Data Analysis & Visualization Practical Assignment 4

October 8, 2022

Name: Amartya Sinha

Roll No: AC-1207

1. Consider two excel files having attendance of a workshop's participants for two days. Each file has three fields 'Name', 'Time of joining', duration (in minutes) where names are unique within a file. Note that duration may take one of three values (30, 40, 50) only. Import the data into two dataframes and do the following:
    1. Perform merging of the two dataframes to find the names of students who had attended theworkshop on both days.
    2. Find names of all students who have attended workshop on either of the days.
    3. Merge two data frames row-wise and find the total number of records in the data frame.
    4. Merge two data frames and use two columns names and duration as multi-row indexes. Generatedescriptive statistics for this multi-index.
    5. Count number of rows with more than one NAvalues

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

[ ]:
```
day1 = pd.read_excel('DAV_Prac_Attendance.xlsx', 'Sheet1')
day2 = pd.read_excel('DAV_Prac_Attendance.xlsx', 'Sheet2')
```

[ ]:
```
day1
```

[ ]:

	Name	Time of joining	Duration (in minutes)
0	Amartya	10:30:00	30
1	Shahnwaz	10:30:00	30
2	Nilesh	10:30:00	30
3	Divyam	10:30:00	30
4	Aditya	10:10:00	50
5	Ayan	10:20:00	40
6	Aman	10:10:00	50
7	Ramit	10:20:00	40
8	Rajat	10:20:00	40
9	Pratham	10:30:00	30
10	Prakash	10:30:00	30
11	Amitesh	10:30:00	30
12	Anam	10:10:00	50
13	Ananya	10:10:00	50

```
14    Vishal        10:20:00                    40
15       Raj        10:30:00                    30
```

[ ]:
```
day2
```

[ ]:
```
         Name Time of joining  Duration (in minutes)
0     Divyam        10:20:00                    40
1   Shahnwaz        10:20:00                    40
2     Nilesh        10:20:00                    40
3    Amartya        10:30:00                    30
4       Anam        10:10:00                    50
5     Ananya        10:10:00                    50
6     Shreya        10:20:00                    40
7     Jishnu        10:20:00                    40
8     Rajesh        10:30:00                    30
9     Piyush        10:30:00                    30
10       Ayan        10:20:00                    40
11   Prakash        10:20:00                    40
```

[ ]:
```
attendance = pd.merge(day1, day2, on='Name')      # merging both dataframe to␣
 ↪get common students
attendance
```

[ ]:
```
        Name Time of joining_x  Duration (in minutes)_x Time of joining_y  \
0   Amartya          10:30:00                       30          10:30:00
1  Shahnwaz          10:30:00                       30          10:20:00
2    Nilesh          10:30:00                       30          10:20:00
3    Divyam          10:30:00                       30          10:20:00
4      Ayan          10:20:00                       40          10:20:00
5   Prakash          10:30:00                       30          10:20:00
6      Anam          10:10:00                       50          10:10:00
7    Ananya          10:10:00                       50          10:10:00

   Duration (in minutes)_y
0                       30
1                       40
2                       40
3                       40
4                       40
5                       40
6                       50
7                       50
```

[ ]:
```
print(np.union1d(day1['Name'], day2['Name']))     # listing student's name who␣
 ↪have attended either of the workshop
```

```
['Aditya' 'Aman' 'Amartya' 'Amitesh' 'Anam' 'Ananya' 'Ayan' 'Divyam'
 'Jishnu' 'Nilesh' 'Piyush' 'Prakash' 'Pratham' 'Raj' 'Rajat' 'Rajesh'
```

```
 'Ramit' 'Shahnwaz' 'Shreya' 'Vishal']
```

[ ]:
```
merged = pd.concat([day1, day2])            # merging both df row-wise as␣
 ↪concat joins row-wise by default on outer-join
display(merged)
print("There are", len(merged), "Records in the above dataframe")       # count␣
 ↪no of records
```

	Name	Time of joining	Duration (in minutes)
0	Amartya	10:30:00	30
1	Shahnwaz	10:30:00	30
2	Nilesh	10:30:00	30
3	Divyam	10:30:00	30
4	Aditya	10:10:00	50
5	Ayan	10:20:00	40
6	Aman	10:10:00	50
7	Ramit	10:20:00	40
8	Rajat	10:20:00	40
9	Pratham	10:30:00	30
10	Prakash	10:30:00	30
11	Amitesh	10:30:00	30
12	Anam	10:10:00	50
13	Ananya	10:10:00	50
14	Vishal	10:20:00	40
15	Raj	10:30:00	30
0	Divyam	10:20:00	40
1	Shahnwaz	10:20:00	40
2	Nilesh	10:20:00	40
3	Amartya	10:30:00	30
4	Anam	10:10:00	50
5	Ananya	10:10:00	50
6	Shreya	10:20:00	40
7	Jishnu	10:20:00	40
8	Rajesh	10:30:00	30
9	Piyush	10:30:00	30
10	Ayan	10:20:00	40
11	Prakash	10:20:00	40

```
There are 28 Records in the above dataframe
```

[ ]:
```
# d. Merge two data frames and use two columns names and duration as multi-row␣
 ↪indexes. Generate
# descriptive statistics for this multi-index.
```

[ ]:
```
merged_attendance = pd.merge(left = day1, right = day2, how = 'outer', on =␣
 ↪['Name','Duration (in minutes)'])       # merging df on name and duration
merged_attendance.set_index(['Name', 'Duration (in minutes)'])                 ␣
 ↪                                  # setting index names
```

```
my_data = merged_attendance.groupby(['Name', 'Duration (in minutes)'])
 ↪                                  # group by on name and duration
my_data.describe()
 ↪                                  # generate descriptive statistics
```

[ ]:

		Time of joining_x				\
		count	unique	top	freq	
Name	Duration (in minutes)					
Aditya	50	1	1	10:10:00	1	
Aman	50	1	1	10:10:00	1	
Amartya	30	1	1	10:30:00	1	
Amitesh	30	1	1	10:30:00	1	
Anam	50	1	1	10:10:00	1	
Ananya	50	1	1	10:10:00	1	
Ayan	40	1	1	10:20:00	1	
Divyam	30	1	1	10:30:00	1	
	40	0	0	NaN	NaN	
Jishnu	40	0	0	NaN	NaN	
Nilesh	30	1	1	10:30:00	1	
	40	0	0	NaN	NaN	
Piyush	30	0	0	NaN	NaN	
Prakash	30	1	1	10:30:00	1	
	40	0	0	NaN	NaN	
Pratham	30	1	1	10:30:00	1	
Raj	30	1	1	10:30:00	1	
Rajat	40	1	1	10:20:00	1	
Rajesh	30	0	0	NaN	NaN	
Ramit	40	1	1	10:20:00	1	
Shahnwaz	30	1	1	10:30:00	1	
	40	0	0	NaN	NaN	
Shreya	40	0	0	NaN	NaN	
Vishal	40	1	1	10:20:00	1	

		Time of joining_y			
		count	unique	top	freq
Name	Duration (in minutes)				
Aditya	50	0	0	NaN	NaN
Aman	50	0	0	NaN	NaN
Amartya	30	1	1	10:30:00	1
Amitesh	30	0	0	NaN	NaN
Anam	50	1	1	10:10:00	1
Ananya	50	1	1	10:10:00	1
Ayan	40	1	1	10:20:00	1
Divyam	30	0	0	NaN	NaN
	40	1	1	10:20:00	1
Jishnu	40	1	1	10:20:00	1
Nilesh	30	0	0	NaN	NaN

```
         40                                  1       1  10:20:00    1
Piyush   30                                  1       1  10:30:00    1
Prakash  30                                  0       0       NaN  NaN
         40                                  1       1  10:20:00    1
Pratham  30                                  0       0       NaN  NaN
Raj      30                                  0       0       NaN  NaN
Rajat    40                                  0       0       NaN  NaN
Rajesh   30                                  1       1  10:30:00    1
Ramit    40                                  0       0       NaN  NaN
Shahnwaz 30                                  0       0       NaN  NaN
         40                                  1       1  10:20:00    1
Shreya   40                                  1       1  10:20:00    1
Vishal   40                                  0       0       NaN  NaN
```

2. Taking Iris data, plot the following with proper legend and axis labels: (Download IRIS datav from:https://archive.ics.uci.edu/ml/datasets/iris or import it from sklearn.datasets)
    1. Plot bar chart to show the frequency of each class label in the data.
    2. Draw a scatter plot for Petal width vs sepal width.
    3. Plot density distribution for feature petal length.
    4. Use a pair plot to show pairwise bivariate distribution in the Iris Dataset.
    5. Compare five summary distribution information of two features petal width and sepal width using boxplots
    6. Compare five point statistical summary of two features petal width and sepal width using appropriate graph
    7. Draw a piechart showing distribution of three classes

[ ]:
```
from sklearn.datasets import load_iris
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:
```
iris = load_iris()
df = pd.DataFrame(iris. data, columns=iris.feature_names)
df['species'] = pd.Categorical.from_codes(iris.target, iris.target_names)
df
```

[ ]:
```
     sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                  5.1               3.5                1.4               0.2
1                  4.9               3.0                1.4               0.2
2                  4.7               3.2                1.3               0.2
3                  4.6               3.1                1.5               0.2
4                  5.0               3.6                1.4               0.2
..                 …                 …                  …                 …
145                6.7               3.0                5.2               2.3
146                6.3               2.5                5.0               1.9
147                6.5               3.0                5.2               2.0
148                6.2               3.4                5.4               2.3
```

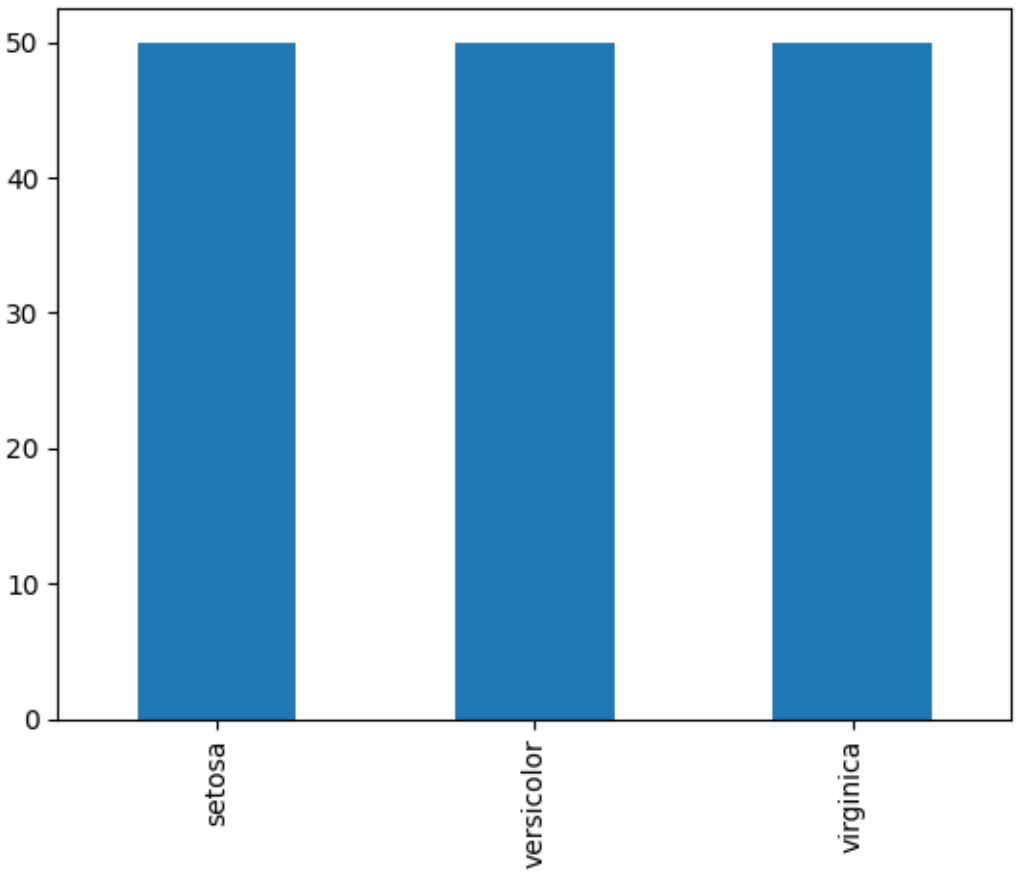
```
149                5.9               3.0                5.1                1.8

       species
0       setosa
1       setosa
2       setosa
3       setosa
4       setosa
..         …
145  virginica
146  virginica
147  virginica
148  virginica
149  virginica

[150 rows x 5 columns]
```

[ ]:
```
df['species'].value_counts()
```

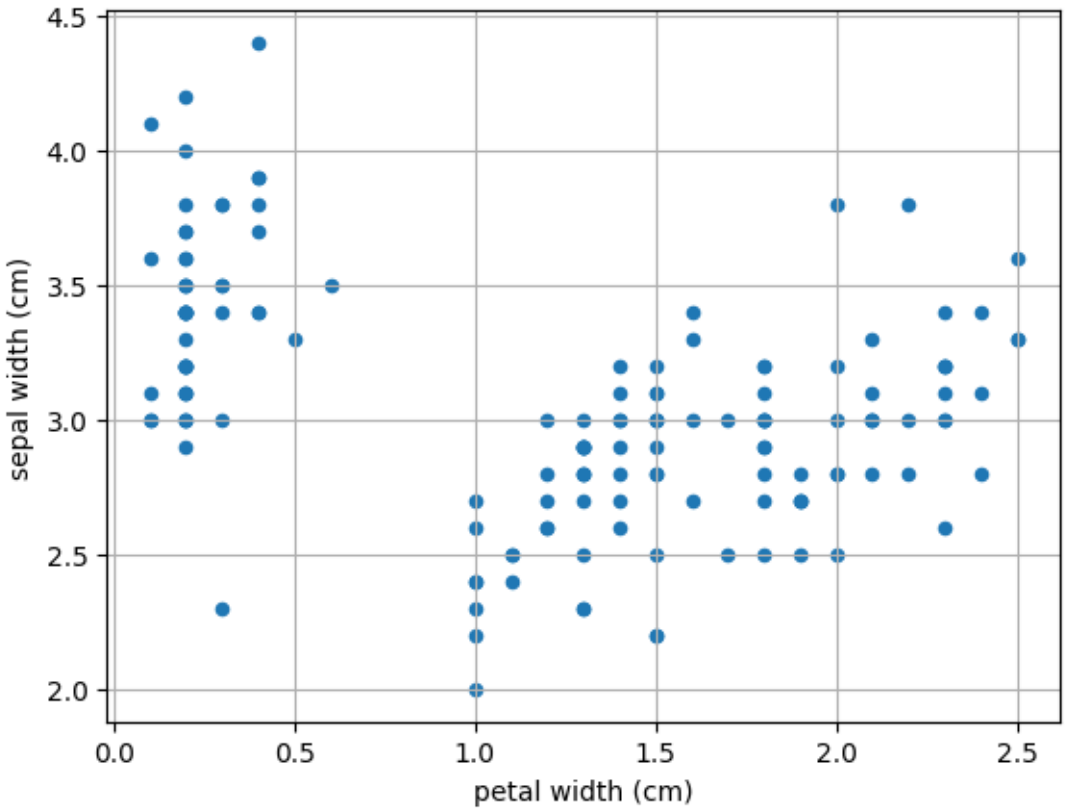
[ ]:
```
setosa        50
versicolor    50
virginica     50
Name: species, dtype: int64
```

[ ]:
```
df['species'].value_counts().plot.bar()     # plot bar graph to show freq of␣
 ↪each species
```

[ ]:
```
<AxesSubplot: >
```

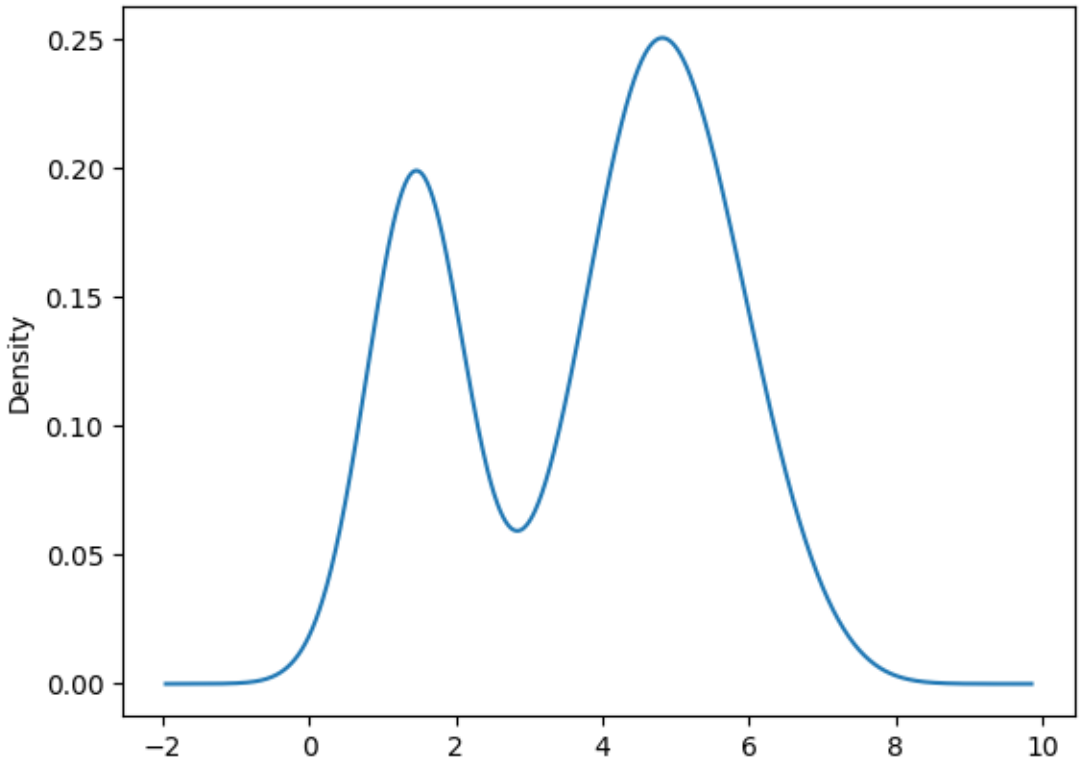


[ ]:
```
df.plot(kind ="scatter",                    # scatter plot to represent petal vs␣
  ↪sepal length
          x ='petal width (cm)',
          y ='sepal width (cm)')
plt.grid()
```



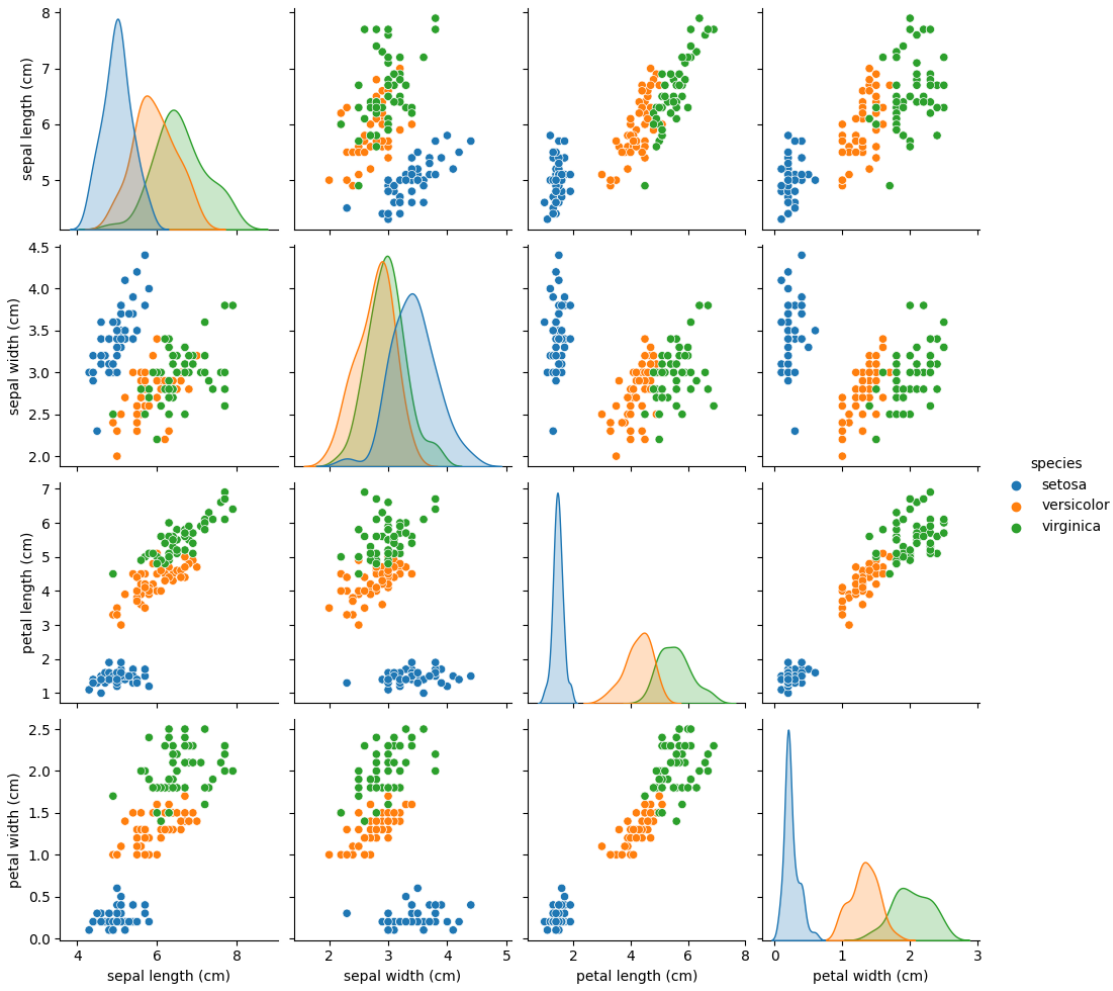
[ ]:
```
df['petal length (cm)'].plot.density()       # density distribution to feature
↪petal length
```

[ ]:
```
<AxesSubplot: ylabel='Density'>
```



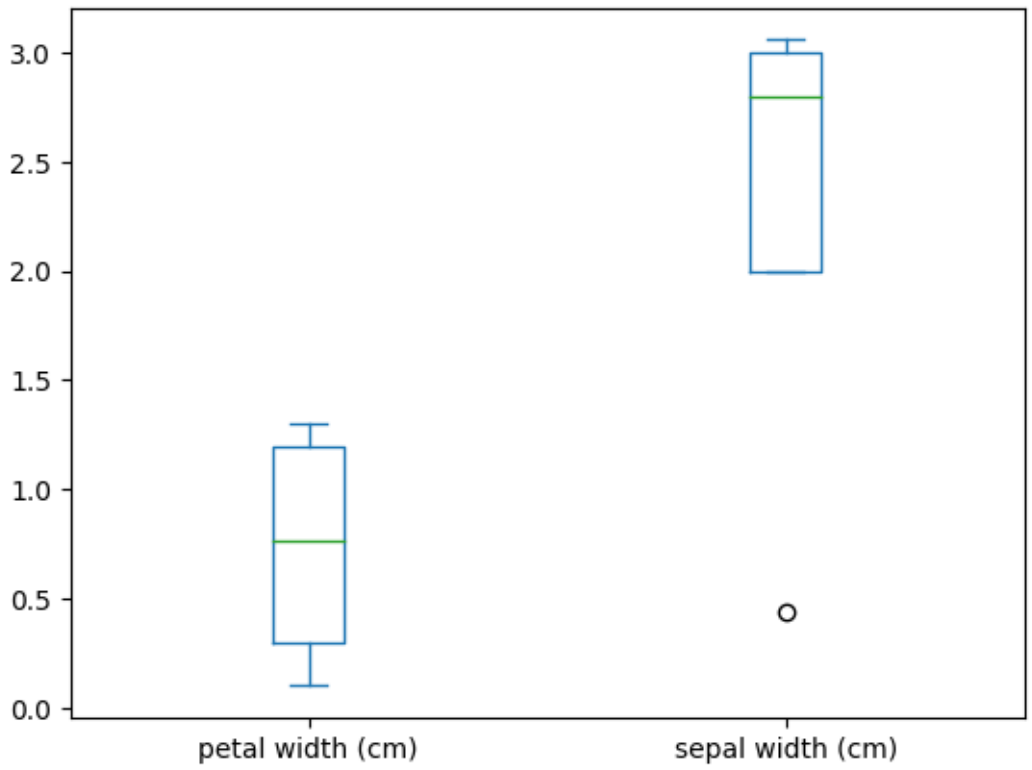
[ ]:
```
sns.pairplot(df, hue = 'species')                    # pairwise bivariate␣
↪distribution using pair plot
```

[ ]:
```
<seaborn.axisgrid.PairGrid at 0x7fb425887400>
```



[ ]:
```
plot_df = df[["petal width (cm)", "sepal width (cm)"]].describe()        #␣
 ↪summary distribution table
plot_df.iloc[1:6].plot.box()                                             # box␣
 ↪plot of summary distribution table
```

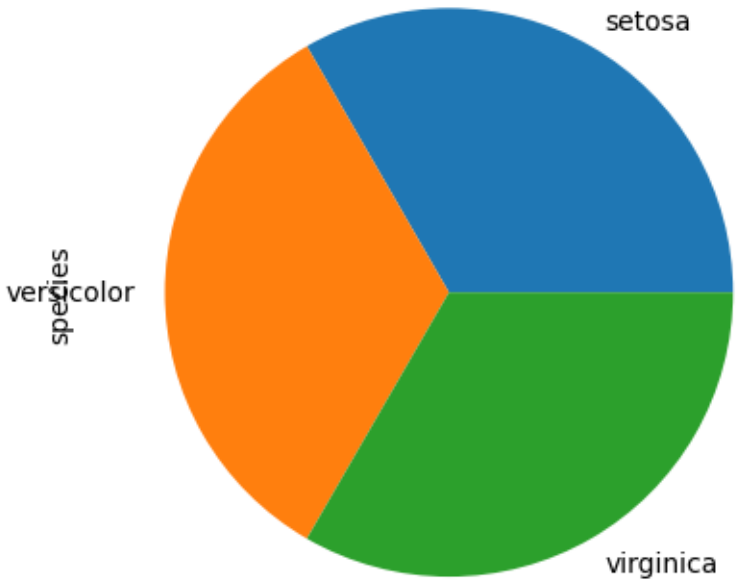
[ ]: <AxesSubplot: >



[ ]:
```
# e and f are same
```

[ ]:
```
df['species'].value_counts().plot.pie()      #piechart showing distribution of␣
↪three classes
```

[ ]:
```
<AxesSubplot: ylabel='species'>
```



3. Create a data frame to store marks of M students for n subjects and do the following:
    1. Find average marks for each student and add as a column
    2. Display average marks of each subject and add as a new row
    3. Compute descriptive statistics subject-wise
    4. Compute grade obtained by each student as per the examination policy of ur course (use lambda function)
    5. Find frequency of each grade for your class
4. Find frequency of each grade obtained by each student and create a new DF as the following and set Rollno as the row index of the DF

[ ]:
```
marks = {'Name': ['Amartya', 'Shahnwaz', 'Nilesh', 'Aditya'], 'DAV':[90, 80,␣
↪85, 70], 'IT':[95,100,96, 85], 'MP':[80,85,70, 65], 'ToC':[85,99,90, 62]}
df = pd.DataFrame(marks)
df
```

[ ]:

	Name	DAV	IT	MP	ToC
0	Amartya	90	95	80	85
1	Shahnwaz	80	100	85	99
2	Nilesh	85	96	70	90
3	Aditya	70	85	65	62

[ ]:
```
df['Average'] = df[['DAV','IT','MP','ToC']].mean(axis=1)              #add␣
↪average marks of each student in column
df
```

[ ]:

	Name	DAV	IT	MP	ToC	Average
0	Amartya	90	95	80	85	87.50
1	Shahnwaz	80	100	85	99	91.00
2	Nilesh	85	96	70	90	85.25
3	Aditya	70	85	65	62	70.50

[ ]:
```
df.loc['Sub Avg'] = df[['DAV','IT','MP','ToC']].mean().round(decimals=1)
  #add avg marks of each sub in row
df
```

[ ]:

	Name	DAV	IT	MP	ToC	Average
0	Amartya	90.0	95.0	80.0	85.0	87.50
1	Shahnwaz	80.0	100.0	85.0	99.0	91.00
2	Nilesh	85.0	96.0	70.0	90.0	85.25
3	Aditya	70.0	85.0	65.0	62.0	70.50
Sub Avg	NaN	81.2	94.0	75.0	84.0	NaN

[ ]:
```
display(df[['DAV','IT','MP','ToC']][0:-1].describe())          #didn't include
  sub avg for descriptive statistics
```

	DAV	IT	MP	ToC
count	4.000000	4.000000	4.000000	4.000000
mean	81.250000	94.000000	75.000000	84.000000
std	8.539126	6.377042	9.128709	15.769168
min	70.000000	85.000000	65.000000	62.000000
25%	77.500000	92.500000	68.750000	79.250000
50%	82.500000	95.500000	75.000000	87.500000
75%	86.250000	97.000000	81.250000	92.250000
max	90.000000	100.000000	85.000000	99.000000

[ ]:
```
df
```

[ ]:

	Name	DAV	IT	MP	ToC	Average
0	Amartya	90.0	95.0	80.0	85.0	87.50
1	Shahnwaz	80.0	100.0	85.0	99.0	91.00
2	Nilesh	85.0	96.0	70.0	90.0	85.25
3	Aditya	70.0	85.0	65.0	62.0	70.50
Sub Avg	NaN	81.2	94.0	75.0	84.0	NaN

[ ]:
```
#calculating grades for each student on the basis of average marks
df['Grades'] = df.apply(lambda x: 'A' if x['Average']>90 else ('B' if
  x['Average']>80 else ('C' if x['Average']>70 else ('F' if x['Average']<33
  else ('D')))), axis=1).head(-1)
```

[ ]:
```
df
```

[ ]:

	Name	DAV	IT	MP	ToC	Average	Grades
0	Amartya	90.0	95.0	80.0	85.0	87.50	B

```
1         Shahnwaz  80.0  100.0  85.0  99.0    91.00      A
2           Nilesh  85.0   96.0  70.0  90.0    85.25      B
3           Aditya  70.0   85.0  65.0  62.0    70.50      C
Sub Avg        NaN  81.2   94.0  75.0  84.0      NaN    NaN
```

[ ]:
```
df.groupby('Grades')['Grades'].count()                    #count frequency of␣
 ↪each grade
```

[ ]:
```
Grades
A    1
B    2
C    1
Name: Grades, dtype: int64
```

[ ]:
```
#add grades of each student subjeect wise
new_df_grades = pd.DataFrame()
new_df_grades['Name'] = df['Name'].head(-1)
new_df_grades.index.names=['Roll No']
for sub in ['DAV', 'IT', 'MP', 'ToC']:
    new_df_grades[sub] = df.apply(lambda x: 'A' if x[sub]>90 else ('B' if␣
 ↪x[sub]>80 else ('C' if x[sub]>70 else ('F' if x[sub]<33 else ('D')))),␣
 ↪axis=1).head(-1)

new_df_grades.index+=1
```

[ ]:
```
new_df_grades
```

[ ]:
```
             Name DAV IT MP ToC
Roll No
1          Amartya   B  A  C   B
2         Shahnwaz   C  A  B   A
3           Nilesh   B  A  D   B
4           Aditya   D  B  D   D
```

[ ]:
```
new_df_grades['Max Grade Obtained'] = new_df_grades.mode(axis=1)[0]
# new_df_grades['Freq'] = new_df_grades.value_counts().mode()
# new_df_grades.mode(axis=1)
new_df_grades['Frequency']=[new_df_grades[['DAV','IT','MP','ToC']].iloc[i].
 ↪value_counts().max() for i in range(len(new_df_grades))]

new_df_grades
```

[ ]:
```
             Name DAV IT MP ToC Max Grade Obtained  Frequency
Roll No
1          Amartya   B  A  C   B                 B          2
2         Shahnwaz   C  A  B   A                 A          2
3           Nilesh   B  A  D   B                 B          2
```

4	Aditya	D	B	D	D		D	3