Assaf Bitton

Q1

The file `valuation.csv` (available on Brightspace[1]) contains market historical data of real estate valuation collected from Sindian Dist., New Taipei City, Taiwan. Our goal is to predict the per-unit-area house price using several important factors that can potentially affect buyers' decision-making.

Features in the dataset:
X1: the transaction date (for example, 2013.250=2013 March, 2013.500=2013 June, etc.)
X2: the house age (unit: year)
X3: the distance to the nearest MRT station (unit: meter)
X4: the number of convenience stores in the living circle on foot (integer)
X5: the geographic coordinate, latitude. (unit: degree)
X6: the geographic coordinate, longitude. (unit: degree)
Y: house price of unit area (10000 New Taiwan Dollar/Ping, where Ping is a local unit, 1 Ping = 3.3 square meters)

a. Load the dataset into R from `valuation.csv`. Split the dataset into training data (first 200 rows) and test data (the rest rows). Name the training and test data `valuation_train` and `valuation_test`, respectively. Where is the 11th house in the test data located (answer with the latitude and longitude)?

   Solution:

```
# Q1.a
#Load dataset into R from 'valuation.csv'
valuation <- read.csv("C:/Users/Lizi/Downloads/valuation.csv")

#Split the dataset into training data and test data
valuation_train <- valuation[1:200, ]
valuation_test <- valuation[-(1:200), ]

#Check to see if observations line up with excel
head(valuation_train)

##   No       X1   X2         X3 X4       X5       X6    Y
## 1  1 2012.917 32.0   84.87882 10 24.98298 121.5402 37.9
## 2  2 2012.917 19.5  306.59470  9 24.98034 121.5395 42.2
## 3  3 2013.583 13.3  561.98450  5 24.98746 121.5439 47.3
## 4  4 2013.500 13.3  561.98450  5 24.98746 121.5439 54.8
## 5  5 2012.833  5.0  390.56840  5 24.97937 121.5425 43.1
## 6  6 2012.667  7.1 2175.03000  3 24.96305 121.5125 32.1
```

[1] Original source https://archive.ics.uci.edu/dataset/477/real+estate+valuation+data+set

```r
tail(valuation_train)
```

```
##      No       X1   X2        X3 X4       X5       X6    Y
## 195 195 2013.500 15.2 3771.8950  0 24.93363 121.5116 29.3
## 196 196 2013.333 15.2  461.1016  5 24.95425 121.5399 34.6
## 197 197 2013.000 22.8  707.9067  2 24.98100 121.5471 36.6
## 198 198 2013.250 34.4  126.7286  8 24.96881 121.5409 48.2
## 199 199 2013.083 34.0  157.6052  7 24.96628 121.5420 39.1
## 200 200 2013.417 18.2  451.6419  8 24.96945 121.5449 31.6
```

```r
head(valuation_test)
```

```
##      No       X1   X2        X3 X4       X5       X6    Y
## 201 201 2013.417 17.4  995.7554  0 24.96305 121.5491 25.5
## 202 202 2013.417 13.1  561.9845  5 24.98746 121.5439 45.9
## 203 203 2012.917 38.3  642.6985  3 24.97559 121.5371 31.5
## 204 204 2012.667 15.6  289.3248  5 24.98203 121.5435 46.1
## 205 205 2013.000 18.0 1414.8370  1 24.95182 121.5489 26.6
## 206 206 2013.083 12.8 1449.7220  3 24.97289 121.5173 21.4
```

```r
tail(valuation_test)
```

```
##      No       X1   X2         X3 X4       X5       X6    Y
## 409 409 2013.417 18.5 2175.74400  3 24.96330 121.5124 28.1
## 410 410 2013.000 13.7 4082.01500  0 24.94155 121.5038 15.4
## 411 411 2012.667  5.6   90.45606  9 24.97433 121.5431 50.0
## 412 412 2013.250 18.8  390.96960  7 24.97923 121.5399 40.6
## 413 413 2013.000  8.1  104.81010  5 24.96674 121.5407 52.5
## 414 414 2013.500  6.5   90.45606  9 24.97433 121.5431 63.9
```

```r
#11th house location in test data
valuation_test[11, c("X5","X6")]
```

```
##           X5       X6
## 211 24.97937 121.5425
```

- The **11th house** in the test data is located at **latitude, longitude = (24.97937, 121.5425).**

b. Use the training data `valuation_train` to fit a simple linear regression model of Y on X2. Interpret the coefficients β_0 (intercept) and β_1 (slope) in terms of the real context. Compute R^2, training MSE and test MSE.

   Solution:

```r
#Q1.b
#Fit a simple linear regression of model 'Y' on 'X2'
Y_X2 <- lm(Y~X2, data = valuation_train)

#interpret the coefficients Beta0 and Beta1
summary(Y_X2)
```

```
##
## Call:
## lm(formula = Y ~ X2, data = valuation_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.859 -11.634   2.028   8.936  31.496
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 44.46714    1.70482  26.083  < 2e-16 ***
## X2          -0.33841    0.08021  -4.219 3.73e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.97 on 198 degrees of freedom
## Multiple R-squared:  0.08249,    Adjusted R-squared:  0.07786
## F-statistic:  17.8 on 1 and 198 DF,  p-value: 3.731e-05
```

- The **intercept** of β_0 **= 44.46714** means that a house with an **X2** value or ***"house age (unit: year)"*** of **0** is expected (predicted) to have a **unit price (Ping)** of **~44.47** . Since the **unit** is in **"Ping"** and each **Ping** is worth **$10,000**, the expected value of a house that is **brand new** (0 years old) is **$444,671.40 per Ping.** Finally, this means that the **total predicted price** of a **brand new house** is found by multiplying its **total size (Ping)** by **$444,671.40 Taiwanese dollars**. Note: This is only true given model **Y_X2** which only models variable X2 (house age) and Y (house price) and therefore all other things are already held equal (since there is only 1 predictor variable).
- The **slope** of β_1 **= -0.33841** means that for every unit change in variable **X2** or for **every 1 unit increase in house age (1 additional year)** the expected decrease of **house price** is **~0.338.** This can be interpreted to mean that **older houses** are expected to be **cheaper** or that as the **house age** increases –> **house price per Ping decreases**. The precise estimate is a **decrease of $3,384.1 per Ping** since the unit is **$10,000 per unit of Ping**.

```
#Q1.b Continued
#Computer R-Squared
summary(Y_X2)$r.squared

## [1] 0.08249071

#Training data MSE
train_pred <- predict(Y_X2, newdata = valuation_train)
train_mse_X2 <- mean((valuation_train$Y - train_pred)^2)
train_mse_X2

## [1] 166.6527

#Test data MSE
test_pred <- predict(Y_X2, newdata = valuation_test)
```

```r
test_mse_X2 <- mean((valuation_test$Y - test_pred)^2)
test_mse_X2

## [1] 188.0664
```

- The R^2 is equal to **0.08249071**. The training MSE is **166.6527**. The test MSE is **188.066**.

c. Repeat part (b) twice, but with X2 replaced by X3 and X4 (skip the interpretation of coefficients part). Among X2, X3 and X4, which variable explains the most variation of Y on the training data? Which variable is the most useful in predicting the house price for the test data? Explain your answers.

   Solution:

```r
#Q1.c
# Model X3
Y_X3 <- lm(Y~X3, data = valuation_train)

train_pred_X3 <- predict(Y_X3, newdata = valuation_train)
train_mse_X3 <- mean((valuation_train$Y - train_pred_X3)^2)

test_pred_X3 <- predict(Y_X3, newdata = valuation_test)
test_mse_X3 <- mean((valuation_test$Y - test_pred_X3)^2)

# Model X4
Y_X4 <- lm(Y~X4, data = valuation_train)

train_pred_X4 <- predict(Y_X4, newdata = valuation_train)
train_mse_X4 <- mean((valuation_train$Y - train_pred_X4)^2)

test_pred_X4 <- predict(Y_X4, newdata = valuation_test)
test_mse_X4 <- mean((valuation_test$Y - test_pred_X4)^2)

summary(Y_X2)$r.squared

## [1] 0.08249071

summary(Y_X3)$r.squared

## [1] 0.5152052

summary(Y_X4)$r.squared

## [1] 0.3726554
```

- Given that the R^2 of **Y_X2** is **0.08249071**, the R^2 of **Y_X3** is **0.5152052**, and the R^2 of **Y_X4** is **0.3726554**: **X3 ("distance to the nearest MRT station)** explains **the most** variation in Y in the training data. It has the highest R^2 value **(0.5152052)**, meaning it accounts for approximately **51.5%** of the variation in house price per Ping.

```
#Q1.c Continued
test_mse_X2

## [1] 188.0664

test_mse_X3

## [1] 113.7552

test_mse_X4

## [1] 135.5365
```

- **X3** is the **most useful predictor** of house price of unit area for the test data. This is because the **model using X3** has the **lowest test MSE** among the three models (Y_X2, Y_X3, Y_X4), meaning **Y_X3** makes the most accurate predictions on test (unseen) data. Note: MSE of **Y_X2** = ***188.0664***, **Y_X3** = ***113.7552***, **Y_X4** = ***135.5365***. **MSE of X3 < X4 < X2**.

d. Use the training data `valuation_train` to fit a multiple linear regression model of Y on X2, X3 and X4 simultaneously. Interpret all the coefficients (including the intercept) in terms of the real context. Compute R^2, training/test MSE, and compare the results to part (b) and (c).

Solution:

```
#Q1.d
#Multiple Linear Regression of Y ~ X2+X3+X4
Y_X2X3X4 <- lm(Y~X2+X3+X4, data = valuation_train)
summary(Y_X2X3X4)

##
## Call:
## lm(formula = Y ~ X2 + X3 + X4, data = valuation_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.962  -5.139  -0.927   4.028  29.328
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 44.6950373  1.9558434  22.852  < 2e-16 ***
## X2          -0.3064168  0.0524983  -5.837 2.18e-08 ***
## X3          -0.0053395  0.0006038  -8.842 5.43e-16 ***
## X4           1.2502186  0.2814432   4.442 1.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 8.455 on 196 degrees of freedom
## Multiple R-squared:  0.6143, Adjusted R-squared:  0.6084
## F-statistic:   104 on 3 and 196 DF,  p-value: < 2.2e-16
```

- The β_0 represents the **expected unit price (per Ping)** for a house given **X2 = 0, X3 = 0, and X4 = 0**. In other words, given that the house is **brand new**, is **0 meters from an MRT station**, and that there are **no nearby convenience stores**… the predicted cost is **44.6950373 X $10,000 per Ping**, which is around **$446,950.37**. Note: It's impractical that any house would be 0 meters from an MRT station since that would mean the house is located on top of the MRT station (not possible). Therefore, this is simply a baseline prediction given the multiple linear regression model.
- The β_1 of **X2 (House Age)** is **-0.3064168**. This means that for every additional **1 year of house age**, the expected **unit price per Ping decreases by 0.3064168**. This means that **older houses are cheaper** and that for each additional year, the unit price is expected to reduce by **$3,064 per Ping** all other variables held equal.
- The β_1 of **X3 (Distance from MRT)** is **-0.0053395**. This means that for every **1 meter increase in distance to the nearest MRT station**, the expected **unit price per Ping** decreases by **0.0053395**. This means that houses **farther** from an MRT station are **cheaper**. For every additional meter from an MRT station, the expected unit price drops by around **$53.40** all other variables held equal.
- The β_1 of **X4 (number of convenience stores in the living circle on foot)** is **1.2502186**. This means that for every additional **convenience store** within **walking distance**, the expected **unit price per ping** increases by **1.2502186**. This means that houses in an area with **more convenience** stores nearby tend to be **more valuable**. Each additional store is expected to increase the unit price by around **$12,502.19**.

```r
#Q1.d Continued
#Computing R-Squared
summary(Y_X2X3X4)$r.squared

## [1] 0.6142555

#Training MSE
train_pred <- predict(Y_X2X3X4, newdata = valuation_train)
train_mse_X2X3X4 <- mean((valuation_train$Y - train_pred)^2)

#Test MSE
test_pred <- predict(Y_X2X3X4, newdata = valuation_test)
test_mse_X2X3X4 <- mean((valuation_test$Y - test_pred)^2)

train_mse_X2X3X4

## [1] 70.06507

test_mse_X2X3X4

## [1] 100.0254
```

- The R^2 of **Y_X2X3X4 (multiple regression)** is **0.6142555**. The **training MSE of Y_X2X3X4** is **70.06507**. The **test MSE of Y_X2X3X4** is **100.0254**.

```r
#Q1.d Continued
summary(Y_X2)$r.squared

## [1] 0.08249071

summary(Y_X3)$r.squared

## [1] 0.5152052

summary(Y_X4)$r.squared

## [1] 0.3726554

summary(Y_X2X3X4)$r.squared

## [1] 0.6142555

train_mse_X2

## [1] 166.6527

train_mse_X3

## [1] 88.05616

train_mse_X4

## [1] 113.9483

train_mse_X2X3X4

## [1] 70.06507

test_mse_X2

## [1] 188.0664

test_mse_X3

## [1] 113.7552

test_mse_X4

## [1] 135.5365

test_mse_X2X3X4

## [1] 100.0254
```

- The **multiple regression model (Y_X2X3X4)** has the **highest** R^2 **(0.6142555)**. This means that it **explains the most variation** in house price per Ping.

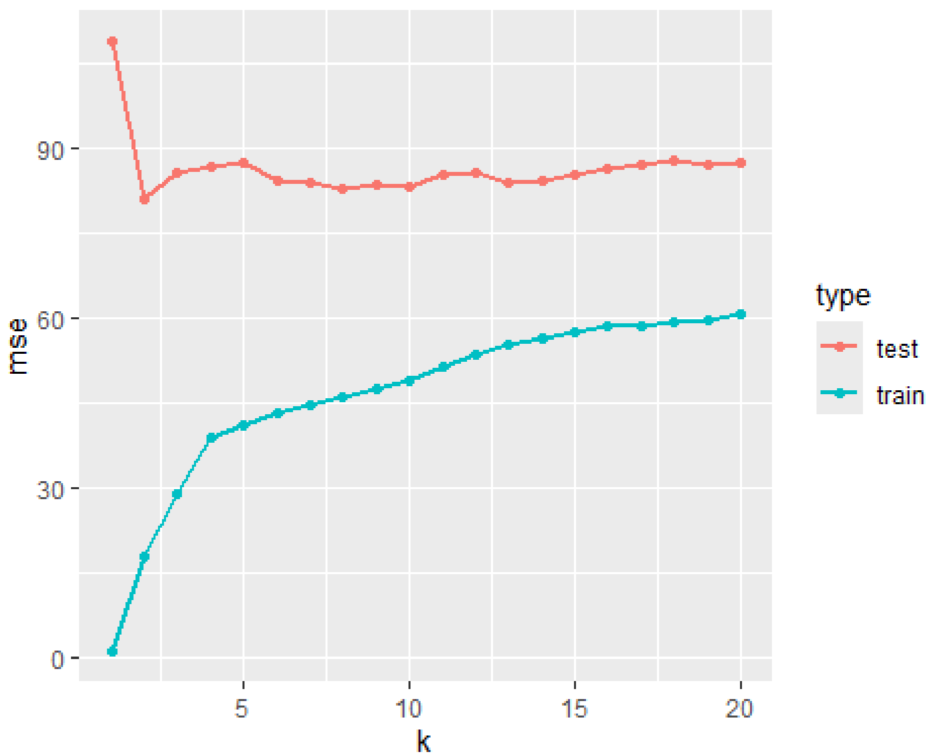
- The **multiple regression model (Y_X2X3X4)** has the **lowest training MSE (70.06507)**. This means that it is the best fit of the training data.
- The **multiple regression model (Y_X2X3X4)** has the **lowest test MSE (100.0254)**. This means that is the **most accurate at predicting unseen data**.
- Among the **single predictor models**, **X3** performs the **best** with an R^2 of **0.5152052** and a test MSE of **113.7552**.
- Among the **single predictor models**, **X2** is the **weakest** predictor, with the lowest R^2 and highest test MSE.

e. Use the knnreg function in the **caret** package to fit a KNN model of Y on X2, X3 and X4 simultaneously, for each K (number of neighbors) value from 1 to 20 with increment 1. Visualize the trends of training and test MSEs in a plot where K is on the horizontal axis and MSEs are on the vertical axis. What is the optimal K and best test error? Compare the result with linear regression in part (d).

Solution:

```
#Q1.e
# Fitting KNN models for values 1:20
library(caret)

## Warning: package 'caret' was built under R version 4.4.3

## Warning: package 'ggplot2' was built under R version 4.4.3

k_seq <- 1:20

mse_seq_tr <- mse_seq_te <- NULL

for (i in seq_along(k_seq)) {
  fit <- knnreg(
    Y~X2+X3+X4,
    data = valuation_train,
    k = k_seq[i]
  )

#Training MSE
train_pred <- predict(fit, newdata = valuation_train)
mse_seq_tr[i] <- mean((valuation_train$Y - train_pred)^2)

#Test MSE
test_pred <- predict(fit, newdata = valuation_test)
mse_seq_te[i] <- mean((valuation_test$Y - test_pred)^2)

}

#Visualize the trends of training and test MSE
```

```r
library(ggplot2)
mse_stacked <- rbind(
  data.frame(k = k_seq, mse = mse_seq_tr, type = "train"),
  data.frame(k = k_seq, mse = mse_seq_te, type = "test")
)

ggplot(mse_stacked, aes(x = k, y = mse, color = type)) +
  geom_line(linewidth = 1) +
  geom_point()
```




```r
#Q1.e Continued
#Optimal K and best test error
min_test_mse <- min(mse_seq_te)
min_test_mse

## [1] 81.09504

best_k <-k_seq[which.min(mse_seq_te)]
best_k

## [1] 2
```

- The **best performing KNN model** (using X2, X3, X4) has **K = 2 neighbors** and a **test MSE** equal to **81.09504**. Therefore, **K = 2** gives the most accurate predictions on unseen data. Compared to the **multiple linear regression model** which has a **test MSE of 100.0254**, the **best KNN model (K=2)** has the **lower test MSE of 81.09504**,

which makes it a better model for the valuation data set.
Q2

The iris data (available in base R) gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are setosa, versicolor, and virginica. Our goal is to predict the species using the given measurements.

To make this into a binary classification problem, we remove the setosa species from iris:

```
data(iris)
iris <- droplevels(iris[which(iris$Species != "setosa"), ]) # remove setosa
set.seed(1)
iris[, 1:4] <- iris[, 1:4] + rnorm(400) # add some noise
```

a. Split the dataset into training and test data, using the following instructions: Put rows 1–30 and 51–80 of iris in the training data; put rows 31–50 and 81–100 in the test data.[2] Name the training and test data iris_train and iris_test, respectively. Make sure that iris_test[34, 1] outputs 7.500214.

   Solution:

```
#Q2. a
iris_train <- iris[c(1:30, 51:80),]
iris_test <- iris[c(31:50, 81:100),]

iris_test[34,1]

## [1] 7.500214
```

b. Use the training data iris_train to fit a logistic regression model, a LDA model and a QDA model of Species on all other variables. Compute the training and test errors for each model (logistic regression/LDA/QDA). Which model has the best prediction accuracy for the iris dataset? Explain your answer.

   Solution:

```
#Q2.b
#Fitting logistic regression model
log_fit <- glm(
  Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
  data = iris_train,
```

[2] Rationale behind the above split strategy: The 1–50 rows of the iris data frame are of species versicolor, whereas the 51–100 rows are of virginica. By combining rows 1–30 and 51–80, we end up with a balanced mix of both species in the training data. The same goes for the test data.

```r
  family = "binomial"
)

#fitting LDA model
library(MASS)

## Warning: package 'MASS' was built under R version 4.4.3

lda_fit <- lda(
  Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
  data = iris_train
)

#fitting QDA model
qda_fit <- qda(
  Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width,
  data = iris_train
)

#Log training and test error
log_train_prob <- predict(log_fit,
                          newdata = iris_train,
                          type = "response")

class_labels <- levels(iris_train$Species)
c0 <- class_labels[1]
c1 <- class_labels[2]

log_train_label <- ifelse(log_train_prob
                          > 0.5,
                          c1,
                          c0)

log_train_error <- mean(log_train_label != iris_train$Species)

log_test_prob <- predict(log_fit, newdata = iris_test,
                         type = "response")

log_test_label <- ifelse(log_test_prob
                         > 0.5,
                         c1,
                         c0)

log_test_error <- mean(log_test_label != iris_test$Species)


#LDA training and test error
lda_train_pred <- predict(lda_fit, newdata = iris_train)
lda_train_error <- mean(lda_train_pred$class != iris_train$Species)
```

```r
lda_test_pred <- predict(lda_fit, newdata = iris_test)
lda_test_error <- mean(lda_test_pred$class != iris_test$Species)

#QDA training and test error
qda_train_pred <- predict(qda_fit, newdata = iris_train)
qda_class <- qda_train_pred$class
qda_train_error <- mean(qda_class != iris_train$Species)

qda_test_pred <- predict(qda_fit, newdata = iris_test)
qda_class <- qda_test_pred$class
qda_test_error <- mean(qda_class != iris_test$Species)

#Log, LDA, QDA errors
log_train_error

## [1] 0.2333333

log_test_error

## [1] 0.15

lda_train_error

## [1] 0.2333333

lda_test_error

## [1] 0.125

qda_train_error

## [1] 0.2333333

qda_test_error

## [1] 0.25
```

- The model that has the **best prediction accuracy** is the one with the **lowest test error**. This is because test error measures prediction accuracy on unseen data, which is a more true test for predictive capability. **LDA** has the best prediction accuracy for the ***iris*** data set. This is because **LDA achieved the lowest test error (0.125)**, which is lower than both the *logistic regression (0.15)* and *QDA (0.25)*.