Homework #4

Assigned: 03/29/2021
Due: 04/13/2021 (11:59 PM on CANVAS)

A. Theory Problems

1. **Logistic Regression as Maximum Likelihood:** As discussed in class, logistic regression can be derived via a probabilistic perspective. Specifically, the model can be defined by assuming a conditional distribution $p(y = 1 \,|\boldsymbol{X}; \boldsymbol{\theta}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T X}}$. Here, $\boldsymbol{X}$ is the input feature vector, $\boldsymbol{\theta}$ is the parameter vector, and y is the output class.

   Show that maximizing the likelihood of the N observed class labels $(y_1, y_2, \dots y_N)$ given the inputs $(\boldsymbol{X}_1, \boldsymbol{X}_2, \dots \boldsymbol{X}_N)$ is the same as minimizing the log-loss, $\sum_{n=1}^{N} -y^{(n)} log(\hat{y}^{(n)}) \; - (1 - y^{(n)}) log(1 - \hat{y}^{(n)})$, where $\hat{y}^{(n)} = \frac{1}{1+e^{-\boldsymbol{\theta}^T \boldsymbol{X}_n}}$.

   *Note**: The likelihood is given by $p(y_1, y_2, \dots y_N | \boldsymbol{X_1}, \boldsymbol{X_2} \dots \boldsymbol{X_N}; \boldsymbol{\theta}) = \prod_{n=1}^{N} p(y_n \,|\boldsymbol{X_n}; \boldsymbol{\theta}\,)$.

2. **Gaussian Naïve Bayes and QDA:** Show that QDA (quadratic discriminative analysis) is equivalent to a Gaussian Naïve Bayes classifier if the covariance matrix, $\boldsymbol{\Sigma}$, is a diagonal matrix and the input features are conditionally independent in each class.

   As a reminder, a diagonal matrix has nonzero values only on the main diagonal:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_{\boldsymbol{11}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Sigma_{\boldsymbol{MM}} \end{pmatrix}$$

B. Coding Problems

3. **Stroke Prediction:** In this problem you will build a binary classifier for predicting if a patient will get a stroke. The dataset can be found in *'healthcare-dataset-stroke-data.csv'*. There are 10 input features which are:

   1) gender: "Male", "Female" or "Other"
   2) age: age of the patient
   3) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension
   4) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
   5) ever_married: "No" or "Yes"
   6) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
   7) Residence_type: "Rural" or "Urban"
   8) avg_glucose_level: average glucose level in blood
   9) bmi: body mass index
   10) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*

   The output is given in the 11th column and is:

   11) stroke: 1 if the patient had a stroke or 0 if not

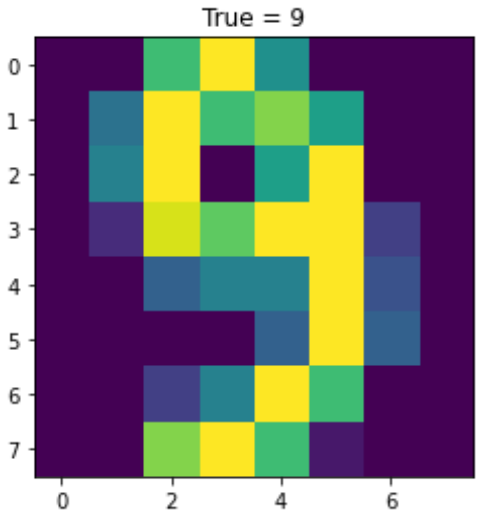
Split the data using a 70-30 split of training to testing data.

a) Using the training data, build a random forest classifier using 100 decision trees.

b) Show a confusion matrix for both the training and testing data and report on the testing accuracy of the model.

Note: Some of the bmi values are unknown and are labeled as N/A, feel free to remove these cases as a pre-processing step.*

4. **Digit Classification Classifier Comparison:** For this problem, you will repeat problem 3 from HW 1 but using several classifiers. As a reminder, an example digit (sampled on an 8x8 pixel grid) looks like the following:



The testing data arrays for this problem contain 1257 labeled cases and are given in the Python .npy files as 'X_Train.npy' and 'y_train.npy' (on CANVAS). The X_train array has size 1257x64. Each row of the array corresponds to one image (as the example above), however, the 8x8 grid of pixels has been reordered to 64x1. *Note* you can view the n-th training example using matplotlib.pyplot.imshow(X_train[n,:].reshape([8,8]))*. The array y_train contains the 1257 integers between 0 and 9 which correspond accordingly to each training example.

a) Build the following classifier models on your testing data:
   i. Perceptron (same as HW 1)
   ii. Logistic Regression
   iii. Gaussian Naïve Bayes

b) For each of the models from (a) show a confusion matrix based on the testing data. Which model has the highest accuracy?

c) For each of the models from (a) determine the time (in seconds) it takes to train each model. Which model is the fastest to train and which one is the slowest?