Perform the following operations using Python on the Facebook metrics data sets

**a.** Create data subsets

**b.** Merge Data

**c.** Sort Data

**d.** Transposing Data

**e.** Shape and reshape Data

import pandas as pd

import numpy as np

from google.colab import files

uploaded = files.upload()

df = pd.read\_csv('dataset\_Facebook.csv')

df

df.describe()

df.shape

#subset1

df1 = df[['pagetotallikes','type','category','postmonth']].loc[0:15]

df1

#subset2

df2 = df[['pagetotallikes','type','category','postmonth']].loc[16:30]

df2

#subset3

df3 = df[['pagetotallikes','type','category','postmonth']].loc[30:45]

df3

* merging = pd.concat([df1,df2,df3])

merging

* df.transpose()
* sort\_values = df.sort\_values('pagetotallikes', ascending=False)

sort\_values

#shaping

shaping = df.shape

shaping

#reshaping

pivot\_table = pd.pivot\_table(df,index=['type','category'], values='comment')

pivot\_table

#extra command to reshape data using array

reshaping\_arr = np.array([1,2,3,4,5,6])

reshaping\_arr.reshape(2,3)

* 1. Perform the following operations using Python on the Air quality and Heart Diseases data sets

**a.** Data cleaning

* 1. **b.** Data integration
  2. **c.** Data transformation
  3. **d.** Error correcting
  4. **e.** Data model building

from google. colab import files uploaded = files. upload()

|  |
| --- |
| Choose Files |

No file chosen Upload widget is only available when the cell has

been executed in the current browser session. Please rerun this cell to enable.

hoar\* t 1 h -f-n hoar•+ (1 h

from google. colab import files uploaded = files. upload()

|  |
| --- |
| Choose Files |

No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

A-i -i -FM r c" A-i -i +\,• r cil import pandas as pd

weather=pd . read\_csv( weatherHistory . csv' )

air\_quality=pd. read\_csv( ' AirQua1ity. csv' , sep=' • ' , decimal=' , heart=pd.  'heart (1) . csv' ) air\_quality. head()

 Date Time CO(GT) pTØ8.SI(CO) NMHC(GT) C6H6(GT) PTØ8.S2(NMHC)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| o 10/03/2004 | 18.00.oo | 2.6 | 1360.0 | 150.0 | 11.9 | 1046.0 |
| 1 10/03/2004 | 19.00.oo | 2.0 | 1292.0 | 112.0 | 9.4 | 955.0 |
| 2 10/03/2004 | 20.00.oo | 2.2 | 1402.0 | 88.0 | 9.0 | 939.0 |
| 3 10/03/2004 | 21.00.oo | 2.2 | 1376.0 | 80.0 | 9.2 | 948.0 |
| 4 10/03/2004 | 22.00.oo | 1.6 | 1272.0 | 51.0 | 6.5 | 836.0 |



# Data Cleaning

## air\_quality. isnull( ) . sum()

Date 114 Time 114

CO(GT) 114

PT08 . (CO) 114 NMHC(GT) 114

C6H6(GT) 114 (NMHC) 114

NOx(GT) 114

PT08 . S3 (NOx) 114

N02(GT) 114

PT08 . S4(N02)

PT08.S5 (03) 114

114

RH 114 AH 114

Unnamed: 15 9471 Unnamed: 16 9471 dtype: int64

### air\_quality. shape

(9471, 17)

air\_quality. drop(columns=[ 'Unnamed: 15' , 'Unnamed: 16' ] , axis=l, inp1ace=True)

air\_quality. inp1ace=True)

## air\_quality. isnull( ) . sum()

|  |  |
| --- | --- |
| Date | 0 |
| Time  CO(GT) | 0 |
| PT08 . (CO) | 0 |
| NMHC(GT) | 0 |
| C6H6(GT) | 0 |
| PT08 (NMHC) | 0 |
| NOx(GT) | 0 |
| PT08 . S3 (NOx) | 0 |
| N02(GT) | 0 |
| PT08 . S4(N02) PT08.S5 (03) | 0  0 |
| AH | 0 |

dtype: int64

# Data Transformation and Error Correcting

air\_quality. rename(columns={ 'T' : 'Temperature' , 'RH 'Humidity' , 'AH' : 'Absolute Humidity' }, inp1ace=True)

air\_quality[ 'Date' ] = pd .  'Date ' ] , format= '%d/%m/%Y' ) air\_quality [ 'Time' ] - pd.to\_timedelta(air\_quality[ 'Time' ] .str. strip() + ' :ØO' )

air\_quality[ ' Datetime ' ] = air\_quality[ 'Date' ] + air\_quality[ 'Time' ]

air\_quality. set\_index( ' Datetime' , inp1ace=True)

## air\_quality. head ( )

Date Time CO(GT) PTØ8.S1(CO) NMHC(GT) C6H6(GT) PTØ8. (NMHC) 

Datetime

2004-03-

2004- 0 days

102.6 1360.0 150.0 11.9 1046.0

03-10 



2004-03-

2004- 0 days

 10 03-102.0 1292.0 112.0 9.4 955.0

2004-03-

2004- 0 days

 10 03-102.2 1402.0 88.0 9.0 939.0

2004-03-

2004- 0 days

10 03-10 212.2 1376.0 80.0 9.2 948.0



21

2004-03-

2004- 0 days

 10 03-101.6 1272.0 51.0 6.5 836.0



## Data Integration

# Making Subset of air quality dataset to show data integration air subsetl=air\_quality[[ 'Date' , 'Time' , 'CO(GT)' PT08.S1(CO) ' NMHC(GT) ' , 'C6H6(GT) ' , ' PT08.S2(NMHC) ' ' Tempera air subset2=air\_qua1ity[[ 'Temperature' , 'Humidity ' , 'Absolute Humidity

### air subsetl. head()

#### Date Time CO(GT) PTØ8.S1(CO) NMHC(GT) C6H6(GT) PTØ8. S2 (NMHC)

Datetime

2004-03-

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 102.6  03-10  air\_ subset 2. head ( ) | 1360.0 150.0 | | 11.9 | 1046.0 |
| Temperature  Datetime | Humidity | Absolute Humidity |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 2004-03-10 | 13.3 | 47.7 | 0.7255 |
| 2004-03-10 | 11.9 | 54.0 | 0.7502 |
| 2004-03-10 21 | 11.0 | 60.0 | 0.7867 |
| 2004-03-10 | 11.2 | 59.6 | 0.7888 |

 2004- 0 days

integrated\_data=pd . merge(air\_subsetl, air\_subset2, on= ' Temperature )

integrated\_data . head( )

Date Time CO(GT) PTØ8. Sl(co) NMHC(GT) C6H6(GT) PTØ8.S2(NMHC) Temperal

2004- 0 days 2.6 1360.0 150.0 11.9 1046.0

03-10 

2004- 0 days

12.6 1360.0 150.0 11.9 1046.0

03-10 

2004- 0 days

2 2.6 1360.0 150.0 11.9 1046.0

4 A.nn.nn

## Data Model building

# We will use heart dataset to for model building heart. head()

age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca 



o

1. 53 1 140 203 1 o 155 1 3.1 o o
2. 70 1 145 174 1 125 1 2.6 o o
3. 61 1 148 203 1 161 0 0.0 2 1
4. 62 o o 138 294 1 1 106 0 1.9 1 3



### heart. isnull() . sum()

age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal target dtype : int64 heart. info()

<class 'pandas. core. frame.DataFrame' > Rangelndex: 1025 entries, O to 1024 Data columns (total 14 columns) :

 Column Non-Null Count Dtype



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| o | age | 1025 | non-null | int64 |
| 1 | sex | 1025 | non-null | int64 |
| 2 | cp | 1025 | non-null | int64 |
| 3 | trestbps | 1025 | non-null | int64 |
| 4 | chol | 1025 | non-null | int64 |
| 5 | fbs | 1025 | non-null | int64 |
| 6 | restecg | 1025 | non-null | int64 |
| 7 | thalach | 1025 | non-null | int64 |
| 8 | exang | 1025 | non-null | int64 |
| 9 | oldpeak | 1025 | non-null | float64 |
| 10 | slope | 1025 | non-null | int64 |
| 11 | ca | 1025 | non-null | int64 |
| 12 | thal | 1025 | non-null | int64 |
| 13 | target | 1025 | non-null | int64 |
| dtypes: float64(1), | | | int64(13) |  |
| memory usage: 112.2 | | | KB |  |

y=heart[[ 'target ' ] ] x=heart . drop(y, axis=l)

y. head(2)

target



1

x. head(2)

age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca 



52

125

212

168

0

1.0

2

2

o o



1

1

1

Y [ 'target ' ] •value\_counts()

1 526

0 499

Name : target, dtype: int64

from sklearn.model\_selection import train\_test\_split

#### x\_train, x\_test, y\_train , (x, y , test\_size=0.2, random\_state=42)

x\_train . shape, x\_test . shape, y\_train . shape, y\_test . shape

( (820, 13) , (205, 13) , (820, 1) , (205, 1) )

from sklearn.linear\_model import LogisticRegression

Ir=LogisticRegression()

model=lr.fit(x\_train,y\_train) print( "Model built successfully")

#### Model built successfully

/usr/10ca1/1ib/python3.9/dist-packages/sk1earn/uti1s/va1idation.py:1143: DataConversionWarning: A column-vector y was Y column\_or\_ld(y, warn=True)

/usr/10ca1/1ib/python3.9/dist-packages/sk1earn/1inear\_mode1/\_10gistic.py:458: ConvergenceWarning: lbfgs failed to conver

STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in: https : / /scikit-learn .org/stable/modules/@reprocessing. html

Please also refer to the documentation for alternative solver options:

htt s : scikit-learn.or stable modules linear model. html#lo istic-re ression n iter i = \_check\_optimize\_result(



#predicting on test data y\_test[ Prediction ' ]=model . predict (x\_test)

y\_test . head()

#### target Prediction

527 1 1

359 1 1

447 0 0

31 1 1

621 0 0

from sklearn.metrics import accuracy\_ score

print(accuracy\_score(y\_test[ ' target ' ]  ' Prediction ' ] ) )

o. 7853658536585366

Visualize the data using Python libraries matplotlib, seaborn by plotting the graphs

Use dataset : Air quality and Heart Diseases

Out[4]:

In

[2]:

In

[3]:

In

[4]:

**age**

**sex**

**cp**

**trestbps**

**chol**

**fbs**

**restecg**

**thalach**

**exang**

**oldpeak**

**slope**

**ca**

**thal**

**num**

**0**

63

1

1

145

233

1

2

150

0

2.3

3

0

6

0

*#Visualize the data using Python libraries matplotlib, seaborn by plotting the graphs for assignment no. 2 and 3 ( Group B)*

*#import dependencies*

**import**

matplotlib

.

pyplot

**as**

plt

**import**

seaborn

**as**

sns

**import**

numpy

**as**

np

**import**

pandas

**as**

pd

df

**=**

pd

.

read\_csv

(

'heartdisease.csv'

)

df

.

head

(

5

)

1. 67 1 4 160 286 0 2 108 1 1.5 2 3 3 2
2. 67 1 4 120 229 0 2 129 1 2.6 2 2 7 1
3. 37 1 3 130 250 0 0 187 0 3.5 3 0 3 0
4. 41 0 2 130 204 0 2 172 0 1.4 1 0 3 0

In

[6]:

In

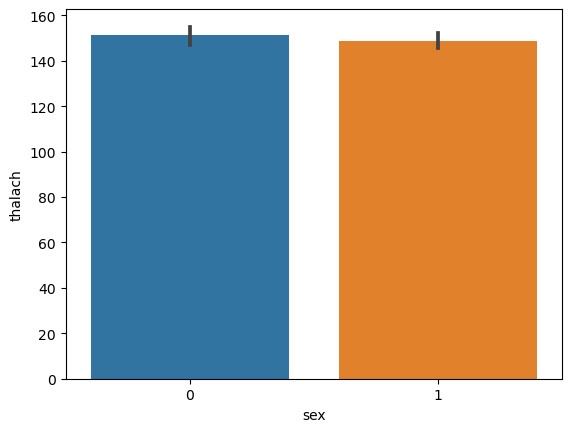
[18]:

Out[18]:

<

AxesSubplot:xlabel='sex', ylabel='thalach'

>



*#Barplot using seaborn*

sns

.

barplot

(

x

**=**

'sex'

,

y

**=**

'thalach'

,

data

**=**

df

)

[20]:

In

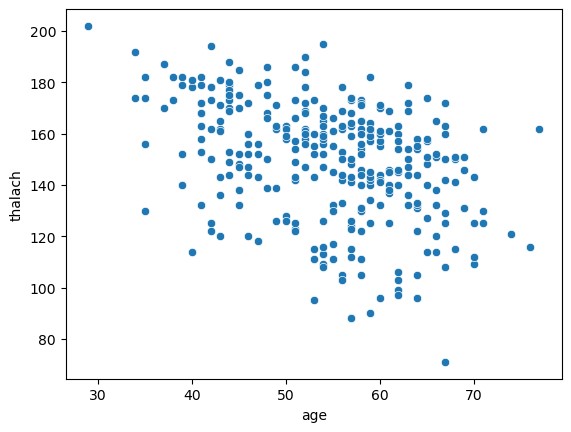
[17]:

Out[20]:

AxesSubplot:xlabel='age', ylabel='thalach'

>

<

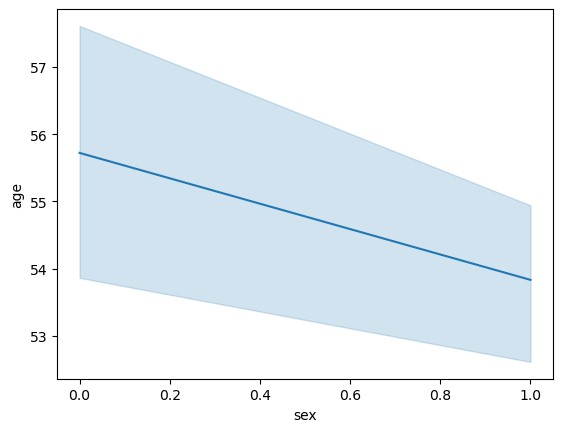


Out[17]:

>

<

AxesSubplot:xlabel='sex', ylabel='age'



*#ScatterPlot using seaborn*

sns

.

scatterplot

(

x

**=**

'age'

,

y

**=**

'thalach'

,

data

**=**

df

)

*#Lineplot using seaborn*

sns

.

lineplot

(

x

**=**

'sex'

,

y

**=**

'age'

,

data

**=**

df

)

[29]:

*#Pairplot using seaborn*

plt

.

figure

(

figsize

**=**

(

12

,

12

))

sns

.

pairplot

(

df

,

hue

**=**

'sex'

,

palette

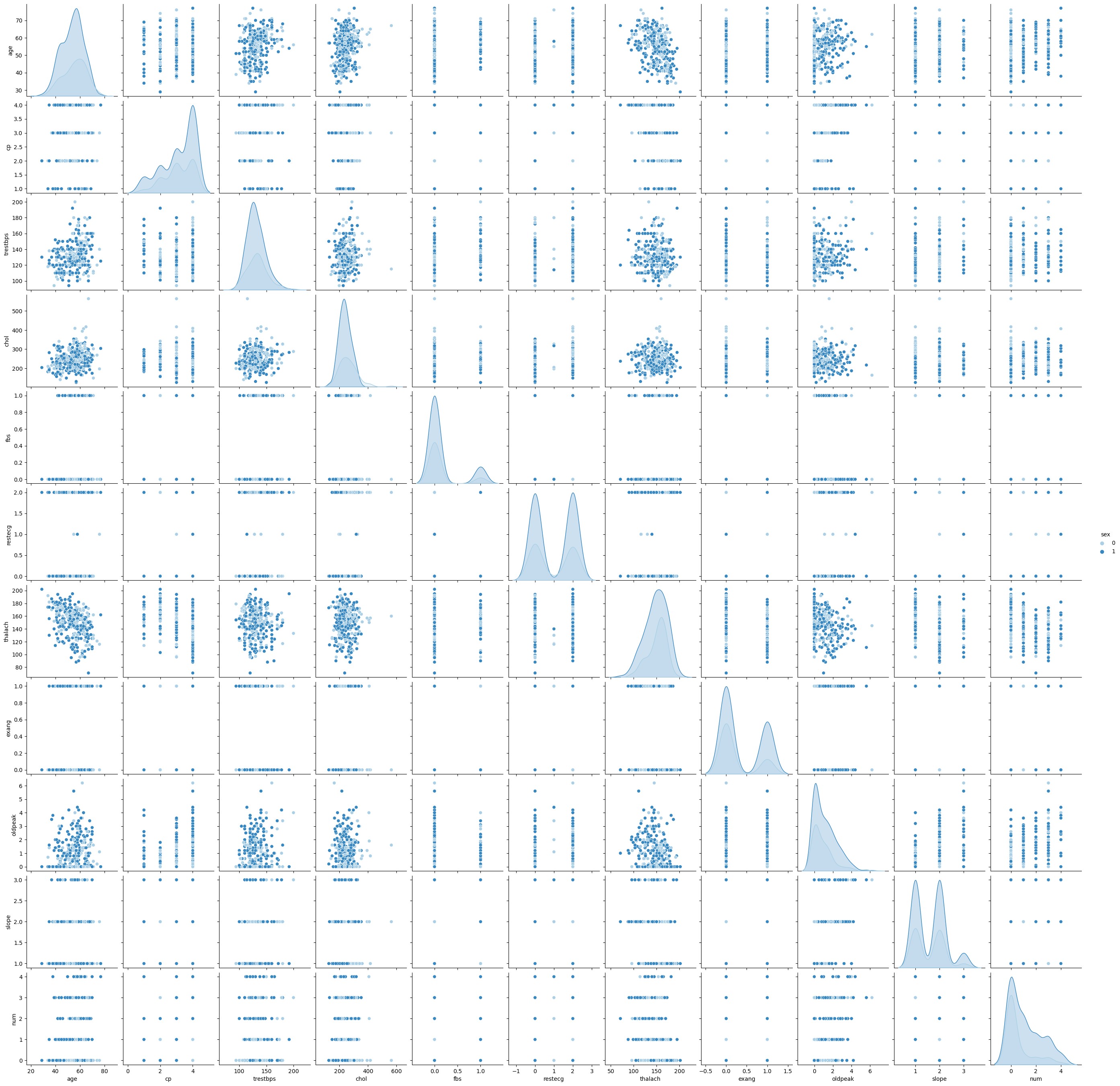
**=**

'Blues'

)

Out[29]: <seaborn.axisgrid.PairGrid at 0xde46aadfd0>

<Figure size 1200x1200 with 0 Axes>



In [ ]:

[25]:

In

[35]:

In

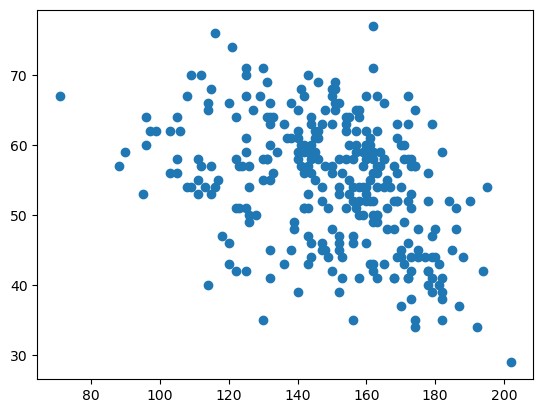
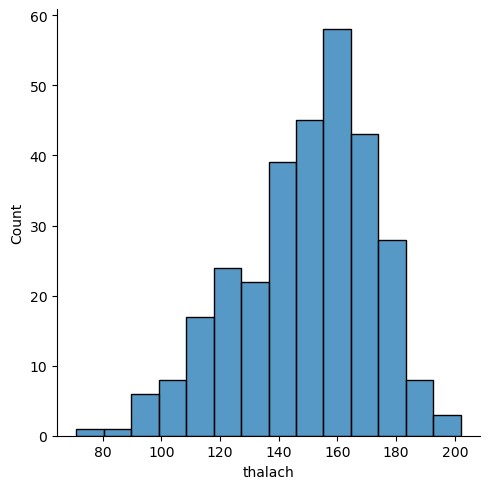
[45]:

Out[25]:

<

seaborn.axisgrid.FacetGrid at 0xde2fc6dd

60>



Out[45]:

**sex**

**1**

206

*#Displot using seaborn*

sns

.

displot

(

df

[

'thalach'

])

*#ScatterPlot using Matplotlib*

plt

.

scatter

(

df

[

'thalach'

]

,

df

[

'age'

])

plt

.

show

()

*#PiePlot using Matplotlib*

sex\_df

**=**

pd

.

DataFrame

(

df

[

'sex'

].

value\_counts

())

sex\_df

In [ ]:

**0** 97

Out[46]: ([<matplotlib.patches.Wedge at 0xde3716a3d0>,

<matplotlib.patches.Wedge at 0xde3716a850>],

[46]:

plt

.

pie

(

sex\_df

[

'sex'

]

,

labels

**=**

sex\_df

.

index

)

[Text(-0.5890242258008583, 0.9290050922463771, '1'),

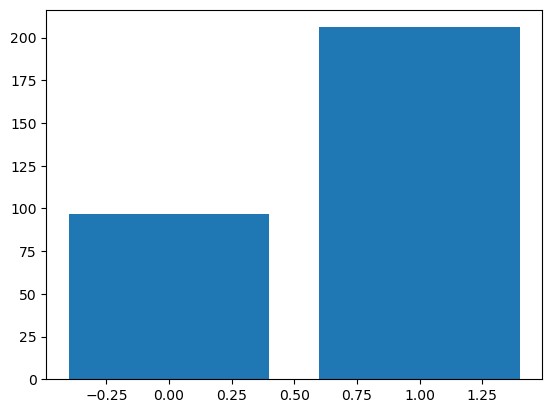
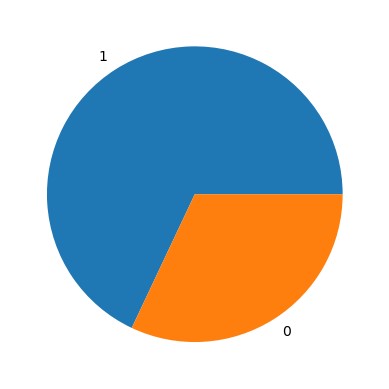
In

[47]:

In

[ ]:

Text(0.5890242258008579, -0.9290050922463774, '0')])



*#Barplot using Matplotlib*

plt

.

bar

(

sex\_df

.

index

,

sex\_df

[

'sex'

])

plt

.

show

()

Create a review scrapper for any ecommerce website to fetch real time comments, reviews, ratings, comment tags, customer name using Python.

**import** pandas **as** pd

**from** bs4 **import** BeautifulSoup

**from** requests **import** get

In [27]:

url**=**'https://www.flipkart.com/search?q=tv+smart+tv&sid=ckf%2Cczl&as=on&as-show=on&otracker=AS\_QueryStore\_OrganicAutoSuggest\_1\_2\_na\_na\_na&otracker1=AS\_QueryStore\_OrganicAutoSuggest\_1\_2\_na\_na\_na&as-pos=1&as-type=RECENT&suggestionId=tv+smart+tv%7CTelevisions&requestId=b12db4dc-db18-45fc-81d3-afdd75727661&as-backfill=on'

url

Out[27]:

'https://www.flipkart.com/search?q=tv+smart+tv&sid=ckf%2Cczl&as=on&as-show=on&otracker=AS\_QueryStore\_OrganicAutoSuggest\_1\_2\_na\_na\_na&otracker1=AS\_QueryStore\_OrganicAutoSuggest\_1\_2\_na\_na\_na&as-pos=1&as-type=RECENT&suggestionId=tv+smart+tv%7CTelevisions&requestId=b12db4dc-db18-45fc-81d3-afdd75727661&as-backfill=on'

In [28]:

response**=**get(url)

response

Out[28]:

<Response [200]>

In [39]:

soup**=**BeautifulSoup(response**.**text,'lxml')

In [40]:

master\_container**=**soup**.**find\_all('div',{'class':'\_2kHMtA'})

In [65]:

tv\_name**=**[]

**for** i **in** range(len(master\_container)):

**try**:

tv\_name**.**append(master\_container[i]**.**find('div',{'class':'\_4rR01T'})**.**text)

**except**:

tv\_name**.**append(**None**)

In [66]:

len(tv\_name)

Out[66]:

24

In [67]:

ratings**=**[]

**for** i **in** range(len(master\_container)):

**try**:

ratings**.**append(master\_container[i]**.**find('div',{'class':'\_3LWZlK'})**.**text)

**except**:

ratings**.**append(**None**)

In [68]:

len(ratings)

Out[68]:

24

In [69]:

price**=**[]

**for** i **in** range(len(master\_container)):

**try**:

price**.**append(master\_container[i]**.**find('div',{'class':'\_30jeq3 \_1\_WHN1'})**.**text)

**except**:

price**.**append(**None**)

In [70]:

len(price)

Out[70]:

24

In [71]:

review**=**[]

**for** i **in** range(len(master\_container)):

**try**:

review**.**append(master\_container[i]**.**find('span',{'class':'\_2\_R\_DZ'})**.**text)

**except**:

review**.**append(**None**)

In [72]:

len(review)

Out[72]:

24

In [73]:

data**=**{"TV":tv\_name,'Ratings':ratings,'Price':price,'Reviews':review}

In [74]:

data**=**pd**.**DataFrame(data)

In [75]:

data

Out[75]:

|  | **TV** | **Ratings** | **Price** | **Reviews** |
| --- | --- | --- | --- | --- |
| **0** | LG 80 cm (32 inch) HD Ready LED Smart WebOS TV | 4.4 | ₹13,999 | 53,655 Ratings & 4,755 Reviews |
| **1** | Mi 5A 80 cm (32 inch) HD Ready LED Smart Andro... | 4.4 | ₹13,999 | 3,51,523 Ratings & 33,161 Reviews |
| **2** | Infinix Y1 80 cm (32 inch) HD Ready LED Smart ... | 4.2 | ₹8,499 | 20,230 Ratings & 2,666 Reviews |
| **3** | OnePlus Y1 80 cm (32 inch) HD Ready LED Smart ... | 4.3 | ₹14,999 | 90,501 Ratings & 7,974 Reviews |
| **4** | realme 80 cm (32 inch) HD Ready LED Smart Andr... | 4.3 | ₹11,999 | 2,47,945 Ratings & 27,853 Reviews |
| **5** | SAMSUNG 80 cm (32 Inch) HD Ready LED Smart Tiz... | 4.4 | ₹13,990 | 85,661 Ratings & 7,438 Reviews |
| **6** | Thomson Alpha 80 cm (32 inch) HD Ready LED Sma... | 4.4 | ₹8,499 | 13,521 Ratings & 2,203 Reviews |
| **7** | Sansui Neo 80 cm (32 inch) HD Ready LED Smart ... | 4.2 | ₹9,990 | 483 Ratings & 74 Reviews |
| **8** | acer I Series 80 cm (32 inch) HD Ready LED Sma... | 4.4 | ₹11,999 | 6,920 Ratings & 1,142 Reviews |
| **9** | OnePlus Y1S 108 cm (43 inch) Full HD LED Smart... | 4.3 | ₹24,999 | 65,539 Ratings & 6,236 Reviews |
| **10** | Sansui 80 cm (32 inch) HD Ready LED Smart Andr... | 4.2 | ₹10,999 | 2,138 Ratings & 215 Reviews |
| **11** | SKYTRON 108 cm (43 inch) Full HD LED Smart And... | 4.2 | ₹15,599 | 11 Ratings & 2 Reviews |
| **12** | SAMSUNG Crystal 4K 108 cm (43 inch) Ultra HD (... | 4.4 | ₹29,990 | 14,205 Ratings & 1,421 Reviews |
| **13** | Coocaa 80 cm (32 inch) HD Ready LED Smart Cool... | 4.2 | ₹7,999 | 19,533 Ratings & 2,970 Reviews |
| **14** | Mi 5A 100 cm (40 inch) Full HD LED Smart Andro... | 4.4 | ₹21,999 | 3,51,523 Ratings & 33,161 Reviews |
| **15** | SKYTRON 108 cm (43 inch) Ultra HD (4K) LED Sma... | 3.9 | ₹21,499 | 96 Ratings & 10 Reviews |
| **16** | realme NEO 80 cm (32 inch) HD Ready LED Smart ... | 4.3 | ₹11,999 | 37,520 Ratings & 3,262 Reviews |
| **17** | OnePlus Y1S 80 cm (32 inch) HD Ready LED Smart... | 4.3 | ₹15,999 | 65,539 Ratings & 6,236 Reviews |
| **18** | SAMSUNG Crystal 4K Neo Series 108 cm (43 inch)... | 4.4 | ₹30,990 | 19,613 Ratings & 2,058 Reviews |
| **19** | Gangnam Street 109 cm (43 inch) Full HD LED Sm... | None | ₹13,499 | None |
| **20** | OnePlus Y1 100 cm (40 inch) Full HD LED Smart ... | 4.3 | ₹21,999 | 65,539 Ratings & 6,236 Reviews |
| **21** | acer 100 cm (40 inch) Full HD LED Smart Androi... | 4.4 | ₹16,999 | 6,920 Ratings & 1,142 Reviews |
| **22** | ONIDA 80 cm (32 inch) HD Ready LED Smart VIDAA TV | 4.3 | ₹11,999 | 1,081 Ratings & 195 Reviews |
| **23** | BeethoSOL 108 cm (43 inch) Full HD LED Smart A... | 4.1 | ₹14,599 | 1,886 Ratings & 197 Reviews |

In [ ]:

13. Perform the following data visualization operations using Tableau on Adult and Iris datasets.

**a.** 1D (Linear) Data visualization

**b.** 2D (Planar) Data Visualization

**c.** 3D (Volumetric) Data Visualization