Tarea sobre los Temas 3 a 5

Grado en Estadística. Universidad de Sevilla

Marta Venegas Pardo

Contents

1 Pregunta 1 (datasets::precip)

En el dataset datasets::precip se recoge la cantidad media de precipitaciones de 70 ciudades de Estados Unidos (unidad = inches).

Extraigo los datos

```
precip<-datasets::precip
# View(precip)
```

```
#dotchart(precip[order(precip)], main = "precip data")
#title(sub = "Average annual precipitation (in.)")
```

```
precip_data<-as.data.frame(precip)
names<-names(precip)
precipdf<-cbind.data.frame(names,precip_data)
colnames(precipdf)<-c("City","Precip")



cbind.data.frame(precipdf[1:35,],precipdf[36:70,])%>%
  kable(booktabs=TRUE,longtable=TRUE) %>%
```

```
  kable_styling(latex_options = c("striped","scale_down")) %>%
  footnote(general = "Annual Precipitation in US Cities")
```

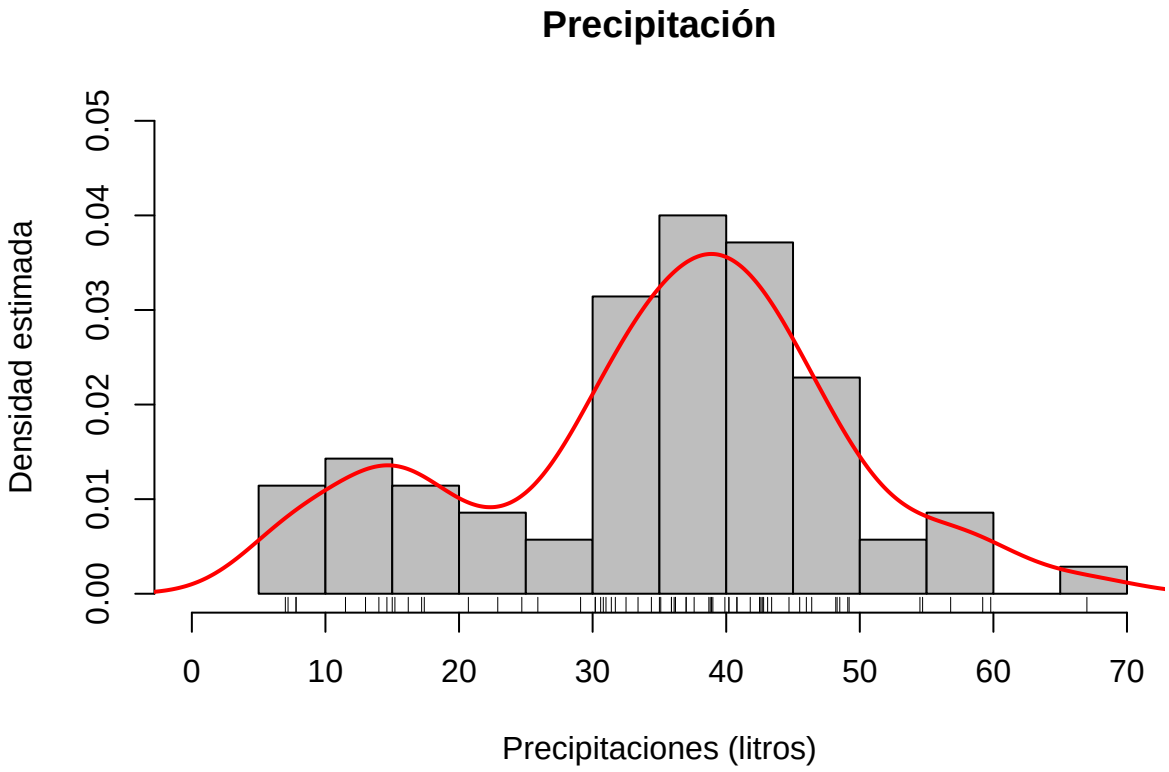
```
## Warning in styling_latex_scale_down(out, table_info): Longtable cannot be
## resized.
```

City	Precip	City	Precip
Mobile	67.0	Reno	7.2
Juneau	54.7	Concord	36.2
Phoenix	7.0	Atlantic City	45.5
Little Rock	48.5	Albuquerque	7.8
Los Angeles	14.0	Albany	33.4
Sacramento	17.2	Buffalo	36.1
San Francisco	20.7	New York	40.2
Denver	13.0	Charlotte	42.7
Hartford	43.4	Raleigh	42.5
Wilmington	40.2	Bismark	16.2
Washington	38.9	Cincinnati	39.0
Jacksonville	54.5	Cleveland	35.0
Miami	59.8	Columbus	37.0
Atlanta	48.3	Oklahoma City	31.4
Honolulu	22.9	Portland	37.6
Boise	11.5	Philadelphia	39.9
Chicago	34.4	Pittsburg	36.2
Peoria	35.1	Providence	42.8
Indianapolis	38.7	Columbia	46.4
Des Moines	30.8	Sioux Falls	24.7
Wichita	30.6	Memphis	49.1
Louisville	43.1	Nashville	46.0
New Orleans	56.8	Dallas	35.9
Portland	40.8	El Paso	7.8
Baltimore	41.8	Houston	48.2
Boston	42.5	Salt Lake City	15.2
Detroit	31.0	Burlington	32.5
Sault Ste. Marie	31.7	Norfolk	44.7
Duluth	30.2	Richmond	42.6
Minneapolis/St Paul	25.9	Seattle Tacoma	38.8
Jackson	49.2	Spokane	17.4
Kansas City	37.0	Charleston	40.8
St Louis	35.9	Milwaukee	29.1
Great Falls	15.0	Cheyenne	14.6
Omaha	30.2	San Juan	59.2

Note:
Annual Precipitation in US Cities

1.1 Realizar una estimación no paramétrica de la función de densidad por el método del núcleo.

```
x=precipdf[,2]
hist(precipdf[,2], br=10,prob=TRUE, main="Precipitación",
     col="gray",
     xlab="Precipitaciones (litros)", ylab="Densidad estimada",
     ylim = c(0,0.05),xlim = c(0,70))
lines(density(x, bw="SJ"),lwd=2,col="red")
rug(x)
```



Hemos elegidio el método SJ, que implementa la propuesta de Sheather y Jones, basado en la estimación del núcleo de f''

1.2 Realizar una estimación no paramétrica de la función de densidad por el método de los logsplines.

Estima el logaritmo de la función de densidad mediante un spline cúbico.

```
library(logspline)
ajuste <- logspline(precip_data)
ajuste # 7nudos, criterio BIC
```

```
##  knots A(1)/D(2)  loglik    AIC minimum penalty maximum penalty
##      4         2 -282.85 578.45           15.02             Inf
##      5         2 -275.34 567.68            0.51           15.02
##      6         2 -275.29 571.82              NA              NA
##      7         2 -274.83 575.15            0.41            0.51
##      8         2 -274.79 579.31              NA              NA
##      9         2 -274.41 582.82            0.02            0.41
##     10         1 -274.40 587.04            0.00            0.02
## the present optimal number of knots is  5
## penalty(AIC) was the default: BIC=log(samplesize): log( 70 )= 4.25
```

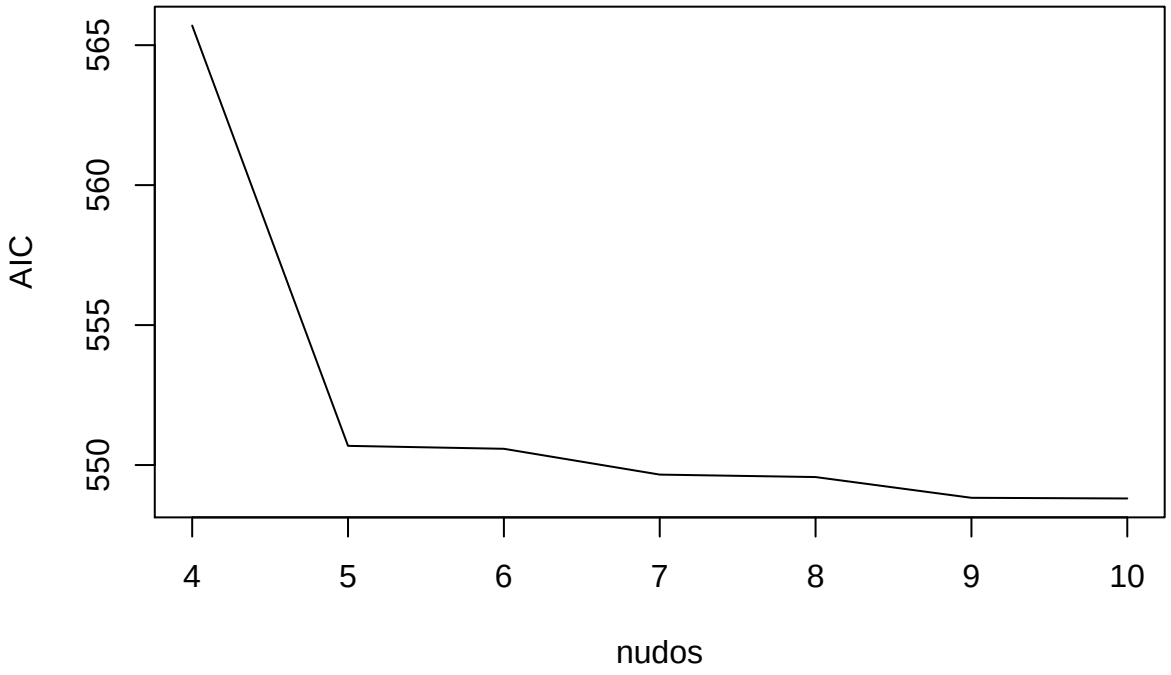
Vamos a comprobar los valores AIC

```
resul<- ajuste$logl
nudos<- resul[,1]
logL<- resul[,3]
AIC<- -2*logL+log(length(precip_data))*(nudos-1)
AIC
```

```
## [1] 565.7001 550.6848 550.5803 549.6576 549.5704 548.8295 548.8057
```

Lo dibujamos

```
plot(nudos,AIC,type="l")
```



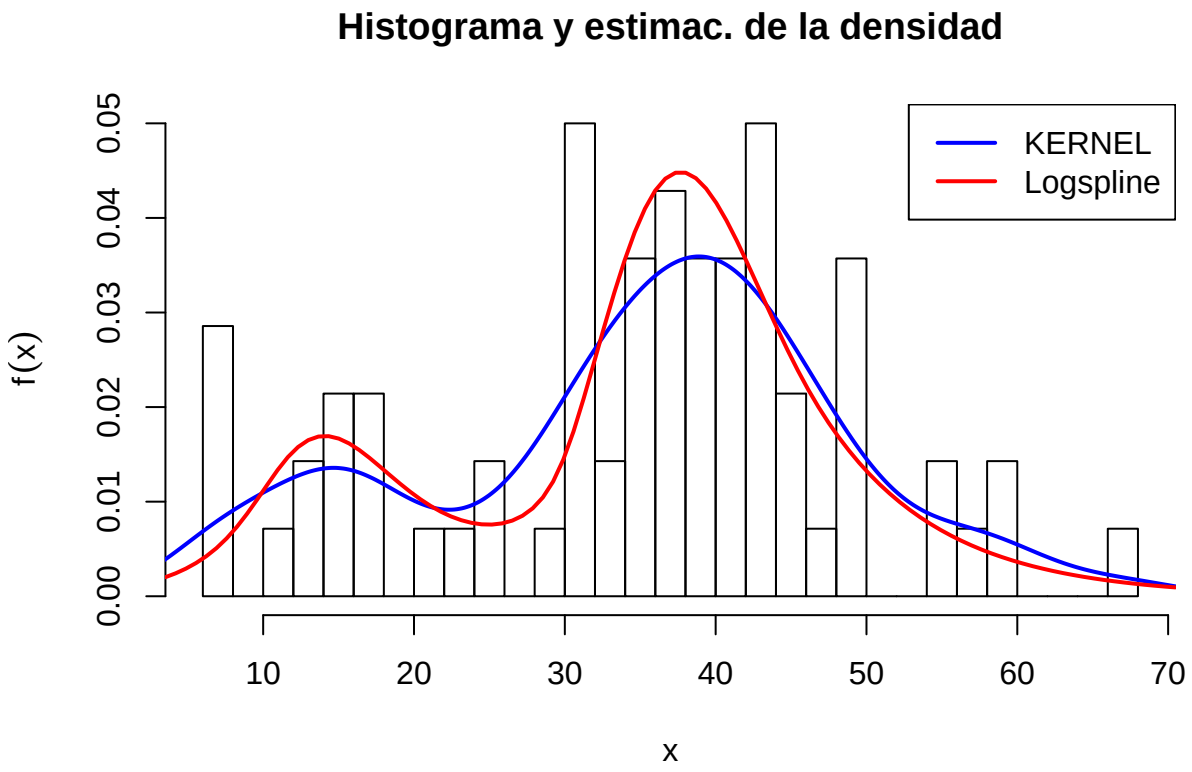
Ahora el histograma

```
hist(x,br=30, prob=TRUE,
     main="Histograma y estimac. de la densidad",
     ylab = expression(hat(f)(x)),
     xlab="x")

lines(density(precipdf[,2],bw="SJ"),col="blue",lwd=2)

plot(ajuste,col="red",lwd=2,add=TRUE)

legend("topright",
       col=c("blue","red"),
       lwd=2,
       legend=c("KERNEL","Logspline"))
```



1.3 Estimar P[precip>42] y el cuantil 0.90.

```
1-plogspline(42, ajuste)
```

```
## [1] 0.2865462
```

```
qlogspline(0.9, ajuste)
```

```
## [1] 50.17582
```

2 Pregunta 2

Cuando cierta empresa recibe una invitación para optar a un contrato, la oferta no se puede completar hasta que sea revisada por cuatro departamentos: Ingeniería, Personal, Legal y Contabilidad. Los departamentos empiezan a trabajar al mismo tiempo, pero lo hacen de forma independiente. El tiempo en semanas que emplean en completar la revisión es una variable aleatoria con las siguientes distribuciones:

- Ingeniería: Exponencial con media 3 semanas
- Personal: Normal con media 4 y desviación típica 1;
- Legal: 2 o 4 semanas, siendo ambos valores equiprobables
- Contabilidad: Uniforme continua en el intervalo (1,5).

Se trata de simular el tiempo W que tarda la empresa en preparar una oferta. Para ello se pueden implementar los siguientes pasos:

2.1 Fijar M (número de ofertas, se recomienda al menos 1000).

```
M=1000 # número de ofertas
```

2.2 Definir una matriz Mx4 donde se irán almacenando los valores generados.

```
matriz=matrix(NA,M,4) # Posteriormente almacenaremos los datos
dim(matriz)
```

```
## [1] 1000    4
```

2.3 Repetir M veces:

2.3.1 Generar de forma independiente los cuatro tiempos según las cuatro distribuciones.

```
W = rep(NA,M)
for (i in 1:M) { # Guardar esos cuatro tiempos en una fila de la matriz
  vIng = rexp(1,1/3)
  VPer = rnorm(1,4,1)
  x=c(2,4)
  vLeg = sample(x,1,replace = T)
  vCon = runif(1,1,5)


# Para cada i genero un valor de W
  W[i] = max(vIng,VPer,vLeg,vCon) #Calcular W como el máximo de los cuatro tiempos.
}
# Al final tendrás una m.a.s. de valores W de tamaño M: W_1, ..., W_M
```

Creo la matriz

```
m= matrix(W,ncol = 4)
colnames(m)<- c("Ingeniería","Personal","Legal","Contabilidad")
head(m)
```

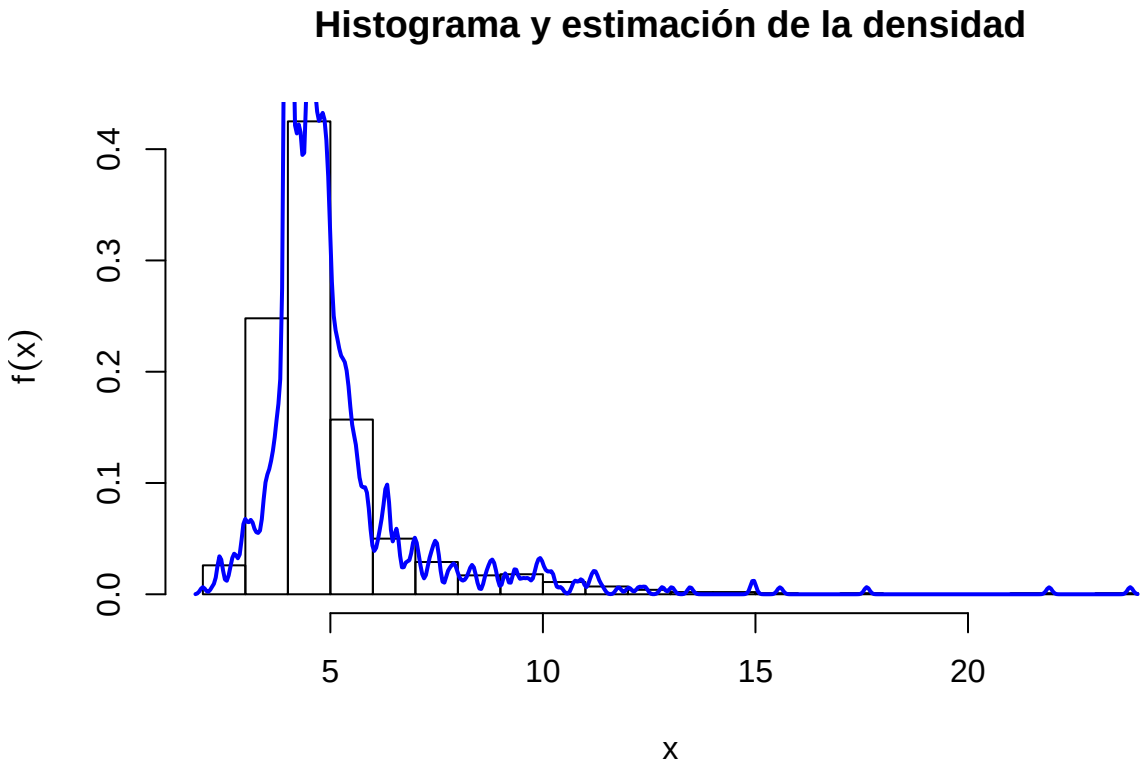
```
##      Ingeniería Personal    Legal Contabilidad
## [1,]   4.918593 4.197908 6.177819     4.962226
## [2,]   4.000000 8.746336 5.177417     6.357534
## [3,]   3.447742 4.000000 4.513778     4.064007
## [4,]   4.843849 4.985646 4.982204     4.663591
## [5,]   4.000000 4.518703 4.202954     4.000000
```

```
## [6,]   4.825545 5.656200 4.318150     5.372187
```

Se pide:

- Estudiar gráficamente la distribución de la variable aleatoria W "tiempo que transcurre hasta completar la oferta".

```
hist(W,
     br=30,
     prob=TRUE,
     main="Histograma y estimación de la densidad",
     ylab=expression(hat(f)(x)),xlab="x")
lines(density(W,bw="SJ"),col="blue",lwd=2)
```



Estimar su media y su mediana.

```
apply(X=m,FUN=median,MARGIN = 2) # Mediana por columnas
```

```
##   Ingeniería     Personal        Legal Contabilidad
##     4.571889     4.542175     4.466591     4.569577
```

```
apply(X=m,FUN=mean,MARGIN = 2) # Media por columnas
```

```
##   Ingeniería     Personal        Legal Contabilidad
##     5.103914     5.004222     4.936415     5.045140
```

Para los datos totales, sin hacerlo por departamento:

Estimación del tiempo medio

```
mean(W)
```

```
## [1] 5.022423
```

Estimación de la mediana:

```
median(W)
```

```
## [1] 4.540465
```

2.3.2 Estimar la probabilidad de que W supere las 6 semanas.

$$P[W > 6]$$

```
ajuste<- logspline(m)
ajuste
```

```
##  knots A(1)/D(2)   loglik     AIC minimum penalty maximum penalty
##      7         2 -1482.64 3006.73           41.05             Inf
##      8         2 -1462.11 2972.58            7.81           41.05
##      9         2 -1461.36 2977.99              NA              NA
##     10         2 -1455.60 2973.36              NA              NA
##     11         2 -1450.40 2969.87            3.36            7.81
##     12         2 -1448.72 2973.42            2.61            3.36
##     13         2 -1447.41 2977.71            1.36            2.61
##     14         2 -1447.15 2984.11              NA              NA
##     15         2 -1446.05 2988.81            0.66            1.36
##     16         2 -1445.72 2995.06            0.00            0.66
##     17         1 -1445.72 3001.97            0.00            0.00
## the present optimal number of knots is  11
## penalty(AIC) was the default: BIC=log(samplesize): log( 1000 )= 6.91
```

Luego, la probabilidad será:

```
1-plogspline(6, ajuste)
```

```
## [1] 0.1416481
```

2.3.3 ¿Cuál es el departamento que suele tardar más en completar la revisión?

```
m %>%
  colSums()/M
```

```
##   Ingeniería     Personal        Legal Contabilidad
##     1.275978     1.251056     1.234104     1.261285
```

El tiempo medio de cada departamento, por tanto, vemos que el departamento de personal es el que es más lento a la hora de hacer una revisión completa.

2.3.4 ¿Cuál es la ordenación más frecuente de los cuatro tiempos?

```
library(modeest)
```

```
## Registered S3 method overwritten by 'rmutil':
##   method         from
##   print.response httr
```

```
apply(X=m,FUN=mfv,MARGIN = 2)
```

```
##   Ingeniería     Personal        Legal Contabilidad
##            4            4            4            4
```

```
mfv(W)
```

```
## [1] 4
```

4 unidades de tiempo (horas)

3 Pregunta 3

Generar aleatoriamente un conjunto de datos donde tengo sentido construir un modelo de clasificación o de predicción. Ajustar el modelo y estimar su capacidad de generalización mediante Jackknife y mediante Validación Cruzada (K=10).

3.1 Generamos los datos con la función sample(aleatoriamente).

Generaremos un dataset con 120 filas y 3 variables, por ejemplo.

```
V=c(rep("Z1",70),rep("Z2",50))
V1=runif(120,5,13)
V2=rnorm(120,80,25)
V3=rnorm(120,30,5)


dataSet=cbind.data.frame(V,V1,V2,V3)

head(dataSet)
```

```
##    V        V1        V2       V3
## 1 Z1  7.831134  94.19140 26.67137
## 2 Z1  5.827370  70.00436 21.49309
## 3 Z1  5.622763  60.25513 34.54121
## 4 Z1 12.982223  56.82322 21.82461
## 5 Z1 11.911554  77.00313 33.33084
## 6 Z1  8.652191 112.66591 26.18429
```

```
summary(dataSet)
```

```
##   V            V1               V2               V3
##  Z1:70   Min.   : 5.132   Min.   : 16.76   Min.   :18.73
##  Z2:50   1st Qu.: 7.122   1st Qu.: 60.51   1st Qu.:26.83
##          Median : 8.436   Median : 77.24   Median :30.64
##          Mean   : 8.891   Mean   : 78.90   Mean   :30.32
##          3rd Qu.:10.789   3rd Qu.: 93.24   3rd Qu.:34.00
##          Max.   :12.982   Max.   :147.90   Max.   :42.55
```

```
dataSet$V = as.factor(dataSet$V)
summary(dataSet)
```

```
##   V            V1               V2               V3
##  Z1:70   Min.   : 5.132   Min.   : 16.76   Min.   :18.73
##  Z2:50   1st Qu.: 7.122   1st Qu.: 60.51   1st Qu.:26.83
##          Median : 8.436   Median : 77.24   Median :30.64
##          Mean   : 8.891   Mean   : 78.90   Mean   :30.32
##          3rd Qu.:10.789   3rd Qu.: 93.24   3rd Qu.:34.00
##          Max.   :12.982   Max.   :147.90   Max.   :42.55
```

3.2 Modelo

Elegimos como variable respuesta la variable V

```
modelo = glm(V~., data = dataSet, family = "binomial")
summary(modelo)
```

```
##
```

```
## Call:
## glm(formula = V ~ ., family = "binomial", data = dataSet)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.372  -1.033  -0.903   1.276   1.606
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.692907   1.387569  -1.220    0.222
## V1           0.047113   0.080756   0.583    0.560
## V2          -0.007299   0.007448  -0.980    0.327
## V3           0.049616   0.036778   1.349    0.177
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 163.01  on 119  degrees of freedom
## Residual deviance: 160.00  on 116  degrees of freedom
## AIC: 168
##
## Number of Fisher Scoring iterations: 4
```

Intervalo de confianza para las estimaciones de las variables, al 95% de nivel de significación:

```
confint(modelo)
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %      97.5 %
## (Intercept) -4.47787589 0.998929819
## V1          -0.11127182 0.206982790
## V2          -0.02226422 0.007142738
## V3          -0.02155000 0.123606470
```

```
exp(coef(modelo))
```

```
## (Intercept)          V1          V2          V3
##   0.1839839   1.0482404   0.9927272   1.0508679
```

3.3 Predicciones

```
predicciones = predict.glm(modelo,newdata = dataSet)
head(predicciones)
```

```
##           1           2           3           4           5           6
## -0.68815954 -0.86294047 -0.15401625 -0.41319157 -0.04003692 -0.80849657
```

```
probabilidad = exp(predicciones)/(1+exp(predicciones))
head(probabilidad)
```

```
##         1         2         3         4         5         6
## 0.3344426 0.2967254 0.4615719 0.3981471 0.4899921 0.3082110
```

Definimos las funciones:

```
abinario=function(x)
{if (x>0.5) {return(1)}
  else{return(0)}
```

```
}
elfactor=function(x)
{if (x==0) {return(("Z1"))}
  if (x==1){return(("Z2"))}}
```

```
estimacionbi=lapply(probabilidad,abinario)
estimacion= lapply(estimacionbi, elfactor)
```

Vamos a ver la tabla con las predicciones, donde podremos ver si nos hemos equivocado o no al predecir.

```
Prediccion=t(as.data.frame(estimacion))
Tabla_Completa=cbind.data.frame(dataSet,Prediccion)
Tabla_Completa
```

```
##       V        V1        V2       V3 Prediccion
## X1   Z1  7.831134  94.19140 26.67137         Z1
## X2   Z1  5.827370  70.00436 21.49309         Z1
## X3   Z1  5.622763  60.25513 34.54121         Z1
## X4   Z1 12.982223  56.82322 21.82461         Z1
## X5   Z1 11.911554  77.00313 33.33084         Z1
## X6   Z1  8.652191 112.66591 26.18429         Z1
## X7   Z1 11.730197  60.56883 18.92951         Z1
## X8   Z1 12.325278 128.99552 32.28219         Z1
## X9   Z1 12.605566  64.22190 24.59744         Z1
## X10  Z1  7.145628  82.83096 19.15777         Z1
## X11  Z1  9.300621  86.95023 29.42450         Z1
## X12  Z1 10.083601 147.89515 33.99443         Z1
## X13  Z1  7.163103  71.27830 30.38798         Z1
## X14  Z1  7.263000  24.88259 28.14099         Z1
## X15  Z1 10.267461  60.34862 25.41482         Z1
## X16  Z1 10.742792  61.29154 22.08805         Z1
## X17  Z1  6.998548  77.48313 26.58654         Z1
## X18  Z1  6.197913  65.13540 32.25886         Z1
## X19  Z1  9.020639  57.10875 36.00443         Z2
## X20  Z1  8.101401  60.35220 25.56787         Z1
## X21  Z1  8.439965  68.32712 28.01212         Z1
## X22  Z1  9.679517  68.71976 32.45281         Z1
## X23  Z1 12.147851  52.65086 35.50419         Z2
## X24  Z1  7.876671  30.20798 26.38783         Z1
## X25  Z1  8.043430  84.29023 34.20148         Z1
## X26  Z1  5.565976 143.98155 21.35245         Z1
## X27  Z1  6.453466 101.88358 35.99024         Z1
## X28  Z1  5.394768  95.64354 35.61459         Z1
## X29  Z1  8.864202  83.93568 39.11952         Z2
## X30  Z1 10.467703  88.26721 25.48799         Z1
## X31  Z1  5.537961  67.60440 29.71588         Z1
## X32  Z1  7.928163  75.16567 26.83575         Z1
## X33  Z1  6.480349  86.23006 23.71894         Z1
## X34  Z1  7.820223  66.72445 39.54153         Z2
## X35  Z1  7.677486 114.58227 31.19308         Z1
## X36  Z1  7.720795  84.14350 27.78934         Z1
## X37  Z1  5.450294  87.92124 25.72074         Z1
## X38  Z1 12.445432 120.17141 27.12906         Z1
## X39  Z1 10.873448  45.00742 31.62934         Z2
## X40  Z1  5.684958 113.65021 34.33892         Z1
```

```
## X41  Z1 12.467580  69.83145 34.49360        Z2
## X42  Z1 11.277025  92.94314 35.95855        Z1
## X43  Z1  7.929373  33.62273 30.43903        Z1
## X44  Z1  5.577896  74.28737 29.40267        Z1
## X45  Z1  8.805604 112.10259 37.29110        Z1
## X46  Z1  9.075268  96.14449 28.70677        Z1
## X47  Z1 12.523052  98.19854 29.11275        Z1
## X48  Z1  6.820646  54.88505 28.66631        Z1
## X49  Z1  8.923869 133.14171 21.45745        Z1
## X50  Z1 10.521934  86.78046 23.58456        Z1
## X51  Z1  9.768695 136.44470 42.55055        Z1
## X52  Z1  7.355360  93.83950 27.13888        Z1
## X53  Z1 12.670261 102.75013 28.40546        Z1
## X54  Z1  7.906679  58.63399 32.37968        Z1
## X55  Z1  6.957814  89.13433 35.97555        Z1
## X56  Z1  6.392689  42.49178 31.48195        Z1
## X57  Z1  7.512713 102.15159 32.63007        Z1
## X58  Z1  7.908162  92.91177 38.46566        Z1
## X59  Z1  5.336029  57.29194 26.42208        Z1
## X60  Z1  7.214377  84.66096 21.64707        Z1
## X61  Z1 11.321607  62.14191 25.61573        Z1
## X62  Z1  7.440743  16.76020 31.64650        Z2
## X63  Z1 11.842764  45.24007 38.53803        Z2
## X64  Z1 12.016202  89.33517 32.66832        Z1
## X65  Z1 11.534531  90.63705 28.30332        Z1
## X66  Z1  8.432653 131.28451 39.99153        Z1
## X67  Z1 12.000421  83.13737 23.05231        Z1
## X68  Z1 12.154961  85.08465 34.47824        Z1
## X69  Z1  6.210381  42.98921 32.20247        Z1
## X70  Z1  7.549891  84.29303 21.05784        Z1
## X71  Z2  6.174942  62.65262 33.53788        Z1
## X72  Z2 10.775734  54.48803 31.77655        Z1
## X73  Z2  7.493594  55.58197 18.72860        Z1
## X74  Z2  8.218211  83.01318 31.67390        Z1
## X75  Z2  5.197023 108.72846 27.93925        Z1
## X76  Z2 10.185262  73.10183 26.81063        Z1
## X77  Z2 10.512585  47.82380 34.02450        Z2
## X78  Z2 10.032009 115.64587 31.13895        Z1
## X79  Z2  5.808289  55.28754 21.02641        Z1
## X80  Z2  9.796831  49.85041 34.89761        Z2
## X81  Z2  5.491389  72.72566 35.10773        Z1
## X82  Z2  9.640288  63.19513 27.70827        Z1
## X83  Z2  6.733612  66.95949 29.31495        Z1
## X84  Z2  8.222864  94.00733 31.45801        Z1
## X85  Z2  8.902374  71.59024 22.20950        Z1
## X86  Z2  7.051733  80.99930 32.58976        Z1
## X87  Z2 12.817999 123.34295 25.24270        Z1
## X88  Z2  7.740516  42.85468 35.84287        Z2
## X89  Z2 12.957635  67.04192 37.57172        Z2
## X90  Z2 12.821471  62.54036 33.09619        Z2
## X91  Z2 12.668668  94.51449 27.87050        Z1
## X92  Z2  9.684571  62.64614 28.55086        Z1
## X93  Z2  6.642805  39.46131 38.01936        Z2
## X94  Z2  5.891419  84.82050 38.91946        Z1
```

```
## X95  Z2  5.651137  68.25035 29.52974         Z1
## X96  Z2  7.690437  88.41875 29.20385         Z1
## X97  Z2 10.178383 116.02441 28.38350         Z1
## X98  Z2  7.954670  88.98050 36.97413         Z1
## X99  Z2  5.131539  88.42339 31.36168         Z1
## X100 Z2  7.502272  86.57443 31.66414         Z1
## X101 Z2 12.694532  94.30663 28.96286         Z1
## X102 Z2  6.759047  43.80888 27.96529         Z1
## X103 Z2  8.051861  83.11431 33.97012         Z1
## X104 Z2 10.114664  74.68359 30.08785         Z1
## X105 Z2  8.152454  98.96326 21.43463         Z1
## X106 Z2  7.466745  36.18870 26.64654         Z1
## X107 Z2 12.940115 123.48492 33.10701         Z1
## X108 Z2 12.715562  67.16032 28.08843         Z1
## X109 Z2 10.827733  80.92316 36.85108         Z2
## X110 Z2  8.707038  52.65347 31.79505         Z1
## X111 Z2 11.177633  60.25000 30.84620         Z1
## X112 Z2  9.808292 124.32632 30.28144         Z1
## X113 Z2  9.428913 104.73967 28.19698         Z1
## X114 Z2 12.565538  68.00812 37.06499         Z2
## X115 Z2  5.614109  75.61683 32.87135         Z1
## X116 Z2 10.650606  93.03713 37.48402         Z1
## X117 Z2 10.848327  91.53065 32.73261         Z1
## X118 Z2  5.517287  49.10817 31.22904         Z1
## X119 Z2 10.020702  59.91042 30.90604         Z1
## X120 Z2 11.498048  74.00052 40.86927         Z2
```

```
length(Tabla_Completa$V)
```

```
## [1] 120
```

```
length(Tabla_Completa$Prediccion)
```

```
## [1] 120
```

```
si=length(which(Tabla_Completa$V==Tabla_Completa$Prediccion))
si/nrow(Tabla_Completa)*100
```

```
## [1] 59.16667
```

3.4 Estimaciones JACKKNIFE

```
n=nrow(dataSet)
prediccionJackknife = numeric(n)
probJack=numeric(n)
for(i in 1:n){
  modelo_i = glm(V~.,data=dataSet[-i,],family = "binomial")
  prJack<-predict.glm(modelo_i,newdata=dataSet[i,])
  probJack[i] = exp(prJack)/(1+exp(prJack))
  binJack= abinario(probJack[i])
  prediccionJackknife[i]=elfactor(binJack)
}

length(which(prediccionJackknife==dataSet$V))*100/n
```

```
## [1] 55
```

```
length(which(prediccionJackknife==Tabla_Completa$Prediccion))*100/n
```

```
## [1] 95.83333
```

3.5 Validación cruzada

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
modelocaret=train(V~.,data=dataSet, method="glm",
                  trControl = trainControl(method="CV",number=10))
modelocaret$results
```

```
##   parameter  Accuracy        Kappa AccuracySD   KappaSD
## 1      none 0.5583333 -0.003207478 0.07905694 0.1742534
```

```
modelocaret$results["Accuracy"]*100
```

```
##   Accuracy
## 1 55.83333
```

4 Pregunta 4

Bootstrap. Implementar una función que calcule el estadístico de Fisher de comparación de coeficientes de correlación lineal:

$$T = \frac{Z_1 - Z_2}{\sqrt{\frac{1}{n_1 - 3} + \frac{1}{n_2 - 3}}}$$

Donde:

- Z1 y Z2 representan la transformación de Fisher de los respectivos coeficientes de correlación lineal para dos grupos
- n1 y n2 son las frecuencias absolutas respectivas de los grupos.

Sobre un conjunto de datos apropiado (que puede ser generado), utilizando el anterior estadístico, realizar e interpretar un test bootstrap bilateral de comparación de los coeficientes de correlación lineal (B=1999).