INF8953CE(Fall 2020) : Machine Learning — Given on: Oct 04, 10:00 pm

Programming Assignment #2 — Due on : Oct 24, 10:00 pm

- This is an individual assignment. However, you are allowed to discuss the problems with other students in the class. But you should write your own code and report.
- If you have any discussion with others, you should acknowledge the discussion in your report by mentioning their name.
- You have to submit the `pdf` copy of the report on gradescope before the deadline. If you handwrite your solutions, you need to scan the pages, merge them to a single `pdf` file and submit. Mark page 1 for the outline item *'Verbosity'* on gradescope.
- When asked to report the parameters, save the corresponding parameters in a text file (`.txt`) with the following template for the name: `Assignment2_<Matricule>_<section #>_<sub-question #>_<sub-sub-question #>`. Submit all the text files and codes compressed to a single file `<your-matricule-ID>.zip` on **Moodle**.
  Note: gradescope doesn't accept .zip.
- Be precise with your explanations in the report. Unnecessary verbosity will be penalized.
- You are free to use libraries with general utilities, such as matplotlib, numpy and scipy for python. However, you should implement the algorithms yourself, which means **you should not use pre-existing implementations of the algorithms as found in SciKit learn, Tensorflow, etc.**!
- If you have questions regarding the assignment, you can ask for clarifications in Piazza.
- For late submission policies, refer `https://chandar-lab.github.io/INF8245E/logistics.html`

1 Linear Classification and Nearest Neighbor Classification

1. You will use a synthetic data set for the classification task that you'll generate yourself. Generate two classes with 20 features each. Each class is given by a multivariate Gaussian distribution, with both classes sharing the same covariance matrix. You are provided with the mean vectors (DS1-m0 for mean vector of negative class and DS1-m1 for mean vector of positive class) and the covariance matrix (DS1-cov).

   Generate 2000 examples for each class, and label the data to be positive if they came from the Gaussian with mean m1 and negative if they came from the Gaussian with mean m0. Randomly pick (without replacement) 20% of each class (i.e., 400 data points per class) as test set, 20% of each class (i.e., 400 data points per class) as validation set set and train the classifiers on the remaining 60% data. When you report performance

results, it should be on the test set. Call this dataset as DS1, and submit it with your code. Follow the instructions from Assignment 1 for data submission format.

2. We first consider the GDA model as seen in class: given the class variable, the data are assumed to be Gaussians with different means for different classes but with the same covariance matrix. This model can formally be specified as follows:

$$Y \sim \text{Bernoulli}(\pi), \quad X \,|\, Y = j \sim \mathcal{N}(\mu_j, \Sigma).$$

   Estimate the parameters of the GDA model using the maximum likelihood approach.

   (a) For DS1, report the best fit accuracy achieved by the classifier.

   (b) Report the coefficients learnt.

3. For DS1, use k-NN to learn a classifier. Repeat the experiment for different values of k and report the performance for each value. We will compare this non-linear classifier to the linear approach, and find out how powerful linear classifiers can be.

   (a) Does this classifier performs better than GDA or worse? Are there particular values of k which perform better? Why does this happen ? Use validation accuracy for model selection.

   (b) Report the best fit accuracy achieved by this classifier.

4. Now instead of having a single multivariate Gaussian distribution per class, each class is going to be generated by a mixture of 3 Gaussians. For each class, we'll define 3 Gaussians, with the first Gaussian of the first class sharing the covariance matrix with the first Gaussian of the second class and so on. For both the classes, fix the mixture probability as (0.1,0.42,0.48) i.e. the sample has arisen from first Gaussian with probability 0.1, second with probability 0.42 and so on. Mean for three Gaussians in the positive class are given as DS2-c1-m1, DS2-c1-m2, DS2-c1-m3. Mean for three Gaussians in the negative class are gives as DS2-c2-m1, DS2-c2-m2, DS2-c2-m3. Corresponding 3 covariance matrices are given as DS2-cov-1, DS2-cov-2 and DS2-cov-3. Now sample from this distribution and generate the dataset similar to question 1. Call this dataset as DS2, and submit it with your code. Follow the instructions from Assignment 1 for data submission format.

5. Now perform the experiments in questions 2 and 3 again, but now using DS2.

   1. Estimate the parameters of the GDA model using the maximum likelihood approach.

      (a) For DS2, report the best fit accuracy achieved by the classifier.

      (b) Report the coefficients learnt.

   2. Does k-NN classifier perform better than GDA or worse? Are there particular values of k which perform better? Why does this happen ?

   3. Report the best fit accuracy achieved by this classifier.

6. Comment on any similarities and differences between the performance of both classifiers on datasets DS1 and DS2?

2 MNIST Handwritten Digits Classification

In this section, we will use the MNIST handwritten digits classification dataset. The task is to classify a given image of a handwritten digit into one of 10 classes representing integer values from 0 to 9, inclusively. The dataset consists of 60,000 training data points and 10,000 test data points. The datapoints are represented as 28×28 pixel grayscale images of handwritten single digits between 0 and 9.

To load the dataset, using the following code. However, remember that for any other implementation you should not use tensorflow/sklearn/keras as mentioned in the instructions.

```
#Import Keras
from tensorflow import keras
# Loading the MNIST dataset
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
```

Use the first 50,000 examples in (x_train, y_train) as your training data and use the last 10,000 examples in (x_train, y_train) as your validation data. The examples are represented as pixel matrices of size (28,28). You should first flatten the image matrix to 784 features before passing it to the classifiers. The pixel values are in the range of (0,255). You should normalize the features by dividing them by 255.

1. First we will consider Gaussian Naïve Bayes (GNB) model for this task. This is same as the GDA model from the previous question but with two modifications: The covariance matrices are not shared and they are diagonal (Naïve Bayes assumption).

   (a) Write down the equations for computing the mean and diagonal covariance matrices for the class conditional densities and also the prior class probabilities using the maximum likelihood approach.

   (b) Estimate the parameters of the GNB model from the dataset. Report the best fit accuracy achieved.

2. Use k-NN to learn a classifier. Repeat the experiment for different values of k and report the performance for each value.

   (a) Are there particular values of k which perform better? Why does this happen ? Use validation accuracy for model selection.

   (b) Report the best fit accuracy achieved by this classifier.

3. Compare the performance of GNB and k-NN for this MNIST classification. If one performs better over the other, explain why.

Instruction for code submission

1. Submit a single zipped folder with your matricule id as the name of the folder. For example if your matricule ID is 12345678, then the submission should be 12345678.zip.
2. You can only use python and you must submit your solution as a jupyter notebook.
3. Make sure all the data files needed to run your code is within the folder and loaded with relative path. We should be able to run your code without making any modifications.

Instruction for report submission

1. Do not include your code in the report!