



**RAJALAKSHMI ENGINEERING COLLEGE**

*Approved by AICTE | Affiliated to Anna University | Accredited by NAAC*

**Department of Computer Science and  
Engineering**

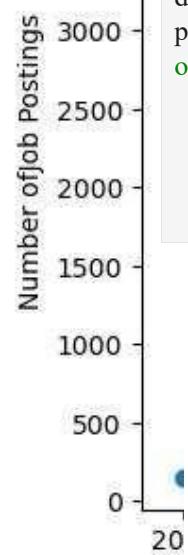
**CS23334 Fundamentals of Data Science Lab  
III semester II Year (2023R)**

**Name of the Student : Rohith R**

**Register Number : 240701804**

### Exercise 1: A]

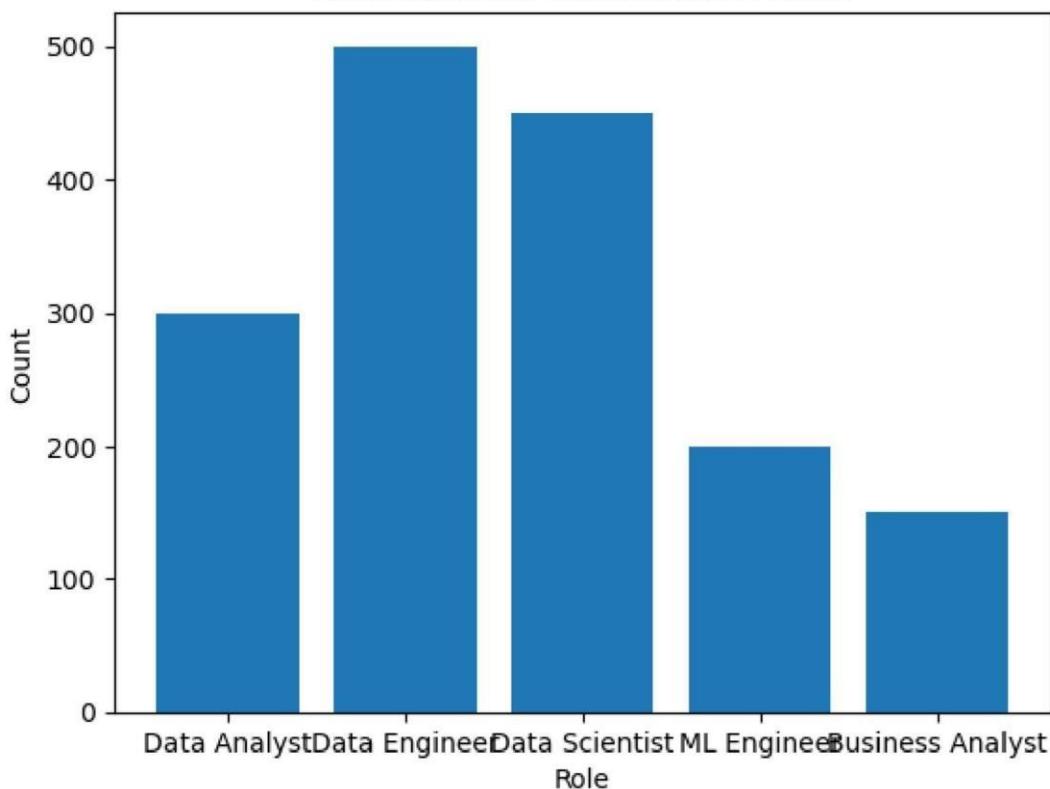
```
import pandas as pd import matplotlib.pyplot as plt  
data = {'Year': list(range(2010, 2021)),  
'Job Postings': [150, 300, 450, 600, 800, 1200, 1600, 2100, 2700,  
3400, 4200]}  
  
df = pd.DataFrame(data) plt.plot(df['Year'], df['Job Postings'], marker='o')  
plt.title("Trend of Data Science Job Postings") plt.xlabel('Year') plt.ylabel('Number  
of Job Postings') plt.show()
```



### B]

```
roles = ['Data Analyst', 'Data Engineer', 'Data Scientist', 'ML  
Engineer',  
'Business Analyst'] counts = [300, 500, 450, 200,  
150] plt.bar(roles, counts)  
plt.title("Distribution of Data Science Roles") plt.xlabel('Role') plt.ylabel('Count')  
plt.show()
```

## Distribution of Data Science Roles



```
structured_data = pd.DataFrame({
    'ID': [1, 2, 3],
    'Name': ['Alice', 'Bob', 'Charlie'],
    'Age': [25, 30, 35]
})
print("Structured Data:\n", structured_data)
unstructured_data ="This is an example of unstructured data. It can be
a piece of text, an image, or a video file."
print("\nUnstructured Data:\n", unstructured_data)
semi_structured_data = {'ID': 1, 'Name': 'Alice', 'Attributes':
{'Height': 165, 'Weight': 68}}
print("\nSemi-structured Data:\n", semi_structured_data)
```

Structured Data:

	ID	Name	Age
0	1	Alice	25
1	2	Bob	30
2	3	Charlie	35

Unstructured Data:

This is an example of unstructured data. It can be a piece of text,  
an image, or a video file.

Semi-structured Data:           {'ID': 1, 'Name': 'Alice', 'Attributes':  
'Height': 165, 'Weight':

## D]

```

from cryptography.fernet import Fernet key =
Fernet.generate_key() f = Fernet(key)
token = f.encrypt(b"Raialakshmi Engineering College") token b'...'
.decrypt(token) b'Raialakshmi Engineering College' key =
Fernet.generate_key() cipher_suite = Fernet(key) plain_text =
b"Raialakshmi Engineering College." cipher_text =
cipher_suite.encrypt(plain_text) decrypted_text =
cipher_suite.decrypt(cipher_text) print("Original Data:", plain_text)
print("Encrypted Data:", cipher_text) print("Decrypted Data:",
decrypted_text)

Original Data: b'Raialakshmi Engineering College.'
Encrypted Data: b'gAAAAABolBkq8OPVjqIo662CR3sV8YryaRBeq-6vsuG-
FyeHltJZePo 537 IUtW3ALng5dvaGzFo5uW23q- hDEwDOVwlwzrGBiOC CleO6dyfujpyEn-
TQnKRvI0mwCCiVnEghUdgV' Decrypted Data: b'Raialakshmi Engineering College.'

```

## Exercise 2

```

import pandas as pd import numpy as np import
matplotlib.pyplot as plt df =
pd.read_csv('E:/sales_data.csv') print(df.head())
print(df.isnull().sum())
df['Sales'].fillna(df['Sales'].mean(), inplace=True)
df.dropna(subset=['Product', 'Quantity', 'Region'], inplace=True) print(df.describe())
product_summary = df.groupby('Product').agg({
    'Sales': 'sum',
    'Quantity': 'sum' }).reset_index() print(product_summary)

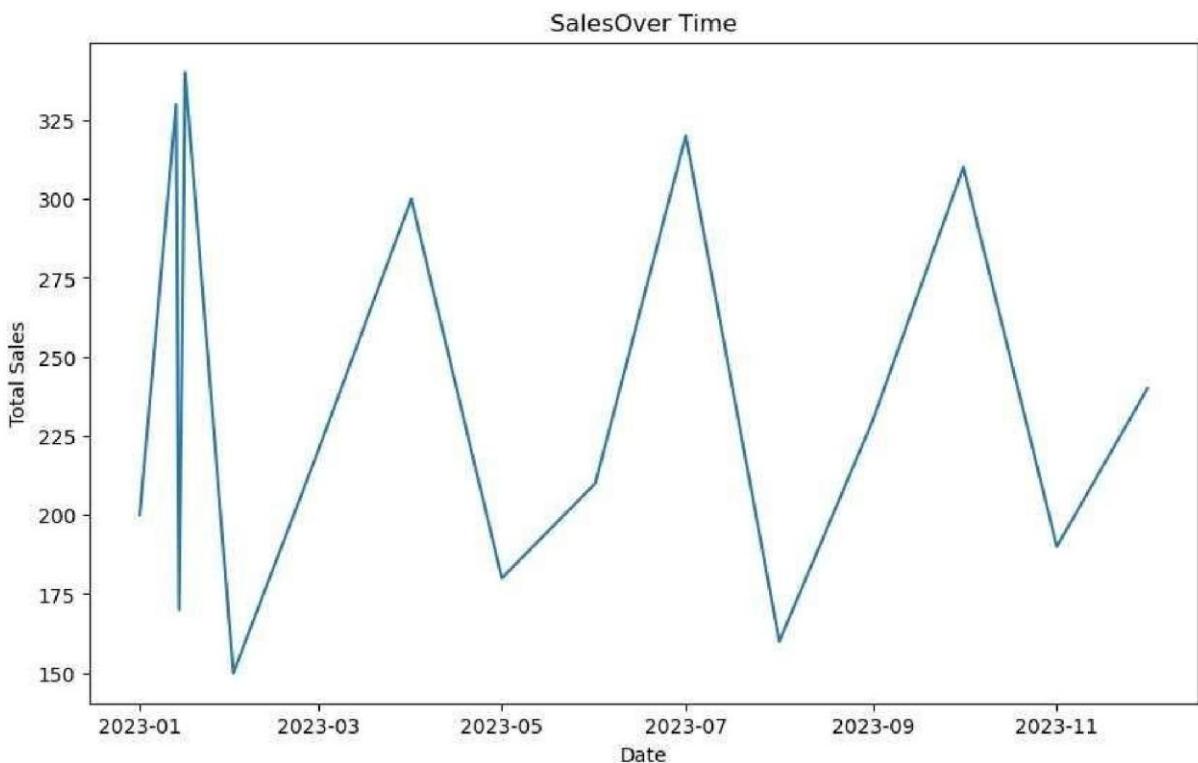
      Date Product Sales Quantity Region
0  01-01-2023  Product A   200       4   North
1  02-01-2023  Product B   150       3   South
2  03-01-2023  Product A   220       5   North

```

```
3 04-01-2023 Product C 300      6 East
4 05-01-2023 Product B 180      4 West
Date      0
Product    0
Sales      0
Quantity   0 Region    0 dtype:
int64
   Sales  Quantity count 16.000000
16.000000 mean 237.500000 5.375000 std
64.031242 1.746425 min
150.000000 3.000000
25% 187.500000 4.000000
50% 225.000000 5.500000 75%
302.500000 7.000000 max 340.000000
8.000000
Product Sales Quantity
0 Product A 1350      33
1 Product B  850      17 2 Product C 1600      36

plt.figure(figsize=(10, 6)) plt.bar(product_summary['Product'], product_summary['Sales'])
plt.xlabel('Product') plt.ylabel('Total Sales') plt.title('Total Sales by Product') plt.show()
df['Date'] = pd.to_datetime(df['Date'])
sales_over_time = df.groupby('Date').agg({'Sales':
'sum'}).reset_index()
```

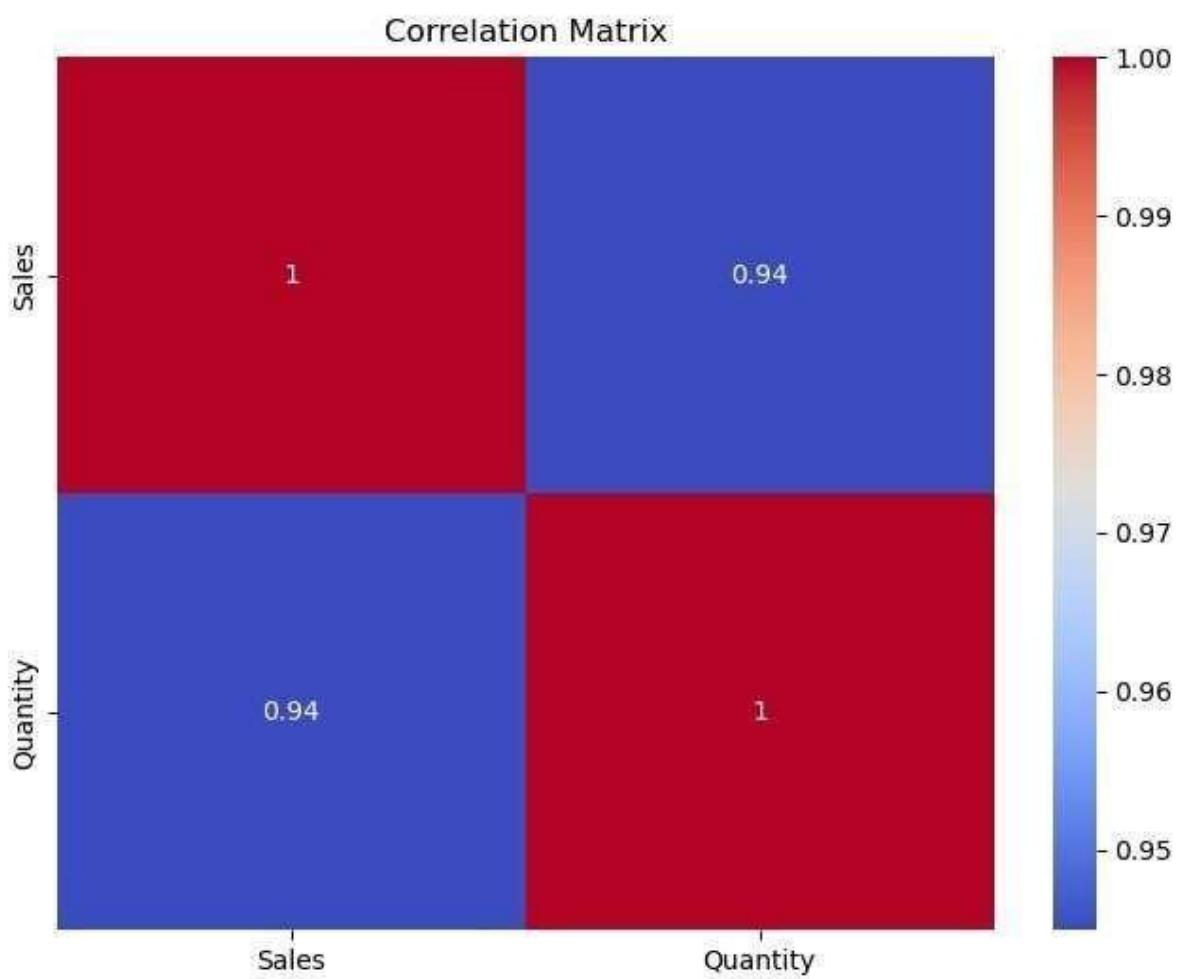
```
plt.figure(figsize=(10, 6)) plt.plot(sales_over_time['Date'],sales_over_time['Sales'])  
plt.xlabel('Date') plt.ylabel('Total Sales') plt.title('SalesOver Time') plt.show() pivot_table  
= df.pivot_table(values='Sales', index='Region', columns='Product', aggfunc=np.sum,  
fill_value=0) print(pivot_table)  
correlation_matrix = df.corr() print(correlation_matrix) import seaborn as sns plt.figure(figsize=(8,  
6)) sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm') plt.title('Correlation Matrix')  
plt.show()
```



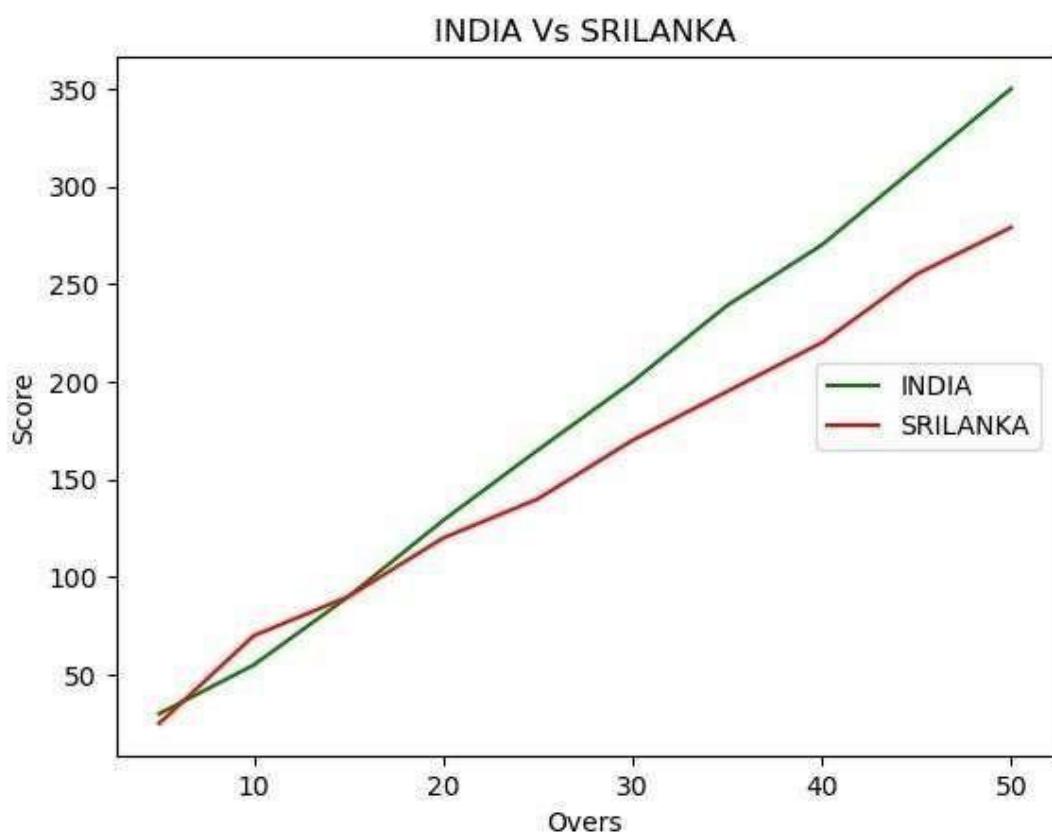
Product Region	Product A	Product B	Product C
East	0	0	160
North	1350	0	0
South	0	480	0
West	0	370	0
	Sales	Quantity	
Sales	1.000000	0.944922	
Quantity	0.944922	1.000000	

```
C:\Users\REC\AppData\Local\Temp\ipykernel_7888\240701101.py:18:
FutureWarning: The default value of numeric_only in DataFrame.corr is
deprecated. In a future version, it will default to False. Select only
valid columns or specify the value of numeric_only to silence this
warning.
```

```
correlation_matrix = df.corr()
```



```
Exercise 3: A] import
matplotlib.pyplot as cricket
Overs=list(range(5,51,5))
Indian_Score=[30,55,90,129,165,200,239,270,310,350]
Srilankan_Score=[25,70,90,120,140,170,195,220,255,279] cricket.title("INDIA Vs
SRILANKA") cricket.xlabel("Overs") cricket.ylabel("Score") cricket.legend()
cricket.plot(Overs,Indian_Score,color="green",label="INDIA")
cricket.plot(Overs,Srilankan_Score,color="red",label="SRILANKA") cricket.legend(loc="center
right")
```



B]

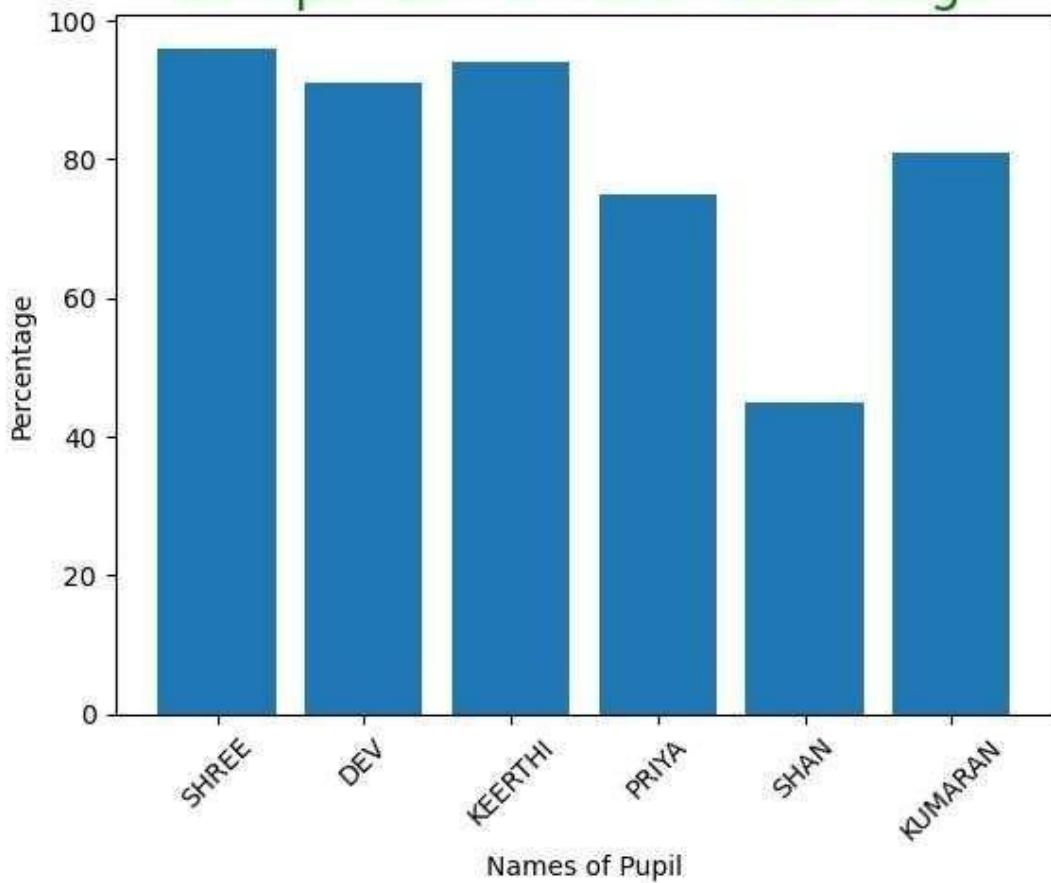
```
Names = ['SHREE', 'DEV', 'KEERTHI', 'PRIYA', 'SHAN', 'KUMARAN'] xaxis = np.arange(len(Names))
Percentage_hsc = [96, 91, 94, 75, 45, 81] hscmark.bar(Names, Percentage_hsc) hscmark.xticks(xaxis,
Names, rotation=45) hscmark.xlabel("Names of Pupil") hscmark.ylabel("Percentage")
hscmark.title("Comparison of HSC Percentage", fontsize=20, color="green") hscmark.show()
```

C] import matplotlib.pyplot as election labels = ['CANDIDATE 1', 'CANDIDATE 2',  
'CANDIDATE 3',  
'CANDIDATE 4'] Votes = [315, 130, 245, 210]

```
colors = ['green', 'yellow', 'red', 'orange'] explode = (0.2, 0, 0, 0) election.pie(Votes, labels=labels,  

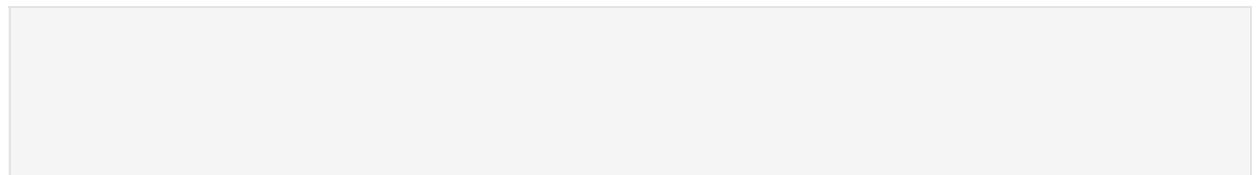
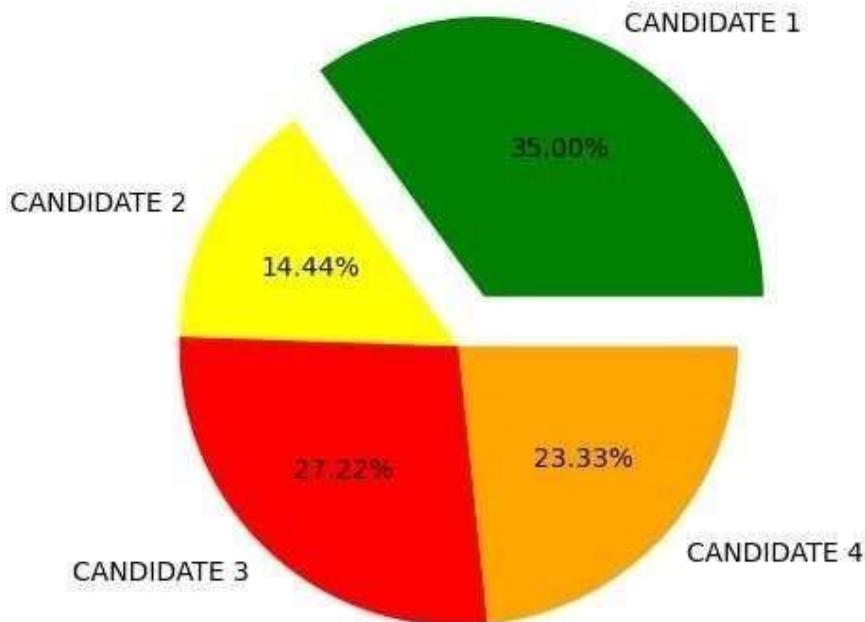
colors=colors, explode=explode, autopct='%0.2f%%')
```

# Comparison of HSC Percentage



```
election.title('Election Results') election.show()
```

### Election Results



```
import pandas as pd df=pd.read_csv("E:\diabetes.csv")
print(df.head()) print(df.info()) print(df.describe())
import matplotlib.pyplot
as plt import seaborn as sns
df.hist(bins=50, figsize=(20,15)) plt.show() sns.pairplot(df) plt.show()

Pregnancies Glucose BloodPressure SkinThickness Insulin BMI \

```

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI
1	85	66	29	0	26.6
6	148	72	35	0	33.6

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import gutenberg nltk.download('gutenberg') nltk.download('punkt')
sample = gutenberg.raw("austen-emma.txt") token = word_tokenize(sample)
wlist = [] for i in range(50):
    wlist.append(token[i]) wordfreq = [wlist.count(w) for w in wlist]
print("Pairs\n" + str(list(zip(wlist, wordfreq))))
```

```
[nltk_data] Downloading package gutenberg to [nltk_data]
```

```
C:\Users\REC\AppData\Roaming\nltk_data...
```

```
[nltk_data] Package gutenberg is already up-to-date!
```

```
[nltk_data] Downloading package punkt to [nltk_data]
```

```
C:\Users\REC\AppData\Roaming\nltk_data..
```

```
[nltk_data] Package punkt is already up-to-date!
```

Pairs

```
[('I', 1), ('Emma', 2), ('by', 1), ('Jane', 1), ('Austen', 1),
('1816', 1), (']', 1), ('VOLUME', 1), ('T', 2), ('CHAPTER', 1), ('T',
2), ('Emma', 2), ('Woodhouse', 1), (';', 5), ('handsome', 1), (';', 5),
('clever', 1), (';', 5), ('and', 3), ('rich', 1), (';',
5),
('with', 2), ('a', 1), ('comfortable', 1), ('home', 1), ('and', 3),
('happy', 1), ('disposition', 1), (';', 5), ('seemed', 1), ('to', 1),
('unite', 1), ('some', 1), ('of', 2), ('the', 2), ('best', 1),
('blessings', 1), ('of', 2), ('existence', 1), (';', 1), ('and', 3),
('had', 1), ('lived', 1), ('nearly', 1), ('twenty-one', 1), ('years', 1),
('in', 1), ('the', 2), ('world', 1), ('with', 2)]
```

Exercise 5:

Exercise 4:

2	8	183	64	0	0	23.3
3	1	89	66	23	94	28.1
4	0	137	40	35		
	168	43.1				

DiabetesPedigreeFunction Age Outcome

0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 768 entries, 0 to 767

Data columns (total 9 columns):

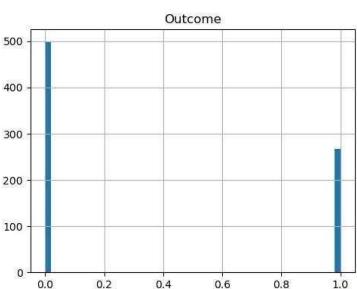
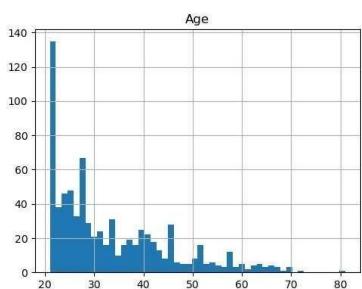
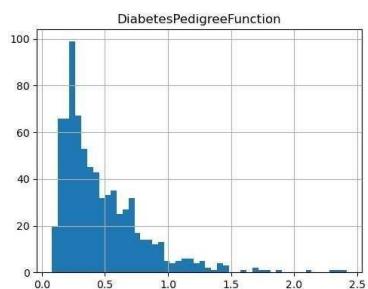
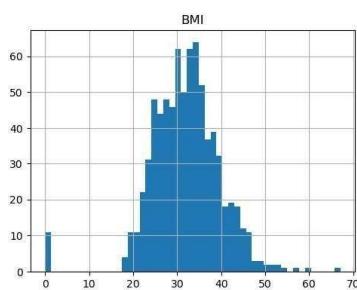
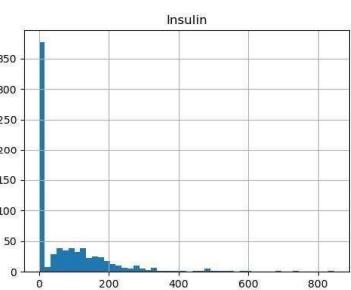
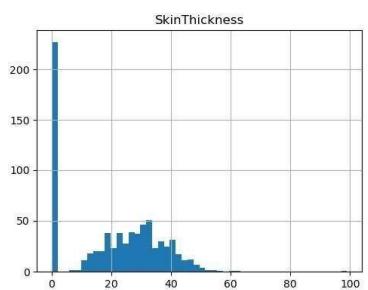
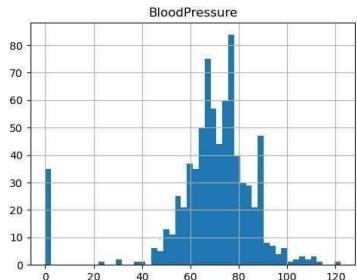
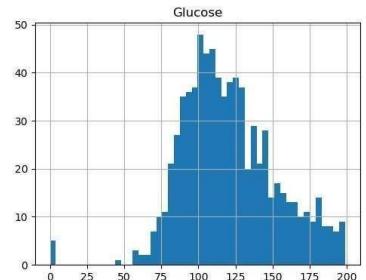
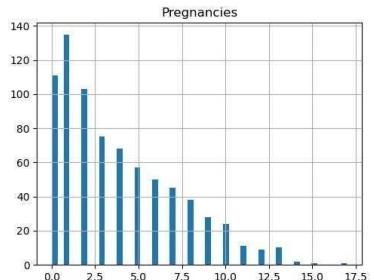
#	Column	Non-Null Count	Dtype
0	Pregnancies	768	non-null int64
1	Glucose	768	non-null int64
2	BloodPressure	768	non-null int64
	SkinThickness	768	non-null int64
3	Insulin	768	non-null int64
5	BMI	768	non-null float64
6	DiabetesPedigreeFunction	768	non-null float64
7	Age	768	non-null int64
	Outcome	768	non-null int64 dtypes: float64(2), int64(7)

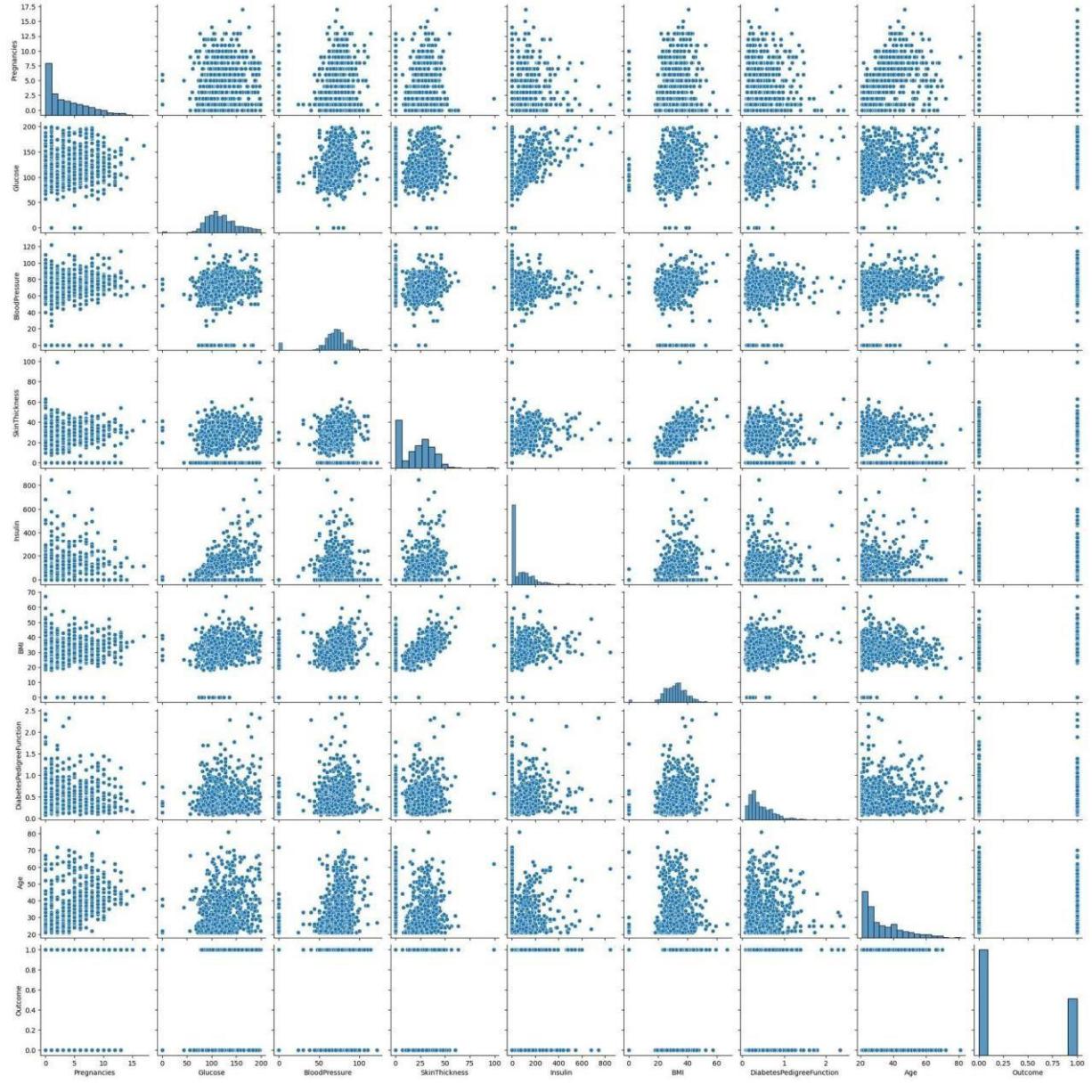
memory usage: 54.1 KB

None

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	\count
768.000000	768.000000	768.000000	768.000000			
768.000000	mean	3.845052	120.894531			
69.105469	20.536458					
79.799479	std	3.369578	31.972618	19.355807	15.952218	
115.244002	min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	

BMI	DiabetesPedigreeFunction		Age	Outcome	count	768.000000
768.000000						
768.000000	768.000000	mean	31.992578		0.471876	33.240885
7.884160						0.348958 std
0.331329	11.760232	0.476951	min	0.000000		0.078000 21.000000
						0.000000
25%	27.300000		0.243750	24.000000	0.000000	
50%	32.000000		0.372500	29.000000	0.000000	
75%	36.600000		0.626250	41.000000	1.000000	
max	17.000000	199.000000	122.000000	99.000000		
846.000000						
max	67.100000		2.420000	81.000000	1.000000	





**Exercise 6:**

```
import numpy as np
import pandas as pd
df=pd.read_csv("E:\Hotel_Dataset.csv")
df.duplicated()
0    False
1    False
2    False
3    False
4    False
5    False
6    False
7    False
8    False
9    True
10   False
dtype: bool
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11 entries, 0 to 10
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   CustomerID      11 non-null     int64  
 1   Age_Group       11 non-null     object  
 2   Rating(1-5)     11 non-null     int64  
 3   Hotel            11 non-null     object  
 4   FoodPreference   11 non-null     object  
 5   Bill             11 non-null     int64  
 6   NoOfPax          11 non-null     int64  
 7   EstimatedSalary  11 non-null     int64   8   Age_Group.1      11
non-null      object  
dtypes: int64(5), object(4)
memory usage: 920.0+ bytes

df.drop_duplicates(inplace=True)
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
1	1	20-25	4	Ibis	veg	1300
\						
0						
1	2	30-35	5	LemonTree	Non-Veg	2000
2	3	25-30	6	RedFox	Veg	1322
3	4	20-25	-1	LemonTree	Veg	1234
4	5	35+	3	Ibis	Vegetarian	989

5

				6	35+	3	Ibys	Non-Veg	1909
6	7	35+	4	RedFox	Vegetarian	1000			
7	8	20-25	7	LemonTree	Veg	2999			

8	9	25-30	2	Ibis	Non-Veg	4510	5
---	---	-------	---	------	---------	------	---

10	30-35	5	RedFox	non-Veg	-6755	40000
----	-------	---	--------	---------	-------	-------

NoOfPax EstimatedSalary Age\_Group.1 0 2

20-25	1 3	59000	30-35	2 2	30000	25-30
3	2	120000	20-25	4	2	45000
		35+				
5	2	122220		35+		
6	-1	21122		35+		
7	-10	345673		20-		
8	3	-99999		25-30		
10	4	87777		30-35		

```
len(df) index=np.array(list(range(0,len(df))))
10 df.set_index(index,inplace=True) index
```

array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

df

CustomerID	Age_Group	Rating(1-5)	Hotel F
------------	-----------	-------------	---------

NoOfPax \

1	20-25	4	Ibis	veg	13
---	-------	---	------	-----	----

2	30-35	5	LemonTree	Non-	
---	-------	---	-----------	------	--

3	25-30	6	RedFox	Veg	
---	-------	---	--------	-----	--

0

4	20-25	-1	LemonTree	Ve	
---	-------	----	-----------	----	--

V

5	35+	3	Ibis	Vegetarian	
---	-----	---	------	------------	--

g

6	35+	3	Ibys	Non-Veg	
---	-----	---	------	---------	--

9

7	35+	4	RedFox	Vegetaria	
---	-----	---	--------	-----------	--

1

0  
2  
1  
3

2

2

3

2

4

2

5

2

6

```

-1
7     8     20-25          7   LemonTree      Veg    2999
-10
8     9     25-30          2   Ibis        Non-Veg   3456
3
9     10    30-35          5   RedFox       non-Veg -6755
4

```

	EstimatedSalary	Age_Group.1
0	40000	20-25
1	59000	30-35
2	30000	25-30
3	120000	20-25
4	45000	35+
5	122220	35+
6	21122	35+
7	345673	20-25
8	-99999	25-30  9
		87777    30-35

```
df.drop(['Age_Group.1'],axis=1,inplace=True)
df
```

NoOfPax \ CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1	20-25	4	Ibis	veg 1300
1	2	30-35	5	LemonTree	Non-Veg 2000
2	3	25-30	6	RedFox	Veg 1322
3	4	20-25	-1	LemonTree	Veg 1234
4	5	35+	3	Ibis	Vegetarian 989
5	6	35+	3	Ibys	Non-Veg 1909
6	7	35+	4	RedFox	Vegetarian 1000
7	8	20-25	7	LemonTree	Veg 2999
8	9	25-30	2	Ibis	Non-Veg 3456
9	10	30-35	5	RedFox	non-Veg -6755

	EstimatedSalary
0	40000
1	59000
2	30000
3	120000
4	45000
5	122220
6	21122
7	345673
8	-99999  9
	87777

```
df.CustomerID.loc[df.CustomerID<0]=np.nan
df.Bill.loc[df.Bill<0]=np.nan
df.EstimatedSalary.loc[df.EstimatedSalary<0]=np.nan
df
```

C:\Users\REC\AppData\Local\Temp \ipykernel\_4252\240701101.py:1:

SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

df.CustomerID.loc[df.CustomerID<0]=np.nan

C:\Users\REC\AppData\Local\Temp\ipykernel\_4252\240701101.PY:2:

SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:

[https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy) df.Bill.loc[df.Bill<0]=np.nan

C:\Users\REC\AppData\Local\Temp\ipykernel\_4252\240701101.py:3:

SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:

[https://pandas.pydata.org/pandasdocs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandasdocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

df.EstimatedSalary.loc[df.EstimatedSalary<0]=np.nan

CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
1.0	20-25	4	Ibis	veg	1300.0

2.0	30-35	5	LemonTree	Non-Veg	2000.0
-----	-------	---	-----------	---------	--------

0

1

2	3.0	25-30	6	RedFox	Veg	1322.0
---	-----	-------	---	--------	-----	--------

3	4.0	20-25	-1	LemonTree	Veg	1234.0
---	-----	-------	----	-----------	-----	--------

4	5.0	35+	3	Ibis	Vegetarian	989.0
---	-----	-----	---	------	------------	-------

5	6.0	35+	3	Ibys	Non-Veg	1909.0
---	-----	-----	---	------	---------	--------

7.0	35+	4	RedFox	Vegetarian	1000.0
-----	-----	---	--------	------------	--------

6

```

7     8.0  20-25      7 LemonTree      Veg 2999.0
8     9.0  25-30      2   Ibis    Non-Veg 3456.0
10.0 30-35       5 RedFox    non-Veg   NaN
      NoOfPax EstimatedSalary
0            2        40000.0
1            3        59000.0
2            2        30000.0
3            2       120000.0
4            2        45000.0
5            2      122220.0
6           -1        21122.0
7          -10      345673.0
8            3        NaN
9            4       87777.0

```

df['NoOfPax'].loc[(df['NoOfPax']<1) | (df['NoOfPax']>20)]=np.nan df

C:\Users\REC\AppData\Local\Temp\ipykernel\_4252\2129877948.py:1:

SettingWithCopyWarning

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: [https://pandas.pydata.org/pandasdocs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandasdocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

df['NoOfPax'].loc[(df['NoOfPax']<1) | (df['NoOfPax']>20)]=np.nan

CustomerID Age\_Group Rating(1-5) Hotel FoodPreference Bill

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
1.0	20-25	4	Ibis	veg	1300.0	
0						
1	2.0	30-35	5	LemonTree	Non-Veg	2000.0
2	3.0	25-30	6	RedFox	Veg	1322.0
3	4.0	20-25	-1	LemonTree	Veg	1234.0
4	5.0	35+	3	Ibis	Vegetarian	989.0
5	6.0	35+	3	Ibys	Non-Veg	1909.0
6	7.0	35+	4	RedFox	Vegetarian	1000.0
7	8.0	20-25	7	LemonTree	Veg	2999.0

```
8      9.0  25-30      2   Ibis    Non-Veg  3456.0
9     10.0  30-35      5  RedFox    non-Veg   NaN
NoOfPax      EstimatedSalary  0   2.0
40000.0
1      3.0      59000.0
2      2.0      30000.0
3      2.0      120000.0
4array(['Ibis', 'LemonTree', 'RedFox', 'Ibys'], dtype=object)
5      2.0      122220.0
6      NaN      21122.0  7  NaN
345673.0 8 3.0      NaN
9      4.0      87777.0
df.Age_Group.unique()
array(['20-25', '30-35', '25-30', '35+'], dtype=object)
```

```
df.Hotel.unique()
df.Hotel.replace(['Ibys'], 'Ibis', inplace=True) df.FoodPreference.unique
<bound method Series.unique of 0      veg
1      Non-Veg
2      Veg
3      Veg
4      Vegetarian
5      Non-Veg
6      Vegetarian
7      Veg
8      Non-Veg
9      non-Veg
Name: FoodPreference, dtype: object>
df.FoodPreference.replace(['Vegetarian', 'veg'], 'Veg', inplace=True) df.FoodPreference.replace(['non-Veg'], 'Non-Veg', inplace=True)

df.EstimatedSalary.fillna(round(df.EstimatedSalary.mean()), inplace=True)
df.NoOfPax.fillna(round(df.NoOfPax.median()), inplace=True) df['Rating(1-5)'].fillna(round(df['Rating(1-5)'].median()), inplace=True)
df.Bill.fillna(round(df.Bill.mean()), inplace=True) df
```

```
\ CustomerID Age_Group  Rating(1-5)  Hotel FoodPreference  Bill
(
1.0  20-25      4   Ibis      Veg  1300.0
```

1  
2  
3  
4  
5  
6

7 8 0 1 2 3

```
NoOfPax EstimatedSalary  
2.0      40000.0 3.0  
59000.0 2.0      30000.0 2.0  
120000.0 2.0      45000.0  
2.0      122220.0     2.0  
21122.0  
2.0      345673.0  
3.0      96755.0  
4.0      87777.0
```

```
4          i)].median()), inplace=True) df  
5  
6  
7  
8  
9
```

```
f['Rating(1-5)'].fillna(round(df['Rating(1-
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0						
1		1.0	20-25	4	Ibis	Veg 1300.0
3		4.0	20-25	-1	LemonTree	Veg 1234.
4		5.0	35+	3	Ibis	Veg 989.0

5 6.0 35+ 3 Ibis Non-Veg 1909.

6 7.0 35+ 4 RedFox Veg 1000.0  
7 8.0 20-25 7 LemonTree Veg 2999.0  
8 9.0 25-30 2 Ibis Non-Veg 3456.0

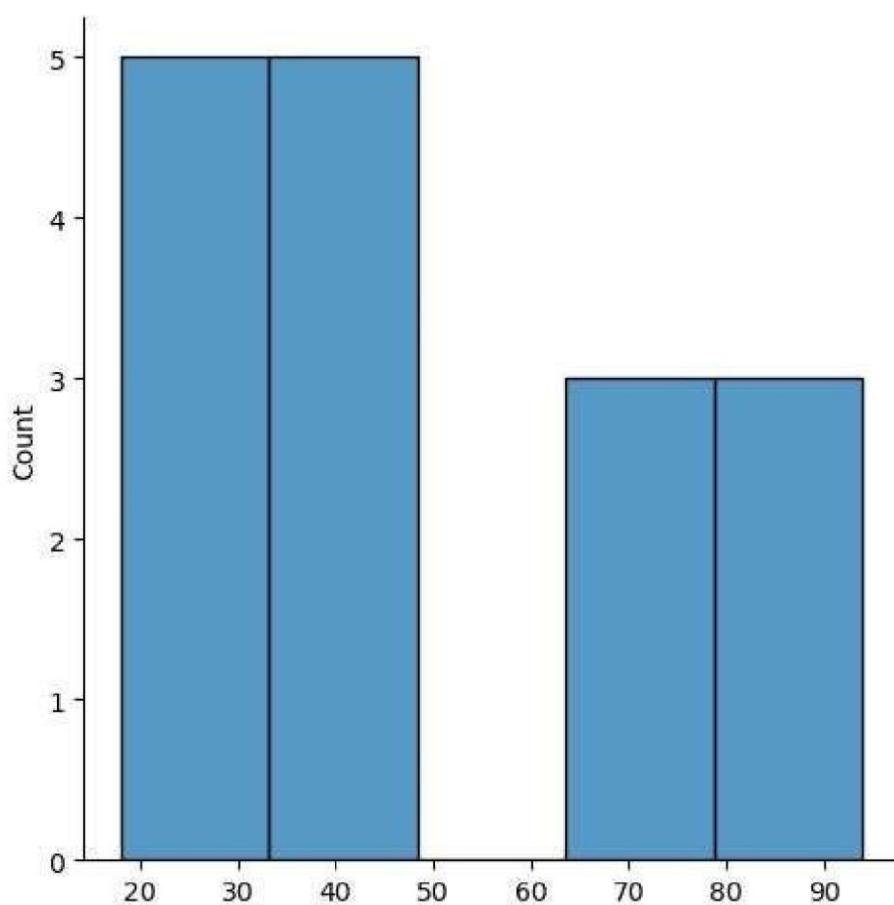
9 10.0 30-35 5 RedFox Non-Veg 1801.0

NoOfPax EstimatedSalary 0 2.0  
40000.0

1 3.0 59000.0  
2 2.0 30000.0  
3 2.0 120000.0  
4 2.0 45000.0  
5 2.0 122220.0  
6 2.0 21122.0 7 2.0

## Exercise 7:

```
import numpy as np array=np.random.randint(1,100,16) # randomly generate 16 numbers between  
1 to 100 array  
array([35, 18, 94, 35, 71, 83, 85, 21, 74, 37, 29, 27, 74, 45, 27,  
48])  
  
array.mean()  
50.1875 np.percentile(array,25)  
28.5 np.percentile(array,50)  
41.0 np.percentile(array,75)  
74.0 np.percentile(array,100)  
94.0 def outDetection(array):  
    sorted(array)  
    Q1,Q3=np.percentile(array,[25,75]) IQR=Q3-Q1 lr=Q1-(1.5*IQR) ur=Q3+(1.5*IQR)  
    return lr,ur lr,ur,outDetection(array) lr,ur  
(-39.75, 142.25)  
import seaborn as sns %matplotlib inline sns.displot(array)  
<seaborn.axisgrid.FacetGrid at 0x1c7ed3de080>
```

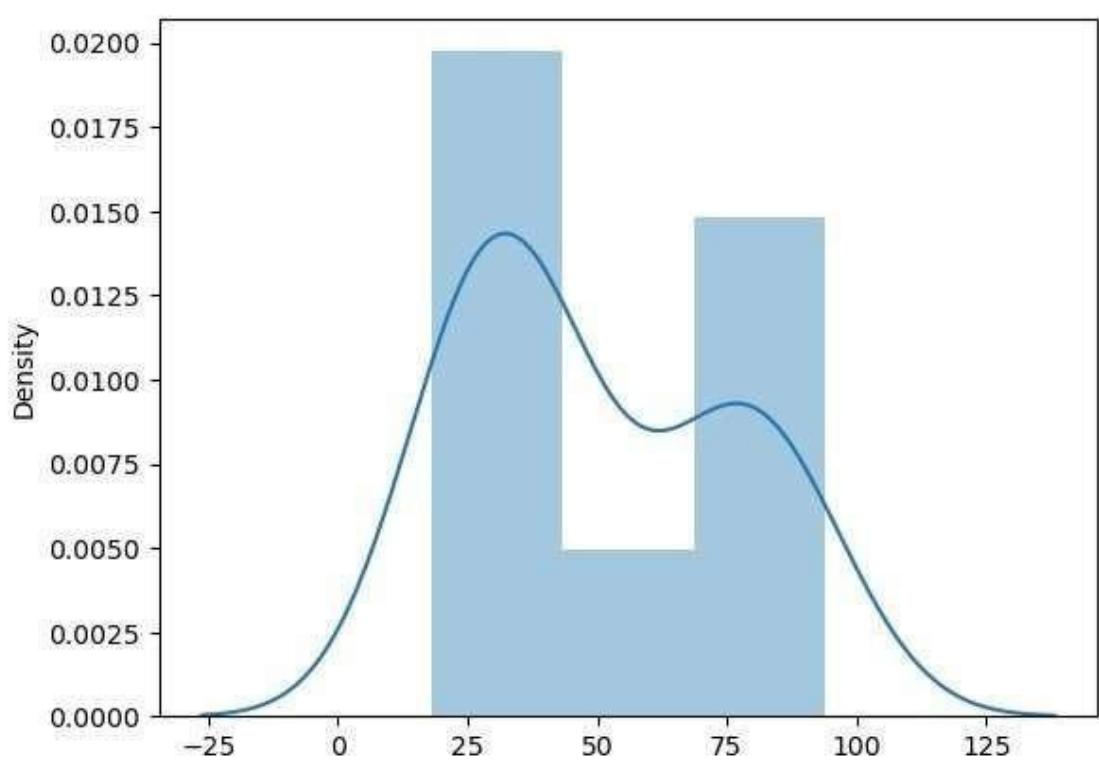


```
sns.distplot(array)
C:\Users\REC\AppData\Local\Temp\ipykernel_5860\240701144 .py:1:
UserWarning: `distplot` is a deprecated function and will be removed in
seaborn
v0.14.0.
```

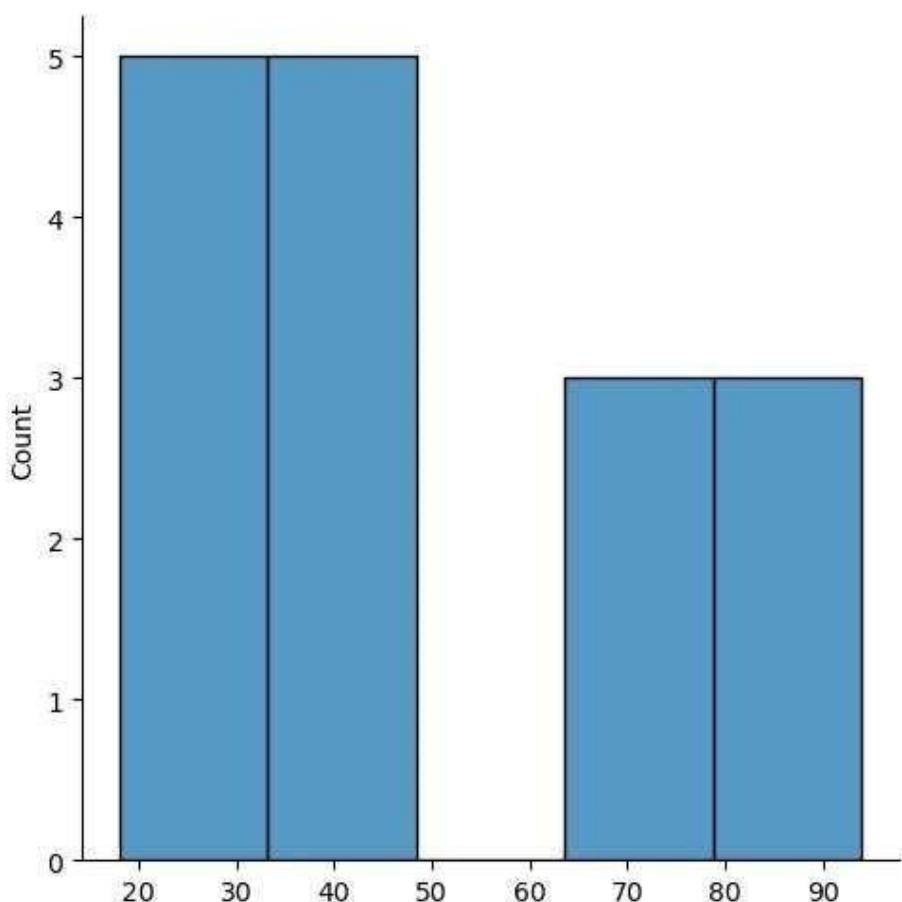
Please adapt your code to use either 'displot' (a figure-level function with similar flexibility) or 'histplot' (an axes-level function for histograms).

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

```
sns.distplot(array)
<Axes: ylabel='Density' >
```



```
new_array=array[(array>lr) & (array<ur)] new_array
array([35, 18, 94, 35, 71, 83, 85, 21, 74, 37, 29, 27, 74, 45, 27,
48])
sns.displot(new_array)
<seaborn.axisgrid.FacetGrid at 0x1c7f392ec80>
```



```
lr1,ur1=outDetection(new_array) lr1,ur1
(-39.75, 142.25)
final_array=new_array[(new_array>lr1) & (new_array<ur1)] final_array
array([35, 18, 94, 35, 71, 83, 85, 21, 74, 37, 29, 27, 74, 45, 27,
48])
sns.distplot(final_array)
```

C:\Users\REC\AppData\Local\Temp\ipykernel\_5860\240701804.py:1:

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

## Exercise 8:

```
import numpy as np import pandas as pd
df=pd.read_csv('E:/pre_process_datasample.csv') df
   Country Age Salary Purchased
0      France 44.0 72000.0    No
1      Spain 27.0 48000.0   Yes
2     Germany 30.0 54000.0    No
3      Spain 38.0 61000.0    No
4     Germany 40.0    NaN   Yes 5 France
          35.0
58000.0    Yes
6 Spain    NaN 52000.0    No 7
France 48.0 79000.0   Yes
8     Germany 50.0 83000.0    No
9     France 37.0 67000.0   Yes
df.head()
```

```
   Country Age Salary Purchased
0      France 44.0 72000.0    No
1      Spain 27.0 48000.0   Yes
2     Germany 30.0 54000.0    No
3      Spain 38.0 61000.0    No
4     Germany 40.0    NaN   Yes
df.Country.fillna(df.Country.mode()[0],inplace=True) features=df.iloc[:, :-1].values label=df.iloc[:, :-1].values
```

```
SimpleImputer()
from sklearn.impute import SimpleImputer
age=SimpleImputer(strategy="mean",missing_values=np.nan)
Salary=SimpleImputer(strategy="mean",missing_values=np.nan) age.fit(features[:,[1]])
Salary.fit(features[:,[2]])
```

```
SimpleImputer()
SimpleImputer()
```

```
SimpleImputer()
features[:,[1]]=age.transform(features[:,[1]])
features[:,[2]]=Salary.transform(features[:,[2]]) features
array(['France', 44.0, 72000.0],
['Spain', 27.0, 48000.0],
```

```
['Germany', 30.0, 54000.0],  
['Spain', 38.0, 61000.0],  
['Germany', 40.0, 63777.7777777778],  
['France', 35.0, 58000.0],  
['Spain', 38.77777777777778, 52000.0],  
['France', 48.0, 79000.0],  
['Germany', 50.0, 83000.0],  
['France', 37.0, 67000.0]], dtype=object)  
  
from sklearn.preprocessing import OneHotEncoder oh =  
OneHotEncoder(sparse_output=False)  
Country=oh.fit_transform(features[:,[0]]) Country  
  
array([[1., 0., 0.],  
[0., 0., 1.],  
[0., 1., 0.],  
[0., 0., 1.],  
[0., 1., 0.],  
[1., 0., 0.],  
[0., 0., 1.],  
[1., 0., 0.],  
[0., 1., 0.], [1., 0., 0.]])  
  
array([[1.0, 0.0, 0.0, 44.0, 72000.0]  
[0.0, 0.0, 1.0, 27.0, 48000.0],  
[0.0, 1.0, 0.0, 30.0, 54000.0],  
[0.0, 0.0, 1.0, 38.0, 61000.0],  
[0.0, 1.0, 0.0, 40.0, 63777.7777777778],  
[1.0, 0.0, 0.0, 35.0, 58000.0],  
[0.0, 0.0, 1.0, 38.7777777777778, 52000.0],  
[1.0, 0.0, 0.0, 48.0, 79000.0],  
[0.0, 1.0, 0.0, 50.0, 83000.0],  
[1.0, 0.0, 0.0, 37.0, 67000.0]], dtype=object)  
  
final_set=np.concatenate((Country,features[:,[1,2]]),axis=1) final_set from  
sklearn.preprocessing import StandardScaler sc=StandardScaler() sc.fit(final_set)  
feat_standard_scaler=sc.transform(final_set) feat_standard_scaler  
array([[ 1.22474487e+00, -6.54653671e-01, -6.54653671e-01,  
7.58874362e-01, 7.49473254e-01],  
[-8.16496581e-01, -6.54653671e-01, 1.52752523e+00,  
-1.71150388e+00, -1.43817841e+00],  
[-8.16496581e-01, 1.52752523e+00, -6.54653671e-01,  
-1.27555478e+00, -8.91265492e-01],
```

```
-1.13023841e-01, -2.53200424e-01],  
[-8.16496581e-01, 1.52752523e+00, -6.54653671e-01,  
 1.77608893e-01, 6.63219199e-16],  
[ 1.22474487e+00, -6.54653671e-01, -6.54653671e-01,           -5.48972942e-01, -5.26656882e-01],  
[-8.16496581e-01, -6.54653671e-01, 1.52752523e+00,  
 0.00000000e+00, -1.07356980e+00],  
[ 1.22474487e+00, -6.54653671e-01, -6.54653671e-01,           1.34013983e+00,  
1.38753832e+00],           [-8.16496581e-01, 1.52752523e+00, -6.54653671e-01,  
1.63077256e+00, 1.75214693e+00],           [ 1.22474487e+00, -6.54653671e-01, -6.54653671e-01,  
-2.58340208e-01, 2.93712492e-01]])
```

```
from sklearn.preprocessing import MinMaxScaler  
mms=MinMaxScaler(feature_range=(0,1)) mms.fit(final_set)  
feat_minmax_scaler=mms.transform(final_set)  
feat_minmax_scaler array([[1.      , 0.      , 0.      , 0.73913043, 0.68571429],
```

2  
752523e+00,

```

[0.          , 0.          , 0.          ],
[0.          , 0.          , 0.13043478, 0.17142857],
[0.          , 0.          , 1.          , 0.47826087, 0.37142857], [0.          , 1.          , 0.          , 0.56521739,
[1.          , 0.          , 0.          , 0.34782609, 0.28571429],
[0.          , 0.          , 1.          , 0.51207729, 0.11428571],
[1.          , 0.          , 0.          , 0.91304348, 0.88571429],
[0.          , 1.          , 0.          , 1.          , 1.          ],
[1.          , 0.          , 0.          , 0.43478261, 0.54285714]])

```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9 Data columns (total 4
columns):
 #   Column   Non-Null Count Dtype  
 --- 
 0   Country    10 non-null    object 
 1   Age         9 non-null    float64 
 2   Salary      9 non-null    float64 
 3   Purchased   10 non-null   object 
 dtypes: float64(2), object(2) memory
 usage: 448.0+ bytes df.Country.mode()

0   France
Name: Country, dtype: object
```

```
df.Country.mode()[0]
'France'
type(df.Country.mode())
pandas.core.series.Series
df.Country.fillna(df.Country.mode()[0],inplace=True) df.Age.fillna(df.Age.median(),inplace=True)
df.Salary.fillna(round(df.Salary.mean()),inplace=True) df
```

```
Country Age Salary Purchased
0 France 44.0 72000.0 No
1 Spain 27.0 48000.0 Yes
2 Germany 30.0 54000.0 No
3 Spain 38.0 61000.0 No
4 Germany 40.0 63778.0 Yes 5 France
   35.0
58000.0 Yes
6 Spain 38.0 52000.0 No
7 France 48.0 79000.0 Yes
8 Germany 50.0 83000.0 No
9 France 37.0 67000.0 Yes
```

```
pd.get_dummies(df.Country)
```

```
France Germany Spain
0 1 0 0
1 0 0 1
2 0 1 0
3 0 0 1
4 0 1 0
5 1 0 0
6 0 0 1
7 1 0 0
8 0 1 0
9 1 0 0
```

```
updated_dataset=pd.concat([pd.get_dummies(df.Country),df.iloc[:, [1,2,3]]],axis=1)
updated_dataset
```

```
France Germany Spain Age Salary Purchased
0 1 0 0 44.0 72000.0 No 1 0 0
1 27.0 48000.0 Yes
2 0 1 0 30.0 54000.0 No
3 0 0 1 38.0 61000.0 No
4 0 1 0 40.0 63778.0 Yes
5 1 0 0 35.0 58000.0 Yes
6 0 0 1 38.0 52000.0 No
```

```
7 1 0 0 48.0 79000.0 Yes
8 0 1 0 50.0 83000.0 No
9 1 0 0 37.0 67000.0 Yes
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9 Data columns (total 4 columns):
 #   Column  Non-Null Count Dtype  
 ---  --       --           --      
 0   Country    10 non-null   object 
 1   Age        10 non-null   float64 
 2   Salary      10 non-null   float64 
 3   Purchased   10 non-null   object dt float64(2), object(2)
```

	France	Germany	Spain	Age	Salary	Purchased
0	1	0	0	44.0	72000.0	
1	0	0	1	27.0	48000.0	
2	0	1	0	30.0	54000.0	
3	0	0	1	38.0	61000.0	
4	0	1	0	40.0	63778.0	
5	1	0	0	35.0	58000.0	
6	0	0	1	38.0	52000.0	
7	1	0	0	48.0	79000.0	
8	0	1	0	50.0	83000.0	
9	1	0	0	37.0	67000.0	

```
memory usage:
```

448.0+ bytes
updated_dataset.Pur
chased.replace(['No',
'Yes'],[0,
updated_dataset
1],inplace=True)
0
1
0

0
1
1
0
1

0
1

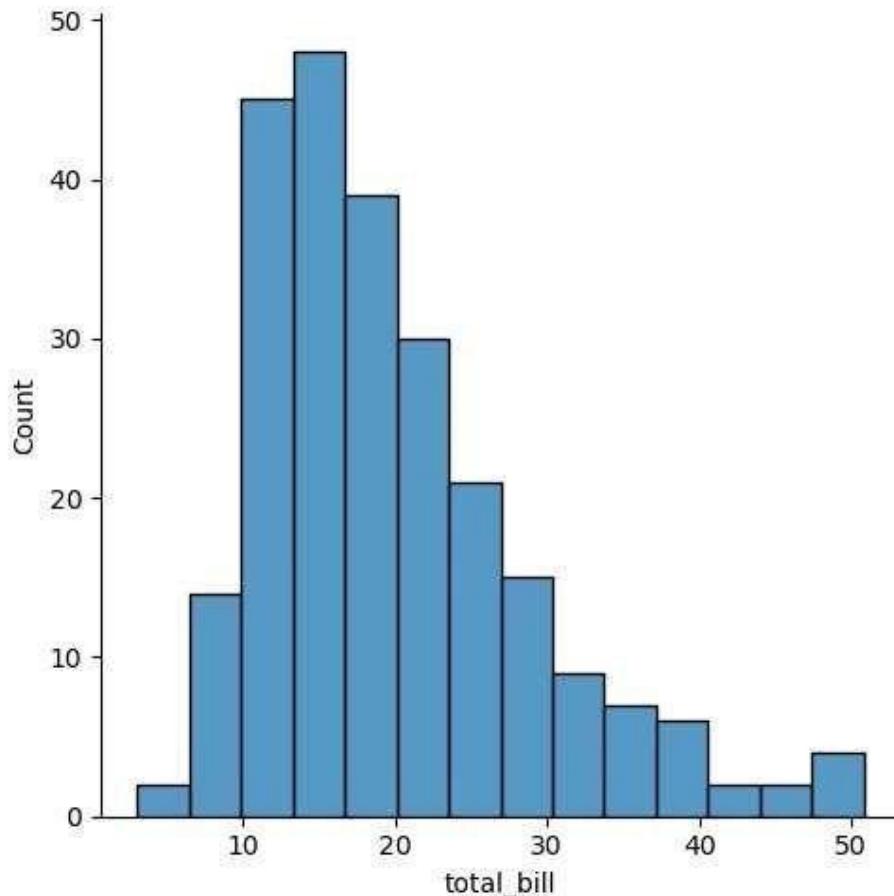
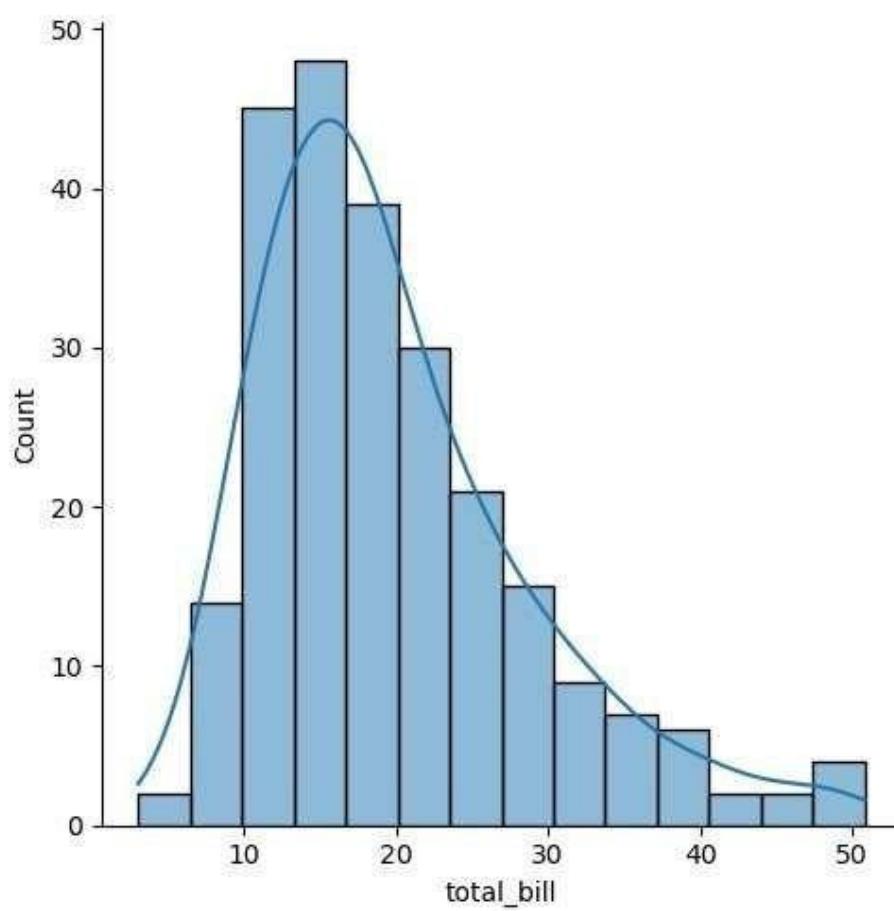
```
import seaborn as sns import pandas as pd import numpy
as np import matplotlib.pyplot as plt
```

	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					

```
%matplotlib inline tips=sns.load_dataset('tips') tips.head()
```

```
sns.displot(tips.total_bill,kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x1cbb0db2d70>
```

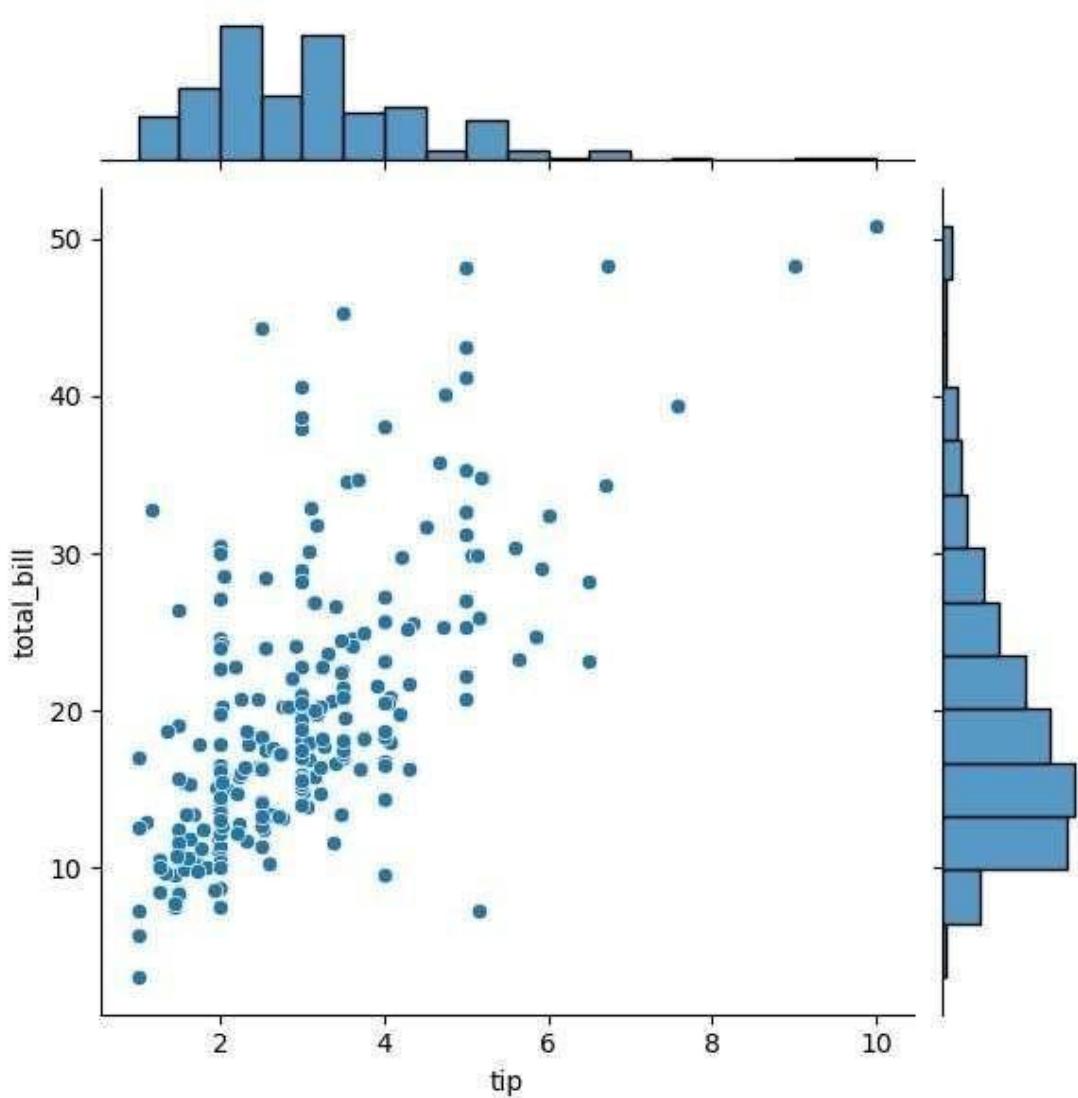


```
sns.jointplot(x=tips.tip,y=tips.total_bill)
```

```
sns.displot(tips.total_bill,kde=False)
```

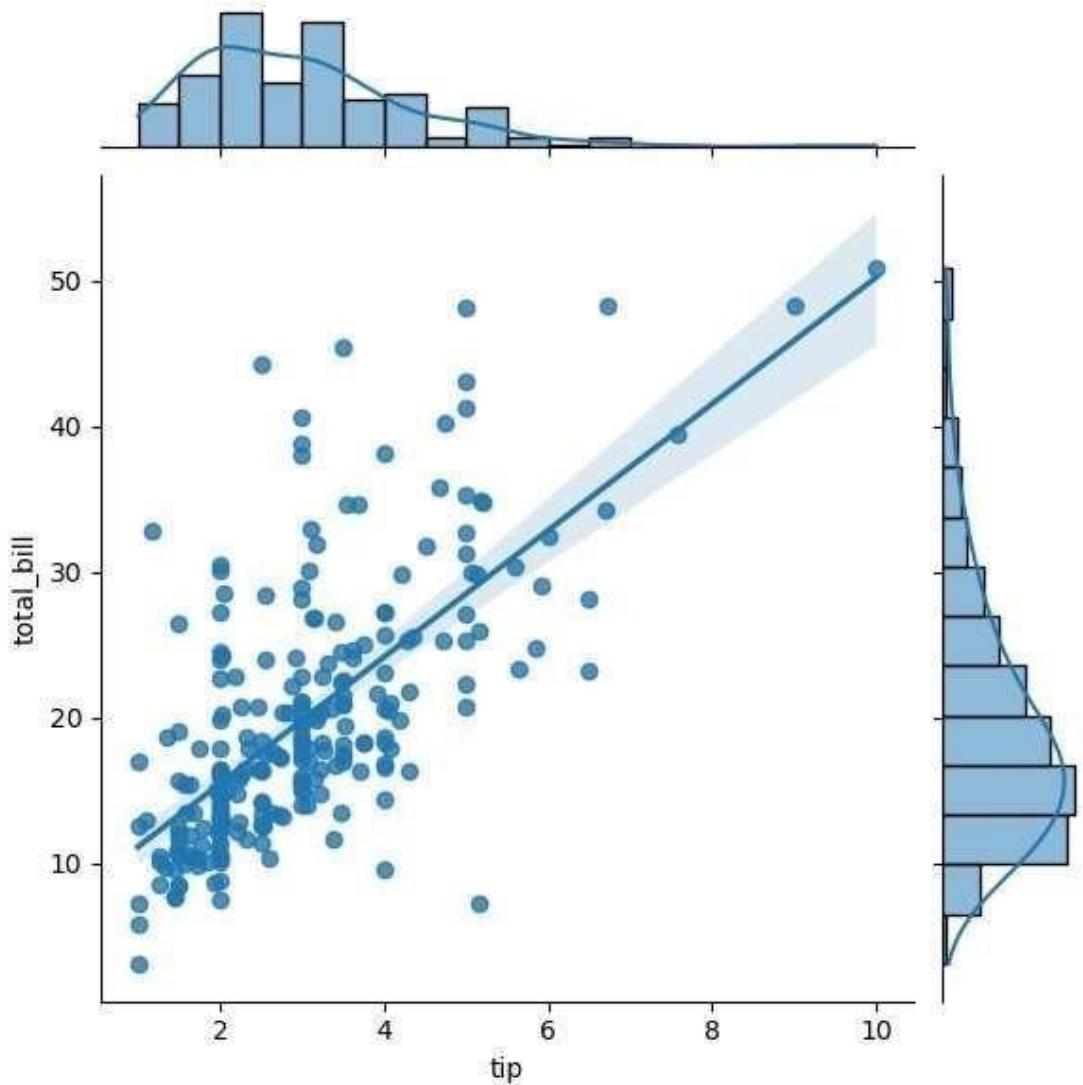
```
<seaborn.axisgrid.FacetGrid at 0x1cbb0f51510>
```

<seaborn.axisgrid.JointGrid at 0x1cbb0db3f70



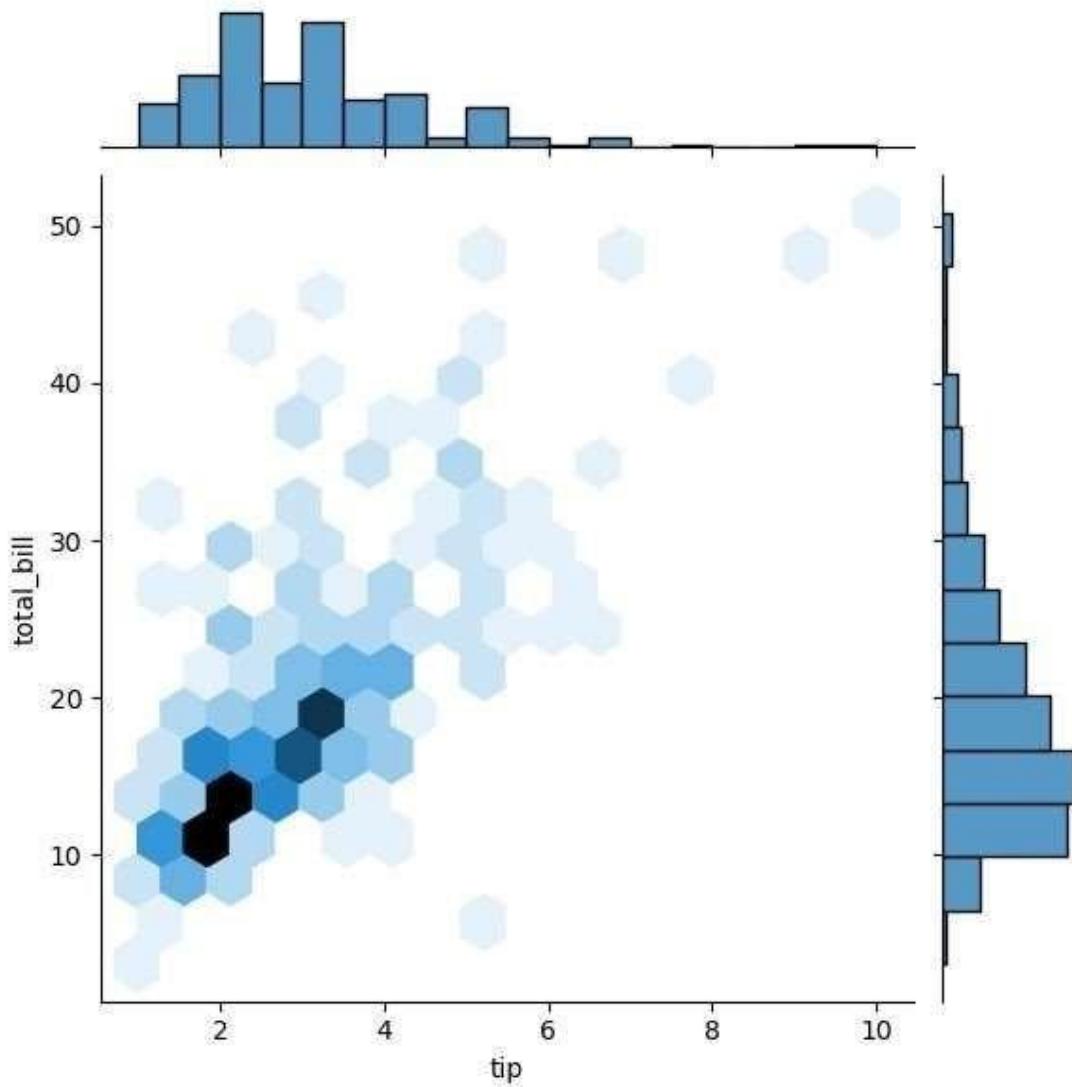
```
sns.jointplot(x=tips.tip,y=tips.total_bill,kind="reg")
```

```
<seaborn.axisgrid.JointGrid at 0x1cbb1f8da20
```



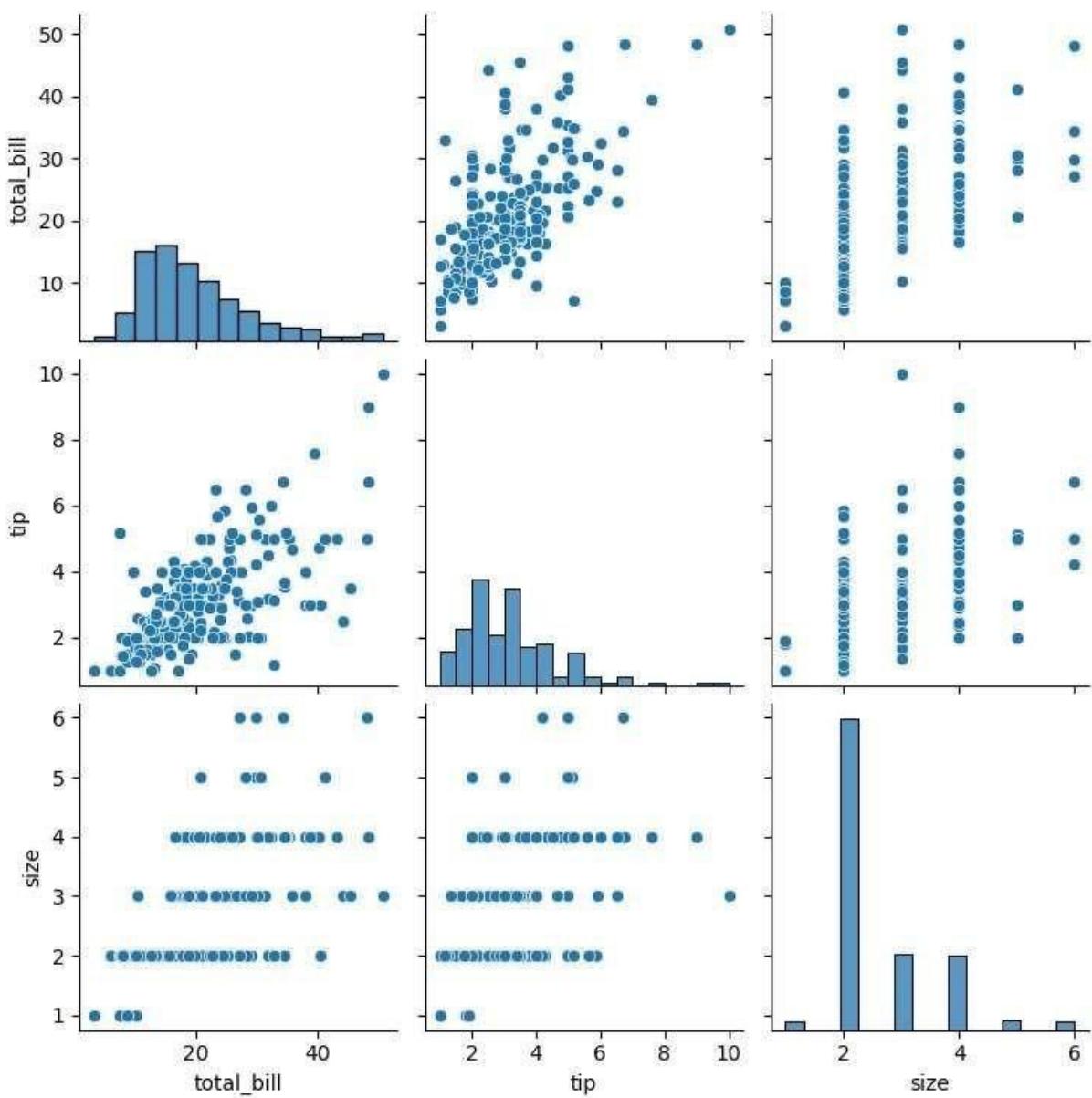
```
sns.jointplot(x=tips.tip,y=tips.total_bill,kind="hex")
```

```
<seaborn.axisgrid.JointGrid at 0x1cbb258da20
```



```
sns.pairplot(tips)
```

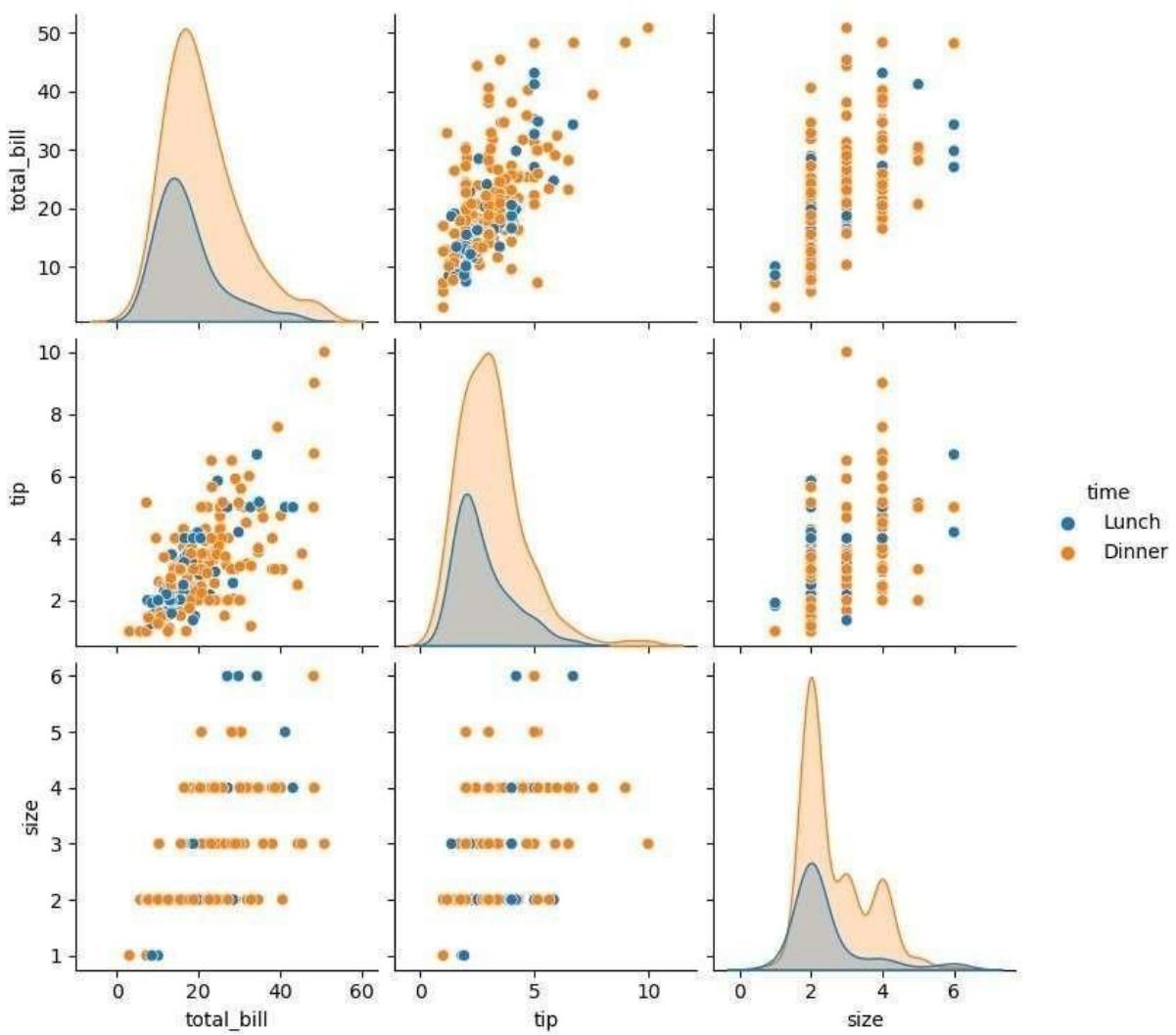
```
<seaborn.axisgrid.PairGrid at 0x1cbb391a7d0>
```



```

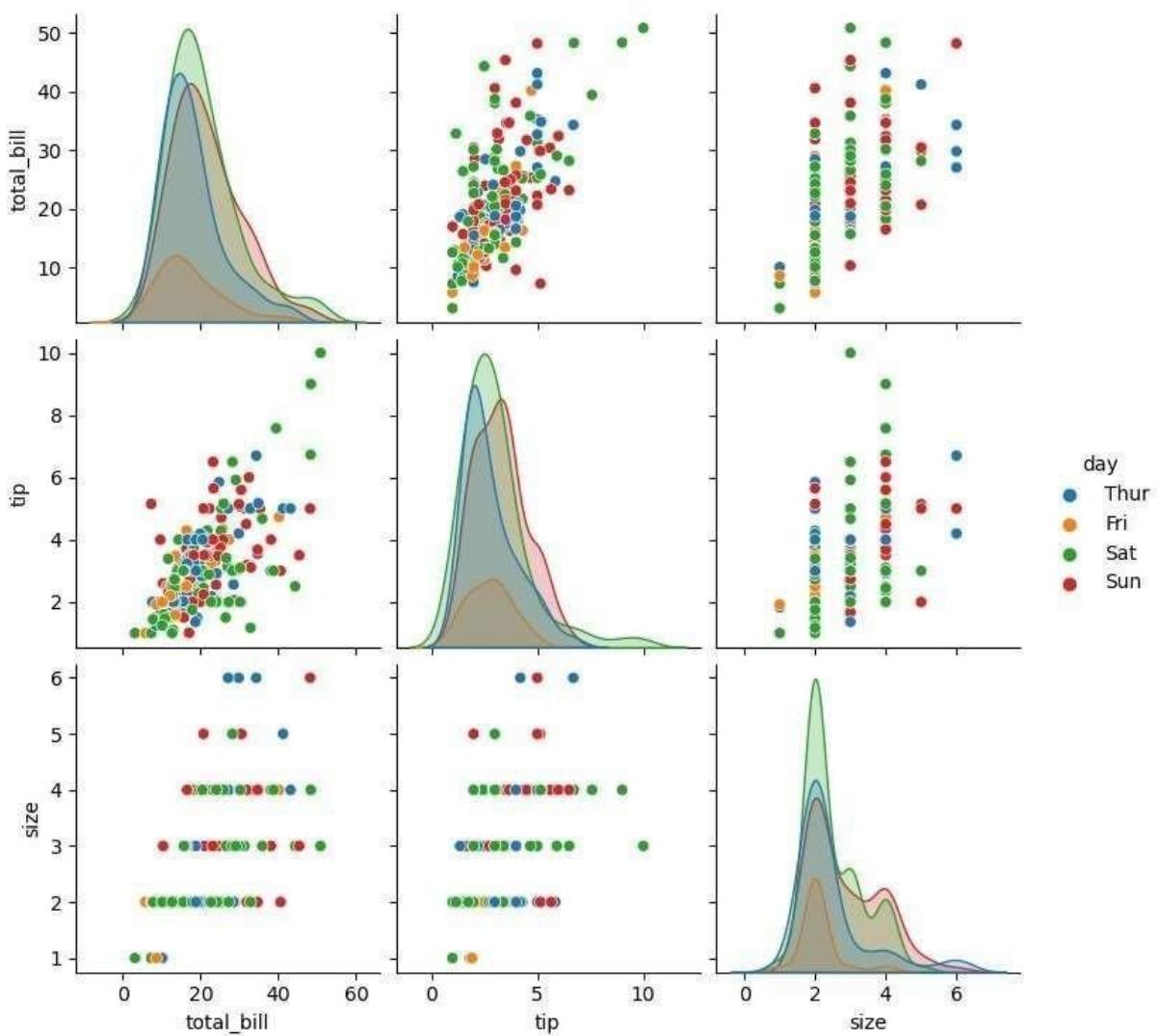
tips.time.value_counts()
Dinner    176
Lunch     68
Name: time, dtype: int64
sns.pairplot(tips,hue='time')
<seaborn.axisgrid.PairGrid at 0x1cbb258d8a0>

```



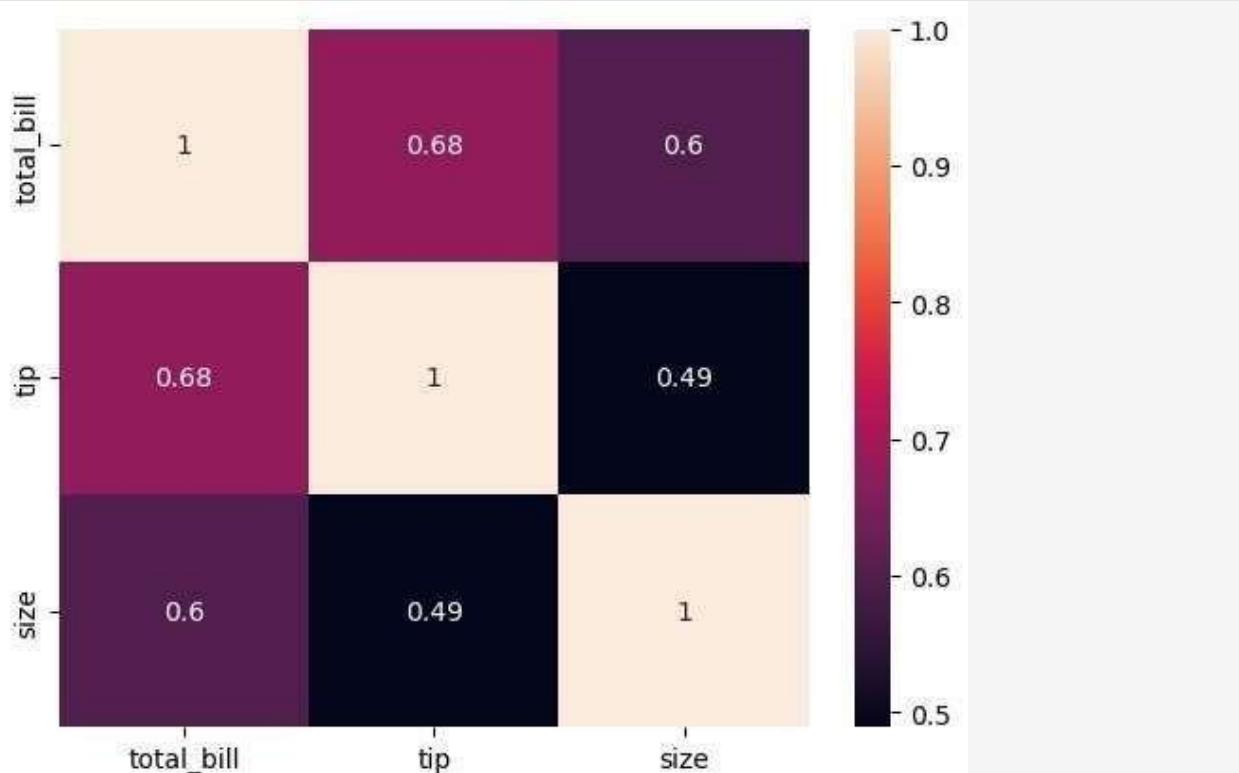
```
sns.pairplot(tips,hue='day')
```

```
<seaborn.axisgrid.PairGrid at 0x1ccb20b9120>
```



```
sns.heatmap(tips.corr(numeric_only=True), annot=True)
```

<Axes:

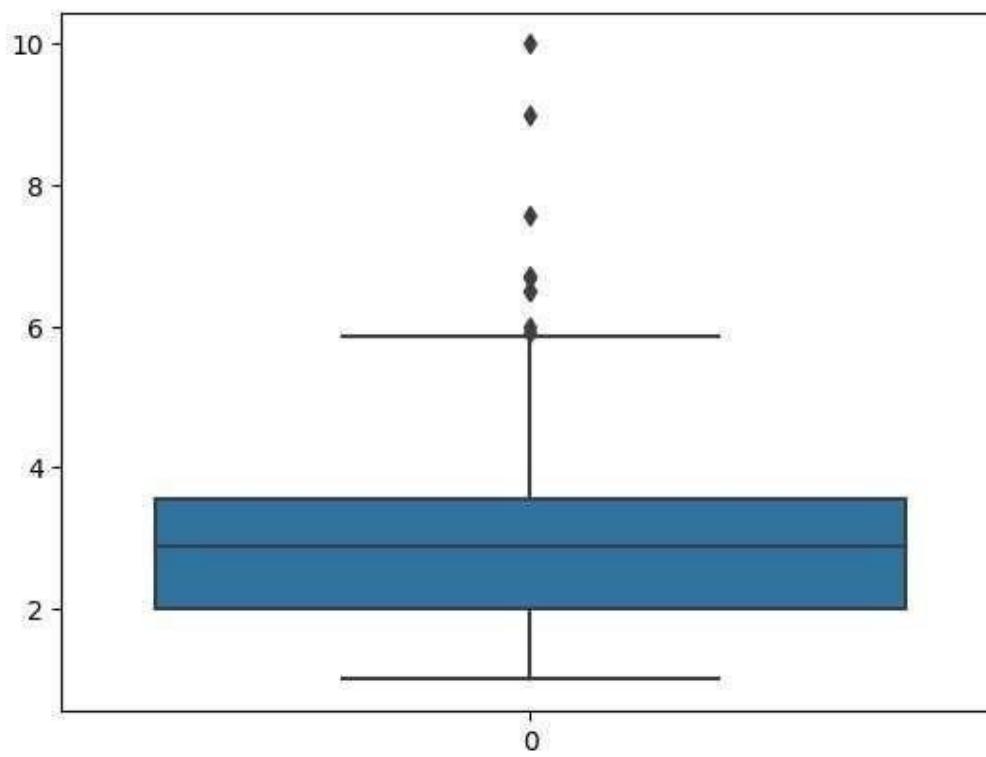
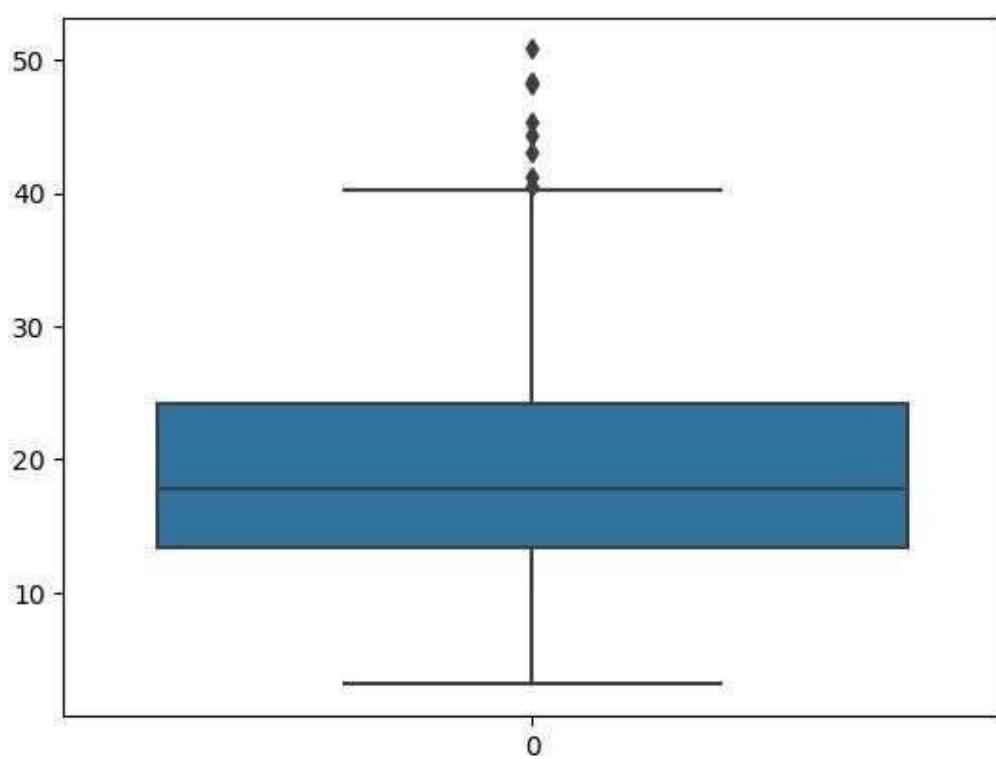


```
sns.boxplot(tips.total_bill)
```

```
<Axes:
```

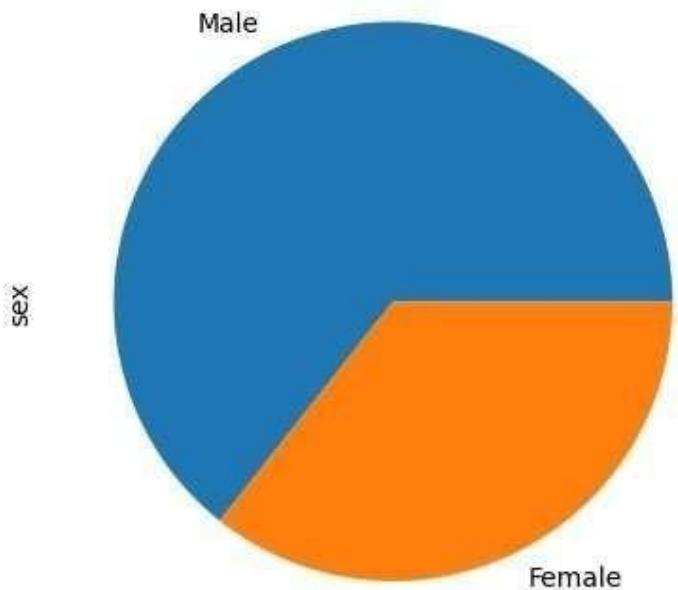
```
sns.boxplot(tips.tip)
```

```
<Axes:
```



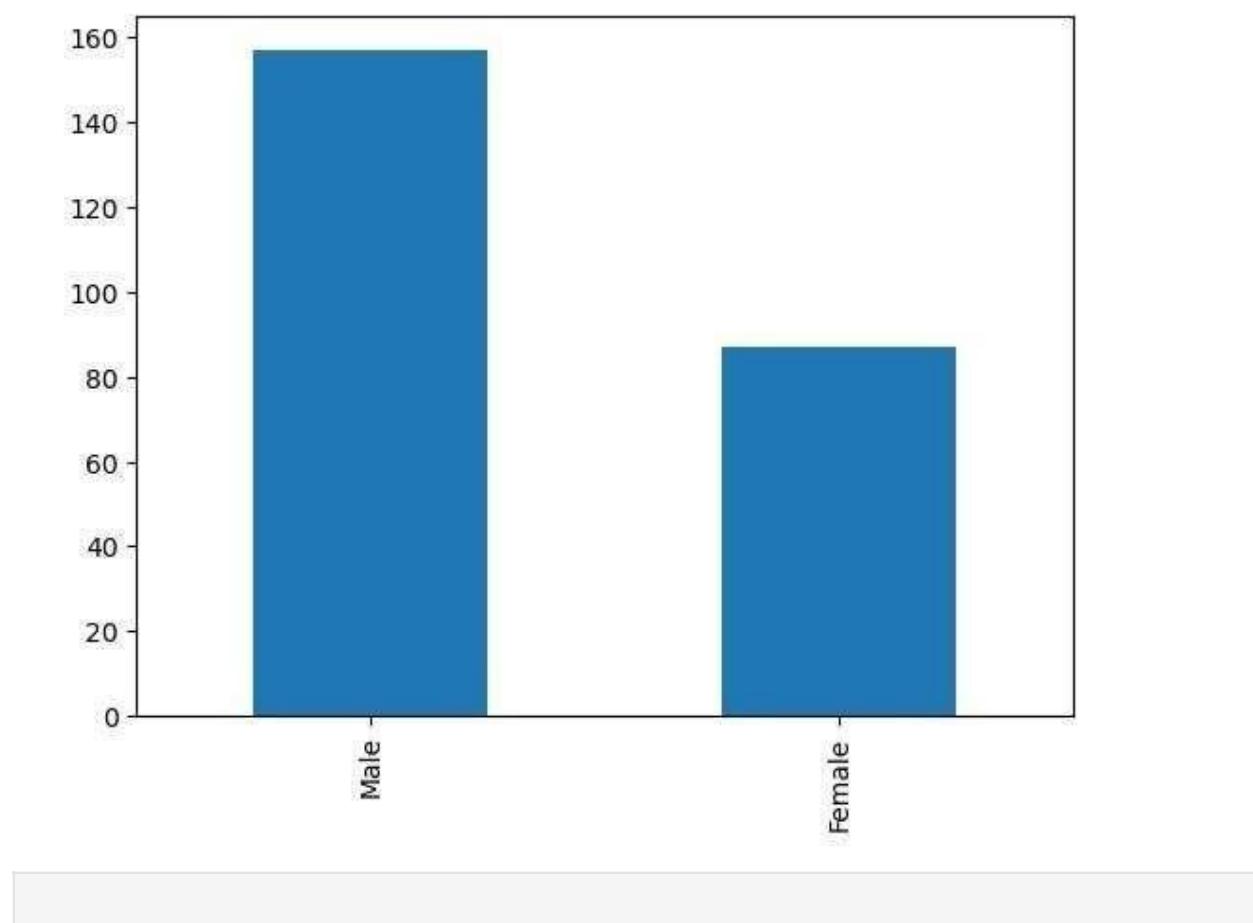
```
tips.sex.value_counts().plot(kind='pie')
```

```
<Axes: ylabel='sex'
```



```
tips.sex.value_counts().plot(kind='bar')
```

```
<Axes:
```



```

import numpy as np import pandas as pd
df=pd.read_csv('E:/Salary_data.csv') df df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29 Data columns (total 2 columns):
 # Column      Non-Null Count Dtype  
 --- 
 0 YearsExperience 30 non-null   float64 1  Salary      30 non-
null   int64  dtypes: float64(1), int64(1) memory usage: 608.0 bytes
df.dropna(inplace=True) df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29 Data columns (total 2 columns):
 # Column      Non-Null Count Dtype  
 --- 
 0 YearsExperience 30 non-null   float64 1  Salary      30 non-
null   int64  dtypes: float64(1), int64(1) memory usage: 608.0 bytes
df.describe()

  YearsExperience    Salary count    30.000000
 30.000000 mean      5.313333  76003.000000 std
 2.837888 27414.429785 min           1.100000
 37731.000000 25%      3.200000  56720.750000
 50%          4.700000  65237.000000 75%
 7.700000
100544.750000 max      10.500000 122391.000000

features=df.iloc[:,[0]].values label=df.iloc[:,[1]].values from
sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2,random_state=42)
from sklearn.linear_model import LinearRegression model=LinearRegression()
model.fit(x_train,y_train)

LinearRegression()

```

```
model.score(x_train,y_train)
0.9645401573418146
model.score(x_test,y_test)
0.9024461774180497
model.coef_
array([[9423.81532303]])
model.intercept_
array([25321.58301178])
import pickle
pickle.dump(model,open('SalaryPred.model','wb'))
model=pickle.load(open('SalaryPred.model','rb')) yr_of_exp=float(input("Enter Years of Experience: "))
yr_of_exp_NP=np.array([[yr_of_exp]]) Salary=model.predict(yr_of_exp_NP)

Enter Years of Experience: 44
print("Estimated Salary for {} years of experience is {}: ".format(yr_of_exp,Salary))

Estimated Salary for 44.0 years of experience is [[439969.45722514]]:
```

```
import numpy as np import pandas as pd  
df=pd.read_csv('E:/Social_Network_Ads.csv') df
```

User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000
1	15810944	Male	35	20000
2	15668575	Female	26	43000
3	15603246	Female	27	57000
4	15804002	Male	19	76000
... 395	15691863	Female	46	41000
396	15706071	Male	51	23000
397	15654296	Female	50	20000
398	15755018	Male	36	33000
399	15594041	Female	49	36000

[400 rows x 5 columns]

```
User ID Gender Age EstimatedSalary Purchased 0 15624510 Male 19 df.head()
```

19000 0

1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

```
features=df.iloc[:,[2,3]].values
```

```
label=df.iloc[:,4].values features
```

```
array([[ 19, 19000],
```

[ 35, 20000],  
[ 26, 43000],  
[ 27, 57000],  
[ 19, 76000],  
[ 27, 58000],  
[ 27, 84000],  
[ 32, 150000],  
[ 25, 33000],  
[ 35, 65000],  
[ 26, 80000],  
[ 26, 52000],  
[ 20, 86000],  
[ 32, 18000],  
[ 18, 82000],  
[ 29, 80000],  
[ 47, 25000],

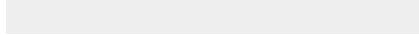




```
[ 45, 26000],  
[ 46, 28000],  
[ 48, 29000],  
[ 45, 22000],  
[ 47, 49000],  
[ 48, 41000],  
[ 45, 22000],  
[ 46, 23000],  
[ 47, 20000],  
[ 49, 28000],  
[ 47, 30000],  
[ 29, 43000],  
[ 31, 18000],  
[ 31, 74000],  
[ 27, 137000],  
[ 21, 16000],  
[ 28, 44000],  
[ 27, 90000],  
[ 35, 27000],  
[ 33, 28000],  
[ 30, 49000],  
[ 26, 72000],  
[ 27, 31000],  
[ 27, 17000],  
[ 33, 51000],  
[ 35, 108000],  
[ 30, 15000],  
[ 28, 84000],  
[ 23, 20000],  
[ 25, 79000],  
[ 27, 54000],  
[ 30, 135000],  
[ 31, 89000],  
[ 24, 32000],  
[ 18, 44000],  
[ 29, 83000],  
[ 35, 23000],  
[ 27, 58000],  
[ 24, 55000],  
[ 23, 48000],  
[ 28, 79000],  
[ 22, 18000],  
[ 32, 117000],  
[ 27, 20000],  
[ 25, 87000],
```



[ 23, 66000], [ 24, 58000],  
[ 32, 120000], [ 59, 83000],

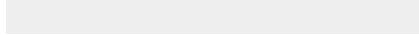




```
[ 24, 19000],  
[ 23, 82000],  
[ 22, 63000],  
[ 31, 68000],  
[ 25, 80000],  
[ 24, 27000],  
[ 20, 23000],  
[ 33, 113000],  
[ 32, 18000],  
[ 34, 112000],  
[ 18, 52000],  
[ 22, 27000],  
[ 28, 87000],  
[ 26, 17000],  
[ 30, 80000],  
[ 39, 42000],  
[ 20, 49000],  
[ 35, 88000],  
[ 30, 62000],  
[ 31, 118000],  
[ 24, 55000],  
[ 28, 85000],  
[ 26, 81000],  
[ 35, 50000],  
[ 22, 81000],  
[ 30, 116000],  
[ 26, 15000],  
[ 29, 28000],  
[ 29, 83000],  
[ 35, 44000],  
[ 35, 25000],  
[ 28, 123000],  
[ 35, 73000],  
[ 28, 37000],  
[ 27, 88000],  
[ 28, 59000],  
[ 32, 86000],  
[ 33, 149000],  
[ 19, 21000],  
[ 21, 72000],  
[ 26, 35000],  
[ 27, 89000],  
[ 26, 86000],  
[ 38, 80000],  
[ 39, 71000],
```



[ 37, 71000], [ 42, 80000],  
[ 38, 61000], [ 37, 55000],

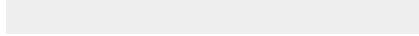




```
[ 40, 57000],  
[ 35, 75000],  
[ 36, 52000],  
[ 40, 59000],  
[ 41, 59000],  
[ 36, 75000],  
[ 37, 72000],  
[ 40, 75000],  
[ 35, 53000],  
[ 41, 51000],  
[ 39, 61000],  
[ 42, 65000],  
[ 26, 32000],  
[ 30, 17000],  
[ 26, 84000],  
[ 31, 58000],  
[ 33, 31000],  
[ 30, 87000],  
[ 21, 68000],  
[ 28, 55000],  
[ 23, 63000],  
[ 20, 82000],  
[ 30, 107000],  
[ 28, 59000],  
[ 19, 25000],  
[ 19, 85000],  
[ 18, 68000],  
[ 35, 59000],  
[ 30, 89000],  
[ 34, 25000],  
[ 24, 89000],  
[ 27, 96000],  
[ 41, 30000],  
[ 29, 61000],  
[ 20, 74000],  
[ 26, 15000],  
[ 41, 45000],  
[ 31, 76000],  
[ 36, 50000],  
[ 40, 47000],  
[ 31, 15000],  
[ 46, 59000],  
[ 29, 75000],  
[ 26, 30000],  
[ 32, 135000],
```



[ 32, 100000], [ 35, 38000],  
[ 25, 90000], [ 37, 33000],

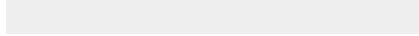




```
[ 33, 69000],  
[ 18, 86000],  
[ 22, 55000],  
[ 35, 71000],  
[ 29, 148000],  
[ 29, 47000],  
[ 21, 88000],  
[ 34, 115000],  
[ 26, 118000],  
[ 34, 43000],  
[ 34, 72000],  
[ 23, 28000],  
[ 35, 47000],  
[ 25, 22000],  
[ 24, 23000],  
[ 31, 34000],  
[ 26, 16000],  
[ 31, 71000],  
[ 32, 117000],  
[ 33, 43000],  
[ 33, 60000],  
[ 31, 66000],  
[ 20, 82000],  
[ 33, 41000],  
[ 35, 72000],  
[ 28, 32000],  
[ 24, 84000],  
[ 19, 26000],  
[ 29, 43000],  
[ 19, 70000],  
[ 28, 89000],  
[ 34, 43000],  
[ 30, 79000],  
[ 20, 36000],  
[ 26, 80000],  
[ 35, 22000],  
[ 35, 39000],  
[ 49, 74000],  
[ 39, 134000],  
[ 41, 71000],  
[ 58, 101000],  
[ 47, 47000],  
[ 55, 130000],  
[ 52, 114000],  
[ 40, 142000],
```



[ 46, 22000], [ 59, 42000],  
[ 48, 96000], [ 52, 150000],

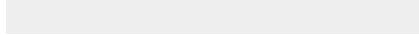




```
[ 35, 58000],  
[ 47, 43000],  
[ 60, 108000],  
[ 49, 65000],  
[ 40, 78000],  
[ 46, 96000],  
[ 59, 143000],  
[ 41, 80000],  
[ 35, 91000],  
[ 37, 144000],  
[ 60, 102000],  
[ 35, 60000],  
[ 37, 53000],  
[ 36, 126000],  
[ 56, 133000],  
[ 40, 72000],  
[ 42, 80000],  
[ 35, 147000],  
[ 39, 42000],  
[ 40, 107000],  
[ 49, 86000],  
[ 38, 112000],  
[ 46, 79000],  
[ 40, 57000],  
[ 37, 80000],  
[ 46, 82000],  
[ 53, 143000],  
[ 42, 149000],  
[ 38, 59000],  
[ 50, 88000],  
[ 56, 104000],  
[ 41, 72000],  
[ 51, 146000],  
[ 35, 50000],  
[ 57, 122000],  
[ 41, 52000],  
[ 35, 97000],  
[ 44, 39000],  
[ 37, 52000],  
[ 48, 134000],  
[ 37, 146000],  
[ 50, 44000],  
[ 52, 90000],  
[ 41, 72000],  
[ 40, 57000],
```



[ 58, 95000], [ 36, 144000],  
[ 45, 131000], [ 35, 77000],

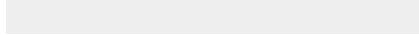




```
[ 55, 125000],  
[ 35, 72000],  
[ 48, 90000],  
[ 42, 108000],  
[ 40, 75000],  
[ 37, 74000],  
[ 47, 144000],  
[ 40, 61000],  
[ 43, 133000],  
[ 59, 76000],  
[ 60, 42000],  
[ 39, 106000],  
[ 57, 26000],  
[ 57, 74000],  
[ 38, 71000],  
[ 49, 88000],  
[ 52, 38000],  
[ 50, 36000],  
[ 59, 88000],  
[ 35, 61000],  
[ 37, 70000],  
[ 52, 21000],  
[ 48, 141000],  
[ 37, 93000],  
[ 37, 62000],  
[ 48, 138000],  
[ 41, 79000],  
[ 37, 78000],  
[ 39, 134000],  
[ 49, 89000],  
[ 55, 39000],  
[ 37, 77000],  
[ 35, 57000],  
[ 36, 63000],  
[ 42, 73000],  
[ 43, 112000],  
[ 45, 79000],  
[ 46, 117000],  
[ 58, 38000],  
[ 48, 74000],  
[ 37, 137000],  
[ 37, 79000],  
[ 40, 60000],  
[ 42, 54000],  
[ 51, 134000],
```



[ 47, 113000],  
[ 36, 125000],  
[ 38, 50000],





```
[ 39, 96000],  
[ 38, 50000],  
[ 49, 141000],  
[ 39, 79000],  
[ 39, 75000],  
[ 54, 104000],  
[ 35, 55000],  
[ 45, 32000],  
[ 36, 60000],  
[ 52, 138000],  
[ 53, 82000],  
[ 41, 52000],  
[ 48, 30000],  
[ 48, 131000],  
[ 41, 60000],  
[ 41, 72000],  
[ 42, 75000],  
[ 36, 118000],  
[ 47, 107000],  
[ 38, 51000],  
[ 48, 119000],  
[ 42, 65000],  
[ 40, 65000],  
[ 57, 60000],  
[ 36, 54000],  
[ 58, 144000],  
[ 35, 79000],  
[ 38, 55000],  
[ 39, 122000],  
[ 53, 104000],  
[ 35, 75000],  
[ 38, 65000],  
[ 47, 51000],  
[ 47, 105000],  
[ 41, 63000],  
[ 53, 72000],  
[ 54, 108000],  
[ 39, 77000],  
[ 38, 61000],  
[ 38, 113000],  
[ 37, 75000],  
[ 42, 90000],  
[ 37, 57000],  
[ 36, 99000],  
[ 60, 34000],
```



[ 54, 70000], [ 42, 54000],  
[ 41, 72000], [ 40, 71000],



```
[ 43, 129000],
```

```
[ 53, 34000],  
[ 47, 50000],  
[ 42, 79000],  
[ 42, 104000],  
[ 59, 29000],  
[ 58, 47000],  
[ 46, 88000],  
[ 38, 71000],  
[ 54, 26000],  
[ 60, 46000],  
[ 60, 83000],  
[ 39, 73000],  
[ 59, 130000],  
[ 37, 80000],  
[ 46, 32000],  
[ 46, 74000],  
[ 42, 53000],  
[ 41, 87000],  
[ 58, 23000],  
[ 42, 64000],  
[ 48, 33000],  
[ 44, 139000],  
[ 49, 28000],  
[ 57, 33000],  
[ 56, 60000],  
[ 49, 39000],  
[ 39, 71000],  
[ 47, 34000],  
[ 48, 35000],  
[ 48, 33000],  
[ 47, 23000],  
[ 45, 45000],  
[ 60, 42000],  
[ 39, 59000],  
[ 46, 41000],  
[ 51, 23000],  
[ 50, 20000],  
[ 36, 33000],  
[ 49, 36000]], dtype=int64)  
label  
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 0  
1,  
1  
0,  
0
```

0

0  
0

0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,



0,

0,  
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,  
0,  
0,  
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,  
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,

```
from sklearn.model_selection import train_test_split LogisticRegression
from sklearn.linear_model import LogisticRegression

for i in range(10):
    x_train,x_test,y_train,y_test=train_test_split(features, target, test_size=0.3, random_state=i)
    model=LogisticRegression()
    model.fit(x_train,y_train)
    print("Test {} Train {} Random State {}".format(model.score(x_test,y_test), model.score(x_train,y_train), i))
```

Test 0.65 Train0.640625 Random State 5  
Test 0.65 Train0.640625 Random State 6  
Test 0.65 Train0.640625 Random State 7  
Test 0.65 Train0.640625 Random State 8  
Test 0.65 Train0.640625 Random State 9  
Test 0.65 Train0.640625 Random State 10  
Test 0.65 Train0.640625 Random State 11  
Test 0.65 Train0.640625 Random State 12  
Test 0.65 Train0.640625 Random State 13  
Test 0.65 Train0.640625 Random State 14  
Test 0.65 Train0.640625 Random State 15  
Test 0.65 Train0.640625 Random State 16  
Test 0.65 Train0.640625 Random State 17  
Test 0.65 Train0.640625 Random State 18  
Test 0.65 Train0.640625 Random State 19  
Test 0.65 Train0.640625 Random State 20  
Test 0.65 Train0.640625 Random State 21  
Test 0.65 Train0.640625 Random State 22  
Test 0.65 Train0.640625 Random State 23  
Test 0.65 Train0.640625 Random State 24  
Test 0.65 Train0.640625 Random State 25  
Test 0.65 Train0.640625 Random State 26  
Test 0.65 Train0.640625 Random State 27  
Test 0.65 Train0.640625 Random State 28

2

Test 0.65 Train0.640625 Random State 33  
Test 0.65 Train0.640625 Random State 34  
Test 0.65 Train0.640625 Random State 35  
Test 0.65 Train0.640625 Random State 36  
Test 0.65 Train0.640625 Random State 37  
Test 0.65 Train0.640625 Random State 38  
Test 0.65 Train0.640625 Random State 39  
Test 0.65 Train0.640625 Random State 40  
Test 0.65 Train0.640625 Random State 41  
Test 0.65 Train0.640625 Random State 42  
Test 0.65 Train0.640625 Random State 43  
Test 0.65 Train0.640625 Random State 44  
Test 0.65 Train0.640625 Random State 45  
Test 0.65 Train0.640625 Random State 46  
Test 0.65 Train0.640625 Random State 47  
Test 0.65 Train0.640625 Random State 48  
Test 0.65 Train0.640625 Random State 49  
Test 0.65 Train0.640625 Random State 50  
Test 0.65 Train0.640625 Random State 51  
Test 0.65 Train0.640625 Random State 52 Test 0.65 Train0.640625 Random State 53

---

Test 0.65 Train0.640625 Random State 54  
Test 0.65 Train0.640625 Random State 55  
Test 0.65 Train0.640625 Random State 56  
Test 0.65 Train0.640625 Random State 57  
Test 0.65 Train0.640625 Random State 58  
Test 0.65 Train0.640625 Random State 59  
Test 0.65 Train0.640625 Random State 60  
Test 0.65 Train0.640625 Random State 61  
Test 0.65 Train0.640625 Random State 62  
Test 0.65 Train0.640625 Random State 63  
Test 0.65 Train0.640625 Random State 64  
Test 0.65 Train0.640625 Random State 65  
Test 0.65 Train0.640625 Random State 66  
Test 0.65 Train0.640625 Random State 67  
Test 0.65 Train0.640625 Random State 68  
Test 0.65 Train0.640625 Random State 69  
Test 0.65 Train0.640625 Random State 70  
Test 0.65 Train0.640625 Random State 71  
Test 0.65 Train0.640625 Random State 72  
Test 0.65 Train0.640625 Random State 73  
Test 0.65 Train0.640625 Random State 74  
Test 0.65 Train0.640625 Random State 75  
Test 0.65 Train0.640625 Random State 76  
Test 0.65 Train0.640625 Random State 77  
Test 0.65 Train0.640625 Random State 78  
Test 0.65 Train0.640625 Random State 79  
Test 0.65 Train0.640625 Random State 80  
Test 0.65 Train0.640625 Random State 81  
Test 0.65 Train0.640625 Random State 82  
Test 0.65 Train0.640625 Random State 83  
Test 0.65 Train0.640625 Random State 84  
Test 0.65 Train0.640625 Random State 85  
Test 0.65 Train0.640625 Random State 86  
Test 0.65 Train0.640625 Random State 87  
Test 0.65 Train0.640625 Random State 88  
Test 0.65 Train0.640625 Random State 89  
Test 0.65 Train0.640625 Random State 90  
Test 0.65 Train0.640625 Random State 91  
Test 0.65 Train0.640625 Random State 92  
Test 0.65 Train0.640625 Random State 93  
Test 0.65 Train0.640625 Random State 94  
Test 0.65 Train0.640625 Random State 95  
Test 0.65 Train0.640625 Random State 96  
Test 0.65 Train0.640625 Random State 97  
Test 0.65 Train0.640625 Random State 98  
Test 0.65 Train0.640625 Random State 99

---

Test 0.65 Train0.640625 Random State 100  
Test 0.65 Train0.640625 Random State 101

Test 0.65 Train0.640625 Random State 102  
Test 0.65 Train0.640625 Random State 103  
Test 0.65 Train0.640625 Random State 104  
Test 0.65 Train0.640625 Random State 105  
Test 0.65 Train0.640625 Random State 106  
Test 0.65 Train0.640625 Random State 107  
Test 0.65 Train0.640625 Random State 108  
Test 0.65 Train0.640625 Random State 109  
Test 0.65 Train0.640625 Random State 110  
Test 0.65 Train0.640625 Random State 111  
Test 0.65 Train0.640625 Random State 112  
Test 0.65 Train0.640625 Random State 113  
Test 0.65 Train0.640625 Random State 114  
Test 0.65 Train0.640625 Random State 115  
Test 0.65 Train0.640625 Random State 116  
Test 0.65 Train0.640625 Random State 117  
Test 0.65 Train0.640625 Random State 118  
Test 0.65 Train0.640625 Random State 119  
Test 0.65 Train0.640625 Random State 120  
Test 0.65 Train0.640625 Random State 121

Test 0.65 Train0.640625 Random State 122

Test 0.65 Train0.640625 Random State 123

Test 0.65 Train0.640625 Random State 124

Test 0.65 Train0.640625 Random State 125  
Test 0.65 Train0.640625 Random State 126  
Test 0.65 Train0.640625 Random State 127  
Test 0.65 Train0.640625 Random State 128  
Test 0.65 Train0.640625 Random State 129  
Test 0.65 Train0.640625 Random State 130  
Test 0.65 Train0.640625 Random State 131  
Test 0.65 Train0.640625 Random State 132  
Test 0.65 Train0.640625 Random State 133  
Test 0.65 Train0.640625 Random State 134  
Test 0.65 Train0.640625 Random State 135  
Test 0.65 Train0.640625 Random State 136  
Test 0.65 Train0.640625 Random State 137  
Test 0.65 Train0.640625 Random State 138

Test 0.65 Train0.640625 Random State 139  
Test 0.65 Train0.640625 Random State 140  
Test 0.65 Train0.640625 Random State 141  
Test 0.65 Train0.640625 Random State 142  
Test 0.65 Train0.640625 Random State 143

Test 0.65 Train0.640625 Random State 144

Test 0.65 Train0.640625 Random State 145

Test 0.65 Train0.640625 Random State 146  
Test 0.65 Train0.640625 Random State 147  
Test 0.65 Train0.640625 Random State 148  
Test 0.65 Train0.640625 Random State 149

Test 0.65 Train0.640625 Random State 151  
Test 0.65 Train0.640625 Random State 152  
Test 0.65 Train0.640625 Random State 153  
Test 0.65 Train0.640625 Random State 154  
Test 0.65 Train0.640625 Random State 155  
Test 0.65 Train0.640625 Random State 156  
Test 0.65 Train0.640625 Random State 157  
Test 0.65 Train0.640625 Random State 158  
Test 0.65 Train0.640625 Random State 159  
Test 0.65 Train0.640625 Random State 160  
Test 0.65 Train0.640625 Random State 161  
Test 0.65 Train0.640625 Random State 162  
Test 0.65 Train0.640625 Random State 163  
Test 0.65 Train0.640625 Random State 164  
Test 0.65 Train0.640625 Random State 165  
Test 0.65 Train0.640625 Random State 166  
Test 0.65 Train0.640625 Random State 167  
Test 0.65 Train0.640625 Random State 168  
Test 0.65 Train0.640625 Random State 169  
Test 0.65 Train0.640625 Random State 170  
Test 0.65 Train0.640625 Random State 171  
Test 0.65 Train0.640625 Random State 172

Test 0.65 Train0.640625 Random State 173

Test 0.65 Train0.640625 Random State 174

Test 0.65 Train0.640625 Random State 175

Test 0.65 Train0.640625 Random State 176  
Test 0.65 Train0.640625 Random State 177  
Test 0.65 Train0.640625 Random State 178  
Test 0.65 Train0.640625 Random State 179  
Test 0.65 Train0.640625 Random State 180  
Test 0.65 Train0.640625 Random State 181  
Test 0.65 Train0.640625 Random State 182  
Test 0.65 Train0.640625 Random State 183  
Test 0.65 Train0.640625 Random State 184  
Test 0.65 Train0.640625 Random State 185  
Test 0.65 Train0.640625 Random State 186  
Test 0.65 Train0.640625 Random State 187  
Test 0.65 Train0.640625 Random State 188  
Test 0.65 Train0.640625 Random State 189  
Test 0.65 Train0.640625 Random State 190  
Test 0.65 Train0.640625 Random State 191

v

.65 Train0.640625 Random State 150

Test 0.65 Train0.640625 Random State 192

Test 0.65 Train0.640625 Random State 193

Test 0.65 Train0.640625 Random State 194

Test 0.65 Train0.640625 Random State 195 Test 0.65 Train0.640625 Random State 196

Test 0.65 Train0.640625 Random State 197  
Test 0.65 Train0.640625 Random State 198  
Test 0.65 Train0.640625 Random State 199

Test 0.65 Train0.640625 Random State 200

Test 0.65 Train0.640625 Random State 201  
Test 0.65 Train0.640625 Random State 202  
Test 0.65 Train0.640625 Random State 203  
Test 0.65 Train0.640625 Random State 204  
Test 0.65 Train0.640625 Random State 205  
Test 0.65 Train0.640625 Random State 206  
Test 0.65 Train0.640625 Random State 207  
Test 0.65 Train0.640625 Random State 208  
Test 0.65 Train0.640625 Random State 209  
Test 0.65 Train0.640625 Random State 210  
Test 0.65 Train0.640625 Random State 211  
Test 0.65 Train0.640625 Random State 212  
Test 0.65 Train0.640625 Random State 213  
Test 0.65 Train0.640625 Random State 214  
Test 0.65 Train0.640625 Random State 215  
Test 0.65 Train0.640625 Random State 216  
Test 0.65 Train0.640625 Random State 217  
Test 0.65 Train0.640625 Random State 218  
Test 0.65 Train0.640625 Random State 219  
Test 0.65 Train0.640625 Random State 220  
Test 0.65 Train0.640625 Random State 221  
Test 0.65 Train0.640625 Random State 222

Test 0.65 Train0.640625 Random State 223  
Test 0.65 Train0.640625 Random State 224  
Test 0.65 Train0.640625 Random State 225  
Test 0.65 Train0.640625 Random State 226  
Test 0.65 Train0.640625 Random State 227  
Test 0.65 Train0.640625 Random State 228  
Test 0.65 Train0.640625 Random State 229  
Test 0.65 Train0.640625 Random State 230  
Test 0.65 Train0.640625 Random State 231  
Test 0.65 Train0.640625 Random State 232  
Test 0.65 Train0.640625 Random State 233  
Test 0.65 Train0.640625 Random State 234  
Test 0.65 Train0.640625 Random State 235  
Test 0.65 Train0.640625 Random State 236  
Test 0.65 Train0.640625 Random State 237

Test 0.65 Train0.640625 Random State 238  
Test 0.65 Train0.640625 Random State 239  
Test 0.65 Train0.640625 Random State 240  
Test 0.65 Train0.640625 Random State 241  
Test 0.65 Train0.640625 Random State 242  
Test 0.65 Train0.640625 Random State 243  
Test 0.65 Train0.640625 Random State 244 Test 0.65 Train0.640625 Random State 245  
Test 0.65 Train0.640625 Random State 246

Test 0.65 Train0.640625 Random State 247  
Test 0.65 Train0.640625 Random State 248

Test 0.65 Train0.640625 Random State 250  
Test 0.65 Train0.640625 Random State 251  
Test 0.65 Train0.640625 Random State 252  
Test 0.65 Train0.640625 Random State 253  
Test 0.65 Train0.640625 Random State 254  
Test 0.65 Train0.640625 Random State 255  
Test 0.65 Train0.640625 Random State 256  
Test 0.65 Train0.640625 Random State 257  
Test 0.65 Train0.640625 Random State 258  
Test 0.65 Train0.640625 Random State 259  
Test 0.65 Train0.640625 Random State 260  
Test 0.65 Train0.640625 Random State 261  
Test 0.65 Train0.640625 Random State 262  
Test 0.65 Train0.640625 Random State 263  
Test 0.65 Train0.640625 Random State 264  
Test 0.65 Train0.640625 Random State 265  
Test 0.65 Train0.640625 Random State 266  
Test 0.65 Train0.640625 Random State 267  
Test 0.65 Train0.640625 Random State 268  
Test 0.65 Train0.640625 Random State 269  
Test 0.65 Train0.640625 Random State 270  
Test 0.65 Train0.640625 Random State 271  
Test 0.65 Train0.640625 Random State 272  
Test 0.65 Train0.640625 Random State 273  
Test 0.65 Train0.640625 Random State 274  
Test 0.65 Train0.640625 Random State 275  
Test 0.65 Train0.640625 Random State 276  
Test 0.65 Train0.640625 Random State 277  
Test 0.65 Train0.640625 Random State 278  
Test 0.65 Train0.640625 Random State 279  
Test 0.65 Train0.640625 Random State 280  
Test 0.65 Train0.640625 Random State 281

Test 0.65 Train0.640625

m State 283 Test 0.65 Train0.640625 Random State 284  
Test 0.65 Train0.640625 Random State 285  
Test 0.65 Train0.640625 Random State 286  
Test 0.65 Train0.640625 Random State 287  
Test 0.65 Train0.640625 Random State 288  
Test 0.65 Train0.640625 Random State 289  
Test 0.65 Train0.640625 Random State 290  
Test 0.65 Train0.640625 Random State 291  
Test 0.65 Train0.640625 Random State 292  
Test 0.65 Train0.640625 Random State 293  
Test 0.65 Train0.640625 Random State 294

Test 0.65 Train0.640625 Random State 282

Test 0.65 Train0.640625 Random State 295  
Test 0.65 Train0.640625 Random State 296

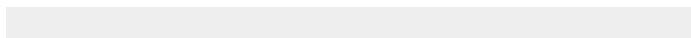
Test 0.65 Train0.640625 Random State 297

Test 0.65 Train0.640625 Random State 299  
Test 0.65 Train0.640625 Random State 300  
Test 0.65 Train0.640625 Random State 301  
Test 0.65 Train0.640625 Random State 302  
Test 0.65 Train0.640625 Random State 303  
Test 0.65 Train0.640625 Random State 304  
Test 0.65 Train0.640625 Random State 305  
Test 0.65 Train0.640625 Random State 306  
Test 0.65 Train0.640625 Random State 307  
Test 0.65 Train0.640625 Random State 308  
Test 0.65 Train0.640625 Random State 309  
Test 0.65 Train0.640625 Random State 310  
Test 0.65 Train0.640625 Random State 311  
Test 0.65 Train0.640625 Random State 312  
Test 0.65 Train0.640625 Random State 313  
Test 0.65 Train0.640625 Random State 314  
Test 0.65 Train0.640625 Random State 315  
Test 0.65 Train0.640625 Random State 316  
Test 0.65 Train0.640625 Random State 317  
Test 0.65 Train0.640625 Random State 318  
Test 0.65 Train0.640625 Random State 319  
Test 0.65 Train0.640625 Random State 320  
Test 0.65 Train0.640625 Random State 321  
Test 0.65 Train0.640625 Random State 322  
Test 0.65 Train0.640625 Random State 323  
Test 0.65 Train0.640625 Random State 324  
Test 0.65 Train0.640625 Random State 325  
Test 0.65 Train0.640625 Random State 326  
Test 0.65 Train0.640625 Random State 327  
Test 0.65 Train0.640625 Random State 328  
Test 0.65 Train0.640625 Random State 329

Test 0.65 Train0.640625 Random State 298

Test 0.65 Train0.640625 Random State 330  
Test 0.65 Train0.640625 Random State 331  
Test 0.65 Train0.640625 Random State 332  
Test 0.65 Train0.640625 Random State 333  
Test 0.65 Train0.640625 Random State 334  
Test 0.65 Train0.640625 Random State 335  
Test 0.65 Train0.640625 Random State 336  
Test 0.65 Train0.640625 Random State 337  
Test 0.65 Train0.640625 Random State 338  
Test 0.65 Train0.640625 Random State 339  
Test 0.65 Train0.640625 Random State 340  
Test 0.65 Train0.640625 Random State 341  
Test 0.65 Train0.640625 Random State 342  
Test 0.65 Train0.640625 Random State 343

Test 0.65 Train0.640625 Random State 344  
Test 0.65 Train0.640625 Random State 345  
Test 0.65 Train0.640625 Random State 346  
Test 0.65 Train0.640625 Random State 347



Test 0.65 Train0.640625 Random State 348  
Test 0.65 Train0.640625 Random State 349  
Test 0.65 Train0.640625 Random State 350  
Test 0.65 Train0.640625 Random State 351  
Test 0.65 Train0.640625 Random State 352  
Test 0.65 Train0.640625 Random State 353  
Test 0.65 Train0.640625 Random State 354  
Test 0.65 Train0.640625 Random State 355  
Test 0.65 Train0.640625 Random State 356  
Test 0.65 Train0.640625 Random State 357  
Test 0.65 Train0.640625 Random State 358  
Test 0.65 Train0.640625 Random State 359  
Test 0.65 Train0.640625 Random State 360  
Test 0.65 Train0.640625 Random State 361  
Test 0.65 Train0.640625 Random State 362  
Test 0.65 Train0.640625 Random State 363  
Test 0.65 Train0.640625 Random State 364  
Test 0.65 Train0.640625 Random State 365  
Test 0.65 Train0.640625 Random State 366  
Test 0.65 Train0.640625 Random State 367  
Test 0.65 Train0.640625 Random State 368  
Test 0.65 Train0.640625 Random State 369  
Test 0.65 Train0.640625 Random State 370  
Test 0.65 Train0.640625 Random State 371  
Test 0.65 Train0.640625 Random State 372  
Test 0.65 Train0.640625 Random State 373  
Test 0.65 Train0.640625 Random State 374  
Test 0.65 Train0.640625 Random State 375  
Test 0.65 Train0.640625 Random State 376  
Test 0.65 Train0.640625 Random State 377  
Test 0.65 Train0.640625 Random State 378  
Test 0.65 Train0.640625 Random State 379  
Test 0.65 Train0.640625 Random State 380  
Test 0.65 Train0.640625 Random State 381  
Test 0.65 Train0.640625 Random State 382  
Test 0.65 Train0.640625 Random State 383  
Test 0.65 Train0.640625 Random State 384  
Test 0.65 Train0.640625 Random State 385  
Test 0.65 Train0.640625 Random State 386  
Test 0.65 Train0.640625 Random State 387  
Test 0.65 Train0.640625 Random State 388  
Test 0.65 Train0.640625 Random State 389  
Test 0.65 Train0.640625 Random State 390  
Test 0.65 Train0.640625 Random State 391  
Test 0.65 Train0.640625 Random State 392  
Test 0.65 Train0.640625 Random State 393  
Test 0.65 Train0.640625 Random State 394

```
0.640625
```

```
0.65
```

```
Test 0.65
```

```
Train0.640625
```

```
Random State
```

```
398
```

```
Test 0.65
```

```
Train0.640625
```

```
Random State
```

```
399
```

```
Test 0.65
```

```
Train0.640625
```

```
Random State
```

```
400
```

```
x_train,x_test,y_train,y_test=train_test_split(features,label,test_size=0.2,random_state=42)  
finalModel=LogisticRegression() finalModel.fit(x_train,y_train)
```

```
LogisticRegression()
```

```
print(finalModel.score(x_train,y_train)) print(finalModel.score(x_test,y_test))
```

```
from sklearn.metrics import classification_report  
print(classification_report(label,finalModel.predict(features)))  
  
precision recall f1-score support  
  
0 0.64 1.00 0.78 257 1 0.00 0.00 0.00  
143 accuracy 0.64 400 macro avg 0.32 0.50  
0.39  
400 weighted avg 0.41 0.64 0.50 400
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\metrics\
```

```
_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0  
in labels with no predicted samples. Use 'zero_division' parameter to control this behavior.
```

```
_warn_prf(average, modifier, msg_start, len(result))
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\metrics\
```

```
_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0  
in labels with no predicted samples. Use 'zero_division' parameter to control this behavior.
```

```
_warn_prf(average, modifier, msg_start, len(result))
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\metrics\
```

```
_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0  
in labels with no predicted samples. Use 'zero_division' parameter to control this behavior.
```

```
_warn_prf(average, modifier, msg_start, len(result))
```

```

import numpy as np import pandas as pd
df=pd.read_csv('E:/Iris.csv') df.info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column      Non-Null Count Dtype  
 ---  --          --          --      
 0   sepal.length    150 non-null   float64 
 1   sepal.width     150 non-null   float64 
 2   petal.length    150 non-null   float64 
 3   petal.width     150 non-null   float64 
 4   variety        150 nonnull    object   dtypes: float64(4), object(1)
memory usage: 6.0+ KB df.variety.value_counts()
Setosa      50
Versicolor  50
Virginica   50
Name: variety, dtype: int64
df.head()
sepal.length  sepal.width  petal.length  petal.width  variety  0      5.1      3.5      1.4      0.2
Setosa
1           4.9       3.0       1.4       0.2 Setosa
2           4.7       3.2       1.3       0.2 Setosa
3           4.6       3.1       1.5       0.2 Setosa
4           5.0       3.6       1.4       0.2 Setosa
features=df.iloc[:, :-1].values label=df.iloc[:, 4].values

from sklearn.model_selection import train_test_split from sklearn.neighbors import KNeighborsClassifier
xtrain,xtest,ytrain,ytest=train_test_split(features,label,test_size=.2,random_state=42) model_KNN=KNeighborsClassifier(n_neighbors=5)
model_KNN.fit(xtrain,ytrain)
KNeighborsClassifier()
print(model_KNN.score(xtrain,ytrain)) print(model_KNN.score(xtest,ytest))

0.9666666666666667
1.0

```

```
from sklearn.metrics import confusion_matrix confusion_matrix(label,model_KNN.predict(features))
array([[50, 0, 0], [0, 47, 3],
       [0, 1, 49]], dtype=int64)
from sklearn.metrics import classification_report
print(classification_report(label,model_KNN.predict(features)))
precision    recall   f1-score   support
Setosa      1.00      1.00      1.00      50
Versicolor  0.98      0.94      0.96      50    Virginica     0.94      0.98      0.96
50 accuracy                           0.97      150 macro avg     0.97      0.97      0.97
150 weighted avg     0.97      0.97      0.97      150
```

```
import numpy as np import pandas as pd import
matplotlib.pyplot as plt import seaborn as sns
%matplotlib inline
df=pd.read_csv('E:/Mall_Customers.csv') df.info()

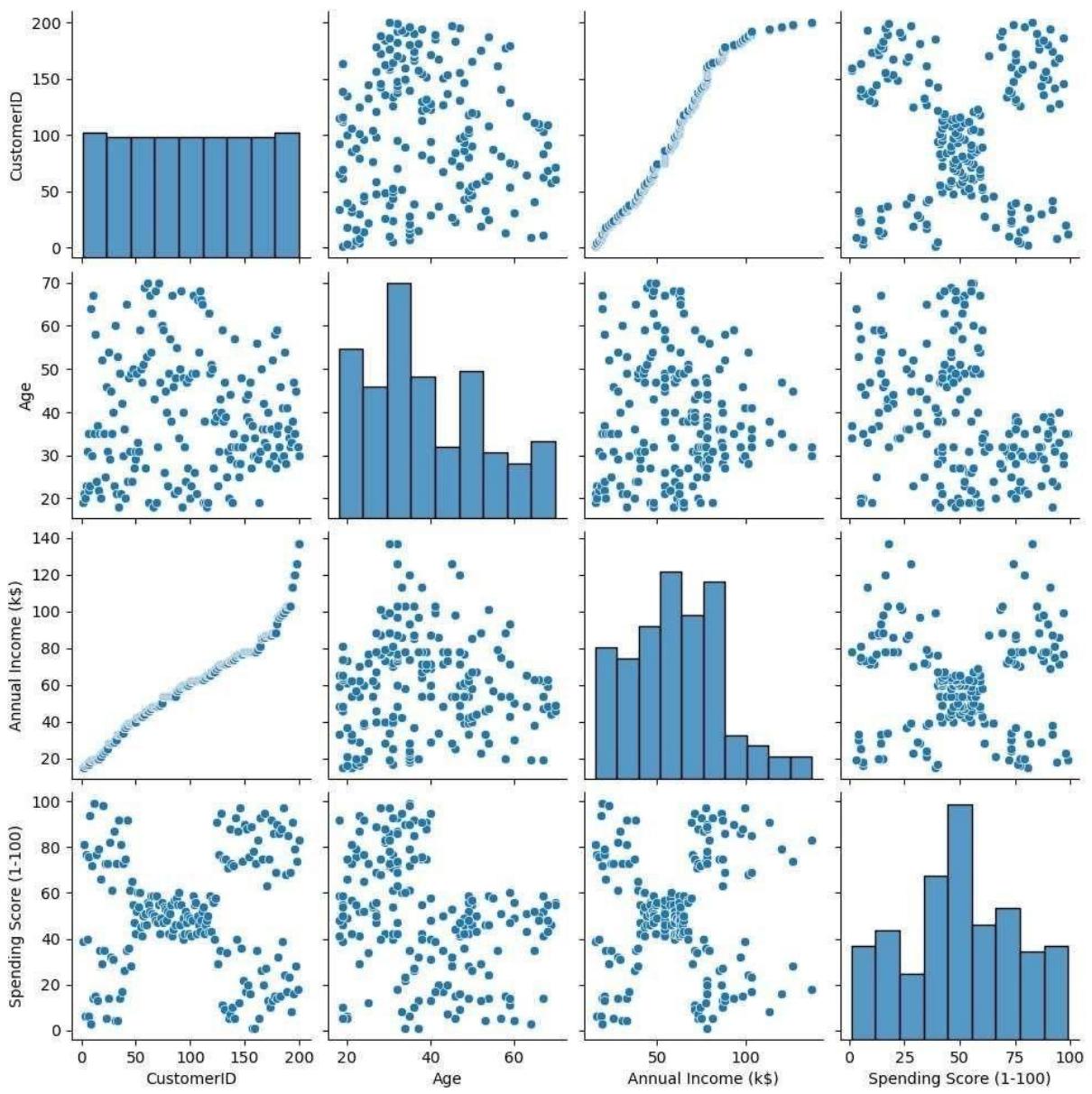
<class 'pandas.core.frame.DataFrame'> RangeIndex: 200 entries, 0 to 199 Data
columns (total 5 columns):
 #   Column      Non-Null Count Dtype
---  --          -----  --  CustomerID      200 non-null  int64
 1   Gender       200 non-null  object
 2   Age          200 non-null  int64
 3   Annual Income (k$)  200 non-null  int64  4   Spending Score (1-100) 200 non-null  int64
dtypes: int64(4),
```

	CustomerID	Gender	Age	Annual Income (k\$)	Spending Score (1-100)
0		1	Male	19	15
1		2	Male	21	15
2		3	Female	20	16
3		4	Female	23	16
4		5	Female	31	17

```
object(1) memory usage: 7.9+ KB df.head()
```

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x1dc59c15c90>
```



```

features =df.iloc[:,[3,4]].values

from sklearn.cluster import KMeans
model = KMeans(n_clusters = 5)
model.fit(features)
KMeans(n_clusters = 5)

```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

```
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

```
KMeans(n_clusters=5)
```

```
Final=df.iloc[:,[3,4]]
Final['label']=model.predict(features)
Final.head()
```

```
C:\Users\REC\AppData\Local\Temp\ipykernel_7552\470183701.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation:

[https://pandas.pydata.org/pandasdocs/stable/user\\_guide/indexing.html#](https://pandas.pydata.org/pandasdocs/stable/user_guide/indexing.html#)

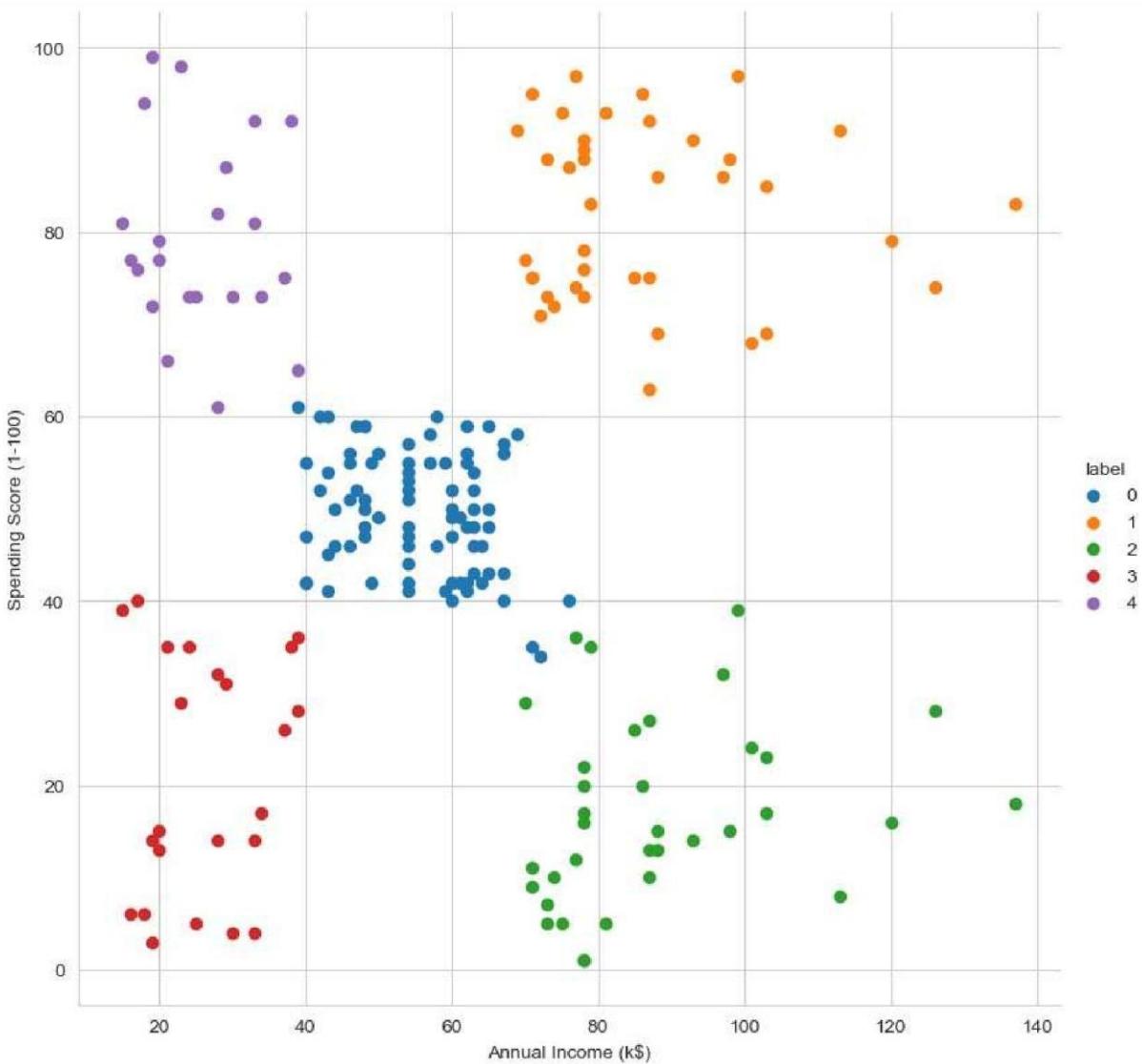
```
Final['label']=model.predict(features) Annual Income (k$) Spending Score
(1                               -100) label
0                  15          39   3
1                  15          81   4
2                  16           6   3
3                  16          77  4 4
```

r  
e  
t  
u  
r  
n  
i  
n  
g  
-

a-view-versus-a-copy

17 40 3

```
sns.set_style("whitegrid") sns.FacetGrid(Final,hue="label",height=8) \
.map(plt.scatter,"Annual Income (k$)", "Spending Score (1-100)") \
.add_legend(); plt.show()
```



```

features_el=df.iloc[:,[2,3,4]].values
from sklearn.cluster import KMeans
wcss=[]
for i in range(1,10):
    model=KMeans(n_clusters=i)
    model.fit(features_el)
    wcss.append(model.inertia_)
plt.plot(range(1,10),wcss)

C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will
change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly
to suppress the warning
    warnings.warn(
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\

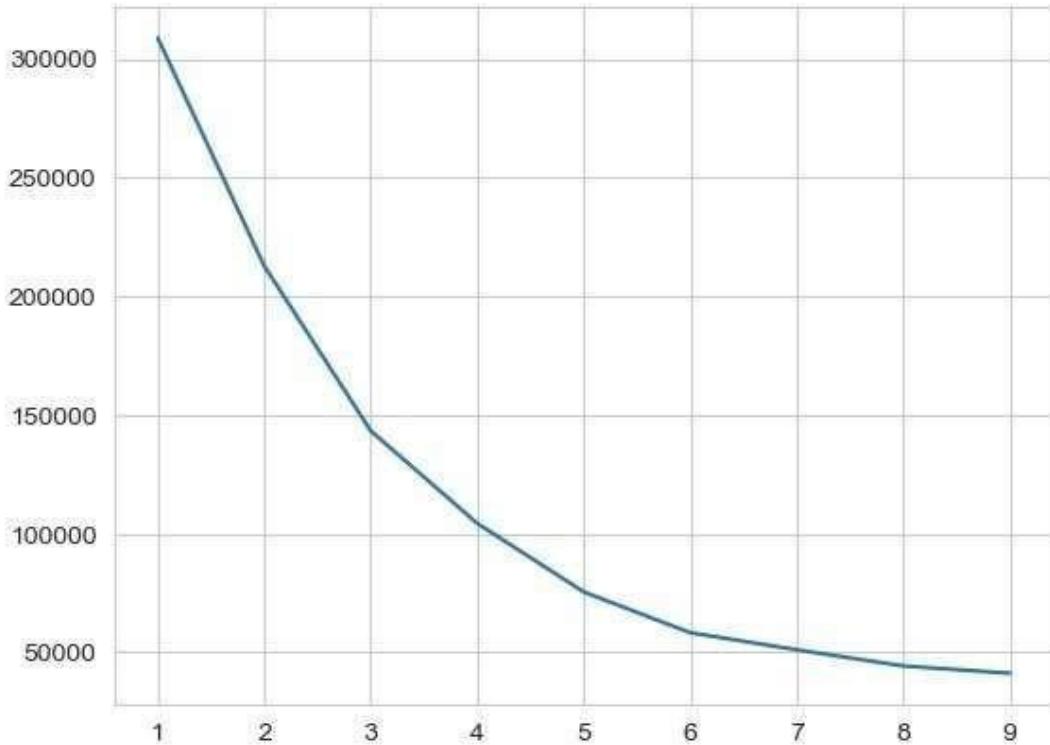
```

\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the  
environment variable  
OMP\_NUM\_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default  
value of `n\_init` will change from 10 to 'auto' in 1.4. Set the value of `n\_init` explicitly  
to suppress the warning  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\  
\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on

Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable  
OMP\_NUM\_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'auto' in 1.4. Set the value of `n\_init` explicitly to suppress the warning warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable  
OMP\_NUM\_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:870: FutureWarning: The default value of `n\_init` will change from 10 to 'auto' in 1.4. Set the value of `n\_init` explicitly to suppress the warning warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP\_NUM\_THREADS=1.

```
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable  
OMP_NUM_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable  
OMP_NUM_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on
```

```
_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the  
environment variable  
OMP_NUM_THREADS=1.  
warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default  
value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly  
to suppress the warning warnings.warn(  
C:\ProgramData\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1382: UserWarning: KMeans is known to have a memory leak on  
Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the  
environment variable  
OMP_NUM_THREADS=1. warnings.warn(  
[<matplotlib.lines.Line2D at 0x1dc61c56380>]
```



T-statistic: 1.993

P-value: 0.0774

Fail to Reject Null Hypothesis → No significant difference.

```
import numpy as np from scipy import stats marks = np.array([72, 68, 75, 70, 74, 69, 71, 73, 70, 72]) mu_0 = 70 t_stat, p_value = stats.ttest_1samp(marks, mu_0) print(f'Tstatistic: {t_stat:.3f}') print(f'P-value: {p_value:.4f}') alpha = 0.05 if p_value<alpha: print("Reject Null Hypothesis → Mean is significantly different from 70.") else: print("Fail to Reject Null Hypothesis")
```

Z-statistic: 2.400

P-value: 0.0164

significant difference."

Reject Null Hypothesis → Mean is significantly different from 50 g.

→ No

)

```
import numpy as np from math import sqrt from scipy.stats import norm x_bar = 51.2 mu_0 = 50 sigma = 3 n = 36 z_stat = (x_bar - mu_0) / (sigma / sqrt(n)) p_value = 2 * (1 - norm.cdf(abs(z_stat))) print(f'Z-statistic: {z_stat:.3f}') print(f'P-value: {p_value:.4f}') alpha = 0.05 if p_value < alpha: print("Reject Null Hypothesis → Mean is significantly different from 50 g.") else: print("Fail to")
```

Reject Null Hypothesis → No significant difference.")

```
import numpy as np from scipy import stats  
A      = [20, 22,  
23]  
B      = [19, 20,  
18] C = [25, 27,  
26] f_stat, p_value = stats.f_oneway(A, B, C)
```

```
print(f'F-statistic: {f_stat:.3f} | print(f'P-
value: {p_value:.4f} )\n\nalpha = 0.05 if p_value < alpha:     print("Reject Null
Hypothesis → Means are significantly different.") else:
print("Fail to Reject Null Hypothesis → No significant
difference.")\nF-statistic: 25.923
P-value: 0.0011
```

Reject Null Hypothesis → Means are significantly different.