CS539 Machine Learning
Homework 1 - Spring 2017

PROF. CAROLINA RUIZ

Due Date: Thursday, February 9th, 2017

HW Instructions

- Carefully study Chapters 1, 2, 3 (except section 3.5), 4, and 5 of the textbook, your class notes, and any other related materials posted on the course webpage.
- Solve each of the problems and exercises assigned in this Homework.
- **Sections B and C are programming assignments to be completed in Matlab** (or R with professor's permission). **You must write your own code.** *No other programming languages are allowed on this project*. Make sure to consult online documentation for Matlab (or for R). Also, my miscellaneous notes on Matlab (and R) may be useful for this project.
- **You don't need to submit your homework solutions. Instead, an in-class Test will be given the day that the homework is due. This Test will evaluate your mastering of the material covered by the homework.**
- *This is meant to be an individual homework*. That is, you are expected to work on this homework on your own to make sure you know the material and know how to solve the problems, since you'll be tested individually in the Test. *Nevertheless you can discuss your questions about the homework on the Canvas' discussion forums, and consult with the professor and the TA during office hours, and with classmates if you have any trouble solving the homework problems*.

Section A: Exercises from the Textbook (75 points)

- **Chapter 2:** (Pages 43-46)
  i. Study solutions to Exercises 1, 2, 3, 4, 5.
  ii. (5 points each) Solve exercises 6, 7, 8, 9, 10, 11.
  iii. (5 points) For the Hypothesis Space H in Exercise 2 (i.e., each hypothesis is a set of rectangles), calculate the VC dimension of H. Explain your answer.

iv. (5 points) (a) Use the formulas derived in the Probably Approximately Correct (PAC) learning section on pp. 29-30 to determine the minimum number of training data instances N needed so that with at least 95% confidence, the probability of misclassifying a data instance with the tightest rectangle hypothesis will be at most 0.01.
(b) With the answer that you obtain from applying the fomulas go back over the sequence of derivations of the formulas to make sure you understand the logic behind this sequence of derivations.

- **Chapter 3:** (Pages 60-64)
  i. Study solutions to Exercises 1, 2, 3, 4.

- **Chapter 4:** (Pages 89-90)
  i. Study solutions to Exercises 4, 5.
  ii. (5 points each) Solve exercises 8, 9.
  iii. (5 points) Solve Exercise 4 assuming that the two means are the same $\mu_1 = \mu_2$, but the standard deviations are different (assume $\sigma_1 > \sigma_2$). Determine how many discriminant points exist and calculate it/them analytically.

- **Chapter 5:** (Pages 112-113)
  i. Study solutions to Exercises 1, 7, 8.
  ii. (5 points each) Solve exercises 4, 5, 6, 9.

Section B: Univariate Data (175 points + bonus points)

Important: When you are asked to randomly generate data, make sure to record the random seed used for the generation so that you can reproduce your experiments later.

I. **Data Generation:**
(5 points) Randomly generate a dataset X with N=1000 consisting of one attribute normally distributed with mean=60 and standard deviation=8.

II. **MLE:**
1. (10 points) Use the formulas (4.8) p. 68 to find the Maximum Likelihood Estimation (MLE) of sample distribution parameters (mean and stardard deviation) directly from the sample. Show your work in the report.
2. (10 points) Use the Maximum Likelihood Estimation (MLE) function provided by Matlab to calculate these parameter values from X. Do these parameter values coincide with the ones you found directly from the formulas above? Explain.

III. **MAP and Bayes' Estimator:**
In this part, you will look at the Maximum A Posteriori (MAP) and Bayes' estimator to estimate the parameter values of the sample X above. Assume that collection of all these possible parameter value estimates is also distributed normally. That is, $X \sim N(\theta,\sigma^2)$ and $\theta \sim N(\mu_0,\sigma_0^2)$. Assume that $\sigma=8$, $\mu_0=60$, $\sigma_0=3$.
1. (10 points) Calculate the MAP estimate and the Bayes' estimate of the mean value used to generate data sample X. Are the MAP estimate and the Bayes' estimate the same in this case?

Why or why not?
2. (5 points) Should the MAP estimate in this case be the same as the mean estimated by MLE? Why or why not?

IV. **Classification:**
1. (5 points) Randomly generate 3 normally distributed samples, each consisting of just one attribute as follows:
   - Sample 1: number of instances: 500, mean=60 and standard deviation=8.
   - Sample 2: number of instances: 300, mean=30 and standard deviation=12.
   - Sample 3: number of instances: 200, mean=80 and standard deviation=4.

   Create a dataset X that consists of these 3 samples, where data instances in Sample i above belong to class C_i, for i=1, 2, 3.
2. (10 points) Following the material presented in Section 4.5 of the textbook, define a precise discriminant function g_i for each class C_i. Remember to apply MLE to estimate the parameters of each of the classes. Show your work.
3. (5 points) Based on these discriminant functions, what would be the chosen class for each of the following inputs: x = 10, 30, 50, 70, 90. Show your work.
4. (15 points) Find analytically the "decision thresholds" (see Fig. 4.2 p. 75) for these 3 classes.
5. (5 points) Implement each of these 3 discriminant function g_i as a new function in Matlab.
6. (5 points) Based on these 3 functions, implement a "decision" function that receives a number x as its input and outputs i, where i is the chosen class for input x. Test your function on inputs: x = 10, 30, 50, 70, 90. Show the results in your report.
7. (5 points) Use your decision function on inputs: x = 0, 0.5, 1, 1.5, ..., 99, 99.5, 100. Do the "decision thresholds" you calculated analytically coincide with the results of this test? Explain.
8. (10 points) Generate a pair of plots like those in Fig. 4.2 for this particular dataset.
9. (10 points) Use stratified random sampling to split your dataset into 2 parts: a training set (with 60% of the data instances) and a validation set (with the remaining 40% of the data instances). Test the "decision" function that you implemented on part 6 above on the validation set. Report the accuracy and the confusion matrix of your decision function, as well as the precision and the recall of your decision function for each of the three classes.

V. **Regression:**
1. (10 points) Create a dataset consisting of one input and one output as follows. For the input, use the dataset X you generated in part I above with N=1000, mean=60 and standard deviation=8.

   For the output, use $r = f(x) + \varepsilon$ where $f(x) = 2 \sin(1.5x)$, and the noise $\varepsilon \sim N(\mu=0,\sigma^2=1)$. (as in the example in Sections 4.6-4.8 pp. 77-87).
2. (5 points) Use random sampling to split your dataset into 2 parts: a training set (with 60% of the data instances) and a validation set (with the remaining 40% of the data instances).
3. (10 points) Create three 2-dimensional plots: one for the entire dataset X, one for the training set, and one for the validation set. In each of these plots, the x axis correspond to the input variable x, and the y axis corresponds to the output (response) variable r.
4. (15 points) Create 5 different regression models over the training set using the regression functionality provided by Matlab:

   $g_k(x| w_k,...,w_0) = w_k x^k + ... + w_1 x + w_0$, for k=0,1,2,3,4. Report the obtained coefficients in your written report.
5. (15 points) Create two 2-dimensional plots: one containing the training set and the 5 fitting

curves, and one containing the validation set and the 5 fitting curves obtained over the training set. In each of these plots, the x axis correspond to the input variable x, and the y axis corresponds to the output (response) variable r.

6. (10 points) Evaluate each of the 5 regression models over the validation set. Report the Sum of Square Errors (SSE), the Root Mean Square Error (RMSE), the Relative Square Error (RSE), and the Coeffient of Determination (R^2) of each regression model over the validation set. If the programming language you are using reports AIC, BIC, and/or log likelihood values, include these values in your report too. Based on these error measures, which model would you pick among the five regression models? Explain.
7. (Bonus points) See if the regression functionality in Matlab allows the use of Akaike information criterion (AIC). and/or the use of Bayesian information criterion (BIC), instead of minimizing SSE, to guide the construction of the regression model. If so, repeat parts 4 and 6 above for AIC and then for BIC. Which of the three approaches produced better results? Explain.

Section C: Multivariate Data (155 points + bonus points)

Important: When you are asked to randomly generate data, make sure to record the random seed used for the generation so that you can reproduce your experiments later.

I. **Multivariate Normal Distribution:**
In this part, you will work with randomly generated datasets with N=1000 data instances and d=20 dimensions (attributes). Each dataset will be generated using a multivariate normal distribution with parameters $\boldsymbol{\mu}$ (1-by-d vector of means, one for each attribute) and $\boldsymbol{\Sigma}$ (d-by-d covariance matrix). To simplify the notation, we'll denote $\boldsymbol{\mu}$ by "trueMeans" and $\boldsymbol{\Sigma}$ by "trueSigma".

- **Multivariate Data Generation:**
(10 points) Use functionality in the programming language you chose (e.g., in Matlab, use the *mvnrnd* function "Multivariate normal random numbers") to randomly generate three multivariate normally distributed datasets X1, X2, X3 as described below.
  - trueMeans: For all three datasets use the same vector of means: trueMeans.
  - trueSigma: The covariance (Sigma) matrix for each dataset is specified below:
    i. Dataset X1 (*arbitrary covariance matrix*): Use trueSigmaA as covariance (Sigma) matrix.
    ii. Dataset X2 (*diagonal covariance matrix*): Use trueSigmaD as covariance (Sigma) matrix.
    iii. Dataset X3 (*identity covariance matrix*): Use the d-by-d identity matrix as covariance (Sigma) matrix.

- **Parameter Estimation:**
(10 points) For each of the datasets X1, X2, and X3 do the following:
  1. Estimate the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ from the dataset. Let's call these estimates "estimatedMeans" and "estimatedSigma". Compare these estimates with the trueMeans and the trueSigma used to generate the dataset and describe your observations.
  2. Devise a good way to plot the dataset in 2 or 3 dimensions.

II. **Multivariate Classification:**

In this part, you will work with datasets that consist of 2 classes C_1 and C_2. These datasets will contain N=1800 data instances and d=20 attributes.

- **Multivariate Data Generation:**
  (10 points) In all cases described below, you will use the multivariate datasets generated in part I above as class C_1 and will generate class C_2 use functionality in the programming language you chose that generates multivariate normally distributed data.
  i. Dataset DX (*classes have different arbitrary covariance matrices)*:
     - The 1,000 data instances in C_1 will be those in the dataset X1 generated above.
     - The 800 data instances in C_2 will be generated using parameters trueMeans2 and trueSigmaA2.
  ii. Dataset SX1 (*classes share the same arbitrary covariance matrix)*:
     - The 1,000 data instances in C_1 will be those in the dataset X1 generated above.
     - The 800 data instances in C_2 will be generated using parameters trueMeans2 and trueSigmaA.
  iii. Dataset SX2 (*classes share the same diagonal covariance matrix)*:
     - The 1,000 data instances in C_1 will be those in the dataset X2 generated above.
     - The 800 data instances in C_2 will be generated using parameters trueMeans2 and trueSigmaD.
  iv. Dataset SX3 (*classes share the identity covariance matrix)*:
     - The 1,000 data instances in C_1 will be those in the dataset X3 generated above.
     - The 800 data instances in C_2 will be generated using parameters trueMeans2 and the d-by-d identity matrix as covariance (Sigma) matrix.

- **Multivariate Discriminant Functions:**
  For each of the 4 datasets under consideration (DX, SX1, SX2, and SX3) do the following:
  1. (8 points) Determine which of the formulas in Section 5.5 of the textbook should be used to define a precise discriminant function g_i for each class C_i of the dataset at hand. Explain your answer.
  2. (12 points) Implement each of these 2 discriminant function g_i as a new function in Matlab.
  3. (4 points) Based on these 2 functions, implement a "decision" function that receives a data instance **x** (which is a 1-by-d vector) as its input and outputs i, where i is the chosen class for input x.
  4. (16 points) Use stratified random sampling to split your dataset into 2 parts: a training set (with 60% of the data instances) and a validation set (with the remaining 40% of the data instances). Test your "decision" function on the validation set. Report the accuracy and the confusion matrix of your decision function, as well as the precision and the recall of your decision function for each of the two classes.
  5. (20 points) Devise a good way of plotting the dataset in 2 or 3 dimensions to see and contrast the shapes of the two classes in the dataset.
     (Bonus points) Add to this plot the decision boundary between the two classes (that is, the curve defined by $P(C_1|\mathbf{x}) = 0.5$).

III. **Multivariate Regression:**

1. (10 points) Create a dataset consisting of d inputs and one output as follows. For the d inputs, use the multivariate dataset X1 you generated in part I above with N=1000, trueMeans and trueSigmaA. For the output, use r = f(**x**) + ε where f(**x**) = 3*average(**x**) - min(**x**), that is the output is three times the average of the d input values minus the minimum input value; and the noise ε ~ N(μ=0,σ^2=1).
2. (5 points) Use random sampling to split your dataset into 2 parts: a training set (with 60% of the data instances) and a validation set (with the remaining 40% of the data instances).
3. (10 points) Create a multivariate linear regression model over the training set using the regression functionality provided by Matlab. Report the obtained regression formula in your written report.
4. (10 points) Evaluate the regression model over the validation set. Report the Sum of Square Errors (SSE), the Root Mean Square Error (RMSE), the Relative Square Error (RSE), and the Coeffient of Determination (R^2) of each regression model over the validation set. If the programming language you are using reports AIC, BIC, and/or log likelihood values, include these values in your report too.
5. (Bonus points) See if the regression functionality in Matlab allows the use of Akaike information criterion (AIC). and/or the use of Bayesian information criterion (BIC), instead of minimizing SSE, to guide the construction of the regression model. If so, repeat part 4 above for AIC and then for BIC. Which of the three approaches produced better results? Explain.
6. Bias and Variance:
    a. (10 points) Construct 10 new different datasets D1, ..., D10 each one consisting of 100 data instances randomly generated with trueMeans and trueSigmaA. For the output, use r = f(**x**) + ε where f(**x**) = 3*average(**x**) - min(**x**) and the noise ε ~ N(μ=0,σ^2=1) as before.
    b. (10 points) Fit a multivariate linear regression formula g_i to each of these datasets.
    c. (10 points) Estimate the bias and the variance using the formulas on slide 24 of Chapter 4 slides (see also Section 4.7 of the textbook). Apply the formulas for bias and variance over the **x**'s in the dataset X1 (together with the output value) that you constructed in part 1 above (hence N=1000 and M=10).
