CS5011 - Introduction to Machine Learning
Programming Assignment - 1

B Adarsh
MM14B001

September 20, 2016

1. SYNTHETIC DATASET CREATION

Problem Statement

You will use a synthetic data set for the classification task. Generate two classes with 20 features each. Each class is given by a multivariate Gaussian distribution, with both classes sharing the same covariance matrix. Ensure that the covariance matrix is not spherical, i.e., that it is not a diagonal matrix, with all the diagonal entries being the same. Generate 2000 examples for each class. Choose the centroids for the classes close enough so that there is some overlap in the classes. Specify clearly the details of the parameters used for the data generation. Randomly pick 30% of each class (i.e., 600 data points per class) as a test set, and train the classifiers on the remaining 70% data When you report performance results, it should be on the left out 30%. Call this dataset at DS1.

Approach

The means have been set using a seeded random function (for the sake of reproducibility), and the separation between the means has been determined through trial-and-error (0.06 in this case) so that there is considerable overlap between the two distributions, which will ensure logically sound Precision-recall values in the succeeding questions. Both the classes have the same diagonal covariance matrices. A multivariate normal distribution of 2000 sample values is generated using these parameters. Random shuffling is done for the sake of consistency while making a 70:30 split of the datapoints.

Output / Results

It is observed that for a very less separation between the mean of the two distributions, the performance values are less than 1. As the separation increases, the scores increase and tend to 1 for a fairly long value of separation.

Two classes, each having 2000 sample values are generated. After random shuffling, 30% of all the points (1200 points) are saved as test set, and the remaining 2800 points are saved as the training set.

2. LINEAR CLASSIFICATION

Problem Statement

For DS1, learn a linear classifier by using regression on indicator variable. Report the best fit accuracy, precision, recall and F-measure achieved by the classifier, along with the coefficients learnt.

Approach

This problem is a case of classification using regression. The linear regressor is first used to solve the problem, and a classification is done from the predicted output based on a

threshold value.

Data is read from the DS1 train and test set generated in Q1. The labels are defined as 0 for class 1, and entries of 1 for class 2. The solution for Linear Regression is first obtained with the help of a function provided by *scikit-learn*, which solves for the general Linear Regression equation:

$$\beta = (X^T X)^{-1} X^T Y$$

From the prediction obtained from the regressor, all values above a threshold of 0.5 are classified with label 1 and the rest are classified with label 0. Performance measurement is done by comparing the prediction with the test set. The performance measures scored by the classifier on the test set of DS1 are:

$$\begin{aligned} Accuracy &= 0.645833333333 \\ Precision &= 0.642276422764 \\ Recall &= 0.658333333333 \\ F-score &= 0.650205761317 \end{aligned}$$

The coefficients learnt by the regressor are:

β_0	β_1	β_2	β_3	β_4	β_5
-0.00774357	-0.01863572	-0.05339317	-0.05377944	-0.13500975	-0.01759954
β_6	β_7	β_8	β_9	β_{10}	β_{11}
-0.1015629	-0.0011287	-0.00268768	-0.05455403	-0.04670928	-0.02261973
β_{12}	β_{13}	β_{14}	β_{15}	β_{16}	β_{17}
-0.03676759	-0.02327394	-0.78608244	-0.02608049	-0.00619185	-0.00641908
β_{18}	β_{19}	β_{20}			
-0.00166683	-0.00985444	-0.00538268			

3. LINEAR CLASSIFICATION

Problem Statement

For DS1, use k-NN to learn a classifier. Repeat the experiment for different values of k and report the performance for each value. Technically this is not a linear classifier, but I want you to appreciate how powerful linear classifiers can be. Do you do better than regression on indicator variables or worse? Are there particular values of k which perform better? Report the best fit accuracy, precision, recall and f-measure achieved by this classifier.

Approach

Data is read from the DS1 test and train files generated in Q1. A kNN-classifier is run on the training data, and the performance measures are correspondingly computed. The

performance values for different values of k are:

k	**Accuracy**	**Precision**	**Recall**	**F-value**
3	0.745	0.751712328767	0.731666666667	0.741554054054
4	0.689166666667	0.809264305177	0.495	0.614270941055
5	0.705833333333	0.70826306914	0.7	0.70410729254
6	0.6625	0.744360902256	0.495	0.594594594595
7	0.659166666667	0.662139219015	0.65	0.656013456686
8	0.649166666667	0.702031602709	0.518333333333	0.596356663471
9	0.659166666667	0.660504201681	0.655	0.657740585774
10	0.64	0.690045248869	0.508333333333	0.585412667946

As observed from the values, the accuracy for the given dataset is good for small values of k, and decreases as k increases. The k-NN classifier performs better than linear classifier in almost all the cases. Perhaps, because of the large training data size and a Gaussian (non-linear) distribution, k-NN finds more robustness as compared to linear classification.

In particular, the k-NN classifier works well for k < 10 for the given dataset, and works best for k = 3.

4. DATA IMPUTATION

Problem Statement

For the regression tasks, you will use the Communities and Crime Data Set from the UCI repository (http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime). This is a real-life data set and as such would not have the nice properties that we expect. Your first job is to make this dataset usable, by filling in all the missing values . Use the sample mean of the missing attribute. Is this is a good choice ? What else might you use? If you have a better method, describe it, and you may use it for filling in the missing data. Turn in the completed data set.

Approach and Observations

Data is read from communities.data.txt datafile. Missing values, initially pressent as "?" are replaced with the sample mean of each variable. After the entire dataset is imputed, it is stored, and additionally five 80:20 splits are created to facilitate for the datasets used in succeeding questions.

The problem statement has suggested replacing the missing values with the mean value, but this may not be a feasible solution always. This strategy can severely distort the distribution for this variable, leading to complications with summary measures including, notably, underestimates of the standard deviation. Moreover, mean imputation distorts relationships between variables by “pulling” estimates of the correlation toward zero.[1]

[1] http://www.stat.columbia.edu/ gelman/arm/missing.pdf

One approach to solving this anomaly is to get information from related observations. As seen from the documentation of the dataset, there are several divisions under common features like race percentage, agw percentage, per capita income etc. Sometimes, having enough information about one of the features like per capita income of one country, can yield missing information on per capita income of another country, given that we have sufficient prescience about the relation between the two countries.

Another approach would be to include an extra indicator for each continuous predictor variable with missingness, identifying which observations on that variable have missing data. Then the missing values in the partially observed predictor can be replaced by zeroes or by the mean. This strategy will yield biased coefficient estimates for the other predictors included in the model because it forces the slope to be the same across both missing-data groups. Adding interactions between an indicator for response and these predictors can help to alleviate this bias.

5. LINEAR REGRESSION

Problem Statement

Fit the above data using linear regression. Report the residual error of the best fit achieved on test data, averaged over 5 different 80-20 splits, along with the coefficients learnt.

Approach and Observations

Data is read from the five CandC train and test files and a linear regression model is learnt from the training set using the function provided by *scikit-learn*. Residual sum of squares error is calculated with respect to the test set.

The best fit (one with the least residual error) was found to occur at index **2** (*CandC-train2.csv*) and the residual error corresponding to the best fit is **5.81754169266**.

The coefficients learnt for each of the 5 split classes are stored in CandC-coeff<i>.csv, where $1 \leqslant i \leqslant 5$.

6. REGULARIZED LINEAR REGRESSION

Problem Statement

Use Ridge-regression on the above data. Repeat the experiment for different values of lambda. Report the residual error for each value, on test data, averaged over 5 different 80-20 splits, along with the coefficients learnt. Which value of lambda gives the best fit? Is it possible to use the information you obtained during this experiment for feature selection? If so, what is the best fit you achieve with a reduced set of features?

Approach and Observations

Data is read from the five CandC train and test files and a ridge regression model is learnt from the training set for various ridge parameters using the function provided by *scikit-learn*. Residual sum of squares error is calculated with respect to the test set for each ridge parameter.

Lambda	Residual Error
0.0010	7.54528002359
0.00289426612472	7.54069486925
0.00837677640068	7.52886844684
0.0242446201708	7.50215743324
0.070170382867	7.45459727875
0.20309176209	7.3926126086
0.587801607227	7.32822756164
1.70125427985	7.27672166509
4.92388263171	7.26442609689
14.251026703	7.30459925745
41.246263829	7.41467507287
119.377664171	7.74451703351
345.510729459	8.7618083087
1000.0	11.0915988962

The best fit (one with the least residual error) was found to occur at ridge parameter of **4.92388263171** (*CandC-train2.csv*) and the residual error corresponding to the best fit is **7.26442609689**.

7. FEATURE EXTRACTION

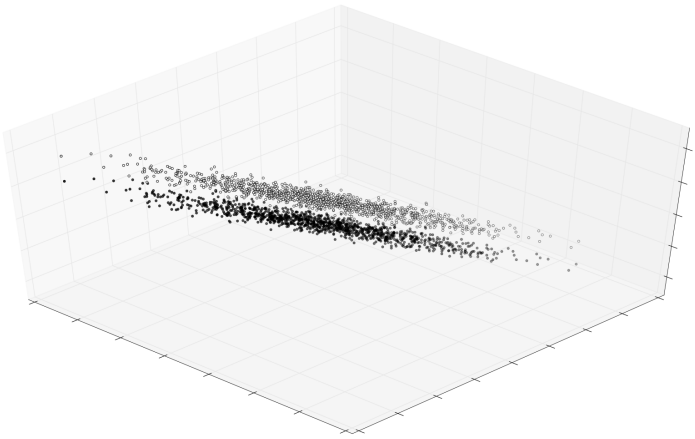
Problem Statement (Q7)

You have been provided with a 3-dimensional dataset (DS3) which contains 2 classes. Perform PCA on the dataset and extract 1 feature and use the data in this projected space to train linear regression with indicator random variables. Use the learnt model to classify the test instances. Report per-class precision, recall and f-measure. Also you have to report the 3-D plot of the dataset and the plot of the dataset in the projected space along with the classifier boundary.

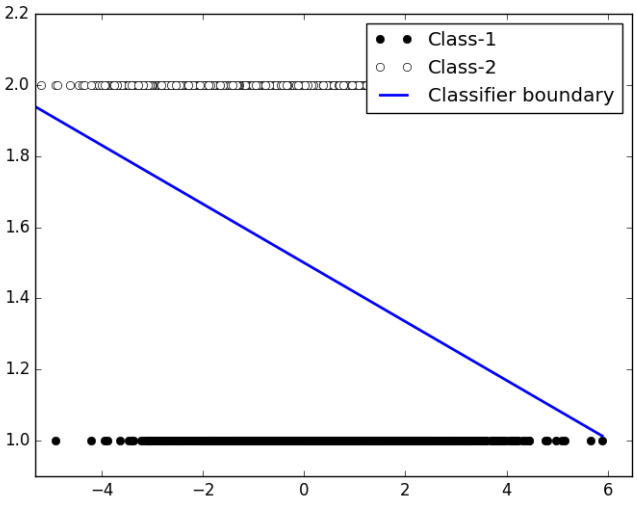
Approach and Observations

Training data is read from the train.csv file, and the labels are read from the train-labels.csv file. PCA is done on the given dataset, and the training and test data are projected along one of the principal component directions. Then, linear regression is done on the indicator variables on the projected dataset to classify the test instances.

Following is the scatter plot of the original training data set:



Following is the plot of the dataset in the projected space, showing the classifier boundary.



The performance values obtained for the two classes are:

Class 1

$$Accuracy = 0.6125$$
$$Precision = 0.611940298507$$
$$Recall = 0.615$$
$$F-score = 0.613466334165$$

Class 2

$$Accuracy = 0.6125$$
$$Precision = 0.613065326633$$
$$Recall = 0.61$$
$$F-score = 0.611528822055$$

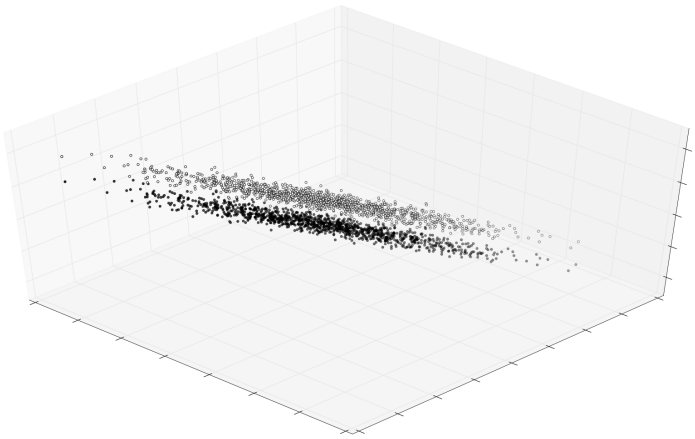
Problem Statement (Q8)

Now use the same dataset and perform LDA on it and project the dataset to the derived feature space. Report per-class precision, recall and f-measure. Also you have to report the 3-D plot of the dataset and the plot of the dataset in the projected space along with the classifier boundary. What do you infer from these two experiments? Which feature extraction technique performs better for this scenario? Why?

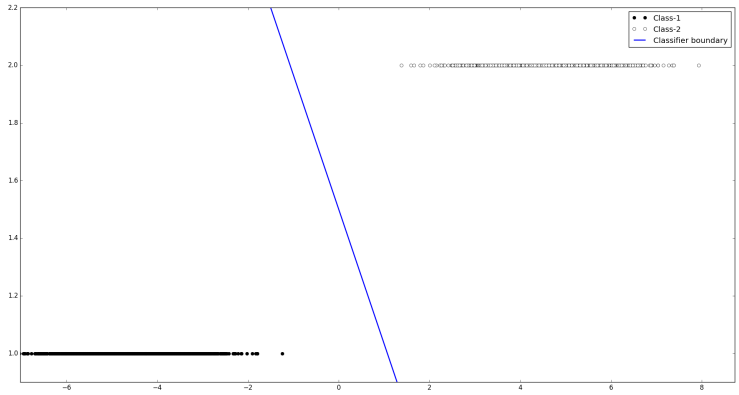
Approach and Observations

Training data is read from the train.csv file, and the labels are read from the train-labels.csv file. LDA is done on the given dataset, and the training and test data are projected along one of the principal component directions. A hyperplane is modelled using linear regression, which is of the form K + LX, and was then used to predict the classes of the test set elements.

Following is the scatter plot of the original training data set:



Following is the plot of the dataset in the projected space, showing the classifier boundary.



The performance values obtained for the two classes are:

Class 1

$$Accuracy = 1.0$$
$$Precision = 1.0$$
$$Recall = 1.0$$
$$F - score = 1.0$$

Class 2

$$Accuracy = 1.0$$
$$Precision = 1.0$$
$$Recall = 1.0$$
$$F-score = 1.0$$

Clearly, LDA has performed better than PCA in this case. The main difference is that PCA is label agnostic; it treats the entire data set as a whole. LDA, on the other hand, tries to explicitly model difference between classes (labels) within the data. In other words, LDA takes the output variable also into consideration for modelling the hyperplane, whereas PCA doesnt́.

8. LOGISTIC REGRESSION

Problem Statement

For this experiment, use only forest and mountain classes in DS2. Perform 2-class Logistic Regression on it. Report per-class precision, recall and f-measure on the same test data you used to test the neural net work. Now perform L 1 -regularized Logistic Regression on the same dataset and report similar performance results. Use l1 logreg code provided by Boyds Group (http://www.stanford.edu/ boyd/l1$_l ogreg$/)

Approach and Observations

In logistic regression, the dependent variable is categorical. Using linear regression in such cases faces 2 problems: it might predict value outside the range of the values which the dependent variable can take, and also solutions get affected due to class imbalance. Logistic regression solves these problems by a different cost function which maximises likelihood.

The performance values obtained for L2 and L1 are:

L2 Class 1

$$Accuracy = 0.525$$
$$Precision = 0.520$$
$$Recall = 0.650$$
$$F-score = 0.5778$$

L2 Class 2

$$Accuracy = 0.525$$
$$Precision = 0.533$$
$$Recall = 0.40$$
$$F-score = 0.0457$$

L1 Class 1

$$Accuracy = 0.625$$
$$Precision = 0.575$$
$$Recall = 0.970$$
$$F-score = 0.716$$

L1 Class 2

$$Accuracy = 0.625$$
$$Precision = 0.857$$
$$Recall = 0.30$$
$$F-score = 0.444$$

L1 regularised logistic regression performs better than L2 regularised logistic regression as we have obtained higher accuracy, precision, recall and F-measure in L1.