TDDE07: Bayesian Learning

Computer Solutions: code

### EXAM 2020-06-04 ###

## Author: David Björelind, davbj395

# ---- 1a ---- #

# See PaperSol

# ---- 1b ---- #

point = 33

# Posterior with u(0.3, 0.7)

thetaGrid = seq(0,1, length = 100)

# Choosing not to make logpostfunc because of simple model with no numerical errors

postfunc = function(theta, a, b, x){

prior = dunif(theta, a, b)

like = dbinom(x, 50, theta)

return(prior\*like)

}

posta = rep(0, length(thetaGrid))

for (i in 1:length(thetaGrid)){

posta[i] = postfunc(thetaGrid[i], 0.3, 0.7, point)

}

posta = posta/sum(posta)/(thetaGrid[2]-thetaGrid[1])

#plot(x=thetaGrid, y=posta, type='l', col="green")

# Posterior with u(0, 1)

postb = rep(0, length(thetaGrid))

for (i in 1:length(thetaGrid)){

postb[i] = postfunc(thetaGrid[i], 0, 1, point)

}

postb = postb/sum(postb)/(thetaGrid[2]-thetaGrid[1])

plot(x=thetaGrid, y=posta, type='l', col="green", main="Green: u(0.3,0.7), Red: u(0,1)")

lines(x=thetaGrid, y=postb, type='l', col="red")

# It can be shown that the distribution with narrower prior gets cut of at theta=0.7

# and has higher peaks

# ---- 1c ---- #

prob1 = sum(postfunc(seq(0,0.5, length=50), 0, 1, 33))/

sum(postfunc(seq(0,1, length=100), 0, 1, 33)) # About 1,4%

prob2 = sum(postfunc(seq(0,0.5, length=50), 0.3, 0.7, 33))/

sum(postfunc(seq(0,1, length=100), 0.3, 0.7, 33)) # About 1,9%

# Probabilities seems reasonalbe when looking at the plots, since most of the density is to

# the right of 0.5

# Probability for the smaller prior is slightly larger, as expected when looking at the

# previous plot

# ---- 2a ---- #

y\_data = titanic$survived

x\_data = as.matrix(titanic[-1])

mu0 = rep(0,5)

tau = 50

nIter = 1000

# Made a correction in BayesProbReg due to nPara not found!

BayesProbReg <- function(y, X, mu\_0, tau, nIter){

# Prior

nPara = length(X[1,]) # I added this line!

priorCov <- tau^2\*diag(nPara)

priorPrec <- solve(priorCov)

# Compute posterior hyperparameters

n = length(y) # Number of observations

n1 = sum(y)

n0 = n - n1

nCovs = dim(X)[2] # Number of covariates

XX = t(X)%\*%X

# The actual sampling

betaSample = matrix(NA, nIter, nCovs)

u <- matrix(NA, n, 1)

beta <- solve(XX,crossprod(X,y)) # OLS estimate as initial value

for (i in 1:nIter){

xBeta <- X%\*%beta

# Draw u | beta

u[y == 0] <- rtnorm(n = n0, mean = xBeta[y==0], sd = 1, lower = -Inf, upper = 0)

u[y == 1] <- rtnorm(n = n1, mean = xBeta[y==1], sd = 1, lower = 0, upper = Inf)

# Draw beta | u

betaHat <- solve(XX,t(X)%\*%u)

postPrec <- XX + priorPrec

postCov <- solve(postPrec)

betaMean <- solve(postPrec,XX%\*%betaHat + priorPrec%\*%mu\_0)

beta <- t(rmvnorm(n = 1, mean = betaMean, sigma = postCov))

betaSample[i,] <- t(beta)

}

return(betaSample=betaSample)

}

# BayesProbReg <- function(y, X, mu\_0, tau, nIter)

samples = BayesProbReg(y\_data, x\_data, mu0, tau, nIter)

plot(samples[,1], type='l', main='"Feature" 1: Intercept')

# Not sure to plot 'Intercept', since it is not really a feature

plot(samples[,2], type='l', main='Feature 2: Adult')

plot(samples[,3], type='l', main='Feature 3: Man')

plot(samples[,4], type='l', main='Feature 4: Class1')

plot(samples[,5], type='l', main='Feature 5: Class2')

# All series look to be reasonable and no convergence errors

hist(samples[,1], 50, main='Distribution of feature 1: Intercept')

hist(samples[,2], 50, main='Distribution of feature 2: Adult')

hist(samples[,3], 50, main='Distribution of feature 3: Man')

hist(samples[,4], 50, main='Distribution of feature 4: Class1')

hist(samples[,5], 50, main='Distribution of feature 5: Class2')

# All histograms seems reasonable

# ---- 2b ---- #

# Since we have a linear loss function, we want to use posterior median as a point estimate!

PE1 = median(samples[,1]) # 0.77

PE2 = median(samples[,2]) # -0.57

PE3 = median(samples[,3]) # -1.42

PE4 = median(samples[,4]) # 1.02

PE5 = median(samples[,5]) # 0.40

# The results are reasonable when compared to the posterior distribution plots

# ---- 2c ---- #

beta25 = samples[,2] + samples[,5]

sum(beta25 > 0)/length(beta25) #13.2 %

# with 13.2% safety we can say that people that were Adults (f2) and Class2 (f5)

# contributed to them surviving (higher chance that y=1).

# With 86.8% safery we can say that these people were worse off on Titanic because

# they were Adults had tickets in Class2.

# ---- 2d ---- #

# See PaperSol

# ---- 3a ---- #

# y: number of medals

# x: log(money spent in M dollars)

y\_data=c(5, 3, 17, 8)

x\_data=log(c(20, 20, 50, 40))

# Making functions that calculates the log posterior values

logPostdens = function(beta, mu, sigma, x, y){

prior = dnorm(beta, mean = mu, sd = sigma, log=TRUE)

like = dpois(y, exp(x\*beta), log=TRUE)

return(prior+like)

}

logPostPoisN = function(beta, mu, sigma, x, y){

s = sum(logPostdens(beta, mu, sigma, x, y))

return(s)

}

startVal = 0

OptimResults<-optim(startVal,logPostPoisN,gr=NULL, 1, 1/10, x\_data, y\_data, method=c("L-BFGS-B"),

control=list(fnscale=-1),hessian=TRUE)

# Here is the optimal beta and sigma from the optimization

opti\_beta = OptimResults$par

opti\_sd = -solve(OptimResults$hessian)

betaGrid = seq(0.5,1.5,length = 1000)

normal\_post = dnorm(betaGrid, mean = opti\_beta, sd=sqrt(opti\_sd))

plot(x = betaGrid, y = normal\_post, type='l')

# Reasonable distribution for B, slightly lower than what the prior suggests

# ---- 3b ---- #

# decrease to 20M or stay at 40M

lossFunc = function(y, x){

s = 4 + exp(x)/50 - sqrt(y)

return(s)

}

# Idea: Make 10000 draws using the B that was approximated in a). For each value in xGrid, compute

# distribution for y. Put values into loss function and plot to see which value minimizes it

nIter = 10000

xGrid = log(seq(0.0001,60, length=nIter))

y\_values = rep(0, nIter)

for (i in 1:nIter){

y\_values[i] = rpois(1, exp(opti\_beta\*xGrid[i]))

}

losses = lossFunc(y\_values, xGrid)

plot(x=exp(xGrid), y = losses, type='l', xlab='Money spent', ylab='Expected loss (L(y,x))')

# Judning from the graph, the loss of 40M is LESS than the loss of 20M

# Therefore, the country should increase spending to 40M

# A plot of the generated y-values with "normal" values of the x-axle

plot(x=exp(xGrid), y = y\_values, type='l', xlab='Money spent', ylab='Expected amount of medals')

# Just a check: We can see that increased spending gives us higher prediction of medals,

# which is reasonable

# ---- 4a ---- #

# See PaperSol

# ---- 4b ---- #

# See PaperSol

# ---- 4c ---- # Naive Bayes Classifier

# See PaperSol for motivation behind calculations

naiveBayes =function(x\_length, x\_weight, gender){

theta = 1/4

length = dnorm(x\_length, 12, 2\*(1+1/20))

weight = dnorm(x\_weight, 280, 50\*(1+1/20))

s\_male =length\*weight\*theta

theta = 3/4

length = dnorm(x\_length, 14, 2\*(1+1/20))

weight = dnorm(x\_weight, 300, 50\*(1+1/20))

s\_female =length\*weight\*theta

# Normalizing so that propabilites are proper due to proportionality, see PaperSol

if (gender =='female'){

return(s\_female/(s\_male+s\_female))

} else {

return(s\_male/(s\_male+s\_female))

}

}

prob = naiveBayes(10, 250, "female") # 36%

# Reasonable since the values are closer that of a males normal values!