Aristidis Chronarakis 389

09/05/18

L7- Data Mining

Homework #1

Exercise 1

The purpose of this homework was to measure the accuracy of four different classifications methods, which were applied on two different datasets. In order to implement these four different methods, built-in functions from Matlab Platform were used. More specifically for each classification method, first the classification model was formed using the required function and suitable parameters. After, this model was cross validated by separating the whole dataset on ten disjoint sets and using nine of them for the training procedure and one for the computation of method's accuracy. The 10-fold cross validation was implemented with Matlab's function crossval() and default parameters.(default parameters = 10-fold cross validation). Final Matlab's kfoldLoss() function with cross validated model as parameter was used to compute the error and therefore the accuracy of the classification method. Below is the Matbal code which implements these 4 four classifications methods.

```
%EXERCISE #1 A.CHRONARAKIS

%load data
%DATA SET #1
load('AchronarakisData.mat');

%DATA SET #2
DS = readtable('Achronarakis_ClassData.txt');
X = DS(:,[1:8]);
classes = DS(:,[9]);

%KNN Classifier
KNNMdl = fitcknn(X,classes,'NumNeighbors',6,'Standardize',1);
CVKNNMdl = crossval(KNNMdl);
classAccuracyKNN = 1 - kfoldLoss(CVKNNMdl)

%NAIVE BAYES Classifier
NBMdl = fitcnb(X,classes,'CrossVal','on'); %Default DistributionNames=Normal
classAccuracyNB = 1 - kfoldLoss(NBMdl)
```

```
%SVM Classifier
%LINEAR
SVMLINEARMdl = fitcsvm(X,classes,'Standardize',true,'KernelFunction','linear');
CVSVMLINEARMdl = crossval(SVMLINEARMdl);
classAccuracySVMLINEAR = 1 - kfoldLoss(CVSVMLINEARMdl)

%RBF
SVMRBFMdl =
fitcsvm(X,classes,'Standardize',true,'KernelFunction','rbf','KernelScale','auto');
CVSVMRBFMdl = crossval(SVMRBFMdl);
classAccuracySVMRBF = 1 - kfoldLoss(CVSVMRBFMdl)

%DECISION TREES Classifier
DTMdl = fitctree(X,classes,'MaxNumSplits',7,'MinLeafSize',1,'MinParentSize',10);
CVDTMdl = crossval(DTMdl);
classAccuracyDT = 1 - kfoldLoss(CVDTMdl);
view(DTMdl,'mode','graph');
```

Dataset #1 (AchronarakisData.mat)

Results

Classification Method		***Accuracy***
kNN	**k=1**	0.633
	k=2	0.623
	k=3	0.664
	k=4	0.649
	k=5	0.649
	k=6	0.646
	k=7	0.662
	k=8	0.672
	k=9	0.683
Naive Bayes		0.522
SVM Linear Kernel		0.509
SVM RBF Kernel	**σ=auto (heuristic)=3.06**	0.771
	σ=default=1	0.615
	σ=5	0.768
	σ=10	0.588
	σ=100	0.502

Decision Trees	**Default**	0.563
	MaxNumSplits = 10 MinLeafSize = 10 MinParentSize = 10	0.532
	MaxNumSplits = 100 MinLeafSize = 10 MinParentSize = 10	0.607
	MaxNumSplits = 10 MinLeafSize = 100 MinParentSize = 10	0.530
	MaxNumSplits = 100 MinLeafSize = 100 MinParentSize = 10	0.505

Results Overview

The kNN classification method for the first datset gives the best accuracy for k=9 but all the different k give very close accuracy results. Naive Bayes & SVM Linear Kernel methods give very poor accuracy results approximately 50%. This means that the classifier will predict the correct class for the input data with a probablity 0.5 the same probability as we flip a coin in order to select the category one of the two categories. SVM RBF Kernel classification method with the suitable tuning for σ can give better accuracy results. With a σ value near to the value that heuristic gives for σ, we take the best result on SVM RBF Kernel. On Decision Trees parameters such as MaxNumSplits, MinLeafSize, MinParentSize can change the depth (size) of the tree and thus the complexity of tree and also the computation time. With the parameter values I choose the best accuracy results are given with MaxNumSplits=100, MinLeafSize=10 and MinParentSize=10. All other accuracy results are very close and therefore we can choose the simplest decision tree with the less complexity in order to have better computation time. From the above results, for the first dataset the best classification method is the SVM RBF Kernel with the σ given from the heuristic.

Dataset #2 (Achronarakis_ClassData.txt)

Results

Classification Method		***Accuracy***
kNN	**k=1**	0.696
	k=2	0.691
	k=3	0.737
	k=4	0.742
	k=5	0.726
	k=6	0.735
	k=7	0.744
	k=8	0.744
	k=9	0.737
Naive Bayes		0.760
SVM Linear Kernel		0.770
SVM RBF Kernel	**σ=auto (heuristic)=1.38**	0.751
	σ=default=1	0.718
	σ=5	0.770
	σ=10	0.7695
	σ=100	0.651
Decision Trees	**Default**	0.694
	MaxNumSplits = 10 MinLeafSize = 10 MinParentSize = 10	0.737
	MaxNumSplits = 100 MinLeafSize = 10 MinParentSize = 10	0.748
	MaxNumSplits = 10 MinLeafSize = 100 MinParentSize = 10	0.737
	MaxNumSplits = 100 MinLeafSize = 100 MinParentSize = 10	0.734

Results Overview

For the second dataset, the kNN classification method gives better accuracy for k>=3, with a little variance on accuracy results. Contrary to the first dataset, on the second dataset Naive Bayes & SVM Linear Kernel methods, give accuracy results close to the best. This means that the second dataset is more suitable for this method than the first. SVM RBF Kernel classification method with a selected value $\sigma=10$ gives the best results between the implementations of SVM RBF Kernel with different σ. It's worth to mention that Matlab's heuristic which is used to compute a σ, gives a $\sigma=1.38$ value which leads to accuracy results not so far from the best. Final, all Decision Trees implementations with different parameters which I use in order to control the complexity of the Decision Tree like MaxNumSplits, MinLeafSize and MinParentSize give almost the same accuracy results. These results are better from the results that Matlab's default Decision Tree implementation gives in which MaxNumSplits=N-1, MinLeafSize=1 and MinParentSize=10. From the above results can someone understand that the best accuracy results on classification problem for the second dataset can be given from Naive Bayes, SVM Linear Kernel and SVM RBF Kernel with s about 5.