Compulsory exercise 1: Group 15

TMA4268 Statistical Learning V2023

Håkon Kjelland-Mørdre, Mathias Karsrud Nordal and Johan Vik Mathisen

17 February, 2023

```
library(ggplot2)
library(class)
library(MASS) # for QDA
library(plotROC)
library(pROC)

library(carData)
library(GGally)

library(ggfortify)
library(dplyr)
```

Problem 1

1

a)

The expected value and the covariance matrix of the ridge regression estimator $\tilde{\beta}$ is given by

$$\begin{aligned} E[\tilde{\beta}] &= E[(X^\top X + \lambda I)^{-1} X^\top Y] \\ &= (X^\top X + \lambda I)^{-1} X^\top E[Y] \\ &= (X^\top X + \lambda I)^{-1} X^\top X \beta \end{aligned}$$

$$\begin{aligned} Cov[\tilde{\beta}] &= Cov[(X^\top X + \lambda I)^{-1} X^\top Y] \\ &= (X^\top X + \lambda I)^{-1} X^\top Cov[Y]((X^\top X + \lambda I)^{-1} X^\top)^\top \\ &= \ldots \\ &= \sigma^2 (X^\top X + \lambda I)^{-1} X^\top X (X^\top X + \lambda I)^{-1} \end{aligned}$$

b)

Let $\tilde{f}(x_0) = x_0^\top \tilde{\beta}$, then the expectation and variance of $\tilde{f}(x_0)$ is

$$\begin{aligned} E[\tilde{f}(x_0)] &= E[x_0^\top \tilde{\beta}] \\ &= x_0^\top E[\tilde{\beta}] \\ &= x_0^\top (X^\top X + \lambda I)^{-1} X^\top X \beta \end{aligned}$$

$$
\begin{aligned}
Var[\tilde{f}(x_0)] &= x_0^\top Var[\tilde{\beta}] x_0 \\
&= x_0^\top \sigma^2 (X^\top X + \lambda I)^{-1} X^\top X (X^\top X + \lambda I)^{-1} x_0
\end{aligned}
$$

c)

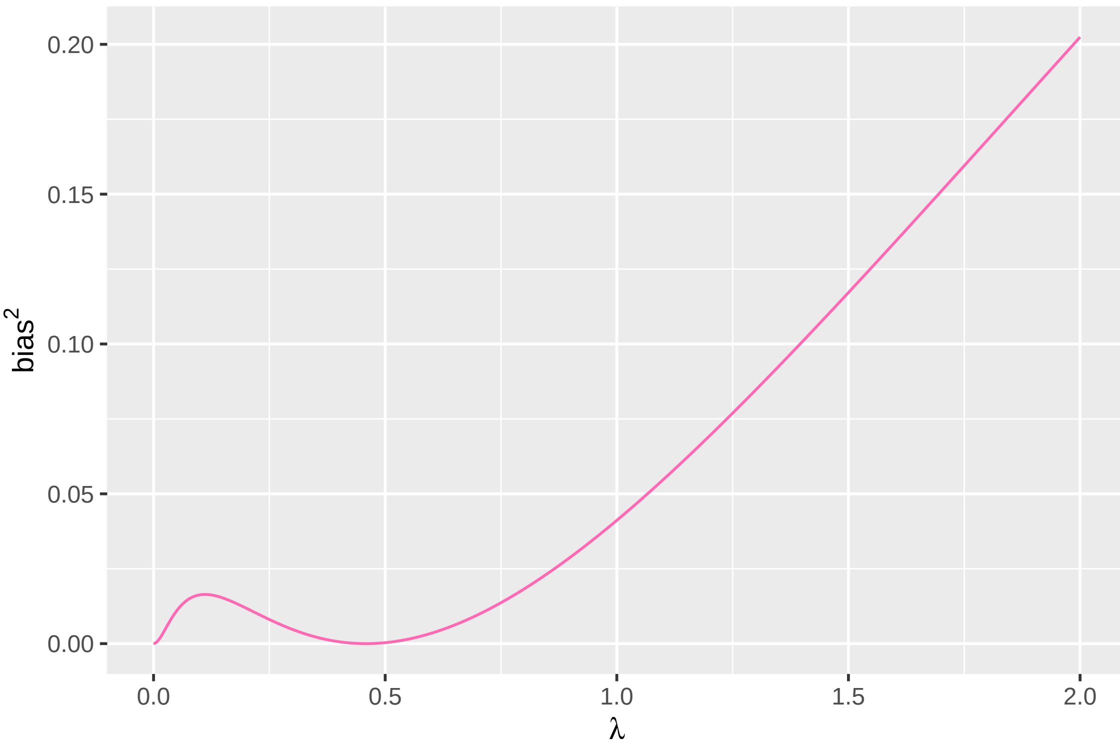
Suppose the true relation between response(s) and target is given by $Y = f(x) + \epsilon$. Any model, $\hat{f}(x)$, is trying to estimate the true function $f(x)$. Thus, there will always be an irreducible error ϵ, which can never be removed (it comes with the true model, so to speak). The variance of the model tells us something about the uncertainty of its predictions. Typically, as the flexibility of the model increases, its variance increases too. The bias tells us how much the model's predictions differ from the true mean (at given points).

d)

$$
\begin{aligned}
E[(y_0 - \tilde{f}(x_0))^2] &= Var(\epsilon) + Var(\tilde{f}(x_0)) + (f(x_0) - E[\tilde{f}(x_0)])^2 \\
&= \sigma^2 + x_0^\top \sigma^2 (X^\top X + \lambda I)^{-1} X^\top X (X^\top X + \lambda I)^{-1} x_0 + (x_0^\top \beta - x_0^\top (X^\top X + \lambda I)^{-1} X^\top X \beta)^2
\end{aligned}
$$

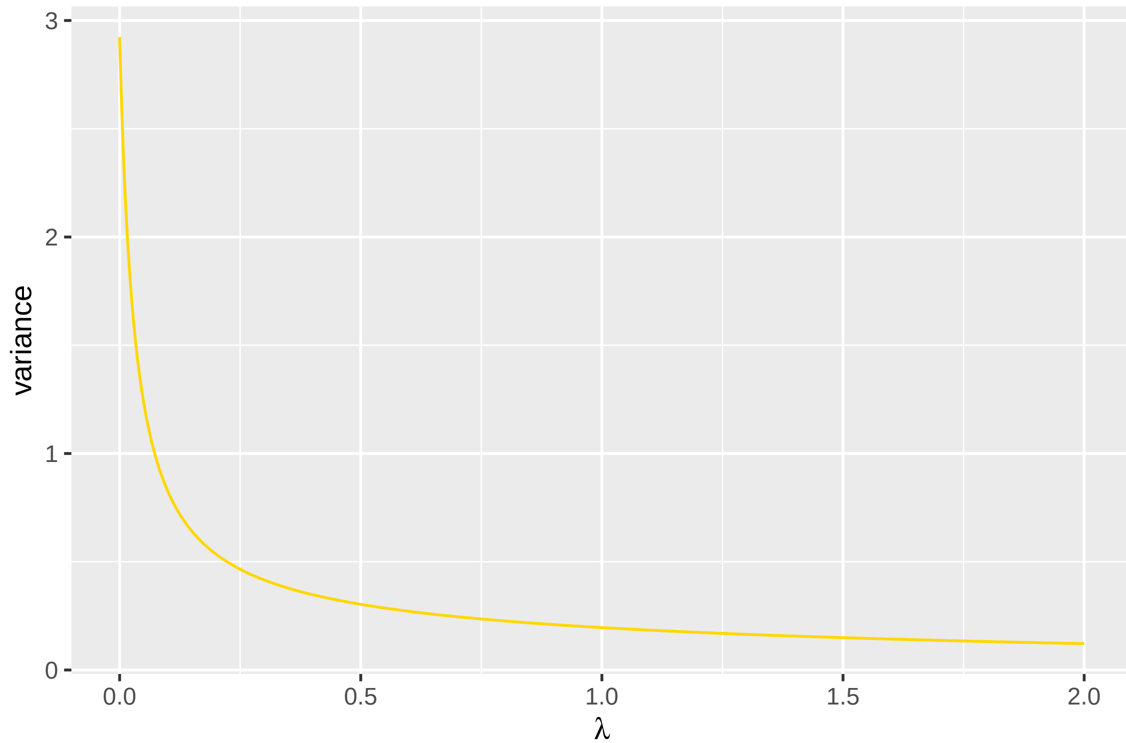
e)

```
#Copying code from the assignment
id <- "1X_8OKcoYbng1XvYFDirxjEWr7LtpNr1m" # google file ID
values <- dget(sprintf("https://docs.google.com/uc?id=%s&export=download", id))

X <- values$X
x0 <- values$x0
beta <- values$beta
sigma <- values$sigma


bias <- function(lambda, X, x0, beta) {
  p <- ncol(X)
  value <- (t(x0) %*% beta - t(x0) %*% solve(t(X) %*% X + lambda * diag(p)) %*%
            t(X) %*% X %*% beta)^2
  return(value)
}
lambdas <- seq(0, 2, length.out = 500)
BIAS <- rep(NA, length(lambdas))

for (i in seq_along(lambdas)) BIAS[i] <- bias(lambdas[i], X, x0, beta)

dfBias <- data.frame(lambdas = lambdas, bias = BIAS)
ggplot(dfBias, aes(x = lambdas, y = bias)) +
  geom_line(color = "hotpink") +
  xlab(expression(lambda)) +
  ylab(expression(bias^2))
```



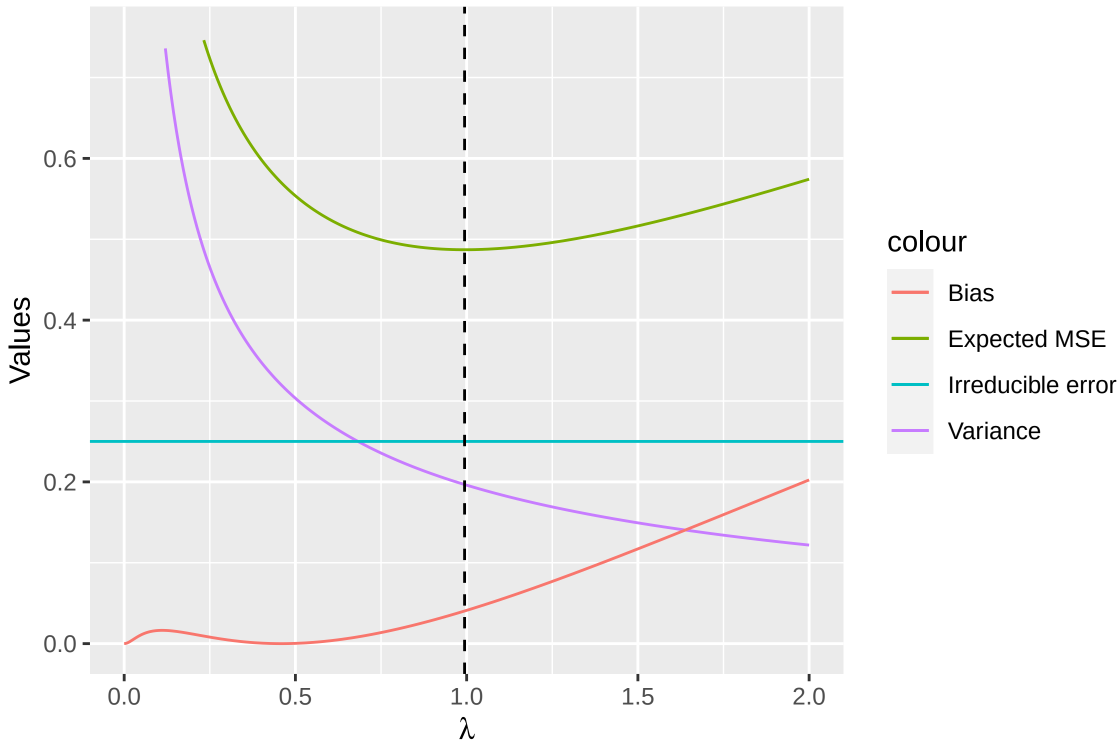
f)

```
variance <- function(lambda, X, x0, sigma) {
  p <- ncol(X)
  inv <- solve(t(X) %*% X + lambda * diag(p))
  value <- (sigma^2) * t(x0) %*% inv %*% t(X) %*% X %*% inv  %*% x0
  return(value)
}
lambdas <- seq(0, 2, length.out = 500)
VAR <- rep(NA, length(lambdas))
for (i in seq_along(lambdas)) VAR[i] <- variance(lambdas[i], X, x0, sigma)
dfVar <- data.frame(lambdas = lambdas, var = VAR)
ggplot(dfVar, aes(x = lambdas, y = var)) +
  geom_line(color = "gold") +
  xlab(expression(lambda)) +
  ylab("variance")
```



g)

```
exp_mse <- VAR + BIAS + sigma^2 #using the calculations above
dfexp_mse <- data.frame(lambdas = lambdas, mse = exp_mse)
minlambda <- lambdas[which.min(exp_mse)]
ggplot() +
    geom_line(dfVar, mapping = aes(x = lambdas, y = var, color = "Variance")) +
    geom_line(dfBias, mapping = aes(x = lambdas, y = bias, color = "Bias")) +
    geom_line(dfexp_mse,
              mapping = aes(x = lambdas, y = mse, color = "Expected MSE")) +
    xlab(expression(lambda)) +
    ylab("Values") + ylim(0, 0.75) +
    geom_hline(aes(yintercept = sigma^2, color = "Irreducible error")) +
    geom_vline(xintercept = minlambda, linetype = "dashed")
```




```r
#The value of lambda that minimizes MSE
print(minlambda)
```

```
## [1] 0.993988
```

Problem 2

a)

```r
# Fit full model
model1 <- lm(salary ~ ., data = Salaries)
summary(model1)
```

```
##
## Call:
## lm(formula = salary ~ ., data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -65248 -13211  -1775  10384  99592
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      65955.2     4588.6  14.374  < 2e-16 ***
## rankAssocProf    12907.6     4145.3   3.114  0.00198 **
## rankProf         45066.0     4237.5  10.635  < 2e-16 ***
## disciplineB      14417.6     2342.9   6.154 1.88e-09 ***
## yrs.since.phd      535.1      241.0   2.220  0.02698 *
## yrs.service       -489.5      211.9  -2.310  0.02143 *
## sexMale           4783.5     3858.7   1.240  0.21584
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22540 on 390 degrees of freedom
## Multiple R-squared:  0.4547, Adjusted R-squared:  0.4463
## F-statistic:  54.2 on 6 and 390 DF,  p-value: < 2.2e-16
```

```r
#res <- model1.matrix(~rank, data = Salaries)
#head(res[, -1])
```

(i) When the `lm()` function encounters a qualitative variable with k levels, the function transforms the variable into $k-1$ variables with binary levels. Implicitly, the function defines the first of the k levels as a reference level. In our case, the reference is rankAsstProf. Moreover, the intercept estimate can be interpreted as the expected salary of an assistant professor, whilst the expected salary of an associate professor and a full professor is the intercept estimate added to the corresponding estimate.

That is, the estimated salary for an associate professor is $\beta_i + \beta_{assocProf} = 65955.2 + 12907.6 = 78868.8$ and the estimated salary for a full professor is $\beta_i + \beta_{prof} = 65955.2 + 45066.0 = 111021.2$.

(ii) We would need to preform an F-test to test whether $\beta_{assocProf} = \beta_{prof} = 0$ at the same time. This is implemented in the `anova()` function

```r
anova(model1)
```

```
## Analysis of Variance Table
##
## Response: salary
##                Df     Sum Sq    Mean Sq  F value    Pr(>F)
## rank            2 1.4323e+11 7.1616e+10 140.9788 < 2.2e-16 ***
## discipline      1 1.8430e+10 1.8430e+10  36.2801 3.954e-09 ***
## yrs.since.phd   1 1.6565e+08 1.6565e+08   0.3261   0.56830
## yrs.service     1 2.5763e+09 2.5763e+09   5.0715   0.02488 *
## sex             1 7.8068e+08 7.8068e+08   1.5368   0.21584
## Residuals     390 1.9812e+11 5.0799e+08
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We see that the F-value associated with rank is very low. Therefore, it is reasonable to suspect that the variable has an impact on the salary of a professor.

b)

```r
sex_model <- lm(salary ~ sex, data = Salaries)
summary(sex_model)
```

```
##
## Call:
## lm(formula = salary ~ sex, data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -57290 -23502  -6828  19710 116455
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   101002       4809  21.001  < 2e-16 ***
```

```
## sexMale        14088       5065   2.782  0.00567 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 30030 on 395 degrees of freedom
## Multiple R-squared:  0.01921,    Adjusted R-squared:  0.01673
## F-statistic: 7.738 on 1 and 395 DF,  p-value: 0.005667
```

Recall that the R^2 values are relative measures of the models lack of fit. Moreover, $R^2 \in [0, 1]$ and $R^2 = 1$ represent a perfect fit. Observe that the adjusted R^2 for the multiple linear regression (`mlr`) model is $R^2_{mlr} = 0.4463$ which is more than 25 times as much as for the model using only sex as a covariate. This indicates that there are many stronger predictors of salary than sex. This is why the p-value of the the last model indicates a stronger correlation than what the mlr model does. In particular, rank is a good predictor of salary. Below we fit a `lm` model with `rank` the only covariate and show its R^2 value.

```r
phd_model <- lm(salary ~ rank, data = Salaries)
summary(phd_model)$r.squared
```

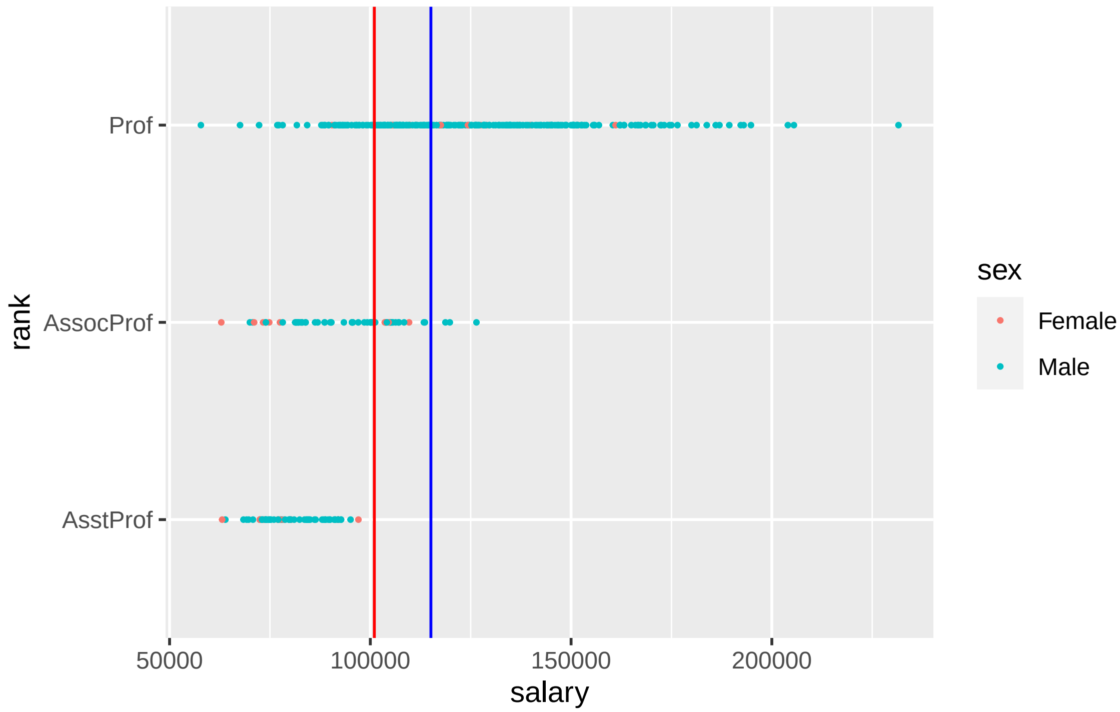
```
## [1] 0.3942513
```

For a more descriptive analysis we have already established that rank is a good predictor for salary, and from the pairs plot we see that the distribution of ranks in the two sexes is quite different. There are more male professors relative to the total number of males in the data set, than for the females. Below we first plot salary against rank and we mark the mean salary of all females (red) and mean salary of all males (blue). In our second plot we include the means of each sex, now sorted by rank. We too include tables of the means used in both plots.

The first plot shows a clear wage gap between male and female, but by including a second covariate, our second plot tells a slightly different story. The wage gap is there, but not nearly as significant the simple model implied.

This is sort of an example of Simpsons paradox.

```r
femaleProf <- subset(Salaries, sex == "Female")
maleProf <- subset(Salaries, sex == "Male")

meanSalMale <- mean(maleProf$salary)
meanSalFemale <- mean(femaleProf$salary)
#Create a table with the data.
rank_labs <- c("Mean salary")
sex_labs <- c("Male", "Female")


data <- c(meanSalMale, meanSalFemale)
means <- matrix(data = data, nrow = 2, ncol = 1,
                dimnames = list(sex_labs, rank_labs))

ggplot(data = Salaries) +
  geom_point(aes(x = salary, y = rank, color = sex), size = .5) +
  geom_vline(aes(xintercept = meanSalMale), color = "blue") +
  geom_vline(aes(xintercept = meanSalFemale), color = "red") +
  labs(caption = "Mean salaries of females (red) and males (blue)")
```



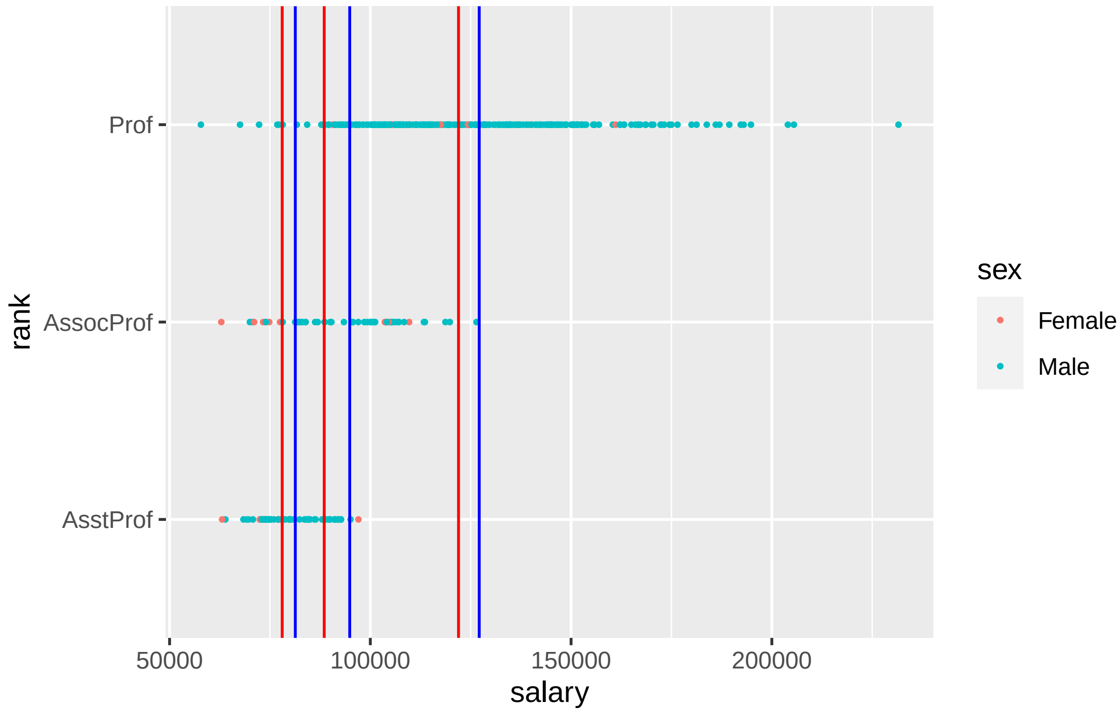
Mean salaries of females (red) and males (blue)

```
print(means)
```

```
##         Mean salary
## Male       115090.4
## Female     101002.4
```

```
femaleProf <- subset(Salaries, sex == "Female" & rank == "Prof")
femaleAssocProf <- subset(Salaries, sex == "Female" & rank == "AssocProf")
femaleAsstProf <- subset(Salaries, sex == "Female" & rank == "AsstProf")

maleProf <- subset(Salaries, sex == "Male" & rank == "Prof")
maleAssocProf <- subset(Salaries, sex == "Male" & rank == "AssocProf")
maleAsstProf <- subset(Salaries, sex == "Male" & rank == "AsstProf")

meanSalFemProf <- mean(femaleProf$salary)
meanSalFemAssocProf <- mean(femaleAssocProf$salary)
meanSalFemAsstProf <- mean(femaleAsstProf$salary)

meanSalMaleProf <- mean(maleProf$salary)
meanSalMaleAssocProf <- mean(maleAssocProf$salary)
meanSalMaleAsstProf <- mean(maleAsstProf$salary)


#Create a table with the data.
rank_labs <- c("AsstProf", "AssocProf", "Prof")
sex_labs <- c("Male", "Female")

means_male <- c(meanSalMaleAsstProf, meanSalMaleAssocProf, meanSalMaleProf)
means_female <- c(meanSalFemAsstProf, meanSalFemAssocProf, meanSalFemProf)

data <- rbind(means_male, means_female)
```

```r
means <- matrix(data = data, nrow = 2, ncol = 3,
                dimnames = list(sex_labs, rank_labs))


#Plot data together with means
ggplot(data = Salaries) +
  geom_point(aes(x = salary, y = rank, color = sex), size = .5) +
  geom_vline(aes(xintercept = meanSalMaleProf), color = "blue") +
  geom_vline(aes(xintercept = meanSalFemProf), color = "red") +
  geom_vline(aes(xintercept = meanSalMaleAssocProf), color = "blue") +
  geom_vline(aes(xintercept = meanSalFemAssocProf), color = "red") +
  geom_vline(aes(xintercept = meanSalMaleAsstProf), color = "blue") +
  geom_vline(aes(xintercept = meanSalFemAsstProf), color = "red") +
  labs(caption = "Mean salaries of females (red) and males (blue)")
```



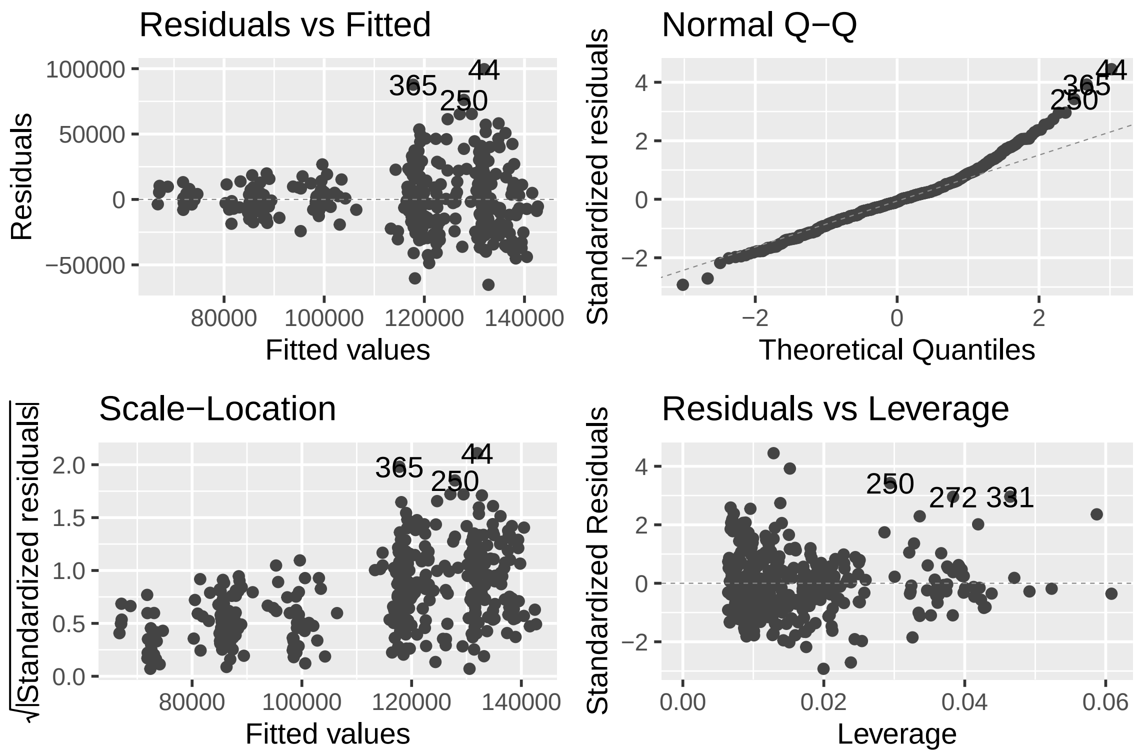
Mean salaries of females (red) and males (blue)

```r
print(means)
```

```
##         AsstProf AssocProf     Prof
## Male    81311.46   94869.7 127120.8
## Female 78049.91   88512.8 121967.6
```

c)

```r
autoplot(model1, smooth.colour = NA)
```



i) The Residual vs. Fitted plot shows clearly that the Var[ϵ_i] is not a constant, but increases with increasing salary. Which means that the assumption of constant variance is not fulfilled. Moreover, from the Q-Q plot it is not evident that the residuals is normaly distributed.

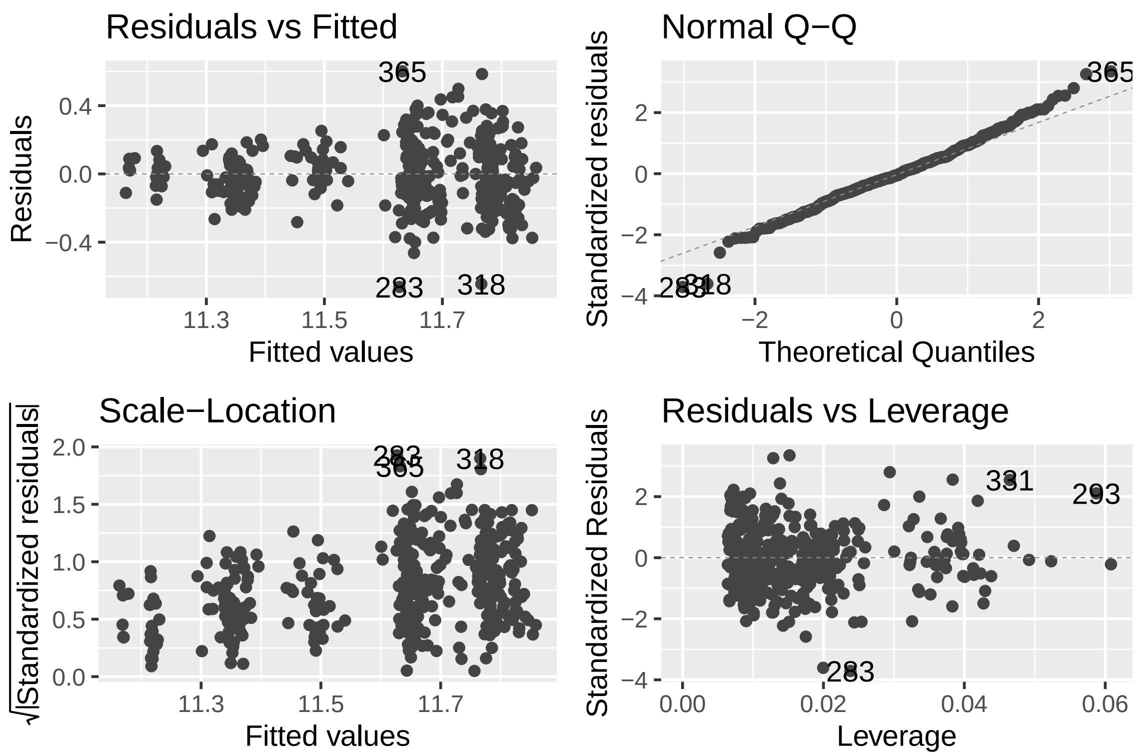
```
model2 <- lm(log(salary) ~ ., data = Salaries)
summary(model2)
```

ii)

```
##
## Call:
## lm(formula = log(salary) ~ ., data = Salaries)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66236 -0.10813 -0.00914  0.09804  0.60107
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   11.164144   0.036794 303.425  < 2e-16 ***
## rankAssocProf  0.153787   0.033239   4.627 5.06e-06 ***
## rankProf       0.449463   0.033979  13.228  < 2e-16 ***
## disciplineB    0.131869   0.018786   7.019 9.94e-12 ***
## yrs.since.phd  0.003289   0.001932   1.702   0.0896 .
## yrs.service   -0.003918   0.001699  -2.305   0.0217 *
## sexMale        0.045583   0.030941   1.473   0.1415
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.1807 on 390 degrees of freedom
## Multiple R-squared:  0.5248, Adjusted R-squared:  0.5175
## F-statistic: 71.79 on 6 and 390 DF,  p-value: < 2.2e-16
```

```
autoplot(model2, smooth.colour = NA)
```



Firstly, the distribution of the residuals appears to be closer to a normal distribution than it was earlier. Furthermore, the spread in the residuals vs fitted plot is drastically decreased. In conclusion; the model assumptions are fulfilled better than in the previous model.

d)

```
model3 <- update(model2, . ~ . + sex * yrs.since.phd)
summary(model3)
```

```
##
## Call:
## lm(formula = log(salary) ~ rank + discipline + yrs.since.phd +
##     yrs.service + sex + yrs.since.phd:sex, data = Salaries)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66187 -0.10831 -0.00951  0.09846  0.60143
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          11.1537511  0.0591759 188.485  < 2e-16 ***
## rankAssocProf         0.1528200  0.0335575   4.554 7.05e-06 ***
## rankProf              0.4482679  0.0344343  13.018  < 2e-16 ***
## disciplineB           0.1317818  0.0188133   7.005 1.09e-11 ***
## yrs.since.phd         0.0039500  0.0035253   1.120   0.2632
```

```
## yrs.service           -0.0038902  0.0017059  -2.280   0.0231 *
## sexMale                0.0574914  0.0614436   0.936   0.3500
## yrs.since.phd:sexMale -0.0007049  0.0031407  -0.224   0.8225
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1809 on 389 degrees of freedom
## Multiple R-squared:  0.5249, Adjusted R-squared:  0.5163
## F-statistic: 61.39 on 7 and 389 DF,  p-value: < 2.2e-16
```

ii) Considering that the p-value associated with the interaction term is very high, i.e. 0.8225 he seems to be wrong in his hypothesis.

e)

```r
# Defining a function to extract R^2 from linear model
rsq <- function(model) {
  return(summary(model)$r.squared)
}

# Generate 1000 bootstrap samples of R^2
set.seed(4268)
N <- 1000
n <- nrow(Salaries)
bootstrapped_rsq <- numeric(N)
for (i in 1:N) {
  index <- sample(n, replace = TRUE)      # Sampling the data set with replacement
  data <- Salaries[index, ]               # Constructing a new sampled data set
  fit <- lm(salary ~., data = data)       # Fiting a model to the sampled data set
  bootstrapped_rsq[i] <- rsq(fit)         # Calculating R^2, and adding it to the vector
}
```

i)

(ii) & (iii) The bootstrap standard error is the sample standard deviation of the N bootstrap samples.

```r
# Standard error
se_rsq <- sd(bootstrapped_rsq)

# 95% quantile intervals of the bootstrapped R^2 values
quantiles <- quantile(bootstrapped_rsq, c(0.025, 0.975))

quantiles
```

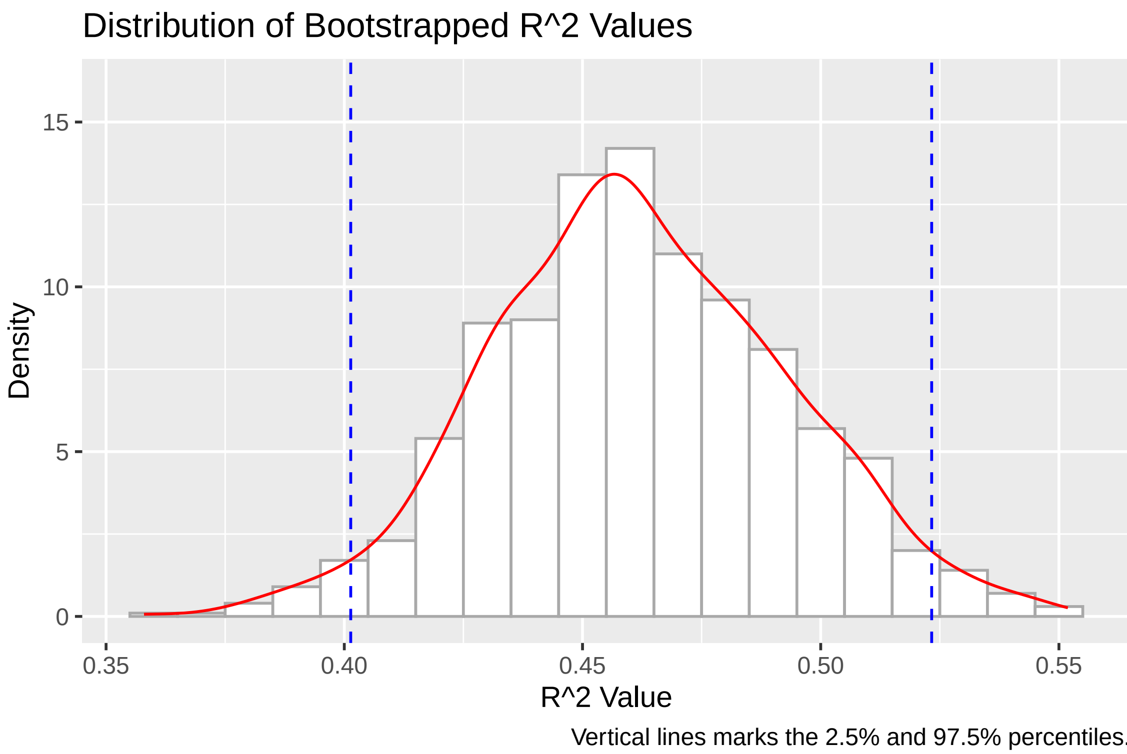
```
##      2.5%     97.5%
## 0.4013497 0.5232911
```

```r
#Standard error
se_rsq
```

```
## [1] 0.03093704
```

```r
# Plot the histogram and quantiles using ggplot2
ggplot(data.frame(bootstrapped_rsq), aes(x = bootstrapped_rsq)) +
  geom_histogram(aes(y = ..density..), color = "darkgray", fill = "white", binwidth = 0.01) +
  geom_density(color = "red") +
```

```
  geom_vline(xintercept = quantiles, color = c("blue", "blue"), linetype = c(2,2)) +
  ylim(0, max(density(bootstrapped_rsq)$y) * 1.2) +
  ggtitle("Distribution of Bootstrapped R^2 Values") +
  xlab("R^2 Value") +
  ylab("Density") +
  labs(caption = "Vertical lines marks the 2.5% and 97.5% percentiles.")
```



(iv) As the distribution looks to be symmetric about its mean and roughly follows a bell shape, it is reasonable to assume that the values are normally distributed.

f)

```
# Make a data frame containing two new observations, corresponding to
# Bert-Ernie's two possible futures
bert_ernie <- data.frame(rank = c("Prof", "Prof"),
                         discipline = c("A", "B"), # Theoretical, applied
                         yrs.since.phd = c(20, 20),
                         yrs.service = c(20, 20),
                         sex = c("Male", "Male"))
#bert_ernie
# Use the full model to predict his salary
preds <- predict(object = model1,
                 newdata = bert_ernie,
                 interval = "prediction", # and this should be prediction not conf.
                 level = 0.95) #  0.95 since we dont care about upper limit
# Check predictions
preds
```

(i)

```
##         fit      lwr      upr
## 1 116715.6 72121.12 161310.0
## 2 131133.2 86606.96 175659.4
```

```
# Check if lower limit for salary in a theoretical field is large enough
preds[1, 2] > 75000
```

```
## [1] FALSE
```

There seems to be many sleepless nights of debugging `Rcode` awaiting.

(ii)

Problem 3

```
bigfoot_original <- readr::read_csv(
"https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2022/2022-09-13/bigfoot.csv"
)



# Prepare the data:
bigfoot <- bigfoot_original %>%
  # Select the relevant covariates:
  dplyr::select(classification, observed, longitude, latitude, visibility) %>%
  # Remove observations of class C (these are second- or third hand accounts):
  dplyr::filter(classification != "Class C") %>%
  # Turn into 0/1, 1 = Class A, 0 = Class B:
  dplyr::mutate(class = ifelse(classification == "Class A", 1, 0)) %>%
  # Create new indicator variables for some words from the description:
  dplyr::mutate(fur = grepl("fur", observed),
                howl = grepl("howl", observed),
                saw = grepl("saw", observed),
                heard = grepl("heard", observed)) %>%
  # Remove unnecessary variables:
  dplyr::select(-c("classification", "observed")) %>%
  # Remove any rows that contain missing values:
  tidyr::drop_na()
```

```
set.seed(2023)
# 70% of the sample size for training set
training_set_size <- floor(0.7 * nrow(bigfoot))
train_ind <- sample(seq_len(nrow(bigfoot)), size = training_set_size)
train <- bigfoot[train_ind, ]
test <- bigfoot[-train_ind, ]
```

a)

(i)

```
# Fitting a logistic regression model using the training set.
# All covariates concidered.
glm_bigfoot <- glm(class ~ .,
                   family = "binomial",
```

```r
                    data = train
                    )
```

```r
#Use the logReg model to predict class in test data.
predict_bigfoot_glm <- predict(glm_bigfoot, test, type = "response")

#If the response variable exeedes 0.5 -> Class A (1), else Class B (0).
predict_glm <- ifelse(predict_bigfoot_glm > 0.5, 1, 0)


#Nr of observations classified as class A.
length(which(predict_glm == 1))
```

```
## [1] 441
```

```r
length(predict_glm)
```

```
## [1] 912
```

(ii)

Our answer is **4)**.

b)

(i)

```r
# Fit the model to training set
bigfoot_qda <- qda(class ~ ., data = train)

#Predicted classes
#Since only 2 classes, 0.5 cutoff is default.
predict_bigfoot_qda <- predict(bigfoot_qda, test)$class

#Corresponding predicted probabilities
prob_bigfoot_qda <- predict(bigfoot_qda, test)$posterior


# Number of reports classified as class A
length(which(predict_bigfoot_qda == 1))
```

```
## [1] 626
```

(ii)

1) True

2) False

3) False

4) False

c)

(i)

```
knn.train <- as.matrix(train)
knn.test <- as.matrix(test)

set.seed(123)

knn.model <- knn(train = knn.train,
              test = knn.test,
              cl = train$class,
              k = 25, prob = TRUE)
```

(ii)

By testing several different values of K and plotting the fraction of correct classification (and/or other types of errors we would like to mimize), and choosing the appropriate K. By choosing large K, the variance will tend to be small, but the particular structure of the training set will strongly impact predictions, hence large bias. With small K our model will tend to have low bias and high variance.

d)

(i)

In this case we are interested in prediction. Predicting the wearabouts of bigfoot is, we assume, of great interest.

If we wanted to model for prediction the exact shape and form of our model would not be of interest, but rather the predictive power. Hence non-parametric models as KNN could be used if the test results were good enough.

If we wanted do inference the relationship of the response and predictiors would be of great importance. This rules out very flexible models.

(ii)

In the following confusion matrices the prediction make out the rows, and the actual values make out the columns.

```
#LogReg confusion matrix
logReg <- table(predict_glm, test$class)
logReg
```

```
##
## predict_glm   0   1
##           0 323 148
##           1 142 299
```

```
#LogReg sensitivity
logReg[2,2]/ sum(logReg[, 2])
```

```
## [1] 0.6689038
```

```
#LogReg specificity
logReg[1,1]/ sum(logReg[, 1])
```

```
## [1] 0.6946237
```

```r
#QDA confusion matrix
qdaTab <- table(predict_bigfoot_qda, test$class)
qdaTab
```

```
##
## predict_bigfoot_qda   0   1
##                   0 228  58
##                   1 237 389
```

```r
#QDA sensitivity
qdaTab[2,2]/ sum(qdaTab[, 2])
```

```
## [1] 0.8702461
```

```r
#QDA specificity
qdaTab[1,1]/ sum(qdaTab[, 1])
```

```
## [1] 0.4903226
```

```r
# KNN confusion matrix (k=25)
knnTab <- table(knn.model, test$class)
knnTab
```

```
##
## knn.model   0   1
##         0 386  85
##         1  79 362
```

```r
#KNN sensitivity
knnTab[2, 2] / sum(knnTab[, 2])
```

```
## [1] 0.8098434
```

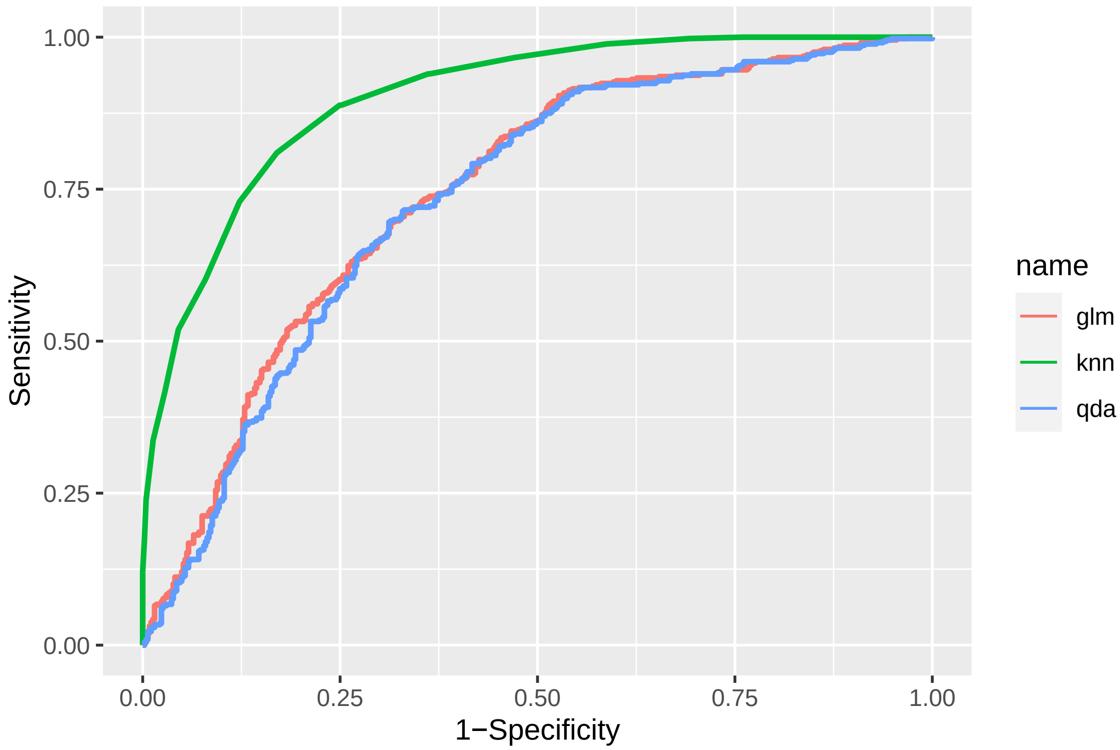
```r
#KNN specificity
knnTab[1, 1] / sum(knnTab[, 1])
```

```
## [1] 0.8301075
```

The sensitivity of a model is telling us how good the model is at classifiying positive observations. That is, the sensitivity is the proportion of correclty classified positive observations (True Positive/Actual positive). Likewise, the specificity of a model is telling us how well the model is doing when it comes to classyfying negative observations (True Negative/Actual Negative).

(iii)

```r
knn.probs <- attributes(knn.model)$prob

classB <- which(knn.model == 0)
knn.probs[classB] <- 1 - knn.probs[classB]

dat <- data.frame(class = test$class, glm = predict_bigfoot_glm,
                qda = prob_bigfoot_qda[, 2], knn = knn.probs)



dat_long <- melt_roc(dat, "class", c("glm", "qda", "knn"))
ggplot(dat_long, aes(d = D, m = M, color = name)) + geom_roc(n.cuts = F) +
  xlab("1-Specificity") + ylab("Sensitivity")
```




```r
glmroc <- roc(response = test$class,
              predictor = predict_bigfoot_glm,
              direction = "<")

qdaroc <- roc(response = test$class,
              predictor = prob_bigfoot_qda[, 2],
              direction = "<")

knnroc <- roc(response = test$class,
              predictor = knn.probs,
              direction = "<")
auc(glmroc)
```

```
## Area under the curve: 0.7458
```

```r
auc(qdaroc)
```

```
## Area under the curve: 0.7354
```

```r
auc(knnroc)
```

```
## Area under the curve: 0.9011
```

(iv)

From the ROC plot and the corresponding AUC scores we see that KNN performs better than both QDA and the logistic regression for all thresholds. Therefore we would choose KNN. If we were interested in inference, we would have to concider either the logistc regression or QDA. As the AUC score of these two models are close, our choice of model would depend on what type of errors we would like to minimize.

Problem 4

a)

Recall that the total prediction error for n observations, CV_n, is given by:

$$CV_n = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i$$

where $\text{MSE}_i = (y_i - \hat{y}_{-i})^2$. Here $\hat{y}_{-i}$ is the prediction made for the ith, excluded observation.

To prove that $CV_N = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_{-i})^2 = \frac{1}{N}\sum_{i=1}^{N}(\frac{y_i - \hat{y}_{-i}}{1-h_i})^2$, we will consider the expression $y_i - \hat{y}_{-i}$

Lets first define some necessary relations:

$$\begin{aligned}
\hat{\boldsymbol{\beta}} &= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \\
\hat{y}_{-i} &= \mathbf{x}_i^T\hat{\boldsymbol{\beta}}_{-i} \\
\boldsymbol{X}_{-i}^T\boldsymbol{X}_{-i} &= \boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{x}_i\boldsymbol{x}_i^T \\
\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i} &= \boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{x}_i y_i \\
h_i &= \boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i
\end{aligned}$$

Consider $\hat{y}_{-i}$:

$$\begin{aligned}
\hat{y}_{-i} = \mathbf{x}_i^T\boldsymbol{\beta}_{-i} &= \mathbf{x}_i^T(\boldsymbol{X}_{-i}^T\boldsymbol{X}_{-i})^{-1}\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i} \\
&= \mathbf{x}_i^T(\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{x}_i\boldsymbol{x}_i^T)^{-1}\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i} \\
&= \mathbf{x}_i^T\left[(\boldsymbol{X}^T\boldsymbol{X})^{-1} + \frac{(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}}{1 - \boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i}\right]\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i}
\end{aligned}$$

In the last equality we apply the Sherman-Morrison formula. Next we divide this expression into two part:

$$\begin{aligned}
\mathbf{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i} &= \mathbf{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{x}_i y_i) \\
&= \mathbf{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - \mathbf{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i y_i \\
&= \mathbf{x}_i^T\hat{\boldsymbol{\beta}} - h_i y_i = \hat{y}_i - h_i y_i
\end{aligned}$$

For the finale part of the expression:

$$\begin{aligned}
\boldsymbol{x}_i^T\frac{(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}}{1 - \boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i}\boldsymbol{X}_{-i}^T\boldsymbol{y}_{-i} &= \boldsymbol{x}_i^T\frac{(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}(\boldsymbol{X}^T\boldsymbol{y} - \boldsymbol{x}_i y_i)}{1 - h_i} \\
&= \frac{h_i\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} - h_i\boldsymbol{x}_i^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_i y_i}{1 - h_i} \\
&= \frac{h_i\hat{y}_i - h_i^2 y_i}{1 - h_i}
\end{aligned}$$

Adding these two expressions together we obtain:

$$\hat{y}_i - h_i y_i + \frac{h_i\hat{y}_i - h_i^2 y_i}{1 - h_i} = \frac{\hat{y}_i - h_i\hat{y}_i - h_i y_i + h_i^2 y_i + h_i\hat{y}_i - h_i^2 y_i}{1 - h_i} = \frac{\hat{y}_i - h_i y_i}{1 - h_i} = \hat{y}_{-i}$$

We were looking for en alternative expression for $y_i - \hat{y}_{-i}$:

$$
\begin{aligned}
y_i - \hat{y}_{-i} &= y_i - \frac{\hat{y}_i - h_i y_i}{1 - h_i} \\
&= \frac{y_i - h_i y_i - \hat{y}_i + h_i y_i}{1 - h_i} \\
&= \frac{y_i - \hat{y}_i}{1 - h_i}
\end{aligned}
$$

Finally, we conclude:

$$
CV_N = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_{-i})^2 = \frac{1}{N} \sum_{i=1}^{N} (\frac{y_i - \hat{y}_i}{1 - h_i})^2
$$

$\square$

b)

i) True *ii)* False *iii)* True *iv)* False