Compulsory exercise 1: Group 39

TMA4268 Statistical Learning V2021

Alexander J Ohrt, Jim Totland

14 February, 2021

Problem 1

a)

By extending the given univariate regression problem to a multivariate regression problem that allows for several observations, we have that $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \varepsilon I$, where I is the identity matrix of the correct dimensions. Hence, we have $\mathsf{E}(\mathbf{Y}) = \mathsf{E}(\mathbf{X}\boldsymbol{\beta} + \varepsilon I) = \mathbf{X}\boldsymbol{\beta}$ and $\mathsf{Cov}(\mathbf{Y}) = \sigma I$, without assuming anything else than that each of the ε's are independent of each other. Then

$$\mathsf{E}(\widetilde{\boldsymbol{\beta}}) = \mathsf{E}((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}) = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathsf{E}(\mathbf{Y}) = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}$$

and

$$\begin{aligned}\mathsf{Cov}(\widetilde{\boldsymbol{\beta}}) &= \mathsf{Cov}((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}) = ((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T)\mathsf{Cov}(\mathbf{Y})((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T)^T \\ &= ((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T)\sigma^2 I\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-T} = \sigma^2((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-T}) \\ &= \sigma^2((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}),\end{aligned}$$

where we have used that $(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-T}$ in the last equality. In both these equations it is apparent that the moments are equal to those of the OLS estimator when $\lambda = 0$.

b)

The requested moments of $\widetilde{f}(\mathbf{x}_0) = \mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}$ are

$$\mathsf{E}(\widetilde{f}(\mathbf{x}_0)) = \mathsf{E}(\mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}) = \mathbf{x}_0^T\mathsf{E}(\widetilde{\boldsymbol{\beta}}) = \mathbf{x}_0^T(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}$$

and

$$\begin{aligned}\mathsf{Cov}(\widetilde{f}(\mathbf{x}_0)) &= \mathsf{Cov}(\mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}) = \mathbf{x}_0^T\mathsf{Cov}(\widetilde{\boldsymbol{\beta}})\mathbf{x}_0 \\ &= \sigma^2\mathbf{x}_0^T((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1})\mathbf{x}_0.\end{aligned}$$

c)

The expected MSE at $\mathbf{x}_0$ is

$$\begin{aligned}
\mathsf{E}[(y_0 - \widetilde{f}(\mathbf{x}_0))^2] &= [\mathsf{E}(\widetilde{f}(\mathbf{x}_0) - f(\mathbf{x}_0)]^2 + \mathsf{Var}(\widetilde{f}(\mathbf{x}_0)) + \mathsf{Var}(\varepsilon) \\
&= [\mathsf{E}(\widetilde{f}(\mathbf{x}_0)) - \mathsf{E}(f(\mathbf{x}_0))]^2 + \mathsf{Cov}(\widetilde{f}(\mathbf{x}_0)) + \sigma^2 \\
&= [\mathbf{x}_0^T(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} - \mathbf{x}_0^T\boldsymbol{\beta}]^2 + \sigma^2\mathbf{x}_0^T((\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1})\mathbf{x}_0 + \sigma^2.
\end{aligned}$$

Since there is no obvious way to simplify this, it will be left like this. This is also practical since it is easy to distinguish between the irreducible error, the variance of prediction and the squared bias.

d)

```
id <- "1X_8OKcoYbng1XvYFDirxjEWr7LtpNr1m" # google file ID
values <- dget(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
X = values$X
dim(X)
```

```
#> [1] 100  81
```

```
x0 = values$x0
dim(x0)
```

```
#> [1] 81  1
```

```
beta=values$beta
dim(beta)
```

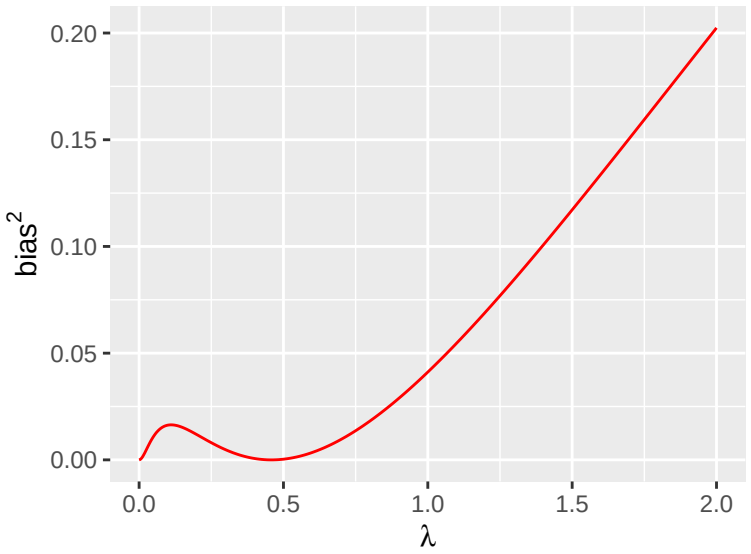
```
#> [1] 81  1
```

```
sigma=values$sigma
sigma
```

```
#> [1] 0.5
```

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

  xlab(expression(lambda))+
  ylab(expression(bias^2))
```

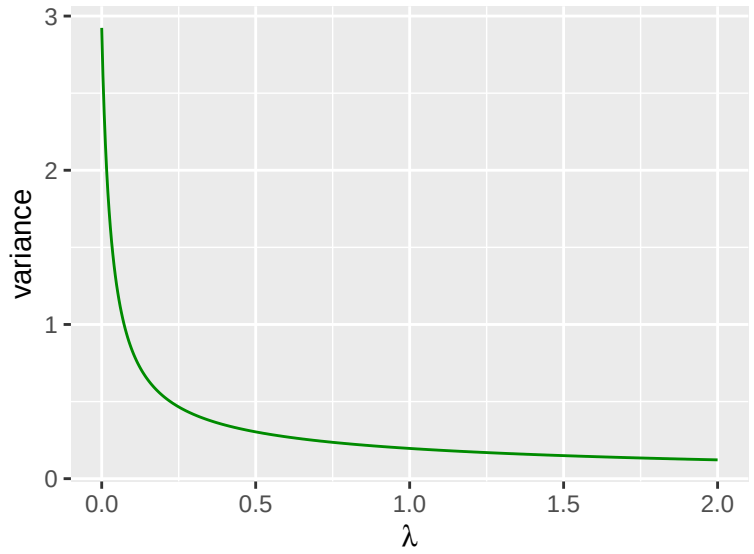


Comments: The graph shows that the bias of the ridge regression estimator increases as λ grows. OLS is unbiased, so, as expected, the bias is zero when $\lambda = 0$. Note that $\lambda \approx 0.5$ appears to be a sweet spot for this estimator when comparing all $\lambda > 0$, since the bias is low there. Perhaps this can be useful later.

e)

```
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 1:length(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 = "green4")+

  xlab(expression(lambda))+
  ylab("variance")
```



Comments: The variance begins at ≈ 2.923 and decreases with λ. Hence, it is apparent that the ridge regression estimator is advantageous, when looking at solely variance, compared to the OLS estimator, since the variance is decreasing with λ. Despite this, when adding the bias and the variance, the OLS estimator may still have a lower expected MSE than the ridge regression estimator. Finally, note that the changes in the variance are larger for $\lambda \in [0, 2]$ than the changes in the bias (as seen in the plot in task d)), which indicates that the variance dominates the change in expected MSE for the ridge regression estimation.

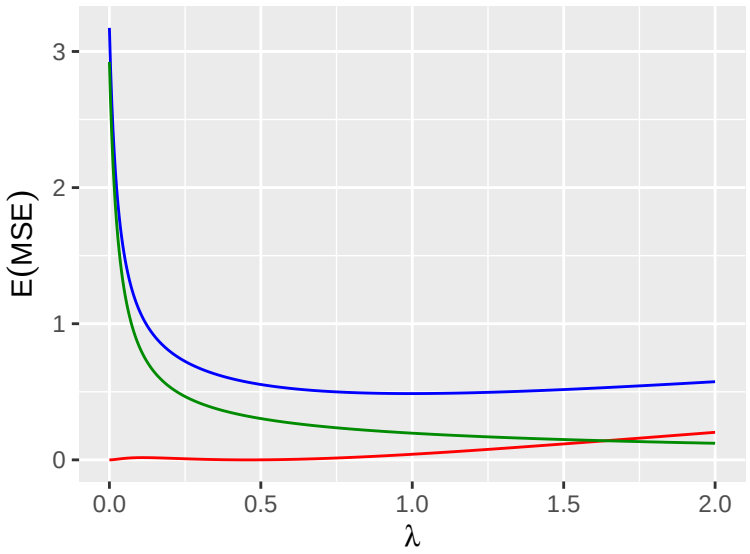
f)

```
exp_mse = BIAS + VAR + sigma^2
lambdas[which.min(exp_mse)]
```

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

```
dfAll = data.frame(lambda = lambdas, bias = BIAS, var = VAR, exp_mse = exp_mse)
ggplot(dfAll)+
  geom_line(aes(x = lambda, y = exp_mse), color = "blue")+

  geom_line(aes(x = lambda, y = bias), color = "red")+
  geom_line(aes(x = lambda, y = var), color = "green4")+
    xlab(expression(lambda))+
    ylab(expression(E(MSE)))
```



Comments: Now we are able to conclude that the ridge regression estimator has a lower expected MSE compared to the OLS estimator, as gathered from the blue line in the plot. The optimal value of λ, which minimizes MSE, is ≈ 0.994. Hence, despite the fact that the bias is higher for the ridge regression estimator, the total expected MSE is lower, since the decrease in variance counters the increase in bias.

Problem 2

```
# read file
id <- "1yYlEl5gYY3BEtJ4d7KWaFGIOEweJIn__" # google file ID
d.corona <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id),header=T)
```

a)

The tables are reported below.

```
# knitr::kable(table(d.corona$deceased)) # Hvorfor ta bort de som fungerer?

table1 <- summarise(d.corona, deceased = sum(d.corona$deceased),
                    non_deceased = nrow(d.corona) - deceased)
knitr::kable(table1)
```

deceased	non_deceased
105	1905

```
table2 <- table(d.corona$country, d.corona$sex)
knitr::kable(table2)
```

	female	male
France	60	54
indonesia	30	39
japan	120	174
Korea	879	654

```r
table3 <- table(d.corona$deceased, d.corona$sex)
rownames(table3) = c("non-deceased", "deceased")
knitr::kable(table3)
```

	female	male
non-deceased	1046	859
deceased	43	62

```r
d.france <- filter(d.corona, country == "France")
table4 <- table(d.france$deceased, d.france$sex)
rownames(table4) = c("non-deceased", "deceased")
knitr::kable(table4)
```

	female	male
non-deceased	55	43
deceased	5	11

b)

```r
glm.fit <- glm(deceased ~ ., family = "binomial", data = d.corona)
summary(glm.fit)
```

```
#>
#> Call:
#> glm(formula = deceased ~ ., family = "binomial", data = d.corona)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -0.9050  -0.3508  -0.2761  -0.2144   3.1165
#>
#> Coefficients:
#>                   Estimate Std. Error z value Pr(>|z|)
#> (Intercept)      -3.993485   0.462190  -8.640  < 2e-16 ***
#> sexmale           0.626068   0.209045   2.995  0.00275 **
#> age               0.027134   0.004736   5.729 1.01e-08 ***
#> countryindonesia -0.411855   0.550051  -0.749  0.45400
#> countryjapan     -1.343383   0.417196  -3.220  0.00128 **
#> countryKorea     -0.773895   0.307980  -2.513  0.01198 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance: 824.32  on 2009  degrees of freedom
#> Residual deviance: 766.16  on 2004  degrees of freedom
#> AIC: 778.16
#>
#> Number of Fisher Scoring iterations: 6
```

(i) The probability to die of covid for a male age 75 living in Korea can be predicted from the model. The prediction is found by

```r
x0 = data.frame(sex = "male", age = 75, country = "Korea")
predict(glm.fit, newdata = x0, type = "response")
```

```
#>         1
#> 0.1084912
```

(ii) The p-value associated with `sexmale` is relatively small, and since the coefficient is positive, this constitutes evidence that males have a higher probability of dying. ER DU UENIG?? (but we would not say that there is clear evidence that males have higher probability to die than females. The p-value could be low by chance also, and since it is not amazingly small, we do not think it is appropriate evidence of the question.)

(iii) Yes. Both the `countryjapan` and `countryKorea` coefficients have low p-values and are negative, which evidence that the probabilty of decease is lower in Japan and Korea compared to the reference category, France. `countryIndonesia` is not significant (high p-value), so there is no evidence that the probability of decease is any higher in Indonesia than in France. HVA TENKER DU?

(iv) Since we have used logistic regression, the odds of decease, given an observation $\boldsymbol{x}$ is

$$\frac{p(\boldsymbol{x})}{1-p(\boldsymbol{x})} = e^{\boldsymbol{x}^T\hat{\boldsymbol{\beta}}}.$$

Thus, the odds of decease increases by a factor of $e^{10\hat{\beta}_{\texttt{age}}} \approx 1.31$ when `age` increases by 10 and all other covariates are held constant.

c)

```r
log.fit1 <- glm(deceased ~. + sex:age, data = d.corona, family="binomial")
summary(log.fit1)
```

```
#>
#> Call:
#> glm(formula = deceased ~ . + sex:age, family = "binomial", data = d.corona)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -0.9111  -0.3500  -0.2768  -0.2150   3.1078
#>
#> Coefficients:
#>                    Estimate Std. Error z value Pr(>|z|)
#> (Intercept)      -3.953580   0.571712  -6.915 4.67e-12 ***
#> sexmale           0.556745   0.624834   0.891 0.372913
#> age               0.026485   0.007261   3.648 0.000265 ***
#> countryindonesia -0.410197   0.550304  -0.745 0.456030
#> countryjapan     -1.344440   0.417364  -3.221 0.001276 **
#> countryKorea     -0.772596   0.308244  -2.506 0.012195 *
#> sexmale:age       0.001111   0.009443   0.118 0.906350
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance: 824.32  on 2009  degrees of freedom
#> Residual deviance: 766.15  on 2003  degrees of freedom
#> AIC: 780.15
```

```
#>
#> Number of Fisher Scoring iterations: 6
```

(i) As the summary shows, the interaction effect between sexmale and age is not significant. Hence, although the coefficient is slightly positive, we cannot conclude that age is a greater risk factor for men. (As seen above, the coefficient for male is positive (0.5567454), which means that a male has higher risk than a female. This is because the value of this coefficient is added on top of the interceipt, which corresponds to an individual that is in the reference level, i.e. a female.)

```
log.fit2 <- glm(deceased ~ . + age:country, data = d.corona, family="binomial")
summary(log.fit2)
```

```
#>
#> Call:
#> glm(formula = deceased ~ . + age:country, family = "binomial",
#>     data = d.corona)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -1.2681  -0.3447  -0.2768  -0.2180   3.0170
#>
#> Coefficients:
#>                       Estimate Std. Error z value Pr(>|z|)
#> (Intercept)          -7.04272    1.72789  -4.076 4.58e-05 ***
#> sexmale               0.62777    0.21056   2.981  0.00287 **
#> age                   0.06693    0.02124   3.151  0.00163 **
#> countryindonesia      4.34249    2.16594   2.005  0.04497 *
#> countryjapan          2.13091    2.03299   1.048  0.29456
#> countryKorea          2.37162    1.75357   1.352  0.17623
#> age:countryindonesia -0.07189    0.03310  -2.172  0.02986 *
#> age:countryjapan     -0.04630    0.02668  -1.736  0.08260 .
#> age:countryKorea     -0.04142    0.02189  -1.892  0.05854 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance: 824.32  on 2009  degrees of freedom
#> Residual deviance: 759.71  on 2001  degrees of freedom
#> AIC: 777.71
#>
#> Number of Fisher Scoring iterations: 6
```

(ii) As seen above, we have fitted the full logistic regression with an interaction term between `age` and `country`. Since `France` is the reference level, and the interaction-term coefficient `age:countryindonesia` has the value -0.0718902 with a relatively low p-value, we can infer that age is a greater risk factor for the French population than for the Indonesian population.

d)

I will make the models before answering the questions, which can/will be deleted later (since these are multiple choice).

```
# Make data sets.
trainID = sample(x = 1:nrow(d.corona), size = 1500, replace = F) # E.g
train = d.corona[trainID,]
```

```r
test = d.corona[!trainID,]
nrow(d.corona)
```

```
#> [1] 2010
```

```r
# Ikke så greit å bruke indekser i d.corona tydeligvis. Kan se mer på dette senere dersom det virker
# som en fornuftig måte å løse oppgaven på...

lda.fit <- lda(deceased ~ ., data = train)
lda.fit.pred <- predict(lda.fit, newdata = test)$class
table(predicted = lda.fit.pred, true = test$deceased)
```

```
#> < table of extent 2 x 0 >
```

```r
# This did not work.
```

EVENTUALLY THERE WILL ONLY BE TRUE OR FALSE IN EACH OF THESE, BUT I HAVE WRITTEN MY THOUGHTS HERE FIRST.

(i) 5.2238806 % divided by the length of the whole dataset! seems to be the percentage of deaths in the dataset. This is not equal to the percentage given in the question (by accident). If they were equal I would say that this statement is TRUE. because, as I have understood it, the "null rate" for misclassification can be obtained in this case by always classifying the individual as healthy.

(ii) Following the argumentation in the above point, LDA is useless if the misclassification rate is higher than approx 5 %. Need to make a LDA in code to test? In this case, we are probably more interested in the probabilites than the classification, so LDA is not so relevant? spørre om dette på mandan!

(iii) Can we answer this from theory?

(iv) The same concern goes for this statement as for the ones above.

Problem 3

```r
# Read file.
id <- "1i1cQPeoLLC_FyAH0nnqCnnrSBpn05_h0" # Google file ID.
diab <- dget(sprintf("https://docs.google.com/uc?id=%s&export=download", id))
t = MASS::Pima.tr2
train = diab$ctrain
test = diab$ctest
```

a)

```r
logReg = glm(diabetes~., data = train, family = "binomial")
summary(logReg)
```

```
#>
#> Call:
#> glm(formula = diabetes ~ ., family = "binomial", data = train)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -2.8155  -0.6367  -0.3211   0.6147   2.2408
#>
#> Coefficients:
#>               Estimate Std. Error z value Pr(>|z|)
```

```
#> (Intercept) -10.583538   1.428276  -7.410 1.26e-13 ***
#> npreg         0.105109   0.062721   1.676 0.093775 .
#> glu           0.035586   0.005892   6.039 1.55e-09 ***
#> bp           -0.014654   0.013982  -1.048 0.294615
#> skin          0.020379   0.020575   0.990 0.321962
#> bmi           0.094683   0.031265   3.028 0.002458 **
#> ped           1.931666   0.529573   3.648 0.000265 ***
#> age           0.038291   0.020247   1.891 0.058594 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance: 381.91  on 299  degrees of freedom
#> Residual deviance: 253.84  on 292  degrees of freedom
#> AIC: 269.84
#>
#> Number of Fisher Scoring iterations: 5
```

(i) We have that

$$p_i = \frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\cdots+\beta_7 x_{i7}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\cdots+\beta_7 x_{i7}}} := \frac{e^{\eta_i(x)}}{1+e^{\eta_i(x)}} \implies \frac{p_i}{1-p_i} = \frac{\frac{e^{\eta_i(x)}}{1+e^{\eta_i(x)}}}{1-\frac{e^{\eta_i(x)}}{1+e^{\eta_i(x)}}} = e^{\eta_i(x)}$$

$$\implies \text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \eta_i(x) = \beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\cdots+\beta_7 x_{i7}$$

(ii) Can be seen below

```
glm.probs <- predict(logReg, newdata = test, type = "response")
glm.preds <- ifelse(glm.probs > 0.5, 1, 0)
conf.table.glm <- table(predicted = glm.preds, true = test$diabetes)
conf.table.glm
```

```
#>          true
#> predicted   0   1
#>         0 137  29
#>         1  18  48
```

The sensitivity is $48/(48+29) \approx 0.623$ and the specificity is $137/(137+18) \approx 0.884$.

b)

(i) (are we supposed to give estimates in this case?: "Explain what they are in the diabetes classification problem") π_k is the prior probability, given by $\pi_k = P(y = k)$. $\boldsymbol{\mu}_k$ is the mean vector of class k, when we have assumed that each $f_k(\mathbf{x})$ is normal. In this case, the class 1, i.e. presence of diabetes, has the mean vector $\boldsymbol{\mu}_1$ and the class 0, i.e. non-presence of diabetes, has the mean vector $\boldsymbol{\mu}_0$. $\boldsymbol{\Sigma}$ is the covariance matrix of each class, when assumed that the distribution of each class are normal. In LDA $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma} \quad \forall k$, where as in QDA each $\boldsymbol{\Sigma}_k$ are allowed to be different. $f_k(x)$ is the density function of X for an observation that comes from class k. These are assumed to be normal in LDA and QDA.

(ii) The fits are seen below

```
lda.diabetes <- lda(diabetes~., data = train)
qda.diabetes <- qda(diabetes~., data = train)
```

```
lda.diabetes.probs <- predict(lda.diabetes, newdata = test)$posterior[, 2] # Only need the prob om diab
lda.preds <- ifelse(lda.diabetes.probs > 0.5, 1, 0)
conf.table.lda.diabetes <- table(predicted = lda.preds, true = test$diabetes)
conf.table.lda.diabetes
```

```
#>          true
#> predicted   0   1
#>         0 138  30
#>         1  17  47
```

```
qda.diabetes.probs <- predict(qda.diabetes, newdata = test)$posterior[, 2]
qda.preds <- ifelse(qda.diabetes.probs > 0.5, 1, 0)
conf.table.qda.diabetes <- table(predicted = qda.preds, true = test$diabetes)
conf.table.qda.diabetes
```

```
#>          true
#> predicted   0   1
#>         0 131  32
#>         1  24  45
```

The sensitivity and specificity for LDA are thus 0.6103896 and 0.8903226, respectively. The sensitivity and specificity of QDA are 0.5844156 and 0.8451613, respectively.

The difference between the methods is that the covariances are equal across all classes (in this case: both classes) in LDA, which gives linear discriminant functions and a linear decision boundary. On the contrary, the covariances in each class when using QDA are allowed to be different, which gives a quadratic discriminant function in each class and a quadratic decision boundary.

c)

(i) In the KNN approach, a new observation is classified by using the k nearest points (in Euclidean norm) to the observation in question, to estimate the probability of the new point belonging to each of the different classes in the response. The new point is classified to the class with the highest estimated probability.

(ii) I would probably use cross validation to test the predictive power for different K (elaborate, saw it in the lecture on linear regression, but not quite sure how it is done in practice).

(iii) The KNN classification fit can be seen below

```
set.seed(123) # For reproducibility.
knn.diabetes <- knn(train = train, test = test, cl = train$diabetes, k=25, prob=T)
conf.table.knn <- table(predicted = knn.diabetes, true = test$diabetes)
conf.table.knn
```

```
#>          true
#> predicted   0   1
#>         0 144  36
#>         1  11  41
```

The sensitivity is $41/(41 + 36) \approx 0.532$ and the specificity is $144/(144 + 11) \approx 0.929$.

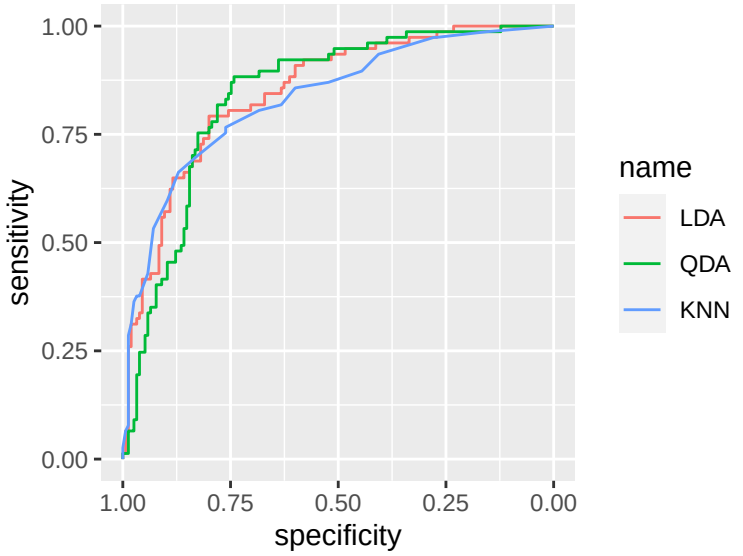
d) ROC curves

```
knn.probs <- ifelse(knn.diabetes == 0, 1 - attributes(knn.diabetes)$prob, attributes(knn.diabetes)$prob

ldaroc = roc(response = test$diabetes, predictor = lda.diabetes.probs, direction = "<")
qdaroc = roc(response = test$diabetes, predictor = qda.diabetes.probs, direction = "<")
```

```
knnroc = roc(response = test$diabetes, predictor = knn.probs, direction = "<")


ggroc(list(LDA  = ldaroc, QDA = qdaroc, KNN = knnroc))
```



The area under the ROC curves is 0.8490155 for LDA, 0.8414747 for QDA and 0.8334311 for KNN. Based on this, LDA preforms the best. KNN is not very interpretable? logistic regression is most intepretable, but that is not included???

Problem 4

a)

$$
\begin{aligned}
\hat{y}_{(-i)} &= \mathbf{x}_i^T \hat{\boldsymbol{\beta}}_{(-i)} \\
&= \mathbf{x}_i^T (X_{(-i)}^T X_{(-i)})^{-1} X_{(-i)}^T \mathbf{y}_{(-i)} \\
&= \mathbf{x}_i^T (X^T X - \mathbf{x}_i \mathbf{x}_i^T)^{-1} (X^T \mathbf{y} - \mathbf{x}_i y_i) \\
&= \mathbf{x}_i^T \left[(X^T X)^{-1} + \frac{(X^T X)^{-1} \mathbf{x}_i \mathbf{x}_i^T (X^T X)^{-1}}{1 - \mathbf{x}_i^T (X^T X)^{-1} \mathbf{x}_i} \right] (X^T \mathbf{y} - \mathbf{x}_i y_i), \quad \text{Sherman-Morrison} \\
&= \mathbf{x}_i^T \left[(X^T X)^{-1} + \frac{(X^T X)^{-1} \mathbf{x}_i \mathbf{x}_i^T (X^T X)^{-1}}{1 - h_i} \right] (X^T \mathbf{y} - \mathbf{x}_i y_i) \\
&= \mathbf{x}_i^T (X^T X)^{-1} X^T \mathbf{y} - \mathbf{x}_i^T (X^T X)^{-1} \mathbf{x}_i y_i + \mathbf{x}_i^T \frac{(X^T X)^{-1} \mathbf{x}_i \mathbf{x}_i^T (X^T X)^{-1}}{1 - h_i} X^T \mathbf{y} - \mathbf{x}_i^T \frac{(X^T X)^{-1} \mathbf{x}_i \mathbf{x}_i^T (X^T X)^{-1}}{1 - h_i} \mathbf{x}_i y_i \\
&= \mathbf{x}_i^T \hat{\boldsymbol{\beta}} - \mathbf{x}_i^T \hat{\beta}_i + \frac{1}{1 - h_i} h_i \mathbf{x}_i^T \hat{\boldsymbol{\beta}} - \frac{1}{1 - h_i} h_i \mathbf{x}_i^T \hat{\beta}_i \quad \text{Usikker på notasjon i 2. og 4. ledd} \\
&=
\end{aligned}
$$

b)

(i) FALSE
(ii) TRUE
(iii) ?log-transform of the response??? (mener de logistic regression her?) (eller mener de transform av predictors?)

(iv) FALSE (since the model is only fitted and trained on one combination of the two folds in the validation set approach, and not "the other way around" also, which would be the case in 2-fold cross validation.)

Problem 5

```
id <- "19auu8YlUJJJUsZY8JZfsCTWzDm6doE7C" # google file ID
d.bodyfat <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id),header=T)
```

a)

```
lm.fit5 <- lm(bodyfat ~ age + weight + bmi, data = d.bodyfat)
summary(lm.fit5)
```

```
#>
#> Call:
#> lm(formula = bodyfat ~ age + weight + bmi, data = d.bodyfat)
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -12.0307  -3.8921  -0.1454   3.8896  12.6272
#>
#> Coefficients:
#>               Estimate Std. Error t value Pr(>|t|)
#> (Intercept) -31.272668   2.807764 -11.138  < 2e-16 ***
#> age           0.133170   0.028282   4.709 4.23e-06 ***
#> weight        0.004075   0.058732   0.069    0.945
#> bmi           1.739406   0.216723   8.026 4.54e-14 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 5.34 on 239 degrees of freedom
#> Multiple R-squared:  0.5803, Adjusted R-squared:  0.575
#> F-statistic: 110.2 on 3 and 239 DF,  p-value: < 2.2e-16
```

The R^2 is ≈ 0.5803.

b)

DO THEY WANT US TO DO IT MANUALLY OR TO USE THE BOOTSTRAP-FUNCTIONS IN R?

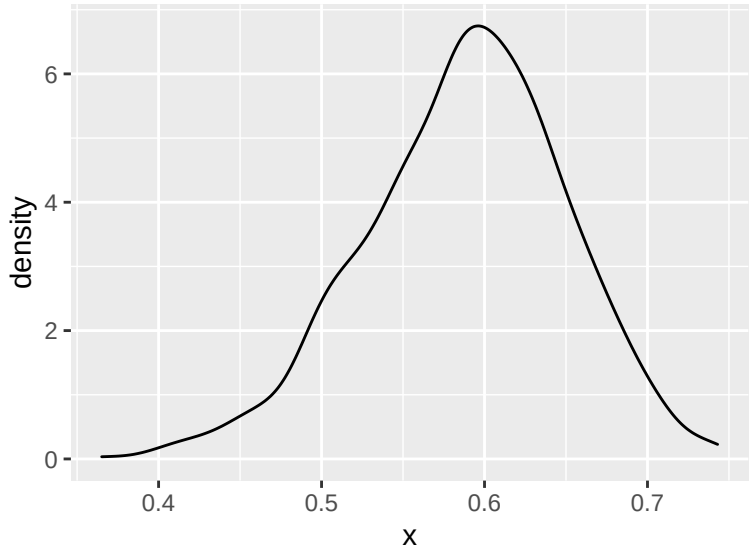
```
set.seed(4268)

# Not sure how to tell it to calculated the r.squared statistic.
# library(boot)
# boot(d.bodyfat, r.squared, R=1000)

# Doing it manually instead
B <- 1000
n <- 100 #e.g
r.squared <- rep(NA, B)
for (b in 1:B){
  dat <- d.bodyfat[sample(x = nrow(d.bodyfat), size = n, replace = T), ]
  fit <- lm(bodyfat ~ age + weight + bmi, data = dat)
  r.squared[b] <- summary(fit)$r.squared
```

```
}

ggplot(data=data.frame(x=r.squared),aes(x=x))+
  geom_density()
```




```
# Derive the standard error and the 95% confidence interval.
# Interpret.
```