CS4011-Assignment 1a

S Aakash(CS15B060)

August 30, 2017

1 Question 1

compute the centroid of class-zero by randomly generating twenty coordinates using np.random.rand function.
Now centroid of class-one = centroid of class zero +[some constant] np.random.rand(20).
We choose the constant factor depending on how much overlap we want.
Now,for covariance matrix,lets generate a random 20 * 20 matrix .Now lets name this matrix A.
The required covariance matrix is A.transpose * A .It is easy to see that the generated matrix is Positive Semidefinite and other constrains like non-diagonality etc. can be checked on this matrix.Note that matrix is generated randomly and there is very small chance for resultant matrix to be diagonal
Now we generate 2000 datasets of class 0 and 2000 datasets of class 1 and split them into training set and test sets.

2 Question 2

We take the test and training datasets from generated dataset and use linear regression model to predict the classes.We learn the parameters from training set and store in coeffs-q2.csv and we predict the value of y for the test set.Threshold is set as 0.5 for this problem,that is any data set with pridicted value larger than 0.5 will be categorised into class-1.
Precision,Recall and F-number are computed.
accuracy :0.988333333333
precision :0.989966555184
recall :0.986666666667
F-estimate:0.988313856427

3 Question 3

In k-NN method,we look at the k-nearest neighbours of a data and make a decision.Here k is the hyperparameter,ie fixing k determines the accuracy.

The optimal choice of the value k is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distict
The graph has been attached indicating the effect of k in determining accuracy,in Dataset folder(q2-graph.png)
k=10
precision :0.805
recall :0.887867647059
F-estimate:0.844405594406
k=20
precision :0.833333333333
recall :0.889679715302
F-estimate:0.860585197935
k=30
precision :0.835
recall :0.865284974093
F-estimate:0.849872773537
k=40
precision :0.851666666667
recall :0.870528109029
F-estimate:0.86099410278
k=50
precision :0.848333333333
recall :0.862711864407
F-estimate:0.855462184874
k=100
precision :0.811666666667
recall :0.841105354059
F-estimate:0.826123833757
k=200
precision :0.791666666667
recall :0.816151202749
F-estimate:0.80372250423
Optimal K:40
F-estimate:0.86099410278

So we can observe,as k increases F-estimate increases upto sometime and then decreases.For this example,choosing k=40 is quite optimal
And we can observe that id doesnot fit better than linear regression.The linear regression had a slightly better F-estimate compared to K-NN.Ofcourse,K-NN is highly dependent on ordering of the datasets,distribution of data,etc.

4 Question 4

We can fill the missing values using mean of the remaining values of feature,but it may not be great always.
Eg:Consider X=[1,1.4,1.6,?,12]
Now our mean estimate says missing value is 4.
But intutively we may feel the missing value would be closer to 1.6 or 2 and we may feel 12 may be some wrongly noted data.
In such a case one can take median of avilable values and fill the missing data
In my training set,median performed better than mean:
mean
Average Sq Error:0.0207384802517
median
Average Sq Error-median:0.0171925764804

Here,i am storing both mean(DS2-mean) imputed and median imputed datasets(DS2-median).
Another method would be to predict these missing values using other features.This would indeed be like linear regression and this process of predicting the missing values can be done by 'Recommender Systems'.

5 Question 5

Here,we perform regression on five different training data splits and get average of mean squared error for corresponding training data sets.
mean
Average Sq Error:0.0207384802517
median
Average Sq Error-median:0.0171925764804
Note that I have performed regression on both mean-Imputed and Median Imputed datasets
The coeffecients are written in coeffs-q5 file

6 Question 6

Various values of lambda were tried,for Ridge regression:
lambda=0.01
Mean squared error: 0.0196987
lambda=0.03
Mean squared error: 0.0194002
lambda=0.1
Mean squared error: 0.0185378
lambda=0.3

Mean squared error: 0.0185058
lambda=1.0
Mean squared error: 0.0186937
lambda=3.0
Mean squared error: 0.0182770
lambda=10.0
Mean squared error: 0.0190283
lambda=30.0
Mean squared error: 0.0200450
Oprimal lambda:3.0
We can observe optimal value of lambda is between 0.1 and 3.
Note that each of mean squared error is average taken over five different splits
Now once we get the coeffecients,we can throw away the coeffecients whose value is quite close to zero relatively over others and corresponding features also as they make very little contribution to prediction.
this can help us reduce size of matrix significantly
Now we make assumption that we will neglect all features whose coeffecients have absolute values less than 0.002
mean-sq-error-reduced-dim:0.0218983590855
This is obviously as expected because we are essentially neglecting some data