MEEM 5990 Applied Machine Learning
Project 1

Total Points: 100

The objective of this project is to compare KNN and perceptron to classify bills as real or fake (bank notes). The dataset is provided as 'bill_authentication.txt'.

Data Set Information:

Data were extracted from images that were taken from genuine and forged banknote-like specimens. For digitization, an industrial camera usually used for print inspection was used. The final images have 400x 400 pixels. Due to the object lens and distance to the investigated object gray-scale pictures with a resolution of about 660 dpi were gained. Wavelet Transform tool were used to extract features from images.

Attribute Information:

Features:

- Variance of Wavelet Transformed image (continuous)
- Skewness of Wavelet Transformed image (continuous)
- Curtosis of Wavelet Transformed image (continuous)
- Entropy of image (continuous)

Label:

- Class (integer): 0 (real) or 1(fake)

Deliverables:

- A single code file which should run on a click when the 'bill_authentication.txt' file is placed in the working directory.
- A pdf file of all the results

Following tasks are to be performed before you submit your project.

1. Accuracy obtained at 80-20 train-test split of the dataset. Take random 20% samples for testing (same test samples should be used in case of KNN and perceptron) and report the accuracy of

KNN and perceptron. Note: Accuracy would depend on the randomly selected test data. Perform the task 5 times to get an error bar (mean and std. deviation).
Plot 1: One bar graph with an error bar. The plot will have 2 bars one for KNN and one for perceptron.

2. Repeat 1st task with 25,30,35,40,45,50,55,60,65,70 % test data.
   Plot 2: Plot accuracy vs test percentage for KNN and perceptron in the same plot. Legend is necessary for this plot.
3. Report perceptron training times, number of iterations for 90,85,80,75,70,65,60 % training data, and the corresponding weights.
   Plot 3: Training time should be reported in form of a graph (training time vs percentage of training data).
   Plot 4: Number of iterations should be reported in form of a graph (**number of iterations** vs percentage of training data). **Set some maximum number of iterations (say 100) and then plot at what number of iterations you get the highest accuracy. For this task, you will have to save weights and accuracy obtained at each iteration.**
   Table 1: **Best** weights should be reported in form of a table. Report means and standard deviation for each weight after the table. **Best weights are the weights corresponding to the highest accuracy.**
   Plot 5: Plot **highest** accuracy and training times in the same plot for 95,90,85,80,75,70,65,60 % training data. You will have to use different scales.
   Note: Training time and number of iterations would depend on the randomly selected test data. Train the perceptron 5 times at each value of % train data and add error bars (mean and std. deviation) to plot3 and plot4.
4. In this task we are going to find out the least significant feature. Use KNN on the data with 3 features at a time. You would be leaving out 1 feature at a time.
   Plot 6: Plot accuracy vs %test data by varying test data from 20%-60% in interval of 5. You should have 5 accuracy lines (All 4 features and 4 combinations of other 3 features). Legend is necessary for this plot. Looking at the accuracy values comment which feature do you think is the least significant.

Your report should consist of the following.
Task 1: Comment which algorithm works better for this dataset based on this task.
Task 2: One plot. Comment on the relation between accuracy and test percentage.
Task 3: 3 plots and 1 table. Comment on the optimum %training data by observing plot 5.
Task 4: One plot. Report which feature you think is the least significant.

You code should produce all 6 plots in order without requiring any modifications in the code. Add a random number generator seed before you code. Shuffle the dataset after you load it before proceeding to any of the given tasks.
To perform the given tasks, it is recommended that you write a function for KNN and Perceptron.

Point distribution:
Task 1: 15 points
Task 2: 15 points

Task 3: 30 points
Task4: 30 points
Readability of the code: 10 points
Check this blog to know about readability of the code
https://blog.pragmaticengineer.com/readable-code/

Acknowledgements: The data was obtained from UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.