Assignment 12 STAT 581

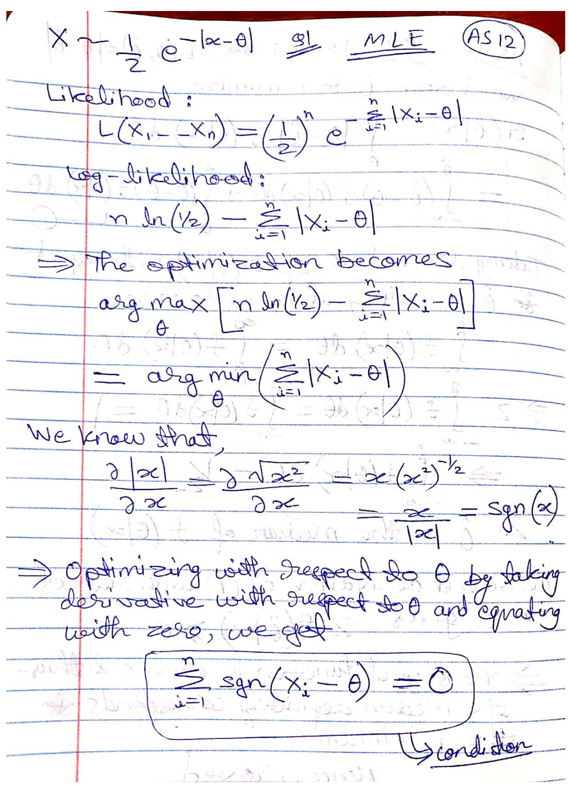
Tianzhi Cao tc324
Abhishek Bhatt ab2083
Siva Harshini Dev Bonam sdb202

November 30, 2020

Problem

Let x 1/2 exp (-—x-q—), show the maximum likelihood estimator of theta is the median.
Then following the proof of theorem 12.8 in the book, argue why the median of the posterior is the Bayes estimator for L1 loss

Answer



$X \sim \frac{1}{2} e^{-|x-\theta|}$ Q1 MLE AS12

Likelihood:

$$L(X_1 \dots X_n) = \left(\frac{1}{2}\right)^n e^{-\sum_{i=1}^{n} |X_i-\theta|}$$

Log-likelihood:

$$n \ln(1/2) - \sum_{i=1}^{n} |X_i - \theta|$$

⇒ The optimization becomes

$$\arg\max_{\theta} \left[n \ln(1/2) - \sum_{i=1}^{n} |X_i - \theta| \right]$$

$$= \arg\min_{\theta} \left(\sum_{i=1}^{n} |X_i - \theta| \right)$$

We know that,

$$\frac{\partial |x|}{\partial x} = \frac{\partial \sqrt{x^2}}{\partial x} = x(x^2)^{-1/2} = \frac{x}{|x|} = sgn(x)$$

⇒ Optimizing with respect to θ by taking derivative with respect to θ and equating with zero, we get

$$\boxed{\sum_{i=1}^{n} sgn(X_i - \theta) = 0}$$

↳ condition

Scanned with CamScanner

Figure 1: Question 1 and 2 Answer(1)

Case 1 : n is odd

If we choose $\hat{\theta} = \text{median}(x_1 -- x_n)$, then we get $\frac{n-1}{2}$ cases where $x_i < \theta$ & other $\frac{n-1}{2}$ cases where $x_i > \theta$.

$\Rightarrow$ $\hat{\theta}$ satisfies the condition for MLE estimator of θ.

Case 2 : n is even

Now we cannot choose one x_i to satisfy the condition for MLE estimator

We can still perform the minimization by assuming that x_1 —— x_n are ordered ($x_1 \le x_2 \le -- x_n$) and then choosing either $x_{n/2}$ or $x_{\frac{(n+1)}{2}}$.

$\Rightarrow$ Median is the MLE estimator for the given distribution.

Figure 2: Question 1 and 2 Answer(2)

AS 12 Q2 For L1 loss, $L(\theta, \hat{\theta}) = |\theta - \hat{\theta}|$

We choose $\hat{\theta}$ to minimize

$$r(\hat{\theta}|x) = \int_{\theta} |\theta - \hat{\theta}| \, f(\theta|x) \, d\theta$$

$$= \int_{-\infty}^{\hat{\theta}} (\hat{\theta} - \theta) f(\theta|x) \, d\theta + \int_{\hat{\theta}}^{\infty} (\theta - \hat{\theta}) f(\theta|x) \, d\theta \quad (1)$$

Taking derivative of $r(\hat{\theta}|x)$ with respect to $\hat{\theta}$ & setting it equal to zero:

$$\int_{-\infty}^{\hat{\theta}} f(\theta|x) \, d\theta = \int_{\hat{\theta}}^{\infty} f(\theta|x) \, d\theta$$

$$\Rightarrow 2 \int_{-\infty}^{\hat{\theta}} f(\theta|x) \, d\theta = \int_{-\infty}^{\infty} f(\theta|x) \, d\theta = 1$$

$$\Rightarrow \int_{-\infty}^{\hat{\theta}} f(\theta|x) \, d\theta = 1/2$$

& $\hat{\theta}$ is the median of $f(\theta|x)$.

The second derivative of (1) with respect to $\hat{\theta}$ gives $2f(\hat{\theta}|x) \geq 0$.

$\Rightarrow$ The original function is convex & thus the median definitely corresponds to the minimum.

Hence, proved.

Scanned with CamScanner

Figure 3: Question 1 and 2 Answer(3)

Problem

Compare the risk of the James stein estimator vs the MLE for k =1 to 100 for random theta i, plot it, use 1000 samples at each k to estimate risk. k is dimension!
Remember Stein estimator is based on $X_i\ N(\theta_i, 1), i = 1, ..., k$
Why is this toy problem a good model for many problems?

Answer

I using below code to compare MLE and JS in k=1 to 100 and draw graph:

```
JS<-function(){
  mleerr=NULL
  jserr=NULL
  for (k in 1:100){
    X<-matrix(nrow=1000,ncol=k)
    theta<-matrix(nrow=1,ncol=k)
    for (j in 1:k){
      theta[1,j]=runif(1,-5,5)
      newdata=rnorm(1000,theta[1,j],1)
      X[,j]=newdata
    }
    MLEerror=0
    steinerror=0
    for (line in (1:1000)){
      MLEerror=MLEerror+sum((theta-mean(X[line,]))^2)
      steinerror=steinerror+sum((theta-max(1-(k-2)/sum(X[line,]^2),0)*X[line,])^2)
    }
    mleerr=c(mleerr,MLEerror/1000)
    jserr=c(jserr,steinerror/1000)
  }
  list(mle=mleerr,js=jserr)
}
res<-JS()
print(res)
plot(1:100,res$mle, xlab = "k", ylab = "Error", type = "l")
lines(1:100, res$js, col = "red")
legend("topleft", legend = c("JSE", "MLE"), col = c("black", "red"),lwd = 2)
}
```

The is result I got for k=1 to 100:

```
> print (res)
$mle
  [1]   1.053322   9.433473  24.910562   7.567643  33.413775  40.331285  51.380925
  [8]  55.008244  48.705798  75.827131 115.167978 121.431616  96.294481 109.500785
 [15] 129.955104 172.283746 144.880120  81.915106 107.174265 174.978515 216.097701
 [22] 171.683329 160.784294 128.841811 215.537273 138.378467 271.977727 264.284253
 [29] 217.126816 212.195934 242.643251 281.213628 333.690420 242.922347 347.057364
 [36] 320.757822 308.838454 322.949937 322.920887 243.446965 325.719786 386.619272
 [43] 351.695142 437.072686 420.409805 428.602817 305.597800 306.842950 341.013882
 [50] 339.623322 414.555782 429.180762 380.681581 412.697886 484.913626 423.551939
 [57] 429.138804 476.358559 579.415108 547.068823 563.968909 505.286937 580.892673
 [64] 465.953259 563.828745 547.113225 539.204142 543.918017 561.741310 618.799072
 [71] 694.803093 575.683885 553.794136 699.167182 580.068661 728.841086 663.014007
 [78] 747.083820 694.483625 640.979519 524.150912 755.462681 646.501179 609.123009
 [85] 644.016939 657.968620 698.839336 622.854693 716.671510 767.990227 728.081372
 [92] 865.802619 771.888986 750.576596 808.039609 718.276694 841.524986 799.756903
 [99] 762.101432 777.285104

$js
  [1]  1.003164  2.032783  2.857481  3.387816  4.663612  5.464824  6.608993
  [8]  7.467392  8.363152  9.390577 10.182557 11.235519 11.807869 13.092188
 [15] 13.921688 14.796270 15.771914 15.717022 17.143550 18.460599 19.161752
 [22] 20.587747 20.684519 20.710084 22.962451 23.189635 24.933372 25.938062
 [29] 25.431258 26.587182 28.038909 29.308440 30.327434 30.169121 32.485414
 [36] 32.646230 33.516761 34.697508 35.284359 35.497744 36.471123 38.494100
 [43] 38.402638 40.325483 40.928707 41.620188 41.531869 41.803831 43.506877
 [50] 44.060194 45.388995 46.837400 47.492915 47.871060 50.136866 50.298529
 [57] 50.668114 52.973375 53.738417 54.540623 55.647270 55.839722 57.197018
 [64] 56.682769 58.751260 59.389361 60.155306 61.032707 61.877628 63.279134
 [71] 64.956998 65.030406 65.128486 67.124095 67.293292 69.102256 70.172158
 [78] 70.731555 71.525646 71.079270 71.162924 74.828329 74.434269 74.227971
 [85] 75.064204 76.455495 77.362372 76.650779 80.073335 81.195425 81.556329
 [92] 83.812167 84.089516 84.227991 84.955145 85.346259 88.143873 87.572748
 [99] 88.622435 88.789373
```

Figure 4: Result of Program

On Graph:

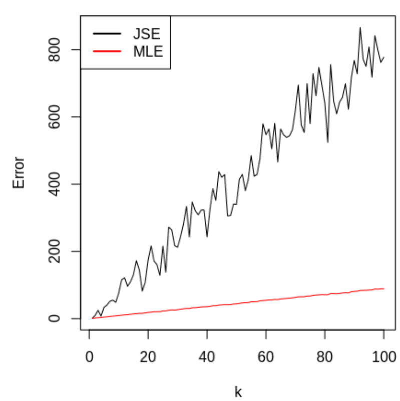


Figure 5: JS VS MLE