Name: Chenyang Yu SUID: 862052273

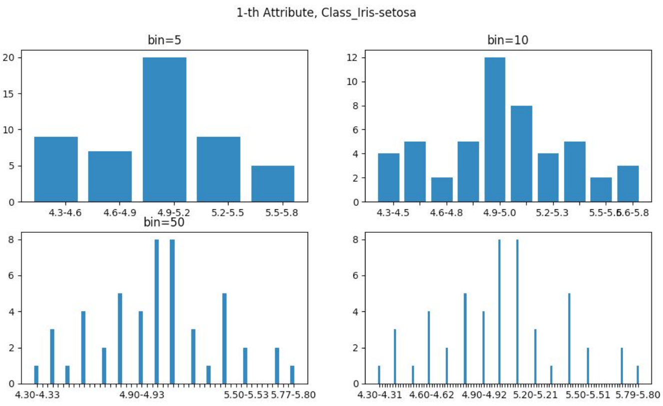
Question 1.

The source code is under folder Q1/. The figure in detail can be found under Q1/Iris_histo Q1/Wine_histo Q1/Iris_box Q1/Wine_box

(1) Date Set: Iris

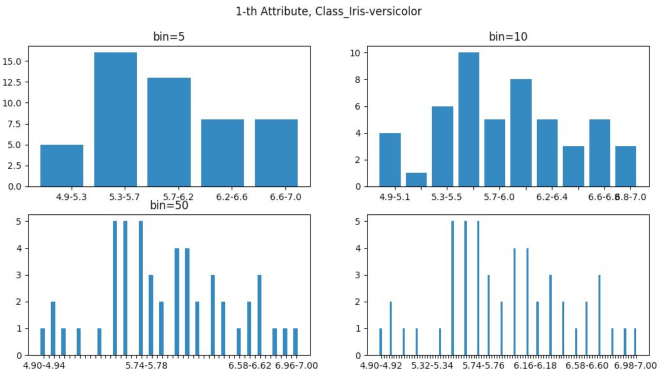
Attribute_1 (Sepal length)

Class_1 (Iris-setosa) with different bin size



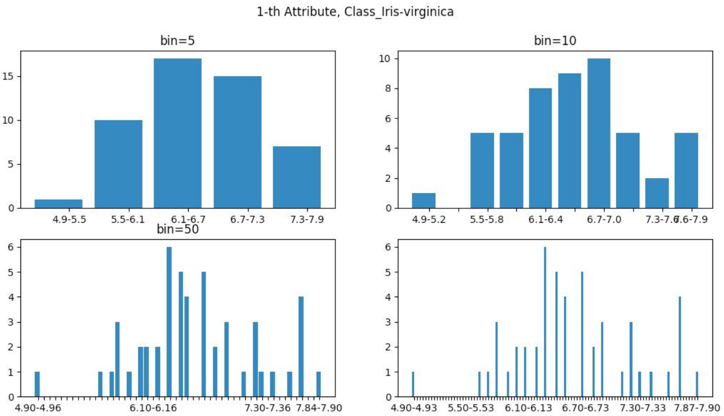
=> most symmetric, mostly unimodal

Class_2 (Iris-versicolor) with different bin size



=> most skewed, mostly unimodal

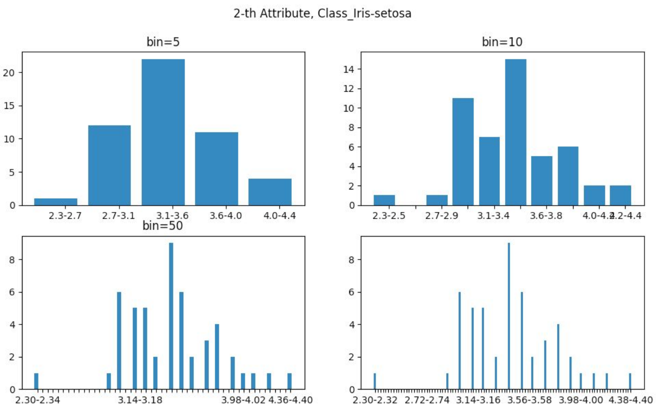
Class_3 (Iris-virginica) with different bin size



=> most symmetric, mostly bi-modal

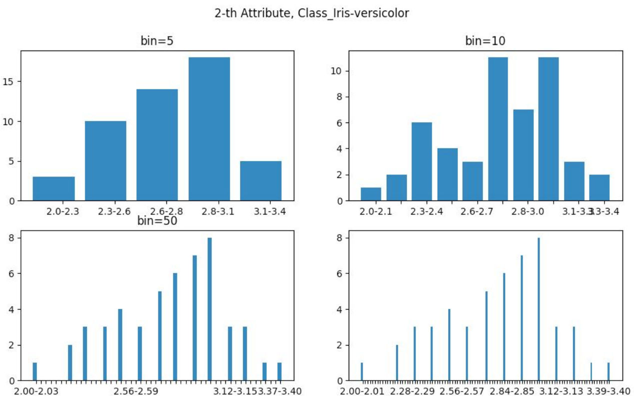
Attribute_2 (Sepal width)

Class_1 (Iris-setosa) with different bin size



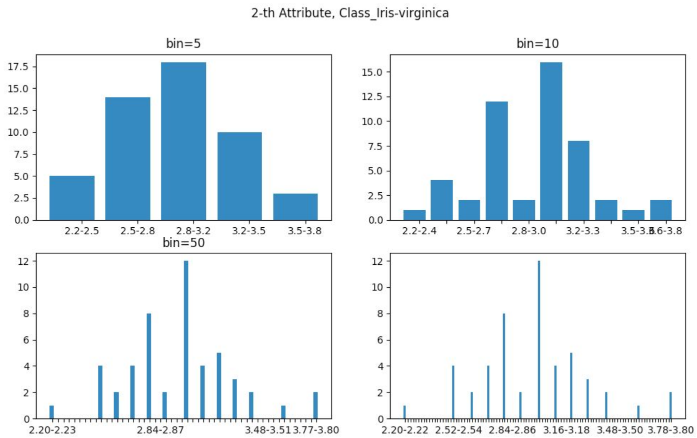
=> most symmetric, mostly unimodal

Class_2 (Iris-versicolor) with different bin size



=> most skewed, mostly unimodal

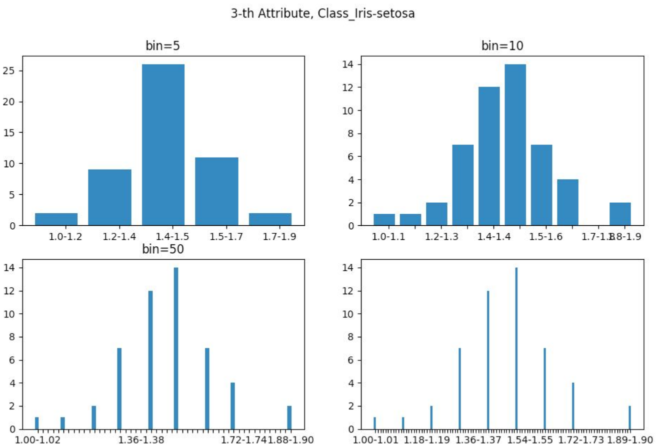
Class_3 (Iris-virginica) with different bin size

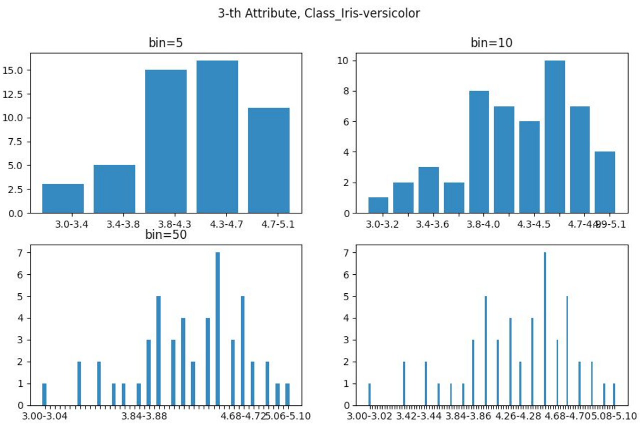


=> most symmetric, mostly bi-modal

Attribute_3 (Petal length)

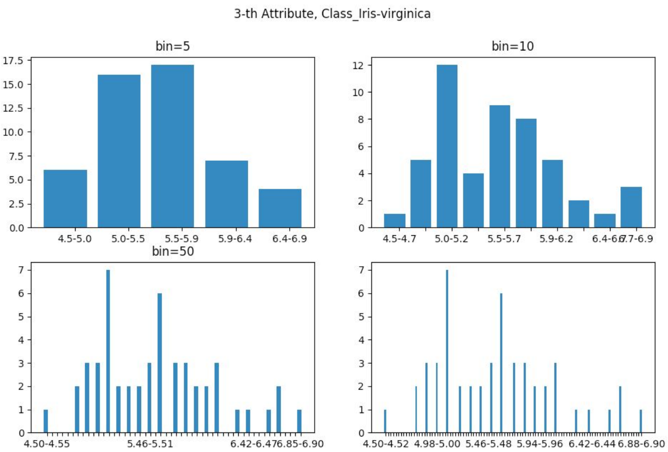
Class_1 (Iris-setosa) with different bin size Class_2 (Iris-versicolor) with different bin size





=> most symmetric, mostly unimodal => most skewed, mostly bi-modal

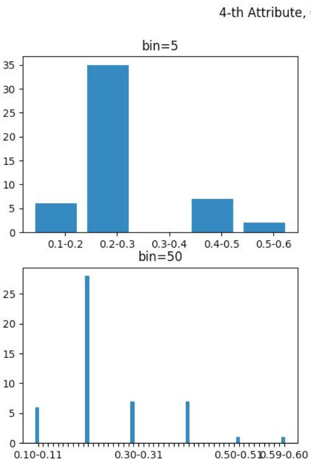
Class_3 (Iris-virginica) with different bin size



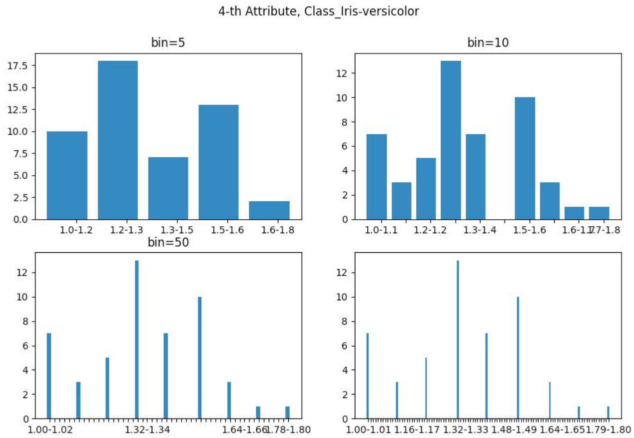
=> most skewed, mostly bi-modal

Attribute_4 (Petal width)

Class_1 (Iris-setosa) with different bin size Class_2 (Iris-versicolor) with different bin size

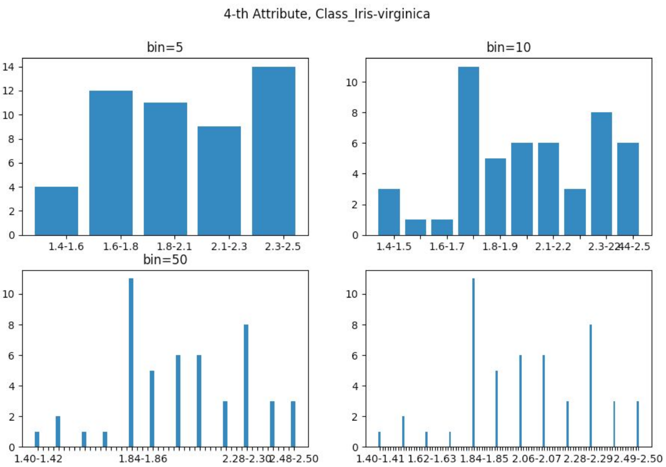


=> most skewed, mostly unimodal



=> most symmetric, mostly bi-modal

Class_3 (Iris-virginica) with different bin size

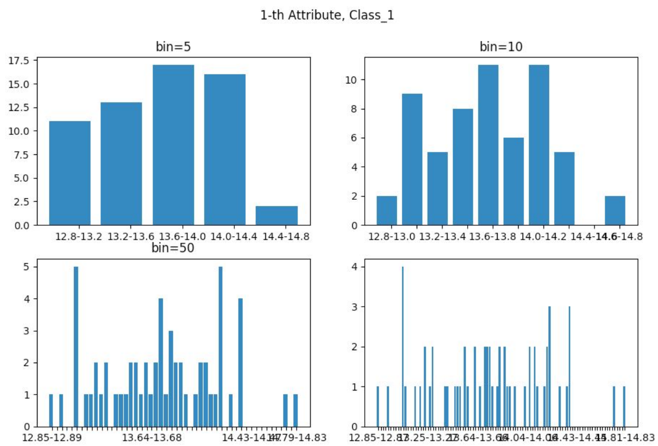


=> most symmetric, mostly bi-modal

Data set: Wine

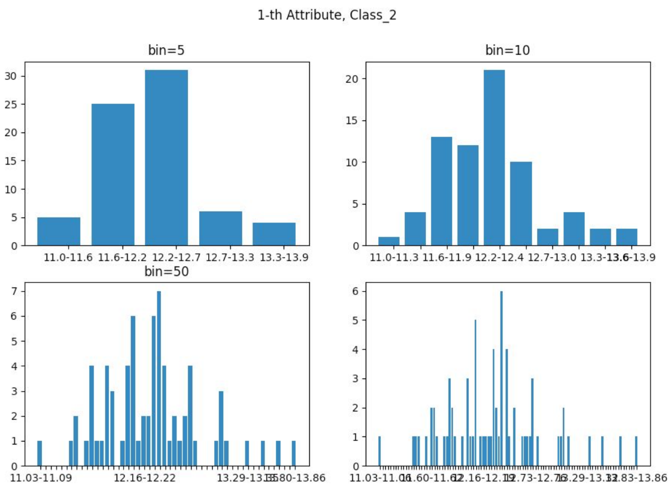
Attribute_1 (Alcohol)

Class_1 with different bin size



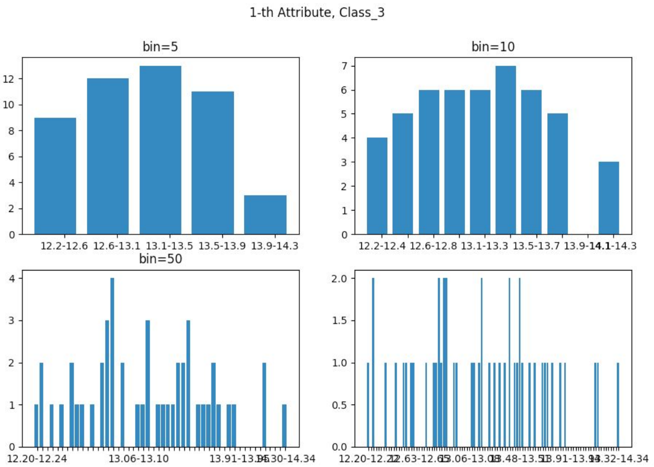
=> most symmetric, mostly uniform

Class_2 with different bin size



=> most symmetric, mostly unimodal

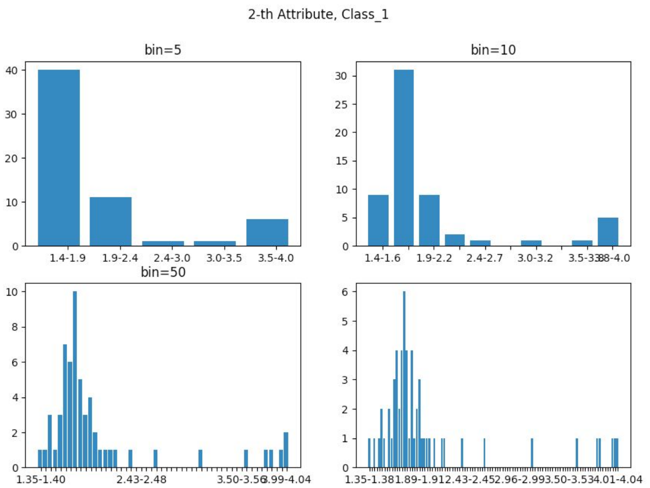
Class_3 with different bin size



=> most symmetric, mostly uniform

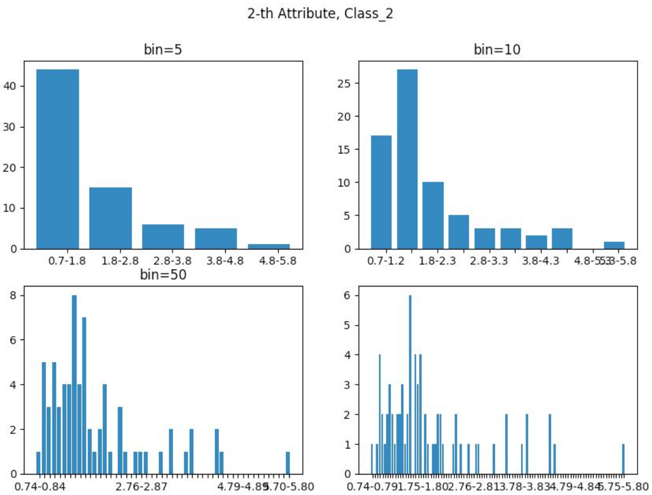
Attribute_2 (Malic acid)

Class_1 with different bin size



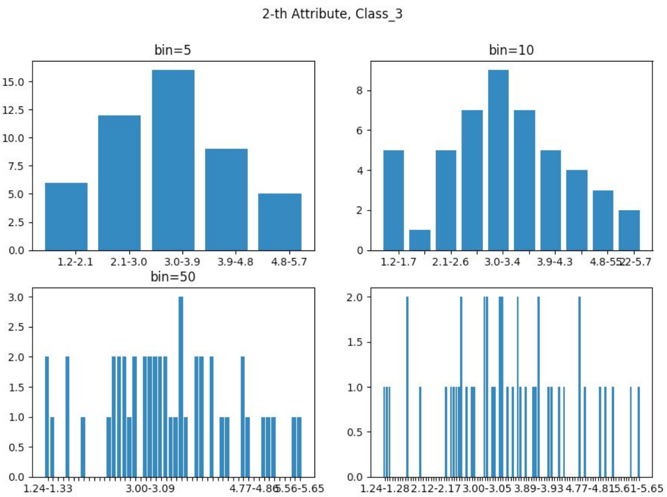
=> most skewed, mostly unimodal

Class_2 with different bin size



=> most skewed, mostly unimodal

Class_3 with different bin size

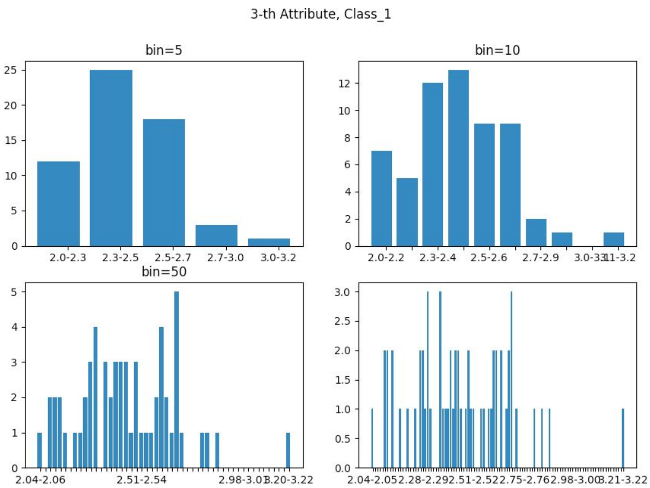


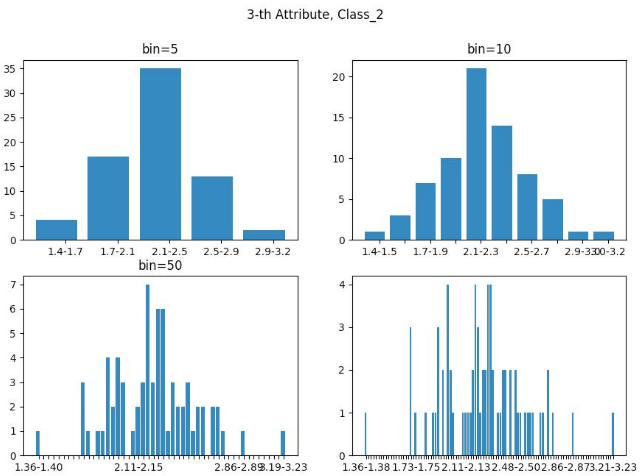
=> most symmetric, mostly unimodal

Attribute_3 (Ash)

Class_1 with different bin size

Class_2 with different bin size

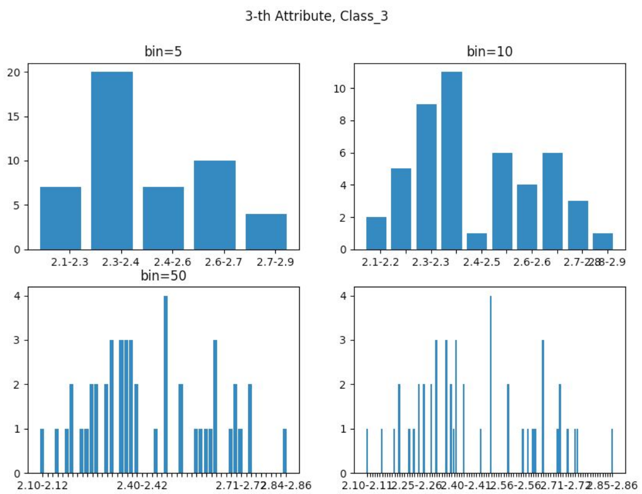




=> most symmetric, mostly unimodal

=> most symmetric, mostly unimodal

Class_3 with different bin size

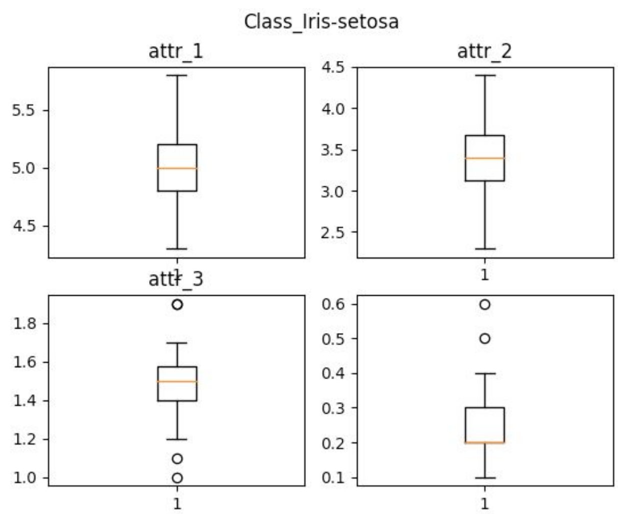


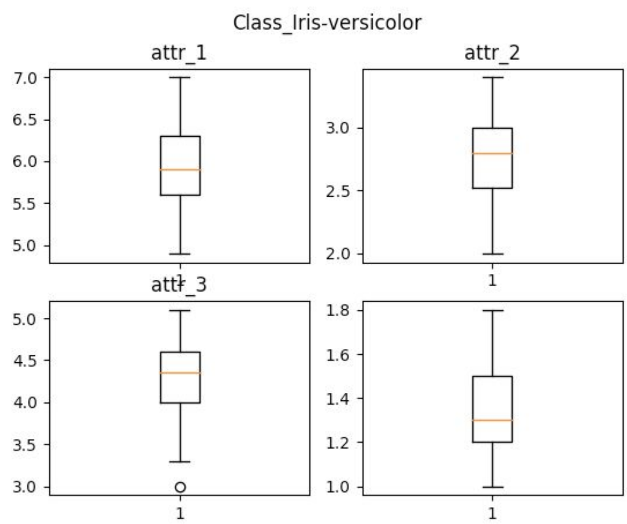
=> most symmetric, mostly bi-modal

(2)Date Set: Iris

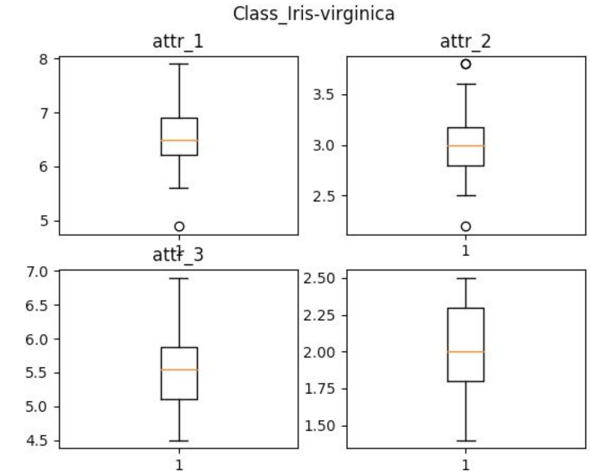
Class_1 (Iris-setosa) with 4 attribute

Class_2 (Iris-versicolor) with 4 attribute



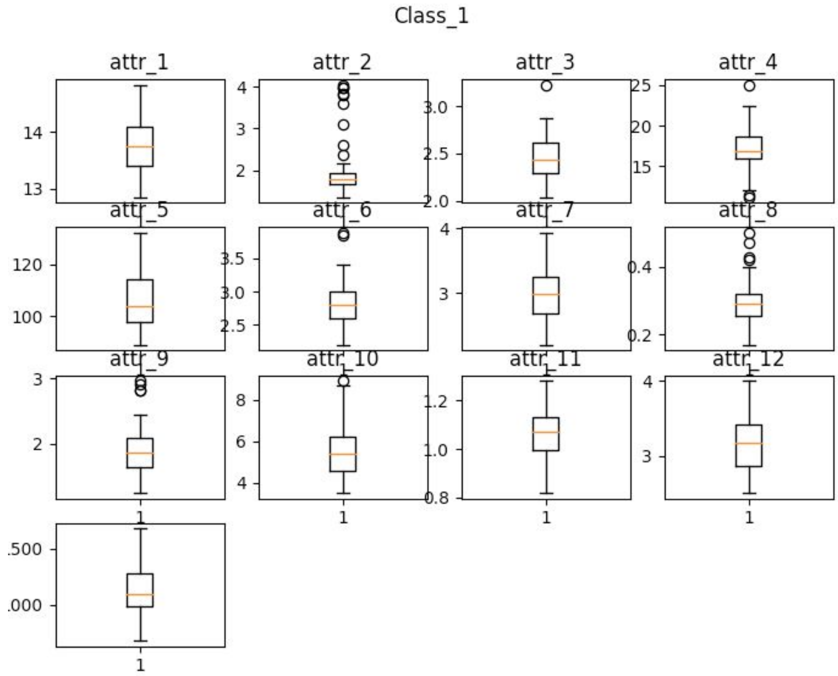


Class_3 (Iris-virginica) with 4 attribute

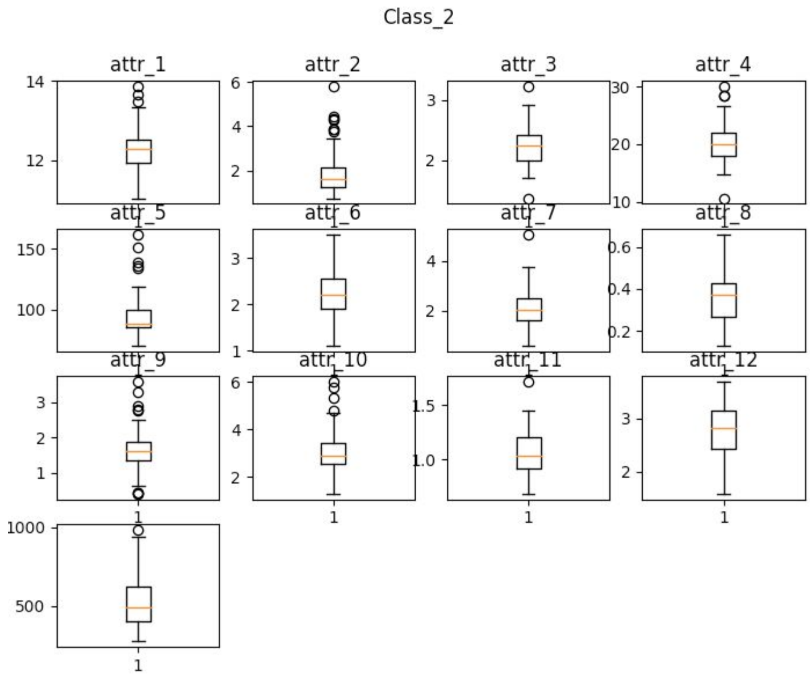


Date Set: Wine

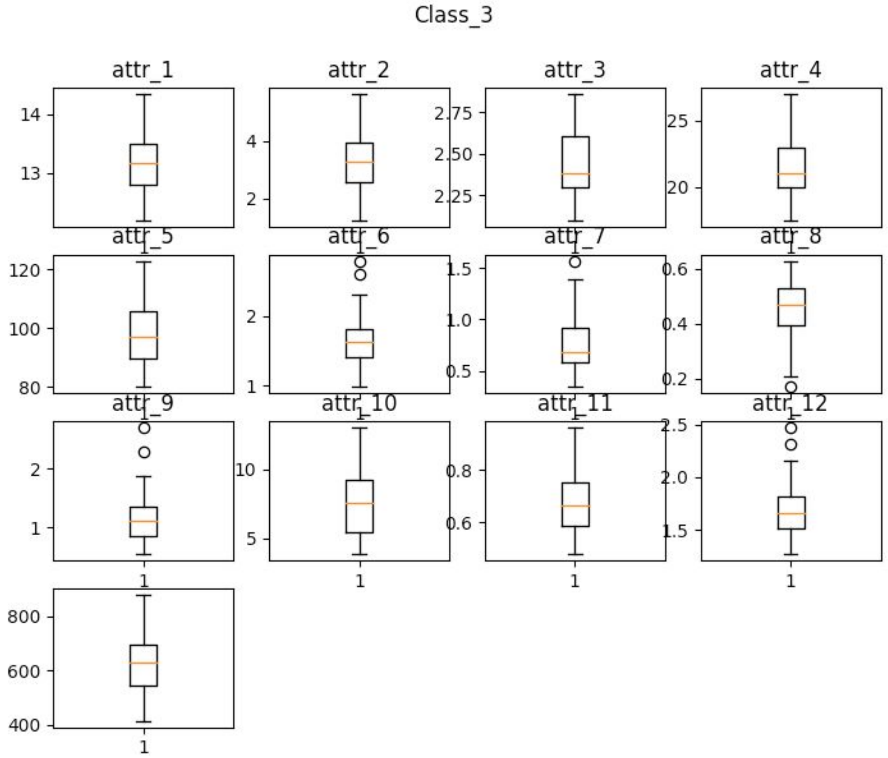
Class_1 with first 13 attribute



Class_2 with first 13 attribute



Class_3 with first 13 attribute



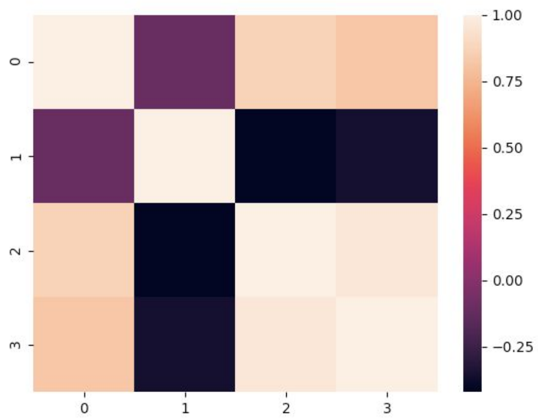
Question 2.

The detail value of pearson correlation is in Q2/pearson. Just attach the heatmap here.

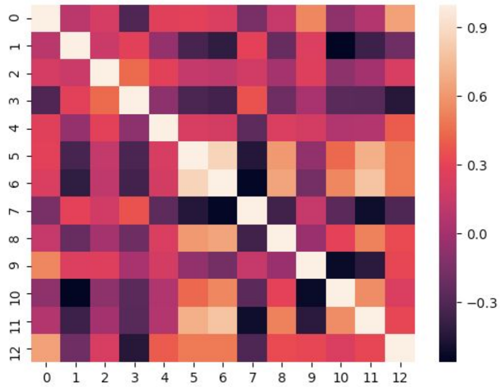
(1)

```
# for two given attribute list, calculate the pearson correlation
def correlation(listA, listB):
    avgA = mean(listA)
    avgB = mean(listB)
    sA = np.std(listA)
    sB = np.std(listB)
    if len(listA) != len(listB):
        print("len wrong in correlation()")
    tmp = 0
    for i in range(len(listA)):
        tmp += (listA[i]-avgA)*(listB[i]-avgB)
    return (tmp/len(listA))/(sA*sB)
```

Data set: Iris



Data set: Wine



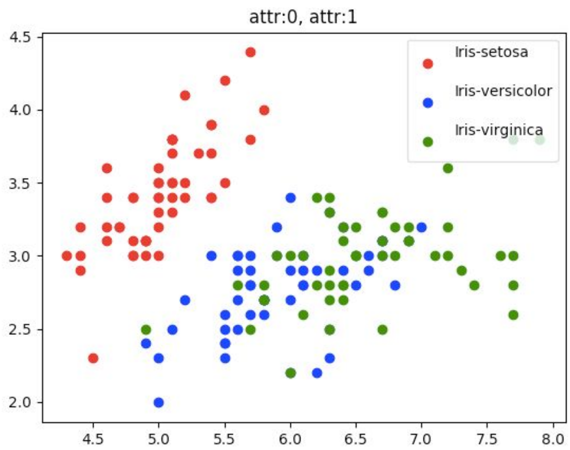
In order to fill the matrix, we have to use 6 correlation() for Iris data set and 3 correlation() for Wine data set (for first 3 attribute only). The number would be $\frac{n^2-n}{2}$, where n is number of attribute.

For Iris data set, attribute (2,3) and (2,4) is most irrelevant. While attribute(3,4) is most relevant.

(2)

The detail figure in given in Q2/scatter_plot/iris

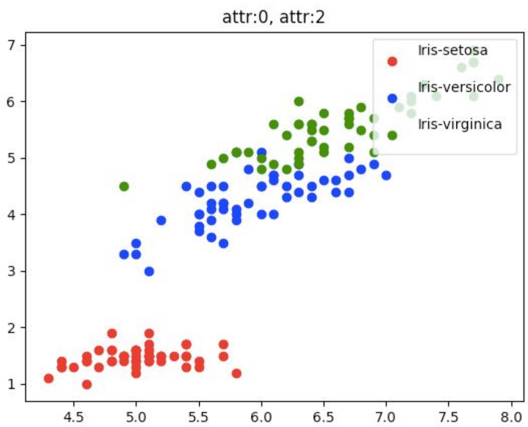
Attribute_1 and attribute_2



=> Obviously Discriminate for for Setosa

Not for other two classes

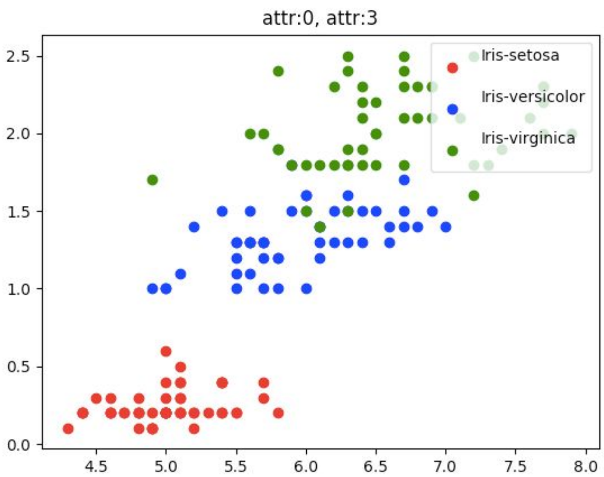
Attribute_1 and attribute_3



=> Discriminate for for Setosa

Little discriminate for other two classes

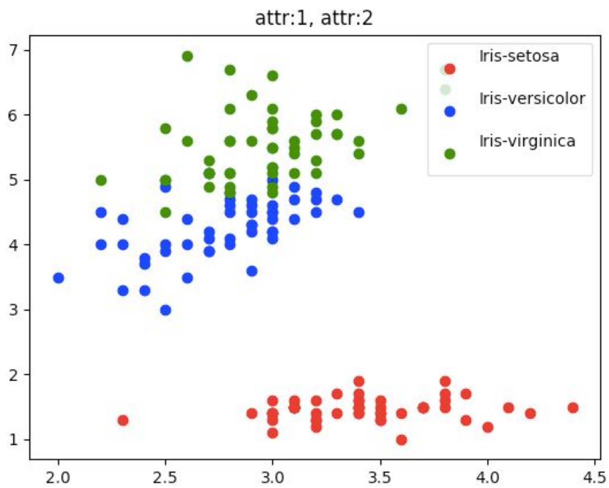
Attribute_1 and attribute_4



=> Discriminate for for Setosa

Little discriminate for Versicolor and Virginica

Attribute_2 and attribute_3

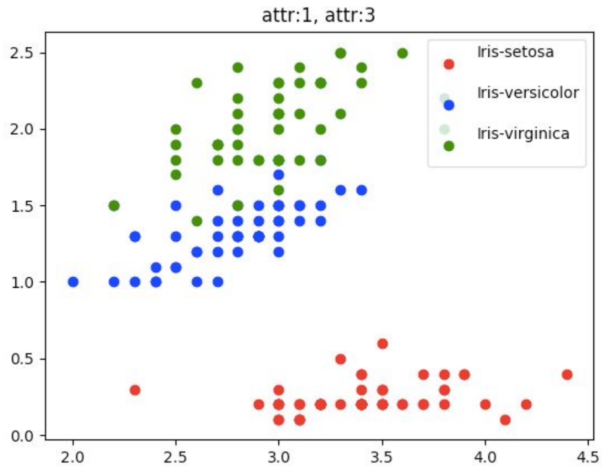


=> Discriminate for for Setosa

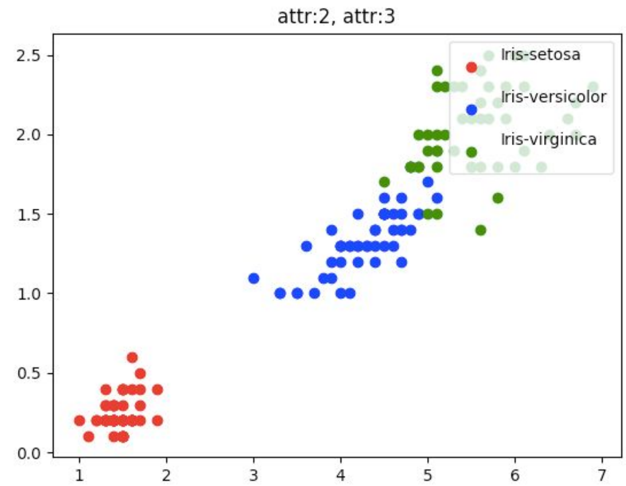
Little discriminate for other two classes

Attribute_2 and attribute_4

Attribute_3 and attribute__4



=> Discriminate for for Setosa
Little discriminate for Versicolor and Virginica



=> Discriminate for for Setosa
Little discriminate for Versicolor and Virginica

Of all feature sets, Setosa can be better discriminate while other two class are not. Scatter plot shows that attribute(1,3) (1,4) (2,3) is little better discriminate for Versicolor and Virginica. Compare with the result in part.1, the attribute (1,2) (2,3) (2,4) has a low correlation. It also shows in part.2 that the scatter plot with lower correlation is more messy. On the other hand, higher correlation like attribute (3,4) shows a linear-like distribution.

(3)

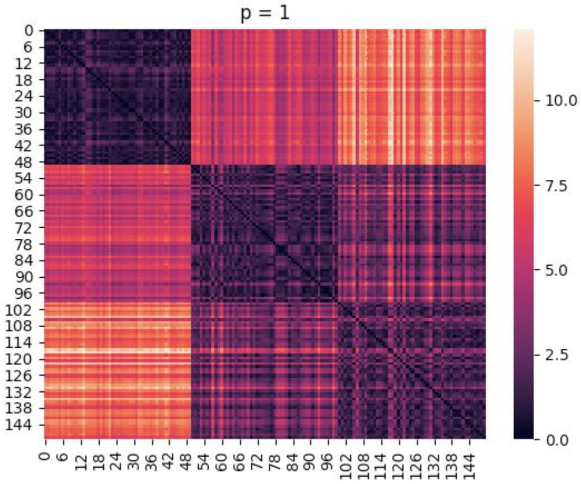
The detail figure is given in Q2/distance

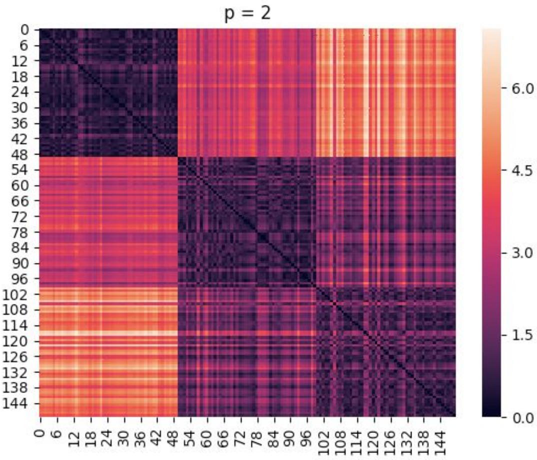
```
size_of_sttribute = 4
size_of_data = 150
list_of_data = [[""] for i in range(size_of_data)]
Class_list = ["Iris-setosa\n", "Iris-versicolor\n", "Iris-virginica\n"]

def distance(Class1, Class2, p):
    sum = 0
    for i in range(size_of_sttribute):
        sum += math.pow(abs(float(list_of_data[Class1][i])-float(list_of_data[Class2][i])),p)
    for i in range(p-1):
        #print("sqrt once")
        sum = math.sqrt(sum)
    return sum
```

The raw data from txt file is load in to list_of_data[], so the argument Class1 and Class is the index to access the entity. (Need to right shift one element when apply on Wine data set)

Iris data set:





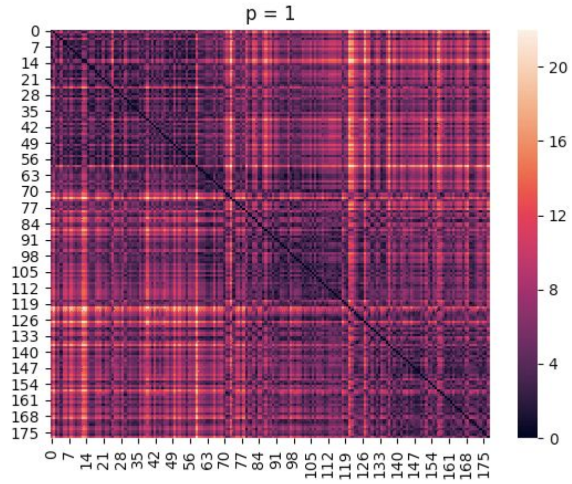
p=1

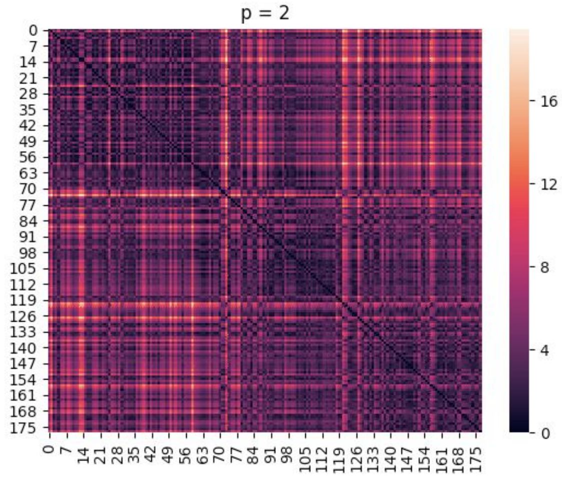
size= 150; Nearest node= 145

p=2

size= 150; Nearest node= 145

Wine data set:





p=1

size= 178; Nearest node= 139

p=2

size= 178; Nearest node= 139

In order to fill the matrix, we have to use 11175 distance() for Iris data set and 15753 distance() for Wine data set. The number would be $\frac{n^2-n}{2}$, where n is number of features.

We also use the compare the distance between entity and nearest entity. For Iris data set, 145/150 nearest entity is the same class. For Wine data set, 139/178 nearest entity is the same class. Both result remains the same at p=1 and p=2. It shows that the distance evolution can be used for clustering (the nearest entity has good possibility belong to the same cluster)