STAT 6340 (Statistical and Machine Learning, Spring 2021)

Mini Project 1

Instructions:

- Due date: Feb 10, 2020.
- Total points = 20 (plus 5 bonus points)
- Submit a typed report.
- It is OK to discuss the project with other students in the class, but each student must write their own code and answers. If your submitted report (including code and answer) is similar (either partially or fully) to someone else's, this will be considered evidence of academic dishonesty, and you will referred to appropriate university authorities.
- Do a good job.
- You must use the following template for your report:

  Mini Project #
  Name
  Section 1. Answers to the specific questions asked
  Section 2: R code. Your code must be annotated. No points may be given if a brief look at the code does not tell us what it is doing.
- Section 1 of the report must be limited to two pages. Also, only those output should be provided in this section that are referred to in the report.

---

1. (10 points) Consider the training and test data posted on eLearning in the files `1-tranining-data.csv` and `1-test-data.csv`, respectively, for a classification problem with two classes.

   (a) Fit KNN with $K = 1, 6, \dots, 200$.

   (b) Plot training and test error rates against K. Explain what you observe. Is it consistent with what you expect from the class?

   (c) What is the optimal value of K? What are the training and test error rates associated with the optimal K?

   (d) Make a plot of the training data that also shows the decision boundary for the optimal K. Comment on what you observe. Does the decision boundary seem sensible?

2. (10 points) Read about the problem of image classification at `http://cs231n.github.io/classification/`. This website also discusses coding, which is to be ignored. Pay particular attention to the CIFAR-10 dataset. You can read a bit more about it at `https://www.cs.toronto.edu/~kriz/cifar.html`. Upon doing so, you would understand that each image in this dataset is 32 pixels wide, 32 pixels tall, and has 3 color channels (red, blue, and green; or RGB). Thus, each image is a $32 \times 32 \times 3$ (3D) array of numbers. Each number represents an intensity that lies between 0 and 255 (dark to light). These image features, totaling $32 * 32 * 3 = 3072$, will serve as predictors. The response is the class of the object shown in the image. There are 10 classes. Thus, in essence, we have a classification problem with $C = 10$ response classes, $p = 3072$ predictors; and the dataset consist of $n = 50000$ observations in the training set and 10000 observations in test set.

   Next, *closely* follow the instructions at `https://keras.rstudio.com` to download and install the `keras` package (that contains the data) together with its `TensorFlow` backend. Then, read the data into `R` and pre-process as follows:

```
library(keras)

cifar <- dataset_cifar10()
str(cifar)

x.train <- cifar$train$x
y.train <- cifar$train$y
x.test <- cifar$test$x
y.test <- cifar$test$y

# reshape the images as vectors (column-wise)
# (aka flatten or convert into wide format)
# (for row-wise reshaping, see ?array_reshape)
dim(x.train) <- c(nrow(x.train), 32*32*3) # 50000 x 3072
dim(x.test) <-  c(nrow(x.test), 32*32*3) # 50000 x 3072

# rescale the x to lie between 0 and 1
x.train <- x.train/255
x.test <- x.test/255

# categorize the response
y.train <- as.factor(y.train)
y.test <- as.factor(y.test)

# randomly sample 1/100 of test data to reduce computing time

set.seed(2021)
id.test <- sample(1:10000, 100)

x.test <- x.test[id.test,]
y.test <- y.test[id.test]
```

Use the entire training set and the resampled test obtained above to answer the following questions. Note that we work with only 1% of the test set to reduce the computing time.

(a) Fit KNN with $K = 50, 100, 200, 300, 400$ and examine the test error rates. (Feel free to explore additional values of K.)

(b) For the best value of K (among the ones you have explored), examine the confusion matrix and comment on your findings.

(c) Briefly explain (in no more than one small paragraph) why using KNN for image classification may not be a good idea.

(You may additionally explore working with the entire test data and how to view the images in `R`. For the latter, see, e.g., `https://rdrr.io/a/github/jlmelville/snedata/src/R/cifar.R`. This is optional, however. Also, please share with class any other useful resources that you discover.)

3. (5 bonus points) Consider the following general model for the training data (Y_i, x_i), $i = 1, \ldots, n$ in a learning problem:

$$Y_i = f(x_i) + \epsilon_i,$$

where f is the true mean response function; and the random errors ϵ_i have mean zero, variance σ^2, and are mutually independent. We discussed this model in the class. Let $\hat{f}$ be the estimator of f

obtained from the training data. Further, let (x_0, Y_0) be a test observation. In other words, x_0 is a future value of x at which we want to predict Y and Y_0 is the corresponding true value of Y. The test observation follows the same model as the training data, i.e.,

$$Y_0 = f(x_0) + \epsilon_0,$$

where ϵ_0 has the same distribution as the ϵ_i for the training data but ϵ_0 is independent of the ϵ_i. Let $\hat{Y}_0 = \hat{f}(x_0)$ be the predicted value of Y_0.

(a) Show that $\text{MSE}\{\hat{f}(x_0)\} = (\text{Bias}\{\hat{f}(x_0)\})^2 + \text{var}\{\hat{f}(x_0)\}$.

(b) Show that $E(\hat{Y}_0 - Y_0)^2 = (\text{Bias}\{\hat{f}(x_0)\})^2 + \text{var}\{\hat{f}(x_0)\} + \sigma^2$.