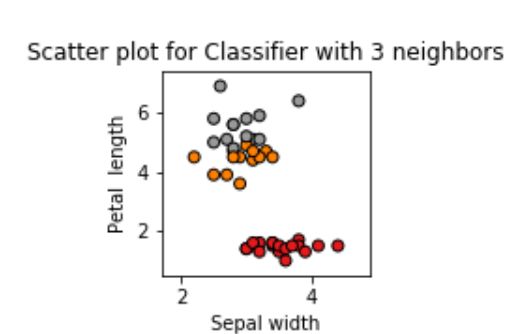
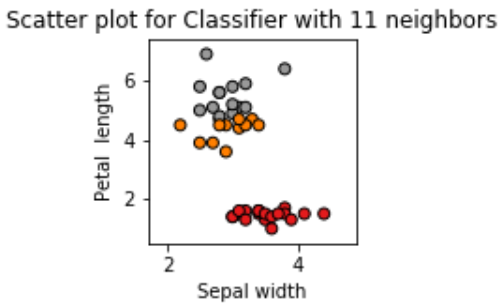
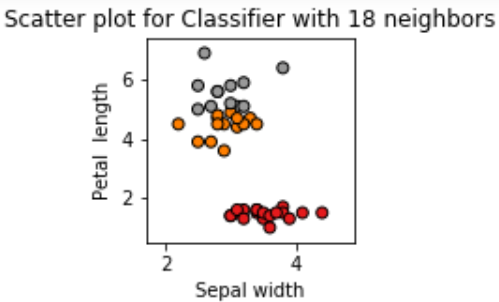
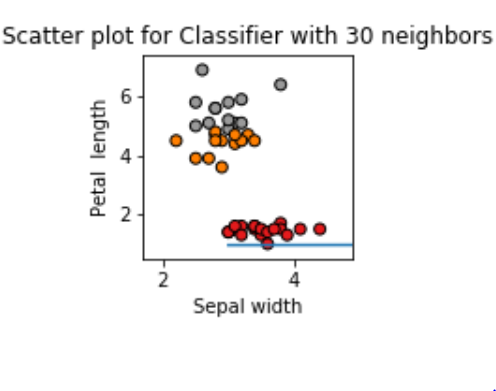
**Report KNN**

1. K -Nearest Neighbor Classifier is a supervised classifier. We used this classifier to infer the target class from a labeled training data. The training dataset is an input object (usually a vector) and its corresponding output value. This training dataset is analyzed to predict values of unknown datasets. The KNN addresses the problems of pattern recognition. The target label can be predicted by finding the nearest neighbor class and the closest class can be identified and measured using Euclidean distance. KNN is not suitable for large dimensional data but is suitable for smaller dimensional data like Iris Data Set. Choosing a small number for k might include noise. Using a larger k value reduces the overall noise but it might be expensive. The optimal k can be chosen by calculating n^1/2 where n is the number of values in the training dataset. Although, it is simple, it gets time consuming and expensive for large datasets.

EUCLIDEAN DISTANCE =   **SQRT ((X1 – X2)2+ (Y1 -Y2)2)**

It simply**calculates the distance of a new data point to all other training data points.**

1. The attributes I considered are sepal width and petal length. The reason why I chose these two attributes are so that I can consider two different features for better comparison. Having only sepal length and sepal width or petal length and petal width may not be an accurate comparison because we are not taking one of them into consideration. So, the comparison is devoid of a feature and it is wholly dependent on only one of the features and might not be completely accurate. I also implemented this practically. When I took the attributes as sepal width and sepal length, the accuracy rates were only around 77% but taking sepal width and petal length the accuracy rose to 97.77%. So, considering all the different features possible would give you a more accurate prediction value.
2. Observing the 2D visualization, the scatter plot shows similar results for all the different values of neighbors. The clustering has been done based on sepal width and petal length. Sentosa represented by 0 is displayed in red, Versicolor represented by 1 in yellow and Virginica represented by 2 in grey. Setosa (red) have sepal width greater than 3 and petal length lesser than 2. Petal length between 3 and 5 is Versicolor(yellow) and sepal width is lesser than 3.5. Petal length above 5 is Virginca(grey). For Neighbors k = 3,11,18 the values are almost similar for the test data.

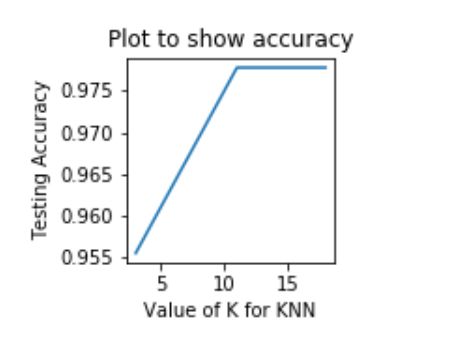
   

1. As the value of k (the number of neighbors) has been increased, the rate of accuracy increased. For k=3, the accuracy was 95.5%, for k =11, the accuracy was 97.7%, for k =18 the accuracy was 97.7% and for k = 30, the accuracy was 100%. This might not always be the case. The output also depends on the random state that you choose for selecting your training data and test data. Given random state = 7, we get relatively low accuracy for k =3,11,18 and 30 being 88%. To get higher accuracy we should try out different random state values as well as different k values.

|  |  |  |
| --- | --- | --- |
| COMPARISON | | |
| K VALUE | ACCURACY | CONFUSION MATRIX |
| K = 3 | 0.9555555555555556 | [[19 0 0]  [ 0 12 1]  [ 0 1 12]] |
| K = 11 | 0.9777777777777777 | [[19 0 0]  [ 0 12 1]  [ 0 0 13]] |
| K = 18 | 0.9777777777777777 | [[19 0 0]  [ 0 13 0]  [ 0 1 12]] |
| K = 30 | 1.0 | [[19 0 0]  [ 0 13 0]  [ 0 0 13]] |

For k = 30 there are no False Positives and Falls negatives. So, the data is errorless, and the precision and recall values are 1. For k =3,8,11 there are false positives and negatives, but false positives and negatives are very less compared to true values, so the precision and recall values are <1.0

**Accuracy for k = 3, 11, 18.**



References:

<https://t4tutorials.com/euclidean-distance-in-data-mining/>

<https://discuss.analyticsvidhya.com/t/how-to-choose-the-value-of-k-in-knn-algorithm/2606>

<https://dataaspirant.com/k-nearest-neighbor-classifier-intro/>

<https://towardsdatascience.com/knn-algorithm-what-when-why-how-41405c16c36f>