A2

Q1

```
h2q1_trn <- read.csv("h2q1-trn-data.csv")
h2q1_tst <- read.csv("h2q1-tst-data.csv")

B = "dodgerblue"
O = "darkorange"

m1_f = function(val1){
  ifelse(val1 > 0, B, O)
}

m2_f = function(val1, val2) {
  ifelse(val2 > val1 + 1, B, O)
}

m3_f = function(val1, val2) {
  ifelse(val2 > val1 + 1, B, ifelse(val2 < val1 -1, B, O))
}

m4_f = function(val1, val2) {
  ifelse(val2 > (val1 + 1) ^ 2, B, ifelse(val2 < -(val1 - 1) ^ 2, B, O))
}

calc_class_error = function(actual, predicted) {
  mean(actual != predicted)
}

m1_trn_pred = m1_f(val1=h2q1_trn$x1)
m1_tst_pred = m1_f(val1=h2q1_tst$x1)
calc_class_error(h2q1_trn$y, m1_trn_pred)
calc_class_error(h2q1_tst$y, m1_tst_pred)


m2_trn_pred = m2_f(val1=h2q1_trn$x1, val2=h2q1_trn$x2)
m2_tst_pred = m2_f(val1=h2q1_tst$x1, val2=h2q1_tst$x2)
calc_class_error(h2q1_trn$y, m2_trn_pred)
calc_class_error(h2q1_tst$y, m2_tst_pred)


m3_trn_pred = m3_f(val1=h2q1_trn$x1, val2=h2q1_trn$x2)
m3_tst_pred = m3_f(val1=h2q1_tst$x1, val2=h2q1_tst$x2)
calc_class_error(h2q1_trn$y, m3_trn_pred)
calc_class_error(h2q1_tst$y, m3_tst_pred)


m4_trn_pred = m4_f(val1=h2q1_trn$x1, val2=h2q1_trn$x2)
m4_tst_pred = m4_f(val1=h2q1_tst$x1, val2=h2q1_tst$x2)
```

```
calc_class_error(h2q1_trn$y, m4_trn_pred)
calc_class_error(h2q1_tst$y, m4_tst_pred)
```

Q2

```
h2q1_trn <- read.csv("h2q1-trn-data.csv")
h2q1_tst <- read.csv("h2q1-tst-data.csv")

B = "dodgerblue"
O = "darkorange"

convert_to_num = function(str) {
  ifelse(str == O, 0, 1)
}

h2q1_trn$y = convert_to_num(h2q1_trn$y)
h2q1_tst$y = convert_to_num(h2q1_tst$y)

calc_class_error = function(actual, predicted) {
  mean(actual != predicted)
}

raw_m1 = glm(y ~ 1, data = h2q1_trn, family = "binomial")
raw_m2 = glm(y ~ ., data = h2q1_trn, family = "binomial")
raw_m3 = glm(y ~ . + I(x1^ 2) + I(x2^ 2), data = h2q1_trn, family =
"binomial")
raw_m4 = glm(y ~ . + I(x1^ 2) + I(x2^ 2) + I(x1*x2), data = h2q1_trn,
family = "binomial")

m1_trn_pred = ifelse(predict(raw_m1, h2q1_trn, type = "response") > 0.5,
1, 0)
m1_tst_pred = ifelse(predict(raw_m1, h2q1_tst, type = "response") > 0.5,
1, 0)
calc_class_error(h2q1_trn$y, m1_trn_pred)
calc_class_error(h2q1_tst$y, m1_tst_pred)


m2_trn_pred = ifelse(predict(raw_m2, h2q1_trn, type = "response") > 0.5,
1, 0)
m2_tst_pred = ifelse(predict(raw_m2, h2q1_tst, type = "response") > 0.5,
1, 0)
calc_class_error(h2q1_trn$y, m2_trn_pred)
calc_class_error(h2q1_tst$y, m2_tst_pred)


m3_trn_pred = ifelse(predict(raw_m3, h2q1_trn, type = "response") > 0.5,
1, 0)
m3_tst_pred = ifelse(predict(raw_m3, h2q1_tst, type = "response") > 0.5,
1, 0)
calc_class_error(h2q1_trn$y, m3_trn_pred)
```

```
calc_class_error(h2q1_tst$y, m3_tst_pred)


m4_trn_pred = ifelse(predict(raw_m4, h2q1_trn, type = "response") > 0.5,
1, 0)
m4_tst_pred = ifelse(predict(raw_m4, h2q1_tst, type = "response") > 0.5,
1, 0)
calc_class_error(h2q1_trn$y, m4_trn_pred)
calc_class_error(h2q1_tst$y, m4_tst_pred)
```

Q3

```
set.seed(123456789)

LOOPS = 1000
MODELS = 3

make_sim_data = function(n_obs = 25) {
  x1 = runif(n = n_obs, min = 0, max = 2)
  x2 = runif(n = n_obs, min = 0, max = 4)
  prob = exp(1 + 2 * x1 - 1 * x2) / (1 + exp(1 + 2 * x1 - 1 * x2))
  y = rbinom(n = n_obs, size = 1, prob = prob)
  data.frame(y, x1, x2)
}

calc_var = function(estimate) {
  mean((estimate - mean(estimate)) ^ 2)
}
calc_bias = function(estimate, truth) {
  (mean(estimate) - truth)^2
}
calc_mse = function(truth, estimate) {
  mean((estimate - truth) ^ 2)
}

results = matrix(0, nrow = LOOPS, ncol = MODELS)
ground_truth = data.frame(x1 = 1, x2 = 1, y = exp(1 + 2 * 1 - 1 * 1) / (1
+ exp(1 + 2 * 1 - 1 * 1)))

for (loop in 1:LOOPS) {
  sim_data = make_sim_data()

  raw_m1 = glm(y ~ 1, data = sim_data, family = "binomial")
  raw_m2 = glm(y ~ ., data = sim_data, family = "binomial")
  raw_m3 = glm(y ~ . + I(x1^ 2) + I(x2^ 2) + I(x1*x2), data = sim_data,
family = "binomial")

  results[loop, 1] = predict(raw_m1, newdata = ground_truth, type =
"response")
```

```
  results[loop, 2] = predict(raw_m2, newdata = ground_truth, type =
"response")
  results[loop, 3] = predict(raw_m3, newdata = ground_truth, type =
"response")
}

calc_var(results[,1])
calc_bias(results[,1], ground_truth$y)
calc_mse(results[,1], ground_truth$y)

calc_var(results[,2])
calc_bias(results[,2], ground_truth$y)
calc_mse(results[,2], ground_truth$y)

calc_var(results[,3])
calc_bias(results[,3], ground_truth$y)
calc_mse(results[,3], ground_truth$y)
```

Q4

```
library(caret) # install.packages('e1071', dependencies=TRUE)
library(class)

set.seed(314)

LOOPS = 51

convert_to_num = function(str) {
  # B -> 0, M -> 1
  ifelse(str == "B", 0, 1)
}

calc_class_error = function(actual, predicted) {
  mean(actual != predicted)
}

wisc_trn = read.csv("wisc-trn.csv")
wisc_trn$class = convert_to_num(wisc_trn$class)
wisc_tst = read.csv("wisc-tst.csv")
wisc_tst$class = convert_to_num(wisc_tst$class)

for (loop in seq(1, LOOPS, by=2)) {
  pred = knn(train = wisc_trn[, -1], test = wisc_trn[, -1], cl =
wisc_trn$class, k = loop)
  trn_err = calc_class_error(pred, wisc_tst$class)

  pred = knn(train = wisc_trn[, -1], test = wisc_tst[, -1], cl =
wisc_trn$class, k = loop)
  tst_err = calc_class_error(pred, wisc_tst$class)
```

```
  cat("k:", loop, "\ttrn_err:", trn_err, "\ttst_err:", tst_err, "\n")
}

base_model = glm(class ~ radius + symmetry, data = wisc_trn, family =
"binomial")

m1 = ifelse(predict(base_model, wisc_tst, type = "response") > 0.1, 1, 0)
m2 = ifelse(predict(base_model, wisc_tst, type = "response") > 0.5, 1, 0)
m3 = ifelse(predict(base_model, wisc_tst, type = "response") > 0.9, 1, 0)

(t1 = table(predicted = m1, actual = wisc_tst$class))
(cm1 = confusionMatrix(t1, positive = "1"))
(t2 = table(predicted = m2, actual = wisc_tst$class))
(cm2 = confusionMatrix(t2, positive = "1"))
(t3 = table(predicted = m3, actual = wisc_tst$class))
(cm3 = confusionMatrix(t3, positive = "1"))
```

Q5

```
library(caret)
library(ellipse)
library(MASS)
library(nnet)
library(e1071)

calc_class_error = function(actual, predicted) {
  mean(actual != predicted)
}

h2q5_trn <- read.csv("h2q5-trn.csv")
h2q5_tst <- read.csv("h2q5-tst.csv")

h2q5_trn$y <- as.factor(h2q5_trn$y)
h2q5_tst$y <- as.factor(h2q5_tst$y)

caret::featurePlot(
  x = h2q5_trn[, c("x1", "x2")],
  y = h2q5_trn$y,
  plot = "ellipse"
)

#Additive Logistic Regression
m_alr = multinom(y ~ x1 + x2, data = h2q5_trn, trace = FALSE)
p_alr_trn = predict(m_alr, newdata = h2q5_trn)
p_alr_tst = predict(m_alr, newdata = h2q5_tst)
calc_class_error(p_alr_trn, h2q5_trn$y)
calc_class_error(p_alr_tst, h2q5_tst$y)

#LDA (est prior)
```

```
m_lda_est = lda(y ~ x1 + x2, data = h2q5_trn)
p_lda_est_trn = predict(m_lda_est, h2q5_trn)$class
p_lda_est_tst = predict(m_lda_est, h2q5_tst)$class
calc_class_error(p_lda_est_trn, h2q5_trn$y)
calc_class_error(p_lda_est_tst, h2q5_tst$y)

#LDA (flt prior)
m_lda_flt = lda(y ~ x1 + x2, data = h2q5_trn, prior = c(1,1,1,1)/4)
m_lda_flt_trn = predict(m_lda_flt, h2q5_trn)$class
m_lda_flt_tst = predict(m_lda_flt, h2q5_tst)$class
calc_class_error(m_lda_flt_trn, h2q5_trn$y)
calc_class_error(m_lda_flt_tst, h2q5_tst$y)

#QDA (est prior)
m_qda_est = qda(y ~ x1 + x2, data = h2q5_trn)
m_qda_est_trn = predict(m_qda_est, h2q5_trn)$class
m_qda_est_tst = predict(m_qda_est, h2q5_tst)$class
calc_class_error(m_qda_est, h2q5_trn$y)
calc_class_error(m_qda_est_tst, h2q5_tst$y)

#QDA (flt prior)
m_qda_flt = qda(y ~ x1 + x2, data = h2q5_trn,  prior = c(1,1,1,1)/4)
m_qda_flt_trn = predict(m_qda_flt, h2q5_trn)$class
m_qda_flt_tst = predict(m_qda_flt, h2q5_tst)$class
calc_class_error(m_qda_flt_trn, h2q5_trn$y)
calc_class_error(m_qda_flt_tst, h2q5_tst$y)

#Naive Bayes (est prior)
m_nb = naiveBayes(y ~ ., data = h2q5_trn)
m_nb_trn = predict(m_nb, newdata =  h2q5_trn)
m_nb_tst = predict(m_nb, newdata =  h2q5_tst)
calc_class_error(m_nb_trn, h2q5_trn$y)
calc_class_error(m_nb_tst, h2q5_tst$y)
```