Semih Akbayrak

CMPE544

Final Project Report

I had chosen Heart Disease dataset of UCI repository for this project. Before starting to my project, I scanned other studies which had used same dataset and all of them approached to dataset as two classes case, such that there is a heart disease threat for patient (1) or not (0). Therefore I converted other degrees of the disease(2,3,4) to 1.

I eliminated some instances which don't include full features, and I divided remaining 297 instances to 3 parts. 100 for training, 100 for validation and 97 for testing.

Before applying feature selection, I used kNN classification. Because the scales of the features are different from each other, I used Mahalanobis distance instead of Eucledian distance. Below in figure1 you can find the error plot for validation set.

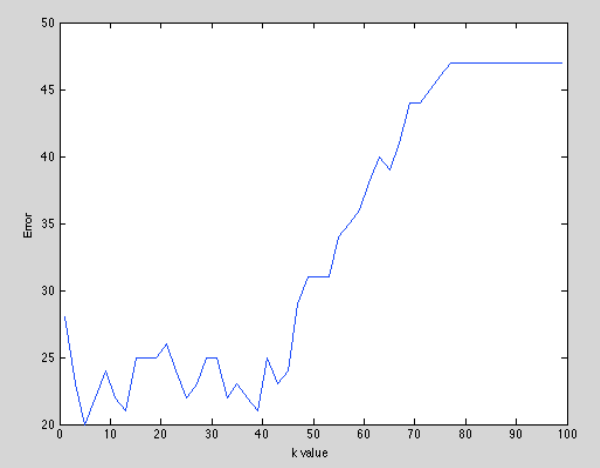

Figure1

As it can be seen k=5 gives the least error in validation set. By taking k as 5, I found accuracy rate of kNN on training set as 74.22%.

Then I applied Correlation-Based Feature Selection (CFS).

$$Merit_s = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}}$$

This equation is heuristic for CFS and hypothesis behind the heuristic is good feature subsets contain features highly correlated with the class, yet uncorrelated each other. In this eq. r_cf represents feature class correlation and r_ff represents feature feature correlation.

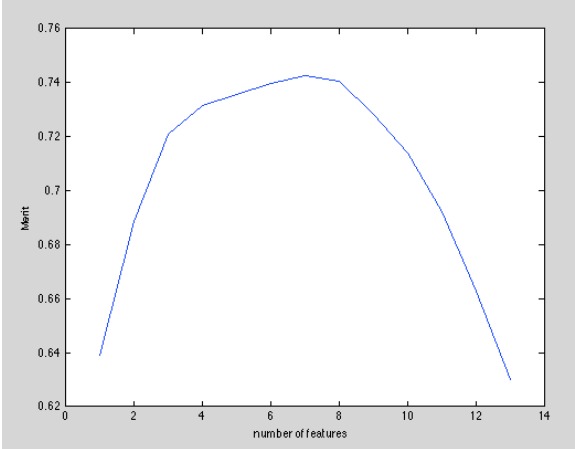

Figure2

Figure2 is the plot of number of features vs Merit. The subset which gives the highest Merit value contains 7 features and these are 13, 3, 12, 10, 9, 7, 2. Thus instead of 13 features, I started to use 7

features for the remaining parts of the project.

I applied kNN on this new dataset with 7 features and k=9, but the accuracy rate was almost same with the full feature dataset which is 76.28%.

I thought that combinations of features in new coordinate axis can show better performance, so I applied Principle Component Analysis(PCA) to improve accuracy. By applying PCA, I reached 5 dimensional new coordinate and then I applied kNN with k value 11 but accuracy didn't increase, I found 74.22%.

Generally, in kNN classification, we take the effects of neighbors same without looking their distances to the data point and all the kNN classifications I've mentioned about so far are also equally weighted kNN. Here I used weight adjusted kNN.

$$w_i = \begin{cases} \dfrac{d_k^{NN} - d_i^{NN}}{d_k^{NN} - d_1^{NN}}, & d_k^{NN} \neq d_1^{NN}, \\ 1, & d_k^{NN} = d_1^{NN}. \end{cases}$$

This equation shows how I decided weights. I used Dudani's weighting method. In this formula, d_1 is distance from data point to nearest neighbor. d_k is distance to kth nearest neighbor and d_i is distance to the ith nearest neighbor. According to this equation, kth nearest neighbor takes weight 0, nearest neighbor takes weight 1 and other neighbors' weights are in the interval (0,1) and inversely proportional to their distances from data point. Weighted kNN performed a little bit better than classical kNN with accuracy rate 79.38%.

In the last step of my project I applied Decision Tree classification. 6 of the 7 features of the dataset were discrete values. Only x10 was numeric. Because of the situation that dataset contains both discrete and numeric values, I used C4.5 algorithm. This time I found the optimal entropy threshold value with validation dataset and I found it as 0.5.

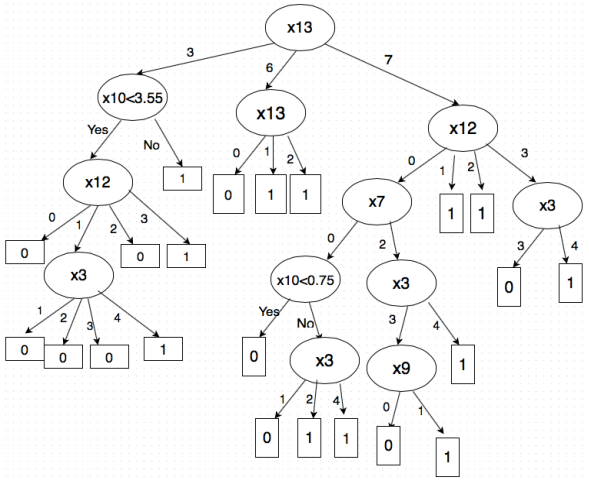

Figure3

Figure 3 is the tree which is determined by algorithm for threshold 0.5. According to tree, feature 2 can be eliminated as well because it isn't used as determination criteria. In the end Decision Tree classifier made classification with 78.35% accuracy with 6 features.