Compulsory exercise 1: Group 39

TMA4268 Statistical Learning V2021

Alexander J Ohrt, Jim Totland

10 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 $\mathsf{E}(\mathbf{Y}) = \mathsf{E}(\mathbf{X}\boldsymbol{\beta} + \varepsilon) = \mathbf{X}\boldsymbol{\beta}$ and $\mathsf{Cov}(\mathbf{Y}) = \sigma I$, without assuming anything else than that each of the ε's are independent of each other.

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

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 $\widetilde{f}(\mathbf{x}_0)$ 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

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

FOR NOE GRISERI, UMULIG Å FORENKLE? SE PÅ DETTE SENERE.

```
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
```

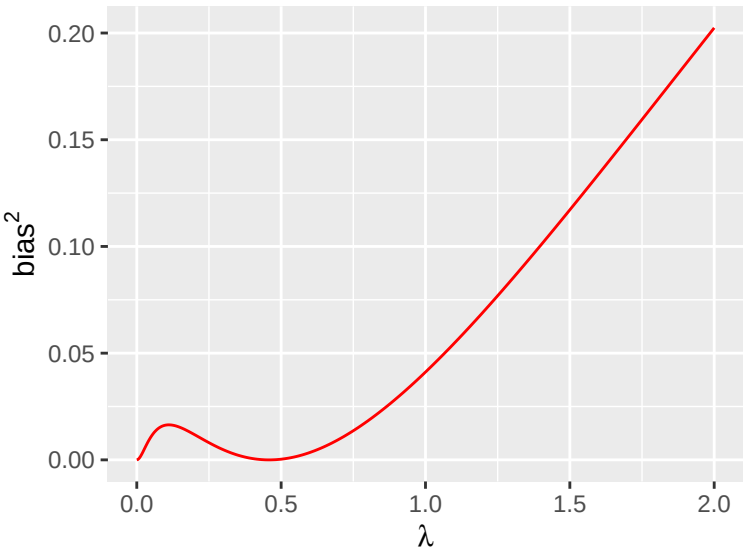
```
sigma=values$sigma
sigma
```

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

d)

```
library(ggplot2)
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))
```

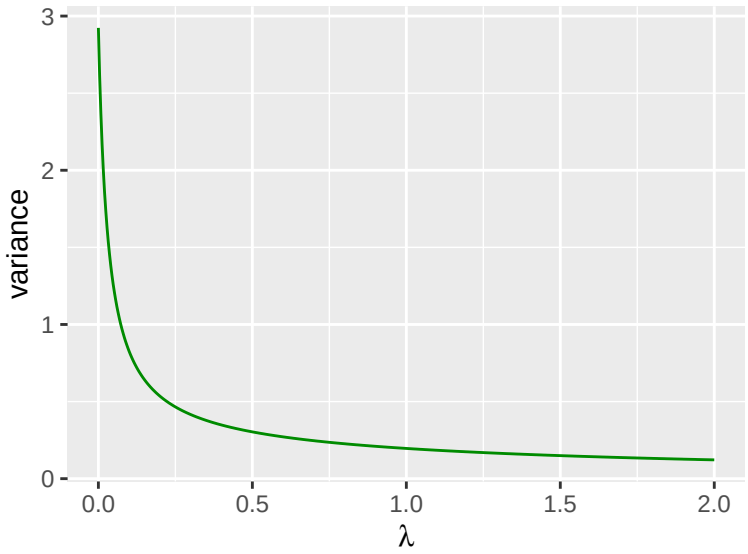


Comments: The graph shows that the bias of the ridge regression estimator increases as λ grows. OLS is unbiased, so, as expected, the bias is zero when $\lambda = 0$. Note that $\lambda \approx 0.5$ appears to be a sweet spot for this estimator when comparing all $\lambda > 0$, since the bias is low 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 %*% t(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")
```

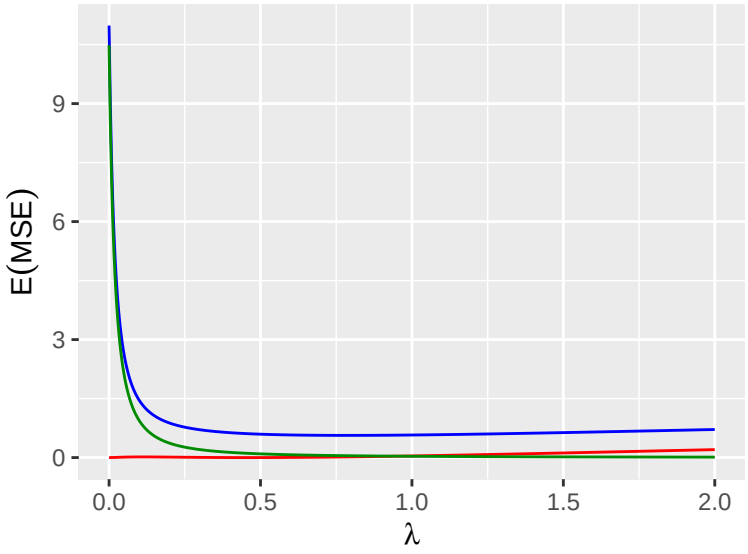


Comments: The variance begins at 2.9233983 and decreases with λ. Hence, it is apparent that the ridge regression estimator is advantageous compared to the OLS estimator, since the variance is decreasing with λ. In the end, when adding the changes in bias and variance, it is at this stage possible for the ridge regression estimator to have a lower MSE than the OLS estimator. However, we do not know if this is the case yet.

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

```
## [1] 1.354709
```

```
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 MSE compared to the OLS estimator, as gathered from the blue line in the plot. The optimal value of λ, which minimizes

MSE, is 1.3547094. 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)

Inspection of the data. Assuming that 0 = False and 1 = True (as usual), which means that a person has not deceased when `deceased` = 0 and vice versa.

```
knitr::kable(table(Deceased = d.corona$deceased)) # Prøvde å sette kolonne-navn, men fikk det ikke til.
```

Deceased	Freq
0	1905
1	105

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

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

```
knitr::kable(table(d.corona$deceased, d.corona$sex))
```

	female	male
0	1046	859
1	43	62

```
knitr::kable(table(d.corona$country, d.corona$deceased)) # Må hente ut øverste raden herfra, men klarte
```

	0	1
France	98	16
indonesia	64	5
japan	283	11
Korea	1460	73

b)

```
# Just in case they are not factors (this should perhaps be deleted later, since we could have checked
d.corona$sex = factor(d.corona$sex)
d.corona$country = factor(d.corona$country)
```

```r
lm.fit <- lm(deceased ~ ., data = d.corona) # perhaps a linear model is not the correct model to use?
summary(lm.fit)
```

```
##
## Call:
## lm(formula = deceased ~ ., data = d.corona)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.20383 -0.07105 -0.04495 -0.02110  1.03018
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.043862   0.025229   1.739 0.082263 .
## sexmale            0.030815   0.009902   3.112 0.001884 **
## age                0.001305   0.000218   5.984 2.57e-09 ***
## countryindonesia -0.053478   0.033584  -1.592 0.111455
## countryjapan     -0.097525   0.024269  -4.018 6.08e-05 ***
## countryKorea     -0.071966   0.021542  -3.341 0.000851 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2193 on 2004 degrees of freedom
## Multiple R-squared:  0.03144,    Adjusted R-squared:  0.02902
## F-statistic: 13.01 on 5 and 2004 DF,  p-value: 1.755e-12
```

(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(lm.fit, newdata = x0, type = "response")
```

```
##         1
## 0.1005556
```

(ii) The p-value for `sexmale` is relatively small, 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) The p-value of `countryjapan` is the smallest of the country-coefficients, which shows that it could potentially be important as a predictor. In the least, it does not exclude this possibility.

(iv) Quantify the odds

c)

(i)

(ii)

I am guessing that "the appropriate model" would be to use linear regression, as used earlier, since this is the only regression-model we have learned thus far. Not quite sure if they want us to leave out some of the predictors however, or if they just want us to keep all of them in the model.

d)

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

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

library(MASS)
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) 5.511811 % 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?

(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_FyAHOnnqCnnrSBpn05_hO" # 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) (assuming they want us to do it theoretically, and not in R, e.g. by plotting) We have that

$$p_i = \frac{e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\cdots+\beta_7 x_{i7}}}{1+e^{\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+\cdots+\beta_7 x_{i7}}} = \frac{e^{\eta_i(x)}}{1+e^{\eta_i(x)}} \implies \frac{p_i}{1-p_i} = \frac{\frac{e^{\eta_i(x)}}{1+e^{\eta_i(x9)}}}{1-\frac{e^{\eta_i(x)}}{1+e^{\eta_i(x)}}} = e^{\eta_i(x)}$$

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

(ii) Can be seen below

```
glm.probs <- predict(logReg, newdata = test, type = "response")
glm.preds <- ifelse(glm.probs > 0.5, 1, 0) #"Present", "Non-present") # Could also use these names.
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 0.6233766 and the specificity is 0.883871. (perhaps show the calculations more clearly in another way later.)

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

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

lda.diabetes.probs <- predict(lda.diabetes, newdata = test)$posterior
lda.preds <- ifelse(lda.diabetes.probs > 0.5, 1, 0)
#conf.table.lda.diabetes <- table(predicted = lda.preds, true = test$diabetes)
#conf.table.lda.diabetes

# Why are the probs half the length of the test data? I cannot seem to find the reason! :(

# Could also use $class below, but not sure how to set the cut-off probability in that case.
qda.diabetes.probs <- predict(qda.diabetes, newdata = test)$posterior
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
```

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. If the estimated probability of the point belonging to a class is larger than a pre-selected threshold (usually 0.5), the point is classified as belonging to this class. (perhaps it is just classified to the class that has the largest estimated probability?)

(ii) I would probably use cross validation to test the predictive power for different K (elaborate).

(iii) The KNN classification fit can be seen below

```r
library(class)
set.seed(123) # for reproducibility.
knn.diabetes <- knn(train = train, test = test, cl = train$diabetes, k=25, prob=T)
#table(predicted = knn.diabetes, true = train$diabetes) # Similar dimensionality problem as above!?!?
```

d) ROC curves

Problem 4

a)