Assignment 2 Report by Anish Prasanna, Carlos Samaniego, Alex Brockman, Andrew Haisfield

Description

Based on our findings, the KNN we implemented from scratch performed worse, in regard to accuracy, than the scikit-learn KNN implementation we compared it to. While our KNN received an accuracy of 60.5%, the scikit-learn KNN scored a 70%. Our explanation for this occurrence is that the scikit-learn KNN implemented its KNN algorithm using weighted distances. In other words, this means that training points closer to the test data point would influence the classification proportionally more so than those further away. Conversely, we had implemented the algorithm by simply averaging the k closest neighbors.

Furthermore, the supplementary algorithms we compared against had mixed results regarding accuracy. The SVM (linear kernel) correctly classified 60%, while the Naive Bayes classifier classified 65% successfully (Figure 2). In comparison, our KNN algorithm was able to classify 60.5% correctly.

With respect to time, our algorithm took 2.641 seconds, while the scikit-learn KNN algorthm ran in 0.015 seconds. Additionally, the other scikit-learn algorithms were also quicker than our KNN implementation. The SVM (linear kernel) took 0.435 seconds, while the Naive Bayes ran in .005 seconds (Figure 1).

In summation, our KNN algorithm performed favorably in comparison to the scikit-learn SVM (linear kernel) in regard to accuracy by .5%. However, the scikit-learn Naive Bayes, and scikit-learn KNN both performed significantly better with comparative accuracy differences (our KNN) of 4.5% and 9.5%, respectively. Lastly, all of the scikit-learn algorithms were comparatively superior in their runtimes to our implementation, with differences of 2.63 seconds (scikit-learn KNN), 2.21 seconds (scikit-learn SVM) and 2.63 (scikit-learn Naive Bayes).

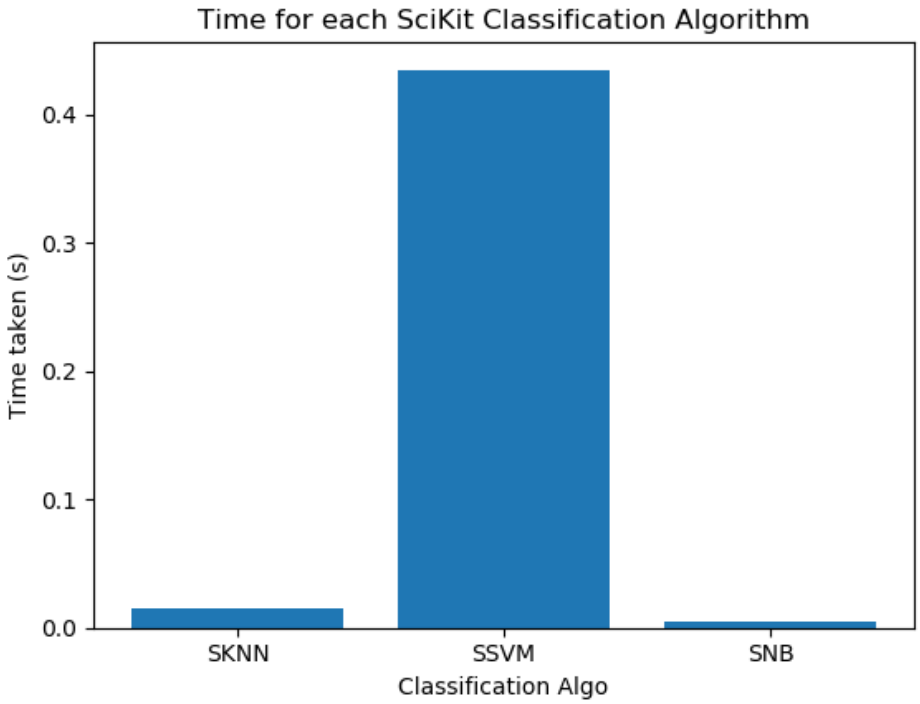

Figure 1: Run Times

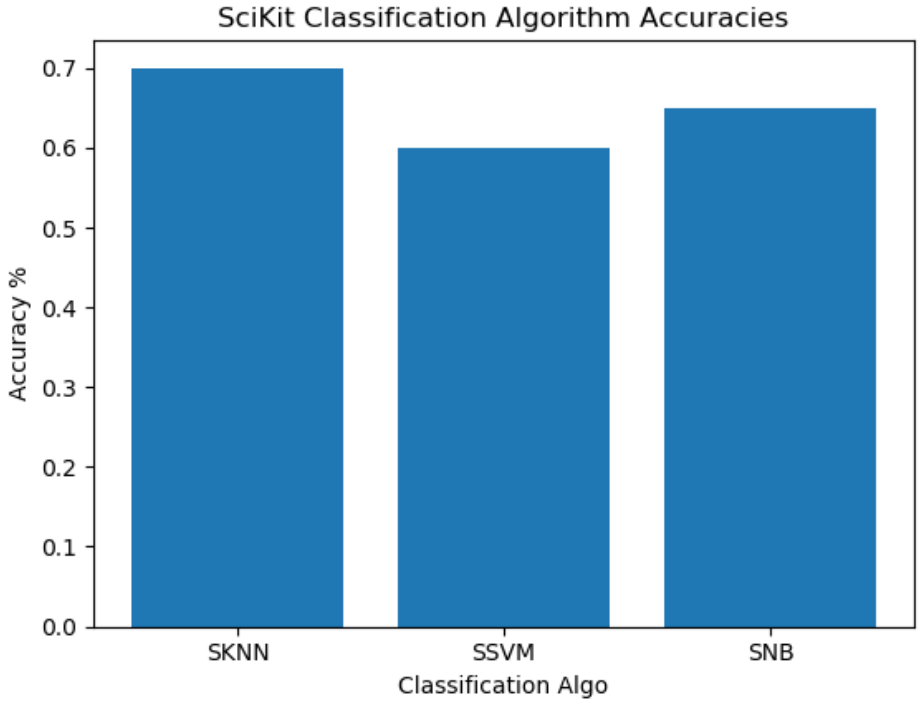

Figure 2: Scikit Accuracies.