Homework 1

Apurva Sharma
GTID: 902490301
asharma70@gatech.edu

1.

Given a vector of inputs

$$X^T = (X_1, X_2, \ldots, X_p),$$

output Y can be predicted as :-

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{D} X_j \hat{\beta}_j \quad -(1)$$

For sake of convenience, we can include constant 1 in X & rewrite the linear model as-

$$\hat{Y} = X^T \hat{\beta} \quad -(2)$$

To fit a linear model to a set of training data we pick β that minimizes the residual sum of squares (RSS).

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - x_i^T \beta)^2 \quad -(3)$$

which can be written as: -

$$RSS(\beta) = (y - X\beta)^T (y - X\beta)$$

β that minimizes RSS can be founds as:-

$$\frac{\delta RSS(\beta)}{\delta \beta} = 0 \Rightarrow -2X^T(y - X\beta) = 0$$

$$\Rightarrow X^T y = X^T X \beta$$

$$\Rightarrow \beta = (X^T X)^{-1} X^T y \quad \text{if } X^T X \text{ is non singular.}$$

2,3 Are included in code apurva_knn.py

Which can be run as :

python <training file> <k> <N folds>

example,
python apurva_knn.py /home/apurva/Desktop/sem3/DM/Assignment1/data/SAheart.data 3 10

Gives output :

Errors Accross Folds---->
[17, 14, 13, 16, 23, 19, 11, 15, 15, 9]
Mean Error Accross N Folds----> 15.2
Output generated in >>> /home/apurva/Desktop/sem3/DM/Assignment1/code /op.csv
Number of Wrong Predictions on when Trained and Tested on Full Dataset 90
% wrong 19.4805194805

And generates op.csv which compares Actual vs Predicted values.

4.

KNN Algorithm:
This seems to be one of the most intuitive ways of going about classification/regression. Basically k- nearest neighbors (in terms of, say Euclidean Distance) of the query point are calculated. Then there are a variety of techniques on how to predict the value for the query point based on these k neighbors. Possible approaches include, mean of the neighbor values (regression & classification), weighted mean, etc.

I take into consideration the following scheme:

Unweighted: Here no weights are attached to the k neighbors while calculating the mean. This has the implication that all neighbors contribute equally, irrespective of their distance from the query point.

Distance Metric: Euclidean Distance

DataSet:

The SA Heart DataSet consists of 9 attributes + 1 prediction attribute.
The dataset has the Row.Num attribute which is removed by the program because it doesnt have any bearing on output attribute and should not be used to discriminate between instances.

Intuitions before experiment:
The classification accuracy should depend on values of k. My intuition says that k=1 may not give consistently good results because there may be cases where there are two results at almost the same distance but in opposite directions. Then, the result would be biased towards the point which is marginally closer. The results should get better with increasing k, but only upto a particular point, after which performance should degrade, as points not that close start contributing to the predictions as well.

The output is predicted as 1 if the predicted value > 0.5, 0 otherwise.

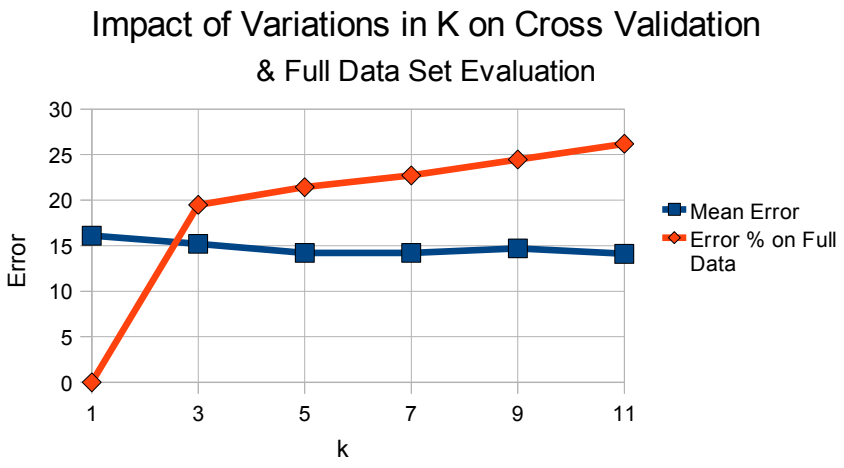
I define Error for 1 fold as the number of wrong predictions.
In that case, mean error for a V cross validation is

$$[\Sigma(i=1,V)\ Error(i)]/V$$

The following are the results for Variations in k using 10 fold cross validation :

K	Mean Error	Error % on Full Data	Accuracy %
1	16.1	0	100
3	15.2	19.48	80.52
5	14.2	21.43	78.57
7	14.2	22.72	77.28
9	14.7	24.45	75.54
11	14.1	26.19	73.8

This can be visualized as:



The effect of overfitting can be seen for k=1 where the error% on entire data set is 0, but the mean error in cross validation is not necessarily the least (in fact its the poorest)
k=11 has a mean error of 14.1 and gives an accuracy of 73.81% on the full dataset but k=5 seems to give results with marginally more mean error (14.2) and an accuracy of 78.58%