Homework 1

You should write up a solution to the conceptual problems. You should program solutions to the implementation problems. These programs should follow good programming practices as laid out in the Resources & Information section of the course.

You can combine these two solution files into one file and submit that or submit two separate files. I would highly recommend using R markdown, quarto, or Jupyter notebooks to do your assignments. With these you can write all of your answers into one file easily!

If you have any questions about this, please just let me know!

# Conceptual Problems

## Problem 1

Consider the supervised learning setup. Recall, we have observations and a model given by

We’ll consider doing a **regression task**. Let’s assume that the errors (’s) are independent and identically distributed, following a Normal distribution with mean zero and variance . Mathematically, we write this as

We’ll assume that the predictors, , are fixed, known values.

Suppose we apply the KNN model with a pre-specified value of and a fixed point of interest. We have an interest in the properties of from this model.

1. Compute the variance of (we wrote down a form for this quantity in the week 2 notes).

Hint: Recall from previous coursework - If we have independent observations (call them ) and are interested in the variability of a mean (call it ), we know

1. Using your answer from (a), what happens to the variance of our estimate as we increase ? What happens as we decrease ? Explain why this makes sense.
2. Suppose we run two separate KNN regressions on the same data set. One model uses and one model uses . For each model we compute the training MSE and the test MSE on an independent test set.
   1. Which model () will have lower training MSE? Explain.
   2. Do we know which one will have lower test MSE? Explain.

## Book Problems

Complete the following problems from the Introduction to Statistical Learning with R book (I’m not sure if the problems are in the same order in the python book so use the R book to identify which problems to do).

Section 2.4

* Book Problem 1
* Book Problem 2
* Book Problem 5
* Book Problem 7. Note: For the KNN classification model, the predicted classification is given by the class the occurs the most for the K neighbors.

# Implementation Problems

## Problem 2

Consider the Boston housing data from the ISLR2 package (a similar package exists in python). Suppose we want to build a prediction model for the medv variable. We did this in our notes using the lstat variable as our predictor.

1. Split the data into a training and test set using a 70/30 split. Using SRSWOR.
2. Consider three separate simple linear regression models: one using age as the predictor, one using rm as the predictor, and one using ptratio as the predictor.
   * Compare these models on the training data set only using 10 fold CV and RMSE as your metric. Note: CV isn’t needed to tune a hyperparameter here but we can still use it to choose between our three candidate SLR models!
   * With your best model, fit it using the entire training data set.
3. Consider three separate KNN models, each using one of the three predictors above.
   * Find the optimal for each of these models using 10 fold CV on the training data only
   * Compare the CV error for the three tuned models and select a best model.
   * With your best model, fit it using the entire training data set.
4. Predict on your test set using both of your ‘best’ models. Compute the test set RMSEs. Which model is the overall best?

## Problem 3

It is important to really understand cross-validation as it is used in many situation. In this problem we’ll do our own basic cross-validation.

Consider the iris data set built into R. This data set can be read into python via the sklearn package using the following code:

from sklearn import datasets  
  
# Load the dataset  
iris = datasets.load\_iris()

1. Randomly split the data into five distinct folds.
2. Using the first four folds, fit an SLR model with Sepal.Length as the predictor and Petal.Length as the response. Predict on the fifth fold and compute the MSE on those predictions.
3. Repeat this process, using the fourth fold as the test fold. Then again, with the third fold as the test fold, etc.
4. Combine the MSE across the folds (averaging them is fine here) to create a CV error.

For this problem, I am not at all concerned with your program’s quality/efficiency. The only requirements are that you are manually doing the process rather than using some package to implement the CV algorithm.