This is sort note on Compleate EDA ----------{Prince Kumar Sharma}

Data Detective : Tips and tricks for coducting effective effective exploratory data analysis

EDA (Exploratory data Analysis)

EDA is an approach to analyzing and undestanding data that involves summrizing ,visulizing and identifying patterns and relationships and approach that can be used in EDA, and specific techniques used will depend on the nature of the data

Few techniques that are given below that are often used in EDA

1. Visulization : PLotting the data in varius ways can help reval patterns and trends. few common types of plots :- scatter,plots,line plots, bar, and histograms,countplots
2. Summary Statistics :- Caluculating summary statistics such as mean, median,and Standard Deviation can provides usefull information about the distribution and spread of the data.
3. Correlation Analysis :- Examining the relationship betweeen different variables can help identify correlations and dependenies
4. Data Cleaning :- Removing Missing or incorrect values and ensuring that the data is in a consistent format is an importanat step in EDA. 5.Dimensionality reduction :-techniques such as principal component analysis (PCA) can be used in reduce the number of dimention int the data ,making easier ot visulize and analyze. 6.Anomaly Detection :- Identifying unusual or unexpected values in the data can be important in identifying errors or outliers.
5. Feature engineering :- creating a new features or transforming existing features can improve the performance of the machine learning models and facilitate analysis.

Overall, The goal of EDA to gain the better understanding of the data ,identify potential issues or problems

Now i will study more detail all the points mentioned above

Visulization

With the help of visulization we easily undestand that's data what wants to say to you

Import Libraray and load any data

In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [4]:
```
x=sns.load_dataset("tips")
```

In [5]:
```
x
```

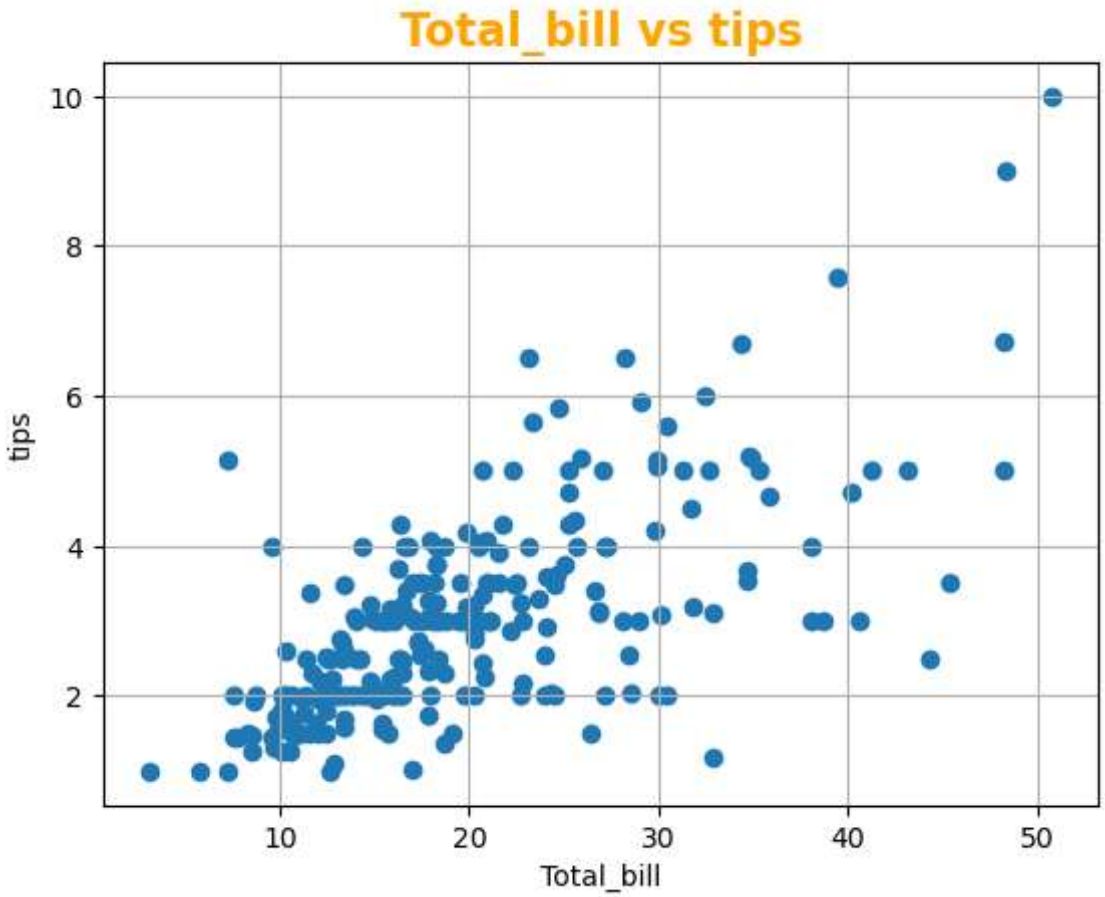
Out[5]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

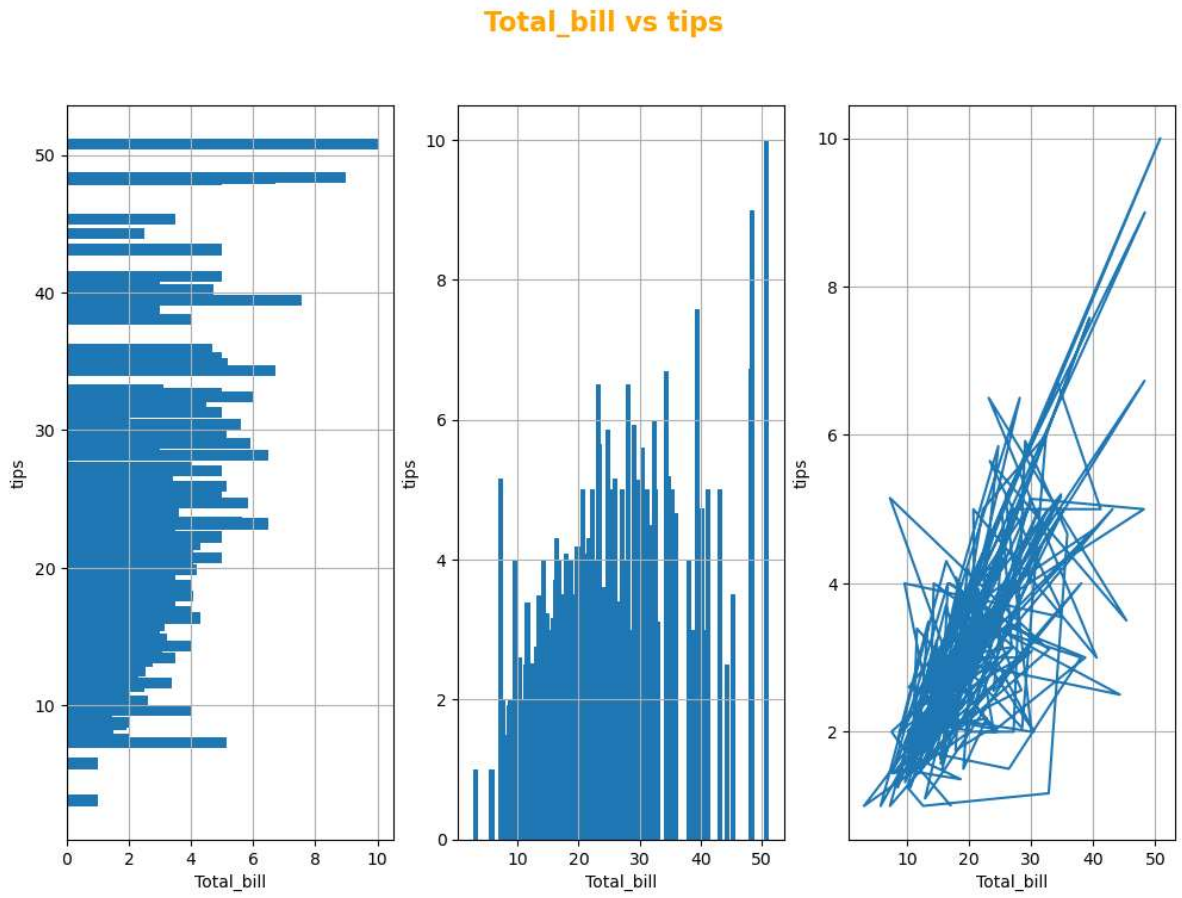
244 rows × 7 columns

In [16]:
```
plt.scatter(x["total_bill"],x["tip"])
plt.grid()
plt.xlabel("Total_bill")
plt.ylabel("tips")
plt.title("Total_bill vs tips",fontweight="bold",fontsize=16,color="orange")
```

Out[16]:
```
Text(0.5, 1.0, 'Total_bill vs tips')
```



In [29]:
```
plt.figure(figsize=(12,8))
plt.subplot(1,3,1)
plt.barh(x["total_bill"],x["tip"])#,capstyle='butt')
plt.grid()
plt.xlabel("Total_bill")
plt.ylabel("tips")
#plt.title("Total_bill vs tips",fontweight="bold",fontsize=16,color="orange")
plt.subplot(1,3,2)
plt.bar(x["total_bill"],x["tip"])#,capstyle='butt')
plt.grid()
plt.xlabel("Total_bill")
plt.ylabel("tips")
plt.subplot(1,3,3)
plt.plot(x["total_bill"],x["tip"])#,capstyle='butt')
plt.grid()
plt.xlabel("Total_bill")
plt.ylabel("tips")
#plt.title("Total_bill vs tips",fontweight="bold",fontsize=16,color="orange")
plt.suptitle("Total_bill vs tips",fontweight="bold",fontsize=16,color="orange"
plt.show()
```



In [ ]:

Summary statistics

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

3. Correlation Analysis

importing the usefull library

In [35]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [36]:
```
x=sns.load_dataset("titanic")
x=sns.load_dataset("tips")
```

In [37]:
```
x
```

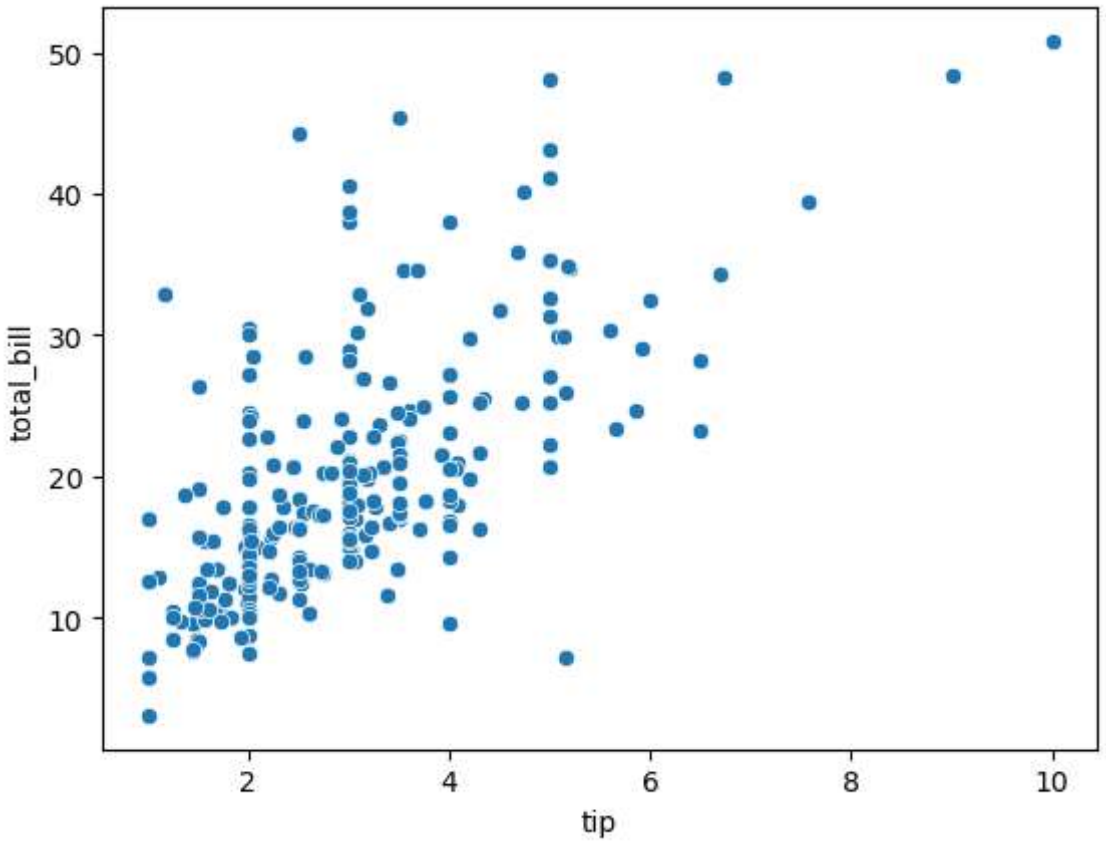
Out[37]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [39]:
```
sns.scatterplot(data=x,x="tip",y="total_bill")
```

Out[39]:
```
<AxesSubplot:xlabel='tip', ylabel='total_bill'>
```

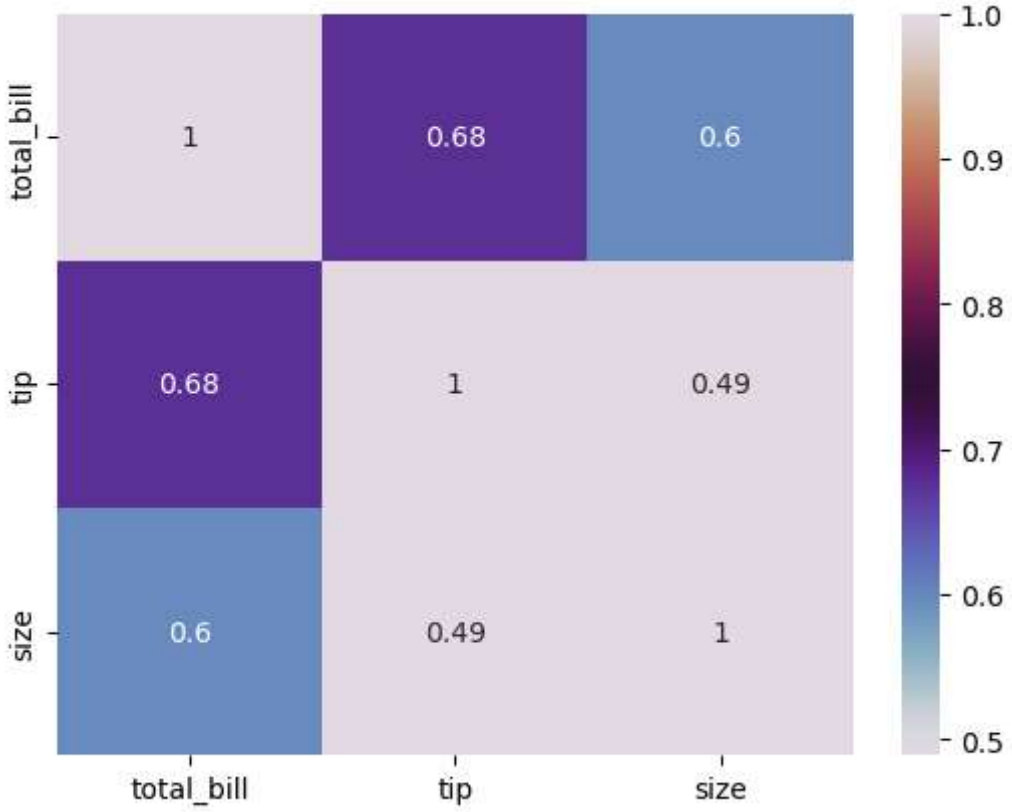


In [40]:
```
x.corr()
```

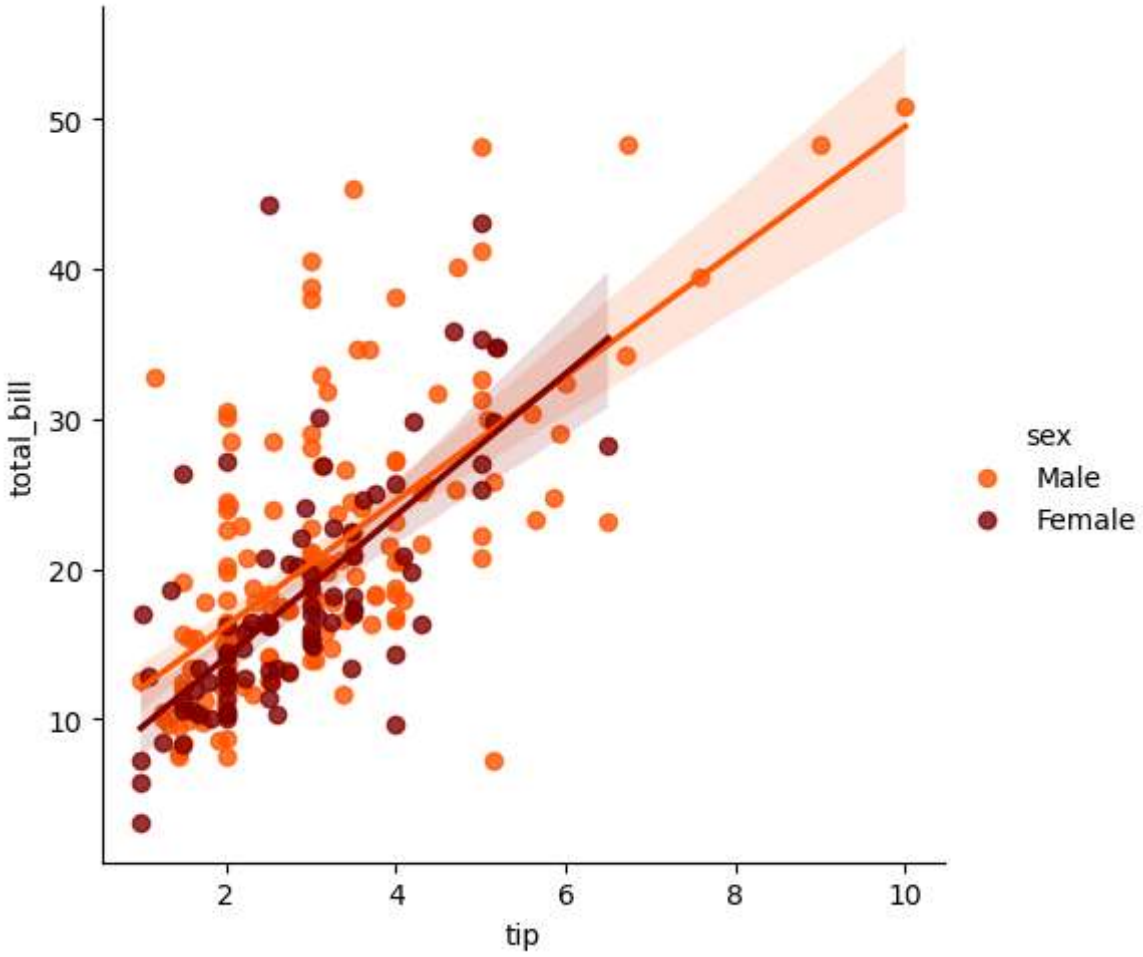
Out[40]:

	total_bill	tip	size
total_bill	1.000000	0.675734	0.598315
tip	0.675734	1.000000	0.489299
size	0.598315	0.489299	1.000000

In [45]:
```
sns.heatmap(x.corr(),annot=True,cmap="twilight");
```



In [60]:
```
sns.lmplot(data=x,x="tip",y="total_bill",hue="sex",palette="gist_heat_r");
```



In [ ]:
```
```

4. Data Cleaning

In [61]:
```
x
```

Out[61]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [63]:
```
x["tip"].mask(x["tip"]>1,100,inplace=True)
```

In [66]:
```
x['tip'].mask(x["tip"]==100,"prince",inplace=True)
```

In [67]:
```
x
```

Out[67]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	prince	Female	No	Sun	Dinner	2
1	10.34	prince	Male	No	Sun	Dinner	3
2	21.01	prince	Male	No	Sun	Dinner	3
3	23.68	prince	Male	No	Sun	Dinner	2
4	24.59	prince	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	prince	Male	No	Sat	Dinner	3
240	27.18	prince	Female	Yes	Sat	Dinner	2
241	22.67	prince	Male	Yes	Sat	Dinner	2
242	17.82	prince	Male	No	Sat	Dinner	2
243	18.78	prince	Female	No	Thur	Dinner	2

244 rows × 7 columns

In [ ]:

5. Dimensionality reduction :

Its reduce the number of dimention in the data then we easily analyze and machine learning model perform well

In [70]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
```

In [82]:
```
x=sns.load_dataset("tips")
```

In [83]:
```
x
```

Out[83]:

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

244 rows × 7 columns

To select the only columns thats in numberr form than use this method [df.select_dtypes(include="number")

In [85]:
```
data=x.select_dtypes(include="number")
```

In [86]:
```
data
```

Out[86]:

	total_bill	tip	size
0	16.99	1.01	2
1	10.34	1.66	3
2	21.01	3.50	3
3	23.68	3.31	2
4	24.59	3.61	4
...	...	...	...
239	29.03	5.92	3
240	27.18	2.00	2
241	22.67	2.00	2
242	17.82	1.75	2
243	18.78	3.00	2

244 rows × 3 columns

In [89]:
```
pca=PCA(n_components=1)
pca.fit(data)
```

Out[89]:
```
PCA(n_components=1)
```

In [92]:
```
data_2=pca.transform(data)
```

In [93]:
```
pca.explained_variance_
```

Out[93]:
```
array([80.46625221])
```

In [97]:
```
data_2[0:10]
```

Out[97]:
```
array([[ -3.02110913],
       [ -9.48741674],
       [  1.29525466],
       [  3.86067374],
       [  4.92348425],
       [  5.73436209],
       [-11.07349643],
       [  7.14415398],
       [ -4.85577404],
       [ -4.97957443]])
```

6. Anomaly Detection

Its identify the unusual and unexpected values that effect my machine learning model accuracy

In [99]:
```
import pandas as pd
import numpy as np
from sklearn.ensemble import IsolationForest
```

In [101]:
```
x=pd.read_csv("C:/Users/Prince kumar/OneDrive/Desktop/ml data by prince/house
```

In [102]:
```
x
```

Out[102]:

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	
0	1	1303834	23	3	single	rented	no	M
1	2	7574516	40	10	single	rented	no	
2	3	3991815	66	4	married	rented	no	
3	4	6256451	41	2	single	rented	yes	
4	5	5768871	47	11	single	rented	no	
...	...	...	...	...	...	...	...	
251995	251996	8154883	43	13	single	rented	no	
251996	251997	2843572	26	10	single	rented	no	
251997	251998	4522448	46	7	single	rented	no	
251998	251999	6507128	45	0	single	rented	no	
251999	252000	9070230	70	17	single	rented	no	

252000 rows × 13 columns

Detecting the Anomaly data

In [115]:
```
isolation=IsolationForest(contamination=0.1)
isolation.fit(x[["Age"]])
```

```
C:\Users\Prince kumar\anaconda3\lib\site-packages\sklearn\base.py:450: UserWa
rning: X does not have valid feature names, but IsolationForest was fitted wi
th feature names
  warnings.warn(
```

Out[115]:
```
IsolationForest(contamination=0.1)
```

In [116]:
```
anomial = isolation.predict(x[["Age"]])
```

In [132]:
```
a=pd.value_counts(anomial).reset_index()
a.columns=["data","number"]
a.index=["usefull","Anomial"]
a
```

Out[132]:

	data	number
usefull	1	230489
Anomial	-1	21511

- (1 --> Usefull data)
- (-1 --> Anomial data)

In [ ]:
```
```

7. Feature engineering

Feature engineering is the process of where we creating new feature from existing data that can be improve the machine models.

In [144]:
```
x=pd.read_csv("C:/Users/Prince kumar/OneDrive/Desktop/ml data by prince/house
```

In [145]:
```
x.head()
```

Out[145]:

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	Pro
0	1	1303834	23	3	single	rented	no	Mechanical_
1	2	7574516	40	10	single	rented	no	Software_D
2	3	3991815	66	4	married	rented	no	Technic
3	4	6256451	41	2	single	rented	yes	Software_D
4	5	5768871	47	11	single	rented	no	Civil

CountEncoding

In [157]:
```
x["Married/Single"]=x["Married/Single"].map(x["Married/Single"].value_counts(r
```

In [158]:
```
x["Married/Single"].value_counts()
```

Out[158]:
```
0.897905    226272
0.102095     25728
Name: Married/Single, dtype: int64
```

In [159]:
```
x["Profession"].value_counts()
```

Out[159]:
```
0.023639    5957
0.023040    5806
0.021417    5397
0.021389    5390
0.021317    5372
0.021266    5359
0.021258    5357
0.021048    5304
0.020956    5281
0.020869    5259
0.020833    5250
0.020702    5217
0.020655    5205
0.020615    5195
0.020548    5178
0.020504    5167
0.020500    5166
0.020349    5128
0.020345    5127
0.020083    5061
0.020052    5053
0.020004    5041
0.019980    5035
0.019802    4990
0.019619    4944
0.019369    4881
0.019302    4864
0.019290    4861
0.019119    4818
0.019079    4808
0.018976    4782
0.018972    4781
0.018937    4772
0.018881    4758
0.018798    4737
0.018766    4729
0.018706    4714
0.018540    4672
0.018524    4668
0.018496    4661
0.018480    4657
0.018393    4635
0.018365    4628
0.018317    4616
0.018246    4598
0.018147    4573
0.017885    4507
0.017829    4493
0.017512    4413
0.016218    4087
0.016063    4048
Name: Profession, dtype: int64
```

In [160]:
```
x["Profession"]=x["Profession"].map(x["Profession"].value_counts(normalize=Tru
```

In [161]:
```
x["Profession"].value_counts()
```

Out[161]:
```
0.023639    5957
0.023040    5806
0.021417    5397
0.021389    5390
0.021317    5372
0.021266    5359
0.021258    5357
0.021048    5304
0.020956    5281
0.020869    5259
0.020833    5250
0.020702    5217
0.020655    5205
0.020615    5195
0.020548    5178
0.020504    5167
0.020500    5166
0.020349    5128
0.020345    5127
0.020083    5061
0.020052    5053
0.020004    5041
0.019980    5035
0.019802    4990
0.019619    4944
0.019369    4881
0.019302    4864
0.019290    4861
0.019119    4818
0.019079    4808
0.018976    4782
0.018972    4781
0.018937    4772
0.018881    4758
0.018798    4737
0.018766    4729
0.018706    4714
0.018540    4672
0.018524    4668
0.018496    4661
0.018480    4657
0.018393    4635
0.018365    4628
0.018317    4616
0.018246    4598
0.018147    4573
0.017885    4507
0.017829    4493
0.017512    4413
0.016218    4087
0.016063    4048
Name: Profession, dtype: int64
```

Target Encoding

In [162]:
```
x=pd.read_csv("C:/Users/Prince kumar/OneDrive/Desktop/ml data by prince/house
```

In [163]:
```
x.head()
```

Out[163]:

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	Pro
0	1	1303834	23	3	single	rented	no	Mechanical_
1	2	7574516	40	10	single	rented	no	Software_D
2	3	3991815	66	4	married	rented	no	Technic
3	4	6256451	41	2	single	rented	yes	Software_D
4	5	5768871	47	11	single	rented	no	Civil

In [166]:
```
x["target_encoded"]=x.groupby("STATE")["Income"].transform("mean")
```

In [167]:
```
x
```

Out[167]:

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	
0	1	1303834	23	3	single	rented	no	M
1	2	7574516	40	10	single	rented	no	
2	3	3991815	66	4	married	rented	no	
3	4	6256451	41	2	single	rented	yes	
4	5	5768871	47	11	single	rented	no	
...	...	...	...	...	...	...	...	
251995	251996	8154883	43	13	single	rented	no	
251996	251997	2843572	26	10	single	rented	no	
251997	251998	4522448	46	7	single	rented	no	
251998	251999	6507128	45	0	single	rented	no	
251999	252000	9070230	70	17	single	rented	no	

252000 rows × 14 columns

In [168]:
```
x["CITY"]=x.groupby("CITY")["Income"].transform('mean')
```

In [169]:
```
x
```

Out[169]:

	Id	Income	Age	Experience	Married/Single	House_Ownership	Car_Ownership	
0	1	1303834	23	3	single	rented	no	M
1	2	7574516	40	10	single	rented	no	
2	3	3991815	66	4	married	rented	no	
3	4	6256451	41	2	single	rented	yes	
4	5	5768871	47	11	single	rented	no	
...	...	...	...	...	...	...	...	
251995	251996	8154883	43	13	single	rented	no	
251996	251997	2843572	26	10	single	rented	no	
251997	251998	4522448	46	7	single	rented	no	
251998	251999	6507128	45	0	single	rented	no	
251999	252000	9070230	70	17	single	rented	no	

252000 rows × 14 columns

In [170]:
```
x["CITY"].value_counts()
```

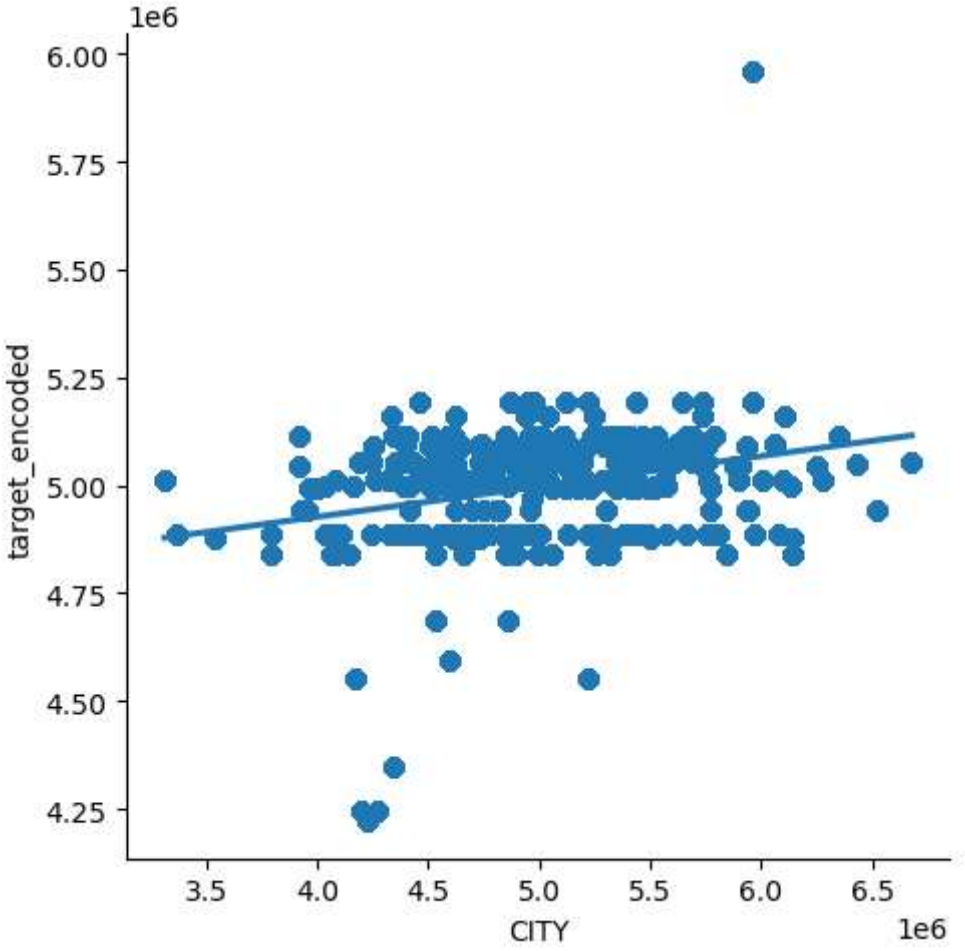
Out[170]:
```
4.999348e+06    1259
5.130638e+06    1208
5.291065e+06    1185
4.813669e+06    1180
5.355433e+06    1172
                ...
5.426260e+06     486
5.939802e+06     459
4.437989e+06     457
4.068519e+06     448
5.124683e+06     431
Name: CITY, Length: 317, dtype: int64
```

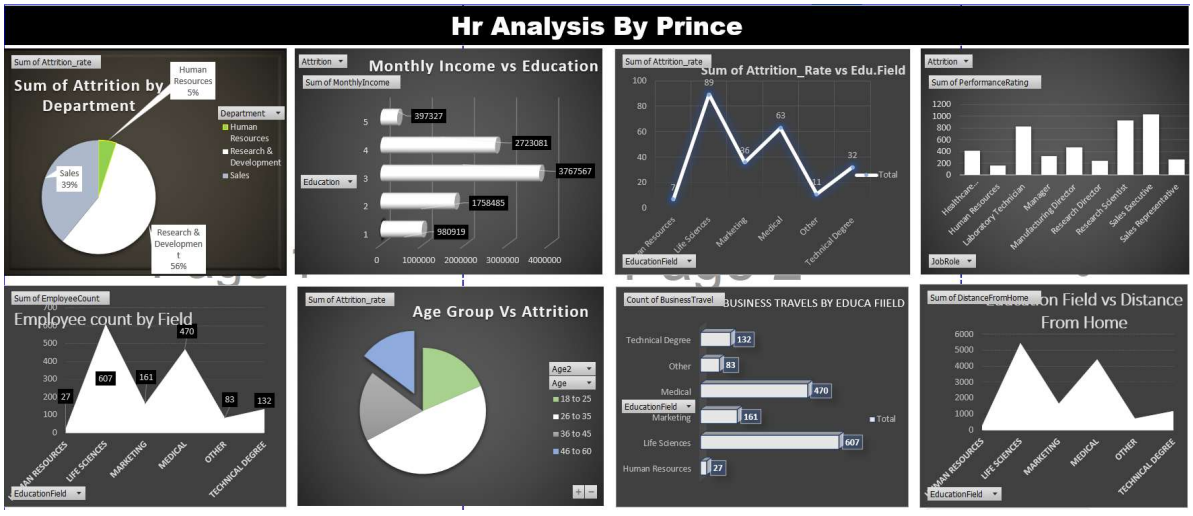
In [ ]:
```

```

In [174]:
```
sns.lmplot(data=x,x='CITY',y="target_encoded")
```

Out[174]:
```
<seaborn.axisgrid.FacetGrid at 0x15caa059280>
```





In [ ]:
```
```