CSC-691 — Data Mining
Assignment 2

Esteban Murillo

Saturday 21st September, 2019

Analysis

After analyzing the yielded results a few times, it is more than clear that library functions are more efficient than anything self-implemented. And well, that is what libraries are for, right? Either way, there are a few takeaways from this.

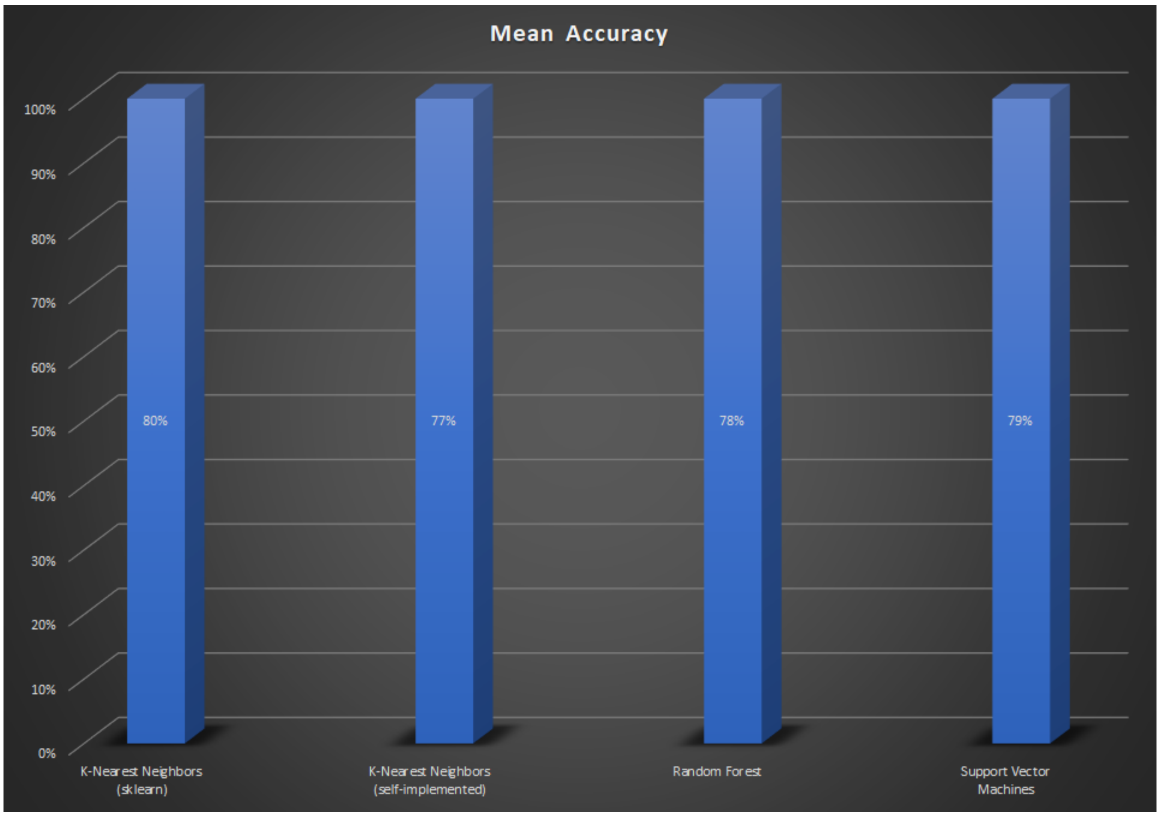


Figure 1: Yielded mean accuracy rates for each of the classifiers

First, and focusing ourselves on Figure 1 — related to accuracy — it is possible that due to some improvements under-the-hood the *sklearn kNN* algorithm is beating ours in terms of accuracy. For example, after going through *sklearn* documentation, we found out that they use a different distance algorithm and this could potentially be one reasons for the disparity. On the other hand, another factor that could be affecting the accuracy of the algorithms is the two different categories of images chosen — ocean and insects — which might not be the easiest categories to differentiate due to the fact that both types of images are very colorful. Clearly if this is indeed a factor, and we were to choose images that are easier to categorize, then, most likely, what we would see, is a boost in the accuracy for all of our algorithms rather than only for one of them. Finally and following the same line of ideas, we believe that it is very possible for the accuracy rate — for all algorithms — to go up if we were to extract more features from the images rather than just the *RGB* values.

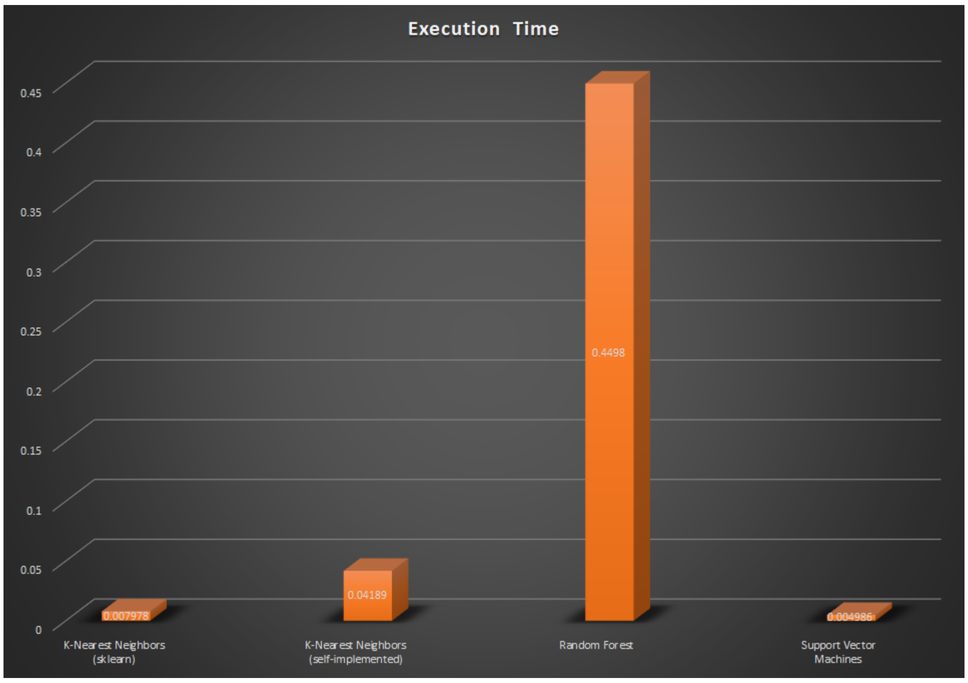
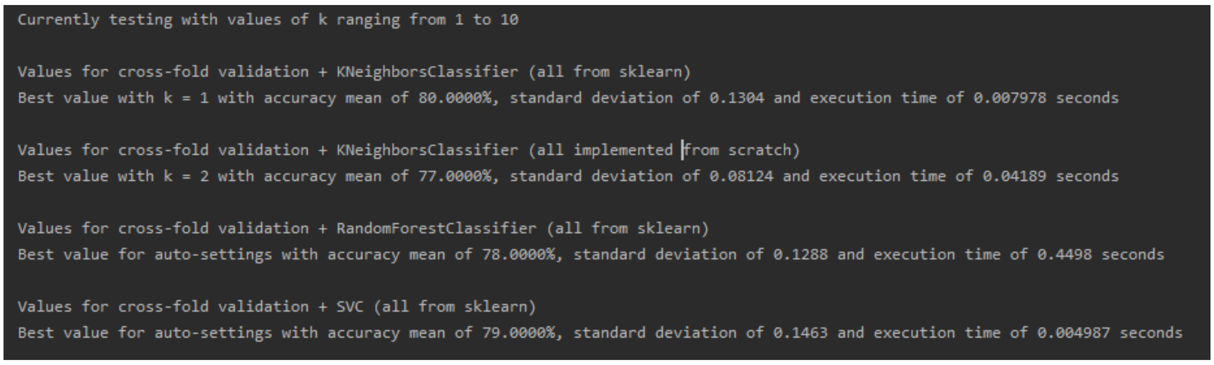


Figure 2: Yielded execution time rates for each of the classifiers

Now, moving on to the next metric — execution time — and focusing ourselves on Figure 2, there is not much that can be said about this, there is a clear winner here and there are no valid excuses to be given. Except for one, maybe. Of course that we paid close attention to efficiency when writing our code, yet we are not experts and a few loops had to be done in order to get the corresponding sets for the cross-fold validation. This could be affecting our performance time as well as the data structures that we used to keep track of all data. In addition to that the implementation of the distance algorithm could be affecting our performance as well. Obviously, there only one exception which is the *Random Forest Classifier* which takes a long time, but that is just the nature of the classifier itself.



```
Currently testing with values of k ranging from 1 to 10

Values for cross-fold validation + KNeighborsClassifier (all from sklearn)
Best value with k = 1 with accuracy mean of 80.0000%, standard deviation of 0.1304 and execution time of 0.007978 seconds

Values for cross-fold validation + KNeighborsClassifier (all implemented from scratch)
Best value with k = 2 with accuracy mean of 77.0000%, standard deviation of 0.08124 and execution time of 0.04189 seconds

Values for cross-fold validation + RandomForestClassifier (all from sklearn)
Best value for auto-settings with accuracy mean of 78.0000%, standard deviation of 0.1288 and execution time of 0.4498 seconds

Values for cross-fold validation + SVC (all from sklearn)
Best value for auto-settings with accuracy mean of 79.0000%, standard deviation of 0.1463 and execution time of 0.004987 seconds
```

Figure 3: Yielded results for our program

As seen on Figure 3, the values for k tend to stay on on the first few values. There were occasions in which this value grew and reached 4, but in general the k value always stays small. To conclude, it is important to emphasize that all of this heavily depends on the shuffling that is done before the data is passed to the different algorithms. There were several times in which we observed that the shuffled data favored the accuracy of both *kNN* algorithms and grew by $\approx 5\%$. Please note that all classifiers receive the same set of shuffled data so that conditions for all are the same.

Notes

- The set for the entirety of pictures used are within the submitted *zip* file. If you wish to change this, just save the desired image set on the same folder and change the values for *image_folder1* and *image_folder2* accordingly in the *global_variables.py* file
- If you wish to follow a closer look of how the whole flow of calculations are made, enable the ***debug*** variable located in the *global_variables.py* file
- Remember to run the code using the **version 3 of the *Python* interpreter**

References

[1] *Introduction to Data Mining.* Retrieved on September 6th.

[2] *Building a k-Nearest-Neighbors (k-NN) Model with Scikit-learn.* Retrieved on September 18th from https://towardsdatascience.com/building-a-k-nearest-neighbors-k-nn-model-with-scikit-learn-51209555453a