assigment1

May 8, 2023

[ ]:

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

[3]:
```
import pandas as pd
path='/content/drive/MyDrive/Dataset/Iris.csv'
dt=pd.read_csv(path)
print(dt)
```

```
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   …            …             …              …             …
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[4]:
```
dt.head()
```

[4]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

[5]:
```
dt.tail()
```

[5]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
145	146	6.7	3.0	5.2	2.3	
146	147	6.3	2.5	5.0	1.9	
147	148	6.5	3.0	5.2	2.0	
148	149	6.2	3.4	5.4	2.3	
149	150	5.9	3.0	5.1	1.8	

	Species
145	Iris-virginica
146	Iris-virginica
147	Iris-virginica
148	Iris-virginica
149	Iris-virginica

[6]:
```
s=10
dt.tail(10)
```

[6]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
140	141	6.7	3.1	5.6	2.4	
141	142	6.9	3.1	5.1	2.3	
142	143	5.8	2.7	5.1	1.9	
143	144	6.8	3.2	5.9	2.3	
144	145	6.7	3.3	5.7	2.5	
145	146	6.7	3.0	5.2	2.3	
146	147	6.3	2.5	5.0	1.9	
147	148	6.5	3.0	5.2	2.0	
148	149	6.2	3.4	5.4	2.3	
149	150	5.9	3.0	5.1	1.8	

	Species
140	Iris-virginica
141	Iris-virginica
142	Iris-virginica
143	Iris-virginica
144	Iris-virginica
145	Iris-virginica

```
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica
```

[7]:
```
dt.index
```

[7]:
```
RangeIndex(start=0, stop=150, step=1)
```

[8]:
```
dt.columns
```

[8]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

[10]:
```
dt.shape
```

[10]:
```
(150, 6)
```

[12]:
```
dt.dtypes
```

[12]:
```
Id                int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

[13]:
```
dt.columns.values
```

[13]:
```
array(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm',
       'PetalWidthCm', 'Species'], dtype=object)
```

[15]:
```
dt.describe()
```

[15]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

[16]:
```
dt.describe(include='all')
```

[16]:
```
                Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
count   150.000000     150.000000    150.000000     150.000000    150.000000
unique         NaN            NaN           NaN            NaN           NaN
top            NaN            NaN           NaN            NaN           NaN
freq           NaN            NaN           NaN            NaN           NaN
mean     75.500000       5.843333      3.054000       3.758667      1.198667
std      43.445368       0.828066      0.433594       1.764420      0.763161
min       1.000000       4.300000      2.000000       1.000000      0.100000
25%      38.250000       5.100000      2.800000       1.600000      0.300000
50%      75.500000       5.800000      3.000000       4.350000      1.300000
75%     112.750000       6.400000      3.300000       5.100000      1.800000
max     150.000000       7.900000      4.400000       6.900000      2.500000

            Species
count           150
unique            3
top     Iris-setosa
freq             50
mean            NaN
std             NaN
min             NaN
25%             NaN
50%             NaN
75%             NaN
max             NaN
```

[18]:
```
dt['Id']
```

[18]:
```
0        1
1        2
2        3
3        4
4        5
      ...
145    146
146    147
147    148
148    149
149    150
Name: Id, Length: 150, dtype: int64
```

[19]:
```
dt.iloc[5]
```

[19]:
```
Id                 6
SepalLengthCm    5.4
SepalWidthCm     3.9
PetalLengthCm    1.7
```

```
PetalWidthCm            0.4
Species         Iris-setosa
Name: 5, dtype: object
```

[20]:
```
dt.iloc[1]
```

[20]:
```
Id                        2
SepalLengthCm           4.9
SepalWidthCm            3.0
PetalLengthCm           1.4
PetalWidthCm            0.2
Species         Iris-setosa
Name: 1, dtype: object
```

[21]:
```
dt.iloc[3]
```

[21]:
```
Id                        4
SepalLengthCm           4.6
SepalWidthCm            3.1
PetalLengthCm           1.5
PetalWidthCm            0.2
Species         Iris-setosa
Name: 3, dtype: object
```

[22]:
```
dt.iloc[3,1]
```

[22]:
```
4.6
```

[23]:
```
dt.iloc[3:5,1:2]
```

[23]:
```
   SepalLengthCm
3            4.6
4            5.0
```

[24]:
```
dt.iloc[3:6,1:2]
```

[24]:
```
   SepalLengthCm
3            4.6
4            5.0
5            5.4
```

[25]:
```
dt.iloc[[1,2,5],[0,1]]
```

[25]:
```
   Id  SepalLengthCm
1   2            4.9
2   3            4.7
5   6            5.4
```

[26]:
```
dt.iloc[2,2]
```

[26]:
```
3.2
```

[27]:
```
dt.isnull()
```

[27]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
..	…	…	…	…	…	…
145	False	False	False	False	False	False
146	False	False	False	False	False	False
147	False	False	False	False	False	False
148	False	False	False	False	False	False
149	False	False	False	False	False	False

[150 rows x 6 columns]

[28]:
```
path="/content/drive/MyDrive/Dataset/dirtydata.csv"
dt=pd.read_csv(path)
dt
```

[28]:

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0

21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

[29]:
```
dt.isnull()
```

[29]:

	Duration	Date	Pulse	Maxpulse	Calories
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
5	False	False	False	False	False
6	False	False	False	False	False
7	False	False	False	False	False
8	False	False	False	False	False
9	False	False	False	False	False
10	False	False	False	False	False
11	False	False	False	False	False
12	False	False	False	False	False
13	False	False	False	False	False
14	False	False	False	False	False
15	False	False	False	False	False
16	False	False	False	False	False
17	False	False	False	False	False
18	False	False	False	False	True
19	False	False	False	False	False
20	False	False	False	False	False
21	False	False	False	False	False
22	False	True	False	False	False
23	False	False	False	False	False
24	False	False	False	False	False
25	False	False	False	False	False
26	False	False	False	False	False
27	False	False	False	False	False
28	False	False	False	False	True
29	False	False	False	False	False
30	False	False	False	False	False
31	False	False	False	False	False

[30]:
```
dt.isnull().sum()
```

[30]:
```
Duration    0
Date        1
Pulse       0
Maxpulse    0
Calories    2
dtype: int64
```

[31]:
```
dt.isnull().sum().sum()
```

[31]:
```
3
```

[32]:
```
dt.isna().sum().sum()
```

[32]:
```
3
```

[33]:
```
dt.isna().sum()
```

[33]:
```
Duration    0
Date        1
Pulse       0
Maxpulse    0
Calories    2
dtype: int64
```

[35]:
```
dt.Calories.isnull().sum()
```

[35]:
```
2
```

[36]:
```
from sklearn import preprocessing
import pandas as pd
path="/content/drive/MyDrive/Dataset/Iris.csv"
dt=pd.read_csv(path)
dt.head
#print(dt)
```

[36]:
```
<bound method NDFrame.head of       Id  SepalLengthCm  SepalWidthCm
PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   …            …           …            …           …
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
```

```
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]>
```

[38]:
```
min_max_scaler=preprocessing.MinMaxScaler()
x=dt.iloc[:,:3]
x_scaler=min_max_scaler.fit_transform(x)
dt_normalized =pd.DataFrame(x_scaler)
dt_normalized
```

[38]:
```
            0         1         2
0    0.000000  0.222222  0.625000
1    0.006711  0.166667  0.416667
2    0.013423  0.111111  0.500000
3    0.020134  0.083333  0.458333
4    0.026846  0.194444  0.666667
..        …       …        …
145  0.973154  0.666667  0.416667
146  0.979866  0.555556  0.208333
147  0.986577  0.611111  0.416667
148  0.993289  0.527778  0.583333
149  1.000000  0.444444  0.416667

[150 rows x 3 columns]
```

[39]:
```
dt['Species'].unique()
```

[39]:
```
array(['Iris-setosa', 'Iris-versicolor', 'Iris-virginica'], dtype=object)
```

[40]:
```
label_encoder=preprocessing.LabelEncoder()
dt['Species']=label_encoder.fit_transform(dt['Species'])
dt['Species'].unique()
```

```
[40]: array([0, 1, 2])
```

assigment2

May 8, 2023

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

[1]:
```
import pandas as pd
import numpy as np
bf=pd.read_csv('/content/drive/MyDrive/2-StudentPerformanceTest1.csv')
print(bf)
```

```
   gender  math score  reading score  writing score  Placement Score  \
0  female        72.0           72.0           74.0             78.0
1  female        69.0           90.0           88.0              NaN
2  female        90.0           95.0           93.0             74.0
3    male        47.0           57.0            NaN             78.0
4    male         NaN           78.0           75.0             81.0
5  female        71.0            NaN           78.0             70.0
6    male        12.0           44.0           52.0             12.0
7    male         NaN           65.0           67.0             49.0
8    male         5.0           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2     NaN
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4     NaN
6                      2  Nashik
7                      1    Pune
8                      0     NaN
```

[2]:
```
bf.isnull()
```

[2]:
```
    gender  math score  reading score  writing score  Placement Score  \
0   False       False          False          False            False
1   False       False          False          False             True
2   False       False          False          False            False
3   False       False          False           True            False
4   False        True          False          False            False
```

```
5   False       False           True          False            False
6   False       False          False          False            False
7   False        True          False          False            False
8   False       False          False          False            False

   placement offer count  Region
0                  False   False
1                  False    True
2                  False   False
3                  False   False
4                  False   False
5                  False    True
6                  False   False
7                  False   False
8                  False    True
```

[3]:
```
s=pd.isnull(bf['reading score'])
bf[s]
```

[3]:
```
   gender  math score  reading score  writing score  Placement Score  \
5  female        71.0            NaN           78.0             70.0

   placement offer count Region
5                      4    NaN
```

[4]:
```
s=pd.isnull(bf['Region'])
bf[s]
```

[4]:
```
   gender  math score  reading score  writing score  Placement Score  \
1  female        69.0           90.0           88.0              NaN
5  female        71.0            NaN           78.0             70.0
8    male         5.0           77.0           89.0             55.0

   placement offer count Region
1                      2    NaN
5                      4    NaN
8                      0    NaN
```

[5]:
```
bf.notnull()
```

[5]:
```
   gender  math score  reading score  writing score  Placement Score  \
0    True        True           True           True             True
1    True        True           True           True            False
2    True        True           True           True             True
3    True        True           True          False             True
4    True       False           True           True             True
5    True        True          False           True             True
```

```
6    True        True           True           True             True
7    True       False           True           True             True
8    True        True           True           True             True

   placement offer count  Region
0                   True    True
1                   True   False
2                   True    True
3                   True    True
4                   True    True
5                   True   False
6                   True    True
7                   True    True
8                   True   False
```

[7]:
```
s=pd.notnull(bf['math score'])
bf[s]
```

[7]:
```
   gender  math score  reading score  writing score  Placement Score  \
0  female        72.0           72.0           74.0             78.0
1  female        69.0           90.0           88.0              NaN
2  female        90.0           95.0           93.0             74.0
3    male        47.0           57.0            NaN             78.0
5  female        71.0            NaN           78.0             70.0
6    male        12.0           44.0           52.0             12.0
8    male         5.0           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2     NaN
2                      2  Nashik
3                      1      Na
5                      4     NaN
6                      2  Nashik
8                      0     NaN
```

[8]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
bf['gender']=le.fit_transform(bf['gender'])
newbf=bf
bf
```

[8]:
```
   gender  math score  reading score  writing score  Placement Score  \
0       0        72.0           72.0           74.0             78.0
1       0        69.0           90.0           88.0              NaN
2       0        90.0           95.0           93.0             74.0
3       1        47.0           57.0            NaN             78.0
```

```
4       1         NaN           78.0           75.0             81.0
5       0        71.0            NaN           78.0             70.0
6       1        12.0           44.0           52.0             12.0
7       1         NaN           65.0           67.0             49.0
8       1         5.0           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2     NaN
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4     NaN
6                      2  Nashik
7                      1    Pune
8                      0     NaN
```

[9]:
```
nd=bf
nd.fillna(0)
```

[9]:
```
   gender  math score  reading score  writing score  Placement Score  \
0       0        72.0           72.0           74.0             78.0
1       0        69.0           90.0           88.0              0.0
2       0        90.0           95.0           93.0             74.0
3       1        47.0           57.0            0.0             78.0
4       1         0.0           78.0           75.0             81.0
5       0        71.0            0.0           78.0             70.0
6       1        12.0           44.0           52.0             12.0
7       1         0.0           65.0           67.0             49.0
8       1         5.0           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2       0
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4       0
6                      2  Nashik
7                      1    Pune
8                      0       0
```

[10]:
```
nbf=bf
nbf.fillna(1)
```

[10]:
```
   gender  math score  reading score  writing score  Placement Score  \
0       0        72.0           72.0           74.0             78.0
```

```
1       0        69.0           90.0           88.0            1.0
2       0        90.0           95.0           93.0           74.0
3       1        47.0           57.0            1.0           78.0
4       1         1.0           78.0           75.0           81.0
5       0        71.0            1.0           78.0           70.0
6       1        12.0           44.0           52.0           12.0
7       1         1.0           65.0           67.0           49.0
8       1         5.0           77.0           89.0           55.0

   placement offer count  Region
0                      1    Pune
1                      2       1
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4       1
6                      2  Nashik
7                      1    Pune
8                      0       1
```

[11]:
```
mv=bf['math score'].mean()
bf['math score'].fillna(value=mv,inplace=True)
bf
```

[11]:
```
   gender  math score  reading score  writing score  Placement Score  \
0       0   72.000000           72.0           74.0             78.0
1       0   69.000000           90.0           88.0              NaN
2       0   90.000000           95.0           93.0             74.0
3       1   47.000000           57.0            NaN             78.0
4       1   52.285714           78.0           75.0             81.0
5       0   71.000000            NaN           78.0             70.0
6       1   12.000000           44.0           52.0             12.0
7       1   52.285714           65.0           67.0             49.0
8       1    5.000000           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2     NaN
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4     NaN
6                      2  Nashik
7                      1    Pune
8                      0     NaN
```

[12]:
```
nbf.replace(to_replace=np.nan,value=-99)
```

[12]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	0	72.000000	72.0	74.0	78.0	1	Pune
1	0	69.000000	90.0	88.0	-99.0	2	-99
2	0	90.000000	95.0	93.0	74.0	2	Nashik
3	1	47.000000	57.0	-99.0	78.0	1	Na
4	1	52.285714	78.0	75.0	81.0	3	Pune
5	0	71.000000	-99.0	78.0	70.0	4	-99
6	1	12.000000	44.0	52.0	12.0	2	Nashik
7	1	52.285714	65.0	67.0	49.0	1	Pune
8	1	5.000000	77.0	89.0	55.0	0	-99

[13]:
```
nbf.dropna()
```

[13]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	0	72.000000	72.0	74.0	78.0	1	Pune
2	0	90.000000	95.0	93.0	74.0	2	Nashik
4	1	52.285714	78.0	75.0	81.0	3	Pune
6	1	12.000000	44.0	52.0	12.0	2	Nashik
7	1	52.285714	65.0	67.0	49.0	1	Pune

[14]:
```
nbf.dropna(how='all')
```

[14]:

	gender	math score	reading score	writing score	Placement Score
0	0	72.000000	72.0	74.0	78.0
1	0	69.000000	90.0	88.0	NaN
2	0	90.000000	95.0	93.0	74.0
3	1	47.000000	57.0	NaN	78.0
4	1	52.285714	78.0	75.0	81.0
5	0	71.000000	NaN	78.0	70.0

```
6       1   12.000000           44.0           52.0             12.0
7       1   52.285714           65.0           67.0             49.0
8       1    5.000000           77.0           89.0             55.0

   placement offer count  Region
0                      1    Pune
1                      2     NaN
2                      2  Nashik
3                      1      Na
4                      3    Pune
5                      4     NaN
6                      2  Nashik
7                      1    Pune
8                      0     NaN
```

[15]:
```
nbf.dropna(axis=1)
```

[15]:
```
   gender  math score  placement offer count
0       0   72.000000                      1
1       0   69.000000                      2
2       0   90.000000                      2
3       1   47.000000                      1
4       1   52.285714                      3
5       0   71.000000                      4
6       1   12.000000                      2
7       1   52.285714                      1
8       1    5.000000                      0
```

[16]:
```
nbf.dropna(axis=0)
```

[16]:
```
   gender  math score  reading score  writing score  Placement Score  \
0       0   72.000000           72.0           74.0             78.0
2       0   90.000000           95.0           93.0             74.0
4       1   52.285714           78.0           75.0             81.0
6       1   12.000000           44.0           52.0             12.0
7       1   52.285714           65.0           67.0             49.0

   placement offer count  Region
0                      1    Pune
2                      2  Nashik
4                      3    Pune
6                      2  Nashik
7                      1    Pune
```

[17]:
```
nbf.dropna(axis=0,how='any')
```

[17]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	0	72.000000	72.0	74.0	78.0	1	Pune
2	0	90.000000	95.0	93.0	74.0	2	Nashik
4	1	52.285714	78.0	75.0	81.0	3	Pune
6	1	12.000000	44.0	52.0	12.0	2	Nashik
7	1	52.285714	65.0	67.0	49.0	1	Pune

assignment3

May 8, 2023

[2]:
```
import pandas as pd
path='/content/drive/MyDrive/Iris.csv'
dt=pd.read_csv(path)
print(dt)
```

```
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   …            …             …              …             …
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[4]:
```
import pandas as pd
path='/content/drive/MyDrive/3-Mall_Customers.csv'
dt=pd.read_csv(path)
print(dt)
```

```
     CustomerID   Genre  Age  Annual Income (k$)  Spending Score (1-100)
0             1    Male   19                  15                      39
1             2    Male   21                  15                      81
2             3  Female   20                  16                       6
3             4  Female   23                  16                      77
4             5  Female   31                  17                      40
..          …     …  …                  …                   …
195         196  Female   35                 120                      79
196         197  Female   45                 126                      28
197         198    Male   32                 126                      74
198         199    Male   32                 137                      18
199         200    Male   30                 137                      83

[200 rows x 5 columns]
```

[5]:
```
dt.mean()
```

```
<ipython-input-5-7f6ce5ad3369>:1: FutureWarning: The default value of
numeric_only in DataFrame.mean is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  dt.mean()
```

[5]:
```
CustomerID              100.50
Age                      38.85
Annual Income (k$)       60.56
Spending Score (1-100)   50.20
dtype: float64
```

[6]:
```
dt.loc[:,'Age'].mean()
```

[6]:
```
38.85
```

[7]:
```
dt.mean(axis=1)[0:4]
```

```
<ipython-input-7-08a1733e33b2>:1: FutureWarning: Dropping of nuisance columns in
DataFrame reductions (with 'numeric_only=None') is deprecated; in a future
version this will raise TypeError.  Select only valid columns before calling the
reduction.
  dt.mean(axis=1)[0:4]
```

[7]:
```
0    18.50
1    29.75
2    11.25
3    30.00
dtype: float64
```

[8]:
```
dt.median()
```

```
<ipython-input-8-6fa999c4056d>:1: FutureWarning: The default value of
numeric_only in DataFrame.median is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  dt.median()
```

[8]:
```
CustomerID                100.5
Age                        36.0
Annual Income (k$)         61.5
Spending Score (1-100)     50.0
dtype: float64
```

[9]:
```
dt.loc[:,'Age'].median()
```

[9]:
```
36.0
```

[10]:
```
dt.median(axis=1)[0:5]
```

```
<ipython-input-10-a296fe9d668e>:1: FutureWarning: Dropping of nuisance columns
in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future
version this will raise TypeError.  Select only valid columns before calling the
reduction.
  dt.median(axis=1)[0:5]
```

[10]:
```
0    17.0
1    18.0
2    11.0
3    19.5
4    24.0
dtype: float64
```

[11]:
```
dt.mode()
```

[11]:

	CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Female	32.0	54.0	42.0
1	2	NaN	NaN	78.0	NaN
2	3	NaN	NaN	NaN	NaN
3	4	NaN	NaN	NaN	NaN
4	5	NaN	NaN	NaN	NaN
..	…	…	…	…	…
195	196	NaN	NaN	NaN	NaN
196	197	NaN	NaN	NaN	NaN
197	198	NaN	NaN	NaN	NaN
198	199	NaN	NaN	NaN	NaN

```
199         200     NaN   NaN                  NaN                      NaN

[200 rows x 5 columns]
```

[12]:
```
dt.loc[:,'Annual Income (k$)'].median()
```

[12]:
```
61.5
```

[13]:
```
dt.loc[:,'Annual Income (k$)'].mode()
```

[13]:
```
0    54
1    78
Name: Annual Income (k$), dtype: int64
```

[14]:
```
dt.min()
```

[14]:
```
CustomerID                    1
Genre                    Female
Age                          18
Annual Income (k$)           15
Spending Score (1-100)        1
dtype: object
```

[15]:
```
dt.loc[:,'Age'].min(skipna=False)
```

[15]:
```
18
```

[16]:
```
dt.max()
```

[16]:
```
CustomerID                  200
Genre                      Male
Age                          70
Annual Income (k$)          137
Spending Score (1-100)       99
dtype: object
```

[17]:
```
dt.loc[:,'Age'].max(skipna=False)
```

[17]:
```
70
```

[18]:
```
dt.std()
```

```
<ipython-input-18-2d6b5fd6a2f0>:1: FutureWarning: The default value of
numeric_only in DataFrame.std is deprecated. In a future version, it will
default to False. In addition, specifying 'numeric_only=None' is deprecated.
Select only valid columns or specify the value of numeric_only to silence this
warning.
  dt.std()
```

[18]:
```
CustomerID                57.879185
Age                       13.969007
Annual Income (k$)        26.264721
Spending Score (1-100)    25.823522
dtype: float64
```

[19]:
```
dt.loc[:,'Age'].std(skipna=False)
```

[19]:
```
13.96900733155888
```

[20]:
```
dt.std(axis=1)[0:4]
```

```
<ipython-input-20-43fd5cce4aaa>:1: FutureWarning: Dropping of nuisance columns
in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future
version this will raise TypeError.  Select only valid columns before calling the
reduction.
  dt.std(axis=1)[0:4]
```

[20]:
```
0    15.695010
1    35.074920
2     8.057088
3    32.300671
dtype: float64
```

[21]:
```
dt.groupby(['Genre'])['Age'].mean()
```

[21]:
```
Genre
Female    38.098214
Male      39.806818
Name: Age, dtype: float64
```

[22]:
```
dt.groupby(['Annual Income (k$)'])['Age'].mean()
```

[22]:
```
Annual Income (k$)
15     20.00
16     21.50
17     26.50
18     29.00
19     49.00
       ...
103    35.75
113    35.50
120    41.00
126    38.50
137    31.00
Name: Age, Length: 64, dtype: float64
```

[23]:
```
newdt=dt.rename(columns={'Annual Income (k$)':'Income'},inplace=False)
newdt
#(newdt.groupby(['Genre']).Income.mean())
```

[23]:

	CustomerID	Genre	Age	Income	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
..	…	…	…	…	…
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

[200 rows x 5 columns]

[24]:
```
newdt=dt.rename(columns={'Annual Income (k$)':'Income'},inplace=False)
#newdt
(newdt.groupby(['Genre']).Income.mean())
```

[24]:
```
Genre
Female    59.250000
Male      62.227273
Name: Income, dtype: float64
```

[25]:
```
newdt=dt.rename(columns={'Spending Score (1-100)':'Score'},inplace=False)
newdt
#(newdt.groupby(['Genre']).Score.mean())
```

[25]:

	CustomerID	Genre	Age	Annual Income (k$)	Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
..	…	…	…	…	…
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

[200 rows x 5 columns]

[26]:
```
newdt=dt.rename(columns={'Spending Score (1-100)':'Score'},inplace=False)
#newdt
(newdt.groupby(['Genre']).Score.mean())
```

[26]:
```
Genre
Female    51.526786
Male      48.511364
Name: Score, dtype: float64
```

[27]:
```
from sklearn import preprocessing
en=preprocessing.OneHotEncoder()
en_dt=pd.DataFrame(en.fit_transform(dt[['Genre']]).toarray())
en_dt
```

[27]:

	0	1
0	0.0	1.0
1	0.0	1.0
2	1.0	0.0
3	1.0	0.0
4	1.0	0.0
..	…	…
195	1.0	0.0
196	1.0	0.0
197	0.0	1.0
198	0.0	1.0
199	0.0	1.0

[200 rows x 2 columns]

[28]:
```
from sklearn import preprocessing
en=preprocessing.OneHotEncoder()
en_dt=pd.DataFrame(en.fit_transform(dt[['Annual Income (k$)']]).toarray())
en_dt
```

[28]:

	0	1	2	3	4	5	6	7	8	9	…	54	55	56	\
0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
2	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
3	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
4	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
..	…	…	…	…	…	…	…	…	…	…	…	…	…	…	
195	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
196	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
197	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
198	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	
199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	…	0.0	0.0	0.0	

```
      57   58   59   60   61   62   63
0    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
2    0.0  0.0  0.0  0.0  0.0  0.0  0.0
3    0.0  0.0  0.0  0.0  0.0  0.0  0.0
4    0.0  0.0  0.0  0.0  0.0  0.0  0.0
..   …  …  …  …  …  …  …
195  0.0  0.0  0.0  0.0  1.0  0.0  0.0
196  0.0  0.0  0.0  0.0  0.0  1.0  0.0
197  0.0  0.0  0.0  0.0  0.0  1.0  0.0
198  0.0  0.0  0.0  0.0  0.0  0.0  1.0
199  0.0  0.0  0.0  0.0  0.0  0.0  1.0

[200 rows x 64 columns]
```

[29]:
```
dt_encode=newdt.join(en_dt)
dt_encode
```

[29]:
```
     CustomerID   Genre  Age  Annual Income (k$)  Score    0    1    2    3  \
0             1    Male   19                  15     39  1.0  0.0  0.0  0.0
1             2    Male   21                  15     81  1.0  0.0  0.0  0.0
2             3  Female   20                  16      6  0.0  1.0  0.0  0.0
3             4  Female   23                  16     77  0.0  1.0  0.0  0.0
4             5  Female   31                  17     40  0.0  0.0  1.0  0.0
..          …     …  …                   …     …  …  …  …  …
195         196  Female   35                 120     79  0.0  0.0  0.0  0.0
196         197  Female   45                 126     28  0.0  0.0  0.0  0.0
197         198    Male   32                 126     74  0.0  0.0  0.0  0.0
198         199    Male   32                 137     18  0.0  0.0  0.0  0.0
199         200    Male   30                 137     83  0.0  0.0  0.0  0.0

       4  …   54   55   56   57   58   59   60   61   62   63
0    0.0  …  0.0  0.0  0.0  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  0.0  0.0  0.0  0.0
2    0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
3    0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
4    0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
..   …  …  …  …  …  …  …  …  …  …  …  …
195  0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
196  0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
197  0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
198  0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0
199  0.0  …  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0

[200 rows x 69 columns]
```

[30]:
```
import pandas as pd
path='https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'
dt=pd.read_csv(path)
print(dt)
```

```
     5.1  3.5  1.4  0.2     Iris-setosa
0    4.9  3.0  1.4  0.2     Iris-setosa
1    4.7  3.2  1.3  0.2     Iris-setosa
2    4.6  3.1  1.5  0.2     Iris-setosa
3    5.0  3.6  1.4  0.2     Iris-setosa
4    5.4  3.9  1.7  0.4     Iris-setosa
..   …  …  …  …             …
144  6.7  3.0  5.2  2.3  Iris-virginica
145  6.3  2.5  5.0  1.9  Iris-virginica
146  6.5  3.0  5.2  2.0  Iris-virginica
147  6.2  3.4  5.4  2.3  Iris-virginica
148  5.9  3.0  5.1  1.8  Iris-virginica

[149 rows x 5 columns]
```

[31]:
```
col_names=['Sepal_Length','Sepal_Width','Petal_Length','Petal_Width','Species']
iris=pd.read_csv(path,names=col_names)
irisSet=(iris['Species']=='Iris-setosa')
print('Iris-setosa')
print(iris[irisSet].describe())
```

```
Iris-setosa
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count      50.00000    50.000000     50.000000     50.00000
mean        5.00600     3.418000      1.464000      0.24400
std         0.35249     0.381024      0.173511      0.10721
min         4.30000     2.300000      1.000000      0.10000
25%         4.80000     3.125000      1.400000      0.20000
50%         5.00000     3.400000      1.500000      0.20000
75%         5.20000     3.675000      1.575000      0.30000
max         5.80000     4.400000      1.900000      0.60000
```

[32]:
```
irisVer=(iris['Species']=='Iris-versicolor')
print('Iris-versicolor')
print(iris[irisVer].describe())
```

```
Iris-versicolor
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count     50.000000    50.000000     50.000000    50.000000
mean       5.936000     2.770000      4.260000     1.326000
std        0.516171     0.313798      0.469911     0.197753
min        4.900000     2.000000      3.000000     1.000000
25%        5.600000     2.525000      4.000000     1.200000
```

```
50%         5.900000     2.800000      4.350000     1.300000
75%         6.300000     3.000000      4.600000     1.500000
max         7.000000     3.400000      5.100000     1.800000
```

[33]:
```
irisVir=(iris['Species']=='Iris-virginica')
print('Iris-virginica')
print(iris[irisVir].describe())
```

```
Iris-virginica
       Sepal_Length  Sepal_Width  Petal_Length  Petal_Width
count      50.00000    50.000000     50.000000     50.00000
mean        6.58800     2.974000      5.552000      2.02600
std         0.63588     0.322497      0.551895      0.27465
min         4.90000     2.200000      4.500000      1.40000
25%         6.22500     2.800000      5.100000      1.80000
50%         6.50000     3.000000      5.550000      2.00000
75%         6.90000     3.175000      5.875000      2.30000
max         7.90000     3.800000      6.900000      2.50000
```

assignment4

May 8, 2023

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[ ]:
```
x=np.array([95,85,80,70,60])
y=np.array([85,95,70,65,70])
```

[ ]:
```
model=np.polyfit(x,y,1)
```

[ ]:
```
model
```

[ ]:
```
array([ 0.64383562, 26.78082192])
```

[ ]:
```
predict=np.poly1d(model)
predict(65)
```

[ ]:
```
68.63013698630135
```

[ ]:
```
y_pred=predict(x)
y_pred
```

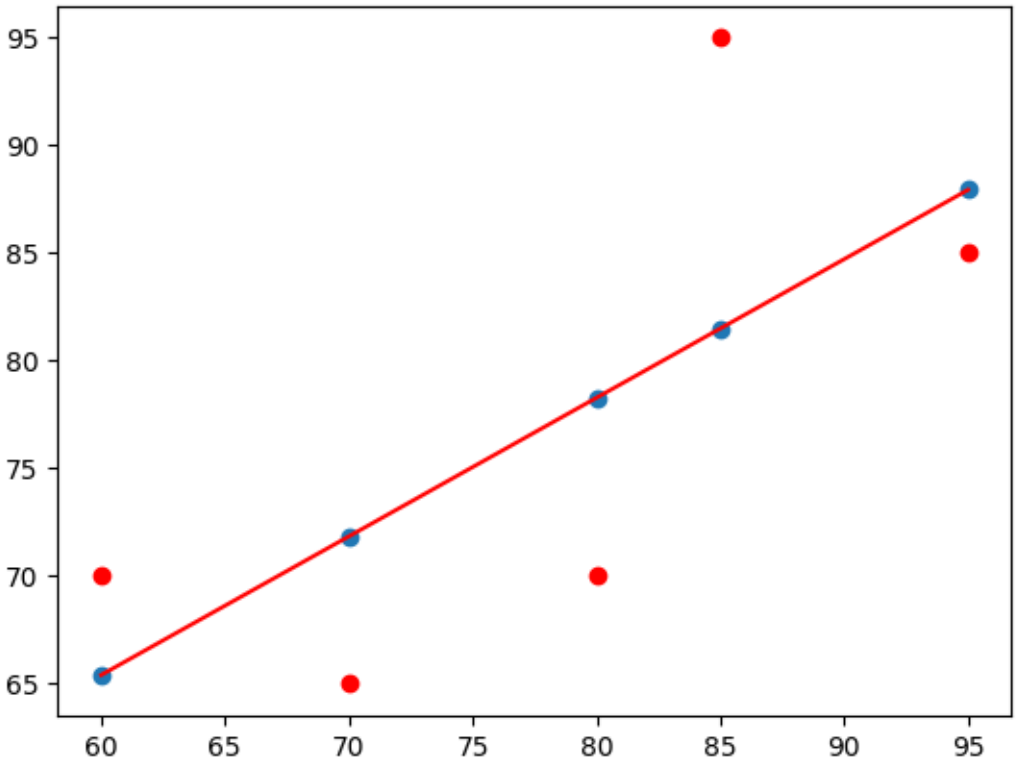
[ ]:
```
array([87.94520548, 81.50684932, 78.28767123, 71.84931507, 65.4109589 ])
```

[ ]:
```
from sklearn.metrics import r2_score
r2_score(y,y_pred)
```

[ ]:
```
0.4803218090889323
```

[ ]:
```
y_line=model[1]+model[0]*x
plt.plot(x,y_line,c='r')
plt.scatter(x,y_pred)
plt.scatter(x,y,c='r')
```

[ ]:
```
<matplotlib.collections.PathCollection at 0x7f923b77a940>
```



[ ]:
```
import numpy as np
import pandas as pd
df=pd.read_csv('/content/drive/MyDrive/boston_train.csv')
print(df)
```

```
         CRIM    ZN  INDUS  CHAS     NOX     RM    AGE     DIS  RAD  TAX  \
0     0.40771   0.0   6.20     1  0.5070  6.164   91.3  3.0480    8  307
1    19.60910   0.0  18.10     0  0.6710  7.313   97.9  1.3163   24  666
2     6.71772   0.0  18.10     0  0.7130  6.749   92.6  2.3236   24  666
3     1.51902   0.0  19.58     1  0.6050  8.375   93.9  2.1620    5  403
4     9.59571   0.0  18.10     0  0.6930  6.404  100.0  1.6390   24  666
..        ...   ...    ...   ...     ...    ...    ...     ...  ...  ...
399   0.02009  95.0   2.68     0  0.4161  8.034   31.9  5.1180    4  224
400   0.04981  21.0   5.64     0  0.4390  5.998   21.4  6.8147    4  243
401   0.08199   0.0  13.92     0  0.4370  6.009   42.3  5.5027    4  289
402   0.37578   0.0  10.59     1  0.4890  5.404   88.6  3.6650    4  277
403   0.10000  34.0   6.09     0  0.4330  6.982   17.7  5.4917    7  329

     PTRATIO       B  LSTAT  PRICE
0       17.4  395.24  21.46   21.7
1       20.2  396.90  13.44   15.0
2       20.2    0.32  17.44   13.4
```

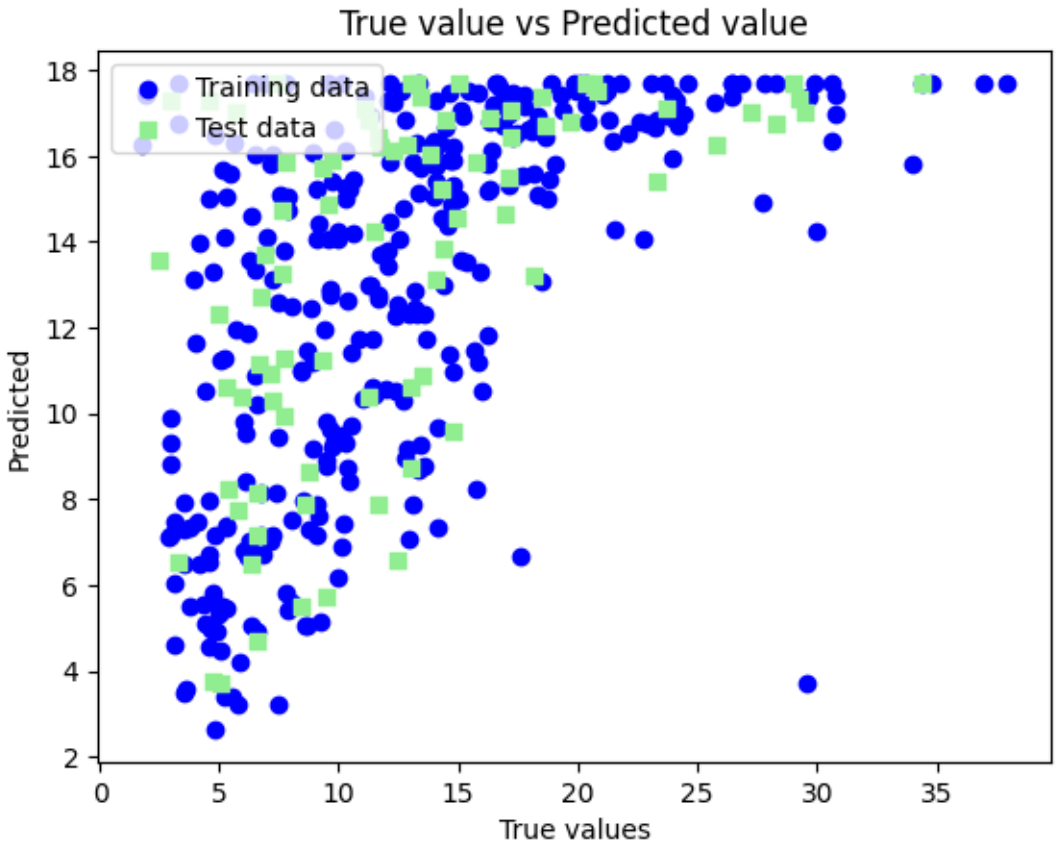
```
3       14.7  388.45   3.32   50.0
4       20.2  376.11  20.31   12.1
..       …     …    …     …
399     14.7  390.55   2.88   50.0
400     16.8  396.90   8.43   23.4
401     16.0  396.90  10.40   21.7
402     18.6  395.24  23.98   19.3
403     16.1  390.43   4.86   33.1

[404 rows x 14 columns]
```

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
boston=pd.read_csv('/content/drive/MyDrive/boston_train.csv')
#from sklearn.datasets import load_boston
#boston=load_boston()
data=pd.DataFrame(boston.AGE)
#data.columns=boston.age
data.head()
data['PRICE']=boston.LSTAT
data.isnull().sum()
x=data.drop(['PRICE'],axis=1)
y=data['PRICE']
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x,y,test_size=0.
 ↪2,random_state=0)
import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
model=lm.fit(xtrain,ytrain)
ytrain_pred=lm.predict(xtrain)
ytest_pred=lm.predict(xtest)
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
from sklearn.metrics import mean_squared_error,r2_score
mse=mean_squared_error(ytest,ytest_pred)
print(mse)
#mse=mean_squared_error(ytest.ytest_pred)
#print(mse)
```

```
34.86541566480911
```

[2]:
```
plt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training data')
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test data')
plt.xlabel('True values')
plt.ylabel('Predicted')
```

```
plt.title("True value vs Predicted value")
plt.legend(loc= 'upper left')
#plt.hlines(y=0,xmin=0,xmax=50)
plt.plot()
plt.show()
```



assignment5

May 8, 2023

[1]:
```
import pandas as pd
import numpy as np
path='/content/drive/MyDrive/5-social_network_ads.csv'
ss=pd.read_csv(path)
print(ss)
```

```
      User ID  Gender  Age  EstimatedSalary  Purchased
0    15624510    Male   19            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
..        …     …  …              …         …
395  15691863  Female   46            41000          1
396  15706071    Male   51            23000          1
397  15654296  Female   50            20000          1
398  15755018    Male   36            33000          0
399  15594041  Female   49            36000          1

[400 rows x 5 columns]
```

[2]:
```
ss.isnull()
```

[2]:
```
     User ID  Gender    Age  EstimatedSalary  Purchased
0      False   False  False            False      False
1      False   False  False            False      False
2      False   False  False            False      False
3      False   False  False            False      False
4      False   False  False            False      False
..       …     …    …              …         …
395    False   False  False            False      False
396    False   False  False            False      False
397    False   False  False            False      False
398    False   False  False            False      False
399    False   False  False            False      False

[400 rows x 5 columns]
```

[3]:
```
ss.head()
```

[3]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	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

[4]:
```
ss['Gender'].value_counts()
```

[4]:
```
Female    204
Male      196
Name: Gender, dtype: int64
```

[5]:
```
#seprating independent variables and dependent variables
X=ss.drop(['Gender'],axis=1)
Y=ss['Gender']
print(X.shape)
print(Y.shape)
```

```
(400, 4)
(400,)
```

[6]:
```
#spitting the module  model_selection
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state=0)
```

[7]:
```
#to know the shape of the train and test dataset
print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(280, 4)
(280,)
(120, 4)
(120,)
```

[8]:
```
#use support vector classifier as classifier
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix

clf=SVC(kernel='linear').fit(X_train,Y_train)
clf.predict(X_train)
```

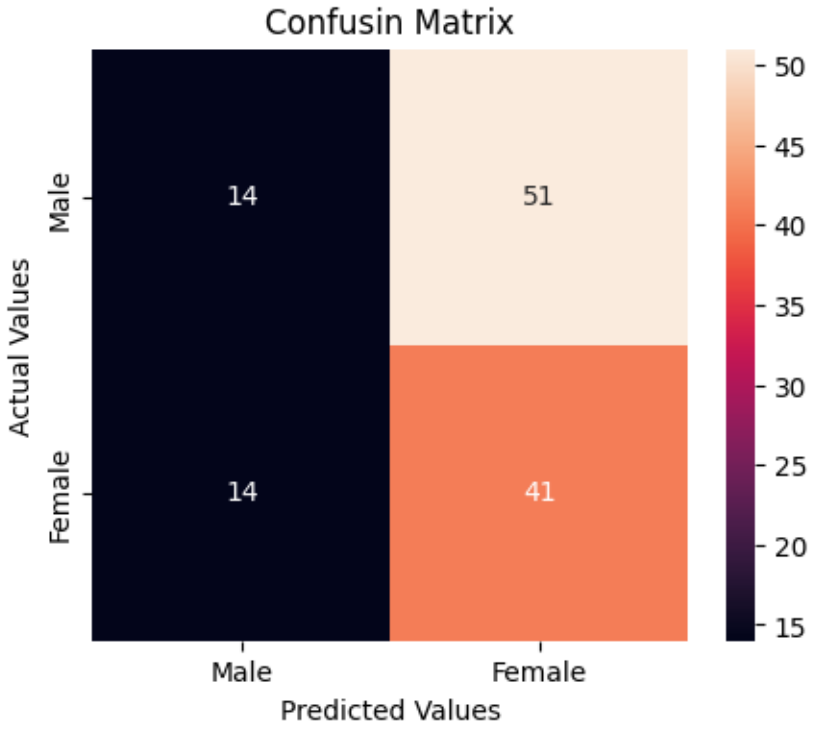
[8]:
```
array(['Female', 'Male', 'Male', 'Female', 'Female', 'Male', 'Male',
       'Female', 'Female', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male',
       'Male', 'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Female', 'Male', 'Female', 'Female', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Female',
       'Male', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Female', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Female', 'Male',
       'Male', 'Female', 'Female', 'Male', 'Female', 'Male', 'Male',
       'Male', 'Male', 'Female', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Female', 'Male',
       'Female', 'Female', 'Female', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Male', 'Female',
       'Male', 'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Male',
       'Female', 'Male', 'Female', 'Male', 'Male', 'Male', 'Male', 'Male',
       'Female', 'Male', 'Male', 'Female', 'Male', 'Male', 'Female',
       'Female', 'Male', 'Male', 'Female', 'Female', 'Female', 'Male',
       'Male', 'Male', 'Male', 'Male', 'Female', 'Male', 'Female',
       'Female'], dtype=object)
```

[9]:
```
import matplotlib.pyplot as plt
import seaborn as sns
#testing the model using X_test and storing the output in Y_pred
Y_pred=clf.predict(X_test)
#creating confusion matrix,whichg compare Y_test and Y_pred
cm=confusion_matrix(Y_test,Y_pred)
#creating s dataframe for aray-formatted confusion matrix will be
cm_ss=pd.DataFrame(cm,index=['Male','Female'],columns=['Male','Female'])
plt.figure(figsize=(5,4))
```

[9]: <Figure size 500x400 with 0 Axes>

<Figure size 500x400 with 0 Axes>

[10]:
```
#plotting the confusion matrix
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(5,4))
sns.heatmap(cm_ss,annot=True)
plt.title('Confusin Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
```



assignment6

May 8, 2023

[2]:
```
import pandas as pd
import numpy as np
path='/content/drive/MyDrive/Iris.csv'
ss=pd.read_csv(path)
print(ss)
```

```
      Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0      1            5.1           3.5            1.4           0.2
1      2            4.9           3.0            1.4           0.2
2      3            4.7           3.2            1.3           0.2
3      4            4.6           3.1            1.5           0.2
4      5            5.0           3.6            1.4           0.2
..   …              …             …              …             …
145  146            6.7           3.0            5.2           2.3
146  147            6.3           2.5            5.0           1.9
147  148            6.5           3.0            5.2           2.0
148  149            6.2           3.4            5.4           2.3
149  150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              …
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[3]:
```
ss.isnull()
```

[3]:
```
        Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  Species
0    False          False         False          False         False    False
1    False          False         False          False         False    False
2    False          False         False          False         False    False
3    False          False         False          False         False    False
4    False          False         False          False         False    False
..     …              …             …              …             …        …
145  False          False         False          False         False    False
146  False          False         False          False         False    False
147  False          False         False          False         False    False
148  False          False         False          False         False    False
149  False          False         False          False         False    False

[150 rows x 6 columns]
```

[4]:
```
ss.head()
```

[4]:
```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa
```

[5]:
```
ss['Species'].value_counts()
```

[5]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: Species, dtype: int64
```

[6]:
```
#seprating independent variables and dependent variables
X=ss.drop(['Species'],axis=1)
Y=ss['Species']
print(X.shape)
print(Y.shape)
```

```
(150, 5)
(150,)
```

[7]:
```
#spitting the module  model_selection
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,random_state=0)
```

[8]:
```
#to know the shape of the train and test dataset
print(X_train.shape)
print(Y_train.shape)
```

```
print(X_test.shape)
print(Y_test.shape)
```

```
(105, 5)
(105,)
(45, 5)
(45,)
```

[9]:
```
#use support vector classifier as classifier
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix

clf=SVC(kernel='linear').fit(X_train,Y_train)
clf.predict(X_train)
```

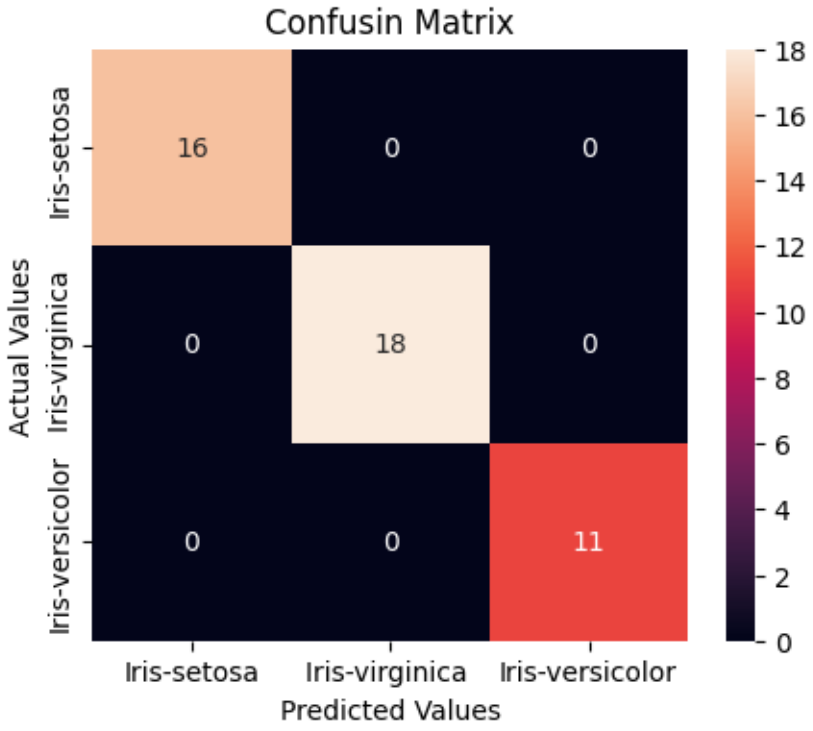
[9]:
```
array(['Iris-versicolor', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-virginica',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-virginica', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa', 'Iris-setosa',
       'Iris-versicolor', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-setosa', 'Iris-versicolor',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-setosa', 'Iris-virginica', 'Iris-setosa',
       'Iris-virginica', 'Iris-virginica', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-versicolor', 'Iris-virginica', 'Iris-virginica',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-setosa', 'Iris-virginica', 'Iris-versicolor',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-virginica', 'Iris-setosa',
       'Iris-setosa', 'Iris-virginica', 'Iris-setosa', 'Iris-virginica',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-virginica', 'Iris-virginica', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-versicolor',
       'Iris-virginica', 'Iris-virginica', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-versicolor', 'Iris-virginica',
       'Iris-setosa'], dtype=object)
```

[10]:
```
import matplotlib.pyplot as plt
import seaborn as sns
#testing the model using X_test and storing the output in Y_pred
Y_pred=clf.predict(X_test)
#creating confusion matrix,whichg compare Y_test and Y_pred
cm=confusion_matrix(Y_test,Y_pred)
#creating s dataframe for aray-formatted confusion matrix will be
cm_ss=pd.DataFrame(cm,index=['Iris-setosa','Iris-virginica␣
↪','Iris-versicolor'],columns=['Iris-setosa','Iris-virginica␣
↪','Iris-versicolor'])
plt.figure(figsize=(5,4))
```

[10]:
```
<Figure size 500x400 with 0 Axes>
```

```
<Figure size 500x400 with 0 Axes>
```

[11]:
```
#plotting the confusion matrix
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(5,4))
sns.heatmap(cm_ss,annot=True)
plt.title('Confusin Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
```



assignment7

May 8, 2023

[ ]:
```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data…
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data…
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

[ ]:
```
True
```

[ ]:
```
text="Tokenization is the first step in text analytics. The process of breaking␣
 ↪down a text paragraph into smaller chunks such or sentences is called␣
 ↪Tokenization."
from nltk import sent_tokenize
tokenized_text=sent_tokenize(text)
tokenized_text
```

[ ]:
```
['Tokenization is the first step in text analytics.',
 'The process of breaking down a text paragraph into smaller chunks such or
sentences is called Tokenization.']
```

[ ]:
```
from nltk import word_tokenize
tokenized_word=word_tokenize(text)
tokenized_word
```

[ ]:
```
['Tokenization',
 'is',
 'the',
 'first',
 'step',
 'in',
```

```
 'text',
 'analytics',
 '.',
 'The',
 'process',
 'of',
 'breaking',
 'down',
 'a',
 'text',
 'paragraph',
 'into',
 'smaller',
 'chunks',
 'such',
 'or',
 'sentences',
 'is',
 'called',
 'Tokenization',
 '.']
```

[ ]:
```
import re
text= "How to remove stop words with NLTK library in Python?"
text= re.sub('[^a-zA-Z]', ' ',text)
text
```

[ ]:
```
'How to remove stop words with NLTK library in Python '
```

[ ]:
```
tokens=word_tokenize(text.lower())
tokens
```

[ ]:
```
['how',
 'to',
 'remove',
 'stop',
 'words',
 'with',
 'nltk',
 'library',
 'in',
 'python']
```

[ ]:
```
filtered_text=[]
for w in tokens:
  if w not in stop_words:
    filtered_text.append(w)
```

```
print("Tokenized Sentence:",tokens)
print("Filterd Sentence:",filtered_text)
```

```
Tokenized Sentence: ['how', 'to', 'remove', 'stop', 'words', 'with', 'nltk',
'library', 'in', 'python']
Filterd Sentence: ['remove', 'stop', 'words', 'nltk', 'library', 'python']
```

[ ]:
```
from nltk.stem import PorterStemmer
e_words=["studies", "studying", "cries", "cry"]
    #'watch','watching','watches','watched']
ps=PorterStemmer()
for w in e_words:
  rootword=ps.stem(w)
  print(rootword)
```

```
studi
studi
cri
cri
```

[ ]:
```
from nltk.stem import WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()
text = "studies studying cries cry"
tokenization = nltk.word_tokenize(text)
for w in tokenization:
 print("Lemma for {} is {}".format(w, wordnet_lemmatizer.lemmatize(w)))
```

```
Lemma for studies is study
Lemma for studying is studying
Lemma for cries is cry
Lemma for cry is cry
```

[ ]:
```
import nltk
from nltk.tokenize import word_tokenize
data="The pink sweater fit her perfectly"
words=word_tokenize(data)
for word in words:
  print(nltk.pos_tag([word]))
```

```
[('The', 'DT')]
[('pink', 'NN')]
[('sweater', 'NN')]
[('fit', 'NN')]
[('her', 'PRP$')]
[('perfectly', 'RB')]
```

[2]:
```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
```

[3]:
```
documentA = 'Jupiter is the largest Planet'
documentB = 'Mars is the fourth planet from the Sun'
```

[5]:
```
bagOfWordsA = documentA.split(' ')
bagOfWordsB = documentB.split(' ')
```

[6]:
```
uniqueWords = set(bagOfWordsA).union(set(bagOfWordsB))
uniqueWords
```

[6]:
```
{'Jupiter',
 'Mars',
 'Planet',
 'Sun',
 'fourth',
 'from',
 'is',
 'largest',
 'planet',
 'the'}
```

[8]:
```
numOfWordsA = dict.fromkeys(uniqueWords, 0)
for word in bagOfWordsA:
  numOfWordsA[word] += 1
#numOfWordsA
```

[9]:
```
numOfWordsB = dict.fromkeys(uniqueWords, 0)
for word in bagOfWordsB:
  numOfWordsB[word] += 1
```

[11]:
```
def computeTF(wordDict,bagOfWords):
  TfDict={}
  bagOfWordCount=len(bagOfWords)
  for word,count in wordDict.items():
    TfDict[word]=count/float(bagOfWordCount)
  return TfDict
```

[12]:
```
tfA=computeTF(numOfWordsA,bagOfWordsA)
tfB=computeTF(numOfWordsB,bagOfWordsB)
tfA
```

[12]:
```
{'is': 0.2,
 'fourth': 0.0,
 'Sun': 0.0,
 'planet': 0.0,
 'Planet': 0.2,
 'Mars': 0.0,
 'largest': 0.2,
```

```
 'the': 0.2,
 'Jupiter': 0.2,
 'from': 0.0}
```

[13]:
```
tfB
```

[13]:
```
{'is': 0.125,
 'fourth': 0.125,
 'Sun': 0.125,
 'planet': 0.125,
 'Planet': 0.0,
 'Mars': 0.125,
 'largest': 0.0,
 'the': 0.25,
 'Jupiter': 0.0,
 'from': 0.125}
```

[14]:
```
def computeIDF(documents):
  import math
  N = len(documents)
  idfDict = dict.fromkeys(documents[0].keys(), 0)
  for document in documents:
    for word, val in document.items():
      if val > 0:
        idfDict[word] += 1
  for word, val in idfDict.items():
    idfDict[word] = math.log(N / float(val))
  return idfDict
```

[15]:
```
idfs = computeIDF([numOfWordsA, numOfWordsB])
idfs
```

[15]:
```
{'is': 0.0,
 'fourth': 0.6931471805599453,
 'Sun': 0.6931471805599453,
 'planet': 0.6931471805599453,
 'Planet': 0.6931471805599453,
 'Mars': 0.6931471805599453,
 'largest': 0.6931471805599453,
 'the': 0.0,
 'Jupiter': 0.6931471805599453,
 'from': 0.6931471805599453}
```

[17]:
```
def computeTFIDF(tfBagOfWords, idfs):
  tfidf = {}
  for word, val in tfBagOfWords.items():
    tfidf[word] = val * idfs[word]
```

```
  return tfidf
```

[18]:
```
tfidfA = computeTFIDF(tfA, idfs)
tfidfA
```

[18]:
```
{'is': 0.0,
 'fourth': 0.0,
 'Sun': 0.0,
 'planet': 0.0,
 'Planet': 0.13862943611198905,
 'Mars': 0.0,
 'largest': 0.13862943611198905,
 'the': 0.0,
 'Jupiter': 0.13862943611198905,
 'from': 0.0}
```

[19]:
```
tfidfB = computeTFIDF(tfB, idfs)
tfidfB
```

[19]:
```
{'is': 0.0,
 'fourth': 0.08664339756999316,
 'Sun': 0.08664339756999316,
 'planet': 0.08664339756999316,
 'Planet': 0.0,
 'Mars': 0.08664339756999316,
 'largest': 0.0,
 'the': 0.0,
 'Jupiter': 0.0,
 'from': 0.08664339756999316}
```

[21]:
```
import pandas as pd
df = pd.DataFrame([tfidfA, tfidfB])
df
```

[21]:

	is	fourth	Sun	planet	Planet	Mars	largest	the	Jupiter	from
0	0.0	0.000000	0.000000	0.000000	0.138629	0.000000	0.138629	0.0	0.138629	0.000000
1	0.0	0.086643	0.086643	0.086643	0.000000	0.086643	0.000000	0.0	0.000000	0.086643

assignment8

May 8, 2023

[1]:
```
pip install seaborn
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-
packages (0.12.2)
Requirement already satisfied: pandas>=0.25 in /usr/local/lib/python3.10/dist-
packages (from seaborn) (1.5.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in
/usr/local/lib/python3.10/dist-packages (from seaborn) (3.7.1)
Requirement already satisfied: numpy!=1.24.0,>=1.17 in
/usr/local/lib/python3.10/dist-packages (from seaborn) (1.22.4)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib!=3.6.1,>=3.1->seaborn) (0.11.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(1.0.7)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(3.0.9)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(1.4.4)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib!=3.6.1,>=3.1->seaborn) (8.4.0)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(23.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(4.39.3)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib!=3.6.1,>=3.1->seaborn)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas>=0.25->seaborn) (2022.7.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.1->seaborn) (1.16.0)
```

[3]:
```
pip install seaborn[stats]
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-
wheels/public/simple/
Requirement already satisfied: seaborn[stats] in /usr/local/lib/python3.10/dist-
packages (0.12.2)
Requirement already satisfied: numpy!=1.24.0,>=1.17 in
/usr/local/lib/python3.10/dist-packages (from seaborn[stats]) (1.22.4)
Requirement already satisfied: matplotlib!=3.6.1,>=3.1 in
/usr/local/lib/python3.10/dist-packages (from seaborn[stats]) (3.7.1)
Requirement already satisfied: pandas>=0.25 in /usr/local/lib/python3.10/dist-
packages (from seaborn[stats]) (1.5.3)
Requirement already satisfied: scipy>=1.3 in /usr/local/lib/python3.10/dist-
packages (from seaborn[stats]) (1.10.1)
Requirement already satisfied: statsmodels>=0.10 in
/usr/local/lib/python3.10/dist-packages (from seaborn[stats]) (0.13.5)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (4.39.3)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (2.8.2)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (1.0.7)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (3.0.9)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib!=3.6.1,>=3.1->seaborn[stats]) (0.11.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (1.4.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from
matplotlib!=3.6.1,>=3.1->seaborn[stats]) (23.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-
packages (from matplotlib!=3.6.1,>=3.1->seaborn[stats]) (8.4.0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas>=0.25->seaborn[stats]) (2022.7.1)
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.10/dist-
packages (from statsmodels>=0.10->seaborn[stats]) (0.5.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages
(from patsy>=0.5.2->statsmodels>=0.10->seaborn[stats]) (1.16.0)
```

[4]:
```
import pandas as pd
import numpy as np
```

```
import matplotlib.pyplot as plt
import seaborn as sb
ds=sb.load_dataset('titanic')
ds.head()
```

[4]:

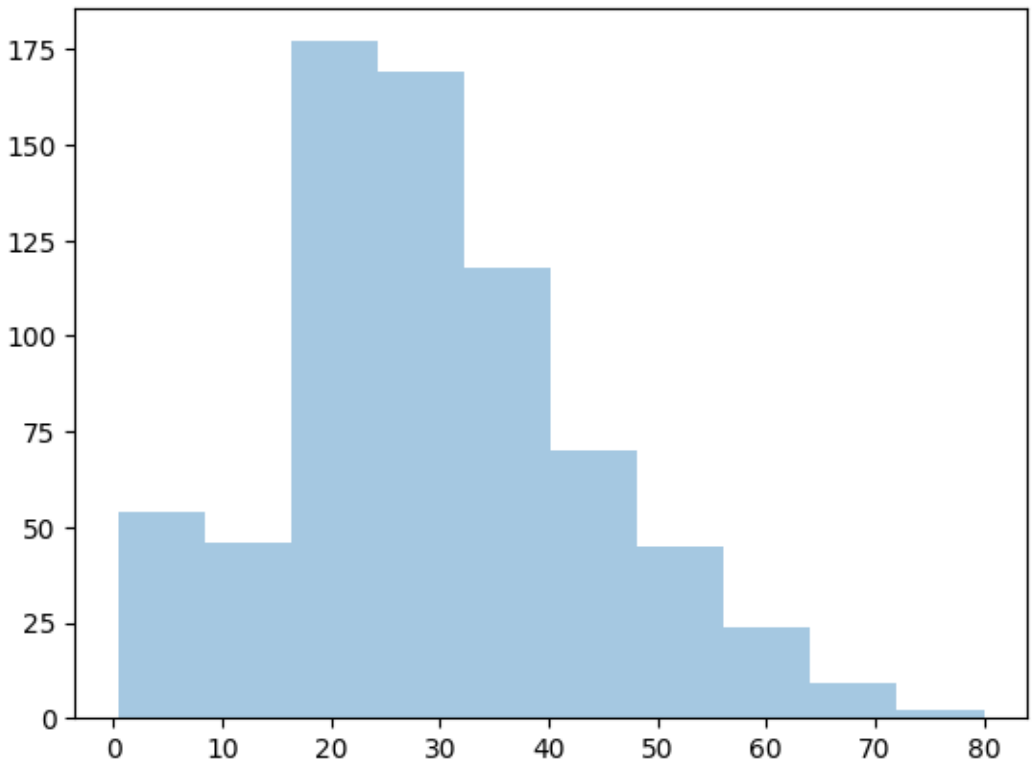
	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	\
0	0	3	male	22.0	1	0	7.2500	S	Third	
1	1	1	female	38.0	1	0	71.2833	C	First	
2	1	3	female	26.0	0	0	7.9250	S	Third	
3	1	1	female	35.0	1	0	53.1000	S	First	
4	0	3	male	35.0	0	0	8.0500	S	Third	

	who	adult_male	deck	embark_town	alive	alone
0	man	True	NaN	Southampton	no	False
1	woman	False	C	Cherbourg	yes	False
2	woman	False	NaN	Southampton	yes	True
3	woman	False	C	Southampton	yes	False
4	man	True	NaN	Southampton	no	True

[5]:
```
import seaborn as sb
sb.distplot(x = ds['age'], bins = 10, kde=False )
```

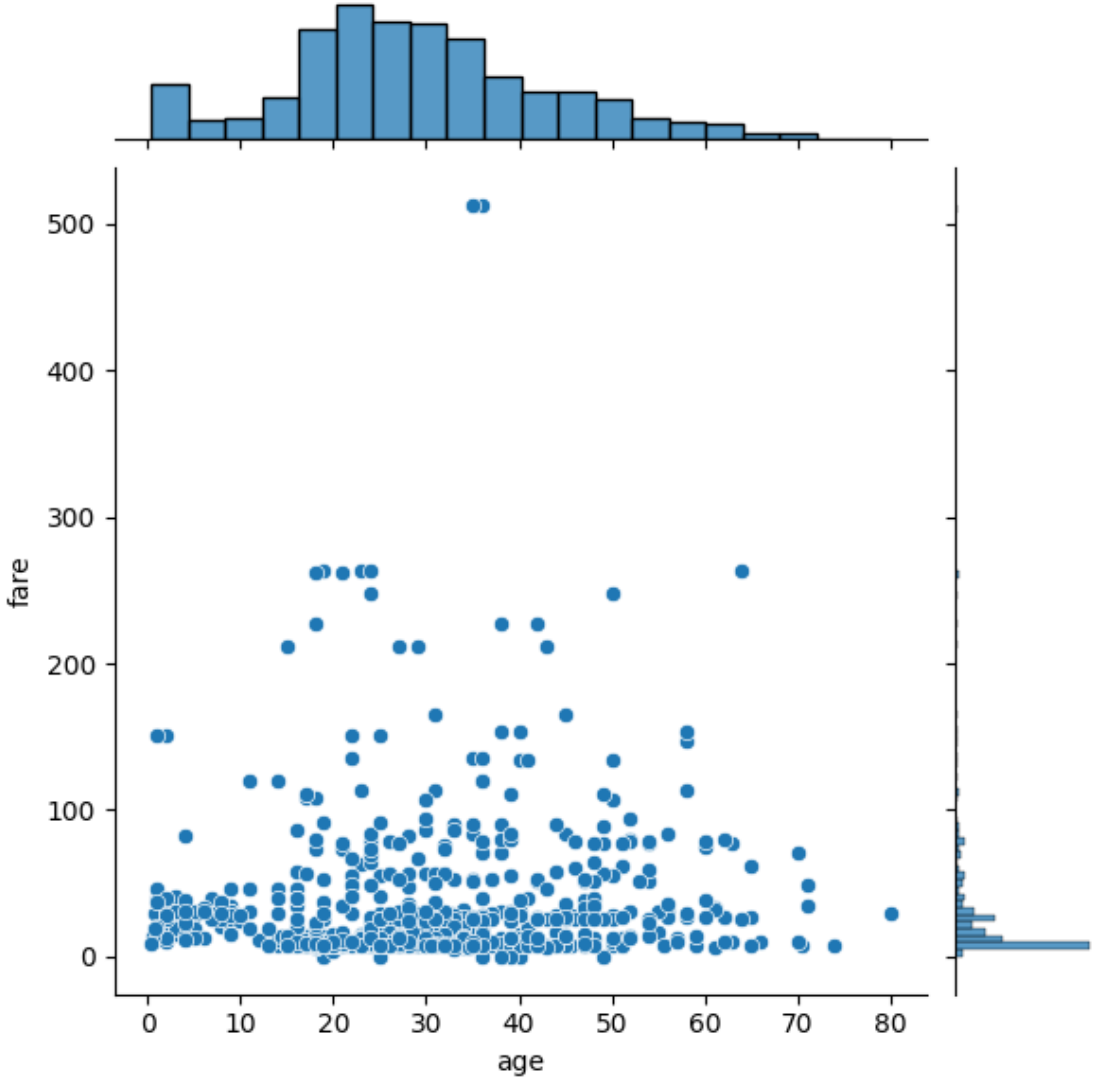
```
<ipython-input-5-e85ac6451960>:2: UserWarning:

`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

  sb.distplot(x = ds['age'], bins = 10, kde=False )
```

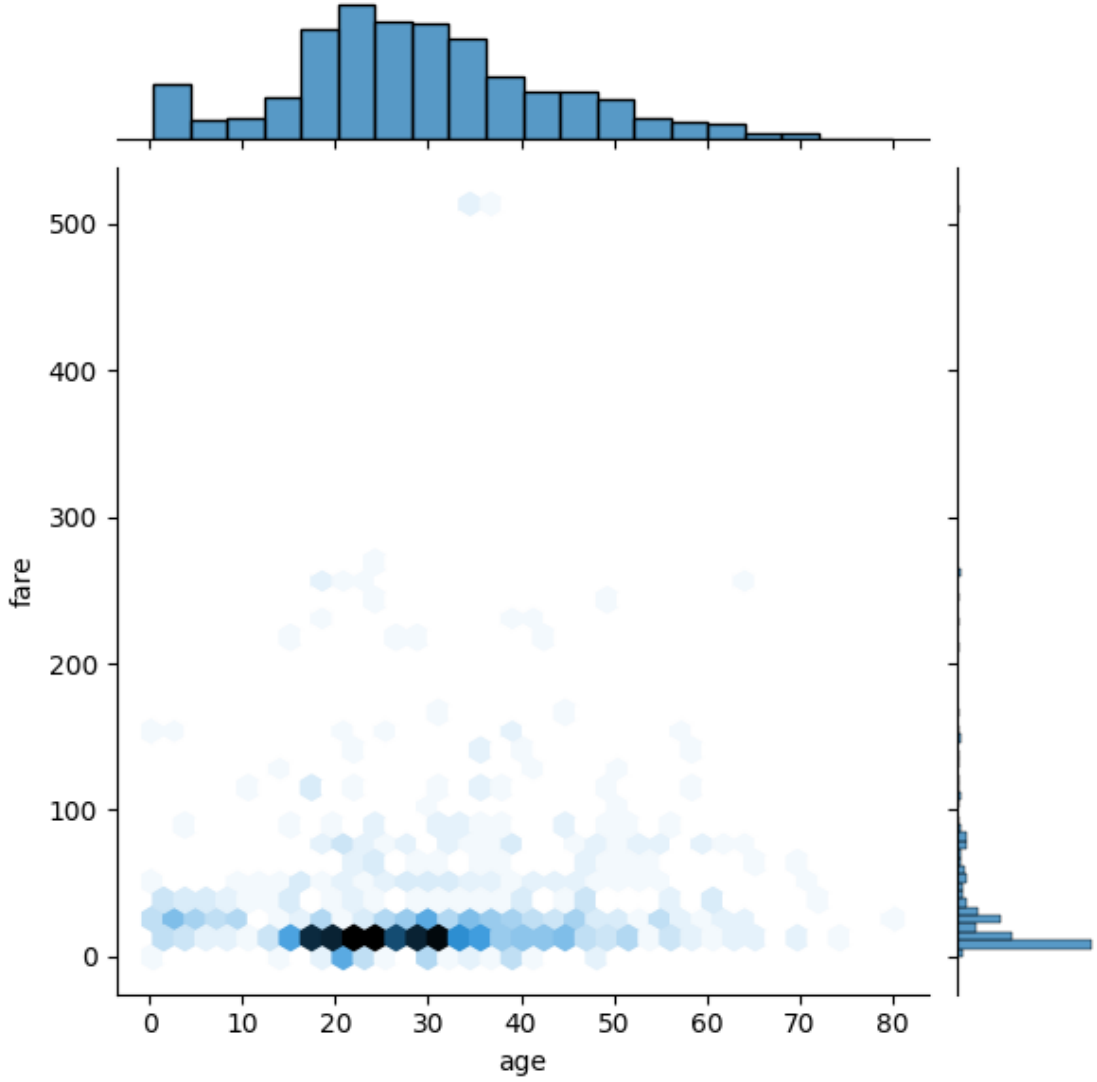
[5]: <Axes: >



[6]:
```
sb.jointplot(x = ds['age'], y = ds['fare'], kind ='scatter')
# For Plot 2
sb.jointplot(x = ds['age'], y = ds['fare'], kind = 'hex')
```

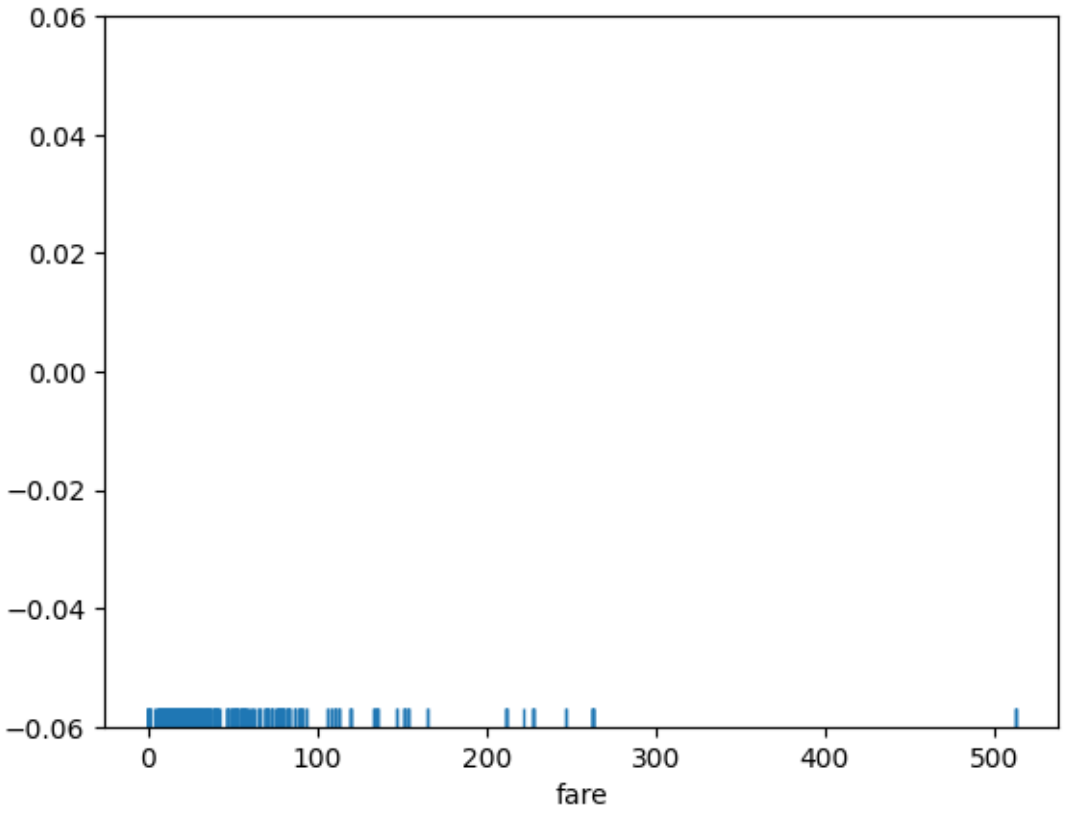
[6]: <seaborn.axisgrid.JointGrid at 0x7fc882ac70a0>


fare
age
0
100
200
300
400
500
0
10
20
30
40
50
60
70
80



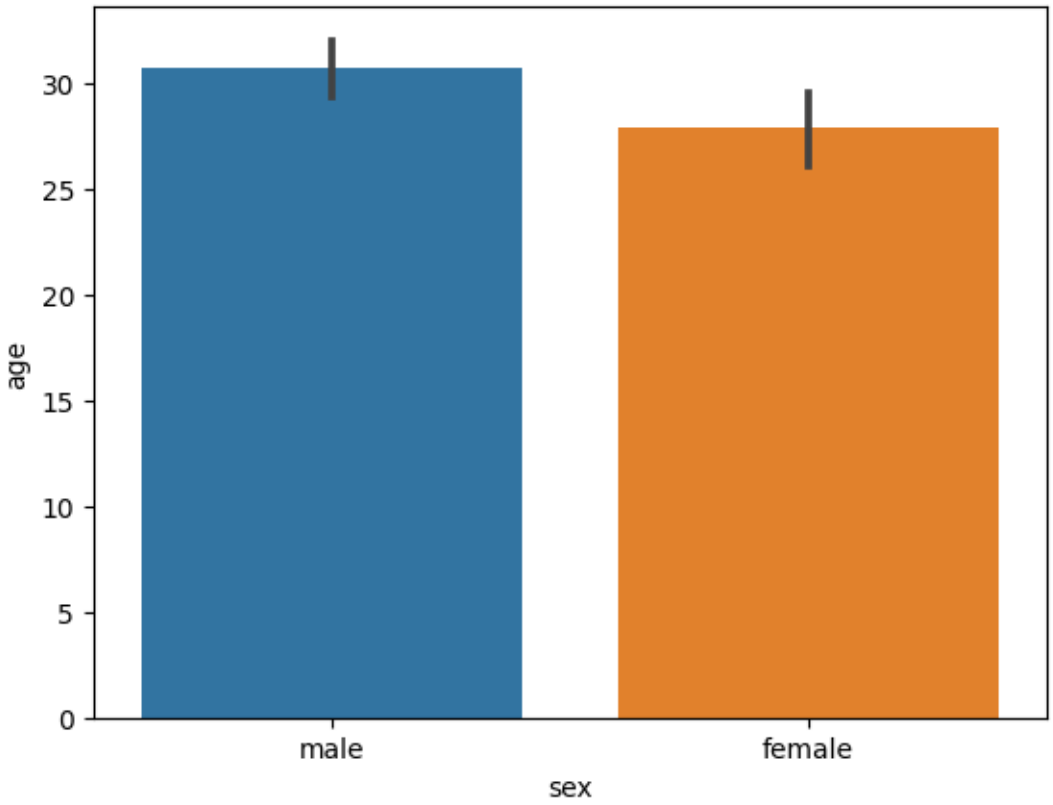
[7]:
```
sb.rugplot(ds['fare'])
```

[7]: <Axes: xlabel='fare'>



[8]:
```
sb.barplot(x='sex', y='age', data=ds)
```

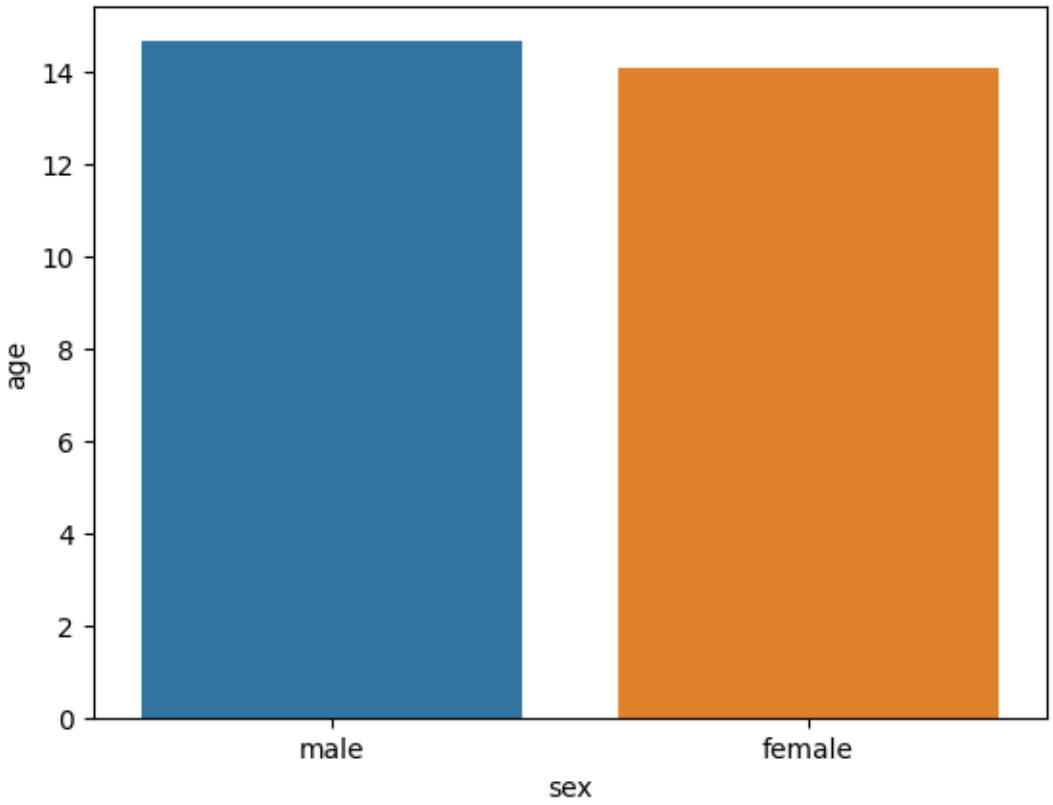
[8]: <Axes: xlabel='sex', ylabel='age'>



[9]:
```
import seaborn as sb
sb.barplot(x='sex', y='age', data=ds, estimator=np.std)
```

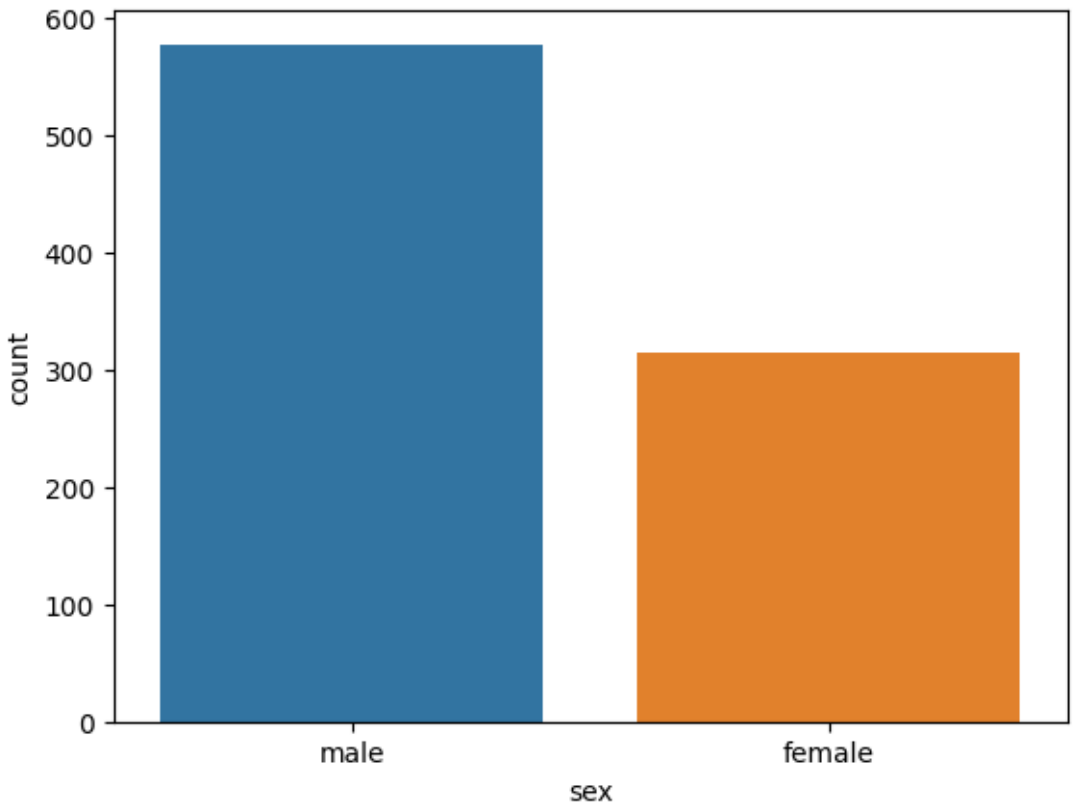
```
/usr/local/lib/python3.10/dist-packages/numpy/lib/nanfunctions.py:1560:
RuntimeWarning: All-NaN slice encountered
  r, k = function_base._ureduce(a,
/usr/local/lib/python3.10/dist-packages/numpy/lib/nanfunctions.py:1560:
RuntimeWarning: All-NaN slice encountered
  r, k = function_base._ureduce(a,
```

[9]:
```
<Axes: xlabel='sex', ylabel='age'>
```



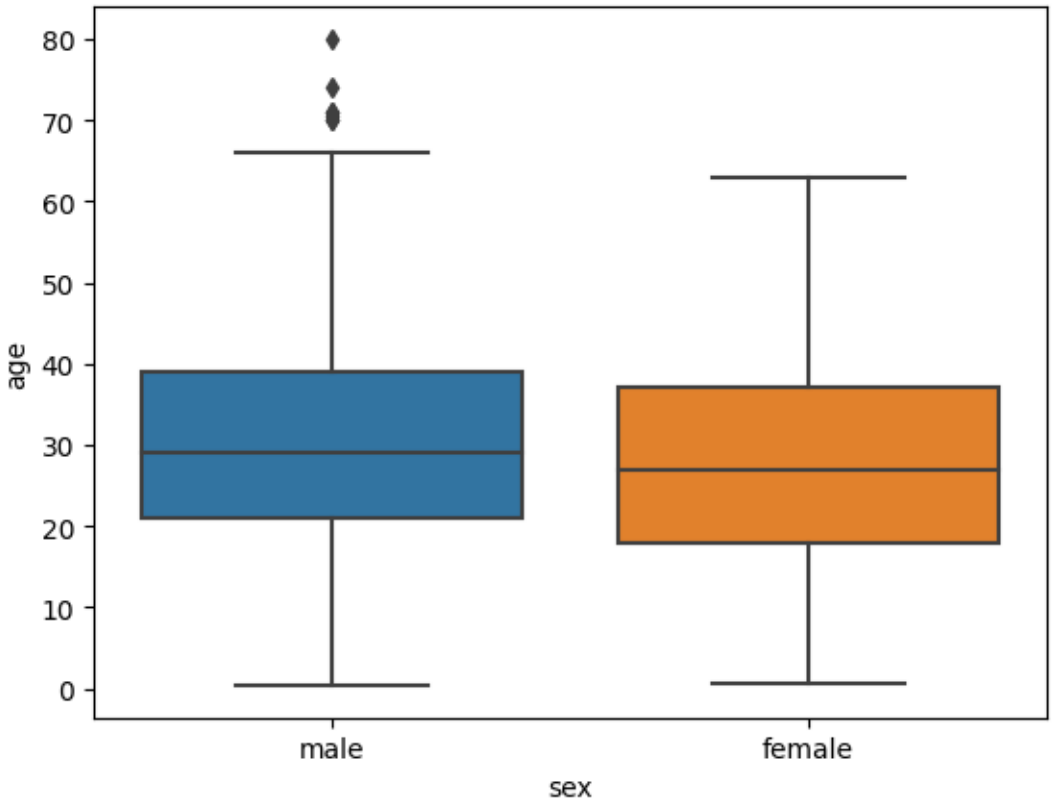
[10]:
```
sb.countplot(x='sex', data=ds)
```

[10]:
```
<Axes: xlabel='sex', ylabel='count'>
```



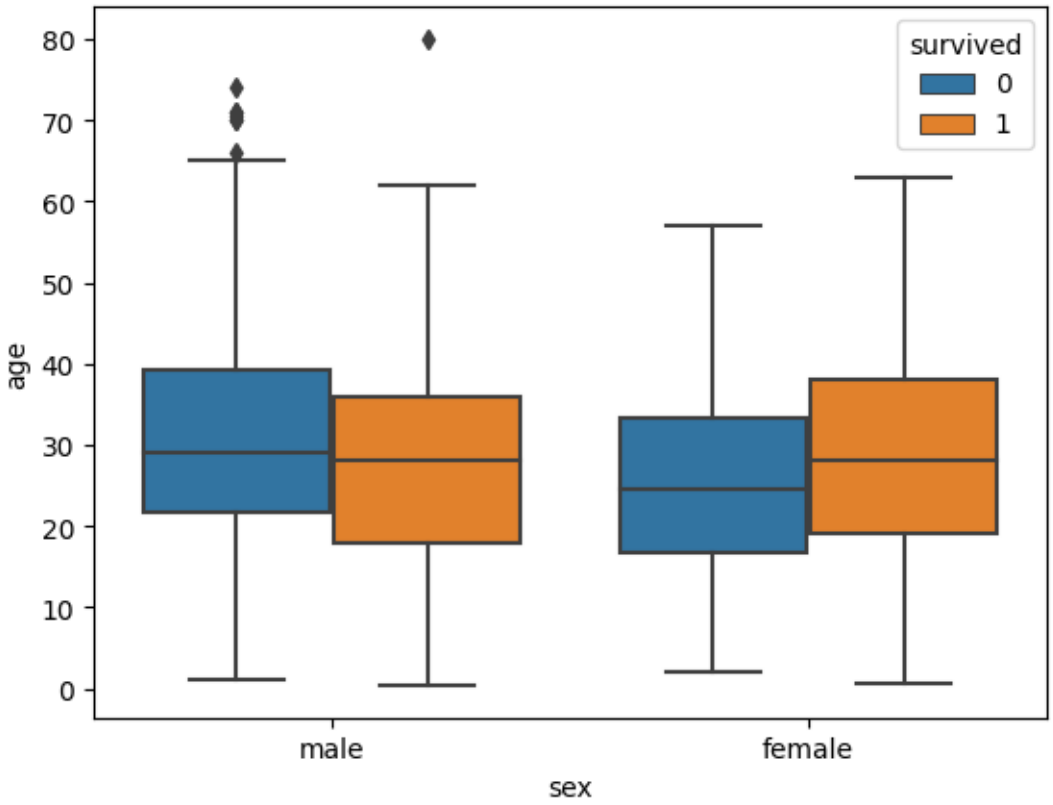
[11]:
```
sb.boxplot(x='sex', y='age', data=ds)
```

[11]:
```
<Axes: xlabel='sex', ylabel='age'>
```



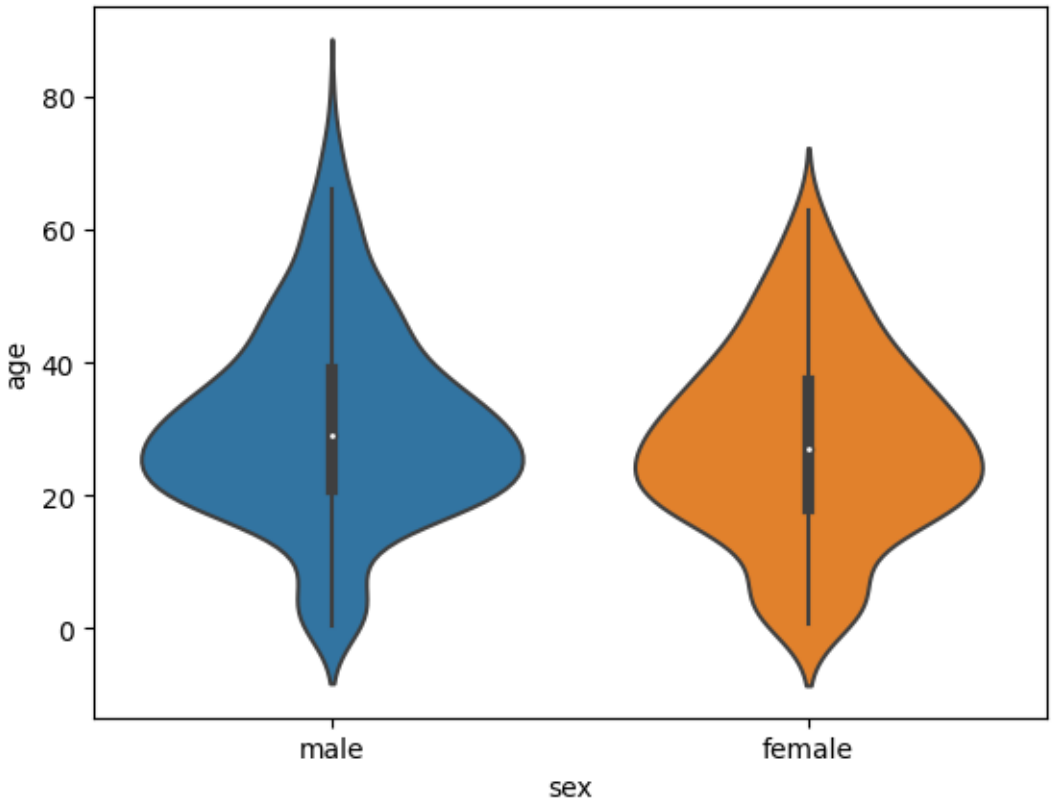
[12]:
```
sb.boxplot(x='sex', y='age', data=ds, hue="survived")
```

[12]:
```
<Axes: xlabel='sex', ylabel='age'>
```



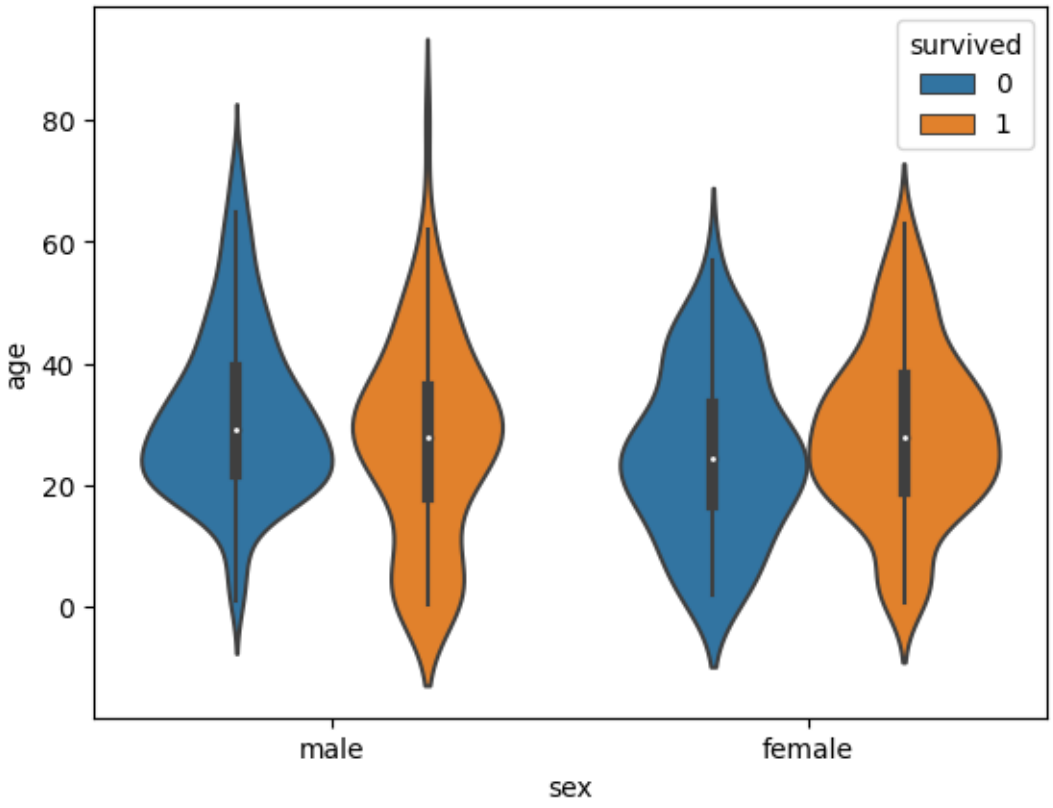
[13]:
```
sb.violinplot(x='sex', y='age', data=ds)
```

[13]:
```
<Axes: xlabel='sex', ylabel='age'>
```



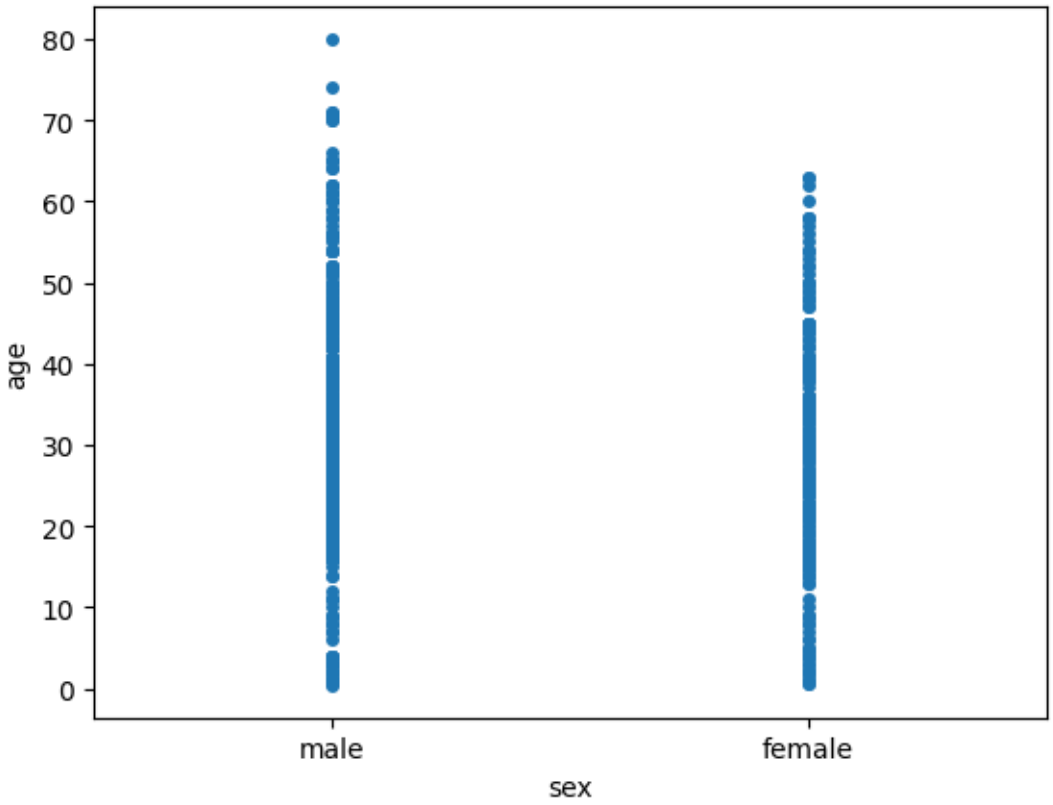
[14]:
```
sb.violinplot(x='sex', y='age', data=ds, hue='survived')
```

[14]: <Axes: xlabel='sex', ylabel='age'>



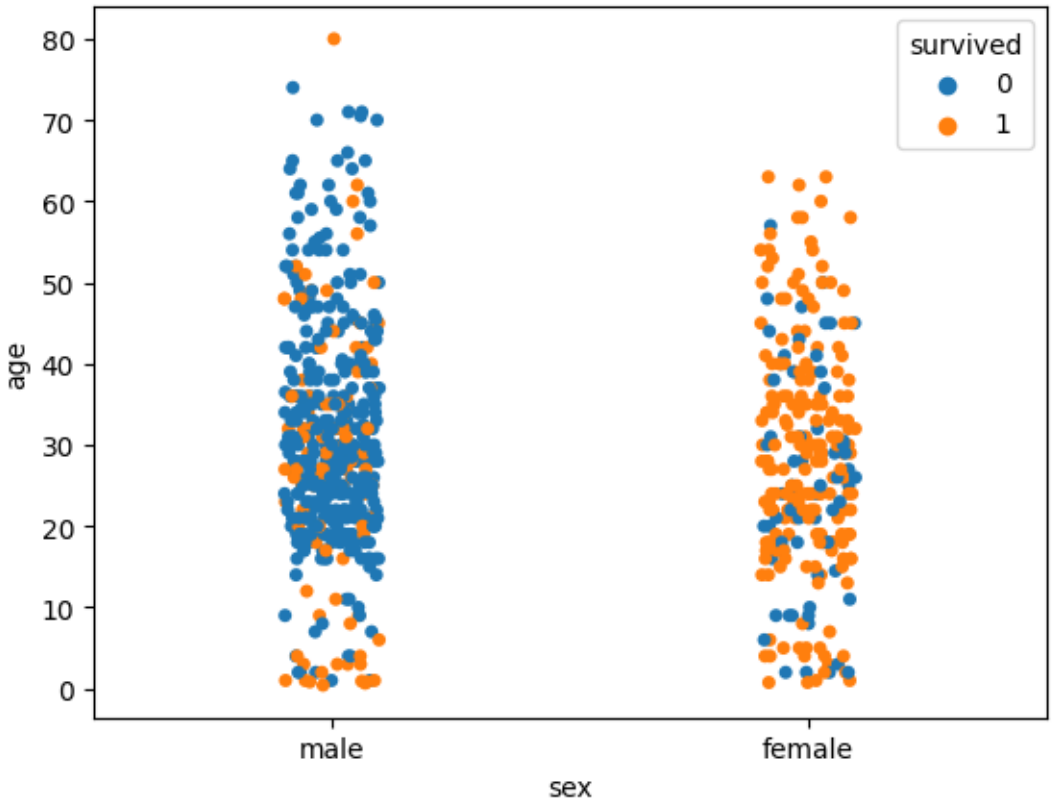
[15]:
```
sb.stripplot(x='sex', y='age', data=ds, jitter=False)
```

[15]:
```
<Axes: xlabel='sex', ylabel='age'>
```



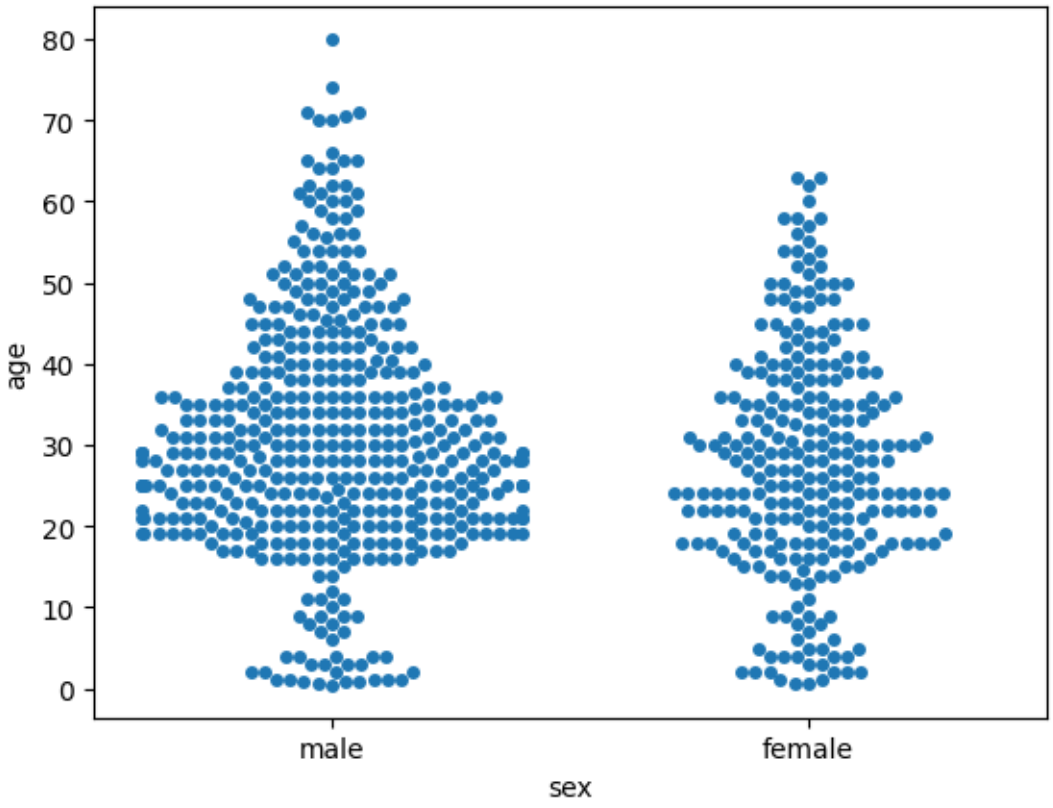
[16]:
```
sb.stripplot(x='sex', y='age', data=ds, jitter=True, hue='survived')
```

[16]:
```
<Axes: xlabel='sex', ylabel='age'>
```



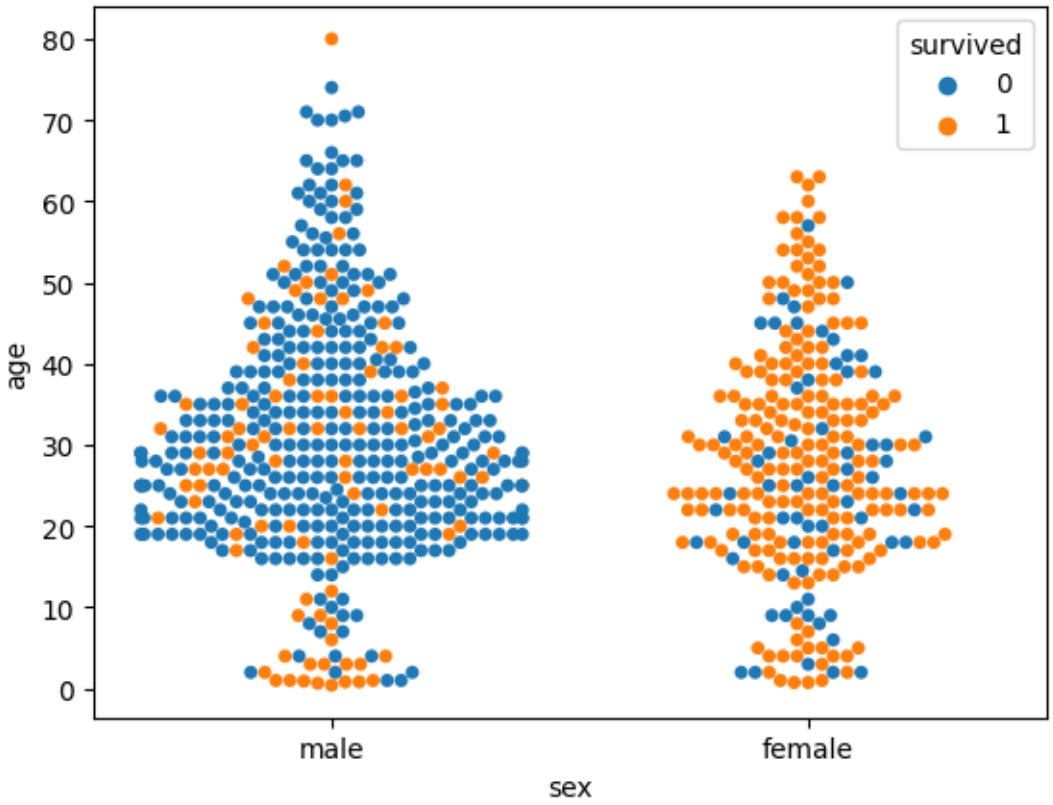
[17]:
```
sb.swarmplot(x='sex', y='age', data=ds)
```

[17]:
```
<Axes: xlabel='sex', ylabel='age'>
```



[18]:
```
sb.swarmplot(x='sex', y='age', data=ds, hue='survived')
```

[18]:
```
<Axes: xlabel='sex', ylabel='age'>
```



[19]:
```
ds.corr()
```

```
<ipython-input-19-31b754434382>:1: 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.
  ds.corr()
```

[19]:

	survived	pclass	age	sibsp	parch	fare
survived	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
pclass	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
age	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067
sibsp	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651
parch	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225
fare	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000
adult_male	-0.557080	0.094035	0.280328	-0.253586	-0.349943	-0.182024
alone	-0.203367	0.135207	0.198270	-0.584471	-0.583398	-0.271832

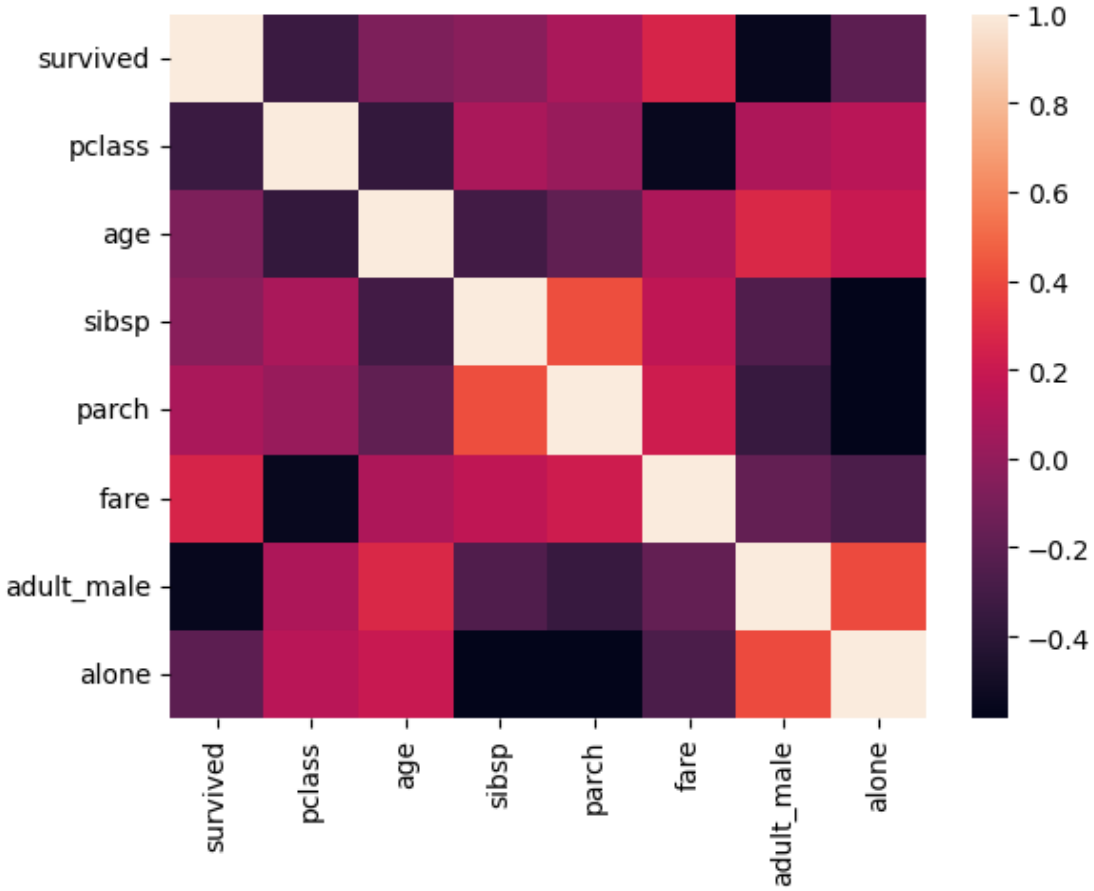
	adult_male	alone
survived	-0.557080	-0.203367
pclass	0.094035	0.135207

```
age           0.280328  0.198270
sibsp        -0.253586 -0.584471
parch        -0.349943 -0.583398
fare         -0.182024 -0.271832
adult_male    1.000000  0.404744
alone         0.404744  1.000000
```

[20]:
```
corr=ds.corr()
sb.heatmap(corr)
```

```
<ipython-input-20-4b64f1a00962>:1: 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.
  corr=ds.corr()
```

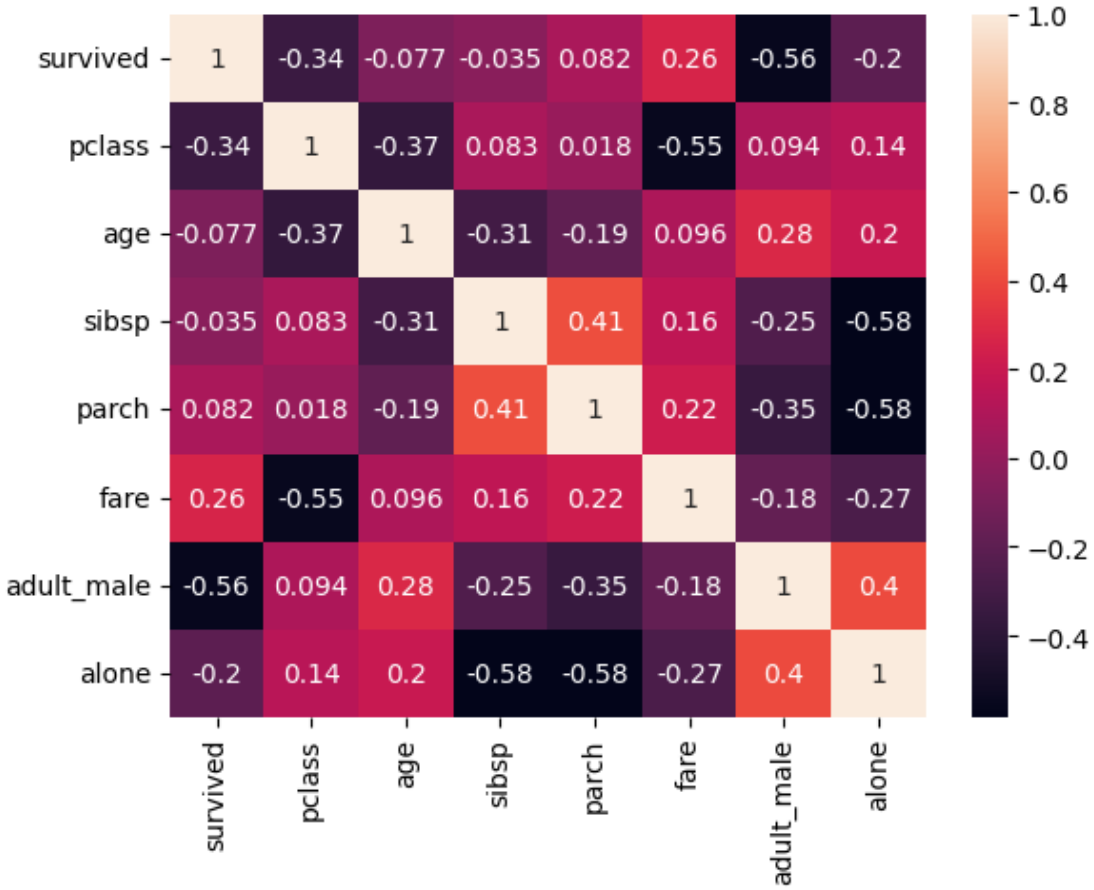
[20]: <Axes: >



[21]:
```
corr=ds.corr()
sb.heatmap(corr,annot=True)
```

```
<ipython-input-21-8fe11ebf8afe>:1: 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.
  corr=ds.corr()
```

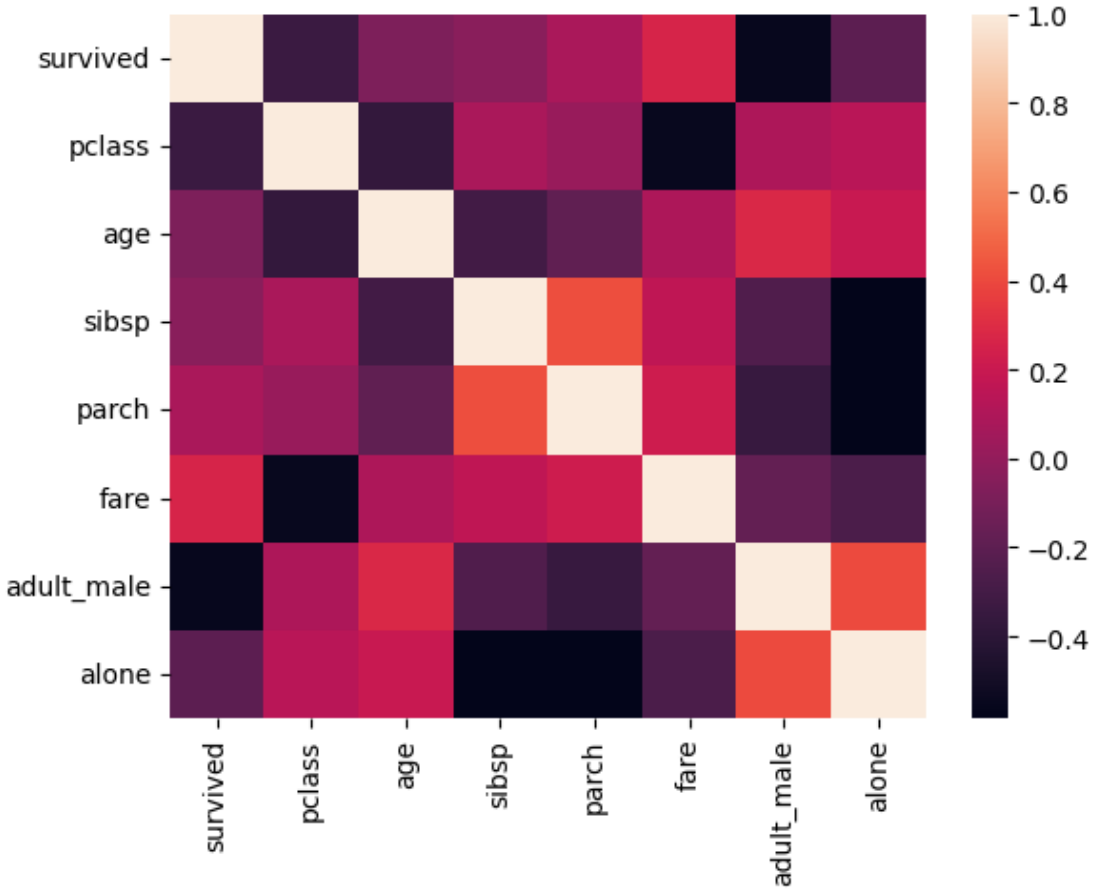
[21]: <Axes: >



[22]:
```
corr=ds.corr()
sb.heatmap(corr)
```

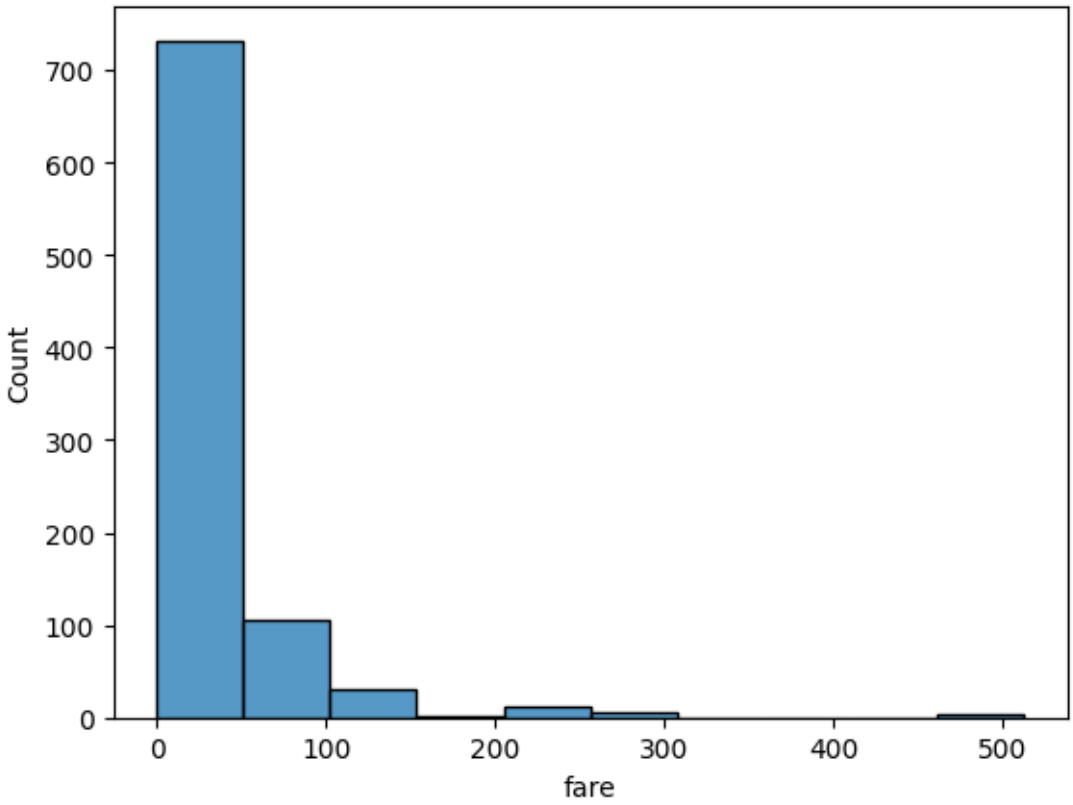
```
<ipython-input-22-4b64f1a00962>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  corr=ds.corr()
```

[22]: <Axes: >



[23]:
```
sb.histplot(ds['fare'], kde=False, bins=10)
```

[23]: <Axes: xlabel='fare', ylabel='Count'>


700
600
500
400
300
200
100
0
Count
0
100
200
300
400
500
fare

assignment9

May 13, 2023

[1]:
```
import seaborn as sb
import matplotlib as plt
db=sb.load_dataset('titanic')
db.head()
```

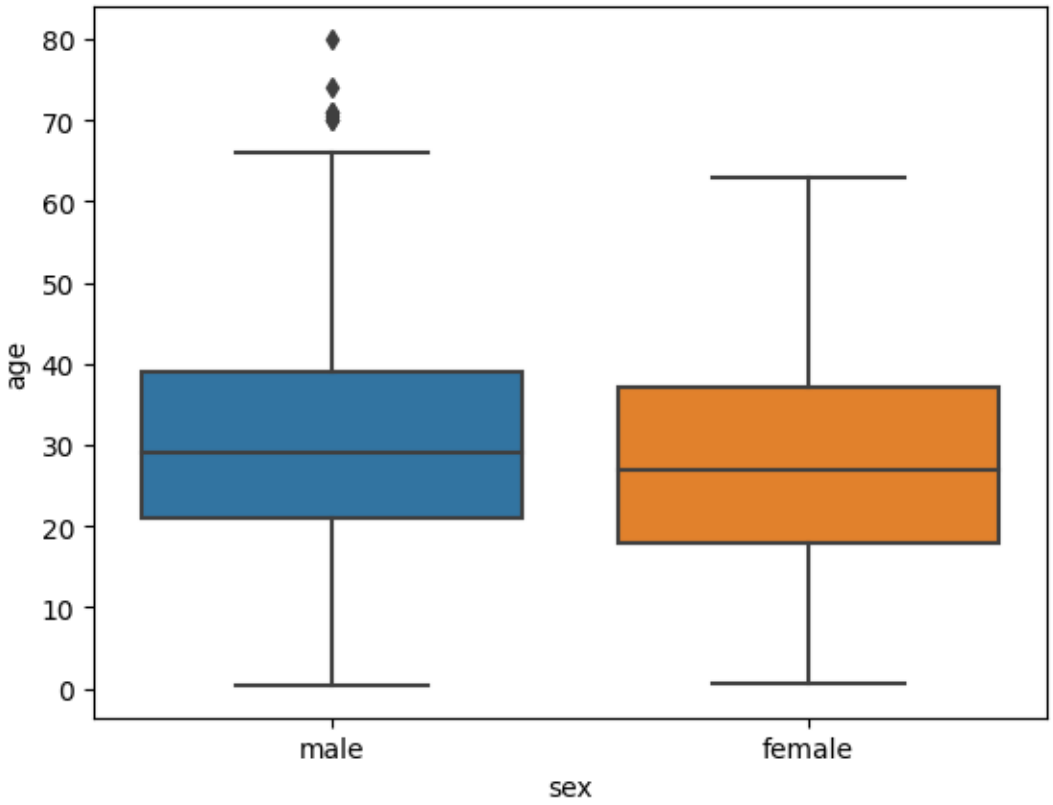
[1]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	\
0	0	3	male	22.0	1	0	7.2500	S	Third	
1	1	1	female	38.0	1	0	71.2833	C	First	
2	1	3	female	26.0	0	0	7.9250	S	Third	
3	1	1	female	35.0	1	0	53.1000	S	First	
4	0	3	male	35.0	0	0	8.0500	S	Third	

	who	adult_male	deck	embark_town	alive	alone
0	man	True	NaN	Southampton	no	False
1	woman	False	C	Cherbourg	yes	False
2	woman	False	NaN	Southampton	yes	True
3	woman	False	C	Southampton	yes	False
4	man	True	NaN	Southampton	no	True

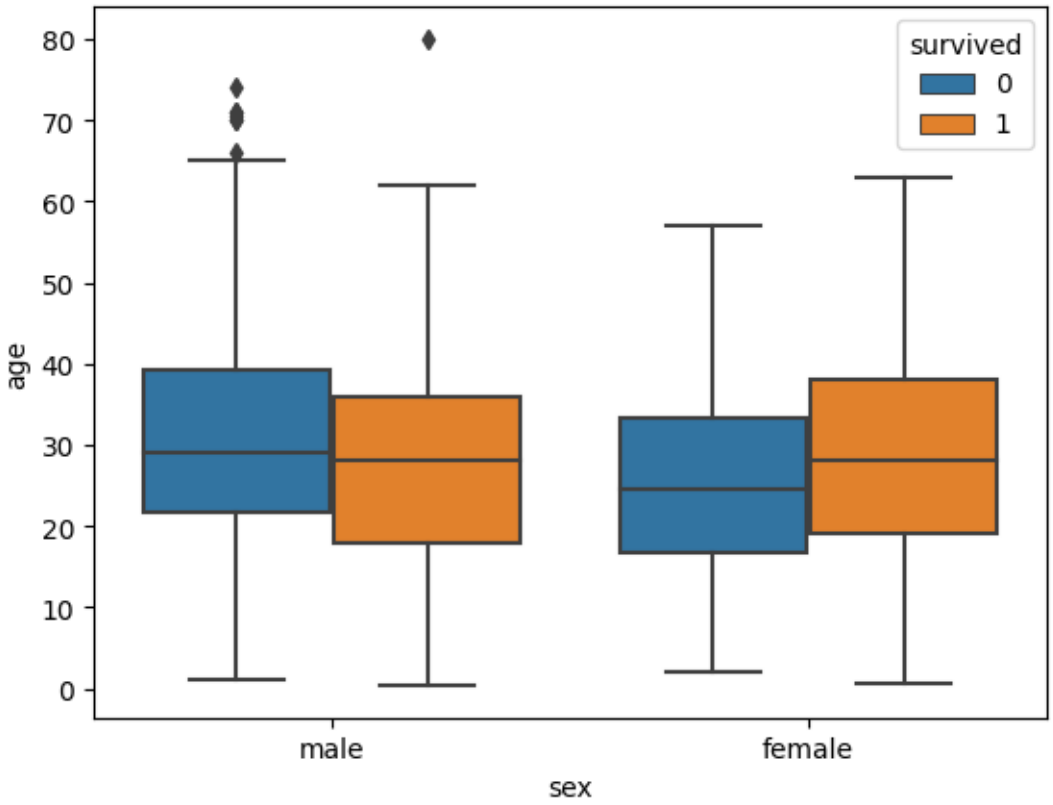
[2]:
```
sb.boxplot(x='sex', y='age', data=db)
```

[2]:
```
<Axes: xlabel='sex', ylabel='age'>
```



[3]:
```
sb.boxplot(x='sex', y='age', data=db, hue='survived')
```

[3]: <Axes: xlabel='sex', ylabel='age'>



[4]:
```
db['survived'].value_counts()
```

[4]:
```
0    549
1    342
Name: survived, dtype: int64
```

[5]:
```
mean=db['age'].mean()
mean
```

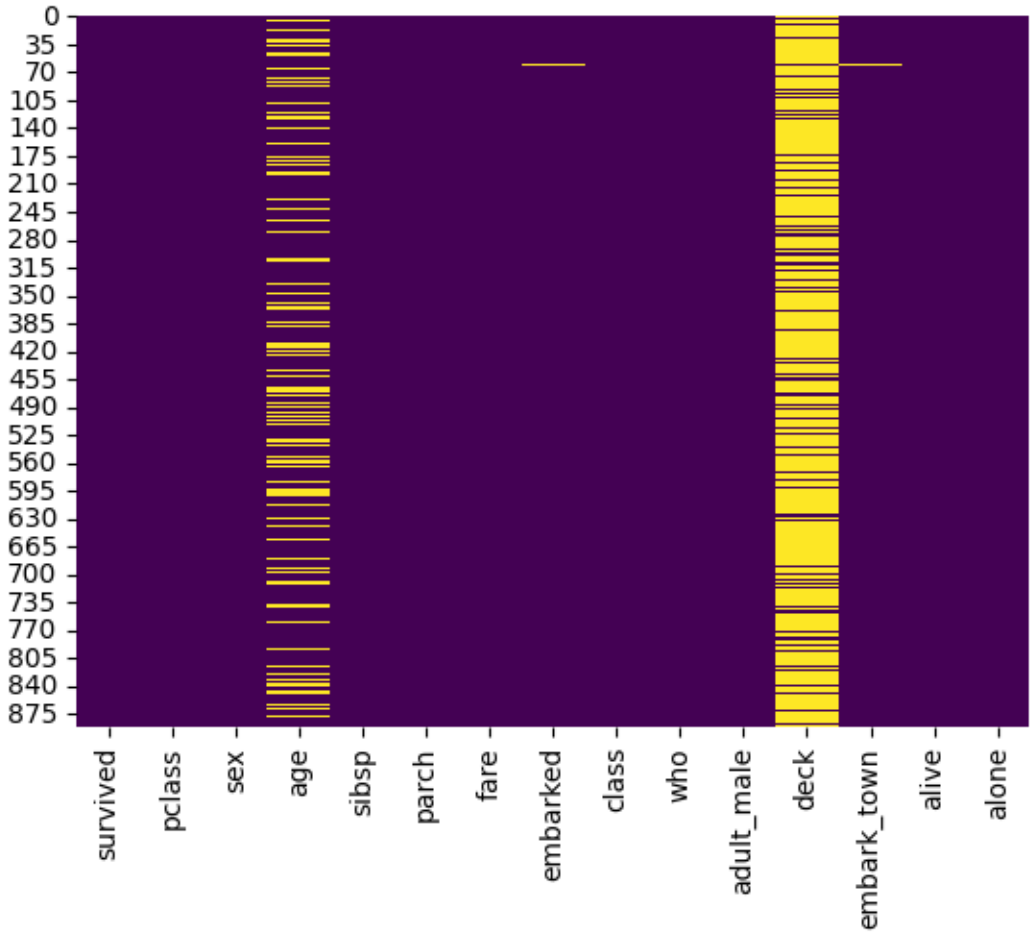
[5]:
```
29.69911764705882
```

[6]:
```
std=db['age'].std()
std
```

[6]:
```
14.526497332334042
```

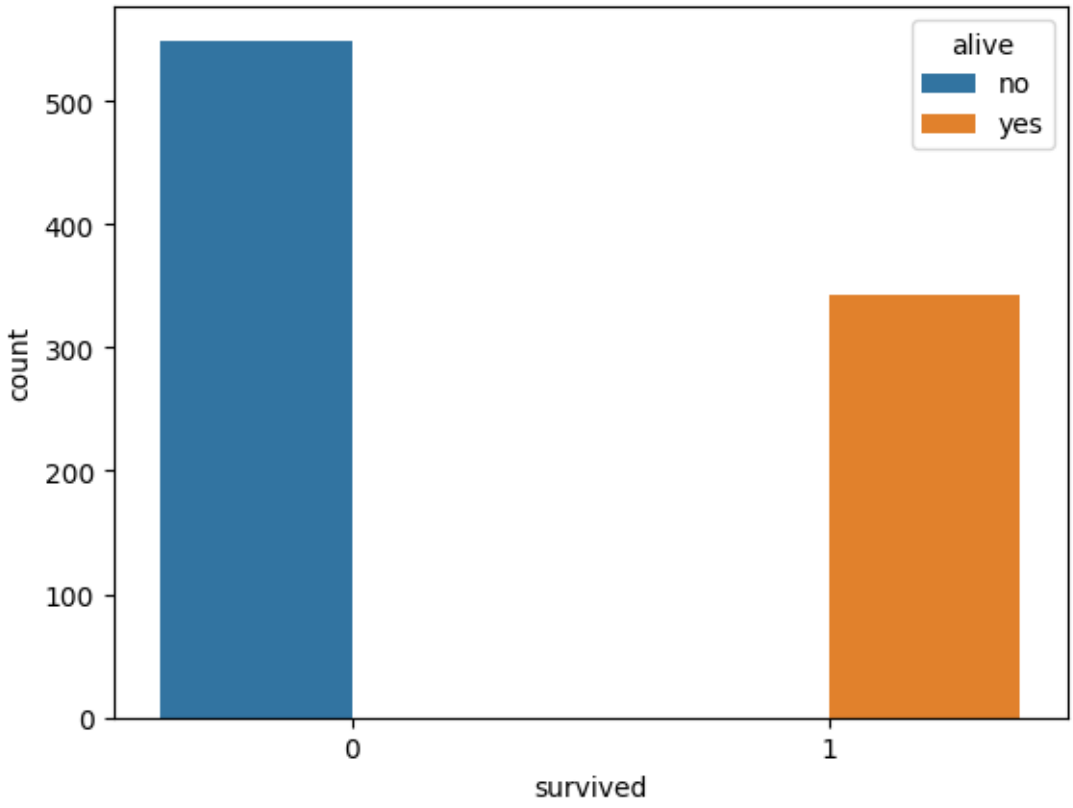
[7]:
```
sb.heatmap(db.isnull(), cmap='viridis', cbar=False)
```

[7]:
```
<Axes: >
```



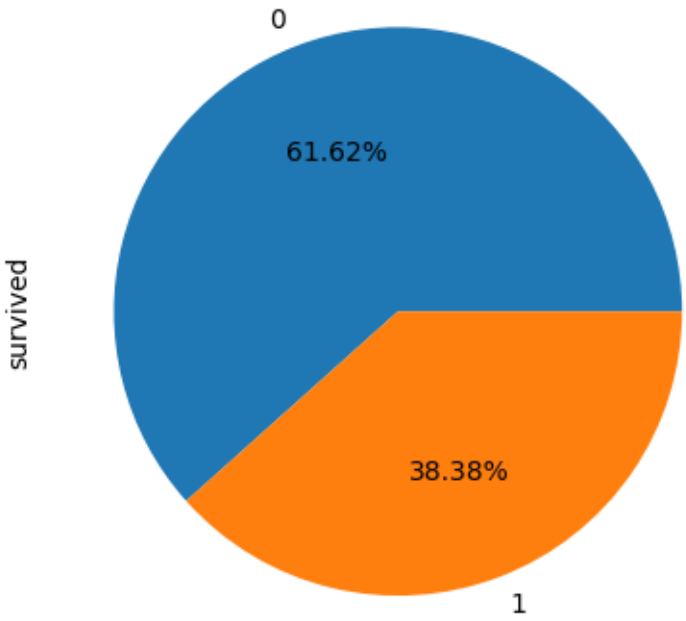
[8]:
```
sb.countplot(data=db, x="survived", hue="alive")
```

[8]:
```
<Axes: xlabel='survived', ylabel='count'>
```



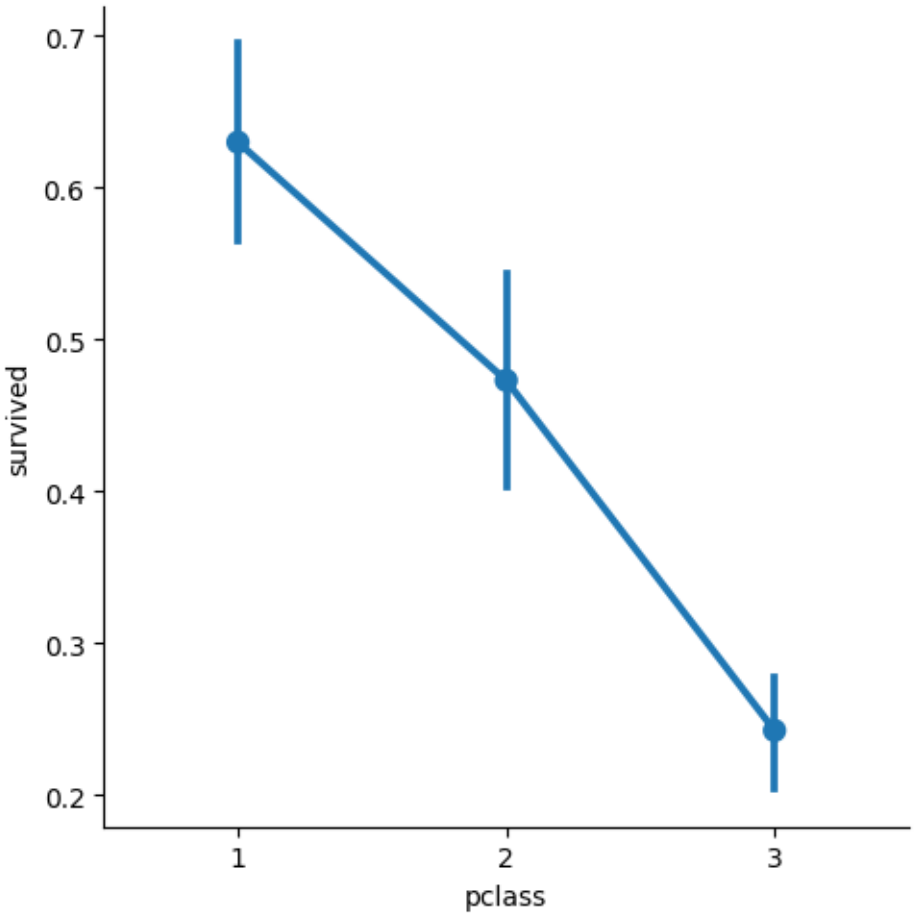
[9]:
```
from enum import auto
explode=[0,0]
db['survived'].value_counts().plot.pie(autopct='%1.2f%%',explode=explode)
```

[9]: <Axes: ylabel='survived'>



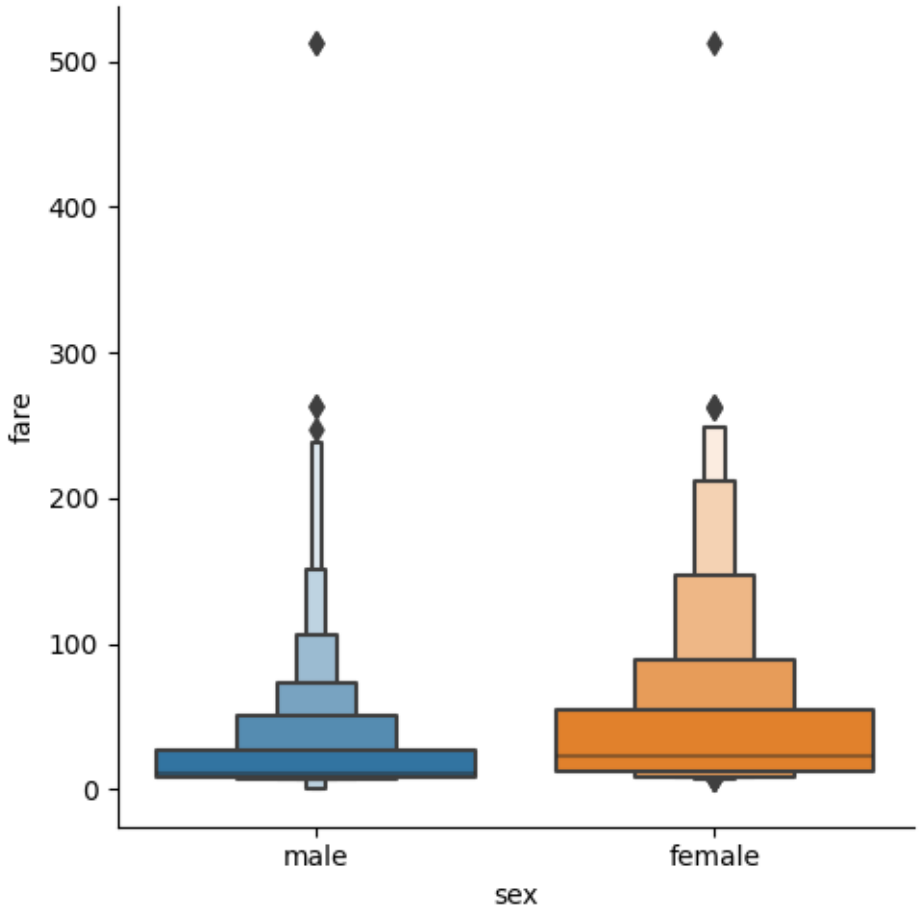
[10]:
```
from ctypes import pointer
sb.catplot(x='pclass',y='survived', data=db, kind='point')
```

[10]: <seaborn.axisgrid.FacetGrid at 0x7f35f0dabd30>



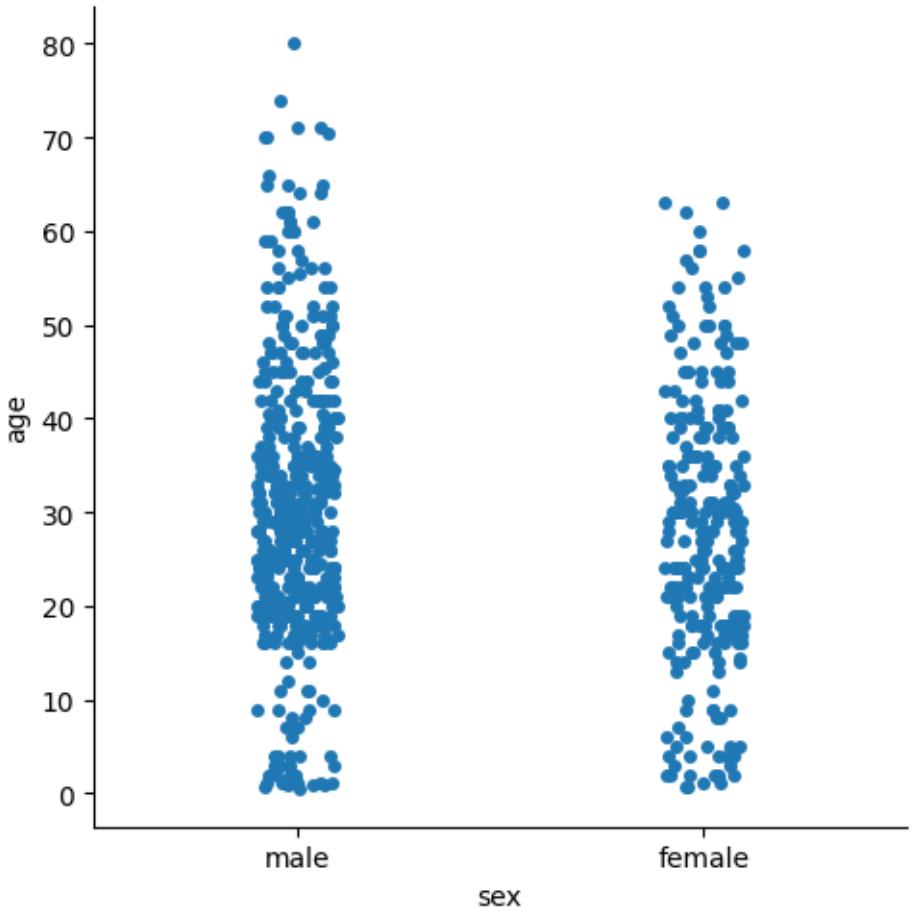
[11]:
```
sb.catplot(x='sex',y='fare', data=db, kind='boxen')
```

[11]: <seaborn.axisgrid.FacetGrid at 0x7f35f0dabb50>



[12]:
```
sb.catplot(x='sex',y='age', data=db)
```

[12]: <seaborn.axisgrid.FacetGrid at 0x7f35f0c86aa0>


80
70
60
50
40
30
20
10
0
age
male
female
sex

assignment10

May 13, 2023

[1]:
```
import seaborn as sb
import matplotlib.pyplot as plt
import pandas as pd
df=pd.read_csv('/content/drive/MyDrive/Dataset/Iris.csv')
df
```

[1]:
```
       Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
0       1            5.1           3.5            1.4           0.2
1       2            4.9           3.0            1.4           0.2
2       3            4.7           3.2            1.3           0.2
3       4            4.6           3.1            1.5           0.2
4       5            5.0           3.6            1.4           0.2
..    ...            ...           ...            ...           ...
145   146            6.7           3.0            5.2           2.3
146   147            6.3           2.5            5.0           1.9
147   148            6.5           3.0            5.2           2.0
148   149            6.2           3.4            5.4           2.3
149   150            5.9           3.0            5.1           1.8

            Species
0       Iris-setosa
1       Iris-setosa
2       Iris-setosa
3       Iris-setosa
4       Iris-setosa
..              ...
145  Iris-virginica
146  Iris-virginica
147  Iris-virginica
148  Iris-virginica
149  Iris-virginica

[150 rows x 6 columns]
```

[2]:
```
print("Printing Head")
df.head()
```

```
Printing Head
```

[2]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

[3]:
```
print("Printing Tail")
df.tail()
```

```
Printing Tail
```

[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	\
145	146	6.7	3.0	5.2	2.3	
146	147	6.3	2.5	5.0	1.9	
147	148	6.5	3.0	5.2	2.0	
148	149	6.2	3.4	5.4	2.3	
149	150	5.9	3.0	5.1	1.8	

	Species
145	Iris-virginica
146	Iris-virginica
147	Iris-virginica
148	Iris-virginica
149	Iris-virginica

[4]:
```
print("Printing Information")
df.info()
```

```
Printing Information
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

[5]:
```
print("Printing Shape")
print(df.shape)
```

```
Printing Shape
```

```
(150, 6)
```

[6]:
```
print("Printing Datatypes")
df.dtypes
```

```
Printing Datatypes
```

[6]:
```
Id                int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

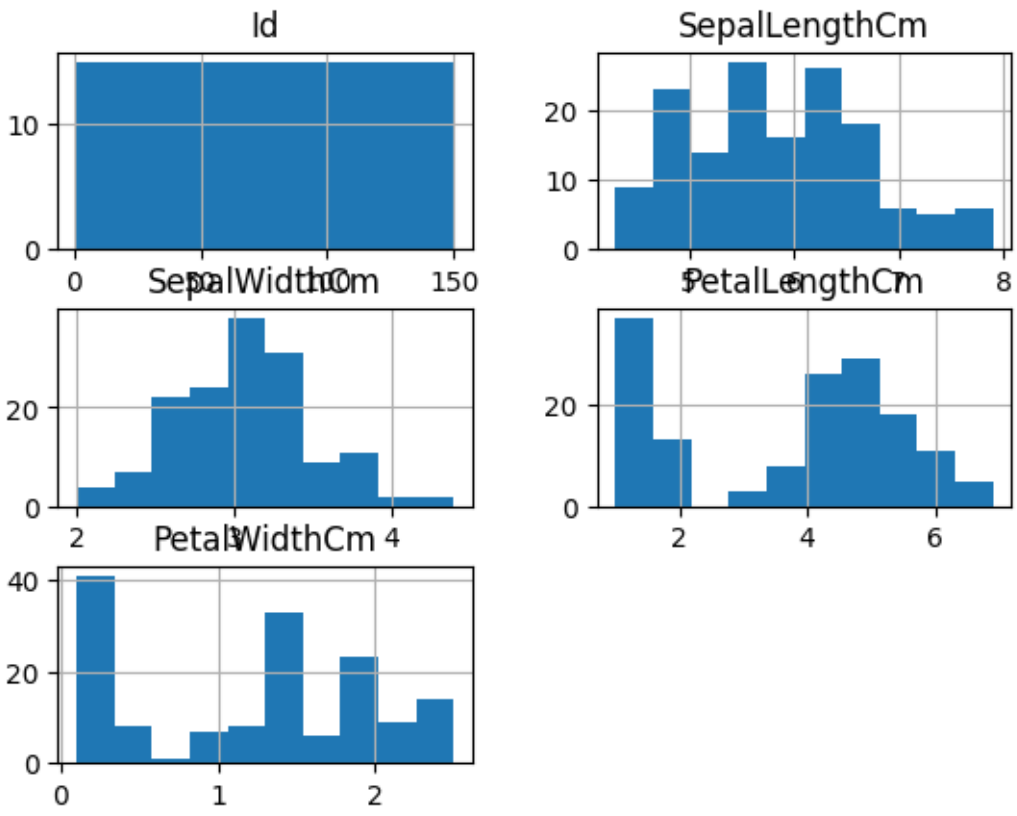
[7]:
```
print("Printing Description")
df.describe()
```

```
Printing Description
```

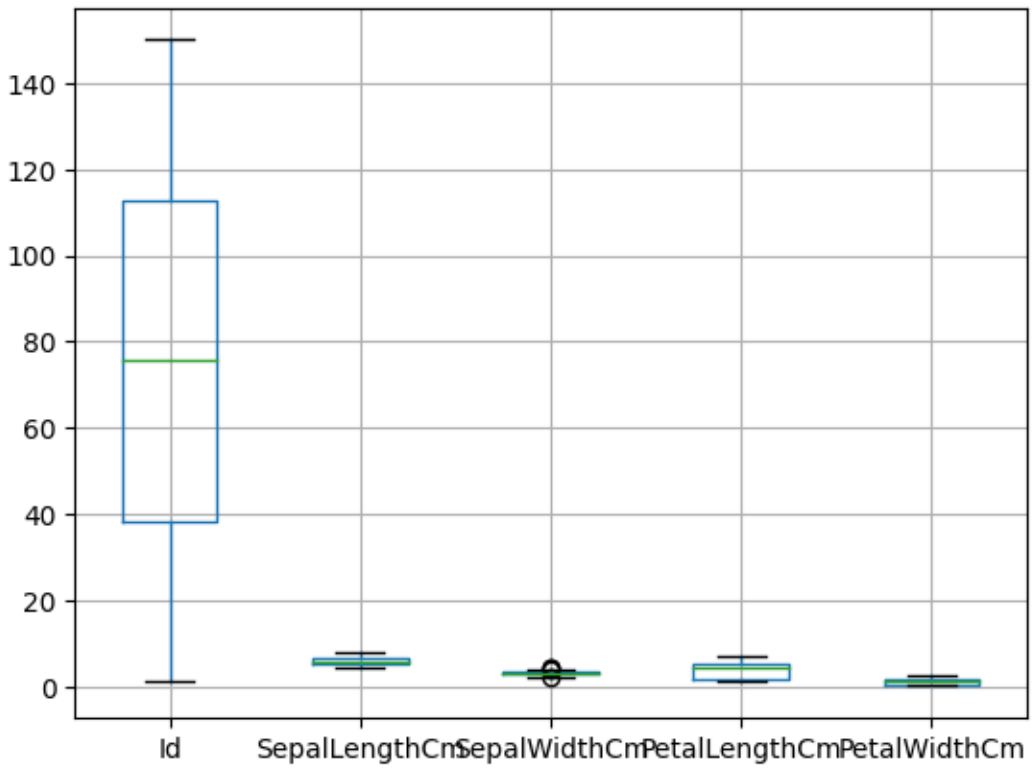
[7]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

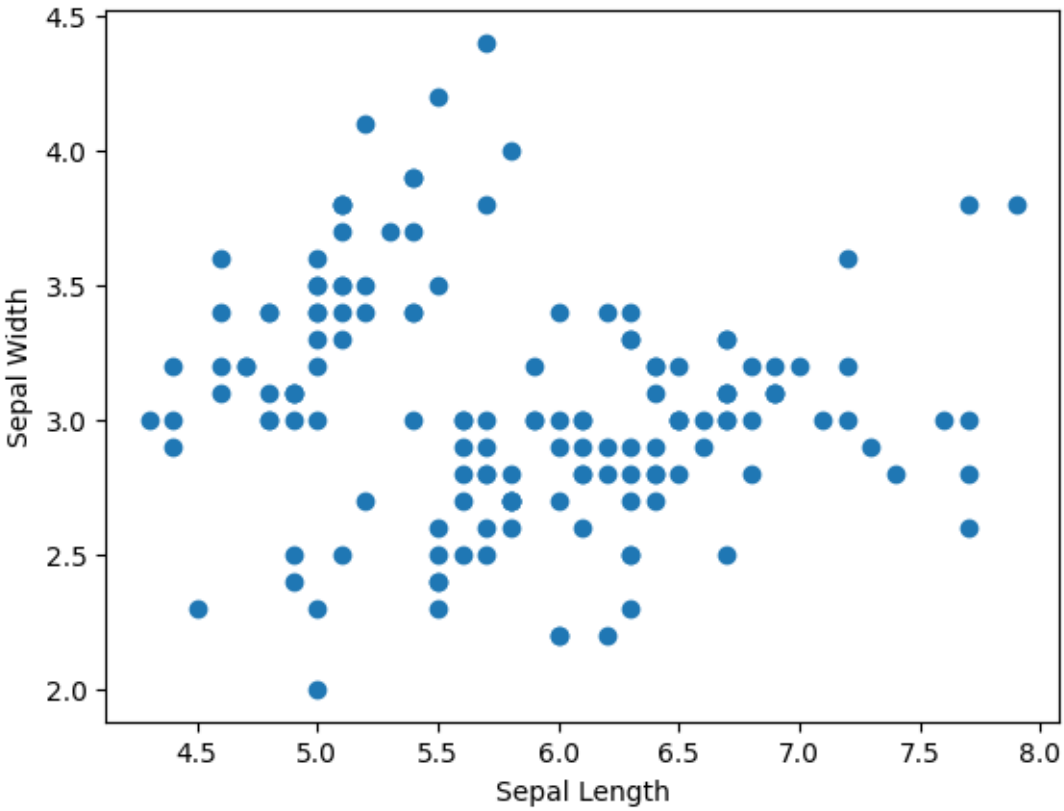
[8]:
```
df.hist()
plt.show()
```



[9]:
```
df.boxplot()
plt.show()
```



[10]:
```
plt.scatter(df["SepalLengthCm"],df["SepalWidthCm"])
plt.xlabel('Sepal Length')
plt.ylabel('Sepal Width')
plt.show()
```


4.5
4.0
3.5
3.0
2.5
2.0
Sepal Width
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Sepal Length

Q. Write a code in JAVA for a simple WordCount application that counts the number of

occurrences of each word in a given input set using the Hadoop MapReduce framework on

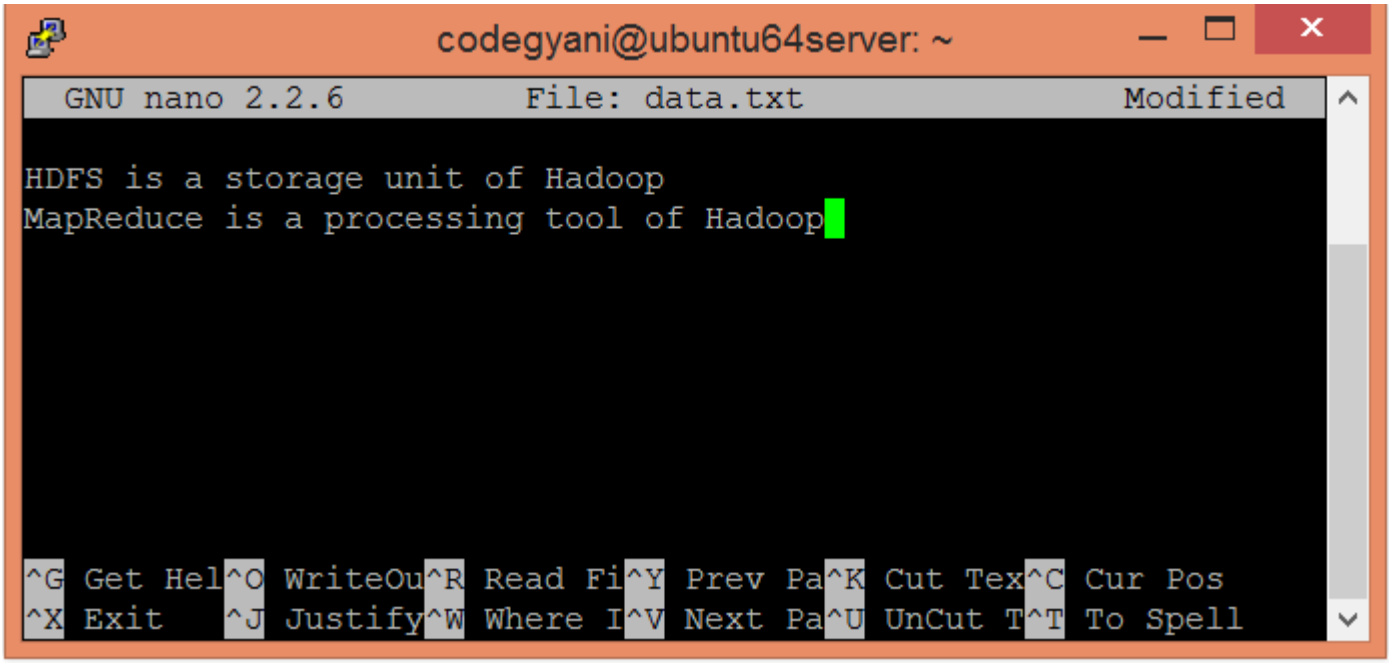
local-standalone set-up.

Input:

Steps to execute MapReduce word count example

- Create a text file in your local machine and write some text into it.

$ nano data.txt



- Check the text written in the data.txt file.
  $ cat data.txt

1. public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
2. private final static IntWritable one = new IntWritable(1);
3. private Text word = new Text();
4. public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
5. Reporter reporter) throws IOException{

```
6.          String line = value.toString();
7.          StringTokenizer  tokenizer = new StringTokenizer(line);
8.          while (tokenizer.hasMoreTokens()){
9.              word.set(tokenizer.nextToken());
10.             output.collect(word, one);
11.         }
12.     }
13.
14. }
```

File: WC_Reducer.java

```
1.  package com.javatpoint;
2.      import java.io.IOException;
3.      import java.util.Iterator;
4.      import org.apache.hadoop.io.IntWritable;
5.      import org.apache.hadoop.io.Text;
6.      import org.apache.hadoop.mapred.MapReduceBase;
7.      import org.apache.hadoop.mapred.OutputCollector;
8.      import org.apache.hadoop.mapred.Reducer;
9.      import org.apache.hadoop.mapred.Reporter;
10.
11.     public class WC_Reducer  extends MapReduceBase implements Reducer<Text,Int
    Writable,Text,IntWritable> {
12.     public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,I
    ntWritable> output,
13.      Reporter reporter) throws IOException {
14.     int sum=0;
15.     while (values.hasNext()) {
16.     sum+=values.next().get();
17.     }
18.     output.collect(key,new IntWritable(sum));
19.     }
20.     }
```

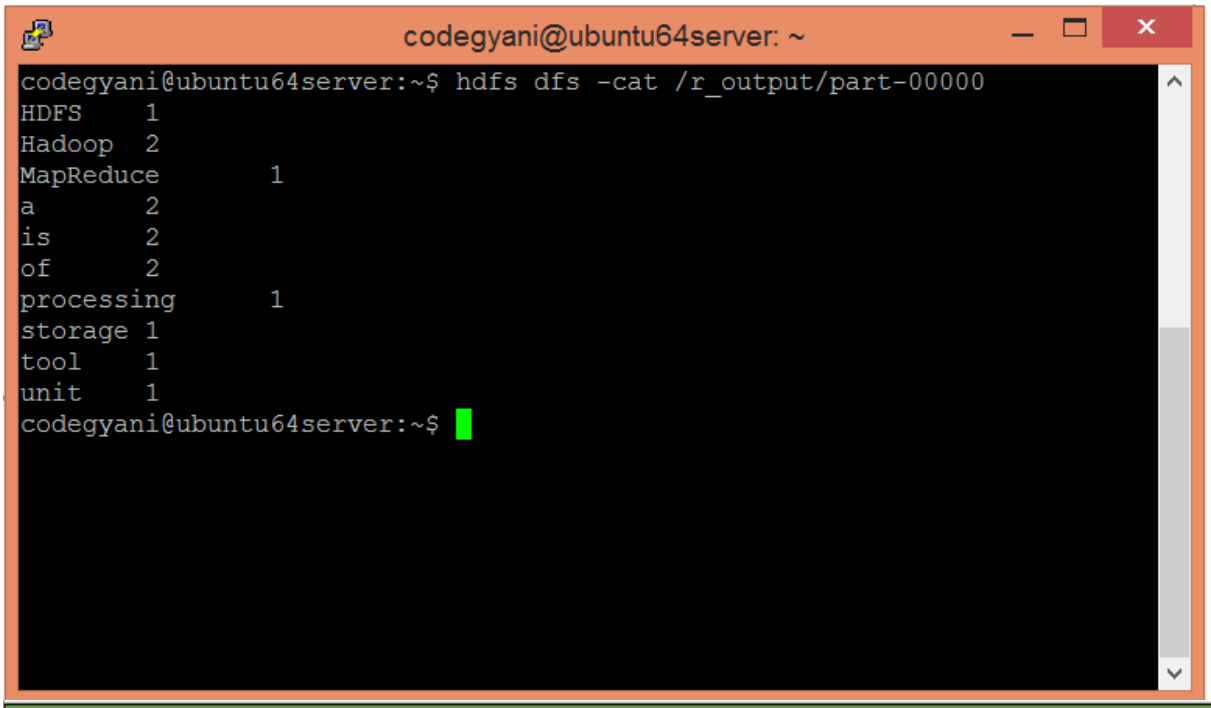
File: WC_Runner.java

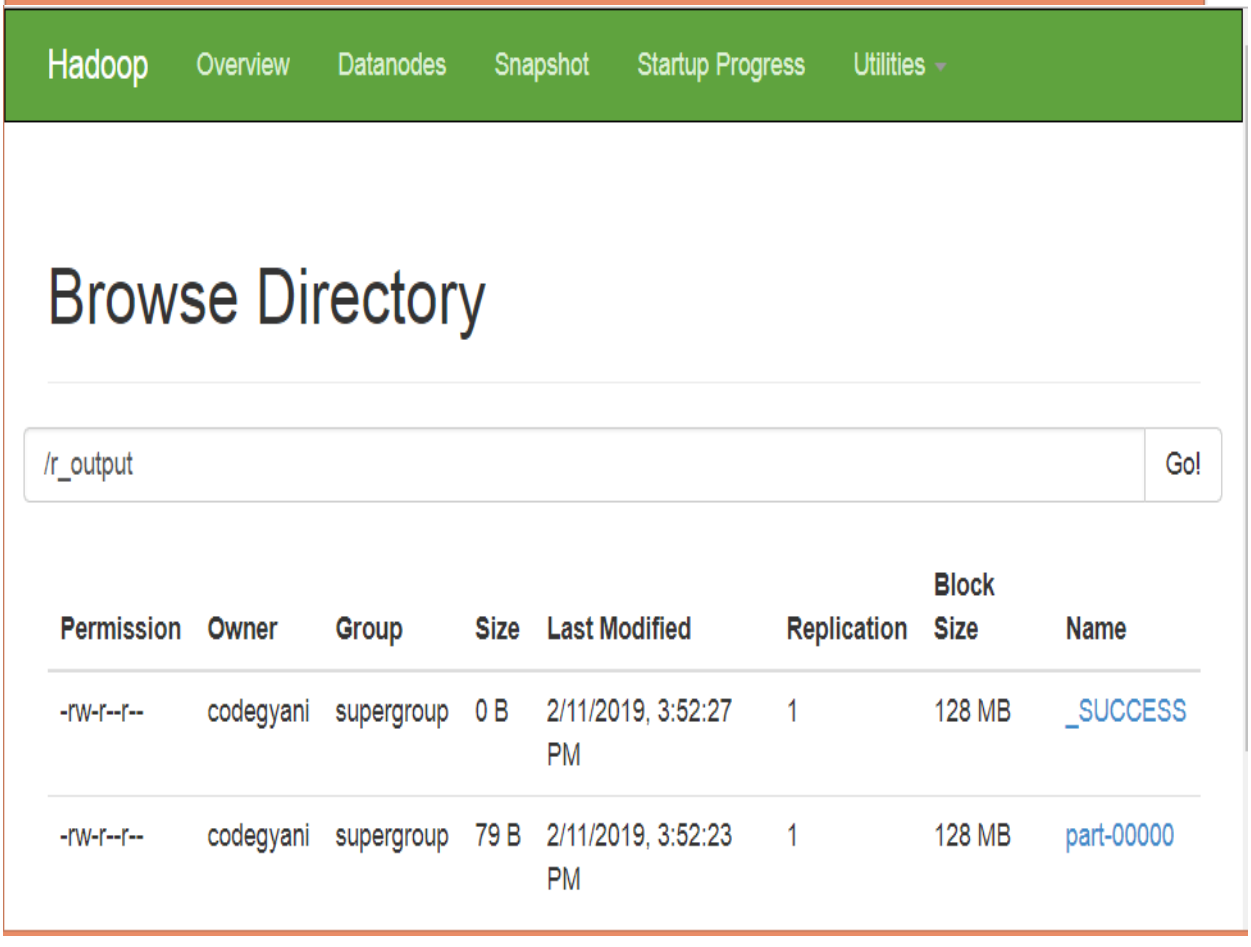
```
1.  package com.javatpoint;
```

```
2.
3.      import java.io.IOException;
4.      import org.apache.hadoop.fs.Path;
5.      import org.apache.hadoop.io.IntWritable;
6.      import org.apache.hadoop.io.Text;
7.      import org.apache.hadoop.mapred.FileInputFormat;
8.      import org.apache.hadoop.mapred.FileOutputFormat;
9.      import org.apache.hadoop.mapred.JobClient;
10.     import org.apache.hadoop.mapred.JobConf;
11.     import org.apache.hadoop.mapred.TextInputFormat;
12.     import org.apache.hadoop.mapred.TextOutputFormat;
13.     public class WC_Runner {
14.        public static void main(String[] args) throws IOException{
15.           JobConf conf = new JobConf(WC_Runner.class);
16.           conf.setJobName("WordCount");
17.           conf.setOutputKeyClass(Text.class);
18.           conf.setOutputValueClass(IntWritable.class);
19.           conf.setMapperClass(WC_Mapper.class);
20.           conf.setCombinerClass(WC_Reducer.class);
21.           conf.setReducerClass(WC_Reducer.class);
22.           conf.setInputFormat(TextInputFormat.class);
23.           conf.setOutputFormat(TextOutputFormat.class);
24.           FileInputFormat.setInputPaths(conf,new Path(args[0]));
25.           FileOutputFormat.setOutputPath(conf,new Path(args[1]));
26.           JobClient.runJob(conf);
27.        }
28.     }
```

Download the source code.

- Create the jar file of this program and name it **countworddemo.jar.**
- Run the jar file hadoop jar /home/codegyani/wordcountdemo.jar com.javatpoint.WC_Runner /test/data.txt /r_output
- The output is stored in /r_output/part-00000


codegyani@ubuntu64server: ~
codegyani@ubuntu64server:~$ hdfs dfs -cat /r_output/part-00000
HDFS 1
Hadoop 2
MapReduce 1
a 2
is 2
of 2
processing 1
storage 1
tool 1
unit 1
codegyani@ubuntu64server:~$


Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Browse Directory
/r_output
Go!
Permission
Owner
Group
Size
Last Modified
Replication
Block Size
Name
-rw-r--r--
codegyani
supergroup
0 B
2/11/2019, 3:52:27 PM
1
128 MB
_SUCCESS
-rw-r--r--
codegyani
supergroup
79 B
2/11/2019, 3:52:23 PM
1
128 MB
part-00000

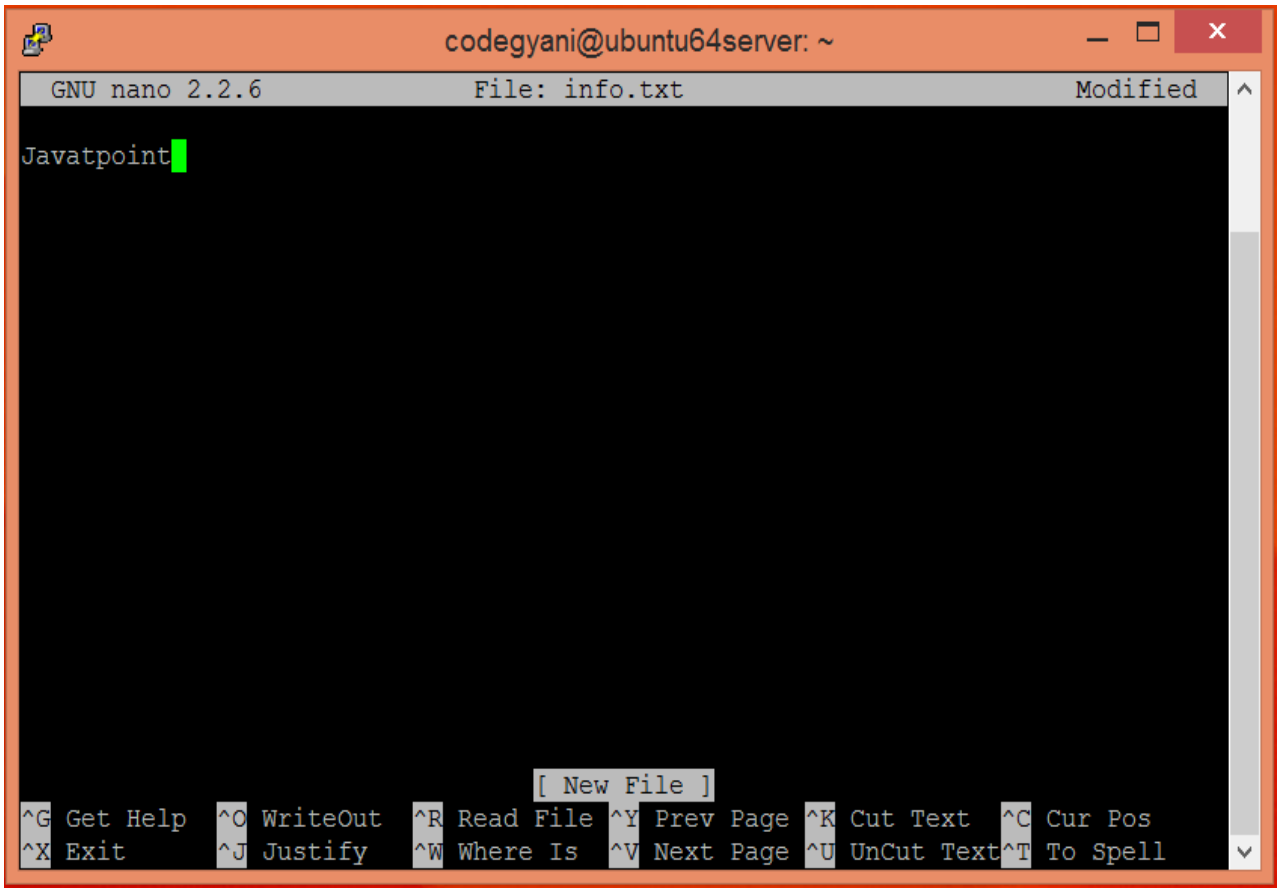
- Now execute the command to see the output. hdfs dfs -cat /r_output/part-0000

Steps to execute MapReduce char count example

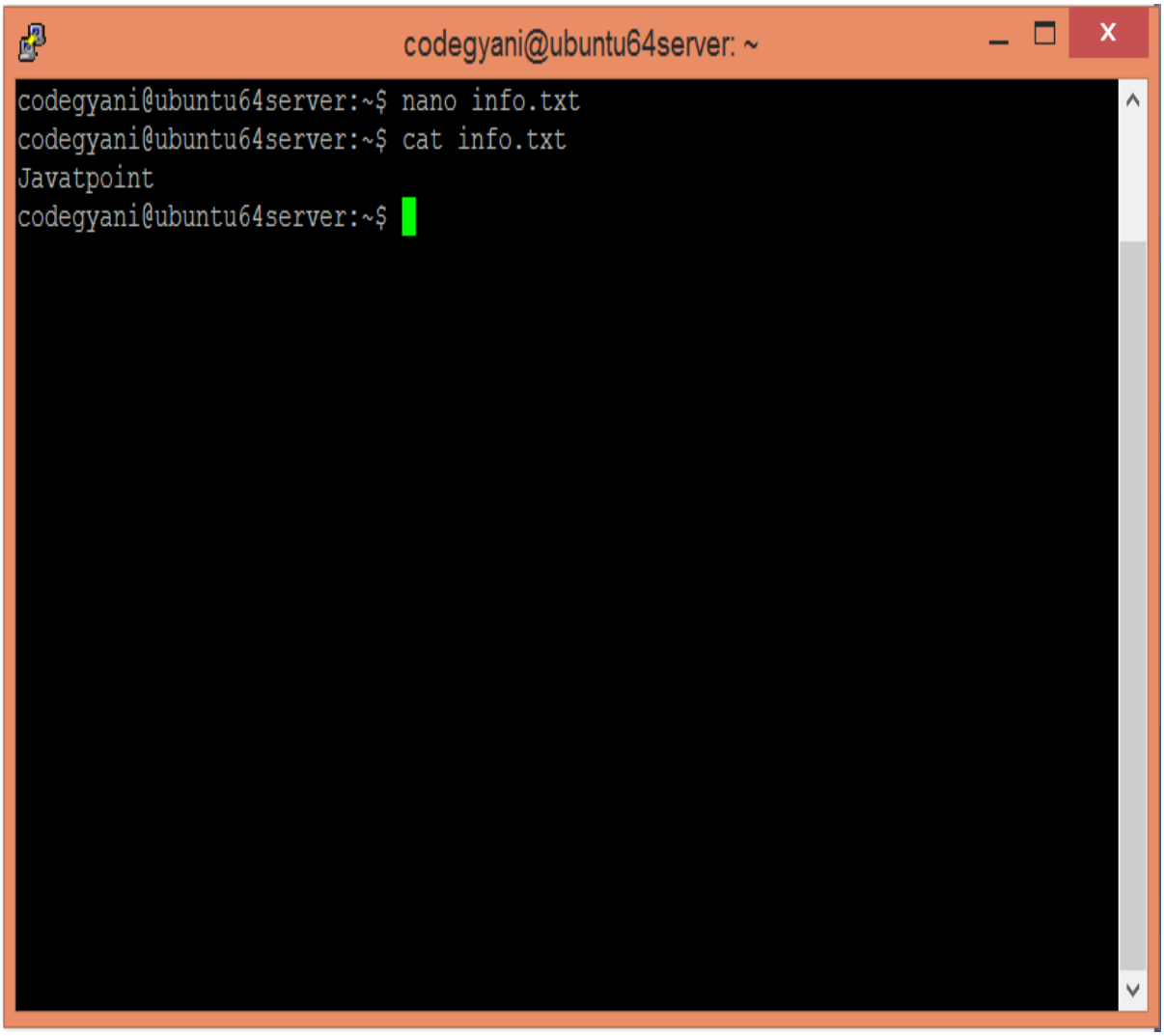
- Create a text file in your local machine and write some text into it.
  $ nano info.txt

Steps to execute MapReduce char count example

- Create a text file in your local machine and write some text into it.
  $ nano info.txt

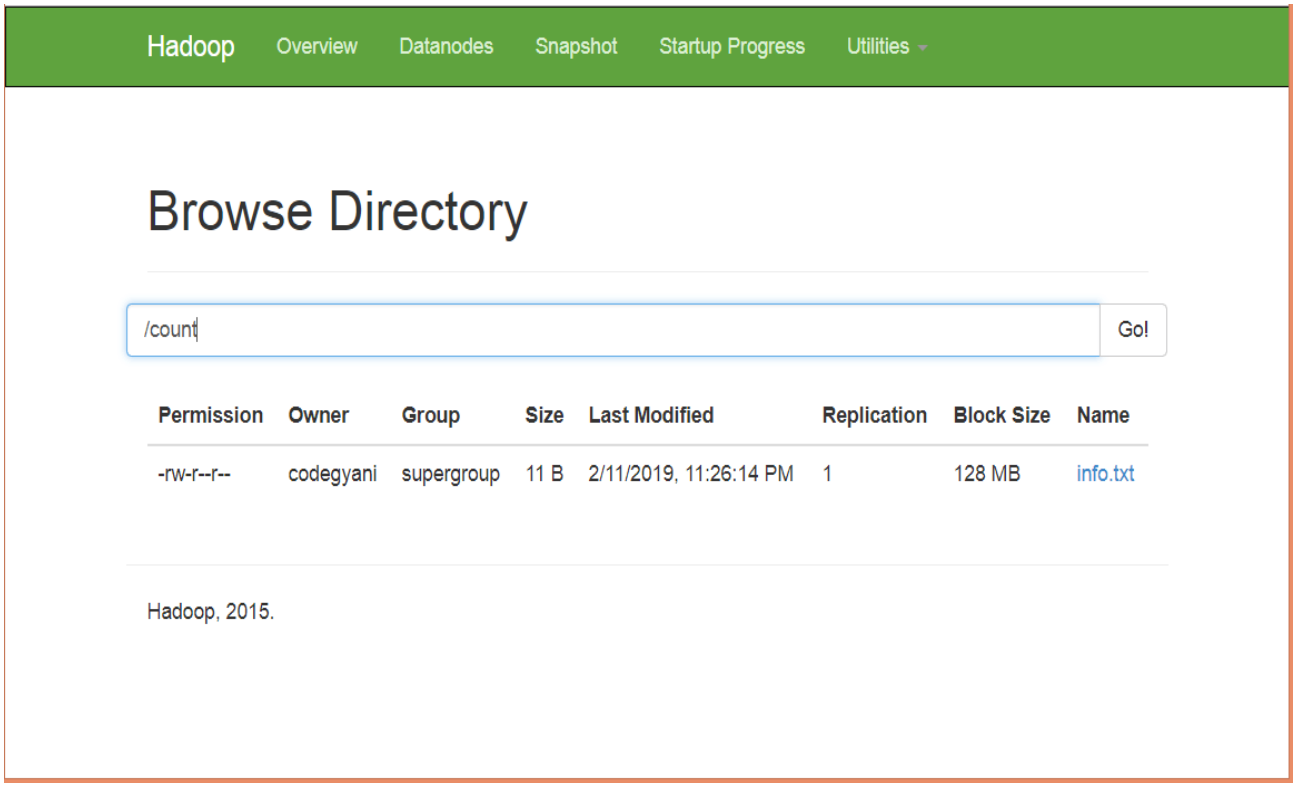


- Check the text written in the info.txt file.
  $ cat info.txt



In this example, we find out the frequency of each char value exists in this text file.

- Create a directory in HDFS, where to kept text file.
  $ hdfs dfs -mkdir /count
- Upload the info.txt file on HDFS in the specific directory.
  $ hdfs dfs -put /home/codegyani/info.txt /count



- Write the MapReduce program using eclipse.

File: WC_Mapper.java

```
1. package com.javatpoint;
2.
3. import java.io.IOException;
4. import org.apache.hadoop.io.IntWritable;
5. import org.apache.hadoop.io.LongWritable;
6. import org.apache.hadoop.io.Text;
7. import org.apache.hadoop.mapred.MapReduceBase;
8. import org.apache.hadoop.mapred.Mapper;
9. import org.apache.hadoop.mapred.OutputCollector;
10. import org.apache.hadoop.mapred.Reporter;
11. public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
12.     public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
13.             Reporter reporter) throws IOException{
14.         String line = value.toString();
15.         String  tokenizer[] = line.split("");
16.         for(String SingleChar : tokenizer)
```

```
17.         {
18.             Text charKey = new Text(SingleChar);
19.             IntWritable One = new IntWritable(1);
20.             output.collect(charKey, One);
21.         }
22.     }
23.
24. }
```

File: WC_Reducer.java

```
1.  package com.javatpoint;
2.      import java.io.IOException;
3.      import java.util.Iterator;
4.      import org.apache.hadoop.io.IntWritable;
5.      import org.apache.hadoop.io.Text;
6.      import org.apache.hadoop.mapred.MapReduceBase;
7.      import org.apache.hadoop.mapred.OutputCollector;
8.      import org.apache.hadoop.mapred.Reducer;
9.      import org.apache.hadoop.mapred.Reporter;
10.
11.     public class WC_Reducer  extends MapReduceBase implements Reducer<Text,Int
    Writable,Text,IntWritable> {
12.     public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,I
    ntWritable> output,
13.      Reporter reporter) throws IOException {
14.     int sum=0;
15.     while (values.hasNext()) {
16.     sum+=values.next().get();
17.     }
18.     output.collect(key,new IntWritable(sum));
19.     }
20.     }
```

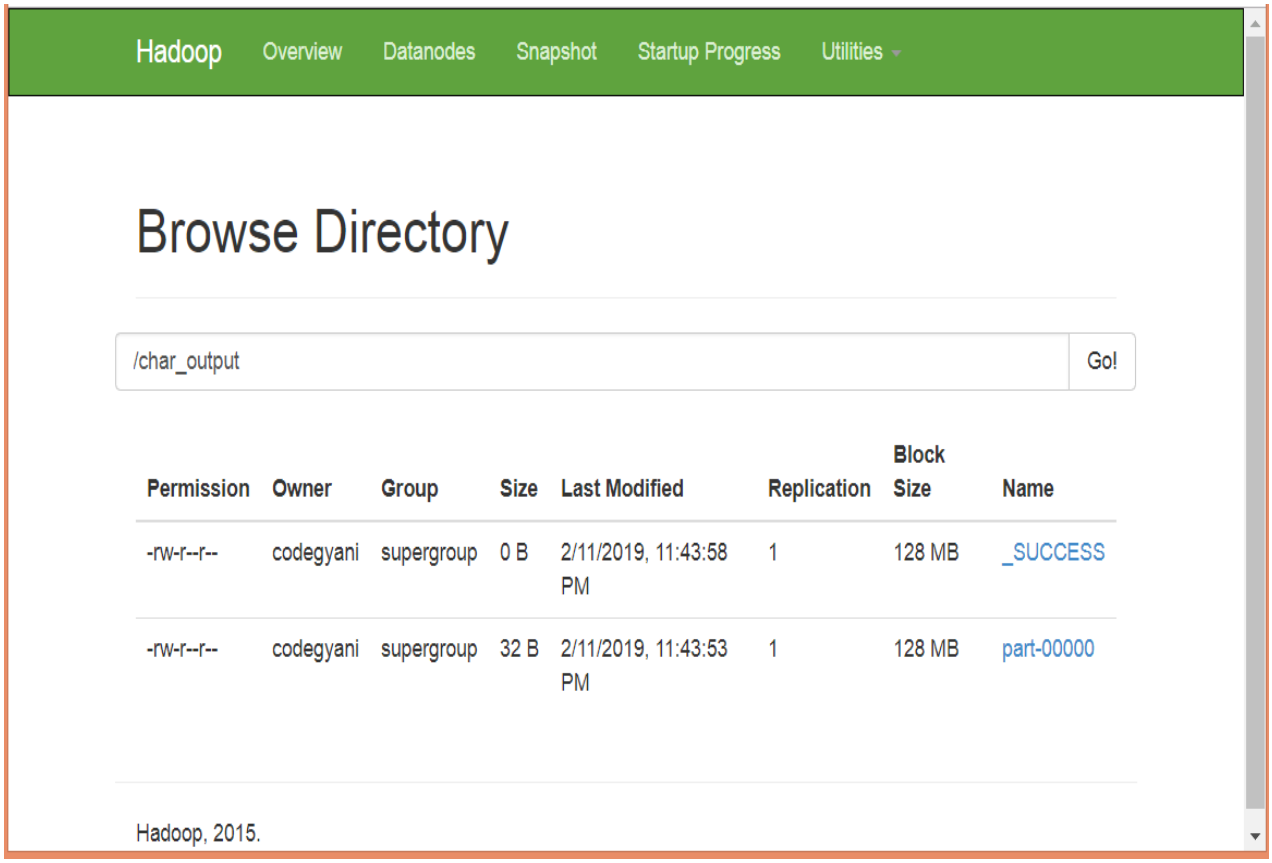
File: WC_Runner.java

```
1.  package com.javatpoint;
2.
```

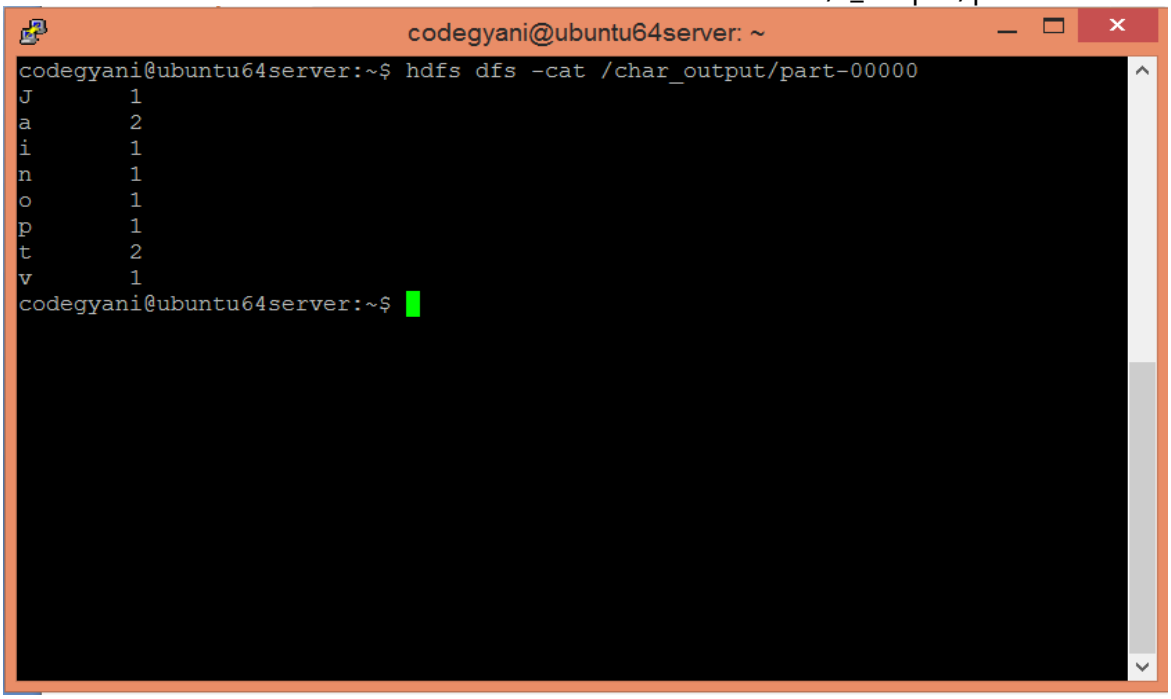
```
3.  import java.io.IOException;
4.  import org.apache.hadoop.fs.Path;
5.  import org.apache.hadoop.io.IntWritable;
6.  import org.apache.hadoop.io.Text;
7.  import org.apache.hadoop.mapred.FileInputFormat;
8.  import org.apache.hadoop.mapred.FileOutputFormat;
9.  import org.apache.hadoop.mapred.JobClient;
10. import org.apache.hadoop.mapred.JobConf;
11. import org.apache.hadoop.mapred.TextInputFormat;
12. import org.apache.hadoop.mapred.TextOutputFormat;
13. public class WC_Runner {
14.     public static void main(String[] args) throws IOException{
15.         JobConf conf = new JobConf(WC_Runner.class);
16.         conf.setJobName("CharCount");
17.         conf.setOutputKeyClass(Text.class);
18.         conf.setOutputValueClass(IntWritable.class);
19.         conf.setMapperClass(WC_Mapper.class);
20.         conf.setCombinerClass(WC_Reducer.class);
21.         conf.setReducerClass(WC_Reducer.class);
22.         conf.setInputFormat(TextInputFormat.class);
23.         conf.setOutputFormat(TextOutputFormat.class);
24.         FileInputFormat.setInputPaths(conf,new Path(args[0]));
25.         FileOutputFormat.setOutputPath(conf,new Path(args[1]));
26.         JobClient.runJob(conf);
27.     }
28. }
```

Download the source code.

- Create the jar file of this program and name it **charcountdemo.jar.**
- Run the jar file hadoop jar /home/codegyani/charcountdemo.jar com.javatpoint.WC_Runner /count/info.txt /char_output
- The output is stored in /char_output/part-00000



- Now execute the command to see the output. hdfs dfs -cat /r_output/part-0000



Q. Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text

input files and finds average for temperature, dew point and wind speed.

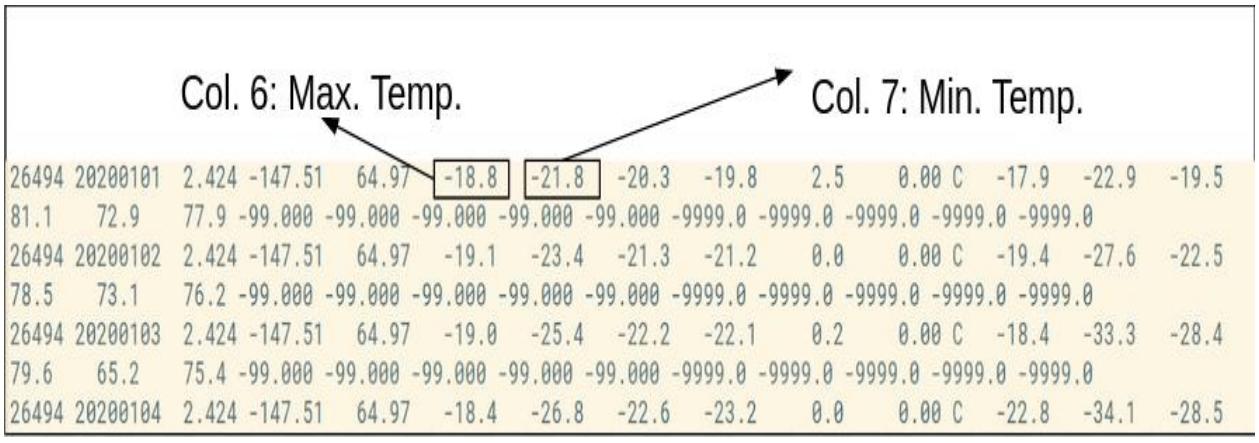
Input:

Step 1:

We can download the dataset from this Link, For various cities in different years. choose the year of your choice and select any one of the data text-file for analyzing. In my case, I have selected *CRND0103-2020-AK_Fairbanks_11_NE.txt* dataset for analysis of hot and cold days in Fairbanks, Alaska.
We can get information about data from *README.txt* file available on the NCEI website.

Step 2:

Below is the example of our dataset where column 6 and column 7 is showing Maximum and Minimum temperature, respectively.

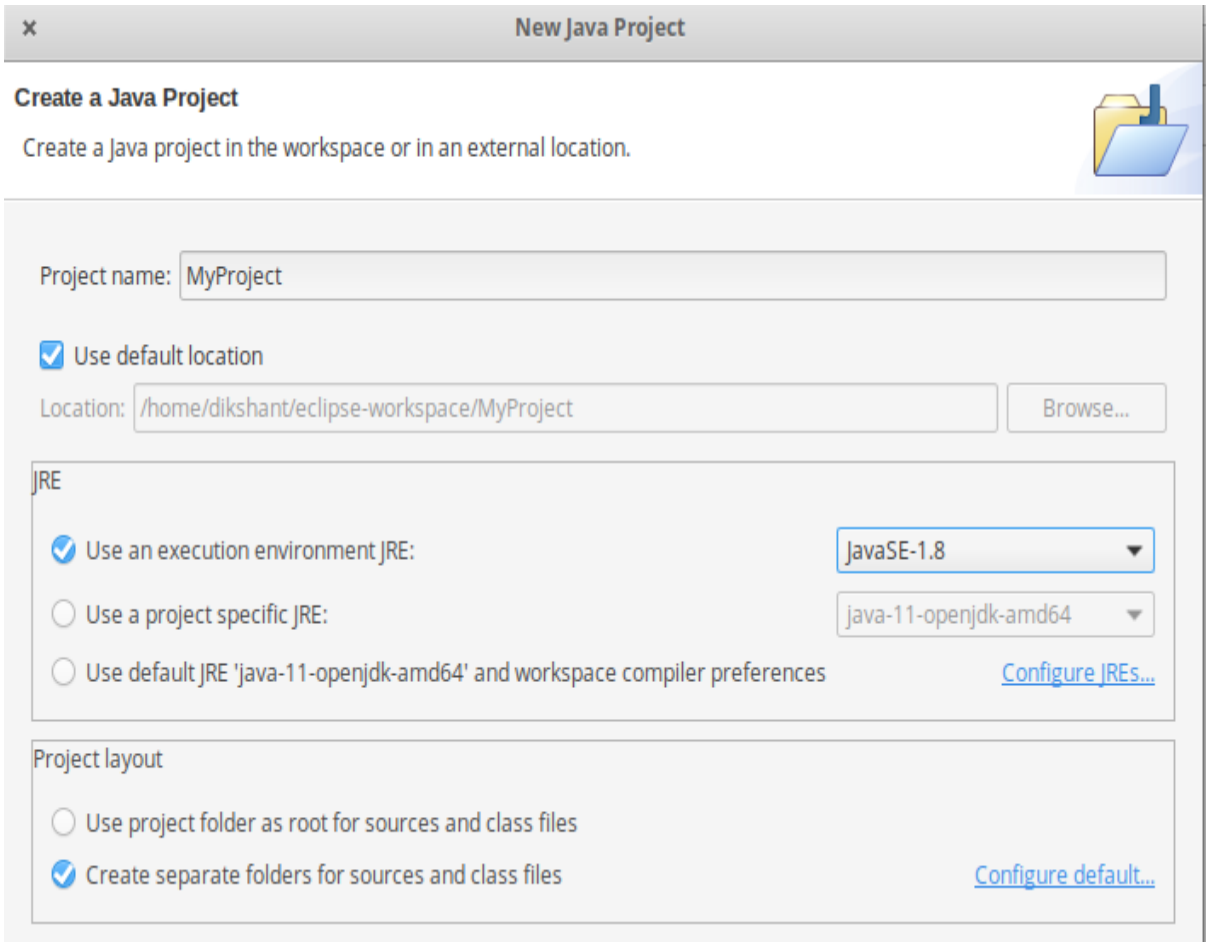


Step 3:

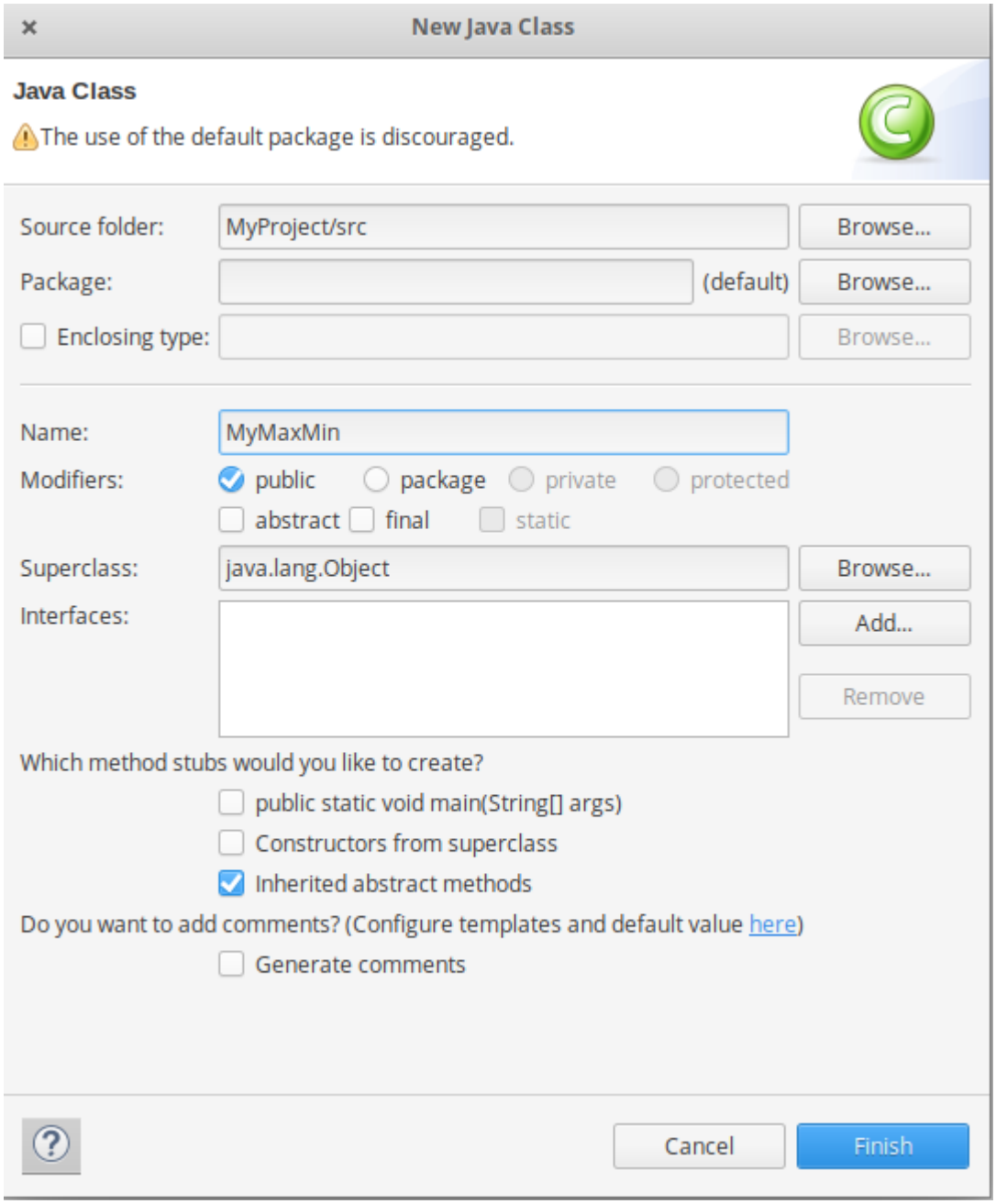
Make a project in Eclipse with below steps:

- First Open **Eclipse** -> then select **File -> New -> Java Project** ->Name it **MyProject** -> then select **use an execution environment** ->

choose **JavaSE-1.8** then **next** -> **Finish**.



- In this Project Create Java class with name **MyMaxMin** -> then click **Finish**



- Copy the below source code to this **MyMaxMin** java class

JAVA

```
// importing Libraries

import java.io.IOException;

import java.util.Iterator;
```

```
import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.conf.Configuration;



public class MyMaxMin {




    // Mapper



    /*MaxTemperatureMapper class is static

     * and extends Mapper abstract class

     * having four Hadoop generics type

     * LongWritable, Text, Text, Text.

    */
```

```
public static class MaxTemperatureMapper extends

        Mapper<LongWritable, Text, Text, Text> {



    /**

    * @method map

    * This method takes the input as a text data type.

    * Now leaving the first five tokens, it takes

    * 6th token is taken as temp_max and

    * 7th token is taken as temp_min. Now

    * temp_max > 30 and temp_min < 15 are

    * passed to the reducer.

    */



// the data in our data set with

// this value is inconsistent data

public static final int MISSING = 9999;



@Override

    public void map(LongWritable arg0, Text Value, Context context)
```

```
        throws IOException, InterruptedException {


// Convert the single row(Record) to

// String and store it in String

// variable name line


String line = Value.toString();


    // Check for the empty line

    if (!(line.length() == 0)) {


        // from character 6 to 14 we have

        // the date in our dataset

        String date = line.substring(6, 14);


        // similarly we have taken the maximum

        // temperature from 39 to 45 characters

        float temp_Max = Float.parseFloat(line.substring(39, 45).trim());


        // similarly we have taken the minimum

        // temperature from 47 to 53 characters
```

```
        float temp_Min = Float.parseFloat(line.substring(47, 53).trim());


        // if maximum temperature is
        // greater than 30, it is a hot day
        if (temp_Max > 30.0) {


            // Hot day
            context.write(new Text("The Day is Hot Day :" + date),
                                    new Text(String.valueOf(temp_Max)));
        }


        // if the minimum temperature is
        // less than 15, it is a cold day
        if (temp_Min < 15) {


            // Cold day
            context.write(new Text("The Day is Cold Day :" + date),
                    new Text(String.valueOf(temp_Min)));
        }
    }
```

```
        }


    }


// Reducer


    /*MaxTemperatureReducer class is static
      and extends Reducer abstract class
      having four Hadoop generics type
      Text, Text, Text, Text.
    */


    public static class MaxTemperatureReducer extends
             Reducer<Text, Text, Text, Text> {


        /**
        * @method reduce
        * This method takes the input as key and
        * list of values pair from the mapper,
        * it does aggregation based on keys and
        * produces the final context.
```

```
    */

    public void reduce(Text Key, Iterator<Text> Values, Context context)
            throws IOException, InterruptedException {

        // putting all the values in
        // temperature variable of type String
        String temperature = Values.next().toString();
        context.write(Key, new Text(temperature));
    }

}

/**
* @method main
* This method is used for setting
* all the configuration properties.
* It acts as a driver for map-reduce
```

```
* code.

*/


public static void main(String[] args) throws Exception {


    // reads the default configuration of the

    // cluster from the configuration XML files

    Configuration conf = new Configuration();


    // Initializing the job with the

    // default configuration of the cluster

    Job job = new Job(conf, "weather example");


    // Assigning the driver class name

    job.setJarByClass(MyMaxMin.class);


    // Key type coming out of mapper

    job.setMapOutputKeyClass(Text.class);


    // value type coming out of mapper

    job.setMapOutputValueClass(Text.class);
```

```
// Defining the mapper class name

job.setMapperClass(MaxTemperatureMapper.class);



// Defining the reducer class name

job.setReducerClass(MaxTemperatureReducer.class);



// Defining input Format class which is

// responsible to parse the dataset

// into a key value pair

job.setInputFormatClass(TextInputFormat.class);



// Defining output Format class which is

// responsible to parse the dataset

// into a key value pair

job.setOutputFormatClass(TextOutputFormat.class);



// setting the second argument

// as a path in a path variable

Path OutputPath = new Path(args[1]);
```

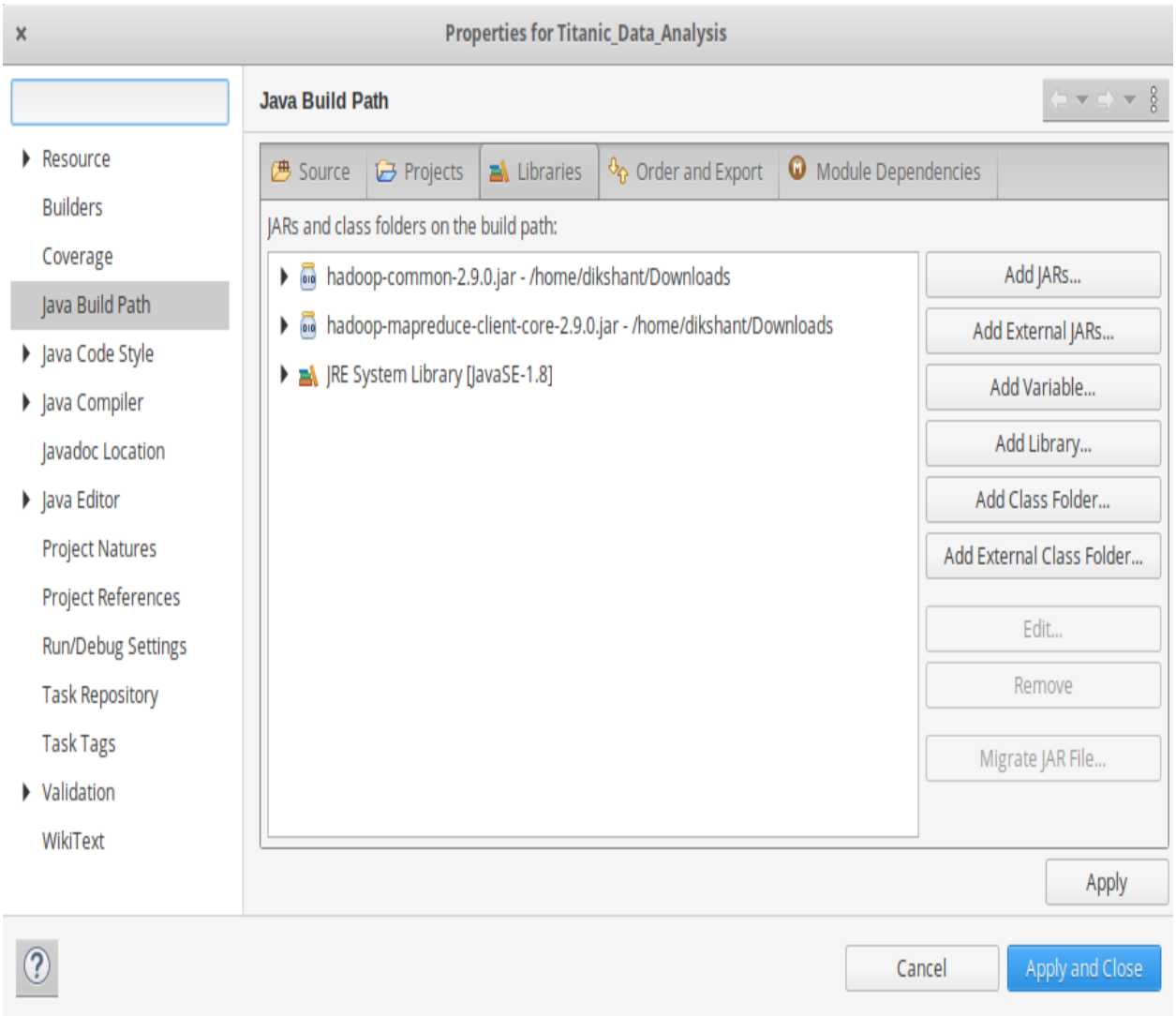
```
        // Configuring the input path

        // from the filesystem into the job

        FileInputFormat.addInputPath(job, new Path(args[0]));



        // Configuring the output path from

        // the filesystem into the job

        FileOutputFormat.setOutputPath(job, new Path(args[1]));



        // deleting the context path automatically

        // from hdfs so that we don't have

        // to delete it explicitly

        OutputPath.getFileSystem(conf).delete(OutputPath);



        // exiting the job only if the

        // flag value becomes false

        System.exit(job.waitForCompletion(true) ? 0 : 1);



    }

}
```

- Now we need to add external jar for the packages that we have import. Download the jar package Hadoop Common and Hadoop MapReduce Core according to your Hadoop version.
  You can check Hadoop Version:

```
hadoop version
```

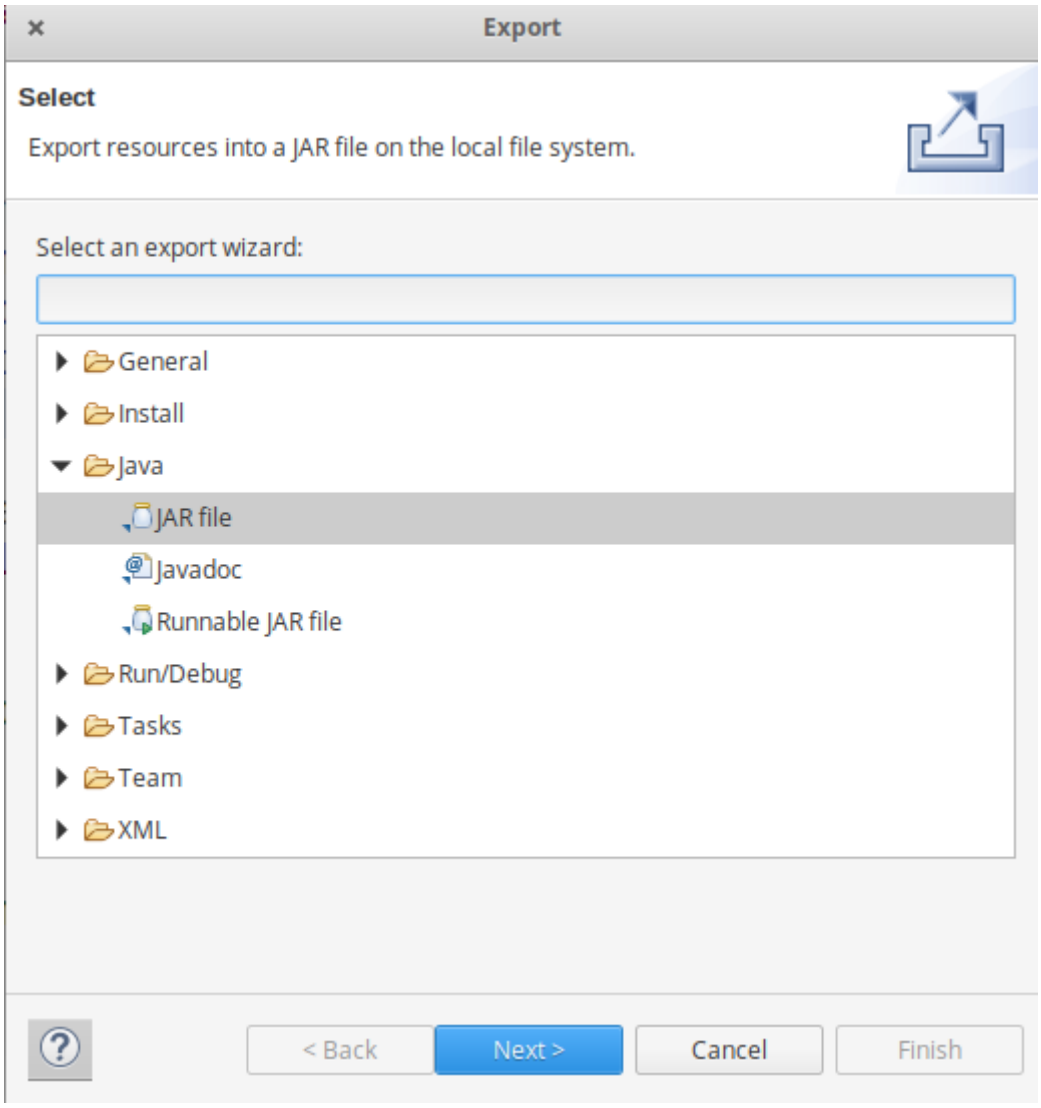
```
dikshant@dikshant-Inspiron-5567:~$ hadoop version
Hadoop 2.9.0
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 756ebc8394e473ac25feac05fa493f6d612e6c50
Compiled by arsuresh on 2017-11-13T23:15Z
Compiled with protoc 2.5.0
From source with checksum 0a76a9a32a5257331741f8d5932f183
```

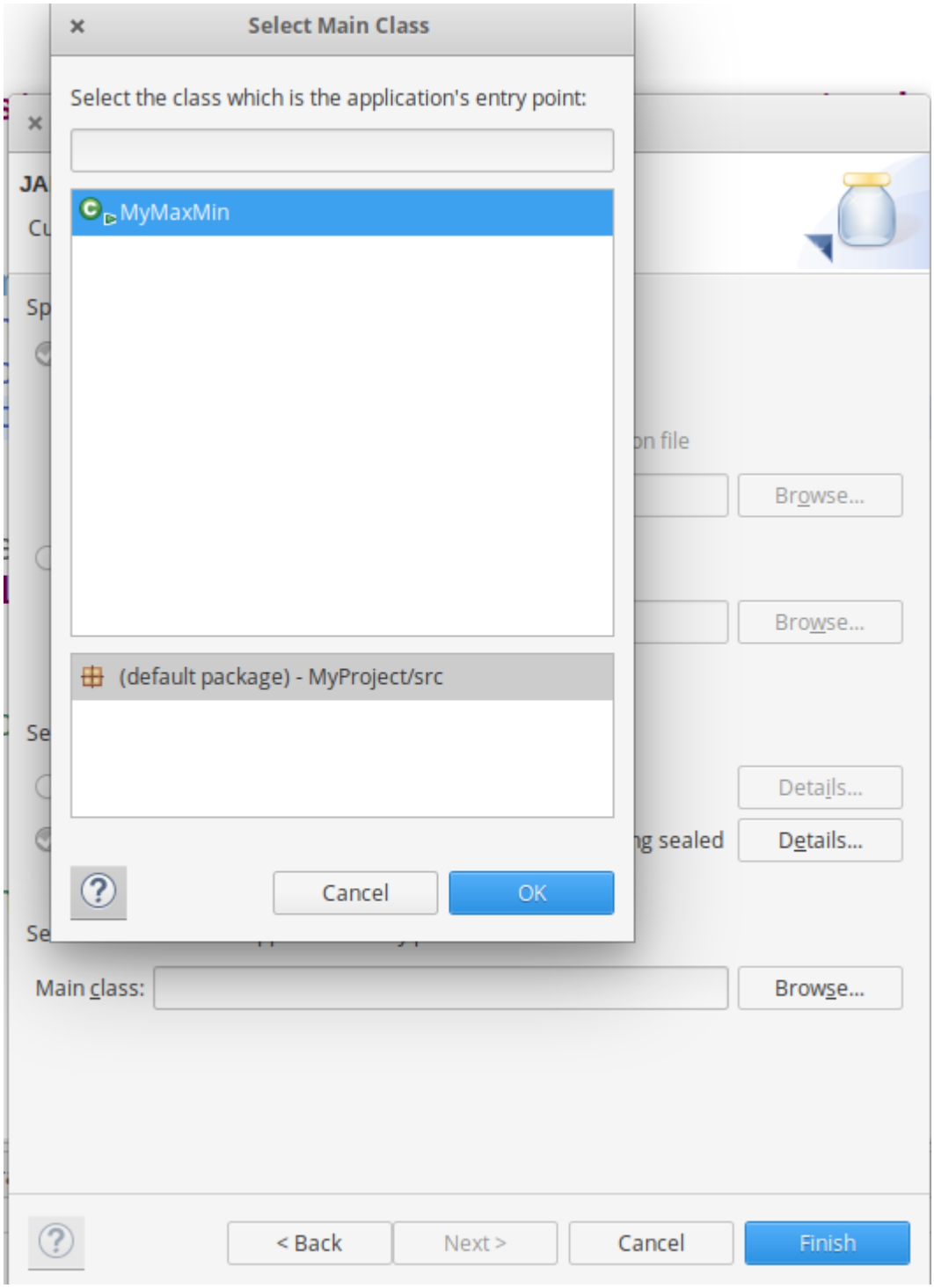
- Now we add these external jars to our **MyProject**. Right Click on **MyProject** -> then select **Build Path**-> Click on **Configure Build Path** and select **Add External jars….** and add jars from it's download location then click -> **Apply and Close**.



- Now export the project as jar file. Right-click on **MyProject** choose **Export..** and go to **Java -> JAR file** click -> **Next** and choose your export destination then click -> **Next**. choose Main Class as **MyMaxMin** by clicking -> **Browse** and then click -

> **Finish** -> **Ok**.


Export
Select
Export resources into a JAR file on the local file system.
Select an export wizard:
General
Install
Java
JAR file
Javadoc
Runnable JAR file
Run/Debug
Tasks
Team
XML
< Back
Next >
Cancel
Finish



Step 4:

Start our Hadoop Daemons

```
start-dfs.sh
start-yarn.sh
```

Step 5:

Move your dataset to the Hadoop HDFS.
Syntax:

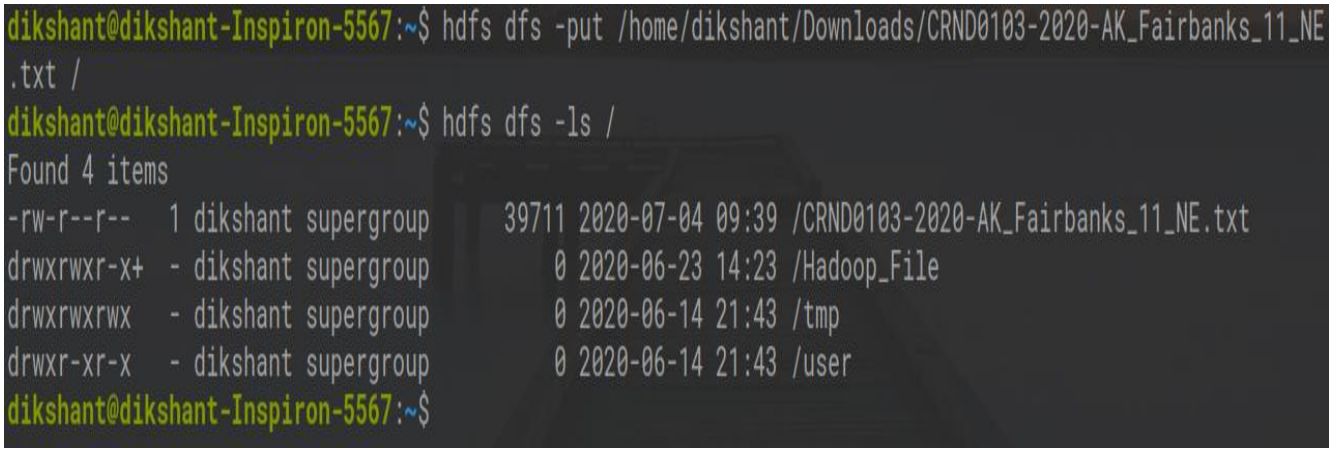
```
hdfs dfs -put /file_path /destination
```

In below command **/** shows the root directory of our HDFS.

```
hdfs dfs -put /home/dikshant/Downloads/CRND0103-2020-
AK_Fairbanks_11_NE.txt /
```

Check the file sent to our HDFS.

```
hdfs dfs -ls /
```



Step 6:

Now Run your Jar File with below command and produce the output in **MyOutput** File.
Syntax:

```
hadoop jar /jar_file_location /dataset_location_in_HDFS /output-
file_name
```

Command:

```
hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-
AK_Fairbanks_11_NE.txt /MyOutput
```

```
dikshant@dikshant-Inspiron-5567:~$ hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-AK_Fairb
anks_11_NE.txt /MyOutput
20/07/04 09:44:40 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.sessi
on-id
20/07/04 09:44:40 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
20/07/04 09:44:41 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Im
```

Step 7:

Now Move to *localhost:50070/*, under utilities select *Browse the file system* and download **part-r-00000** in **/MyOutput** directory to see result.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	dikshant	supergroup	38.78 KB	Jul 04 09:39	1	122.07 MB	CRND0103-2020-AK_Fairbanks_11_NE.txt
drwxrwxr-x+	dikshant	supergroup	0 B	Jun 23 14:23	0	0 B	Hadoop_File
drwxr-xr-x	dikshant	supergroup	0 B	Jul 04 09:44	0	0 B	MyOutput
drwxrwxrwx	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	tmp
drwxr-xr-x	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	user

/MyOutput Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	dikshant	supergroup	0 B	Jul 04 09:44	1	122.07 MB	_SUCCESS
-rw-r--r--	dikshant	supergroup	3.85 KB	Jul 04 09:44	1	122.07 MB	part-r-00000

Step 8:

See the result in the Downloaded File.

```
1 The Day is Cold Day :20200101   -21.8
2 The Day is Cold Day :20200102   -23.4
3 The Day is Cold Day :20200103   -25.4
4 The Day is Cold Day :20200104   -26.8
5 The Day is Cold Day :20200105   -28.8
6 The Day is Cold Day :20200106   -30.0
7 The Day is Cold Day :20200107   -31.4
8 The Day is Cold Day :20200108   -33.6
9 The Day is Cold Day :20200109   -26.6
10 The Day is Cold Day :20200110   -24.3
```

In the above image, you can see the top 10 results showing the cold days. The second column is a day in yyyy/mm/dd format. For Example, **20200101** means

```
year = 2020

month = 01

Date = 01
```