Compulsory exercise 1: Group 11

TMA4268 Statistical Learning V2023

Torbjørn Vatne, Ludvik Braathen and Johan Bjerkem

16 February, 2023

```
install.packages("knitr")  # probably already installed
install.packages("rmarkdown")  # probably already installed
install.packages("ggplot2")  # plotting with ggplot2
install.packages("dplyr")  # for data cleaning and preparation
install.packages("tidyr")  # also data preparation
install.packages("carData")  # dataset
install.packages("class")  # for KNN
install.packages("pROC")  # calculate roc
install.packages("plotROC")  # plot roc
install.packages("ggmosaic")  # mosaic plot
install.packages("knitr")
install.packages("formatR")
library(knitr)
install.packages("tinytex")
tinytex::install_tinytex()
```

Problem 1 (9P)

a) (1P)

Here we find the expected value:

$$
\begin{aligned}
E(\widetilde{\boldsymbol{\beta}}) &= E((\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{Y}) && (1)\\
&= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top E(\mathbf{Y}) && (2)\\
&= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top E(\mathbf{X}\boldsymbol{\beta} + \varepsilon) && (3)\\
&= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top(\mathbf{X}\boldsymbol{\beta} + \mathbf{0}) && (4)\\
&= (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} && (5)
\end{aligned}
$$

Here we find the variance-covarince matrix:

$$
\begin{aligned}
Cov(\widetilde{\boldsymbol{\beta}}) &= Cov((X^TX + \lambda I)^{-1}X^TY) && (6)\\
&= (X^TX + \lambda I)^{-1}X^T Cov(Y)((X^TX + \lambda I)^{-1}X^T)^T && (7)\\
&= (X^TX + \lambda I)^{-1}X^T\sigma^2 I((X^TX + \lambda I)^{-1}X^T)^T && (8)
\end{aligned}
$$

b) (2P)

First we find the expected value:

$$E(\mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}) = \mathbf{x}_0^T E(\widetilde{\boldsymbol{\beta}}) \tag{9}$$
$$= \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} \tag{10}$$

Then we find the variance: (Here we need to remember the rule for variance where we need to multiply the same term transposed at the back of the variance term.)

$$Var(\mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}) = \mathbf{x}_0^T Var(\widetilde{\boldsymbol{\beta}})\mathbf{x}_0 \tag{11}$$
$$= \mathbf{x}_0^T Var(\widetilde{\boldsymbol{\beta}})\mathbf{x}_0 \tag{12}$$
$$= \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\sigma^2((\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top)^\top\mathbf{x}_0 \tag{13}$$

c) (1P)

Explain with words how we can interpret the three terms: bias, variance and irreducible error.

Bias is the error that comes by oversimplifying the model. This means underfitting the model, meaning high bias will over simplify the relationships in the data. Variance, on the other hand, is the error that comes by creating a too complex mode. This means overfitting the model, where the model tries to hard to fit the function to the noise of the data. There is a famous plot of the relationship between bias and variance. When we train a model we start with a high bias that decays as we train the model. The variance starts low and increases as we train the model. We want to hit the sweet spot where bias and variance intercepts, as this is the point with the minimal total error. Irreducible error is the error that comes with the data. As the name suggests this error cannot be eliminated by any model, as it is inherit by the data and is not due to bias or variance.

d) (2P)

$$E[(y_0 - \widetilde{f}(\mathbf{x}_0))^2] = Var(\widetilde{f}(\mathbf{x}_0)) + (Bias(\widetilde{f}(\mathbf{x}_0)))^2 + Var(\varepsilon) \tag{14}$$

$$Var(\widetilde{f}(\mathbf{x}_0)) = \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\sigma^2((\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top)^\top\mathbf{x}_0 \tag{15}$$

$$(Bias(\widetilde{f}(\mathbf{x}_0)))^2 = (E(\mathbf{y}_0 - \widetilde{f}(\mathbf{x}_0)))^2 \tag{16}$$
$$(Bias(\widetilde{f}(\mathbf{x}_0)))^2 = (E(\mathbf{y}_0) - E(\widetilde{f}(\mathbf{x}_0)))^2 \tag{17}$$
$$(Bias(\widetilde{f}(\mathbf{x}_0)))^2 = (\mathbf{x}_0^T\boldsymbol{\beta} - \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta})^2 \tag{18}$$

$$Var(\varepsilon) = \sigma^2 \tag{19}$$

$$E[(y_0 - \widetilde{f}(\mathbf{x}_0))^2] = \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\sigma^2((\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top)^\top\mathbf{x}_0 + (\mathbf{x}_0^T\boldsymbol{\beta} - \mathbf{x}_0^T(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta})^2 + \sigma^2 \tag{20}$$

Plotting the bias-variance trade-off

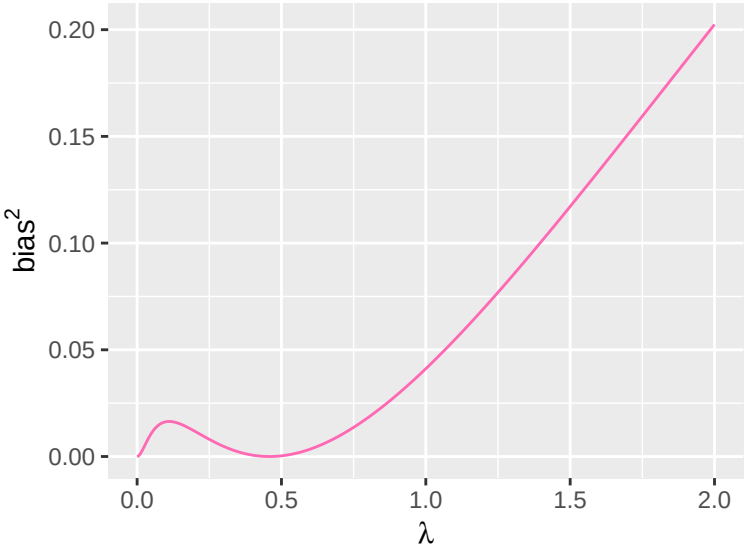
The estimator $\widetilde{\boldsymbol{\beta}}$ is a function of the tuning parameter λ, which controls the bias-variance trade-off. Using the decomposition derived in c) we will plot the three elements (bias, variance and irreducible error) using the

values in the code chunk below. `values` is a list with the design matrix `X`, the vector $\mathbf{x}_0$ as `x0`, the parameters vector `beta` and the irreducible error `sigma`.

```
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
```

e) (1P)

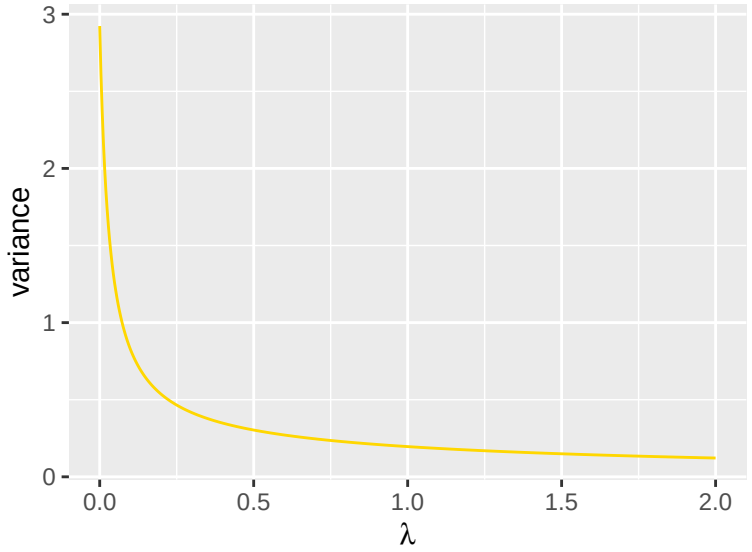
```
library(ggplot2)
bias <- function(lambda, X, x0, beta) {
    p <- ncol(X)
    inv <- solve(t(X) %*% X + lambda * diag(p))
    value <- (t(x0) %*% beta - t(x0) %*% inv %*% t(X) %*% X %*% beta)^2
    return(value)
}
lambdas <- seq(0, 2, length.out = 500)
BIAS <- rep(NA, length(lambdas))
for (i in seq_along(lambdas)) 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) (1P)

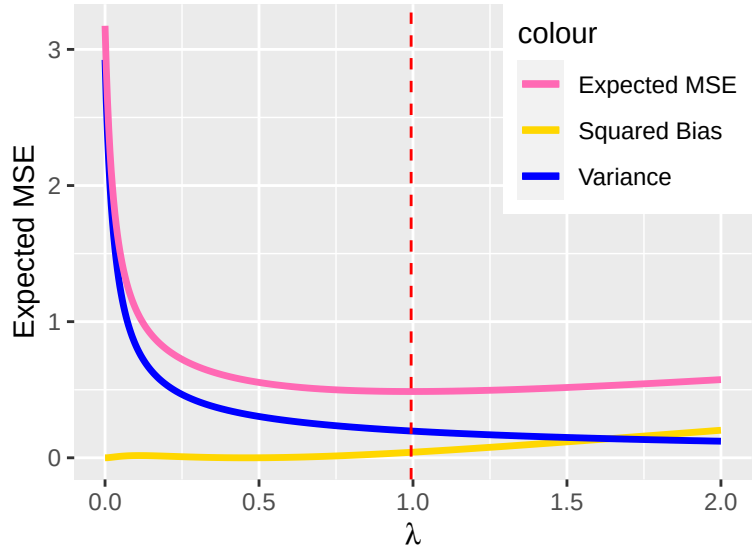
```
variance <- function(lambda, X, x0, sigma) {
    p <- ncol(X)
    inv <- solve(t(X) %*% X + lambda * diag(p))
    value <- sigma^2 * t(x0) %*% inv %*% t(X) %*% t(inv %*% t(X)) %*% x0
    return(value)
}
lambdas <- seq(0, 2, length.out = 500)
VAR <- rep(NA, length(lambdas))
```

```
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) (1P)

```
exp_mse <- BIAS + VAR + sigma^2
best_lambda <- lambdas[which.min(exp_mse)]
df <- merge(dfBias, dfVar, by = "lambdas")
df$exp_mse <- exp_mse
ggplot(df, aes(x = lambdas)) + geom_line(aes(y = bias, color = "Squared Bias"), linewidth = 1.2) +
    geom_line(aes(y = var, color = "Variance"), linewidth = 1.2) + geom_line(aes(y = exp_mse,
    color = "Expected MSE"), linewidth = 1.2) + geom_vline(xintercept = best_lambda,
    color = "red", linetype = "dashed") + scale_color_manual(values = c("hotpink",
    "gold", "blue")) + xlab(expression(lambda)) + ylab("Expected MSE") + theme(legend.position = c(1,
    1), legend.justification = c(1, 1))
```



Problem 2 (15P)

Bert-Ernie has his eye on a career in academia, but has not really decided on a field yet. Like most academics, his greatest concern is his future salary as a tenured professor. He comes across the `Salaries` dataset from the `carData R` package, and decides to perform a statistical analysis to try to find out how he can maximize his future salary as an academic.

To get more information about the covariates in the dataset, you should run `?Salaries` after loading the data. It's also a good idea to do a *descriptive analysis* of the data before fitting any models. A good starting point can be the `ggpairs()` function from the `GGally` package.

```
library(carData)
GGally::ggpairs(Salaries)
```

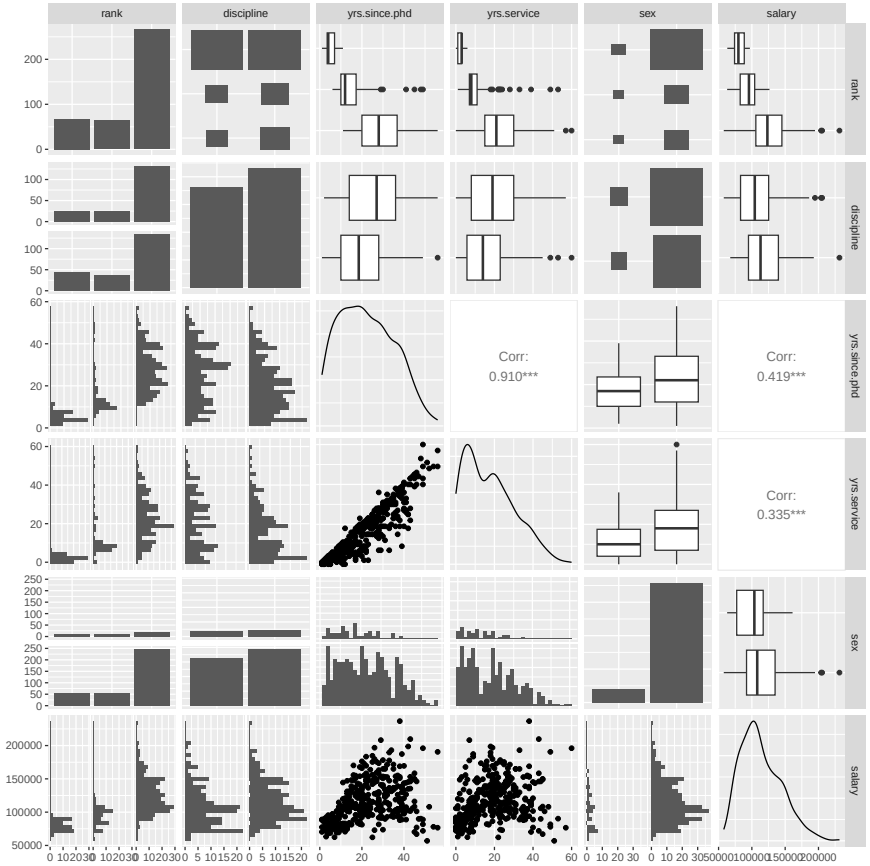


Figure 1: Pairs plot of the academic salary data set.

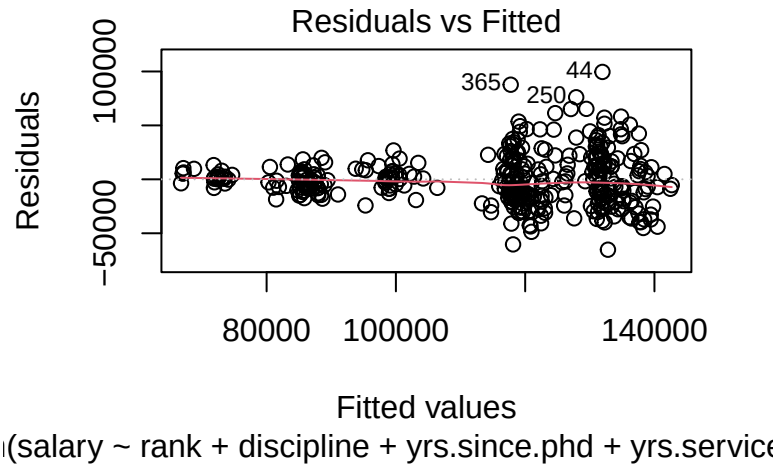
After doing a descriptive analysis, Bert-Ernie makes a multiple linear regression model with all covariates included, which he calls `model1`.

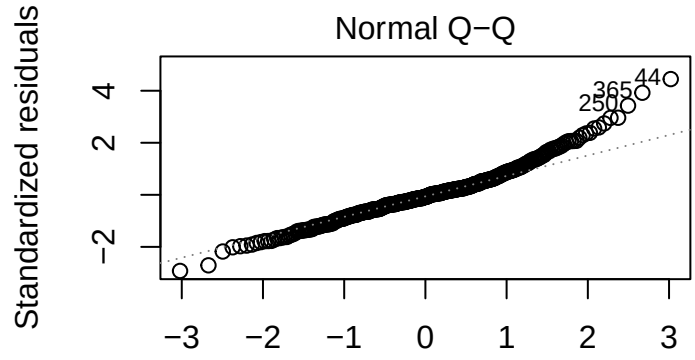
```
# Fit full model
`?`(Salaries)
model1 <- lm(salary ~ rank + discipline + yrs.since.phd + yrs.service + sex, data = Salaries)
summary(model1)
```

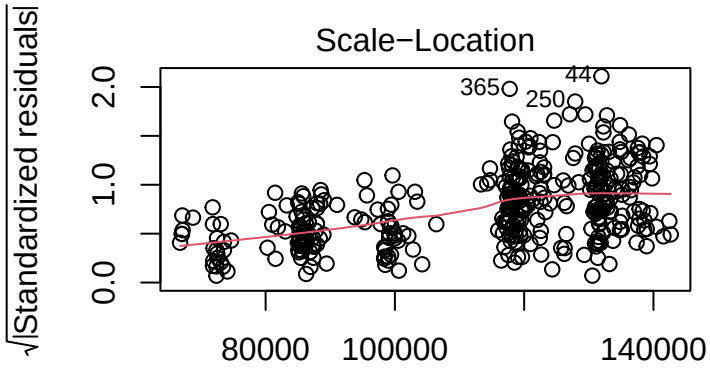
```
##
## Call:
## lm(formula = salary ~ rank + discipline + yrs.since.phd + yrs.service +
```

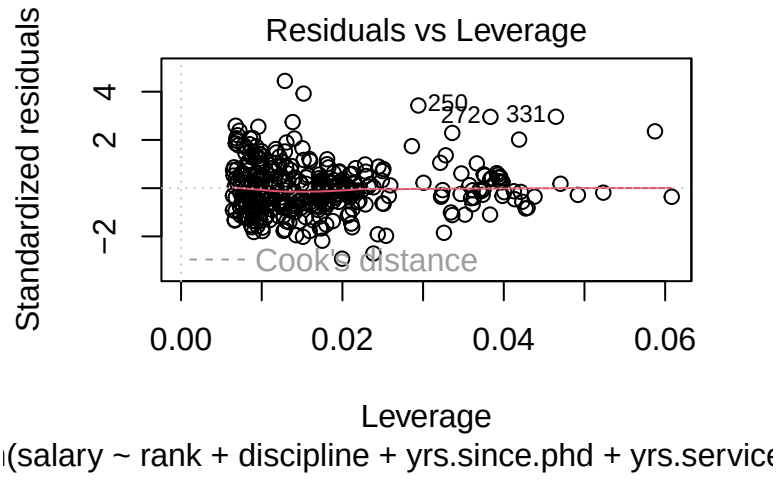
```
##     sex, data = Salaries)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -65248 -13211  -1775  10384  99592
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    65955.2     4588.6  14.374  < 2e-16 ***
## rankAssocProf  12907.6     4145.3   3.114  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
```

```
plot(model1)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(salary ~ rank + discipline + yrs.since.phd + yrs.service
365
250
44


Scale−Location
√|Standardized residuals|
Fitted values
lm(salary ~ rank + discipline + yrs.since.phd + yrs.service
365
250
44



a) Categorical variables (2P)

The covariate `rank` is categorical, with the three possible values `AsstProf` (assistant professor), `AssocProf` (associate professor) and `Prof` (full professor).

i) How does the `lm()` function encode for this type of covariate, and what are the interpretations of the `rankAssocProf` and `rankProf` estimates, respectively? (1P)

FIKS FORMATERING The *lm*() function by default uses the first of the three levels (of the covariate rank), `AsstProf`, as a reference level, and then converts the other levels, `AssocProf` and `Prof`, to a set of 'dummy variables'. These dummy variables are represented as binary variables. Naturally, only one of the levels can be 'active' at once, as a professor can only be either of the three different types of professors. The estimated coefficient for associate professor represents the difference in mean salary between associate professors and assistant professors (reference level), while the estimated coefficient for a full professor represents the difference in mean salary between full professors and assistant professors. In other words, associate professors are expected to earn at average 12907 more than an assistant professor, and full professors are expected to earn at average 45066 more than an assistant professor.

The output from `summary()` tells us the *p*-values of `rankAssocProf` and `rankProf`, but not `rank` as a whole.

ii) Perform an appropriate test to check whether there is evidence that `rank` as a whole has an impact on `salary`. The output from `summary()` is not enough to do this. (1P)

```
library(carData)
r.salary.rank <- lm(salary ~ yrs.service + yrs.since.phd + rank, data = Salaries)
r.salary.no.rank <- lm(salary ~ yrs.service + yrs.since.phd, data = Salaries)
anova(r.salary.rank, r.salary.no.rank)
```

```
## Analysis of Variance Table
##
## Model 1: salary ~ yrs.service + yrs.since.phd + rank
## Model 2: salary ~ yrs.service + yrs.since.phd
##   Res.Df        RSS Df   Sum of Sq      F    Pr(>F)
## 1    392 2.1839e+11
## 2    394 2.9487e+11 -2 -7.6488e+10 68.648 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we've done a comparison of two models, Model 1 and Model 2. In Model 1 we've included the variables: `yrs.service`, `yrs.since.PhD`, and `rank`. In Model 2 we also include `yrs.service` and `yrs.since.PhD`, but excluded the `rank`-variable. Running *anova*() on these two models, gives us an F-value of 68.648 and a p-value of 2.2e-16. Determining which values of an F-test that are considered 'good' is not easy. In this case, with a relatively high F-value, together with a p-value which is practically 0, we consider that there is enough evidence that `rank` as a whole does have an impact on `salary`.

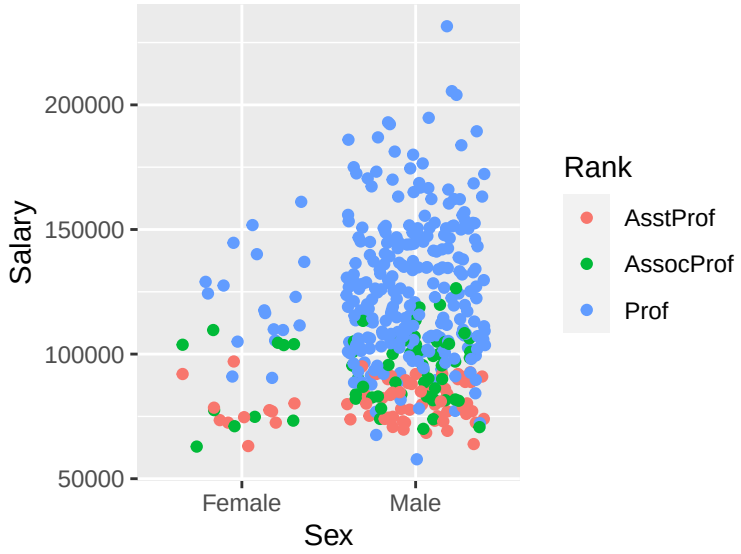
b) Conflicting model results? (2P)

Having heard of the gender wage-gap, Bert-Ernie is surprised to see that `model1` finds no evidence that `sex` has an effect on `salary`. To make sure nothing has gone wrong, he fits a new model with `sex` as the only covariate (along with an intercept).

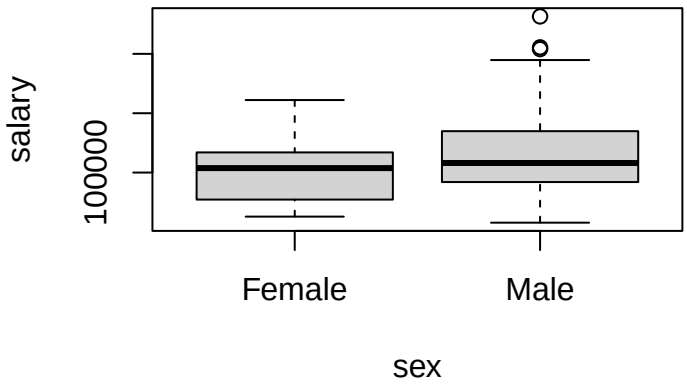
```
library(ggplot2)
sex_model <- lm(salary ~ sex, data = Salaries)
summary(sex_model)
```

```
##
## 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
```

```
ggplot(Salaries, aes(x = sex, y = salary, color = rank)) + geom_jitter() + labs(x = "Sex",
    y = "Salary", color = "Rank") + scale_x_discrete(labels = c("Female", "Male"))
```




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



The new model finds strong evidence that `sex` has an effect on `salary`. How would you explain the fact that `sex_model` finds evidence of `sex` impacting `salary`, but the full model `model1` does not? Make one or two figures to demonstrate your explanation, with proper figure captions explaining the figure(s). (2P)

SVAR TIL B): A possibility is that the relationship between sex and salary is confounded with other variables in the full model. This might happen when another variable is associated with both the predictor (sex) and the outcome (salary), causing the apparent relationship between the predictor and outcome to 'lie'. To try and demonstrate this, we first created a a scatter-plot which shows the salary of men and women, with different colors representing the different levels of rank. We also created a box-plot of the salary for men and women, but without consideration of the rank. The box-plot clearly shows that there's a difference between the salary for men and women. Though when we compare the box-plot to the scatter-plot, we see.....

Hint: The descriptive analysis might be useful to identify what is going on.

c) Model assumptions and transformation (2P)

Checking the assumptions of `model1`, Bert-Ernie observes that the assumptions of the linear regression model are not fulfilled in `model1`.

```
library(ggplot2)
library(ggfortify)
autoplot(model1, smooth.colour = NA)
```

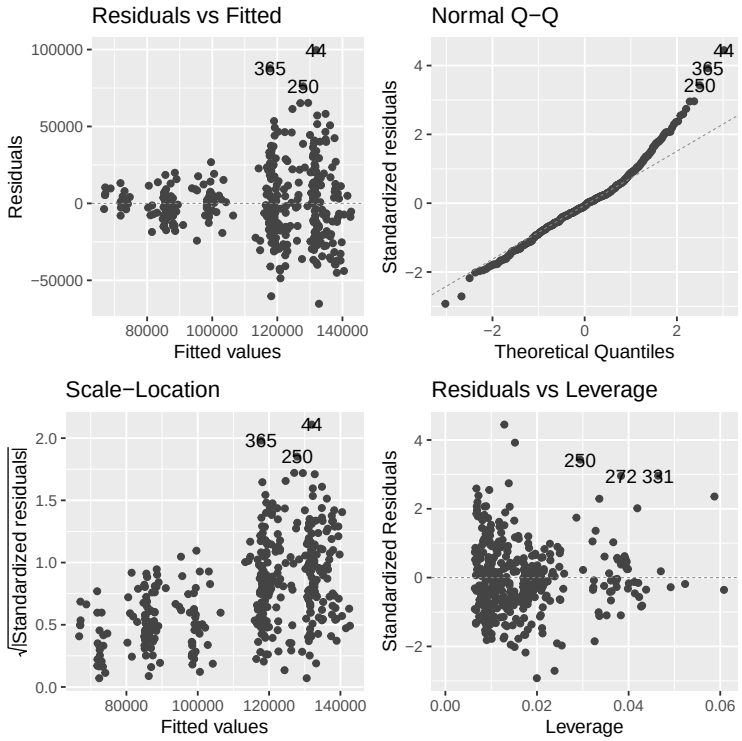


Figure 2: Diagnostic plots for 'model1'.

i) Based on the figure below, which assumptions are not fulfilled? (1P)

Residuals vs fitted: We can clearly see that the expected value of the residuals are zero, which fulfills the first assumption. It looks like the variance of the residuals increase as the salary/fitted values increase, which does not fulfill the second assumption.

Normal Q-Q: From this plot the residuals seems fairly normal distributed, at least up until. . . . . .

Scale-location: We again see a pattern where the variance of the residuals increase as the salary/fitted values increase, which can confirm our earlier conclusion about the second assumption not being fulfilled.

Residual vs leverage: There is a couple of outliers with high leverage, Cooks-distance. . . . . .

ii) Bert-Ernie has heard that sometimes such issues may be solved by transforming the response or covariates of the model, so he decides to try treating the log-transform of `salary` as the response in a new model. Implement the new model, and call it `model2`. Have the model assumptions improved in `model2` compared to `model1`? (1P)

```
log_salary <- log(Salaries$salary)
# Får feilmelding under knitting (pga log_salary elns) dersom jeg skriver det
# slik: model2 <- lm(log_salary ~ . -salary)
model2 <- lm(log_salary ~ rank + discipline + yrs.since.phd + yrs.service + sex -
```

```
    salary, data = Salaries)
summary(model2)
```

```
##
## Call:
## lm(formula = log_salary ~ rank + discipline + yrs.since.phd +
##     yrs.service + sex - 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
```

```
library(ggplot2)
autoplot(model2, smooth.colour = NA)
```

d) Interactions (2P)

Bert-Ernie hypothesizes that the impact of `sex` on salary in academia might be stronger in older generations. While keeping the log-transformed response and all the other covariates in the model he therefore decides to design a model with an interaction term between `sex` and `yrs.since.phd`.

i) Implement the model with the interaction term, and call it `model3` (1P).

```
model3 <- lm(log_salary ~ rank + discipline + yrs.since.phd + yrs.service + sex +
    sex:yrs.since.phd - salary, data = Salaries)
summary(model3)
```

```
##
## Call:
## lm(formula = log_salary ~ rank + discipline + yrs.since.phd +
##     yrs.service + sex + sex:yrs.since.phd - salary, 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 ***
```

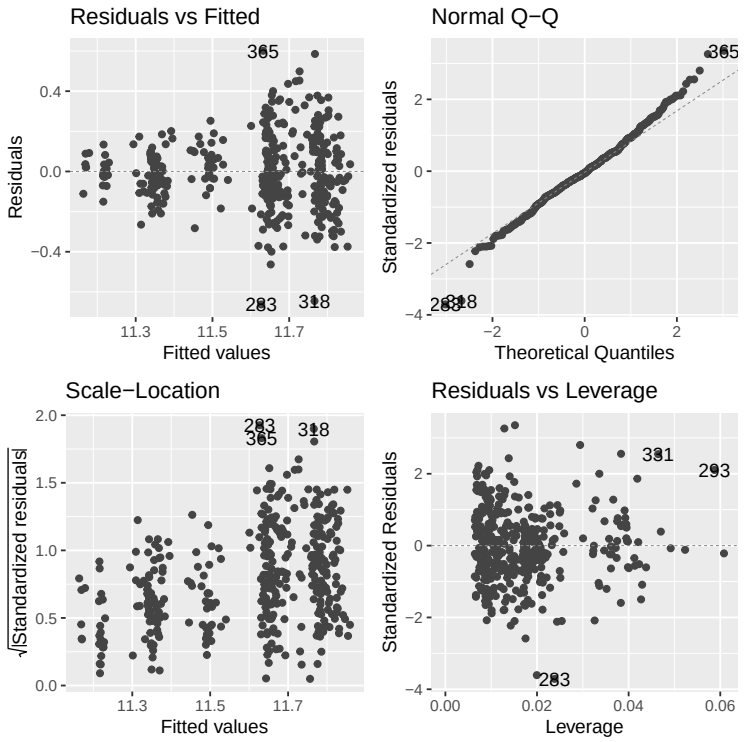


Figure 3: Diagnostic plots for 'model2'.

```
## 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) Interpret the results from `model3`. Is Bert-Ernie's hypothesis correct? (1P)

From the summary of the linear model, we can see that the p-value for the interaction term is very high, and we conclude with Bert-Ernie's hypothesis being incorrect.

e) Bootstrap (4P)

You might have noticed that the R^2 we get from `summary()` is just a point estimate, without any uncertainty associated to it. We can use the bootstrap to estimate the uncertainty of the R^2 in our models. To this end:

(i) Generate 1000 bootstrap samples of the R^2 in `model1`. Use `set.seed(4268)` at the beginning of the bootstrap iterations to ensure that your results are reproducible (2P).

```
set.seed(4268)
```

```r
# Function to extract the R^2-value from the linear model
getR2 <- function(data, indices) {
    fit <- lm(salary ~ rank + discipline + yrs.since.phd + yrs.service + sex, data = data[indices,
        ])
    summary(fit)$r.squared
}

library(boot)
boot_results <- boot(data = Salaries, statistic = getR2, R = 1000, strata = Salaries$rank)
boot_results
```

```
##
## STRATIFIED BOOTSTRAP
##
##
## Call:
## boot(data = Salaries, statistic = getR2, R = 1000, strata = Salaries$rank)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 0.4546766 0.009072358  0.02803583
```

(ii) Plot the respective distribution of the values (0.5P).
(iii) Find the standard error and the 95% quantile interval (1P).
(iv) Interpret what you see (0.5P).

Throughout, show your R code and printouts.

f) Picking a field (3P)

We (and Bert-Ernie) now assume that `model1` is the correct (that is, the data-generating) model. After some character development (namely, all this R coding is giving him a headache) Bert-Ernie has figured out that he would rather work in a theoretical field than an applied field. But, money-conscious as always, he decides that he will follow this dream if and only if his model predicts that 20 years after finishing his PhD he will be earning *at least* 75 000$ every nine months, with 95% certainty. He does not care how much he earns, as long as it is more than 75 000$. He is sure that he will get a permanent position immediately upon finishing his PhD (so his `yrs.since.phd` will equal his `yrs.service`), that he will have become full professor within the 20 years and that his sex will still be male.

```r
# 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"))
# Use the full model to predict his salary
preds <- predict(object = model1,
                 newdata = bert_ernie,
                 interval = "confidence",
                 level = 0.975) # Not 0.95 since we don't care about upper limit
# Check predictions
preds
```

```
##        fit      lwr      upr
```

```
## 1 116715.6 110985.0 122446.2
## 2 131133.2 126145.6 136120.8
```

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

```
## [1] TRUE
```

He sees that the lower limit of the computed interval beats 75 000$ by a fair amount, so he breathes a sigh of relief. However, he has made two errors in the call to `predict()`.

i) Correct his mistakes, and make the calculation he was actually interested in, still using `predict()`. Will he still be able to follow his dream after making the correction, or are many sleepless nights of debugging R code looming in his future? (1P)

ii) Write down an analytic expression for the correct lower limit, and implement the analytic expression in R to make sure you get the same result as from `predict()`. (2P)

Hint: see Recommended Exercises Week 3.

R-hint: In the implementation you might have use for the functions `model.matrix()`, `coef()`, `sigma()`, `df.residual()`, `qt()`, `sqrt()`, `solve()` and `t()`, and the matrix multiplication operator `%*%`.

Problem 3 (13P)

```
bigfoot_original <- readr::read_csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/maste
library(dplyr)
# 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) (2P)

```
glm.bigfoot = glm(class ~ ., data = train, family = binomial)
glm.probs <- predict(glm.bigfoot, test, type = "response")
glm.pred <- rep(0, 912)
```

```
glm.pred[glm.probs > 0.5] <- 1
table(glm.pred)
```

```
## glm.pred
##   0   1
## 471 441
```

```
print("441 sights classified as clear sightings.")
```

```
## [1] "441 sights classified as clear sightings."
```

(ii) (1P) Single choice:

4) We multiply by 3.641.

b) (3P)

```
require(MASS)
qda.fit <- qda(class ~ ., data = train)

qda.preds <- predict(qda.fit, test)
table(qda.preds$class)
```

```
##
##   0   1
## 286 626
```

```
print("626 sights classified as clear sightings.")
```

```
## [1] "626 sights classified as clear sightings."
```

(ii) (2P)

True, False, False, False

c) (2P)

```
require(class)
knnMod <- knn(train = train, test = test, cl = train$class, k = 25, prob = TRUE)
probKNN <- ifelse(knnMod == 0, 1 - attributes(knnMod)$prob, attributes(knnMod)$prob)

table(knnMod)
```

```
## knnMod
##   0   1
## 471 441
```

```
print("441 sights classified as clear sightings.")
```

```
## [1] "441 sights classified as clear sightings."
```

(ii) (1P)

Choosing an optimal value for k comes down to the bias-variance trade-off. A low k will lead to a flexible fit, for example for $k = 1$ all the model will interpolate all of the training data. Here the variance is high and the bias is low. Increasing k will increase to higher bias and lower variance. This means the outliers will effect the model less.

There are no predefined statisticial models for choosing k.I would create a plot between different k values and their errors. Based on this plot i would choose the k that leads to the lowest error.

d) (6P)

We now wish to compare the performance of the three models (logistic regression, QDA and KNN) on this dataset, in order to report back to BFRO on our results.

i) (2P) In this case, are we interested in prediction or inference? What implications would it have for the model choice if we wanted to do prediction, and what implications would it have if we wanted to do inference? In conclusion, does the question of whether our aim is prediction or inference exclude any of the three candidate models in this case? (comment briefly)

Prediction: Given a new measurement, we want to use an existing data set to build a model that reliably chooses the correct classification. If we wanted to do inference we would be interested in the interpretability of the model. In the case of inference QDA and KNN would be less interesting models. As the goal is prediction all the models are interesting, but we would choose the model with the lowest error for our predicting (our a combination of the models that produces the best result).

ii) (3P)

Confusion matrix for logistic regression:

```
table(glm.pred, test$class)
```

```
##
## glm.pred   0   1
##        0 323 148
##        1 142 299
```

```
299/(299 + 148)
```

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

```
323/(323 + 142)
```

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

Sensitivity = 299 / (299 + 148) = 66.9% Specificity = 323 / (323 + 142) = 69.5%

Confusion matrix for QDA:

```
table(qda.preds$class, test$class)
```

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

```
389/(389 + 58)
```

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

```
228/(228 + 237)
```

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

Sensitivity = 389 / (389 + 58) = 87.0% Specificity = 228 / (228 + 237) = 49.0%

Confusion matrix for KNN:

```
table(knnMod, test$class)
```

```
##
## knnMod   0   1
##      0 386  85
##      1  79 362
```

```
362/(362 + 85)
```

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

```
386/(386 + 79)
```

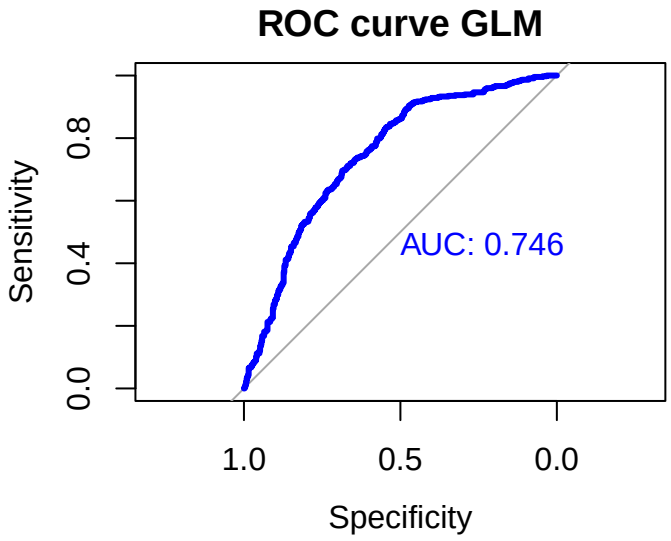
```
## [1] 0.8301075
```

Sensitivity = 362 / (362 + 85) = 81.0% Specificity = 386 / (386 + 79) = 83.0%

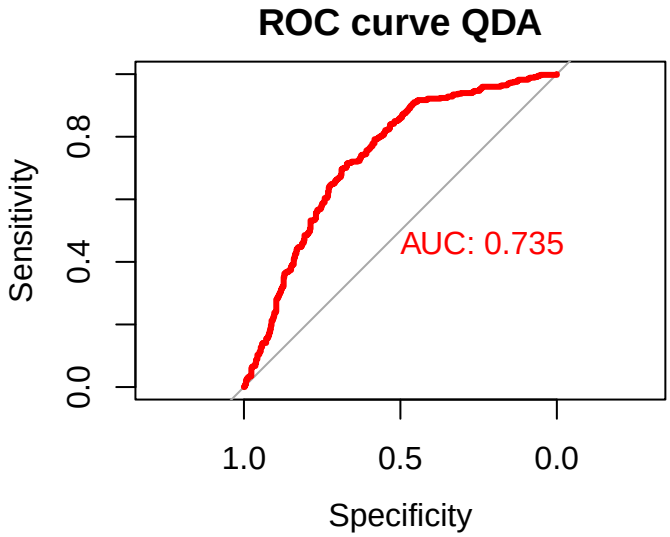
Sensitivity means how often is a true bigfoot sighting actually classified as Class A by the model. Specificity mean how often is a false bigfoot sighting actually classified as Class B by the model. Both are measures of accuracy. Good percentages for each score depends on the situation, there is no given standard that is universal.

iii) (1P).

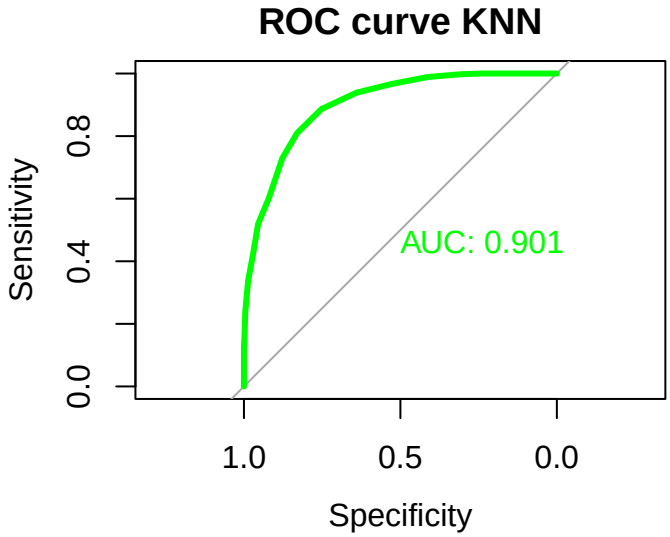
```
library(pROC)


roc_glm <- roc(response = test$class, predictor = glm.probs)
plot(roc_glm, col = "blue", lwd = 3, print.auc = TRUE, main = "ROC curve GLM")
```




```
roc_qda <- roc(response = test$class, predictor = qda.preds$posterior[, "1"])
plot(roc_qda, col = "red", lwd = 3, print.auc = TRUE, main = "ROC curve QDA")
```




```
roc_knn <- roc(response = test$class, predictor = probKNN)
plot(roc_knn, col = "green", lwd = 3, print.auc = TRUE, main = "ROC curve KNN")
```



iv) (1P)

QDA and logistic regression performs pretty equal, but the ROC plot for KNN is substantially better. Because of this i would choose the KNN model.

Problem 4 (6P)

a) (4P)

To calculate the LOOCV statistic when we have N training data points we typically have to fit N models and evaluate the error of each fit on its left out observation. This task can be computationally intensive when N is large. Fortunately, for linear models there is formula for doing this with fitting just the full data model.

Show that for the linear regression model $Y = \mathbf{X}\beta + \varepsilon$ the LOOCV statistic can be computed by the following formula

$$\mathrm{CV} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2 \ ,$$

where $h_i = \mathbf{x}_i^\top \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{x}_i$ and $\mathbf{x}_i^\top$ is the ith row of $\mathbf{X}$.

Hints:

1. You need to calculate $\hat{y}_{(-i)}$, which is the predicted value of y when the ith observation is kept out. This can be written as $\hat{y}_{(-i)} = \mathbf{x}_i^\top \hat{\boldsymbol{\beta}}_{(-i)}$.
2. $\mathbf{X}_{(-i)}^\top \mathbf{X}_{(-i)} = \mathbf{X}^\top \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^\top$, where $\mathbf{X}_{(-i)}$ is the $\mathbf{X}$ matrix without the ith row.
3. Analogously $\mathbf{X}_{(-i)}^\top \mathbf{y}_{(-i)} = \mathbf{X}^\top \mathbf{y} - \mathbf{x}_i y_i$.
4. The Sherman–Morrison formula will be useful.

We have:

$$\hat{y}_{(-i)} = \mathbf{x}_i^\top \hat{\boldsymbol{\beta}}_{(-i)} \tag{21}$$

$$\hat{\boldsymbol{\beta}}_{(-i)} = (X_{(-i)}^T X_{(-i)})^{-1} X_{(-i)}^T Y_{(-i)} \tag{22}$$

Now we can find the LOOCV statistic as follows:

$$CV = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{23}$$

$$= (X_{(-i)}^T X_{(-i)})^{-1} X_{(-i)}^T Y_{(-i)} \tag{24}$$

b) Multiple choice (2P)

Say for each statement below whether it is true or false.

(i) The LOOCV will lead to more bias, but less variance than 10-fold CV in the estimated prediction error.
(ii) The formula from **a)** is not valid for polynomial regression.
(iii) The formula from **a)** is valid when we use the log transform of the response in linear regression.
(iv) The validation set-approach is the same as 2-fold CV.