STA 314H1S: Problem Set 1

Except for question 6, the questions on this assignment are practice for the quiz on Friday, and are not to be handed in.
For question 6, please bring a printout of your output together with the source code. You will be asked to submit this printout at the end of the quiz.

1. (*Review of MLE*) For each of the following distributions, derive a general expression for the Maximum Likelihood Estimator (MLE); don't bother with the second derivative test. Then use the data to calculate a numerical estimate.

   (a) $p(x) = \theta(1-\theta)^x$ for $x = 0, 1, \ldots$, where $0 < \theta < 1$. Data: `4, 0, 1, 0, 1, 3, 2, 16, 3, 0, 4, 3, 6, 16, 0, 0, 1, 1, 6, 10.` Answer: 0.2061856

   (b) $f(x) = \frac{\alpha}{x^{\alpha+1}}$ for $x > 1$, where $\alpha > 0$. Data: `1.37, 2.89, 1.52, 1.77, 1.04, 2.71, 1.19, 1.13, 15.66, 1.43` Answer: 1.469102

   (c) $f(x) = \frac{\tau}{\sqrt{2\pi}} e^{-\frac{\tau^2 x^2}{2}}$, for x real, where $\tau > 0$. Data: `1.45, 0.47, -3.33, 0.82, -1.59, -0.37, -1.56, -0.20` Answer: 0.6451059

   (d) $f(x) = \frac{1}{\theta} e^{-x/\theta}$ for $x > 0$, where $\theta > 0$. Data: `0.28, 1.72, 0.08, 1.22, 1.86, 0.62, 2.44, 2.48, 2.96` Answer: 1.517778

2. (*Review of Linear Algebra*) The following questions are for you to review some basic linear algebra that will be relevant later in this course.

   (a) Which statement is true?

      i. $(\mathbf{B}+\mathbf{C})\mathbf{A} = \mathbf{AB} + \mathbf{AC}$

      ii. $(\mathbf{B}+\mathbf{C})\mathbf{A} = \mathbf{BA} + \mathbf{CA}$

      iii. $(\mathbf{AB})^\top = \mathbf{A}^\top \mathbf{B}^\top$

      iv. $(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top$

   (b) Recall that an inverse of the square matrix $\mathbf{A}$ (denoted $\mathbf{A}^{-1}$) is defined by $\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$. Prove that inverses are unique, as follows. Let $\mathbf{B}$ and $\mathbf{C}$ both be inverses of $\mathbf{A}$. Show that $\mathbf{B} = \mathbf{C}$.

   (c) Let $\mathbf{A}$ be a square matrix with the determinant of $\mathbf{A}$ (denoted $|\mathbf{A}|$) equal to zero. What does this tell you about $\mathbf{A}^{-1}$? No proof is required here.

   (d) Suppose that the square matrices $\mathbf{A}$ and $\mathbf{B}$ both have inverses. Using the definition of an inverse, prove that $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$. Becuause you are using the definition, you have two things to show.

   (e) Let $\mathbf{X}$ be an n by p matrix with $n \neq p$. Why is it incorrect to say that $(\mathbf{X}^\top\mathbf{X})^{-1} = \mathbf{X}^{-1}\mathbf{X}^{\top -1}$?

   (f) Let $\mathbf{A}$ be a non-singular square matrix. Prove $(\mathbf{A}^\top)^{-1} = (\mathbf{A}^{-1})^\top$.

(g) Using Question 2f, prove that the if the inverse of a symmetric matrix exists, it is also symmetric.

(h) Let $\mathbf{a}$ be an $n \times 1$ matrix of real constants. How do you know $\mathbf{a}^\top \mathbf{a} \geq 0$?

3. (*Multiple Linear Regression*) Assume the following multiple linear regression model

$$y_i = \mathbf{x}_i^T \boldsymbol{\beta} + \boldsymbol{\epsilon}_i,$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$. The feature $\mathbf{x}_i \in \mathbb{R}^p$ has mean $\mathbb{E}(\mathbf{x}_i) = \mathbf{0} \in \mathbb{R}^p$ and variance $\text{Var}(\mathbf{x}_i) = \mathbf{I} \in \mathbb{R}^{p \times p}$, being independent of the error ϵ_i. Assume there are n pairs of independent observations $(y_i, \mathbf{x}_i)_{i=1}^n$, where $\boldsymbol{y} \in \mathbb{R}^n$ denotes the outcome vector and $\mathbf{X} \in \mathbb{R}^{n \times p}$ denotes the design matrix (i.e. the ith row of the $\mathbf{X}$ is $\mathbf{x}_i^T$).

(a) Compute $\mathbb{E}(y_i|\mathbf{x}_i)$ and $\text{Var}(y_i|\mathbf{x}_i)$, as well as $\mathbb{E}(y_i)$ and $\text{Var}(y_i)$.

(b) Write out the conditional distribution of $\boldsymbol{y}$ given all features $\mathbf{X}$. Can you determine the marginal distribution of $\boldsymbol{y}$ based on the information above?

(c) Derive the estimate $\hat{\beta}$ using Least-Square and MLE.

(d) Using your answer above for $\hat{\beta}$, write out the prediction $\hat{y}$ for the training data $\mathbf{X}$, and $\hat{y}_{new}$ for at a new feature $\mathbf{x}_{new}$.

(e) Estimate the training MSE with the empirical MSE (conditional on $\mathbf{X}$). Does this quantity remind you of the MLE of a certain parameter, which you learned from STA302? (*hint:* Recall from STA302, the residual vector $\hat{\epsilon} = \mathbf{My}$, where the residual maker matrix $\mathbf{M} = \mathbf{I}_n - \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$).

(f) Derive the formula of training MSE (i.e. the expected value of your answer in (e)), you can use without proof that for any random vector $\mathbf{z}$ and symmetric matrix $\mathbf{A}$ if $\mathbb{E}(\mathbf{z}) = \mu$ and $\text{Var}(\mathbf{z}) = \boldsymbol{\Sigma}$, then $\mathbb{E}(\mathbf{z}^T\mathbf{A}\mathbf{z}) = \mu^T\mathbf{A}\mu + tr(\mathbf{A}\boldsymbol{\Sigma})$).

(g) Now do question (f) for the testing MSE. For simplicity, assume without the loss of generality that there is one single prediction (i.e. $\hat{y}_{new} \in \mathbb{R}$ and $\mathbf{x}_{new} \in \mathbb{R}^{p \times 1}$).

(h) Now suppose the regression parameter β is known, redo questions (d-g).

(i) Compare your answers from (h) with (d-g). Will knowing β improve your testing MSE? What about your training MSE?

(j) Suppose your collaborator wants you to derive a better prediction $f(x_{new})$ with testing MSE smaller than 1. Is it possible? (Note: σ is unknown now.)

4. (*Bias-Variance Tradeoff*) Assume the data $\{(y_i, x_i)\}_{i=1}^n$ are independently generated from the following model

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

where $\epsilon_i \overset{iid}{\sim} N(0, \sigma^2)$ and f is some unknown function. The target is to predict the outcome y_0 at a new location x_0.

(a) What is $\mathbb{E}(y_i|x_i)$ and $\text{Var}(y_i|x_i)$. Can we say the outcomes $\{y_i\}_{i=1}^n$ are i.i.d distributed given $\{x_i\}_{i=1}^n$?

(b) Since you don't know what f is, you decide to fit a simple linear regression

$$y_i = \beta x_i + \epsilon_i.$$

This is a misspecified model, and you obtain your prediction $\hat{y}(x_0)$ from this model. Write out $\hat{y}(x_0)$, as a function of $\{(y_i, x_i)\}_{i=1}^n$ and x_0.

(c) Assume the training outcome $\{(y_i)\}_{i=1}^n$ is not observed yet (hence random), compute the testing MSE of $\hat{y}(x_0)$. Clearly decompose the MSE into bias, variance, and Bayes error (*Hint: your MSE should be a function of* $f(x_0)$).

(d) Assume the entire training data $\{(y_i, x_i)\}_{i=1}^n$ is already observed (hence fixed), compute the testing MSE of $\hat{y}(x_0)$. Clearly decompose the MSE into bias, variance, and Bayes error (*Hint: your MSE should be a function of* $\{y_i\}_{i=1}^n$).

(e) After you fit the simple linear model, you notice both the training and testing MSE are very high. So you want to consider a more flexible model instead

$$y_i = \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \epsilon_i.$$

What will happen to your training MSE? Can you make the same conclusion for the testing MSE? Also comment on how each component of the testing MSE changes.

5. (*KNN regression*) Consider the same generating model as in question 4, but now instead of using linear regression, you decide to make prediction $\hat{y}(x_0)$ using KNN method. Recall $d_i(x_0) = |x_i - x_0|$ and $\mathcal{N}_0$ is the set of the indices of the nearest K points.

(a) Recall the type of KNN regression discussed in the lecture, rewrite the prediction $\hat{y}(x_0) = \sum_{i=1}^n w_i(x_0) y_i$, for a set of (non-negative) weights $\{w_i(x_0)\}_{i=1}^n$ such that $\sum_{i=1}^n w_i(x_0) = 1$ (*Hint: consider indicator functions*).

(b) Assume all the features $\{x_i\}_{i=1}^n$ are unique, when $K = 1$, what is the empirical training MSE? What if $K = n$?

(c) Suppose in your training data, each observed feature x_i is ranged between 0 to 1. Explain why the predicted $\hat{y}(x_0)$ will be particularly unreliable for $x_0 > 1$ or $x_0 < 0$ (*Hint: what will be the function $\hat{y}(x_0)$ for $x_0 > 1$?*)

(d) One disadvantage of the type of KNN regression is that the prediction $\hat{y}(x_0)$ as a function of x_0 is rough. Specifically, the function $\hat{y}(x_0)$ is discontinuous in most cases. Comment on why this is true, and for which choice of K the resulting $\hat{y}(x_0)$ will be continuous?

(e) Using the representation in part(a), how can you fix the problem in part(d)? (*Hint: think about modifying the weight function $w_i(x_0)$.*)

6. (*Computation*) Consider the same generating model as in question 4, with the true function $f(x) = 2\sin(x)\log(x)$, and variance parameter $\sigma = 1$. In this question, we will assume $x_i \overset{iid}{\sim} \mathrm{Unif}[1, 10]$.

   (a) Generate $n = 200$ observations from this model, and randomly separate half of the data as training data, the other half of data as testing data. Produce a scatterplot for each of them.
   Some starter code to help you get started:

```r
library(tidyverse)
set.seed(12345)
n <- 200
# define the function
f <- function(x) {#TODO}
# generate data
x <- #TODO
y <- #TODO
# data separation
training_data <- #TODO
testing_data <- #TODO
# plot
plot(y ~ x, data = training_data, main = "training")
plot(y ~ x, data = testing_data, main = "testing")
```

   (b) Now, fit the following linear regression model:

$$y_i = \beta_1 \sin(x_i) + \beta_2 \sin(x_i/2) + \epsilon_i,$$

   and plot its prediction of the training/testing data.
   Some starter code to help you get started:

```r
# fit the model
mod1 <- lm(formula = #TODO, data = #TODO)
# prediction
pred_training <- as.numeric(predict(object = mod1, newdata = #TODO))
pred_testing <- as.numeric(predict(object = mod1, newdata = #TODO))
# plot
plot(pred_training ~ training_data$x, main = "training")
plot(pred_testing ~ testing_data$x, main = "testing")
```

   (c) For the model above, compute the empirical training/testing MSE. Which one do you expect to be larger? Is one *always* larger than the other?

   (d) In practice, it is often hard to evaluate the specific value of bias and variance of $\hat{y}(x_0)$. But in this question, we can do it empirically by repeatedly simulating the training data from the true model. Estimate the bias and variance component of $\hat{y}(x_0)$ when $x_0 = 3$ in this way.
   Some starter code to help you get started:

```r
set.seed(12345)
yhat <- c()
# The location to predict
x0 <- 3
data_to_predict <- data.frame(x = x0)
# Let's repeat the previous procedure for B times
B = 500

for (i in 1:B) {
  # Simulate new training outcomes
  y_new <- #TODO
  training_data_new <- data.frame(x = training_data$x, y = y_new)
  # Fit new model
  modnew <- #TODO
  # Obtain the prediction
  yhat[i] <- #TODO
}

# Variance of the prediction
var(yhat)

# Bias of the prediction
mean(yhat - f(x0))
```

(e) Now, redo part (b-c) using a KNN model with $K = 3$.
Some starter code to help you get started:

```r
library(tidyverse)
knn_regression <- function(k, training, testing){
  result <- numeric(length = nrow(testing))
  for (i in 1:length(result)) {
    x0 <- as.numeric(testing[i, , drop = F])
    distance_vec <- #TODO
    N0 <- #TODO
    result[i] <- #TODO
  }
  result
}
knn_prediction_training <- knn_regression(k = 3, training =
    training_data, testing = training_data[,1,drop = F])
knn_prediction_testing <- knn_regression(k = 3, training =
    training_data, testing = testing_data[,1,drop = F])
plot(knn_prediction_training ~ training_data$x, main = "training"
    )
plot(knn_prediction_testing ~ testing_data$x, main = "testing")
training_MSE <- mean((knn_prediction_training - training_data$y)
    ^2)
training_MSE
testing_MSE <- mean((knn_prediction_testing - testing_data$y)^2)
testing_MSE
```

(f) Plot the training and testing MSE for the KNN method, for K from 1 to 50.

Some starter code to help you get started:

```
kvec <- #TODO
training_MSE_vec <- c()
testing_MSE_vec <- c()
for (i in 1:length(kvec)) {
  knn_prediction_training <- #TODO
  knn_prediction_testing <- #TODO
  training_MSE <- #TODO
  training_MSE_vec[i] <- training_MSE
  testing_MSE <- #TODO
  testing_MSE_vec[i] <- testing_MSE
}
plot(training_MSE_vec~kvec, type = 'o', col = "red")
lines(testing_MSE_vec~kvec, type = 'o', col = "blue")
```