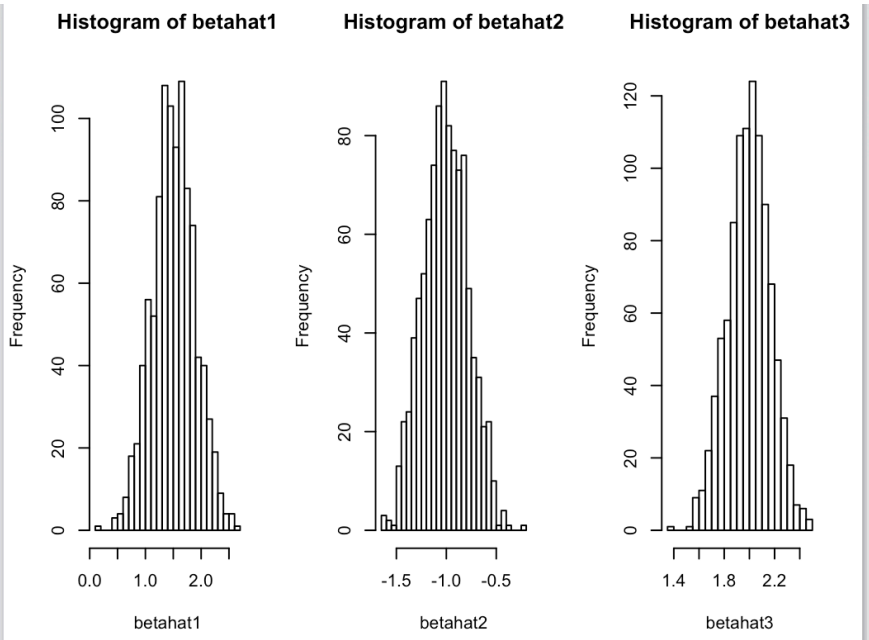
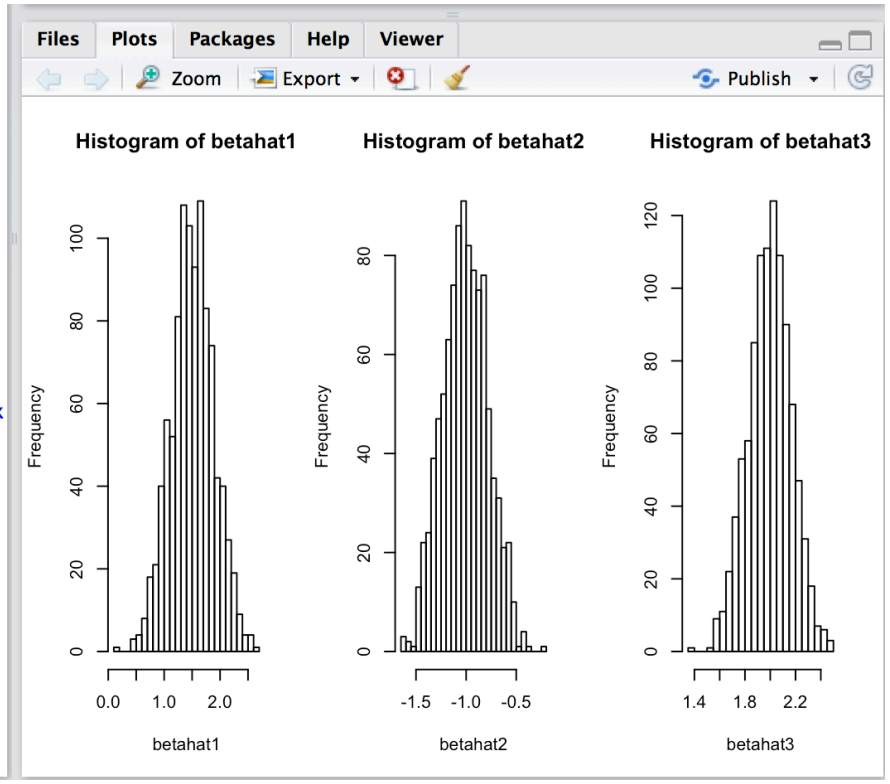
```
> #1(a)
> epsilon=matrix(rnorm(10),10, 1)
> k=c(1,1,1,1,1,1,1,1,1,1,2,-1,3,3,2,1,0,0,-1,0,-2,-2,-2,3,3,3,0,0,0,1)
> x=matrix(k,10,3)
> beta=matrix(c(1.5,-1,2),3,1)
> y=x%*%beta+epsilon
> y
            [,1]
 [1,] -4.8883372
 [2,] -0.7598943
 [3,] -7.7125448
 [4,]  5.1472573
 [5,]  6.1365537
 [6,]  7.1424400
 [7,]  1.4410498
 [8,]  1.1606626
 [9,]  2.8988171
[10,]  5.1614267
> betahat=solve((t(x))%*%x)%*%t(x)%*%y
> betahat
          [,1]
[1,]  1.899628
[2,] -1.403146
[3,]  2.339867
>
```

```
> #1(b)
> variance=solve((t(x))%*%x)
> variance
             [,1]         [,2]         [,3]
[1,]  0.139180672 -0.042016807 -0.003413866
[2,] -0.042016807  0.050420168 -0.008403361
[3,] -0.003413866 -0.008403361  0.027442227
> #The true variances are 0.139180672, 0.050420168, 0.027442227, the diagonal element
s of the matrix. Since we already know the sigma^2 from the distribution, we use the
diagonal elements of the matrix.
> rnorm(10)
 [1]  1.04709062 -0.04582358 -1.63658905  0.30050136  0.48873296 -0.37607279
 [7] -1.30709720 -0.49069162 -0.65719153  0.13075534
> #1(c)
> residuals=y-x%*%(betahat)
> n=10
> p=3
> sigma_squared_hat=(t(residuals)%*%residuals)/(n-p)
> sigma_squared_hat
          [,1]
[1,] 0.4988337
>
```

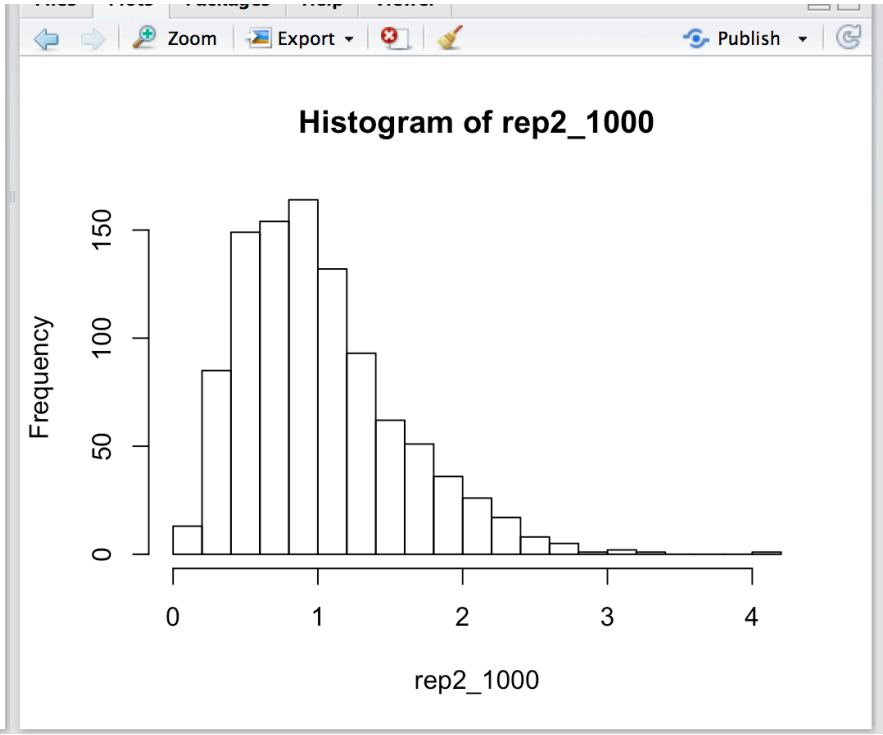
```
> #1(d)
> rep1_1000=replicate(1000,{epsilon=matrix(rnorm(10),10, 1)
+ k=c(1,1,1,1,1,1,1,1,1,1,2,-1,3,3,2,1,0,0,-1,0,-2,-2,-2,3,3,3,0,0,0,1)
+ x=matrix(k,10,3)
+ beta=matrix(c(1.5,-1,2),3,1)
+ y=x%*%beta+epsilon
+ y
+ betahat=solve((t(x))%*%x)%*%t(x)%*%y})
> betahat1=rep1_1000[1,,]
> betahat2=rep1_1000[2,,]
> betahat3=rep1_1000[3,,]
> par(mfrow=c(1,3))
> hist(betahat1, 20)
> hist(betahat2,20)
> hist(betahat3,20)
> rep_variance=replicate(1000, {variance=solve((t(x))%*%x)})
> rep_variance
, , 1

             [,1]         [,2]         [,3]
[1,]  0.139180672 -0.042016807 -0.003413866
[2,] -0.042016807  0.050420168 -0.008403361
[3,] -0.003413866 -0.008403361  0.027442227
```



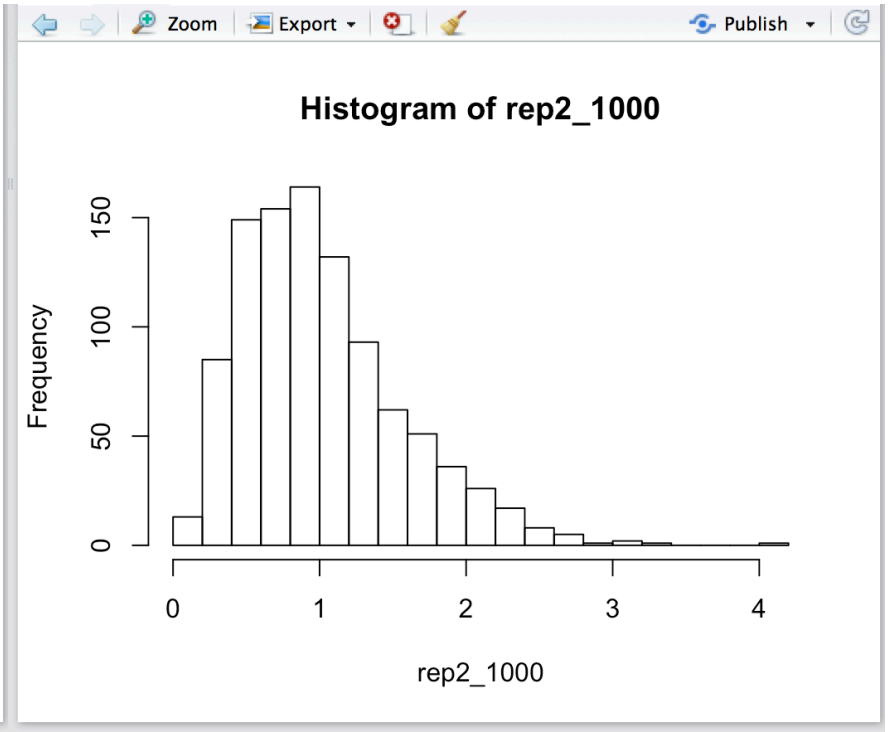
```
             [,1]        [,2]        [,3]
[1,]  0.139180672 -0.042016807 -0.003413866
[2,] -0.042016807  0.050420168 -0.008403361
[3,] -0.003413866 -0.008403361  0.027442227

, , 111

             [,1]        [,2]        [,3]
[1,]  0.139180672 -0.042016807 -0.003413866
[2,] -0.042016807  0.050420168 -0.008403361
[3,] -0.003413866 -0.008403361  0.027442227

 [ reached getOption("max.print") -- omitted 889 matrix slice(s) ]
> #Yes it matches with question 2, since x matrix doesn't change, so solve((t(x))%*%x
) doesn't change.
>

>

>

>

>
```



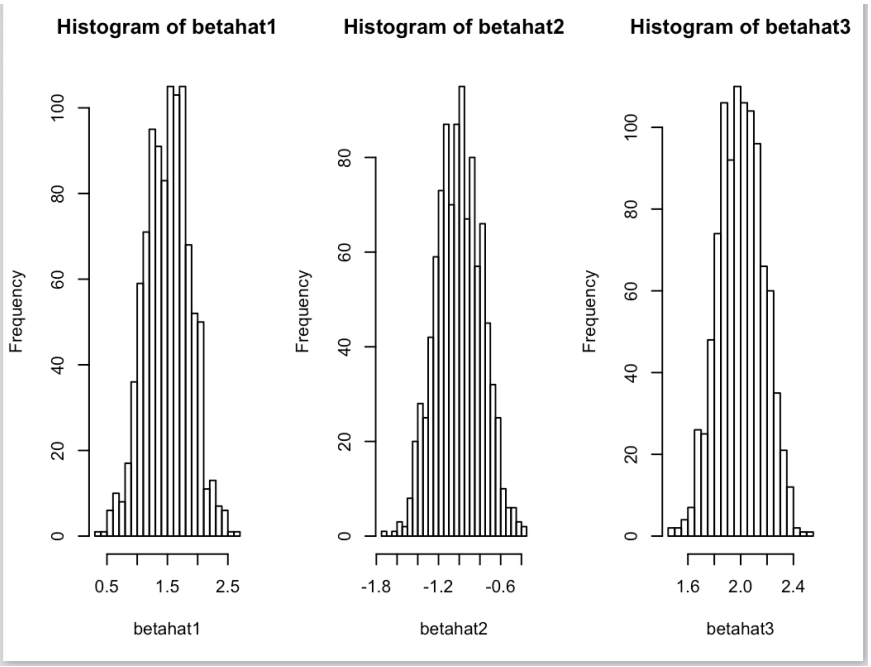
```
>

> #1(e)
> rep2_1000=replicate(1000,{epsilon=matrix(rnorm(10),10, 1)
+ k=c(1,1,1,1,1,1,1,1,1,2,-1,3,3,2,1,0,0,-1,0,-2,-2,-2,3,3,3,0,0,0,1)
+ x=matrix(k,10,3)
+ beta=matrix(c(1.5,-1,2),3,1)
+ y=x%*%beta+epsilon
+ y
+ betahat=solve((t(x))%*%x)%*%t(x)%*%y
+ residuals=y-x%*%(betahat)
+ n=10
+ p=3
+ sigma_squared_hat=(t(residuals)%*%residuals)/(n-p)
+ sigma_squared_hat})
> rep2_1000
   [1] 0.37294989 0.88138790 1.32005828 1.18870535 1.17267187 0.76961407
   [7] 1.17777118 0.82683511 0.86781298 0.87639437 1.25883376 1.48273549
  [13] 0.14501848 1.06062674 0.84277436 1.07790509 1.22375059 0.99688822
  [19] 0.41631691 0.68140069 0.92476477 1.27727300 0.43688055 2.39736923
  [25] 0.84514177 0.36566637 0.78852181 0.96034137 1.51547418 0.75225346
  [31] 1.08129276 1.21170686 1.99974715 0.99621229 0.69914283 1.02357067
  [37] 0.76670136 1.30782661 1.42152929 0.79560557 0.85040739 0.36912815
  [43] 1.08318028 1.80000126 1.77417468 1.33500311 1.89669472 1.45175732
  [49] 0.70312516 1.29878288 0.82215427 0.60452982 0.29104405 1.26046647
```



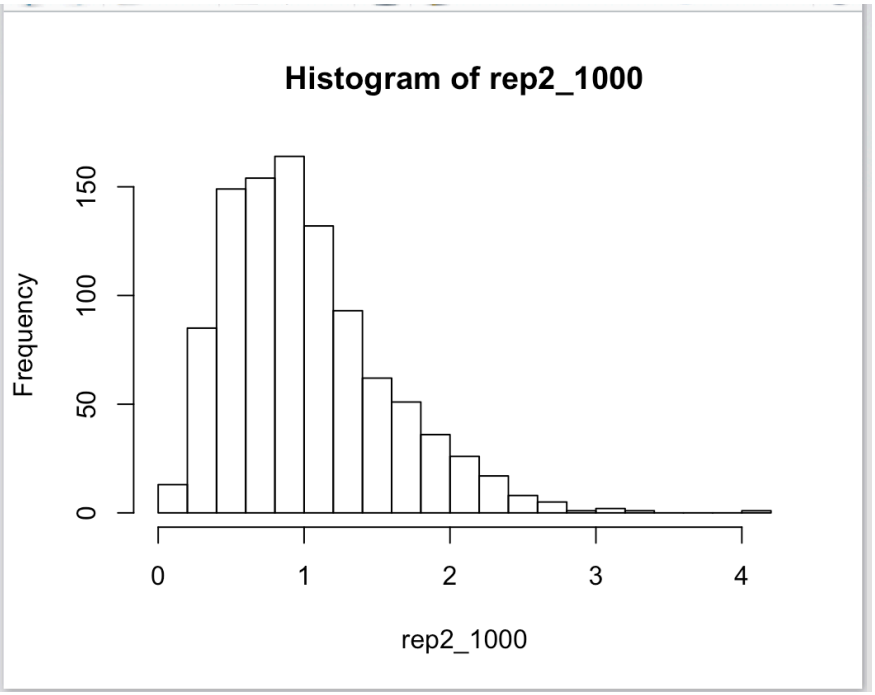
```
 [979] 1.38798903 1.51145947 0.14542974 1.05705633 0.87547491 0.37909478
 [985] 1.69537898 1.91085604 1.24071965 0.37787918 0.93359543 0.31328230
 [991] 0.65978533 3.02216011 0.80705511 0.85159876 1.26469606 0.46910179
 [997] 1.50246618 2.70844448 0.33820213 0.35268587
> par(mfrow=c(1,1))
> hist(rep2_1000, 20)
> #No, it's skewed to the right which is not consistent with the real sigma_squared.
>
>
>
>
>
>
>
>
>
>
>
>
```



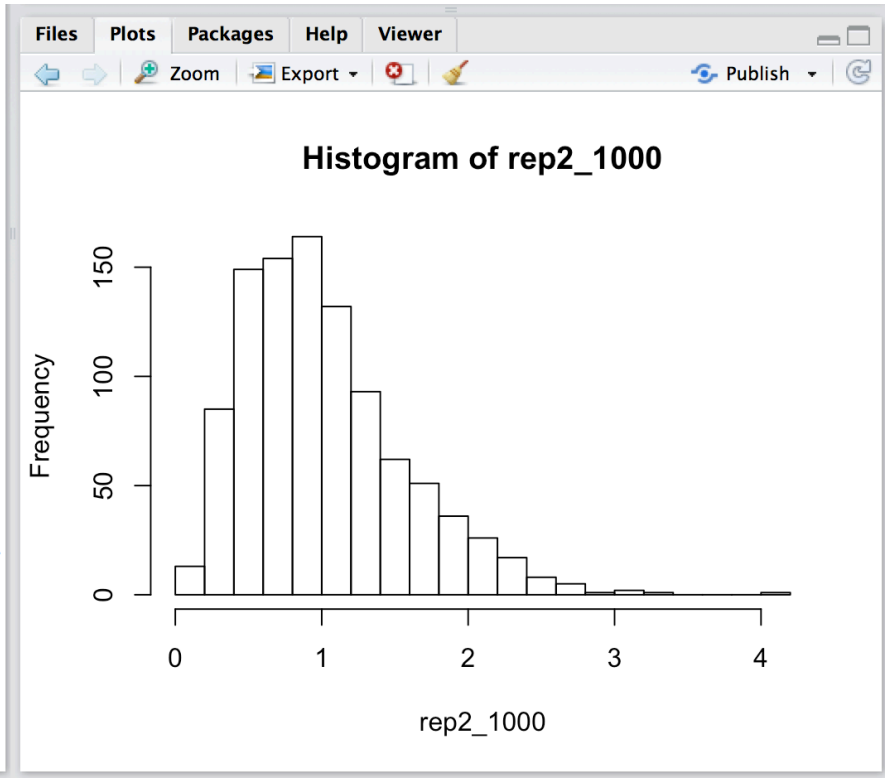
```
> #1(f)
> #repeat 1(d), but use uniform distribution instead. To have a distribution of mean=
0, variance=1, uniform distribution ranges from -sqrt(3) to sqrt(3).
> rep3_1000=replicate(1000,{epsilon=matrix(runif(10, -sqrt(3), sqrt(3)),10, 1)
+ k=c(1,1,1,1,1,1,1,1,1,1,2,-1,3,3,2,1,0,0,-1,0,-2,-2,-2,3,3,3,0,0,0,1)
+ x=matrix(k,10,3)
+ beta=matrix(c(1.5,-1,2),3,1)
+ y=x%*%beta+epsilon
+ y
+ betahat=solve((t(x))%*%x)%*%t(x)%*%y})
> betahat1=rep3_1000[1,,]
> betahat2=rep3_1000[2,,]
> betahat3=rep3_1000[3,,]
> par(mfrow=c(1,3))
> hist(betahat1, 20)
> hist(betahat2,20)
> hist(betahat3,20)
> rep_variance=replicate(1000, {variance=solve((t(x))%*%x)})
> rep_variance
, , 1

             [,1]         [,2]         [,3]
[1,]  0.139180672 -0.042016807 -0.003413866
```



```
> #repeat 1(e)
> rep4_1000=replicate(1000,{epsilon=matrix(runif(10, -sqrt(3), sqrt(3)),10, 1)
+ k=c(1,1,1,1,1,1,1,1,1,2,-1,3,3,2,1,0,0,-1,0,-2,-2,-2,3,3,3,0,0,0,1)
+ x=matrix(k,10,3)
+ beta=matrix(c(1.5,-1,2),3,1)
+ y=x%*%beta+epsilon
+ y
+ betahat=solve((t(x))%*%x)%*%t(x)%*%y
+ residuals=y-x%*%(betahat)
+ n=10
+ p=3
+ sigma_squared_hat=(t(residuals)%*%residuals)/(n-p)
+ sigma_squared_hat})
> rep4_1000
   [1] 1.1528212 1.5425539 1.1864890 1.4133954 0.7020750 0.7011015 1.0853353
   [8] 0.8674718 1.2763360 1.0178298 1.0327253 0.7187822 0.6593050 1.4593949
  [15] 0.7896107 0.4699347 0.5096571 1.0573041 1.0823079 1.4757958 0.4772630
  [22] 1.5276661 1.4534649 1.3326345 0.4985062 0.9416444 1.8948119 0.9148308
  [29] 1.0468781 0.8609546 0.6529702 0.4830985 1.4863136 0.9679671 0.7355441
  [36] 1.5590870 0.6300119 1.3753081 0.8474735 1.8690905 1.4119723 0.6167742
  [43] 0.8813804 1.1488219 0.9162191 1.3411262 1.2083320 0.9906364 1.1947783
  [50] 0.9633744 1.0429800 0.6195582 1.1021166 1.3809231 0.5515925 1.3873644
  [57] 1.2200237 1.8325040 0.6472582 0.6858052 0.9234991 0.7441810 1.5702445
  [64] 0.3764530 0.1384158 1.3045534 1.1517689 0.8212319 1.3194713 1.3087337
```



```
 [897] 0.9933875 1.1150075 1.2787153 0.3875909 1.2462545 0.6443029 1.1723503
 [904] 1.2065607 0.9956747 0.8495592 1.2113481 0.7828551 1.0365298 0.6327345
 [911] 1.0211896 0.9198102 0.9389786 0.6663179 1.3779159 1.4829172 1.7938918
 [918] 1.0490833 1.2703584 0.9129159 1.4768640 0.3526069 1.4549413 0.4332808
 [925] 0.6293008 1.0221430 1.2769555 1.1040372 1.3199864 0.6897757 1.2957216
 [932] 0.4272256 0.5243997 1.5427640 1.3834458 0.7069348 1.1581798 0.4600917
 [939] 0.4201916 1.4929266 0.6579147 1.5775417 1.4655874 0.8844907 0.5770278
 [946] 0.9653191 1.1316370 1.0786449 1.0809679 0.3658806 1.6394989 1.1012136
 [953] 1.0516968 0.9412177 0.3337220 0.3509925 1.4304940 0.8807559 1.0841660
 [960] 1.2390255 1.0751085 1.1124788 0.7161074 1.2412565 0.6051931 0.7614783
 [967] 1.2644418 0.9011759 0.8646296 0.7700526 1.0886670 0.8813749 0.9890500
 [974] 1.0823453 1.0219535 0.8650129 1.3916074 0.7062075 1.7458684 1.0049160
 [981] 1.0738469 0.9414338 1.1558207 0.5400250 0.7725986 0.4099463 1.7391587
 [988] 0.5869425 1.2430819 0.8030045 1.1047086 1.3877885 0.7936096 1.7349253
 [995] 0.7899202 1.3969442 0.9242031 0.8634626 0.4353390 0.5600170
> par(mfrow=c(1,1))
> hist(rep2_1000, 20)
>
> #It doesn't change much.
> #The results of the original question wouldn't change much if the residuals all fol
low a distribution with the same expectation and variance.
>
>
>
>
>
> |
```



```
> #3(a)
> data(prostate, package="faraway")
> limod=lm(lpsa~lcavol+lweight+age+lbph+svi+lcp+gleason+pgg45,data=prostate)
> summary(limod)

Call:
lm(formula = lpsa ~ lcavol + lweight + age + lbph + svi + lcp +
    gleason + pgg45, data = prostate)

Residuals:
    Min      1Q  Median      3Q     Max
-1.7331 -0.3713 -0.0170  0.4141  1.6381

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.669337   1.296387   0.516  0.60693
lcavol       0.587022   0.087920   6.677 2.11e-09 ***
lweight      0.454467   0.170012   2.673  0.00896 **
age         -0.019637   0.011173  -1.758  0.08229 .
lbph         0.107054   0.058449   1.832  0.07040 .
svi          0.766157   0.244309   3.136  0.00233 **
lcp         -0.105474   0.091013  -1.159  0.24964
gleason      0.045142   0.157465   0.287  0.77503
pgg45        0.004525   0.004421   1.024  0.30886
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7084 on 88 degrees of freedom
Multiple R-squared:  0.6548,    Adjusted R-squared:  0.6234
F-statistic: 20.86 on 8 and 88 DF,  p-value: < 2.2e-16

> predict(limod, new=data.frame(lcavol=1.45, lweight=3.59801,age=63,lbph=0.3001,svi=0
, lcp=-0.79851,gleason=7.0000, pgg45=15), interval="prediction")
       fit       lwr      upr
1 2.418773 0.9939479 3.843599
```

```
> #3(b)
> predict(limod, new=data.frame(lcavol=1.45, lweight=3.59801,age=20,lbph=0.3001,svi=0
, lcp=-0.79851,gleason=7.0000, pgg45=15), interval="prediction")
       fit       lwr      upr
1 2.361519 0.9272333 3.795805
>
> #3(c)
> #Remove the elements that are insignificant at 5% significance level
> #Remove age, lbph, lcp, gleason, pgg45
> limod=lm(lpsa~lcavol+lweight+svi,data=prostate)
> #Repeat the prediction in question a and b
> predict(limod, new=data.frame(lcavol=1.45, lweight=3.59801,age=63,lbph=0.3001,svi=0
, lcp=-0.79851,gleason=7.0000, pgg45=15), interval="prediction")
       fit       lwr      upr
1 2.361519 0.9272333 3.795805
> #The prediction is wider than question(a) but narrower than question(b)
> #I prefer the predictions in (c), since it removes the elements not significant, th
e prediction is more precise.
>
```