Rohan Tandon
CS 589
02/14/17

Report HW1: Classification

1. The classifiers I have used are:
    a. K Nearest Neighbors: This classifier is a supervised learning algorithm which predicts labels based on the neighboring values of the input. The classifier classifies each new instance x using a majority vote over its set of 'k' nearest neighbors $N_k(x)$ computed using any distance function d: R^D X $R^D \rightarrow R$

    Mathematical Expression: $f_{KNN}(x) = \arg\max_{y \in Y} \sum_{I \in Nk(x)} I[yi = y]$

    The hyperparameter that is to be set is 'k' i.e. the number of neighbors to consider.

    This classifier was selected to find out whether the features increase/decrease directly proportional to the distance.

    b. Decision Trees: It is a parametric supervised learning algorithm which classifies data cases based on particular governing features into a binary tree structure.

    Mathematical Expression: $f_{DT}(x) = y$, $y \in$ Set of Labels L & y satisfies features till depth 'd'.

    The hyperparameter is the depth of the decision tree. This decides the fitting as an increase in depth means that an input will have to filtered more number of times based on the feature at each tree level.

    The classifier was selected because it will give us a clear discriminating feature of the dataset and we can know how each feature divides the data.

    c. Multilevel Perceptron: It is an artificial neural network which is based on the feedforward artificial neural network. It consists of 3 layers: input layer which is the dataset, a hidden layer where inputs are mapped to the output layer by performing a logistic function on the inputs and an output layer consisting of the labels.

    Mathematical Expression: If the node's bias is b and weight is w then the equation is:

    $$Y(x) = f_k(\ldots f_2(w_2^T f_1(w_1^T x + b_1) + b_2) + \ldots b_{k)}$$

    Hyperparameter to be selected is the number of nodes in the hidden layer.
    Note: The above equation has been used from: https://spark.apache.org/docs/1.5.0/ml-ann.html

    This classifier was selected because it is good to learn non-linear models.

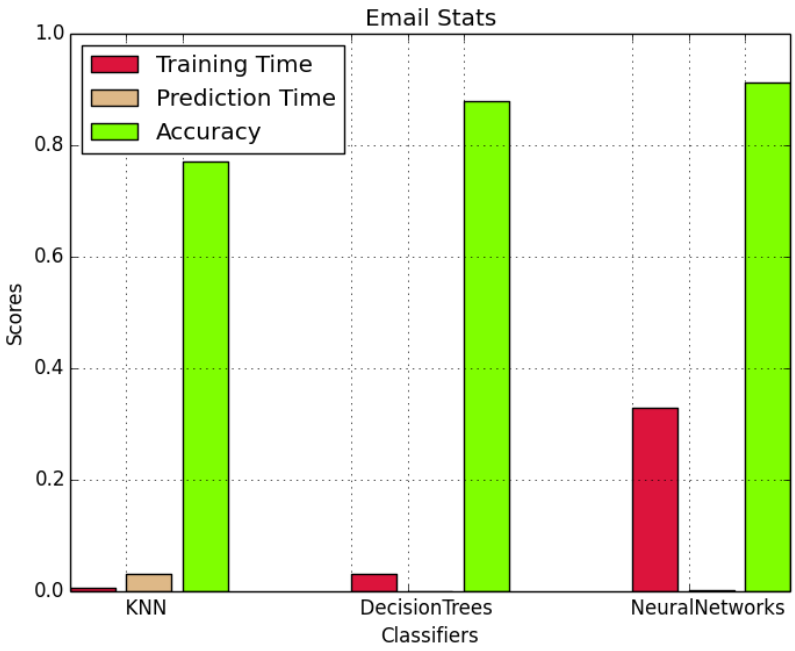
2. KNN is not mathematically related to Decision Trees or MultiLevelPerceptron. The decision boundary geometry depends on the number of neighbors selected, so greater the 'k', there will higher bias and low variance. KNN is easy to implement, and generalizes well when large amount

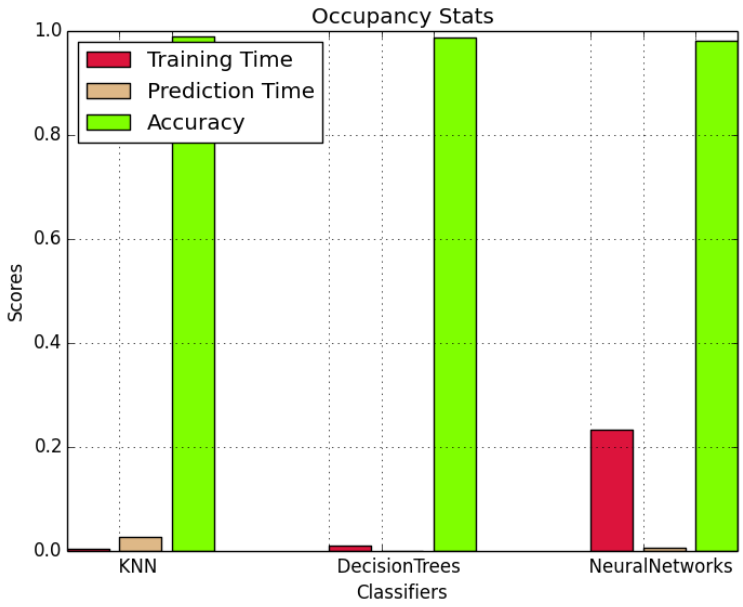
of training data is present. The disadvantage of KNN is that it requires lots of storage because it has to store the training data.

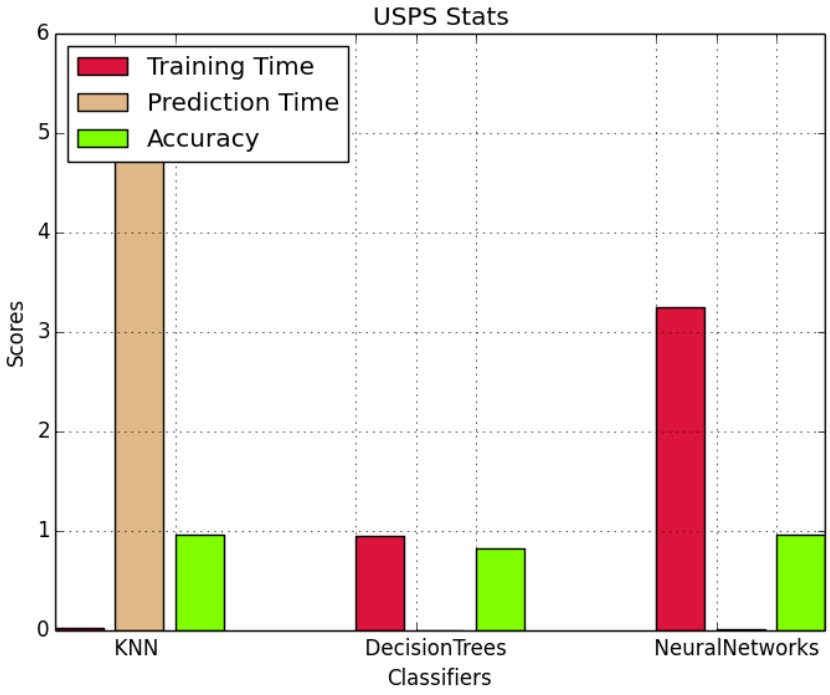
Decision trees are easy to interpret. Regarding the decision boundary geometry, deep trees have low bias and high variance while shallow trees have high bias and low variance.

MLP neural network generally has high variance and high bias for a less number of nodes and low bias/low variance for lots of nodes. It is tough to understand and implement.

3. A)







NOTE: Since the training and prediction timings are extremely small in the Email and Occupancy data sets, they are not visible in the graph. In the USPS graph, the knn prediction time takes a few seconds and hence has such a great height.

B) The Multilevel Perceptron classifier works best. The decision tree classifier works the fastest. The decision tree classifier has the best speed-accuracy trade off.

4. A) Pseudo code for method:
   Shuffle the train data and store the seed value
   For each hyperparameter divide the training data into training set L and test set T
   Partition L into k folds
   Use k-1 sets for training and 1 set for testing
   Calculate accuracy and store
   Repeat K times
   Finally test the classifier against original test set T.

   I have tried to implement cross-validation in the pseudo code. The data is portioned randomly using a seed value and that seed value is stored to save the according labels. The hyperparameter which generalizes the best is then finally selected.

   B) This approach was selected because it shuffles the data randomly, this ensures that the model is exposed to various inputs.

   C) In code method is hyperParameterSelection.

5. A)
   B)
   C) Yes this improves the accuracy.

6. I used Bagging to improve the accuracy rate.

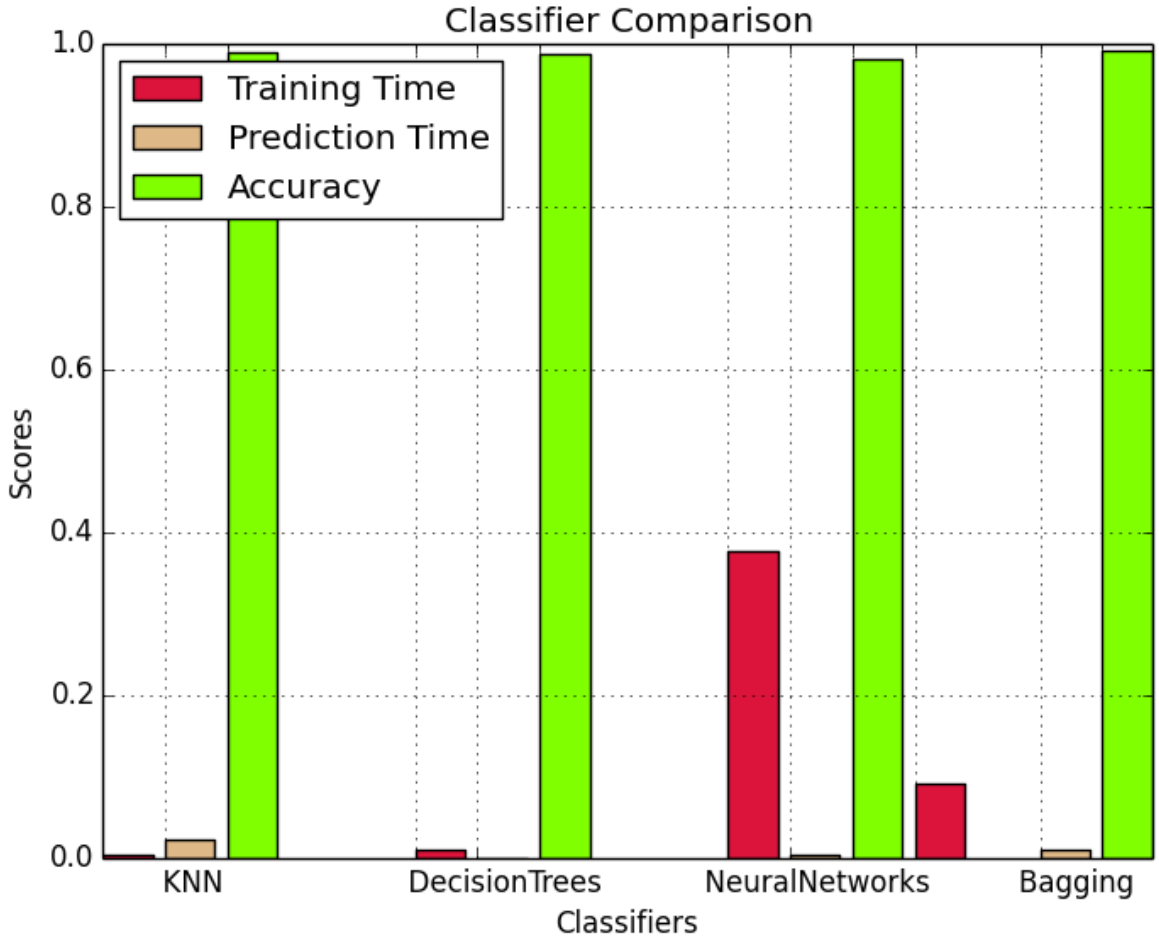
   Bagging: This an ensemble method. This classifier split the training dataset into different chunks, applies a classifier on each chunk and gives the average of all the classifiers as the output.

   Mathematical Expression: $f_{Bagging} = y$, $y = avg(classifier1, classifier2,\ldots)$

   The hyperparameter is the number of classifiers considered. Example: If the hyperparameter is 10 , then for each chunk 10 decision trees will be selected and applied and the output will be the average of all the 10 decision trees.

   This classifier was selected to reduce the variance of the outcomes.

   These are the stats of the bagging classifier when applied on the occupancy dataset:



This shows that bagging is faster than the other classifiers in terms of time taken for fitting and predicting. Also, the accuracy is slightly better for the occupational data.

Bagging.csv is the file uploaded on Kaggle, whose accuracy is 99.158 and is also my top score for that dataset. This shows that using ensemble classifier improves the accuracy.,