Classification I

Report

Machine learning and Data mining II

Nguyễn Mạnh Hưng - 22BI13183

Nguyễn Trọng Minh - 22BI13304

**TABLE OF CONTENT**

1. **K-nearest neighbor classification**
2. Perform K-NN
3. Normalization
4. Apply PCA
5. Improvement
6. **Support Vector Machine**
7. Analyze the dataset
8. Set up the SVM
9. SVM’s performance
10. Multi-class datasets
11. **K-nearest neighbor classification**
12. **Perform K-NN**

In this lab work, we chose 2 data sets both are binary classified. One contains data about 100 mushroom individuals and classified into edible or not.

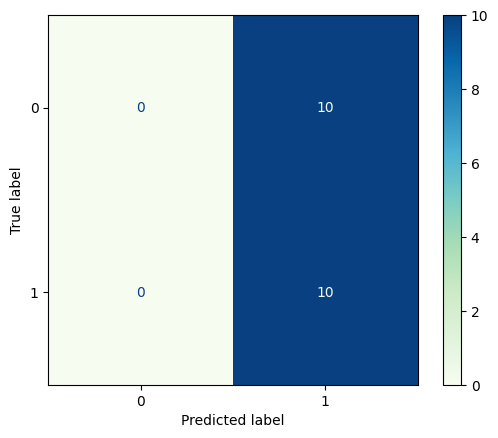
Data about mushroom

The other one is about 100 phones with their spec, divided into 4 price range.



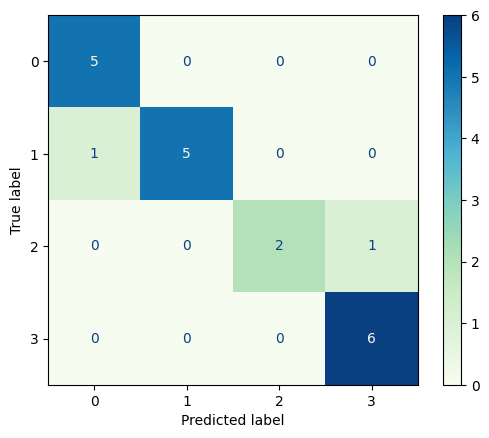
Data about phone price

After a quick look into the data set, we can start applying K-nearest neighbors classification algorithm. The algorithm is quite simple in the idea, using the distance between each point to classify a point to the same group of the nearest data point. And we got this confusion matrix for the mushroom data with a small testing data set of 20 data points separated from training data:



Mushroom dataset’s testing result

As we can see that the predict result only get to 50% accuracy, since we haven’t normalized the data yet. On the other hand, the phone data gave a 90% accuracy prediction.



Phone dataset’s confusion matrix

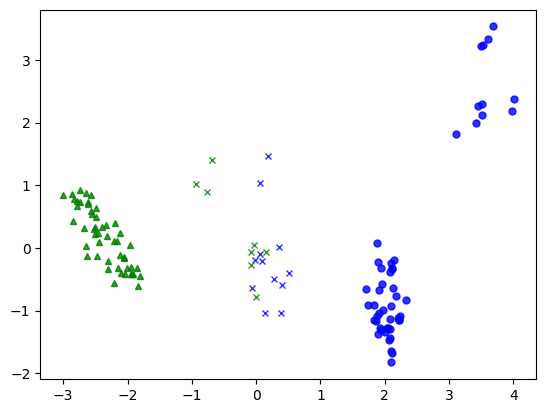
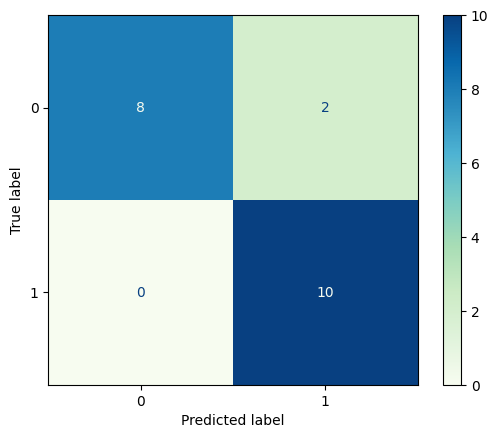
We might try to vary k from 5 up to 20 to see if there is any different. With the mushroom data, there is totally no different with all the k value in the given range. On the other hand, the prediction with phone data varies slightly with k value. At k=10, the accuracy is down a little bit with 1 more misclassify, but when k=15 and above, the accuracy is back to normal at 90%.

1. **Normalize Data**

Since K-NN measure the distance between each data points, it’s recommended to normalize the data before calculation. And we will do it in this section then recalculate K-NN performance after that.

To normalize dataset, we simply divide each column by its standard deviation and subtract by mean to center it for later PCA.

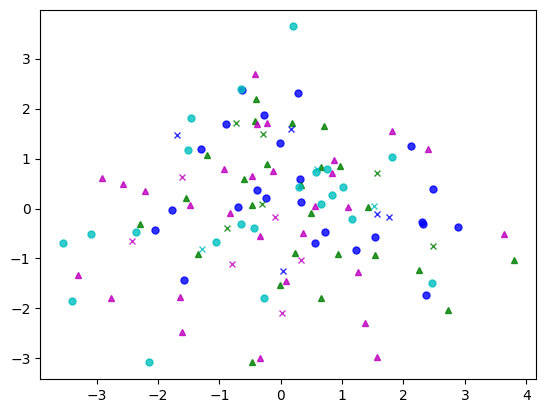
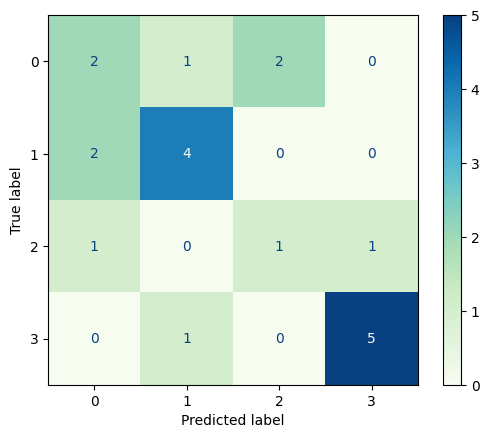
After that, we can perform K-NN on the dataset again and see the result.

Plotted result and confusion matrix of mushroom data

As we can see from the plotted result, x is the test data, there are some points that are closer to blue dot but classified into green category. This is because data lost in PCA. We also got a huge performance improvement, from 50% to 90% accuracy with test data.

In contrary, the phone data experienced a reduction in accuracy, from 90% to 60%. This is because the normalized data is scattered and became very hard to classify.

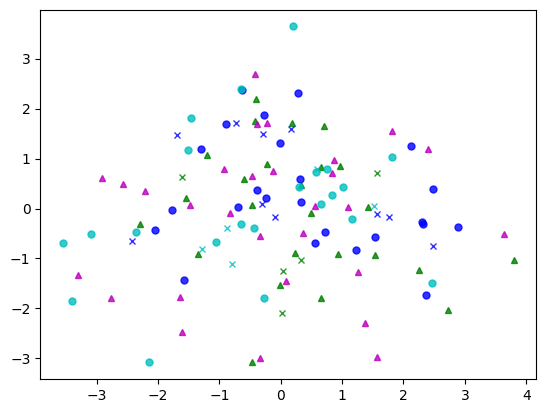
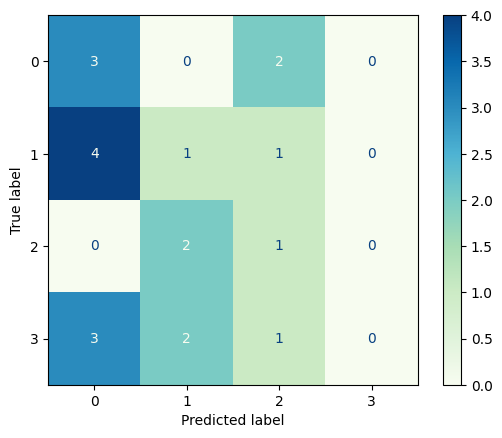
 

Plotted result and confusion matrix of phone data

1. **Apply PCA**

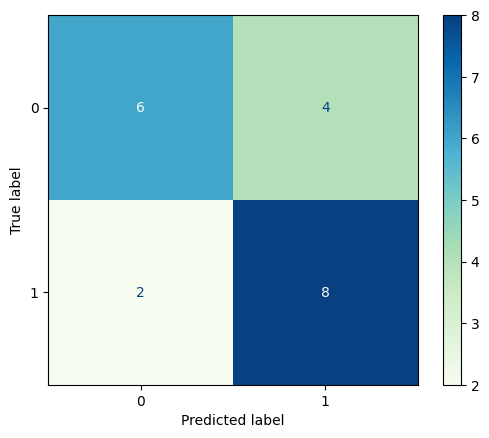
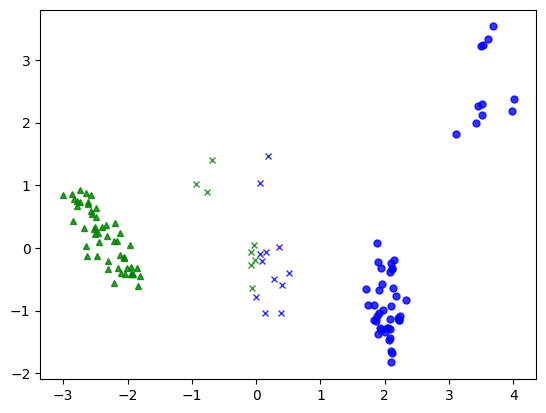
Now we try to apply K-NN on a 2D dataset transformed by PCA and see the result.

For the phone data, there is a steep fall in accuracy since it’s now only 25%.

Plotted result and confusion matrix of phone data after PCA

The mushroom dataset also experienced an accuracy drop, from 90% to 70%. This fall in accuracy is not as big as phone data because this data is linearly separable, not like the phone data. We can also notice that the plotted data doesn’t contain any outliners.



Plotted result and confusion matrix of mushroom data after PCA

1. **Improvement**

Since we have 100 data points in each data set, we can apply a 5-Fold cross validation with each test data set contains 20 points. We can train our model 4 times to get a better performance.