Compulsory exercise 1

TMA4268 Statistical Learning V2019

Michail Spitieris, Andreas Strand and Mette Langaas

Deadline: February 22 at 16 on Bb

Contents

(Last changes: 05.02: clarify Q11 cv.se, 31.01:first version)

Introduction

Maximal score is 15 points. You need a score of 6/15 for the exercise to be approved. Your score will make up 15% points of your final grade.

Supervision

All supervision is in Smia.

- Thursday February 7, 14.15-15: introduction to R Markdown

General supervision:

- Thursdays 16.15-18, in addition
- Monday February 11 at 8.15-10
- Thursday February 14 at 14.15-16

Practical issues

- Maximal group size is 3 - join a group (self enroll) before handing in on Bb.
- Remember to write your names and group number on top of your submission.
- The exercise should be handed in as one R Markdown file and a pdf-compiled version of the R Markdown file (if you are not able to produce a pdf-file directly please make an html-file, open it in your browser and save as pdf - no, not landscape - but portrait please). We will read the pdf-file and use the Rmd file in case we need to check details in your submission.
- In the R-chunks please use both `echo=TRUE` and `eval=TRUE` to make it simpler for us to read and grade.
- Please do not include all the text from this file (that you are reading now) - we want your R code, plots and written solutions - use the template: https://www.math.ntnu.no/emner/TMA4268/2019v/CompEx1mal.Rmd
- Please not more than 10 pages in your pdf-file! (This is a request, not a requirement.)
- Please save us time and NOT submit word or zip - or only Rmd - that only results in extra work for us!

R packages

You need to install the following packages in R to run the code in this file.

```r
install.packages("knitr") #probably already installed
install.packages("rmarkdown") #probably already installed
install.packages("GLMsData") #data set for Problem 1
install.packages("ggplot2") #plotting with ggplot (all code provided)
install.packages("nortest") #test that data comes from normal distribution
install.packages("class")# for function knn
install.packages("colorspace") #for nice colors in provided plotting code
install.packages("caret") #for confusion matrices
install.packages("pROC") #for ROC curves
```

Problem 1: Multiple linear regression

[Maximal score: 4 points]

The lung capacity data `lungcap` (from the `GLMsData` R package) gives information on health and on smoking habits of a sample of 654 youths, aged 3 to 19, in the area of East Boston during middle to late 1970s.

We will focus on modelling forced expiratory volume `FEV`, a measure of lung capacity. For each person in the data set we have measurements of the following 5 variables:

- `FEV` the forced expiratory volume in litres, a measure of lung capacity; a numeric vector,
- `Age` the age of the subject in completed years; a numeric vector,
- `Ht` the height in inches; a numeric vector,
- `Gender` the gender of the subjects: a numeric vector with females coded as 0 and males as 1,
- `Smoke` the smoking status of the subject: a numeric vector with non-smokers coded as 0 and smokers as 1

First we transform the height from inches to cm. Then a multiple normal linear regression model is fitted to the data set with `log(FEV)` as response and the other variables as covariates. The following R code may be used.

```r
library(GLMsData)
data("lungcap")
lungcap$Htcm=lungcap$Ht*2.54
```

```r
modelA = lm(log(FEV) ~ Age + Htcm + Gender + Smoke, data=lungcap)
summary(modelA)
```

```
##
## Call:
## lm(formula = log(FEV) ~ Age + Htcm + Gender + Smoke, data = lungcap)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.63278 -0.08657  0.01146  0.09540  0.40701
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.943998   0.078639 -24.721  < 2e-16 ***
## Age          0.023387   0.003348   6.984  7.1e-12 ***
## Htcm         0.016849   0.000661  25.489  < 2e-16 ***
## GenderM      0.029319   0.011719   2.502   0.0126 *
## Smoke       -0.046067   0.020910  -2.203   0.0279 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1455 on 649 degrees of freedom
## Multiple R-squared:  0.8106, Adjusted R-squared:  0.8095
## F-statistic: 694.6 on 4 and 649 DF,  p-value: < 2.2e-16
```

Understanding model output

We call the model fitted above `modelA`.

Q1: Write down the equation for the fitted `modelA`.

Q2: Explain (with words and formulas) what the following in the `summary`-output means, use the `Age` and/or the `Smoke` covariate for numerical examples.

- `Estimate` - in particular interpretation of `Intercept`
- `Std.Error`
- `Residual standard error`
- `F-statistic`

Model fit

Q3: What is the proportion of variability explained by the fitted `modelA`? Comment.

Q4: Run the code below to produce diagnostic plots of "fitted values vs. standardized residuals" and "QQ-plot of standardized residuals" to assess the model fit. Comment on what you see.

```r
library(ggplot2)
# residuls vs fitted
ggplot(modelA, aes(.fitted, .stdresid)) + geom_point(pch = 21) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_smooth(se = FALSE, col = "red", size = 0.5, method = "loess") +
  labs(x = "Fitted values", y = "Standardized residuals",
       title = "Fitted values vs. standardized residuals",
       subtitle = deparse(modelA$call))
```

```r
# qq-plot of residuals
ggplot(modelA, aes(sample = .stdresid)) +
  stat_qq(pch = 19) +
  geom_abline(intercept = 0, slope = 1, linetype = "dotted") +
  labs(x = "Theoretical quantiles", y = "Standardized residuals",
       title = "Normal Q-Q", subtitle = deparse(modelA$call))

# normality test
library(nortest)
ad.test(rstudent(modelA))
```

Q5: Now fit a model, call this `modelB`, with `FEV` as response, and the same covariates as for `modelA`. Would you prefer to use `modelA` or `modelB` when the aim is to make inference about `FEV`? Explain what you base your conclusion on.

Hypothesis test and confidence interval

We use `modelA` and focus on addressing the association between `Age` and the response.

Q6: We would like to test if `Age` has an effect on `log(FEV)` in `modelA`, that is we want to test $H_0 : \beta_{\text{Age}} = 0$ against $H_1 : \beta_{\text{Age}} \neq 0$. What type of test can we perform? Perform the test and report the p-value. At which significance levels will you reject the null hypothesis?

Q7: Construct a 99% confidence interval for β_{Age} (write out the formula and calculate the interval numerically). Explain what this interval tells you. From this confidence interval, is it possible for you know anything about the value of the p-value for the test in Q6? Explain.

Prediction

Consider a 16 year old male. He is 170 cm tall and not smoking.

```r
new = data.frame(Age=16, Htcm=170, Gender="M", Smoke=0)
```

Q8: What is your best guess for his `log(FEV)`? Construct a 95% prediction interval for his forced expiratory volume `FEV`. Comment. Hint: first contruct values on the scale of the response `log(FEV)` and then transform the upper and lower limits of the prediction interval. Do you find this prediction interval useful? Comment.

Problem 2: Classification

[Maximal score: 7 points]

We will look at statistics from the four major tennis tournaments in 2013. For more information on the data set see https://archive.ics.uci.edu/ml/datasets/Tennis+Major+Tournament+Match+Statistics.

Each row in the data set contains information about one match. The variables that end with `.1` relate to player 1, while `.2` concerns player 2.

In tennis, you have two attempts at the serve. You lose the point if you fail both. The number of these double faults commited is given in the variable `DBF`, while the variable `ACE` is the number of times your opponent fails to return your serve. Similarily, unforced errors (mistakes that are supposedly not forced by good shots of your opponent) are called `UFE`. A skilled player will score many aces while committing few double faults and unforced errors.

Each match involves two players, and there are no draws in tennis. The `result` of a match is either that

- player 1 wins, coded as `1` (so, success for player 1) or that
- player 2 wins, coded as `0` (so, failure for player 1).

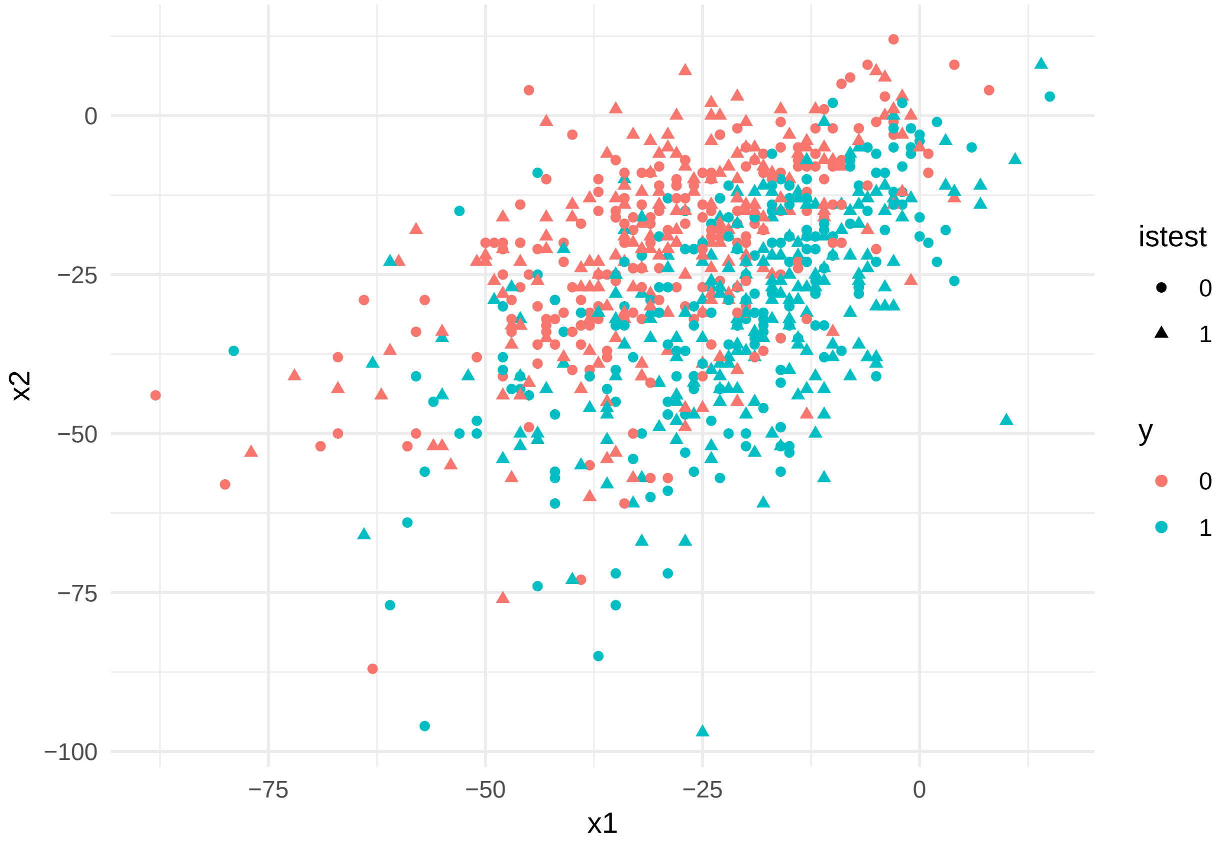
For our two players, player 1 and player 2, the quality of player k ($k = 1, 2$) can be summarized as $x_k = \text{ACE}.k - \text{UFE}.k - \text{DBF}.k$.

We will try to predict the outcome of a match (success or failure for player 1) just using the quality statistics x_1 from player 1 and x_2 from player 2.

The code below puts the result of each match and the quality score of each player in a data frame. The rows in the data set used for training are called `tr`, which means that test indices are `-tr`.

In the scatter plot the training observations are depictured as dots (circles) and the test observations as triangles. Matches where player 1 won is in turquoise and where player 2 won in red.

```r
library(class)# for function knn
library(caret)# for confusion matrices
library(ggplot2)# for ggplot
raw = read.csv("https://www.math.ntnu.no/emner/TMA4268/2019v/data/tennis.csv")
M = na.omit(data.frame(y=as.factor(raw$Result),
                       x1=raw$ACE.1-raw$UFE.1-raw$DBF.1,
                       x2=raw$ACE.2-raw$UFE.2-raw$DBF.2))
set.seed(4268) # for reproducibility
tr = sample.int(nrow(M),nrow(M)/2)
trte=rep(1,nrow(M))
trte[tr]=0
Mdf=data.frame(M,"istest"=as.factor(trte))
ggplot(data=Mdf,aes(x=x1,y=x2,colour=y,group=istest,shape=istest))+
  geom_point()+theme_minimal()
```



K-nearest neighbour classification

For a positive integer $K$ let $\mathcal{N}_r$ be the $K$ points in the training data nearest to $x = (x_1, x_2)$.

Q9: Write down the mathmatical formula for the K-nearest neighbours estimator $\hat{y}(x) \in \{0, 1\}$.

Q10: Compute the misclassification error for the training data and the test data using KNN classification for all $K$ in `ks=1:30`. See `?class::knn` for help on the function `knn` in the `class` R package. Save the error rates of the training and test sets as `train.e` and `test.e`, each vectors of length 30.

Cross-validation

The optimal $K$ can be chosen by cross-validation. Consider the code below where we divide the training data into 5 folds using `createFolds` from the R package `caret`. The entry (j, k) of the matrix `cv` is the CV error rate for test fold j where $K = k$ for the KNN-classifier.

```r
set.seed(0)
ks = 1:30 # Choose K from 1 to 30.
idx = createFolds(M[tr,1], k=5) # Divide the training data into 5 folds.
# "Sapply" is a more efficient for-loop.
# We loop over each fold and each value in "ks"
# and compute error rates for each combination.
# All the error rates are stored in the matrix "cv",
# where folds are rows and values of $K$ are columns.
cv = sapply(ks, function(k){
  sapply(seq_along(idx), function(j) {
    yhat = class::knn(train=M[tr[ -idx[[j]] ], -1],
               cl=M[tr[ -idx[[j]] ], 1],
               test=M[tr[ idx[[j]] ], -1], k = k)
    mean(M[tr[ idx[[j]] ], 1] != yhat)
  })
})
```

Q11: Compute the average `cv.e` and standard error `cv.se` of the average CV error over all 5 folds (that is, standard deviation over the 5 folds divided by sqrt of 5). Store the $K$ corresponding to the smallest CV error as `k.min`.

```r
cv.e = # fill in
cv.se = #fill in
k.min = # fill in
```

Q12: Plot the misclassification errors using the code below. Will the bias in $\hat{y}(x)$ increase with $K$? What about the variance?

```r
library(colorspace)
co = rainbow_hcl(3)
par(mar=c(4,4,1,1)+.1, mgp = c(3, 1, 0))
plot(ks, cv.e, type="o", pch = 16, ylim = c(0, 0.7), col = co[2],
     xlab = "Number of neighbors", ylab="Misclassification error")
arrows(ks, cv.e-cv.se, ks, cv.e+cv.se, angle=90, length=.03, code=3, col=co[2])
lines(ks, train.e, type="o", pch = 16, ylim = c(0.5, 0.7), col = co[3])
lines(ks, test.e, type="o", pch = 16, ylim = c(0.5, 0.7), col = co[1])
legend("topright", legend = c("Test", "5-fold CV", "Training"), lty = 1, col=co)
```

Q13: Run the code below. Instead of simply choosing the $K$ that results in the smallest CV error rate, we will try a different strategy shown in the first line of the code below. Explain the proposed strategy for choosing K.

```r
k = tail(which(cv.e < cv.e[k.min] + cv.se[k.min]), 1)
size = 100
xnew = apply(M[tr,-1], 2, function(X) seq(min(X), max(X), length.out=size))
grid = expand.grid(xnew[,1], xnew[,2])
grid.yhat = knn(M[tr,-1], M[tr,1], k=k, test=grid)
np = 300
par(mar=rep(2,4), mgp = c(1, 1, 0))
contour(xnew[,1], xnew[,2], z = matrix(grid.yhat, size), levels=.5,
        xlab=expression("x"[1]), ylab=expression("x"[2]), axes=FALSE,
        main = paste0(k,"-nearest neighbors"), cex=1.2, labels="")
points(grid, pch=".", cex=1, col=grid.yhat)
points(M[1:np,-1], col=factor(M[1:np,1]), pch = 1, lwd = 1.5)
legend("topleft", c("Player 1 wins", "Player 2 wins"),
       col=c("red", "black"), pch=1)
box()
```

ROC and AUC

Q14:

Now select the optimal choice of K that you found in Q13. Use the training set to produce the probability for player 1 winning the match for the matches in the test set (see code below). Then produce an ROC based on the test set. Explain how the ROC is produced. Also report the AUC and explain why random guessing would produce an AUC of 0.5.

```r
K=# your choice from Q13

# knn with prob=TRUE outputs the probability of the winning class
# therefore we have to do an extra step to get the probability of player 1 winning
KNNclass=class::knn(train=M[tr,-1], cl=M[tr,1], test=M[-tr,-1], k = K,prob=TRUE)
KNNprobwinning=attributes(KNNclass)$prob
KNNprob= ifelse(KNNclass == "0", 1-KNNprobwinning, KNNprobwinning)
# now KNNprob has probability that player 1 wins, for all matches in the test set

library(pROC)
# now you use predictor=KNNprob and response=M[-tr,1]
# in your call to the function roc in the pROC library
```

A competing classifier

Let us consider a different classifier, namely $\tilde{y}(x) = \text{argmax}_k(x_k)$, which is giving the win to the player with the highest quality score.

Q15: Add the decision boundary of $\tilde{y}(x)$ in the plot from Q13. Make confusion matrices and report the estimated misclassification error of $\hat{y}(x)$ from Q13 and $\tilde{y}(x)$ using the test data. Which classifier do you prefer? Hint: the R package `caret` can easily be used for producing a confusion matrix.

Problem 3: Bias-variance trade-off

[Maximal score: 4 points]

In this problem we will first mathematically derive analytical formulas for the bias and variance terms for two regression estimators, and then plot curves in R.

Let $\mathbf{x}$ be a $(p+1)\times 1$ vector of covariates (including a constant 1 as the first term). We look at a regression problem

$$Y = f(\mathbf{x}) + \varepsilon, \text{ where } \mathrm{E}(\varepsilon) = 0 \text{ and } \mathrm{Var}(\varepsilon) = \sigma^2.$$

Assume that we know that the true function is a linear combination of observed $(p+1)$ covariates $f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta}$. This means that the irreducible error is $\mathrm{Var}(\varepsilon) = \sigma^2$.

We will consider two different estimators for $\boldsymbol{\beta}$, and let the prediction of a new response at some covariate vector $\mathbf{x}_0$ be

$$\hat{f}(\mathbf{x}_0) = \mathbf{x}_0^T\hat{\boldsymbol{\beta}} \text{ or } \widetilde{f}(\mathbf{x}_0) = \mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}$$

for the two estimators $\hat{\boldsymbol{\beta}}$ and $\widetilde{\boldsymbol{\beta}}$ to be given below.

The estimators are based on a training set, where $\mathbf{X}$ is a $n\times(p-1)$ design matrix and $\mathbf{Y}$ is a $n\times 1$ vector of reponses, where we assume that the observed responses are independent of each other. That is, we assume that the expected vector is $\mathrm{E}(\mathbf{Y}) = \mathbf{X}\boldsymbol{\beta}$ and the variance-covariance matrix is $\mathrm{Cov}(\mathbf{Y}) = \sigma^2\mathbf{I}$.

Classical least squares estimator

The first estimator $\hat{\boldsymbol{\beta}}$ we will consider is the least squares estimator

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

Q16: Find the expected value vector and variance-covariance matrix for $\hat{\boldsymbol{\beta}}$.

Q17: Use this to find the expected value and variance for $\hat{f}(\mathbf{x}_0) = \mathbf{x}_0^T\hat{\boldsymbol{\beta}}$, that is $\mathrm{E}(\hat{f}(\mathbf{x}_0)$ and $\mathrm{Var}(\hat{f}(\mathbf{x}_0))$.

Q18: Use what you found in Q17 to write out $\mathrm{E}[(Y_0 - \hat{f}(\mathbf{x}_0))^2]$ as a sum of the squared bias, variance and irreducible error, that is

$$\mathrm{E}[(Y_0 - \hat{f}(\mathbf{x}_0))^2] = [\mathrm{E}(\hat{f}(\mathbf{x}_0) - f(\mathbf{x}_0)]^2 + \mathrm{Var}(\hat{f}(\mathbf{x}_0)) + \mathrm{Var}(\varepsilon)$$

Ridge regression estimator

A competing estimator is given as

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

for some choice of a numerical parameter λ (called a *regularization parameter*). Observe that the ridge regression estimator equals the least squares estimator for $\lambda = 0$.

Q19: Find the expected value vector and variance-covariance matrix for $\widetilde{\boldsymbol{\beta}}$.

Q20: Use this to find the expected value and variance for $\widetilde{f}(\mathbf{x}_0) = \mathbf{x}_0^T\widetilde{\boldsymbol{\beta}}$, that is $\mathrm{E}(\widetilde{f}(\mathbf{x}_0)$ and $\mathrm{Var}(\widetilde{f}(\mathbf{x}_0))$.

Q21: Use what you found in Q20 to write out $\mathrm{E}[(Y_0 - \widetilde{f}(\mathbf{x}_0))^2]$ as a sum of the squared bias, variance and irreducible error, that is

$$\mathrm{E}[(Y_0 - \widetilde{f}(\mathbf{x}_0))^2] = [\mathrm{E}(\widetilde{f}(\mathbf{x}_0) - f(\mathbf{x}_0)]^2 + \mathrm{Var}(\widetilde{f}(\mathbf{x}_0)) + \mathrm{Var}(\varepsilon)$$

Plotting the three components

We will now plot the three components (squared bias, variance and irreducible error) as a function of λ for one set of values for $\mathbf{X}$, $\mathbf{x}_0$, $\boldsymbol{\beta}$ and σ.

```
values=dget("https://www.math.ntnu.no/emner/TMA4268/2019v/data/BVtradeoffvalues.dd")
X=values$X
dim(X)
x0=values$x0
dim(x0)
beta=values$beta
dim(beta)
sigma=values$sigma
sigma
```

```
## [1] 100  81
## [1] 81  1
## [1] 81  1
## [1] 0.5
```

Observe that all the elements `X`, `x0` and `beta`, are given as matrices, to fit into your formulas. Hint: we perform matrix multiplication using `%*%`, transpose of a matrix `A` with `t(A)` and inverse with `solve(A)`.

Q22: Write a function `sqbias` with input parameters `lambda`, `X`, `x0`, `beta` and return the squared bias (a scalar). Plot the function (use colour red) for values of λ in `thislambda=seq(0,2,length=500)`, see code below. Does this curve look like you expected?

```
sqbias=function(lambda,X,x0,beta)
{
  p=dim(X)[2]
  value= #HERE YOU FILL IN
  return(value)
}
thislambda=seq(0,2,length=500)
sqbiaslambda=rep(NA,length(thislambda))
for (i in 1:length(thislambda)) sqbiaslambda[i]=sqbias(thislambda[i],X,x0,beta)
plot(thislambda,sqbiaslambda,col=2,type="l")
```

Q23: Write a function `variance` with input parameters `lambda`, `X`, `x0`, `sigma` which return the variance (a scalar). Plot the function (use colour blue) for values of λ in `thislambda=seq(0,2,length=500)`, see code below. Does this curve look like you expected?

```
variance=function(lambda,X,x0,sigma)
{
  p=dim(X)[2]
  inv=solve(t(X)%*%X+lambda*diag(p))
  value=#HERE YOU FILL IN
  return(value)
}
thislambda=seq(0,2,length=500)
variancelambda=rep(NA,length(thislambda))
for (i in 1:length(thislambda)) variancelambda[i]=variance(thislambda[i],X,x0,sigma)
plot(thislambda,variancelambda,col=4,type="l")
```

Q24: Then plot the squared bias (red), the variance (blue), the irreducible error (orange) and the sum of the three (black) as function of lambda. See code below (then `sqbiaslambda` and `variancelambda` need to be calculated in Q22 and Q23). What is the optimal value of λ for this problem?

```r
tot=sqbiaslambda+variancelambda+sigma^2
which.min(tot)
thislambda[which.min(tot)]
plot(thislambda,tot,col=1,type="l",ylim=c(0,max(tot)))
lines(thislambda, sqbiaslambda,col=2)
lines(thislambda, variancelambda,col=4)
lines(thislambda,rep(sigma^2,500),col="orange")
abline(v=thislambda[which.min(tot)],col=3)
```