COMPUTER SCIENCE AND ENGINEERING IITM

Ishu Dharmendra Garg (CS13B060)

November 25, 2017

1 LINEAR CLASSIFIER

1.1 DATA GENERATION

The various parameters are generated as described below

- ***Covariance Matrix*** - The covariance matrix is a 10 by 10 diagonal square matrix. To make it non spherical ,I have ensured that all the diagonal entries not the same.
- ***Mean Vectors***- The first mean vector v1 is of length 10 with each entry randomly selected between [4,5]. The second vector v2 is generated by adding a constant vector v3 to v1. Also all the entries of v3 are the same. SO varying v3, we can control the distance between the two mean vectors namely v1 and v2.For the assignment the experimets were done on

$$\begin{bmatrix} 1 & 1.5 & 2 & 5.5 & 4 & 2 & 6 & 3 & 4 & 5 \\ 7 \end{bmatrix}$$

  v1 = [4.1576, 4.9706, 4.9572, 4.4854, 4.8003, 4.1419, 4.4218, 4.9157, 4.7922, 4.9595]
  v3 = c[1, 1, 1, 1, 1, 1, 1, 1, 1, 1] where c can be varied.
  v2 = v1 + v3

- ***Data set***- The data points were generated as described in the problem statement. Let us call the 60% of the data generated to be DS1 and the rest be denoted by DS2. The DS1 has 1200 data points with 600 data points from each class. Similarly DS2 has 800 data points wiht 400 data points from each class.
- ***Results***- Since each time the program runs a new data set is generated and thus the results may also vary for each run of the program. And thus the result documented in the report may also differ. But on an average all the outputs have the same qualitative nature as well as some common patters associated with them.

1.2 LINEAR CLASSIFIER

Linear regression was done on indicator matrix corresponding to the data points of DS1. The results of the linear classifier learnt on DS2 with different values of c are as follows.
Note - The points on the plane were classified as class 1 data points.

The mis - classification error percentage for class1 and class2 on test data (DS2) is shown in Table1.1 and the coefficients of the hyperplane learned is shown in Table 1.2 below corresponding to some values of c.

c	Error % for Class 1	Error % for Class 2
1.50	3.2500	3.1250
1.00	7.0000	6.5000
0.50	15.6250	14.1250
0.1	22.5000	23.8750

Table 1.1: Error percentage for different c

c	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10
1.5	4.955	-0.209	-0.166	-0.117	-0.0319	-0.057	-0.106	-0.042	-0.090	-0.065	-0.039
1.0	5.014	-0.254	-0.180	-0.129	-0.047	-0.064	-0.103	-0.041	-0.076	-0.054	-0.034
0.5	3.933	-0.212	-0.102	-0.098	-0.050	-0.050	-0.099	-0.029	-0.083	-0.049	-0.041
0.1	1.389	-0.061	-0.027	-0.062	0.003	-0.027	-0.030	-0.029	-0.034	-0.026	-0.003

Table 1.2: Coefficients of the separating plane

1.3 KNN CLASSIFIER

Given a training set the KNN classification for a point from test data was done by finding the majority class of the K nearest data points form training data(DS1) to that data point form test data. The results of the KNN classifier learnt on DS1 with different values of c are as follows.

I observed that for a particular c, typically high values of k gave better results (i.e. less error percentages). And as k becomes higher the results becomes more stable and further do not change much with increase of k and also for different training data sets.
Also it has been observed that the linear classifier performs better than the KNN classifier for such a given probability distribution. This can be seen by comparing the Table1.1 and Table1.3.

Note - In case of a tie during taking the majority the test point is classified as class 1.

The mis - classification error percentage for class1 and class2 on test data (DS2) is shown in Table1.3 corresponding to some values of c.

c	Error % for Class 1	Error % for Class 2
1.50	9.875	9.625
1.00	17.625	17.750
0.50	33.500	33.875
0.1	45.875	46.125

Table 1.3: Error percentage for different c for k = 10

Mis-classification error percentage for different values of k for particular value of c are shown in Table1.4 is shown below .

k	Error % for Class 1	Error % for Class 2
1	42.375	42.375
5	40.500	40.500
20	38.625	37.750
40	35.250	35.625
70	33.625	35.250
100	34.125	33.875
150	31.750	32.000
175	31.875	31.875
200	30.500	30.125

Table 1.4: Error percentage for different k for c = 0.5

2 Linear Regression

2.1 Data Generation

I made the data usable by filling the missing value by the sample mean of the attribute.
TODO - Tell why averaging is good
After making the data usable shuffled it and then divided the the whole data set in 5 parts. Each part was saved in the data folder with the naming convention as mentioned in the question.

2.2 Linear Regressor

Linear regression was done on whole of the training data. Then the RSS error was determined on whole of the training data itself.

RSS Error = 32.4703928902
coefficients learned are reported in the file named '../data/results/linearModelCoeff.csv'.

2.3 Ridge Regressor

Ridge-regression on the given data was done for different values of lambda. Table below shows the residual error for each value, on test data, averaged over 5 different 80 âĹŠ 20 splits. The coefficients are can be found at "../data/results/RidgeModelCoeffLamLambdaFoldFold Number.csv".
Tuning the lambda for 4 decimal places the least error is model was found out at
lambda = 0.0486
error = 7.55544125852

The errors for the various values of lambda is shown in the table

lambda	Error
0.000000	94.7466129147
0.010000	7.61379218152
0.020000	7.5754562232
0.030000	7.56142330114
0.040000	7.55640686134
0.050000	7.55546245671
0.060000	7.5565557932
0.070000	7.55873403277
0.080000	7.56151187799
0.090000	7.56462725935
0.100000	7.56793255605

Table 2.1: Error percentage for different k for c = 0.5

Is it possible to use the information you obtained during this experiment for feature TODO - feature selection wala "dataSet.png"

Figure 3.1: Caption

3 Feature Extraction

3.1 PCA

Principle Component Analysis was done on the given 3-D data set. Out of the three components I used the first two components and projected data on them. Then the linear regression was done for both the cases.

The first two principle components are
TODO - vectorize them
dir1 =
dir2 =