# HOMEWORK 3

Neal Satitsumpun
9078066702

**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?
   diagonal elemetns have to be non-negative.

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.)
   Let $x_{ij}$ be $j^{th}$ feature in $i^{th}$ training sample from $D$
   $\hat{\mu}_j = \frac{1}{|D|}\sum_{i=1}^{|D|} x_{ij}$ and $\hat{\sigma}_j = \sqrt{\frac{1}{|D|}\sum_{i=1}^{|D|}(x_{ij} - \hat{\mu}_j)^2}$
   The preprocessed $\tilde{x}_{ij}$ from $x_{ij}$ is

   $$\tilde{x}_{ij} = \frac{x_{ij} - \hat{\mu}_j}{\hat{\sigma}_j}$$

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

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

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

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

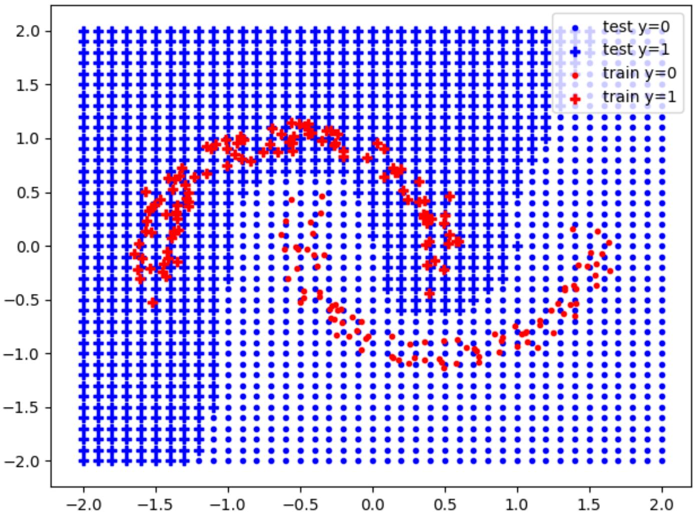
$$d(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\mathbf{x} - x')^\top A(\mathbf{x} - x')}.$$

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.
Since $\tilde{\mathbf{x}} = L\mathbf{x}$ and $\tilde{\mathbf{x}}' = L\mathbf{x}'$, $\tilde{\mathbf{x}} - \tilde{\mathbf{x}}' = L\mathbf{x} - L\mathbf{x}' = L(\mathbf{x} - \mathbf{x}')$
$d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{x}')^\top I(\tilde{\mathbf{x}} - \tilde{x}')} = \sqrt{(L(\mathbf{x} - x'))^\top I(L(\mathbf{x} - x'))} = \sqrt{(\mathbf{x} - x')^\top L^\top IL(\mathbf{x} - x')}$
$\sqrt{(\mathbf{x} - x')^\top (L^\top L)(\mathbf{x} - x')} = \sqrt{(\mathbf{x} - x')^\top A(\mathbf{x} - x')} = d_A(\mathbf{x}, \mathbf{x}')$

6. (5 pts) Geometrically, what does $L\mathbf{x}$ do to the point $\mathbf{x}$? Explain in simple English.
It scales each dimension of x by different positive amount so that the spread of each dimension is about the same.

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) Since $\tilde{\mathbf{x}} = UL\mathbf{x}$ and $\tilde{\mathbf{x}}' = UL\mathbf{x}'$, $\tilde{\mathbf{x}} - \tilde{\mathbf{x}}' = UL\mathbf{x} - UL\mathbf{x}' = UL(\mathbf{x} - \mathbf{x}')$
$d_I(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}') = \sqrt{(\tilde{\mathbf{x}} - \tilde{x}')^\top I(\tilde{\mathbf{x}} - \tilde{x}')} = \sqrt{(UL(\mathbf{x} - x'))^\top I(UL(\mathbf{x} - x'))} = \sqrt{(\mathbf{x} - x')^\top L^\top U^\top IUL(\mathbf{x} - x')}$
$\sqrt{(\mathbf{x} - x')^\top L^\top (U^\top U)L(\mathbf{x} - x')} = \sqrt{(\mathbf{x} - x')^\top L^\top IL(\mathbf{x} - x')} = \sqrt{(\mathbf{x} - x')^\top A(\mathbf{x} - x')} = d_A(\mathbf{x}, \mathbf{x}')$
(ii) It scales each dimension of x by different positive amount so that the spread of each dimension is about the same. Then, it rotate x around some particular axis.

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 $-2, -1.9, -1.8, \dots, 1.9, 2$, 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.



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).
   unnormalized error rate: 0.000000
   unnormalized error rate: 0.000000
   unnormalized error rate: 0.000000
   unnormalized error rate: 0.000000
   unnormalized error rate: 0.000000
   average rate: 0.000000

   (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.)
   normalized error rate: 0.025000
   normalized error rate: 0.075000
   normalized error rate: 0.075000
   normalized error rate: 0.150000
   normalized error rate: 0.076923
   average error rate: 0.080385

   (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.
   Normalization worsen the 1NN. The last 2 features might be more important features in classifying data than first 4 features. The first 4 features might be just noise. Because, after normalization, every features have the same importance, the 1NN performed worse.

10. (Again. 10 pts) Repeat the above question, starting from D2b.txt.
    (a)
    unnormalized error rate: 0.300000
    unnormalized error rate: 0.075000
    unnormalized error rate: 0.225000
    unnormalized error rate: 0.150000
    unnormalized error rate: 0.205128
    average rate: 0.191026
    (b)
    normalized error rate: 0.000000
    normalized error rate: 0.000000
    normalized error rate: 0.000000
    normalized error rate: 0.000000
    normalized error rate: 0.000000
    average error rate: 0.000000
    (c)
    the normalization improves the accuracy of 1NN. The importance of both features could be equally important but the scale of both features are different.

11. (5 pts) What do you learn from Q9 and Q10?
    Normalization should not be automatically applied to the dataset. We might need some domain knowledge to help analyze the significance of each feature.

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.

    - have to set "doNotNormalize" to true for the 5-fold without normalization

- (Q9)
  - without normalization
  Correctly Classified Instances 200 100 %
  Incorrectly Classified Instances 0 0 %
  - with normalization Correctly Classified Instances 189 94.5 %
  Incorrectly Classified Instances 11 5.5 %

- (Q10)
  - without normalization
  Correctly Classified Instances 161 80.5 %
  Incorrectly Classified Instances 39 19.5 %
  - with normalization
  Correctly Classified Instances 200 100 %
  Incorrectly Classified Instances 0 0 %

  Results are similar to Q9 and Q10 where normalization worsen Q9 and improve Q10