HOMEWORK 3

HAMSALEKHA PREMKUMAR

Instructions: Although this is a programming homework, you only need to hand in a pdf answer file. There is no need to submit the latex source or any code. You can choose any programming language, as long as you implement the algorithm from scratch.
Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file to Canvas. Please check Piazza for updates about the homework.

1 A Simplified 1NN Classifier

You are to implement a 1-nearest-neighbor learner for classification. To simplify your work, your program can assume that

- each item has d continuous features $\mathbf{x} \in \mathbb{R}^d$
- binary classification and the class label is encoded as $y \in \{0, 1\}$
- data files are in plaintext with one labeled item per line, separated by whitespace:

$$\begin{array}{cccc} x_{11} & \dots & x_{1d} & y_1 \\ & & \dots & \\ x_{n1} & \dots & x_{nd} & y_n \end{array}$$

Your program should implement a 1NN classifier:

- Use Mahalanobis distance d_A parametrized by a positive semidefinite (PSD) diagonal matrix A. For $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$,
$$d_A(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_A = \sqrt{(\mathbf{x} - x')^\top A(\mathbf{x} - x')}.$$
We will specify A in the questions below. (Hint: d is dimension while d_A with a subscript is distance)
- If multiple training points are the equidistant nearest neighbors of a test point, you may use any one of those training points to predict the label.
- You do not have to implement kd-tree.

2 Questions

1. (5 pts) What is the mathematical condition on the diagonal elements for a diagonal matrix A to be PSD?

   The diagonal elements of a PSD diagonal matrix should be greater than or equal to 0.

2. (5 pts) Given a training data set D, how do we preprocess it to make each feature dimension mean 0 and variance 1? (Hint: give the formula for $\hat{\mu}_j, \hat{\sigma}_j$ for each dimension j, and explain how to use them to normalize the data. You may use either the $\frac{1}{n}$ or $\frac{1}{n-1}$ version of sample variance. You may assume the sample variances are non-zero.)

   For $j = 1, 2, 3, ...k$,where k = number of features and n = number of sample data

$$\hat{\mu}_j = \frac{1}{n} \sum_{i=1}^{n} x_{i,j}$$

$$\hat{\sigma}_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{i,j} - \hat{\mu}_j)^2}$$

Data is normalised by subracting the feature column's mean, $\hat{\mu}_j$ from each entry in the corresponding feature column and dividing by standard deviation $\hat{\sigma}_j$. i.e.,
For $j = 1, 2, 3....k$, $x_{i,j} = \frac{x_{i,j} - \hat{\mu}_j}{\hat{\sigma}_j}$ for $i = 1, 2, ...n$

3. (5 pts) Let $\tilde{\mathbf{x}}$ be the preprocessed data. Give the formula for the Euclidean distance between $\tilde{\mathbf{x}}, \tilde{\mathbf{x}}'$.

$$D_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{\sum_{i=1}^{k}(\tilde{x}_i - \tilde{x}'_i)^2}$$

Vectorised notation,

$$D_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')^\top(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')}$$

4. (5 pts) Give the equivalent Mahalanobis distance on the original data $\mathbf{x}, \mathbf{x}'$ by specifying A. (Hint: you may need $\hat{\mu}_j, \hat{\sigma}_j$)

$$d_A(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_A = \sqrt{(\mathbf{x} - \mathbf{x}')^\top A(\mathbf{x} - \mathbf{x}')}.$$

where A is inverse of variance-covariance matrix. Using the normalization expression and simplifying, we get

$$A = \begin{bmatrix} \frac{1}{\hat{\sigma}_1^2} & 0 & \dots & 0 \\ 0 & \frac{1}{\hat{\sigma}_2^2} & \dots & \dots \\ \vdots & \vdots & \ddots & \\ 0 & \dots & \dots & \frac{1}{\hat{\sigma}_d^2} \end{bmatrix}$$

5. (5 pts) Let the diagonal elements of A be $a_{11}, \dots, a_{dd}$. Define a diagonal matrix L with diagonal $\sqrt{a_{11}}, \dots, \sqrt{a_{dd}}$. Define $\tilde{\mathbf{x}} = L\mathbf{x}$. Prove that $d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = d_A(\mathbf{x}, \mathbf{x}')$ where I is the identity matrix.

$$d_A(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}')^\top A(\mathbf{x} - \mathbf{x}')} = \sqrt{\begin{bmatrix} x_1 - x_1{}' & x_2 - x_2{}' & \dots & x_d - x_d{}' \end{bmatrix} \begin{bmatrix} a_{11} & \dots & 0 \\ 0 & a_{22} & \dots \\ 0 & \dots & a_{dd} \end{bmatrix} \begin{bmatrix} x_1 - x_1{}' \\ x_2 - \tilde{x_2}{}' \\ \vdots \\ x_d - x_d{}' \end{bmatrix}}$$

$$d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')^\top I(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')} = \sqrt{(\mathbf{x} - \mathbf{x}')^\top L^T L(\mathbf{x} - \mathbf{x}')}$$

$$= \sqrt{\begin{bmatrix} \tilde{x_1} - \tilde{x_1}{}' & \tilde{x_2} - \tilde{x_2}{}' & \dots & \tilde{x_d} - \tilde{x_d}{}' \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \dots & 0 \\ 0 & \sqrt{a_{22}} & \dots \\ 0 & \dots & \sqrt{a_{dd}} \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \dots & 0 \\ 0 & \sqrt{a_{22}} & \dots \\ 0 & \dots & \sqrt{a_{dd}} \end{bmatrix} \begin{bmatrix} \tilde{x_1} - \tilde{x_1}{}' \\ \tilde{x_2} - \tilde{x_2}{}' \\ \vdots \\ \tilde{x_d} - \tilde{x_d}{}' \end{bmatrix}}$$

It can be observed that,

$$L^T L = A$$

Thus $d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = d_A(\mathbf{x}, \mathbf{x}')$

6. (5 pts) Geometrically, what does $L\mathbf{x}$ do to the point $\mathbf{x}$? Explain in simple English.
L is a diagonal matrix constructed from A. A is the inverse variance-covariance matrix with diagonal elements corresponding to variances of the ith feature from its mean. Essentially, the diagonal elements of L are $\frac{1}{\sqrt{\hat{\sigma}_j}}$. This term scales the vector connecting the center of all the points to the point x. In effect, this is results the variables having variance 1 and in turn a distribution with standard deviation equal to 1.

7. (10 pts) Let U be any orthogonal matrix. Define $\tilde{\mathbf{x}} = UL\mathbf{x}$. (i) Prove that $d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = d_A(\mathbf{x}, \mathbf{x}')$ again. (ii) Geometrically, what does $UL\mathbf{x}$ do to the point $\mathbf{x}$? Explain in simple English. (i)

$$d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')^\top I(\tilde{\mathbf{x}} - \tilde{\mathbf{x}}')}$$

Since $\tilde{\mathbf{x}} = UL\mathbf{x}$,

$$d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(UL\mathbf{x} - UL\mathbf{x}')^T I(UL\mathbf{x} - UL\mathbf{x}')} = \sqrt{(\mathbf{x} - \mathbf{x}')^T L^T U^T UL(\mathbf{x} - \mathbf{x}')}$$

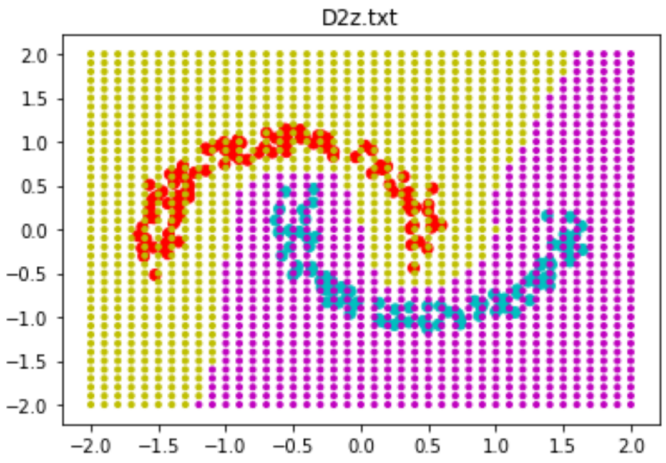
Since $U^T U = I$, and $L^T L = A$

$$= \sqrt{(\mathbf{x} - \mathbf{x}')^T A(\mathbf{x} - \mathbf{x}')} = d_A(\mathbf{x}, \mathbf{x}')$$

(ii)We know that, multiplying a vector by an orthogonal matrix reflects the vector in some plane and/or rotates it. $UL\mathbf{x}$ produces components in the direction of the eigen vector by projecting the datapoint in the direction of the eigen vectors. In effect, the eigen vectors become the new axes(bases) and the datapoints are scaled by weights depending on the feature correlations. This rotation of axes does not affect the distance.

8. (20 pts) Use the whole D2z.txt as training set. Use Euclidean distance (i.e. $A = I$). Visualize the predictions of 1NN on a 2D grid $[-2 : 0.1 : 2]^2$. That is, you should produce test points whose first feature goes over $-2, -1.9, -1.8, \ldots, 1.9, 2$, so does the second feature independent of the first feature. You should overlay the training set in the plot, just make sure we can tell which points are training, which are grid.

   a) Red: train points belonging to class 1
   b) Cyan: train points belonging to class 0
   c) Yellow: grid points classified as class 1
   d) Magenta: grid points classified as class 0



9. (To normalize, or not to normalize?) Start from D2a.txt. Perform 5-fold cross validation.

   (a) (5 pts) Do not normalize the data. Report 1NN cross validation error rate for each fold, then the average (that's 6 numbers).

Fold number	Number of misclassified points
1	0
2	0
3	0
4	0
5	0

Total misclassification $= 0$
misclassification rate$= 0\%$
Accuracy$= 100\%$

(b) (5 pts) Normalize the data. Report 1NN cross validation error rate (again 6 numbers). (Hints: Do not normalize the labels! The relevant quantities should be estimated from the training portion, but applied to both training and validation portions. This should happen 5 times. Also, you would either change $\mathbf{x}$ into $\tilde{\mathbf{x}} = L\mathbf{x}$ but then use Euclidean distance on $\tilde{\mathbf{x}}$, or do not change $\mathbf{x}$ but use an appropriate A; don't mix the two.)

Fold number	Number of misclassified points
1	4
2	3
3	4
4	6
5	3

Total misclassification $= 20$
misclassification rate$= \frac{20}{200} * 100 = 10\%$
Accuracy$= 80\%$

(c) (5 pts) Look at D2a.txt, explain the effect of normalization on CV error. Hint: the first 4 features are different than the next 2 features.
Visualizing the distribution of the features, x_1, x_2, x_3, x_4, it can be seen they are are similar and have low values while the ones ofx_5 and x_6 are have higher values. Normalization bring all these features in the range 0 to 1, essentially weighing all features equally. It is observed that the accuracy decreases which implies that bringing all values in the range on 0 to 1 is not useful.This gives an intuition that for this dataset, it is necessary for different features to be weighed differently to fit the model to the training data.

10. (Again. 10 pts) Repeat the above question, starting from D2b.txt.
(a)

Fold number	Number of misclassified points
1	6
2	8
3	6
4	11
5	8

Total misclassification $= 39$
misclassification rate$= \frac{39}{200} * 100 = 19.5\%$
Accuracy$= 80.5\%$
(b)

Fold number	Number of misclassified points
1	0
2	0
3	0
4	0
5	0

Total misclassification $= 0$
misclassification rate$= 0\%$
Accuracy$= 100\%$
(c) In this dataset, it is observed that x_1 and x_2 have different ranges. But it is observed that the CV error reduced with normalization. This gives an intuition that reducing the values to the range 0 to 1 and thereby considering equal importance to both features is necessary to better fit a model to the given training data.

11. (5 pts) What do you learn from Q9 and Q10?
Benefits of normalization depends on the dataset. It is useful in most cases but not all. Having prior knowledge about the importance of features or analyzing the data can help decide the necessity of normalization.

12. (Weka, 10 pts) Repeat Q9 and Q10 with Weka. Convert appropriate data files into ARFF format. Choose classifiers / lazy / IBk. Set $K = 1$. Choose 5-fold cross validation. Let us know what else you needed to set. Compare Weka's results to your Q9 and Q10.

dontNormalize parameter of the ***distanceFunction*** of the ***nearestNeighbourAlgorithm*** had to be toggled on and off to obtain the below results. It can be observed that the respective results in both Q9 and Q10 match.

(1a.) D2a.txt with No normalization

```
Test mode:    5-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         200               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0062
Root mean squared error                  0.0062
Relative absolute error                  1.2422 %
Root relative squared error              1.2383 %
Total Number of Instances              200

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     0
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     1
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

   a   b   <-- classified as
  92   0 |   a = 0
   0 108 |   b = 1
```

(1b.) D2a.txt with normalization

```
Test mode:    5-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         189               94.5      %
Incorrectly Classified Instances        11                5.5      %
Kappa statistic                          0.8892
Mean absolute error                      0.0605
Root mean squared error                  0.2332
Relative absolute error                 12.1739 %
Root relative squared error             46.7722 %
Total Number of Instances              200

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.935    0.046    0.945      0.935    0.940      0.889    0.944     0.913     0
                 0.954    0.065    0.945      0.954    0.949      0.889    0.944     0.926     1
Weighted Avg.    0.945    0.057    0.945      0.945    0.945      0.889    0.944     0.920

=== Confusion Matrix ===

   a   b   <-- classified as
  86   6 |   a = 0
   5 103 |   b = 1
```

(2a.) D2b.txt with No normalization

```
Test mode:    5-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         161               80.5    %
Incorrectly Classified Instances        39               19.5    %
Kappa statistic                          0.6084
Mean absolute error                      0.1988
Root mean squared error                  0.4389
Relative absolute error                 40      %
Root relative squared error             88.0469 %
Total Number of Instances              200

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.804    0.194      0.779      0.804    0.791      0.609    0.805     0.717     0
                 0.806    0.196      0.829      0.806    0.817      0.609    0.805     0.772     1
Weighted Avg.    0.805    0.195      0.806      0.805    0.805      0.609    0.805     0.747

=== Confusion Matrix ===

  a  b   <-- classified as
 74 18 |  a = 0
 21 87 |  b = 1
```

(2b.) D2b.txt with normalization

```
Test mode:    5-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         200               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0.0062
Root mean squared error                  0.0062
Relative absolute error                  1.2422 %
Root relative squared error              1.2383 %
Total Number of Instances              200

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 1.000    0.000      1.000      1.000    1.000      1.000    1.000     1.000     0
                 1.000    0.000      1.000      1.000    1.000      1.000    1.000     1.000     1
Weighted Avg.    1.000    0.000      1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

   a   b   <-- classified as
  92   0 |   a = 0
   0 108 |   b = 1
```