,y_test))
print('Accuracy of the model:', accuracy_score(y_test, y_pred))
print("ROC-AUC score test dataset: ", roc_auc_score(y_test, proba))
print('-'*70)
print(classification_report(y_test, y_pred))
print('-'*70)
cm1 = (confusion_matrix(y_test, y_pred))
print('Confusion Metrix')
print(confusion_matrix(y_test, y_pred))
```

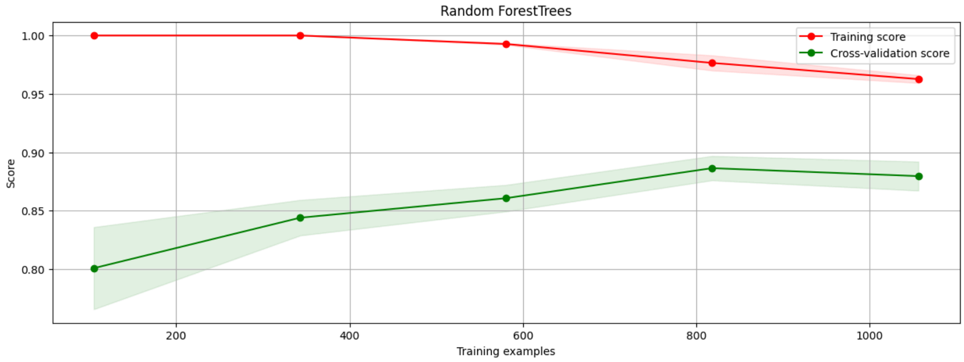
```
Train data accuracy: 0.9295987887963664
Test data accuracy: 0.8670694864048338
Accuracy of the model: 0.8670694864048338
ROC-AUC score test dataset:  0.9324061087735702
----------------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.88      0.92      0.90       207
           1       0.85      0.78      0.82       124

    accuracy                           0.87       331
   macro avg       0.86      0.85      0.86       331
```

```
weighted avg       0.87      0.87      0.87       331

----------------------------------------------------------------------
Confusion Metrix
[[190  17]
 [ 27  97]]
```

```
rf_clf_imp1 = rf_clf1.feature_importances_
```

```
gbc1 = GradientBoostingClassifier()
gbc1.fit(X_train, y_train)
y_pred = gbc1.predict(X_test)
proba =gbc1.predict_proba(X_test)[:, 1]
```

```
plot_learning_curve(gbc1, X_train, y_train, "Random ForestTrees")
```

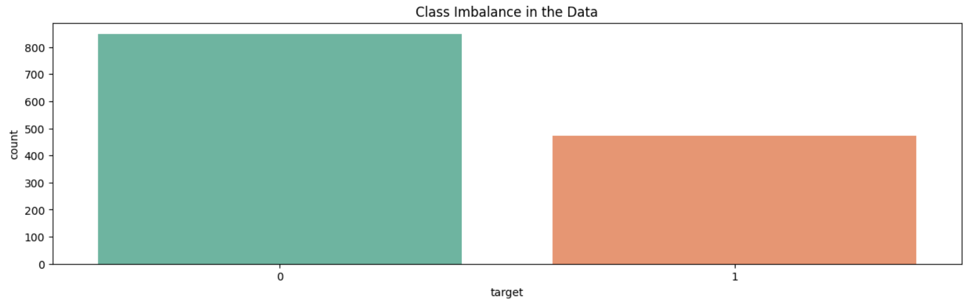


```
print('Test Score : ', gbc1.score(X_test, y_test))
print('Accuracy Score : ', accuracy_score(y_test, y_pred))
print("ROC-AUC score test dataset: ", roc_auc_score(y_test, proba))
print('-'*60)
print(classification_report(y_test, y_pred))
print('-'*60)
print('Confusion Matrix')
cm2 = (confusion_matrix(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
print('-'*60)
```

```
Test Score :  0.9003021148036254
Accuracy Score :  0.9003021148036254
ROC-AUC score test dataset:  0.9490416082281441
------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.91      0.94      0.92       207
           1       0.89      0.84      0.86       124

    accuracy                           0.90       331
   macro avg       0.90      0.89      0.89       331
weighted avg       0.90      0.90      0.90       331

------------------------------------------------------------
Confusion Matrix
[[194  13]
 [ 20 104]]
------------------------------------------------------------
```

```
# Class Imbalance Treatment
plt.figure(figsize=(15,4))
sns.countplot(x=y_train,palette='Set2')
plt.title('Class Imbalance in the Data')
plt.show()
```

```
<ipython-input-84-ee7e7afc7f5a>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.countplot(x=y_train,palette='Set2')
```



```
(y_train.value_counts()*100)/len(y_train)
```

```
0    64.118092
1    35.881908
Name: target, dtype: float64
```

```
from imblearn.over_sampling import SMOTE
```

```
smot = SMOTE(random_state=42)
X_train_smot,y_train_smot = smot.fit_resample(X_train,y_train.ravel())
```

```
X_train_smot.shape,y_train_smot.shape
```

```
((1694, 15), (1694,))
```

```
X_test.shape,y_test.shape
```

```
((331, 15), (331,))
```

```
from collections import Counter
c = Counter(y_train_smot)
print(c)
```

```
Counter({0: 847, 1: 847})
```

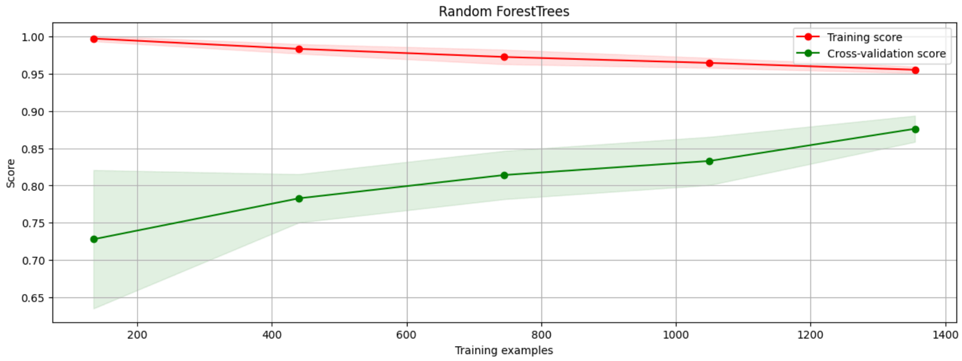
```
# Random Forest Classifier
clf = RandomForestClassifier()
clf.fit(X_train_smot,y_train_smot)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
clf = RandomForestClassifier(criterion='gini',max_depth=8,
                             max_features='sqrt',n_estimators= 19)
clf.fit(X_train_smot,y_train_smot)
```

```
▾               RandomForestClassifier
RandomForestClassifier(max_depth=8, n_estimators=19)
```

```
plot_learning_curve(clf, X_train_smot, y_train_smot, "Random ForestTrees")
```



```
y_pred = clf.predict(X_test)
print('-'*70)
print(classification_report(y_test, y_pred))
print('-'*70)
print('Confusion Metrix')
cm3 = confusion_matrix(y_test, y_pred)
print(confusion_matrix(y_test, y_pred))
```

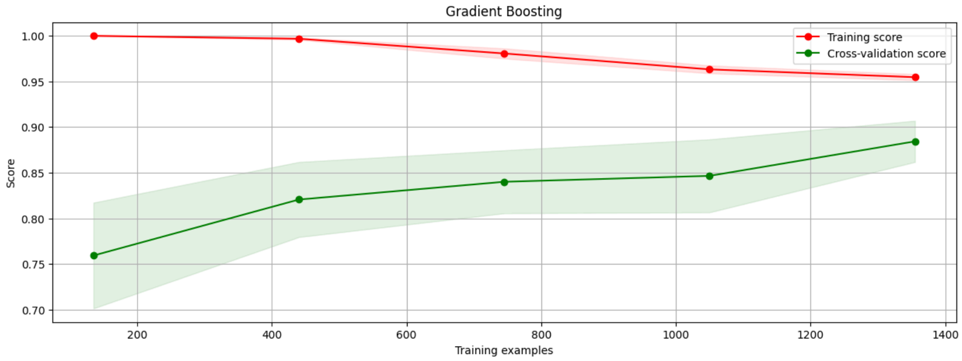
```
----------------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.95      0.86      0.90       207
           1       0.79      0.92      0.85       124

    accuracy                           0.88       331
   macro avg       0.87      0.89      0.87       331
weighted avg       0.89      0.88      0.88       331

----------------------------------------------------------------------
Confusion Metrix
[[177  30]
 [ 10 114]]
```

```
rf_clf_imp2= clf.feature_importances_
```

```
# Gradient Boosting
gbc2 = GradientBoostingClassifier()
gbc2.fit(X_train_smot, y_train_smot)
y_pred1 = gbc2.predict(X_test)
gbc_clf_imp2 = gbc2.feature_importances_
print('-'*60)
print(classification_report(y_test, y_pred1))
print('-'*60)
cm4 = confusion_matrix(y_test, y_pred1)
print('Confusion Matrix')
print(cm4)
print('-'*60)
```

```
------------------------------------------------------------
              precision    recall  f1-score   support

           0       0.93      0.89      0.91       207
           1       0.83      0.90      0.86       124

    accuracy                           0.89       331
   macro avg       0.88      0.89      0.89       331
weighted avg       0.90      0.89      0.89       331

------------------------------------------------------------
Confusion Matrix
[[185  22]
 [ 13 111]]
------------------------------------------------------------
```

```
plot_learning_curve(gbc2, X_train_smot, y_train_smot, "Gradient Boosting")
```



```
data1 = pd.DataFrame({'Column_Name':X.columns,
                      'RandomForestClassifier':rf_clf_imp1,
                      'XGBClassifier':gbc_clf_imp1})
```

```
data2 = pd.DataFrame({'Column_Name':X.columns,
                      'RandomForestClassifier':rf_clf_imp2,
                      'XGBClassifier':gbc_clf_imp2})
```

```
data1
```

	Column_Name	RandomForestClassifier	XGBClassifier
0	Reportings	0.229412	0.421604
1	Driver_ID	0.032307	0.011521
2	Age	0.025386	0.007870
3	Gender	0.006252	0.001717
4	Education_Level	0.008902	0.000841
5	Grade	0.029274	0.001956
6	Total Business Value	0.194754	0.124730
7	Income	0.110159	0.017137
8	Joining Designation	0.025428	0.005577
9	Quarterly Rating	0.050003	0.027930
10	month	0.017283	0.005742
11	year	0.210721	0.343451
12	Raise	0.021396	0.000000
13	Promotion	0.016191	0.018855
14	Cities	0.022533	0.011070

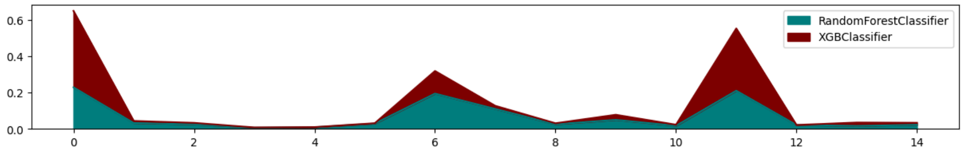
Next steps: Generate code with data1 View recommended plots

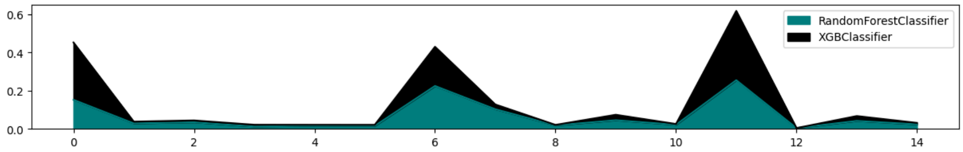
```
data2
```

	Column_Name	RandomForestClassifier	XGBClassifier
0	Reportings	0.152520	0.300373
1	Driver_ID	0.028835	0.009306
2	Age	0.034609	0.009589
3	Gender	0.012509	0.009140
4	Education_Level	0.016576	0.004790
5	Grade	0.015843	0.005402
6	Total Business Value	0.224981	0.205370
7	Income	0.103132	0.024976
8	Joining Designation	0.018496	0.002461
9	Quarterly Rating	0.045447	0.028597
10	month	0.021756	0.003862
11	year	0.255233	0.363063
12	Raise	0.004321	0.000000
13	Promotion	0.041896	0.025453
14	Cities	0.023845	0.007618

Next steps: Generate code with data2 View recommended plots

```
data1.plot(kind="area", figsize = (15,2),color=['teal','maroon'])

data2.plot(kind="area", figsize = (15,2),color=['teal','black'])

plt.show()
```





```
# calculating precision, reall and f1_score for every
tp1,fp1,fn1,tn1 =cm1[0][0],cm1[0][1],cm1[1][0],cm1[1][1]
tp2,fp2,fn2,tn2 =cm2[0][0],cm2[0][1],cm2[1][0],cm2[1][1]
tp3,fp3,fn3,tn3 =cm3[0][0],cm3[0][1],cm3[1][0],cm3[1][1]
tp4,fp4,fn4,tn4 =cm4[0][0],cm4[0][1],cm4[1][0],cm4[1][1]
precision1 = tp1/(tp1+fp1)
recall1 = tp1/(tp1+fn1)
precision2 = tp2/(tp2+fp2)
recall2 = tp2/(tp2+fn2)
precision3 = tp3/(tp3+fp3)
recall3 = tp3/(tp3+fn3)
precision4 = tp4/(tp4+fp4)
recall4 = tp4/(tp4+fn4)
f1_1 = (2*precision1*recall1)/(precision1+recall1)
f1_2 = (2*precision2*recall2)/(precision2+recall2)
f1_3 = (2*precision3*recall3)/(precision3+recall3)
f1_4 =(2*precision4*recall4)/(precision4+recall4)
```

```
df = pd.DataFrame({'Model':['RandomForest','GradientBoosting','RandomForest','GradientBoosting'],
                   'Class':['imbalanced','imbalanced','balanced','balanced'],
                   'True_pos':[tp1,tp2,tp3,tp4],
                   'Fal_pos':[fp1,fp2,fp3,fp4],
                   'Fal_neg':[fn1,fn2,fn3,fn4],
                   'True_neg':[tn1,tn2,tn3,tn4],
                   'F1_score%':[f1_1*100,f1_2*100,f1_3*100,f1_4*100],
                   'Precision%':[precision1*100,precision2*100,precision3*100,precision4*100],
                   'Recall%':[recall1*100,recall2*100,recall3*100,recall4*100]})
```

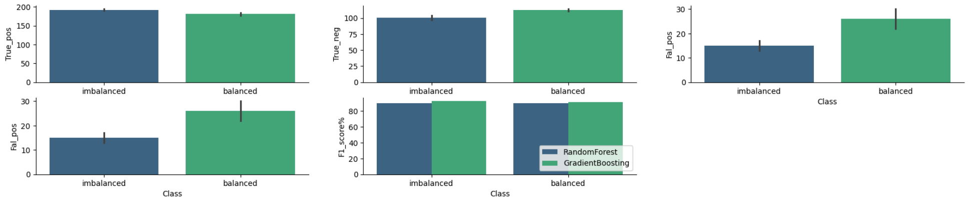
```
df
```

	Model	Class	True_pos	Fal_pos	Fal_neg	True_neg	F1_score%	Precision%
0	RandomForest	imbalanced	190	17	27	97	89.622642	91.787440
1	GradientBoosting	imbalanced	194	13	20	104	92.161520	93.719807
2	RandomForest	balanced	177	30	10	114	89.847716	85.507246
3	GradientBoosting	balanced	185	22	13	111	91.358025	89.371981

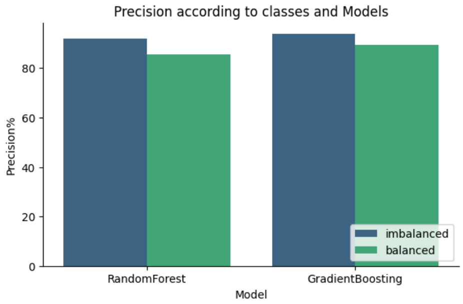
Next steps: Generate code with df   View recommended plots

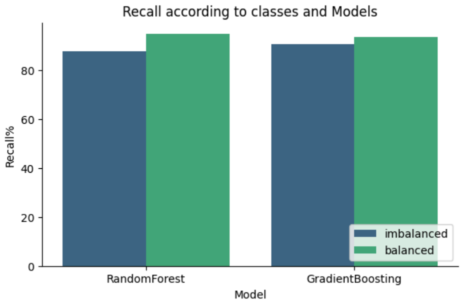
```
plt.figure(figsize=(22,4))
plt.subplot(2,3,1)
sns.barplot(x=df.Class,y=df.True_pos,palette='viridis')
# plt.show()
plt.subplot(2,3,2)
sns.barplot(x=df.Class,y=df.True_neg,palette='viridis')
# plt.show()
plt.subplot(2,3,3)
sns.barplot(x=df.Class,y=df.Fal_pos,palette='viridis')
# plt.show()
plt.subplot(2,3,4)
sns.barplot(x=df.Class,y=df.Fal_pos,palette='viridis')
plt.subplot(2,3,5)
sns.barplot(x=df.Class,y=df['F1_score%'],palette='viridis',hue=df.Model)
plt.legend(loc='lower right')
sns.despine()
plt.show()
```

```
<ipython-input-106-a04a1f9ff36e>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x=df.Class,y=df.True_pos,palette='viridis')
<ipython-input-106-a04a1f9ff36e>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x=df.Class,y=df.True_neg,palette='viridis')
<ipython-input-106-a04a1f9ff36e>:9: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x=df.Class,y=df.Fal_pos,palette='viridis')
<ipython-input-106-a04a1f9ff36e>:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0

  sns.barplot(x=df.Class,y=df.Fal_pos,palette='viridis')
```



```
plt.figure(figsize=(15,4))
plt.subplot(1,2,1)
sns.barplot(x=df.Model,y=df['Precision%'],hue=df.Class,palette='viridis')
plt.title('Precision according to classes and Models')
plt.legend(loc='lower right')
plt.subplot(1,2,2)
sns.barplot(x=df.Model,y=df['Recall%'],hue=df.Class,palette='viridis')
plt.title('Recall according to classes and Models')
plt.legend(loc='lower right')
sns.despine()
plt.show()
```





```
# Insights & Recommendations
# So we see that there are 57% male employees and 43% female employees.
# The percentages of employees with different education levels are almost same for level 1 & 2.
# 97.3% of the employees who did not get a raise.
# Almost 43% of the employees joined at lowest designation (1). 34% joined at level 2, 20% at level 3 and below 2% joined at higher level
# Majority (35%) of the employees currently are at designation level 2, followed by designation level 1 (31%) and 3 (26%). Less than 5% o
# Only 54.6% of the employees received a promotion, while 45.4% did not. However, only 2.6% received a raise in income.
# Number of employees has been increase with increase in year as well as number of reportings.
# 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.
# Top reporting days is 24 days.
# About 55% of the reportings of the employees has got Quarlerly Rating 1.
# Number of reportings increases with increase in Income as well as Total Business Value.
# Recall increased after treatment of data imbalance and is performing bettee in Gradient Boosting.
# Precision dropped after treatment of data imbalance and is performing better in Random Forest.
# F1_score incresed after the treatment of imabalanced data and in Gradient Boosting.
```

```
# Insights & Recommendations
# So we see that there are 57% male employees and 43% female employees.
# The percentages of employees with different education levels are almost same for level 1 & 2.
# 97.3% of the employees who did not get a raise.
# Almost 43% of the employees joined at lowest designation (1). 34% joined at level 2, 20% at level 3 and below 2% joined at higher level
# Majority (35%) of the employees currently are at designation level 2, followed by designation level 1 (31%) and 3 (26%). Less than 5% o
# Only 54.6% of the employees received a promotion, while 45.4% did not. However, only 2.6% received a raise in income.
# Number of employees has been increase with increase in year as well as number of reportings.
# 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.
# Top reporting days is 24 days.
# About 55% of the reportings of the employees has got Quarlerly Rating 1.
# Number of reportings increases with increase in Income as well as Total Business Value.
# Recall increased after treatment of data imbalance and is performing bettee in Gradient Boosting.
# Precision dropped after treatment of data imbalance and is performing better in Random Forest.
# F1_score incresed after the treatment of imabalanced data and in Gradient Boosting.
```