Lab1.Report

Arun Uppugunduri

12 januari 2020

## Assigment 1

The data file spambase.xlsx contains information about the frequency of various words, characters etc for a total of 2740 e-mails. Furthermore, these e-mails have been manually classified as spams (spam = 1) or regular e-mails (spam = 0). 1. Import the data into R and divide it into training and test sets (50%/50%) by using the following code: n=dim(data)[1] set.seed(12345) id=sample(1:n, floor(n\*0.5)) train=data[id,] test=data[-id,]

1. use logistic regresion functions (glm() and predcit() ) to classify the training and test data by the classsification principle Yhat = 1 if p(Y = 1|X) > 05 ptherwise Yhat = 0 and report the confusion matrices (use table()) and the misclassification rates for training and test data. Analyse the obtained results.
2. Use logistic regression to classify the test data by the classification principle as above but > 0.8

and report the confusion matrices (use table()) and the misclassification rates for training and test data. Compare the results. What effect did the new rule have? 4. Use standard classifier kknn() with K=30 from package kknn, report the the misclassification rates for the training and test data and compare the results with step 2

1. Repeat step 4 for K=1 and compare the results with step 4. What effect does the decrease of K lead to and why?

model <- glm(Spam~., family = binomial(link='logit'), data =train)  
summary(model)  
test.predict <- predict(model, newdata = test, type ="response")  
testdata = ifelse(test.predict >0.5, "1", "0")

Create the model annde make the prediciton. summary gives us the importance on the each prediction has

## [1] 0.1715328

## [1] 0.1605839

That the values are similar indicates that we have chosen a good model

library(kknn)  
kknnmodel = kknn(as.factor(Spam)~., train, test,k =30)  
kknnmodel = kknn(as.factor(Spam)~., train, train,k =30)  
kknnprediction = (fitted(kknnmodel))  
confusion = table(kknnprediction, train$Spam)  
print(confusion)  
kknnrate = 1 - sum(diag(confusion))/sum(confusion)  
print(kknnrate)  
  
kknnmodel2 = kknn(as.factor(Spam)~., train, test,k =1)  
kknnmodel2 = kknn(as.factor(Spam)~., train, train,k =1)  
kknnprediction2 = (fitted(kknnmodel2))  
confusion2 = table(kknnprediction2, train$Spam)  
kknnrate2 = 1 - sum(diag(confusion2))/sum(confusion2)  
print(confusion2)  
print(kknnrate2)

Having k = 30 makes the Missclass error on test very bad in comparision this indicates that the model is overfited. Lowering the k value natually gets the missclass better on the train data since when applying the prediciton it will loog at its 1 nearest neghbour which will be itself => missclass = 0. On the test data it becomes worse. The model is underfitted since it has to few values to base its prediction on.

### Assignment 2

Assignment 2. Inference about lifetime of machines . To be solved by TDDE01 students The data file machines.xlsx contains information about the lifetime of certain machines, and the company is interested to know more about the underlying process in order to determine the warranty time. The variable is following: . Length: shows lifetime of a machine

#1. Import the data to R.  
#2: Assume probability model p(x|theta) = theta\*e^(-theta\*x) for x = Length in which observations are independent and identically distributed. What is the distribution type of x. Write a function that computes the log-likelihood log p(x|theta) for a given theta and a given data vector x. Plot the curve showing the dependence of log-likelihood on theta where the entire data is used for fitting. What is the maximum likelihood value of theta according to plot?  
#3: Repeat step 2 but use only 6 first observations from the data, and put the two log-likelihood curves  
#(from step 2 and 3) in the same plot. What can you say about reliability of the maximum likelihood solution in  
#each case?  
  
#4: Assume now a Bayesian model with p(x|theta)=theta\*e^(-theta\*x) and a prior p(theta)=lambda\*e^(-lambda\*x), lambda=10  
#Write a function computing l(theta)=log(p(x|theta)\*p(theta)). What kind of measure is actually computed by this  
#function? Plot the curve showing the dependence of l(theta) on theta computed using the entire data and overlay it with a plot from step 2. Find an optimal theta and compare your result with the previous findings.   
  
checktheta = function(data){  
 check = seq(from = 0, to = 50, by = 0.1)  
 length = length(check)  
 place=1  
 vector = 1:length(check)  
 for (i in check){  
 vector[place] = log.like(i, data)  
 place=place+1  
 }  
 return (vector)  
}  
x = seq(from = 0, to = 10, by = 0.01)  
length = length(x)  
place=1  
vector = 1:length(check)  
for (i in x){  
 vector[place] = log.like(i, data)  
 place=place+1  
}  
  
  
plott1= plot(x,vector)  
  
max\_theta = function(data){  
 n=length(data[,1])  
 return(n/(sum(data)))  
}  
max\_theta(data)  
  
  
log.like(max\_theta(data),data)  
  
 like = function(theta){  
 x=log(prod((theta\*exp(-theta\*data))))  
 return(x)  
}  
  
theta = seq(from = 0, to = 10, by = 0.01)  
vec = sapply(theta,like)  
plot(theta, sapply(theta, like), xlab ="theta", ylab ="y")  
  
par(mfrow=c(1,1))  
 curve(dim(data)[1]\*log(x) - x\*sum(data), from = 0, to = 10)  
 curve(dim(data)[1]\*log(x-1) - x\*sum(data), from = 0, to = 10, add = TRUE, col = "red")  
   
curve(dim(machines\_1\_)[1]\*log(x) - x\*sum(machines\_1\_), from = 0, to = 10)  
  
newlike = function(theta, data){  
 n = length(data)  
 x=n\*log(theta) - theta\*sum(data)  
 return(x)  
}  
vec2 = sapply(theta,data,newlike)  
  
  
theta = seq(from = 0, to = 10, by = 0.01)  
newvec = newlike(theta, data$Length) #varör olika svar om bara data, är ju samma värden ?  
plot(theta,sapply(theta, like), xlab = "theta", ylab="y")  
lines(theta, newlike(theta, data$Length), col="red", add = TRUE)  
theta[which.max(newvec)] #max when theta = 1.13  
theta[which.max(vec)] #max when theta = 1.13  
optimize(f = like, c(0,9), maximum=TRUE)$maximum #1.126201  
optimize(f = newlike, c(0,9), maximum=TRUE, data =data$Length)$maximum #-42  
  
# cu  
# 2.3 Only data[1:6]   
plot(theta,newlike(theta,data$Length), xlab ="theta", ylab="y", ylim=c(-60,0), xlim=c(0,8))  
lines(theta,newlike(theta,data$Length[1:6]), col ="red" )  
  
  
plot(theta,newlike(theta,data$Length), xlab ="theta", ylab="y", ylim=c(-60,0), xlim=c(0,8))  
points(theta,newlike(theta,data$Length[1:6]), col ="red" )  
  
#uppg2.4  
  
 bay.like = function(theta){  
 lamda=10  
 x=log(prod((theta\*exp(-theta\*data)))\*lamda\*exp(-lamda\*theta))  
 return(x)  
}  
  
 bayesian = function(theta, data){  
 lamda = 10  
 n = length(data)  
 x=n\*log(theta) - theta\*sum(data) + log(lamda) -lamda\*theta  
 return (x)  
 }  
   
#med wn bra prior så smalnar sannolikheten av. Får ett bättre resultat  
bay = bayesian(theta,data$Length)  
plot(theta,bay, xlab ="theta", ylab="y")  
lines(theta, newlike(theta, data$Length), col="red")  
  
theta[which.max(bay)] #max when theta = 0.91  
optimize(f = bayesian, c(0,9), maximum=TRUE, data=data$Length)$maximum # last maximum to get the max  
  
bay.vec = sapply(theta, bay.like)  
plot(theta,bay.vec)  
theta[which.max(bay.vec)]  
optimize(f = bay.like, c(0,9), maximum=TRUE)$maximum # last maximum to get the max  
  
  
set.seed(12345)  
x = rexp(50, 1.26201)  
par(mfrow=c(1,2))  
breaks = seq(0, 10, 0.5)  
hist(x, main ="random", xlab ="Length", breaks = breaks)  
hist(x, main ="random", xlab ="Length")  
  
#hist(data$Length, main="Given", xlab = "Length")  
#hist(x)

Coclusion: the histogram shows us that the distrubutin ins fairly smilar beteen the actual and the predicted data. This concludes that the model is accurattly fitted to the distrubution model

### Assignment 4

Linear regression and regularization

# 2: Consider model Mi in which Moisture is normally distributed, and the expected Moisture is a polynomial function

# of Protein including the polynomial terms up to power of i (i.e. M1 is a linear model, M2 is a quadratic model and so on). Report a probabilistic model that describes Mi. Why is it appropriate to use MSE criterion when fitting this model to a training data?

# Conclusion: A probabilistic model describing M(i) is: M(i) = w0 + w1 \* X + w2 \* X2 + … + wi \* Xi (3) The MSE

# criterion is a suitable method since it punishes outliers to a larger extent. This creates a better fitted model

# compared to when you punish the absolute value. This reduces the risk of an overfitted model.

# 3: Divide the data into training and validation sets (50%/50%) and fit models Mi, i=1,…,6. For each model, record the training and the validation MSE and present a plot showing how training and validation MSE depend on i (write some R code to make this plot). Which model is best according to the plot? How do the MSE values change and why? Interpret this picture in terms of bias-variance tradeoff.