A4

Kerui Du

05/10/2021

Q1

a

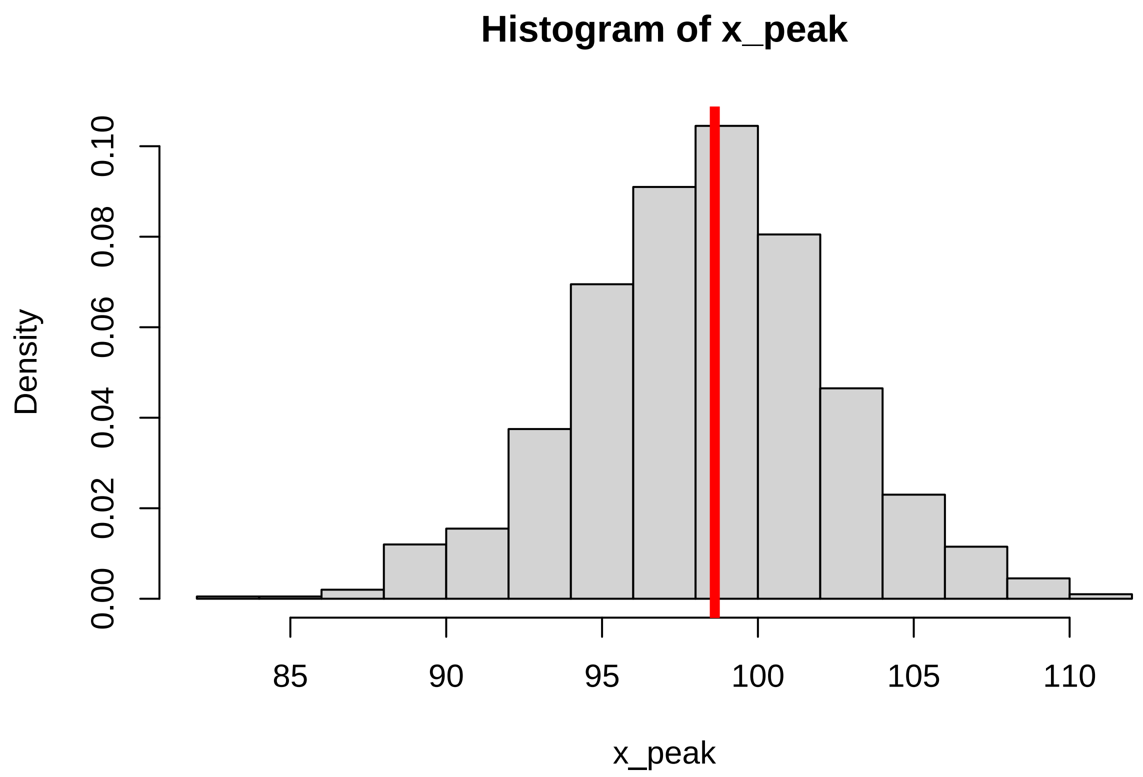
```
Caffeine2.df=data.frame(n=rep(300,5),levels=seq(0,200,50),
                        A_grades=c(109,155,175,158,103))
fit=glm(cbind(A_grades,n-A_grades)~levels+I(levels^2),family=binomial,data = Caffeine2.df)

ns=Caffeine2.df$n
xs=Caffeine2.df$levels
prob = with(Caffeine2.df,A_grades/n)

est = matrix(0,nr=1000,nc=2)
x_peak = -coef(fit)[2]/coef(fit)[3]/2

for (i in 1:nrow(est)) {
  ys = rbinom(length(ns),ns,prob)
  model=glm(cbind(ys,ns-ys)~xs+I(xs^2),family=binomial)
  est[i,]=coef(model)[2:3]
}

hist(-est[,1]/est[,2]/2,probability = T,xlab = 'x_peak',main='Histogram of x_peak')
abline(v=x_peak,lwd=5,col='red')
```



x_peak looks normality distributed and the peak of the histogram is around 98, which is pretty close to the x_peak(red line).

1b

```
c(2*x_peak-quantile(-est[,1]/est[,2]/2,.975),
  2*x_peak-quantile(-est[,1]/est[,2]/2,.025))
```

```
##    levels    levels
##  90.61776 107.43167
```

1c

The interval we obtained above is similar to the result that we gain from assignment 3, and it seems that the non-parametric CIs are slightly wider than the parametric CIs.

Q2

2a

```
train.df = read.table("data/train.txt")
names(train.df) = c("D", paste("V", 1:256, sep=""))
all(!is.na(train.df)) # examine if NA exist
```

```
## [1] TRUE
```

```
all(train.df$D==7|train.df$D==3) # D has value either 7 or 3
```

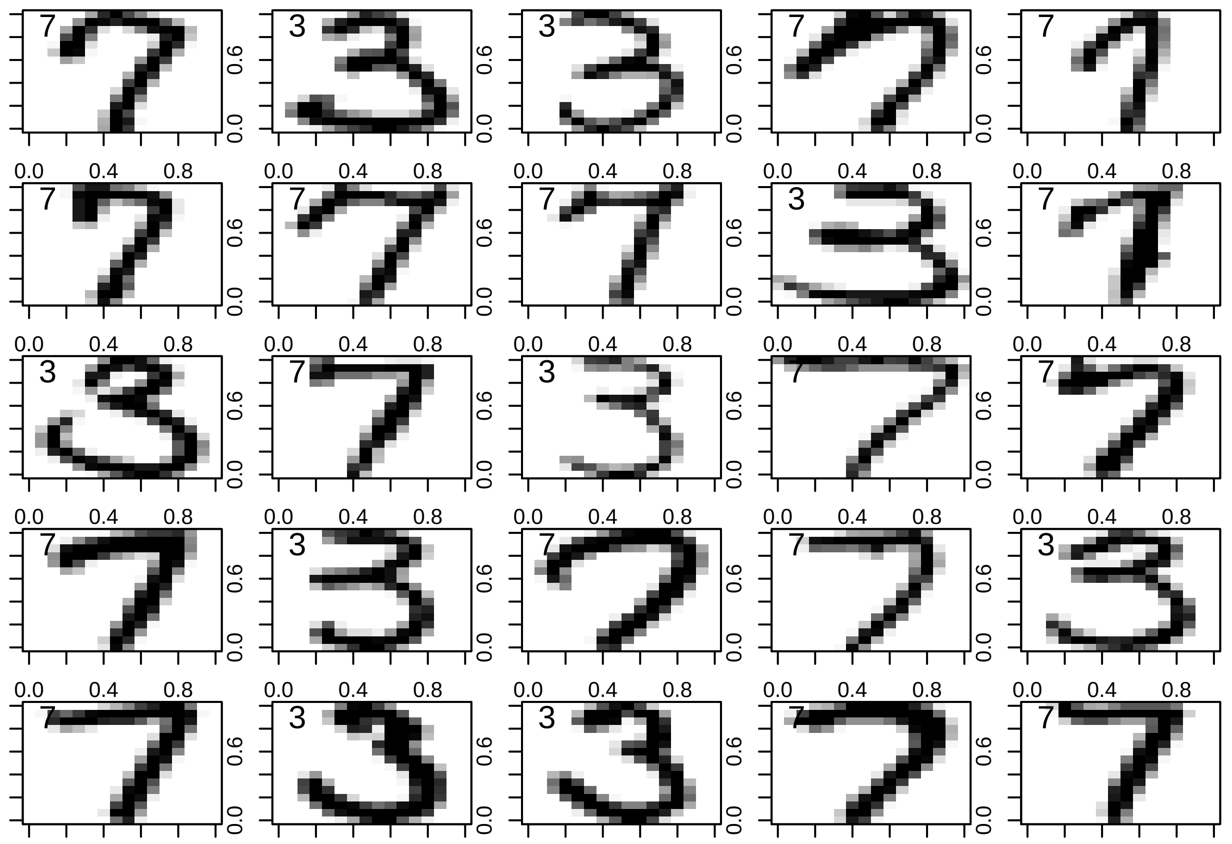
```
## [1] TRUE
```

```
all(train.df[-1]<=1 & train.df[-1]>=-1) # rest of columns between -1 to 1
```

```
## [1] TRUE
```

2b

```
mod=par(mfrow=c(5,5), mar = c(1,1,1,1))
for(k in 1:25){
  z = matrix(unlist(train.df[k,-1]), 16,16)
  zz = z
  for(j in 16:1)zz[,j]=z[,17-j]
  image(zz, col = gray((32:0)/32))
  box()
  text(0.1,0.9,train.df$D[k],cex=1.5)
}
```



```
par(mod)
```

By looking these image, we can easily detect that the there is a significant difference between digit 3 and digit 7 at bottom of each image, therefore the position v179/v180; v195/v196 should have value close to 1.

2c

```
corr = cor(train.df[,1:257])[,1]
varb=names(sort(abs(corr),decreasing = T))[2:21];varb
```

```
##  [1] "V185" "V170" "V105" "V220" "V235" "V201" "V229" "V120" "V219" "V230"
## [11] "V104" "V189" "V205" "V169" "V234" "V121" "V204" "V186" "V173" "V221"
```

2d

```
Y = train.df$Y = ifelse(train.df$D==7,1,0)
train.df1 = data.frame(train.df[varb],Y)
Full.mod=glm(Y~.,family = binomial,data = train.df1)
fitted_logit = predict(Full.mod,train.df1)
head(fitted_logit)
```

```
##          1          2          3          4          5          6
##   9.250851 -14.077921 -10.735400   2.836090   5.742092   7.653596
```

2e

```r
tb = table(round(train.df1$Y), round(fitted(Full.mod)));tb
```

```
##
##       0   1
##   0 651   7
##   1   7 638
```

```r
sum(tb[tb!=diag(tb)])/sum(tb)
```

```
## [1] 0.01074444
```

2f

```r
null.model = glm(Y~1, data=train.df, family=binomial)
#step(Full.mod, direction = "backward")
sub.mod = step(null.model, formula(Full.mod),direction="both", trace=0)
tb = table(round(train.df1$Y), round(fitted(sub.mod)));tb
```

```
##
##       0   1
##   0 650   8
##   1   9 636
```

```r
sum(tb[tb!=diag(tb)])/sum(tb)
```

```
## [1] 0.01304682
```

The value of this model is pretty much same to the value from full.mod above.

2g

```r
train.df2 = data.frame(train.df1[, names(sub.mod$coefficients)[-1]],Y)

set.seed(330)
PE.fun <- function() {
  sub <- sample(nrow(train.df2), floor(nrow(train.df2) * 0.9)) # 9 portions
  training <- train.df2[sub, ] # 90% for training
  testing <- train.df2[-sub, ] # 10% for testing
  glm.fit = glm(Y~.,data = training,family = binomial)
  tb=table(round(testing$Y),round(predict(glm.fit,testing[,-11],type = 'resp')))
  (tb[1,2]+tb[2,1])/sum(tb) # prediction error
}
PE.fun()
```

```
## [1] 0.01526718
```

The prediction error in cross validation depends on the observations that have been assigned in each training and testing dataset, then slightly different value from above, but it's close enough.

2h

```r
test.df=read.table('data/test.txt')
Y=test.df$Y=ifelse(test.df$V1==7,1,0)

test1.df=data.frame(test.df[names(coef(sub.mod)[-1])],Y)
pr_out=table(test1.df$Y,round(predict(sub.mod,test1.df[,-11],type = 'resp')));pr_out
```

```
##
##       0   1
##   0 162   4
##   1  17 130
```

```r
sum(pr_out[pr_out!=diag(pr_out)])/sum(pr_out)
```

```
## [1] 0.06709265
```

An estimate of the "out-of-sample" prediction error for the models from previous question is around 0.067, not much difference by 'in-sample' prediction error, which indicates both models are good for prediction.