The Johns Hopkins University

EN.685.621.81.SP20 Algorithms for Data Science

Programming Assignment 1

Ivan Sheng

Initial Observations:

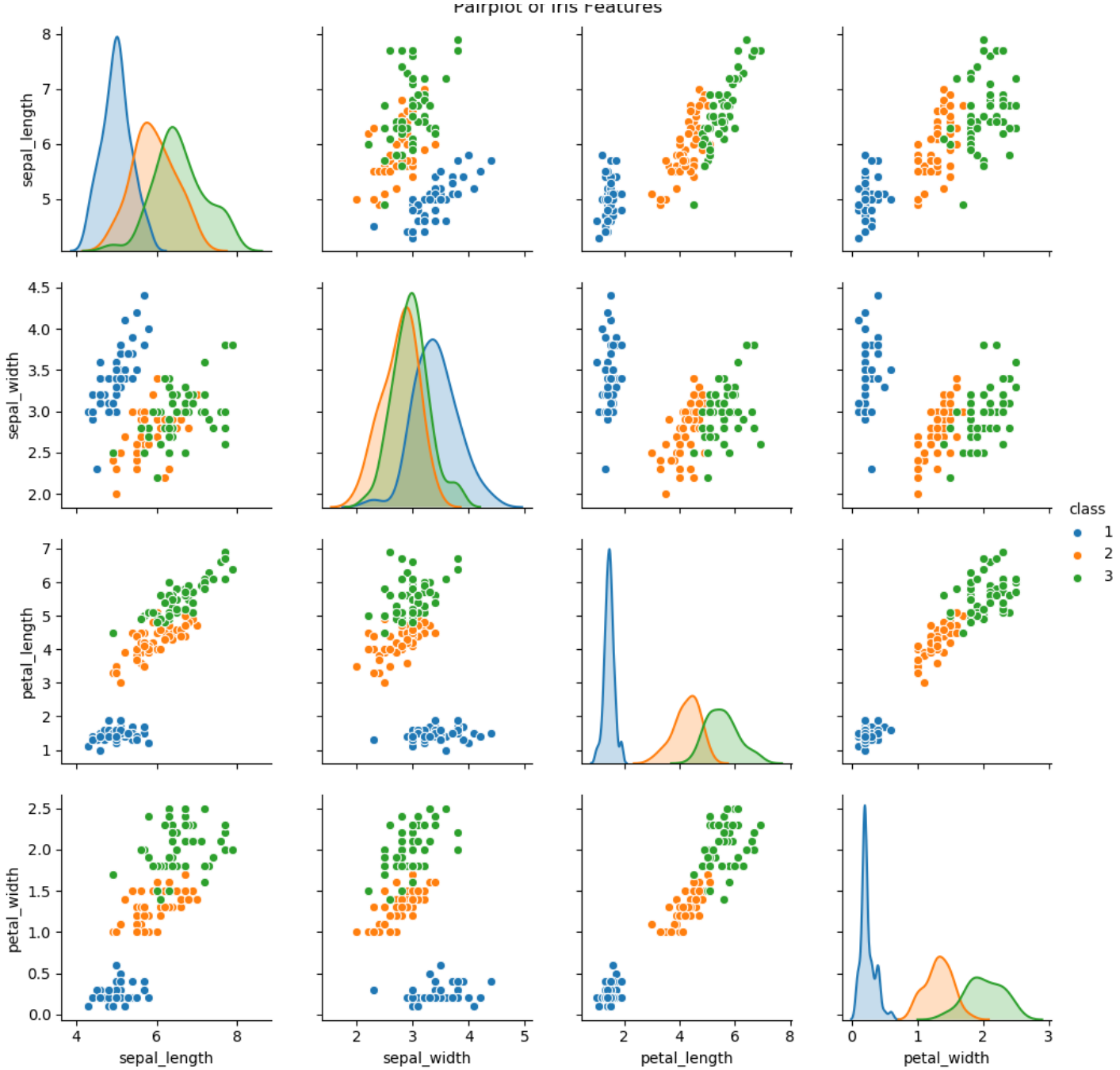

Figure 1: Pair-Scatter plots and Distribution Curves

From the initial observations, petal_width and petal_length look like the features with the best separability since their scatter plots don't seem to overlap clusters too much; there is minor overlap between Versicolor (class 2) and Virginica (class 3), but Sentosa (class 1) is clearly separated. Sepal_width looks practically impossible to separate classes; Versicolor and Virginica are practically on top of each other and Sentosa (class 1) heavily overlaps the two classes, as well.

Sorting:

Algorithm Pseudocode:

1. First sort the dataset by a specific feature in ascending order.
2. Cut the dataset into tertiles of size 50.
3. Create a confusion matrix and calculate the number of true positives

Efficiency:

The algorithm above takes advantage of the sort_values() method for dataframes, which defaults to quicksort. This is known to have a big-O of $O(n^2)$, which should be the highest order in the above sorting algorithm.

Results:

The idea behind sorting the features individually is that since there are three classes with equal sample sizes, if the flowers were perfectly separatable, then we would expect that each class would fall within their respective tertiles. Therefore, the metric that I will be using to determine separability will be percentage of tertile true positive classifications. Presented below are the confusion matrices from the sorting algorithm:

Sepal_length

Class	**1**	**2**	**3**
1	43	7	0
2	6	31	13
3	1	12	37

Tertile 1 True Positives for Class 1 : 0.86
Tertile 2 True Positives for Class 2 : 0.62
Tertile 3 True Positives for Class 3 : 0.74

Sepal_width

Class	**1**	**2**	**3**
1	1	13	36
2	30	16	4
3	19	21	10

Tertile 1 True Positives for Class 1 : 0.02
Tertile 2 True Positives for Class 2 : 0.32
Tertile 3 True Positives for Class 3 : 0.2

petal_length

Class	**1**	**2**	**3**
1	50	0	0
2	0	47	3
3	0	3	47

Tertile 1 True Positives for Class 1 : 1.0
Tertile 2 True Positives for Class 2 : 0.94
Tertile 3 True Positives for Class 3 : 0.94

petal_width

Class	**1**	**2**	**3**
1	50	0	0
2	0	46	4
3	0	4	46

Tertile 1 True Positives for Class 1 : 1.0
Tertile 2 True Positives for Class 2 : 0.92
Tertile 3 True Positives for Class 3 : 0.92

After running the algorithm on the iris dataset for each feature, it looks like the best features to separate the three plant types are petal_length and petal_width. Sentosa fell perfectly within its own tertile, however for both Versicolor and Virginica, there was a bit of confusion between the right distribution tail of Versicolor and the left distribution tail of Virginica. Sepal_length did an acceptable job; however, it was difficult to separate Versicolor from both Sentosa and Virginica. As for sepal_width, there was too much overlap between the three classes, it was almost indistinguishable.

We can further confirm these findings by plotting out the distributions:

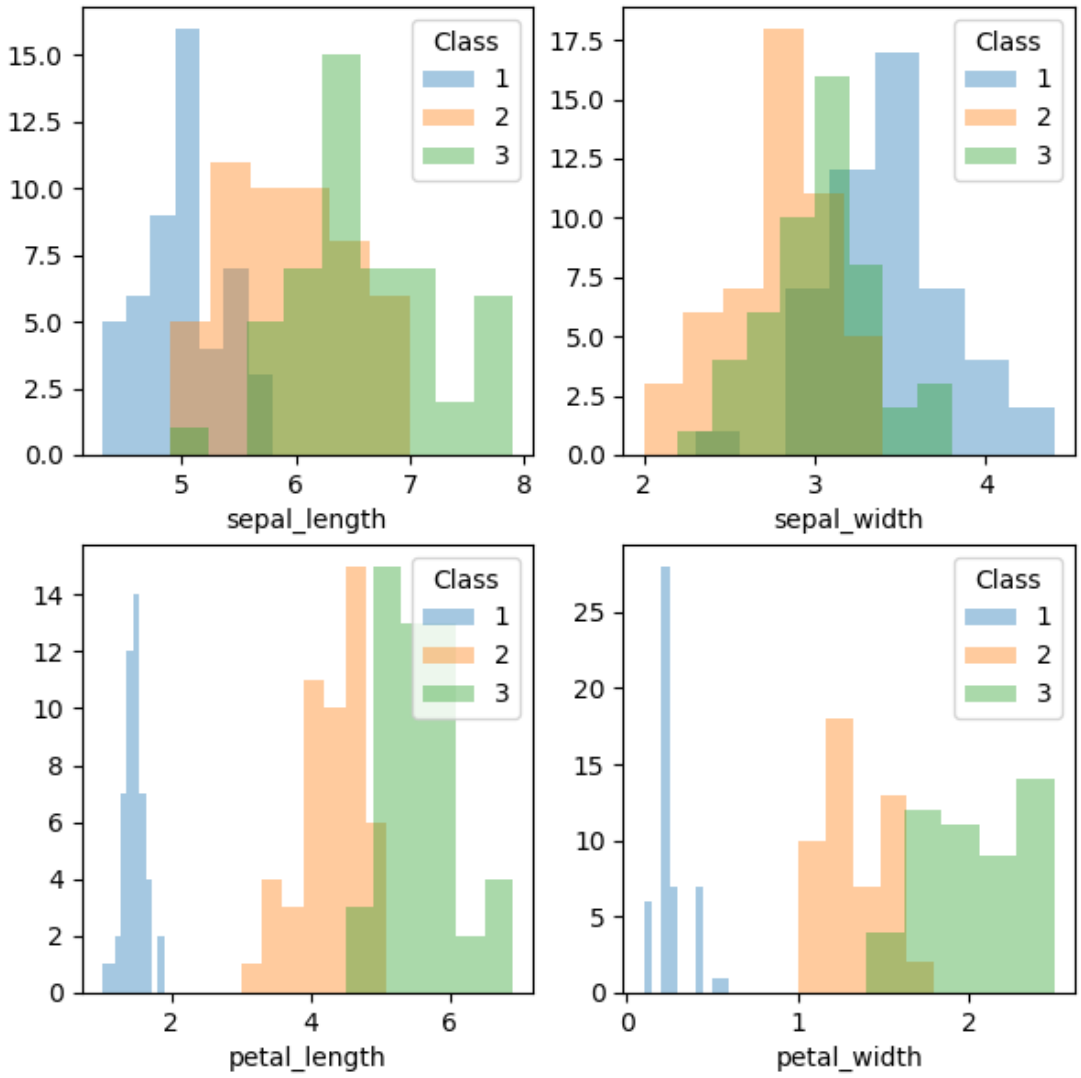

Figure 1: Original Data Distribution Histogram

As shown in the confusion matrices, there is too much overlap in sepal_width; petal_length and petal_width have good separation with minor tail overlap; and Versicolor's sepal_length is overlapping both Sentosa and Virginica.

Outlier Removal:

Algorithm Pseudocode:

1. Calculate the Mahalanbois Distance:
    a. Calculate the mean of each feature of the iris class
    b. Subtract the mean of the iris class from the actual iris class' data
    c. Compute the covariance matrix of the iris class and invert it
    d. Take the dot product of part 1b and the inverted covariance matrix from 1c
    e. Take the dot product of part 1d and the transposed matrix of 1b
    f. Extract the diagonal from the resulting matrix, which equates to the Mahalanbois Distance
2. Calculate Cv (critical value):
    a. Calculate the probability value: 1-alpha
    b. Set degrees of freedom (numerator) to the number of features
    c. Set degrees of freedom (denominator) to sample size – number of features – 1

d. Find the inverse F distribution probability for the parameters mentioned in parts 2a, 2b, 2c
e. Calculate the Cv using the values calculated above

3. Remove Outliers
    a. Create a Boolean column for the following conditional: mahalanbois distance > Cv
    b. Remove all indices where boolean index contains 0 (false)

Efficiency:

- Calculating the Mahalanbois Distance should be $O(N*n^2)$ because the largest growing term would come from calculating the covariance matrix where the mean of the data would need to be calculated for each N features iteratively since the resulting matrix is of size n^2.
- Calculating the critical value is O(1)
- Removing outliers would be O(n) since time would grow with the size of the boolean column parameter.
- Overall efficiency should be $O(n^2)$

Results:

When looking at the scatter plots of iris features in pairs, the following visual deductions about obvious outliers (highlighted in red) were made of each class:

	Sentosa	Versicolor	Virginica
petal_length vs petal_width	There's no clear trend between petal length and width, so it's difficult to distinguish any outliers. Points 22, 24, and 43 are suspectable.	There is a linear trend, with obvious outliers at 20 and 48. Potential outliers include 12, 17, etc.	There seems to be a linear trend, but it's hard to tell which points are outliers. Potential outliers include points 48, and 14
sepal_length vs sepal_width	There is an obvious linear trend between sepal length and width. Point 41 seems to be the obvious outlier.	The trend is linearly increasing, but all outliers aren't too noticeable. Obvious outliers are points 12, 18, 37, 22. Points 7, 9, 10, 43, and 48 seem too far from where the rest of the data clusters too.	While a trend isn't very noticeable, there are numerous outliers such as points 6, 17, 31
sepal_length vs petal_length	Although the trend isn't entirely clear, there are noticeable outliers in points 14, 22, 24, 44.	There is a linear trend. The obvious outliers are points 7, 9, 10, 43, and 48.	There is a very clear linear trend. Point 6 is a very obvious outlier.
sepal_length vs petal_width	Although the trend isn't entirely clear, there are noticeable outliers in where petal width is greater than 0.4cm	There is no clear trend, but based on how the points cluster, point 20 looks like an outlier.	There is no clear trend, but based on how the points cluster on the scatter plot, it looks like point 6, 19, 33, and 34 are outliers.
petal_length vs sepal_width	While the trend isn't obvious, it does appear to be linearly increasing with an obvious outlier at point 41, with potential outliers in 22, 14, 16, 33, 15, and 32.	There looks to be a direct relationship with numerous outliers such as 10, 37, 18, 22, 33, 48.	There is no clear trend though it appears to be a direct relationship. Obvious outliers include points 6, 8, 19, 22 and 18

To confirm the visual observations summarized in the table above, Wilk's outlier removal technique can be implemented to discover the outliers; this technique assesses the normality between samples. A threshold is determined by a critical value (Cv), which is based off the F distribution. We compare this critical value with each data point's Mahalanobis distance (D^2), which measures the distance from a point to a specific cluster of points – the closer it is, the more likely we can classify it with that cluster. Data points are classified as outliers if their Mahalanobis distance is greater than or equal to the critical value: $D_i^2 \geq C_V$ is true.

Further investigation into outlier removal can be made by looking at the scatter plots which plotted the data both before and after outlier removal. The following visuals shows the scatter plots by class:

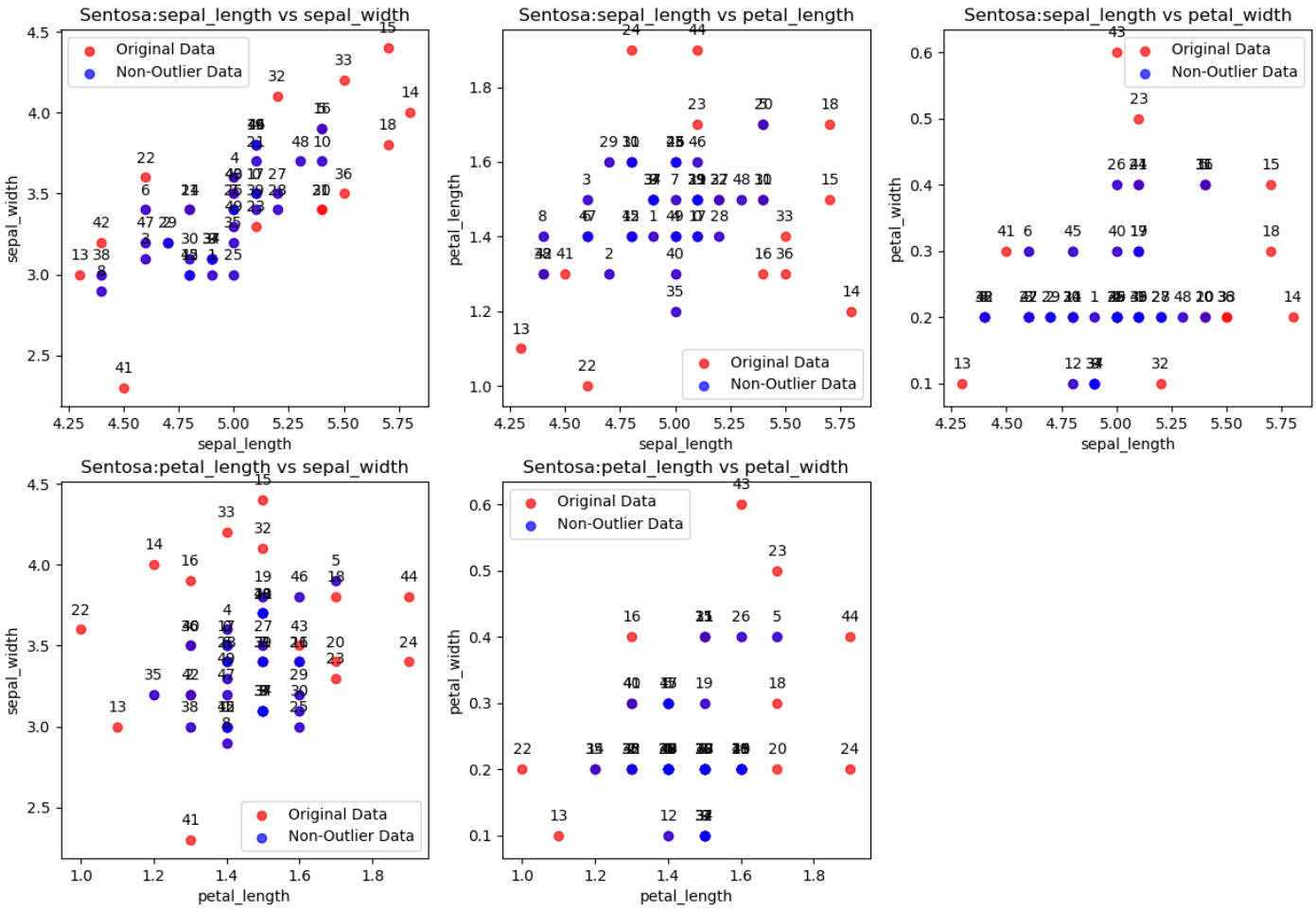

Figure 2: Sentosa Scatter Plots (Before and After Outlier Removal)

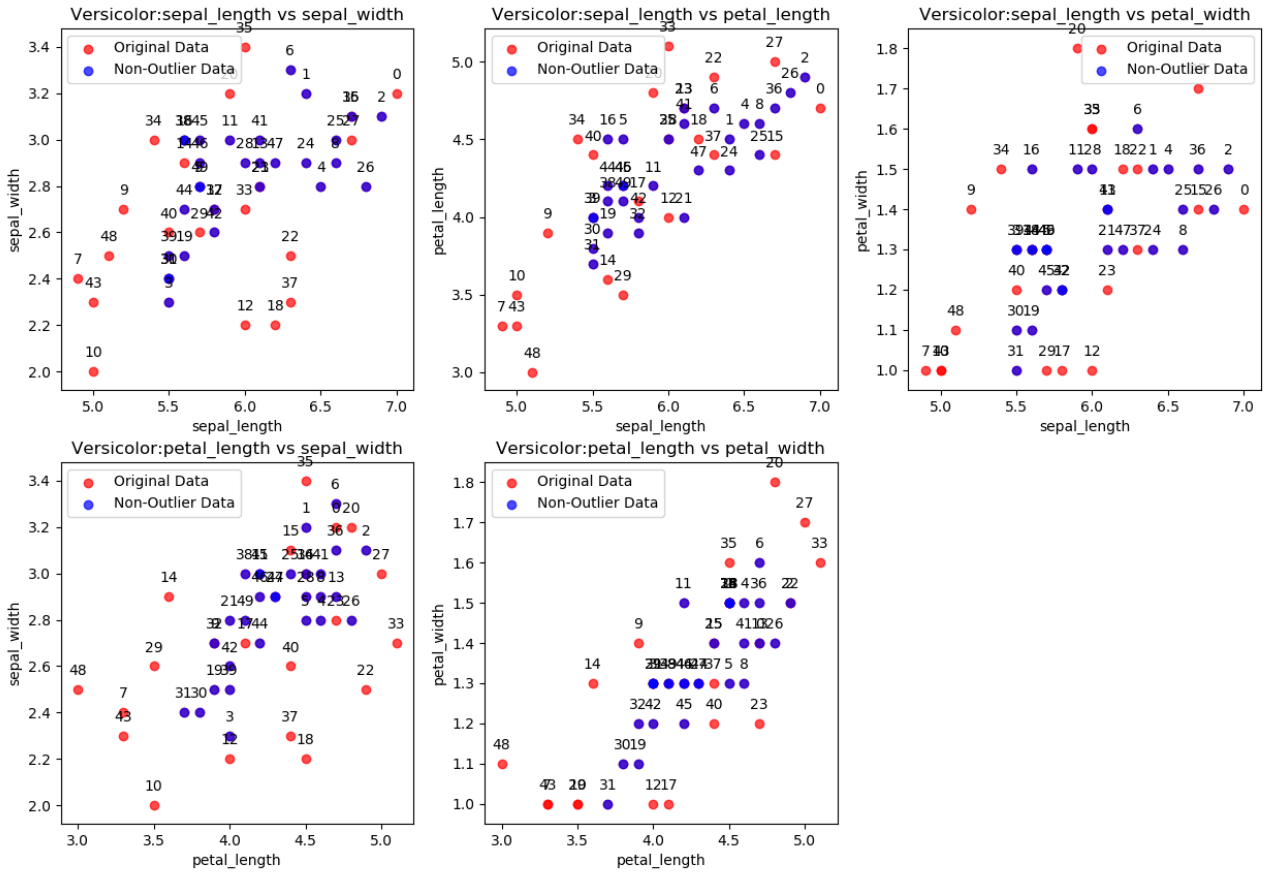

Figure 3: Versicolor Scatter Plots (Before and After Outlier Removal)

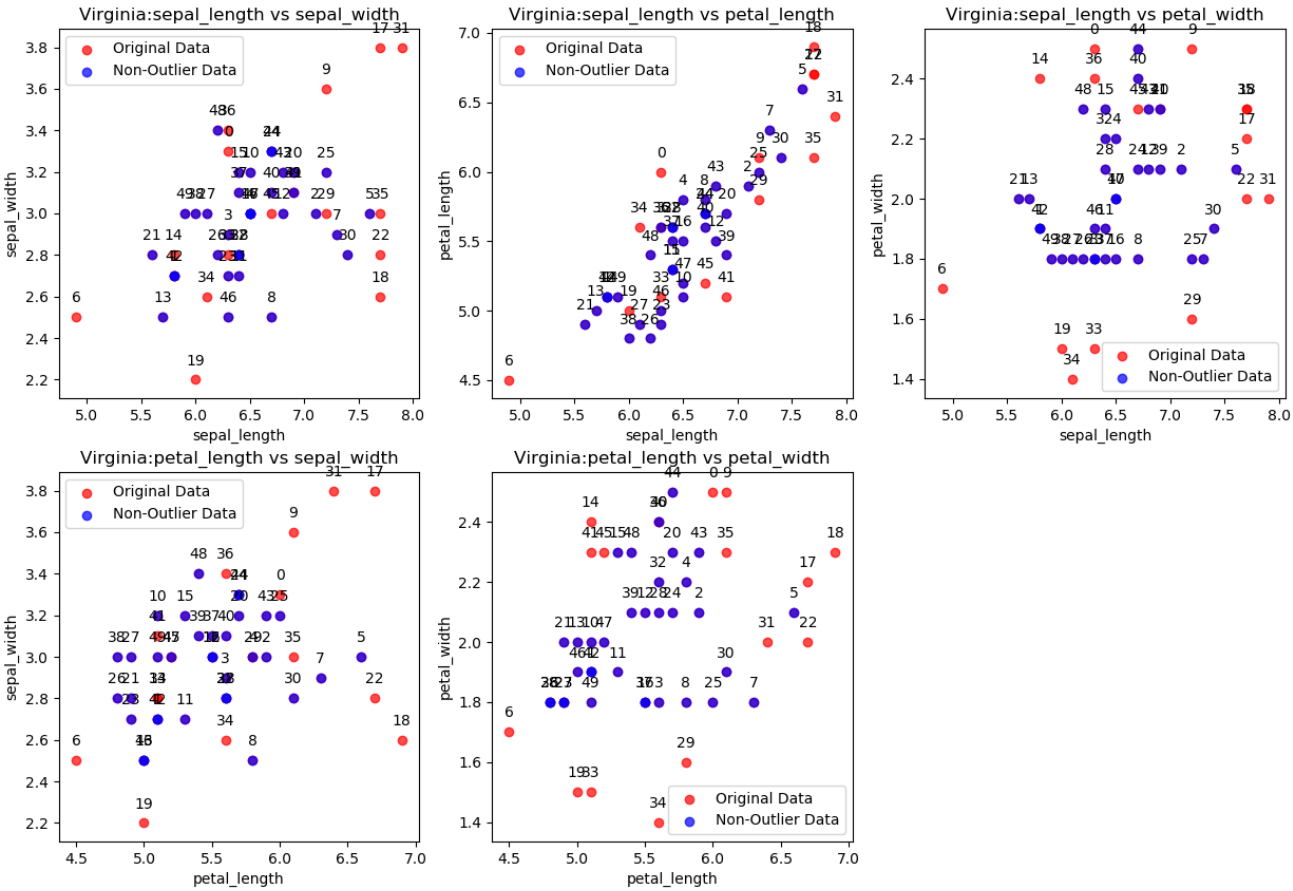

Figure 4: Virginica Scatter plots (Before and After Outlier Removal)

Feature Ranking:

Algorithm Pseudocode:

1. Normalize the data using the min-max method so all features have an equal influence on Fisher's discriminant ratio.
2. To calculate Fisher's discriminant ratio:
    a. Calculate the square of the differences of their mean for each feature
    b. Calculate the sum of their variances.
    c. Divide the two calculations
    d. For each class pairing (not including identical pairings), sum the results from part 2c
3. Sort the resulting array of FDR scores by descending order.

Efficiency:

In this algorithm, the highest order should be from within the nested for-loops. Since the for-loops have a known number of loops, they don't scale with n: meaning their efficiency would be constant. However, the mean calculation has a big-O of O(n) since the summing calculation of mean will scale with the size of the input array. This means that big-O due to the nest for loops would result in $O(C^2*n)$ where C is the constant number of times the nested for-loops iterate for, and since in big-O notation we take the highest order term, this algorithm would result in O(n) efficiency.

Results:

Fisher's Linear Discriminant Ratio, is a measure used to determine the separability of individual features by calculating the discrimination of sets of numbers (the ratio between the within-class and between-class scatter matrices). Based on the FDR scores run on the normalized iris data set, it looks like the best feature to separate classes would be petal_width.

Feature	FDR Score
petal_width	0.772
petal_length	0.396
sepal_length	0.292
sepal_width	0.113

Another metric we could use to complement the ranking method implemented in my programming assignment is to use the bhattacharyya distance. The bhattacharyya distance is made up of two terms: the left term measures the class separability by mean difference, while the right term measures the class separability by covariance difference.

Bhattacharyya Distance				
Feature Comparisons	**sepal_length**	**sepal_width**	**petal_length**	**petal_width**
sentosa-versi	0.026287	0.008888	0.011892	0.015148
sentosa-virginica	0.004104	0.002957	0.017909	0.008392
virginica-versi	0.020410	0.001766	0.011469	0.044021

Looking at the bhattacharrya distance results for the normalized data set, we come to the same final conclusion as seen in FDR, petal_width is the best feature to separate the three classes by, especially between Virginica and Versicolor, which have been the two most overlapping classes. Sepal_width is still the worst class to separate by, having the lowest scores in each feature comparison.

We can further verify these results by plotting the distributions of the dataset after outlier removal to observe separation between classes by feature. According to our feature rankings, petal_width should have the most obvious separation, while sepal_width is expected to still overlap quite a bit.

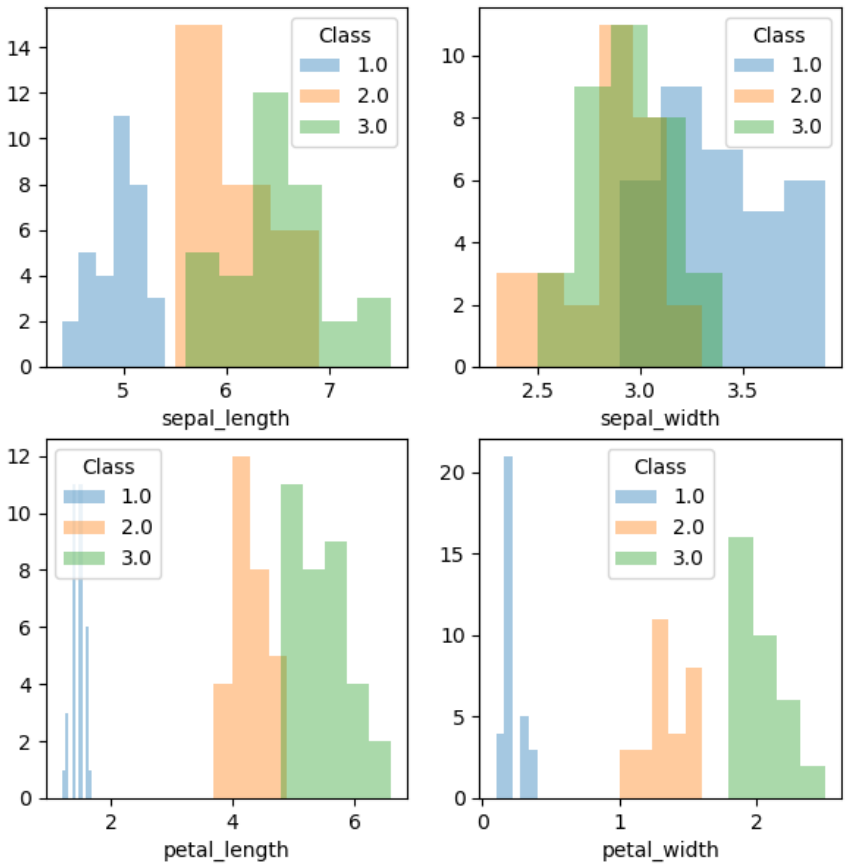

Figure 5: Data after Outlier Removal Distribution Histograms

When looking at the distribution plot of the iris data after removing the outliers, we can visually confirm that there is clear separation for petal_width between the 3 classes with no overlap, as observed by our feature ranking techniques. Moving on to petal_length, there is still a minor amount of overlap between Versicolor and Virginica, and it looks like Sentosa has too much separation within its own class. Sepal_length has clearer separation between Sentosa and the other two classes, but the overlap between Versicolor and Virginica is still too great to consider it a good separating feature, despite it coming close to petal_length's rank.

The one feature that couldn't separate any of the plant types was sepal_width. In fact, its distribution looks worse after outlier removal – Versicolor now looks like it skews left, which Virginica almost entirely overlaps; there is also no clear separability between Sentosa and the other two classes, as we have seen in petal_length, petal_width, and sepal_length. We can conclude that these three plant types' sepal widths are too similar in terms of mean and standard deviation.

Conclusion:

After running the iris dataset through multiple methods to identify separability amongst plant features, the results show that petal_width is the best feature to separate the three classes. When sorting the original iris dataset by petal_width, there was clear separation for Sentosa, and only four points overlapped between Versicolor and Virginica, according to the confusion matrices and distribution histograms. After outlier removal, the distribution histogram showed very clear distinctions between the three plant types for petal_width, although there is minor separation within the Sentosa class. Additionally, the Fisher's Linear Discriminant Ratio and Bhattacharyya Distance for petal_width was the highest amongst all iris features.

Initially, petal_length looked like the better feature to separate with – slightly out performing petal_width in the confusion matrix. However, after outlier removal, not only is there still minor overlap between Versicolor and Virginica, but there are also glaring gaps within the Sentosa class too – a problem that was only minor in petal_width's case. Meanwhile, sepal_length still maintained a good Sentosa distribution, but it had difficulty separating Versicolor and Virginica.

Sepal_width was never in contention. From the start, its class distributions overlapped too much – their mean and standard deviations were too similar.

I personally feel that programming assignment 1 was a good application of data processing and how it should flow – from your initial findings and visualizations, to outlier identification and removal, to normalization and feature ranking once the data has been processed. There are obviously scenarios where significantly greater number of features would impact our process since visualization would be too difficult, but the work done in programming assignment 1 helped to develop better intuition since visualization was really only necessary to verify our numerical observations. However, that's not to say that data visualization isn't important. As seen in the feature ranking section, there were differences between the ranking of the middle two features, petal_length and sepal_length, in our Fisher's linear discriminant ratio and bhattacharyya distance calculations. Looking at the distribution plots allowed for a visual interpretation of which feature had better separability.