1 Iris Dataset

1
```
import pandas as pd

names = ['sepalLength', 'sepalWidth', 'petalLength', 'petalWidth', 'class']
iris = pd.read_csv('iris.data', sep=',',names=names)
```

1.1 Summary Statistics

minimum value

2
```
iris.min()
```

2
```
sepalLength            4.3
sepalWidth               2
petalLength              1
petalWidth             0.1
class          Iris-setosa
dtype: object
```

maximun value

3
```
iris.max()
```

3
```
sepalLength               7.9
sepalWidth                4.4
petalLength               6.9
petalWidth                2.5
class          Iris-virginica
dtype: object
```

mean

4
```
iris.mean()
```

4
```
sepalLength    5.843333
sepalWidth     3.054000
petalLength    3.758667
petalWidth     1.198667
dtype: float64
```

range

5
```
for item in names:
    print(item+"["+str(iris[item].min())+","+str(iris[item].max())+"]")
```

```
sepalLength[4.3,7.9]
sepalWidth[2.0,4.4]
petalLength[1.0,6.9]
petalWidth[0.1,2.5]
class[Iris-setosa,Iris-virginica]
```

standard deviation

6
```
iris.std()
```

6
```
sepalLength    0.828066
sepalWidth     0.433594
petalLength    1.764420
petalWidth     0.763161
dtype: float64
```

variance

7
```
iris.var()
```

7
```
sepalLength    0.685694
sepalWidth     0.188004
petalLength    3.113179
petalWidth     0.582414
dtype: float64
```

count

8
```
iris.count()
```

8
```
sepalLength    150
sepalWidth     150
petalLength    150
petalWidth     150
class          150
dtype: int64
```

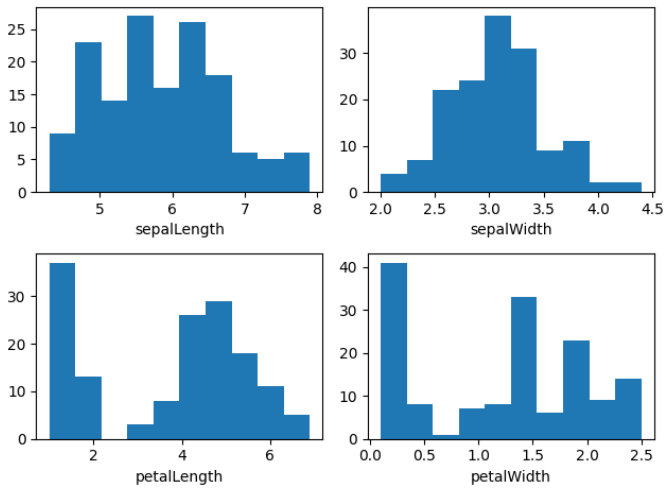
percentiles

9
```
iris.quantile(q=0.25)
iris.quantile(q=0.5)
iris.quantile(q=0.75)
```

9
```
sepalLength    6.4
sepalWidth     3.3
petalLength    5.1
petalWidth     1.8
Name: 0.75, dtype: float64
```

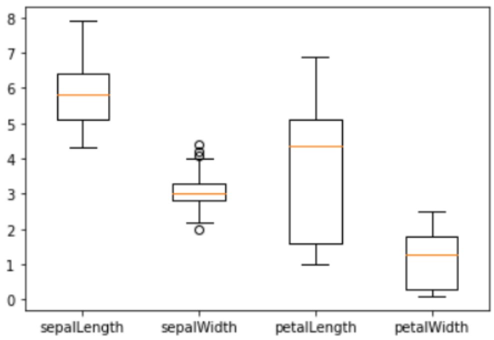
Histograms: To illustrate the feature distributions, create a histogram for each feature in the
combine them all into a single plot. When generating histograms for this assignment, use the
provides a graphical representation of the distribution of the data.

10
```
import matplotlib.pyplot as plt
features=iris[['sepalLength', 'sepalWidth', 'petalLength', 'petalWidth']]
labels='sepalLength', 'sepalWidth', 'petalLength', 'petalWidth'
plt.figure()
plt.subplot(221)
plt.hist(iris['sepalLength'])
plt.xlabel("sepalLength")
plt.subplot(222)
plt.hist(iris['sepalWidth'])
plt.xlabel("sepalWidth")
plt.subplot(223)
plt.hist(iris['petalLength'])
plt.xlabel("petalLength")
plt.subplot(224)
plt.hist(iris['petalWidth'])
plt.xlabel("petalWidth")
plt.show()
```



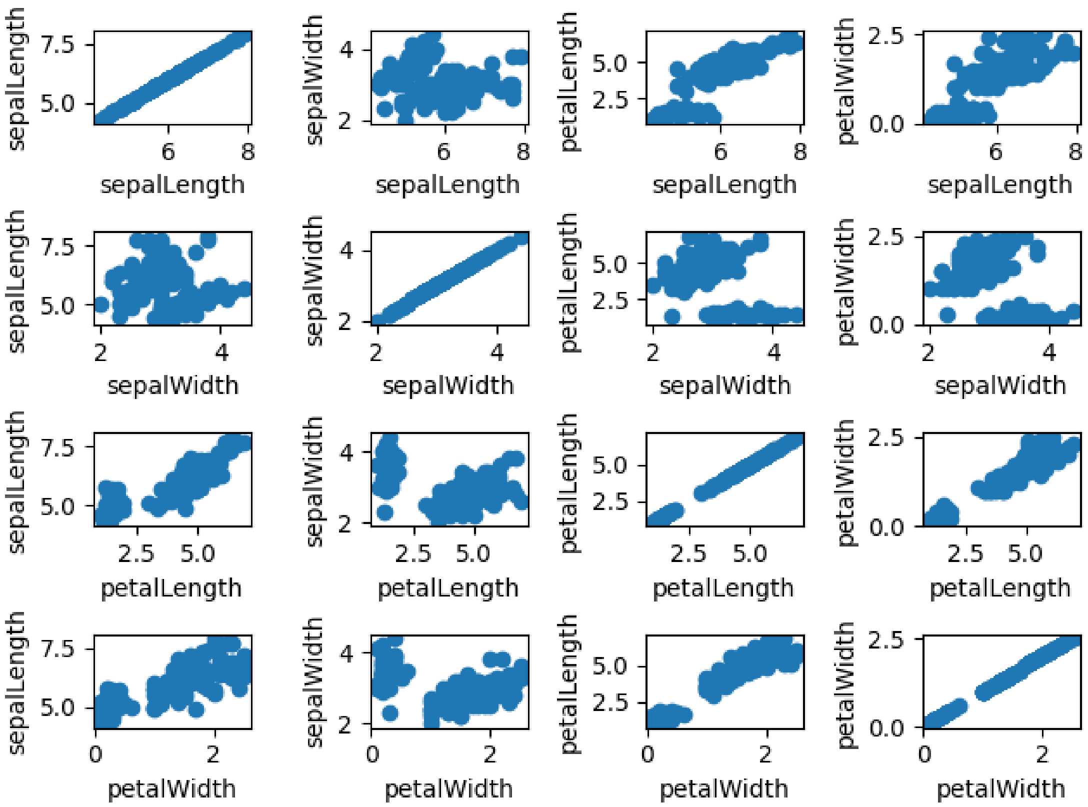
Box Plots: To further understand the data, create a boxplot for each feature in the dataset. Present all the box
boxplot provides a graphical representation of the location and variation of the data through their quartiles;
comparing distributions and identifying outliers.

11
```
plt.figure()
plt.boxplot([iris['sepalLength'],iris['sepalWidth'],iris['petalLength'],iris['petalWidth']],labels=labels)
plt.show()
```



Pairwise Plot: To understand the relationship between the features, create a scatter plot
the dataset, there should be nC2 plots.

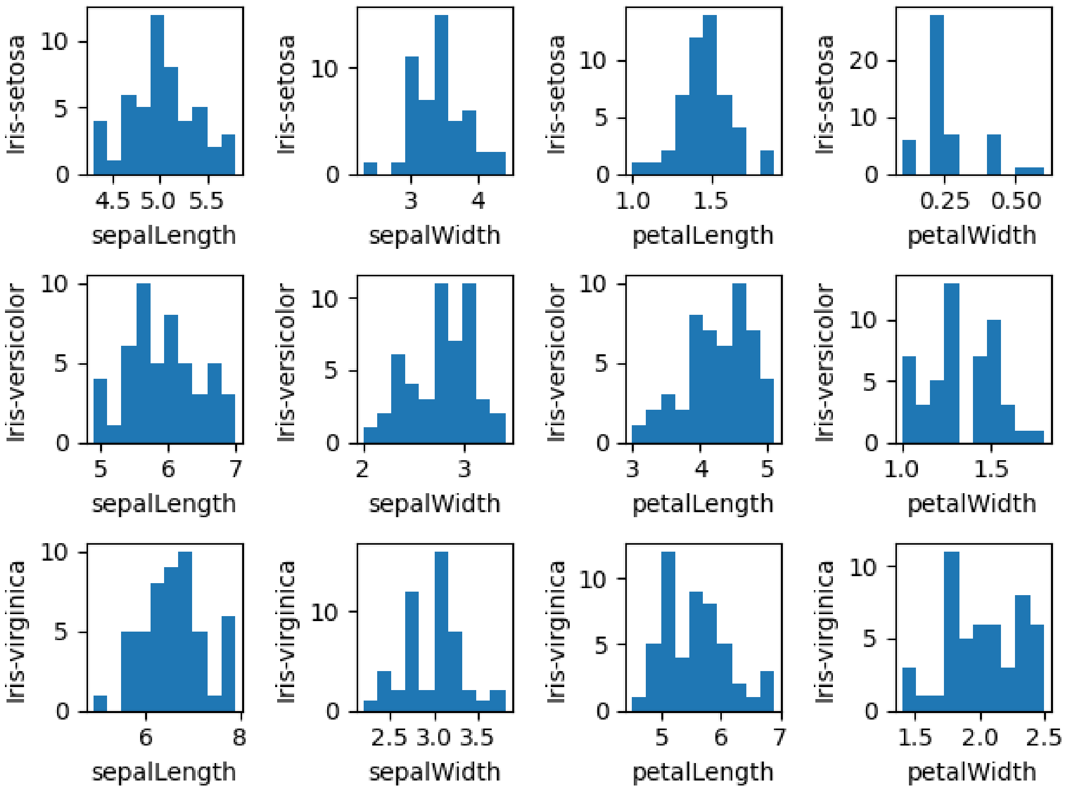
12
```
plt.figure()
i=1
for item1 in features:
    for item2 in features:
        plt.subplot(4,4,i)
        i+=1
        plt.scatter(iris[item1],iris[item2])
        plt.xlabel(item1)
        plt.ylabel(item2)
plt.show()
```



Class-wise Visualization: Create histograms for each feature in a similar way for each of tl

13

```
plt.figure()
i=1
for item in features:
    plt.subplot(3,4,i)
    plt.hist(iris[iris['class']=="Iris-setosa"][item])
    plt.ylabel("Iris-setosa")
    plt.xlabel(item)
    plt.subplot(3,4,i+4)
    plt.hist(iris[iris['class']=="Iris-versicolor"][item])
    plt.ylabel("Iris-versicolor")
    plt.xlabel(item)
    plt.subplot(3,4,i+8)
    plt.hist(iris[iris['class']=="Iris-virginica"][item])
    plt.ylabel("Iris-virginica")
    plt.xlabel(item)
    i+=1
plt.show()
```



1.3 Conceptual Questions

1. How many features are there? What are the Types of the features (e.g., numeric, nominal, discrete, continuous)?

There are five features. Four of them are numeric features and one is string features.

2. From the histograms of the whole data, how do the shapes of the histograms for petal length and petal width differ from those for sepal length and sepal width? Is there a particular value of petal length (which ranges from 1.0 to 6.9) where the distribution of petal lengths (as illustrated by the histogram) could be best segmented into two parts?

The histograms for petal length and petal width are higher at the edge and lower at the center. The histograms for sepal length and sepal width are higher at the center and lower at the edge. The particular values of petal length are zero and more than 30. In about 2.2, pental lengths could be best segmented.

3. Based upon these boxplots, is there a pair of features that appear to have significantly different medians? Recall that the degree of overlap between variability is an important initial indicator of the likelihood that differences in means or medians are meaningful. Also, based solely upon the box plots, which feature appears to explain the greatest amount of the data?

Sepal length. petal length

4. From the pairwise plots of the features, which features are most correlated from the plots? Mention at least three pairs.

sepallength and sepalwidth, sepal width and depallength, petal width and sepal length

5. Compare the histograms of each class to the histograms of the whole dataset. What differences do you see in the shapes?

The histograms of each class are higher at the center While some of the histograms of the whole dataset are higher at the edge

2 Air Quality Dataset

14
```
import pandas as pd
air = pd.read_excel('AirQualityUCI.xlsx')
```

2.1 Summary Statistics

minimum value

15
```
air.min()
```

15
```
Date           2004-03-10 00:00:00
Time                      00:00:00
CO(GT)                        -200
PT08.S1(CO)                   -200
NMHC(GT)                      -200
C6H6(GT)                      -200
PT08.S2(NMHC)                 -200
NOx(GT)                       -200
PT08.S3(NOx)                  -200
NO2(GT)                       -200
PT08.S4(NO2)                  -200
PT08.S5(O3)                   -200
T                             -200
RH                            -200
AH                            -200
dtype: object
```

maximun value

16
```
air.max()
```

16
```
Date           2005-04-04 00:00:00
Time                      23:00:00
CO(GT)                        11.9
PT08.S1(CO)                2039.75
NMHC(GT)                      1189
C6H6(GT)                   63.7415
PT08.S2(NMHC)                 2214
NOx(GT)                       1479
PT08.S3(NOx)               2682.75
NO2(GT)                      339.7
PT08.S4(NO2)                  2775
PT08.S5(O3)                2522.75
T                             44.6
RH                          88.725
AH                         2.23104
dtype: object
```

mean

17
```
air[air.columns[1:]].mean()
```

17
```
CO(GT)            -34.207524
PT08.S1(CO)      1048.869652
NMHC(GT)         -159.090093
C6H6(GT)            1.865576
PT08.S2(NMHC)     894.475963
NOx(GT)           168.604200
PT08.S3(NOx)      794.872333
NO2(GT)            58.135898
PT08.S4(NO2)     1391.363266
PT08.S5(O3)       974.951534
```

```
PT08.S5(O3)       974.951534
T                   9.776600
RH                 39.483611
AH                 -6.837604
dtype: float64
```

range

19
```python
for item in air.columns:
    print(item+"["+str(air[item].min())+","+str(air[item].max())+"]")
```

```
Date[2004-03-10 00:00:00,2005-04-04 00:00:00]
Time[00:00:00,23:00:00]
CO(GT)[-200.0,11.9]
PT08.S1(CO)[-200.0,2039.75]
NMHC(GT)[-200,1189]
C6H6(GT)[-200.0,63.74147644829163]
PT08.S2(NMHC)[-200.0,2214.0]
NOx(GT)[-200.0,1479.0]
PT08.S3(NOx)[-200.0,2682.75]
NO2(GT)[-200.0,339.7]
PT08.S4(NO2)[-200.0,2775.0]
PT08.S5(O3)[-200.0,2522.75]
T[-200.0,44.60000038147]
RH[-200.0,88.72500038147]
AH[-200.0,2.2310357155831864]
```

standard deviation

20
```
air.std()
```

20
```
CO(GT)             77.657170
PT08.S1(CO)       329.817015
NMHC(GT)          139.789093
C6H6(GT)           41.380154
PT08.S2(NMHC)     342.315902
NOx(GT)           257.424561
PT08.S3(NOx)      321.977031
NO2(GT)           126.931428
PT08.S4(NO2)      467.192382
PT08.S5(O3)       456.922728
T                  43.203438
RH                 51.215645
AH                 38.976670
dtype: float64
```

variance

21
```
air.var()
```

21
```
CO(GT)             6030.636106
PT08.S1(CO)      108779.263095
NMHC(GT)          19540.990493
C6H6(GT)           1712.317143
PT08.S2(NMHC)    117180.176653
NOx(GT)           66267.404793
PT08.S3(NOx)     103669.208719
NO2(GT)           16111.587462
PT08.S4(NO2)     218268.721729
PT08.S5(O3)      208778.379165
T                  1866.537024
RH                 2623.042273
AH                 1519.180817
dtype: float64
```

count

22
```
air.count()
```

22
```
Date             9357
Time             9357
CO(GT)           9357
PT08.S1(CO)      9357
NMHC(GT)         9357
C6H6(GT)         9357
PT08.S2(NMHC)    9357
NOx(GT)          9357
PT08.S3(NOx)     9357
NO2(GT)          9357
PT08.S4(NO2)     9357
PT08.S5(O3)      9357
T                9357
RH               9357
AH               9357
dtype: int64
```

percentiles

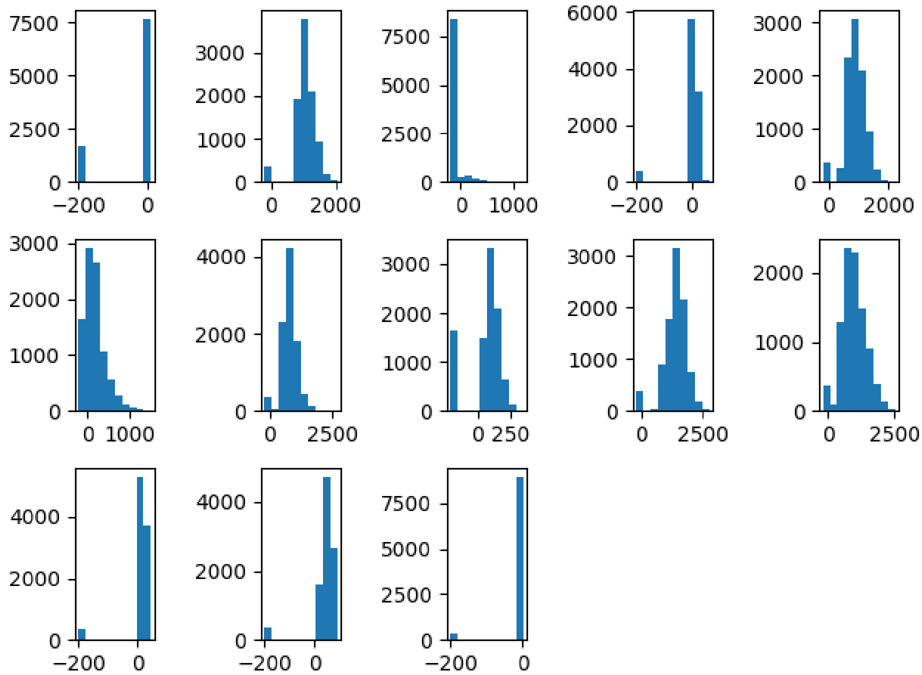
23
```
air.quantile(q=0.25)
air.quantile(q=0.5)
air.quantile(q=0.75)
```

23
```
CO(GT)              2.600000
PT08.S1(CO)      1221.250000
NMHC(GT)         -200.000000
C6H6(GT)           13.636091
PT08.S2(NMHC)    1104.750000
NOx(GT)           284.200000
PT08.S3(NOx)      960.250000
NO2(GT)           133.000000
PT08.S4(NO2)     1662.000000
PT08.S5(O3)      1255.250000
T                  24.075000
RH                 61.875000
AH                  1.296223
Name: 0.75, dtype: float64
```

2.2 Data Visualization

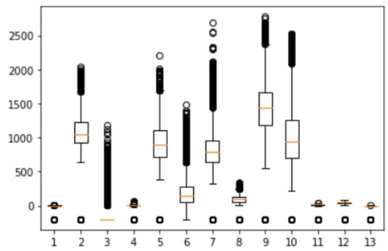
historgrams with outliers

24
```
import matplotlib.pyplot as plt
plt.figure()
i=1
for item in air.columns[2:]:
    plt.subplot(3,5,i)
    plt.hist(air[item])
    i+=1
plt.show()
```



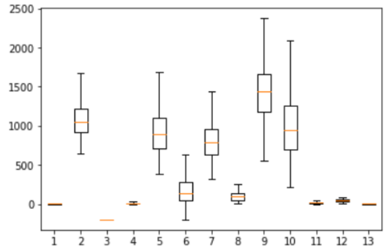
boxplots without outliers

```
features=air.columns[2:]
plt.figure()
plt.boxplot([air[features[0]],air[features[1]],air[features[2]],air[features[3]],air[features[4]],air[features[5]],air[features[6]],air[features[7]],air[feat
plt.show()
```



boxplots with outliers

```
plt.figure()
plt.boxplot([air[features[0]],air[features[1]],air[features[2]],air[features[3]],air[features[4]],air[features[5]],air[features[6]],air[features[7]],air[feat
plt.show()
```



2.2 Data Visualization

28
```
air[air.columns[1:]].mean()
```

28
```
CO(GT)             -33.451844
PT08.S1(CO)       1098.246709
NMHC(GT)          -160.317944
C6H6(GT)            10.011073
PT08.S2(NMHC)      937.862020
NOx(GT)            162.834931
PT08.S3(NOx)       834.204508
NO2(GT)             58.224741
PT08.S4(NO2)      1456.386983
PT08.S5(O3)       1020.464706
T                   18.293017
RH                  49.242447
AH                   1.024218
dtype: float64
```

1. From the histograms, what abnormality can you see?

There are some histograms much higher than others.

2. What abnormality can you see from the summary statistics?

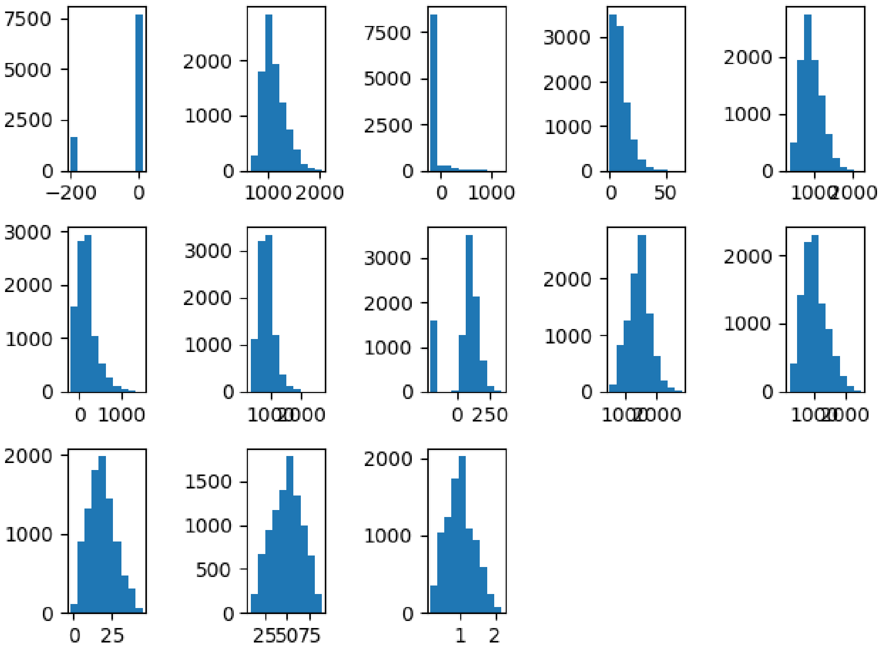
The variance and deviation are abnormally high.

3. How can you remove the abnormality from the data?

Using the box chart approach, outliers exceeding the upper quartile by 1.5 times the distance or the lower quartile by 1.5 times the distance are counted as outliers, filled with the median

4. Show how the histograms look after removing the abnormalities from the data?

27
```
import numpy as np
air = pd.read_excel('AirQualityUCI.xlsx')
for item in air.columns[2:]:
    a = air[item].quantile(0.75)
    b = air[item].quantile(0.25)
air[(air[item]>=(a-b)*1.5+a)|(air[item]<=b-(a-b)*1.5)]=np.nan
air.fillna(air.median(),inplace=True)
```



7500
5000
2500
0
−200
0
2000
1000
0
1000
2000
7500
5000
2500
0
0
1000
3000
2000
1000
0
0
50
2000
1000
0
1000
2000
3000
2000
1000
0
0
1000
3000
2000
1000
0
1000
2000
3000
2000
1000
0
0
250
2000
1000
0
1000
2000
2000
1000
0
1000
2000
2000
1000
0
0
25
1500
1000
500
0
25
50
75
2000
1000
0
1
2