CSCI 5521: Introduction to Machine Learning (Spring 2020)[1]

Homework 2

Questions

1. (**30 points**) In this problem, you will implement a program to fit two multivariate Gaussian distributions to the 2-class data and classify the test data by computing the log odds $log\frac{P(C_1|x)}{P(C_2|x)}$. The priors $P(C_1)$ and $P(C_2)$ should be estimated from the training data. Three pairs of training data and test data are given. The parameters μ_1, μ_2, $\mathbf{S_1}$ and $\mathbf{S_2}$, the mean and covariance for class 1 and class 2, are learned in the following three models for each training data and test data pair,

   - **Model 1**: Assume independent $\mathbf{S_1}$ and $\mathbf{S_2}$ (the discriminant function is as equation (5.17) in the textbook).
   - **Model 2**: Assume $\mathbf{S_1} = \mathbf{S_2}$. In other words, shared $\mathbf{S}$ between two classes (the discriminant function is as equation (5.22) in the textbook).
   - **Model 3**: Assume $\mathbf{S_1}$ and $\mathbf{S_2}$ are diagonal within $\mathbf{S_1}$ and $\mathbf{S_2}$: $\mathbf{S_1} = diag(\sigma_1)$, $\mathbf{S_2} = diag(\sigma_2)$. This is the Naive Bayes model given in equation (5.24) in the textbook.

   (a) (**10 points**) Write the likelihood function and derive $\mathbf{S_1}$ and $\mathbf{S_2}$ by maximum likelihood estimation of model 2 and model 3.

   (b) (**10 points**) Your program should return and print out the learned parameters $P(C_1), P(C_2)$, μ_1 and μ_2 of each data pair to either terminal or PyCharm console. Your implementation of model 1 and model 2 should return and print out the learned parameters $\mathbf{S_1}, \mathbf{S_2}$. Your implementation of model 3 will return and print out σ_1 and σ_2.

   (c) (**10 points**) For each test set, print out the error rates of each model to either terminal or PyCharm console (three models per each test set). Match each data pair to one of the models and justify your answer. Also, explain the difference in your results in the report.

[1]Instructor: Rui Kuang (kuang@umn.edu). TAs: Tianci Song (song0309@umn.edu) and Ruyuan (wanxx199@umn.edu).

2. In this problem, you will apply dimension reduction and classification on the Optdigits dataset provided in `optdigits_train.txt` and `optdigits_test.txt`.

   (a) (**5 points**) Implement k-Nearest Neighbor (KNN) to classify the Optdigits dataset with $k = \{1, 3, 5, 7\}$. Print out the error rate on the test set for each value of k to either terminal or PyCharm console.

   (b) (**10 points**) Implement your own version of Principal Component Analysis (PCA) and apply it the Optdigits training data. Generate a plot of *proportion of variance* (see Figure 6.4 (b) in the main textbook), and select the minimum number (K) of eigenvectors that explain at least 90% of the variance. Show both the plot and K in the report. Project the training and test data to the K principal components and run KNN on the projected data for $k = \{1, 3, 5, 7\}$. Print out the error rate on the test set for each value of k to either terminal or PyCharm console.

   (c) (**5 points**) Next, project both the training and test data to $\mathbb{R}^2$ using only the first two principal components to plot all samples in the projected space and label some data points with the corresponding digit in 10 different colors for the 10 types of digits for a good visualization (similar to Figure 6.5).

   (d) (**10 points**) Implement your own version of Linear Discriminant Analysis (LDA) and apply it to compute a projection only using the Optdigits training data into L dimensions (L = 2, 4, 9). Run KNN on the projected data for $k = \{1, 3, 5\}$. Print out the error rate on the test set for each combination of k and L to either terminal or PyCharm console. (Hint: numpy.linalg module has function pinv() which can be used to invert singular matrix as an approximation.)

   (e) (**10 points**) Similarly, project both the training and test data to $\mathbb{R}^2$ with the LDA projections and, plot all samples in the projected space and label some data points with the corresponding digit in 10 different colors for the 10 types of digits.

3. In this problem, you will work on dimension reduction and classification on a Faces dataset from the UCI repository[2]. We provided the processed files `face_train_data_960.txt` and `face_test_data_960.txt` with 500 and 124 images, respectively. Each image is of size 30×32 with the pixel values in a row in the files and the last column identifies the labels: 1 (sunglasses), and 0 (open)

[2]https://archive.ics.uci.edu/ml/datasets/CMU+Face+Images

of the image. You can visualize the ith image with the following Python command line:

```
import numpy as np
import matplotlib.pyplot as plt
plt.imshow(np.reshape(img_data, (30, 32)))
```

(a) (**10 points**) Implement PCA and apply it to find the principal components with combined training and test sets. First, visualize the first 5 eigen-faces using a similar command line as above.

(b) (**10 points**) Repeat what you did in question 2 (b), using PCA and KNN on this Faces dataset.

(c) (**10 points**) Use the first $K = \{10, 50, 100\}$ principle components to approximate the first five images of the training set (first row of the data matrix) by projecting the centered data using the first K principal components then "back project" (weighted sum of the components) to the original space and add the mean. For each K, plot the reconstructed image. Explain your observations in the report.
(**Hint:** Read section 6.3 on page 126 and 127 of the textbook for the projection and "back projection" to the original space.)

Instructions

- Solutions to all questions must be included in a report including result explanations, learned parameter values and all error rates and plots.
- All programming questions must be written in Python, no other programming languages will be accepted. And only numpy, scipy and matplotlib can be relied on to implement the algorithm. The code must be able to be executed from either terminal or PyCharm console on the cselabs machines. Each function must take the inputs in the order specified and print/display the required output to either terminal or PyCharm console. For each part, you can submit additional files/functions (as needed) which will be used by the main functions specified below. Put comments in your code so that one can follow the key parts and steps. **Please follow the rules strictly. If we cannot run your code, you will receive no credit.**
- **Question 1:**

  - MultiGaussian(*training_data*: file name of the training data, *testing_data*: file name of the testing data, *Model*: the model number). The function must output the learned parameters and error rates as required in Question 1.

- **Question 2:**
  - myKNN(training_data, test_data, k). The function returns the prediction for the test set.
  - myPCA(data, num_principal_components). The function returns the principal components and the corresponding eigenvalues.
  - myLDA(data, num_principal_components). The function returns the projection matrix and the corresponding eigenvalues.
  - script_2a.py, script_2b.py and script_2c.py Script files that solves question 2 (a), (b), (c), (d) and (e) calling the appropriate functions, do the plots and print values asked.

- **Question 3:**
  - script_3a.py, script_3b.py and script_3c.py Script files that solves question 3 (a), (b) and (c) calling the appropriate functions, do the plots and print values asked.

- For each dataset, rows are the samples and columns are the features with the last column containing the label.

- You can use the *eigh* function in the *linalg* module of *numpy* to calculate eigenvalues and eigenvectors (If you use *eig* function in the *linalg* module of *numpy*, you might have complex numbers in your eigenvalues). To obtain distance between each pair of samples in KNN, you might consider to use *cdist* in the *spatial.distance* module of *scipy*. To visualize the projected data, you can use the *scatter* function in the *pyplot* module of *matplotlib*, and for adding text to corresponding point, you can use either *text* or *annotate* function in the *pyplot* module of *matplotlib*.

Submission

- **Things to submit:**

1. hw2_sol.pdf: A document which contains the report with solutions to all questions.
2. MultiGaussian: Code for Question 1.
3. myKNN.py, myPCA.py, myLDA.py, script_2a.py, script_2b.py, script_2c.py, script_2d.py, script_2e.py Code for Question 2.
4. script_3a.py, script_3b.py, script_3c.py Code for Question 3.
5. Any other files, except the data, which are necessary for your code.

- **Submit**: All materials must be zipped in one file, and submitted electronically via canvas.