```
# UCLA Extension - Introduction to Data Science
#
# Homework #3 Solutions
#
# (c) Copyright 2016-2019 - AMULET Analytics
# ---------------------------------------------------------------

# ---------------------------------------------------------------
# Question 1 - Supervised Machine Learning - Linear Regression
# ---------------------------------------------------------------

library(ISLR)

data(Auto)
head(Auto)

summary(Auto)

# Fit simple linear regression model
attach(Auto)    # So you won't have to keep repeating "Auto"

# Fit a linear model with response mpg and one predictor horsepower
lm = lm(mpg~horsepower)
summary(lm)               # Print all components of the fit

# Call:
# lm(formula = mpg ~ horsepower)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -13.5710  -3.2592  -0.3435   2.7630  16.9240
#
# Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
# (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
# horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 4.906 on 390 degrees of freedom
# Multiple R-squared:  0.6059,  Adjusted R-squared:  0.6049
# F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16

# Analysis: yes there is a negative linear relationship
# between horsepower and mpg since the F-statistic is >> 1
# and the p-value for the F-statistic is close to zero.
# R-squared could be higher, but OK at 0.6


# Predicted mpg associated with horsepower=98
predict.lm(lm, data.frame(horsepower=98), interval="confidence")
#        fit      lwr      upr
# 1 24.46708 23.97308 24.96108

# Another prediction method using the model coefficients
lm$coefficients
# (Intercept)  horsepower
#  39.9358610  -0.1578447

mpg1 <- lm$coefficients[1] + 98*lm$coefficients[2]
mpg1   # Same prediction as above
# (Intercept)
#    24.46708


# Plot linear model fit: regression line has negative slope
plot(horsepower, mpg)
abline(lm)

# Plot diagnostic plots: residual plot shows evidence of
# non-linearity.
par(mfrow=c(2,2))
plot(lm)

# Pairs plot: locate plot for mpg/horsepower, notice negative correlation
par(mfrow=c(1,1))
pairs(Auto)

# Correlation matrix: see correlation for mpg/horsepower=-0.79
cor(subset(Auto, select=-name))
#                    mpg  cylinders displacement horsepower     weight
#mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
#cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
#displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
#horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
```

```
#weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
#acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
#year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
#origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
#             acceleration       year     origin
#mpg             0.4233285  0.5805410  0.5652088
#cylinders      -0.5046834 -0.3456474 -0.5689316
#displacement   -0.5438005 -0.3698552 -0.6145351
#horsepower     -0.6891955 -0.4163615 -0.4551715
#weight         -0.4168392 -0.3091199 -0.5850054
#acceleration    1.0000000  0.2903161  0.2127458
#year            0.2903161  1.0000000  0.1815277
#origin          0.2127458  0.1815277  1.0000000


# -----------------------------------------------------------------
# Question 2 - Supervised Machine Learning - Classification
# -----------------------------------------------------------------

library(ISLR)
summary(Auto)

# Create binary categorical variable: 1 if mpg contains a value >
# median, and 0 if mpg contains a value below its median
attach(Auto)
mpg01 = rep(0, length(mpg))   # Start with all 0
mpg01[mpg > median(mpg)] = 1  # Selectively set 1 based on median

# Make copy of Auto data frame, and add new variable mpg01
auto_df = Auto
auto_df = data.frame(auto_df, mpg01)

# Calculate correlation matrix
cor(auto_df[, -9])    # Leave out name variable

# Anti correlated with cylinders, weight, displacement,
# horsepower, and mpg.
pairs(Auto)  # doesn't work well since mpg01 is 0 or 1

# Create training and test logical indexes
train_index = (year%%2 == 0)  # if the year is even
test_index = !train_index

auto_train = auto_df[train_index, ]   # Create training set df
auto_test = auto_df[test_index, ]     # Create test set df
mpg01_test = mpg01[test_index]

# Logistic regression with family=binomial. LR models the
# probability that the response belongs to a particular category.
# In this case, mpg01=0 or mpg01=1
glm_fit = glm(mpg01 ~ cylinders + weight + displacement + horsepower,
              data = auto_df, family = binomial, subset = train_index)

# Use trained model glm_fit to make test set predictions
glm_probs = predict.glm(glm_fit, newdata=auto_test, type = "response")
glm_pred = rep(0, length(glm_probs))
glm_pred[glm_probs > 0.5] = 1
mean(glm_pred != mpg01_test)

# Test error rate: 12.1%
#[1] 0.1208791


# -----------------------------------------------------------------
# Question 3 - Unsupervised Machine Learning - K-means Clustering
# -----------------------------------------------------------------

data(iris)

# Set up case for knowledge discovery using clusters
df <- data.frame(x=iris$Sepal.Length,y=iris$Sepal.Width)

# Number of initial centroids=3

# Need to set seed to get same results because kmeans() uses a
# random number generator to come up with the centers if you use the
# centers argument
set.seed(314)

# Initial number of centroids=3
kc <- kmeans(df,centers=3, iter.max=10)   # Create a kmeans object
names(kc)
#[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
#[6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Show which cluster each data point assigned to
kc$cluster     # This is an integer vector
#  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
# [37] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 3 1 3 1 3 1 3 3 3 3 3 3 1 3 3 3 3 3 3
# [73] 3 3 1 1 1 1 3 3 3 3 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 1 1 1 1 3 1
#[109] 1 1 1 1 1 3 3 1 1 1 1 3 1 3 1 3 1 1 3 3 1 1 1 1 1 3 3 1 1 1 3 1 1 1 3 1
#[145] 1 1 3 1 1 3

kc$centers
#         x        y
#1 6.812766 3.074468
#2 5.006000 3.428000
#3 5.773585 2.692453

# Plot clusters
# We discover 3 clusters: red, green, black with centroids
par(mar=rep(0.2,4))
plot(df$x,df$y,col=kc$cluster,pch=19,cex=2)   # Plot clusters
points(kc$centers,col=1:3,pch=3,cex=3,lwd=3)  # Plot centroids


# -----------------------------------------------------------------
# Bonus example - Multiple Linear Regression
# -----------------------------------------------------------------

library(ISLR)

data(Auto)
head(Auto)

summary(Auto)

# Scatterplot matrix showing correlations between all variables
pairs(Auto)

# Compute correlation matrix
cor(subset(Auto, select=-name))
#                     mpg  cylinders displacement horsepower     weight
#mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
#cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
#displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
#horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
#weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
#acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
#year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
#origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
#             acceleration       year     origin
#mpg             0.4233285  0.5805410  0.5652088
#cylinders      -0.5046834 -0.3456474 -0.5689316
#displacement   -0.5438005 -0.3698552 -0.6145351
#horsepower     -0.6891955 -0.4163615 -0.4551715
#weight         -0.4168392 -0.3091199 -0.5850054
#acceleration    1.0000000  0.2903161  0.2127458
#year            0.2903161  1.0000000  0.1815277
#origin          0.2127458  0.1815277  1.0000000

# Fit multi-variate linear model
lm.fit1 = lm(mpg~.-name, data=Auto)
summary(lm.fit1)

# Call:
# lm(formula = mpg ~ . - name, data = Auto)
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -9.5903 -2.1565 -0.1169  1.8690 13.0604
#
#Coefficients:
#                Estimate Std. Error t value Pr(>|t|)
# (Intercept)  -17.218435   4.644294  -3.707  0.00024 ***
# cylinders     -0.493376   0.323282  -1.526  0.12780
# displacement   0.019896   0.007515   2.647  0.00844 **
# horsepower    -0.016951   0.013787  -1.230  0.21963
# weight        -0.006474   0.000652  -9.929  < 2e-16 ***
# acceleration   0.080576   0.098845   0.815  0.41548
# year           0.750773   0.050973  14.729  < 2e-16 ***
# origin         1.426141   0.278136   5.127 4.67e-07 ***
# ---
# Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#
# Residual standard error: 3.328 on 384 degrees of freedom
# Multiple R-squared:  0.8215,  Adjusted R-squared:  0.8182
# F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16

# Yes, there is a relatioship between the predictors and the
```

```
# response by testing the null hypothesis of whether all the
# regression coefficients are zero. The F -statistic is far
# from 1 (with a small p-value), indicating evidence against
# the null hypothesis.

# Looking at the p-values associated with each predictor's
# t-statistic, we see that displacement, weight, year, and
# origin have a statistically significant relationship, while
# cylinders, horsepower, and acceleration do not.

# The regression coefficient for year, 0.7508, suggests that
# for every one year, mpg increases by the coefficient. In other
# words, cars become more fuel efficient every year by almost
# 1 mpg / year.

# Diagnostic plots of linear regression fit.
par(mfrow=c(2,2))
plot(lm.fit1)

# The fit does not appear to be accurate because there is a
# discernible curve pattern to the residuals plots. From the
# leverage plot, point 14 appears to have high leverage,
# although not a high magnitude residual.
```