Smith

Jhonatan Smith Garcia

2/2/2022

Punto 2)

MSVs - aplicado) En este ejercicio, se utilizará el enfoque de máquinas de soporte vectorial para predecir si un automóvil determinado posee un alto o bajo consumo de combustible basado en el conjunto de datos Auto (librería ILSR).

```
#librerias
library(e1071)
library(ISLR)
```

a) Cree una variable binaria que tome un 1 para automóviles con millaje por galón por encima de la mediana, y un 0 para automóviles con millaje por debajo de la mediana.

```
data(Auto)

m <- median(Auto$mpg) # Calculo de la mediana
y <- ifelse(Auto$mpg>m, 1,0) # Si la vble mpg>mediana, agregue 1, si no, agregue 0
y <- as.factor(y) # asigna a la vble creada como categoria, no como numero
dat<- data.frame(Auto,y) # Se crea el nuevo Dataset
```

b). Ajuste un clasificador de soporte vectorial a los datos con varios valores del parámetro cost para predecir si un automóvil posee millaje altomo bajo. Informe los errores de validación cruzada asociados con diferentes valores de este parámetro. Comente sobre sus resultados.

```
#validacion cruzada para mirar el mejor smv con varios valores para cost
set.seed(32668)
vec <- tune(svm ,y~.,data=dat ,kernel ="linear",
              ranges =list(cost=c( 0.01, 0.1,0.5, 1,10,100,1000) ))
```

```
summary(vec)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.007692308
##
## - Detailed performance results:
##    cost       error dispersion
## 1 1e-02 0.073910256 0.04414558
```

```
## 2 1e-01 0.045897436 0.02906486
## 3 5e-01 0.012756410 0.01808165
## 4 1e+00 0.007692308 0.01238579
## 5 1e+01 0.017948718 0.02110955
## 6 1e+02 0.030705128 0.02357884
## 7 1e+03 0.030705128 0.02357884
```

Con cost = 1 se obtiene el menor error de validación cruzada, el cual fue 0.007692308.

c) Ahora repita el inciso anterior, esta vez utilizando una máquina de soporte vectorial (svm) con una base de kernels radiales y polinomiales, con diferentes valores de los hiperparámetros cost, gamma o degree según el kernel y . Comente sus resultados.

kernel radial

```
set.seed(23522)
vec <- tune(svm , y~., data=dat, kernel = "radial",
              ranges=list(cost=c(0.1,1,10,100,1000),
                          gamma=c(0.5,1,2,3,4) ))
```

```
summary(vec)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10   0.5
##
## - best performance: 0.04596154
##
## - Detailed performance results:
##     cost gamma      error dispersion
## 1  1e-01   0.5 0.07910256 0.04058164
## 2  1e+00   0.5 0.04602564 0.03782208
## 3  1e+01   0.5 0.04596154 0.02888973
## 4  1e+02   0.5 0.04596154 0.02888973
## 5  1e+03   0.5 0.04596154 0.02888973
## 6  1e-01   1.0 0.55365385 0.03913705
## 7  1e+00   1.0 0.06128205 0.04841637
## 8  1e+01   1.0 0.06128205 0.05134540
## 9  1e+02   1.0 0.06128205 0.05134540
## 10 1e+03   1.0 0.06128205 0.05134540
## 11 1e-01   2.0 0.55365385 0.03913705
## 12 1e+00   2.0 0.11250000 0.09473562
## 13 1e+01   2.0 0.10987179 0.08036018
## 14 1e+02   2.0 0.10987179 0.08036018
## 15 1e+03   2.0 0.10987179 0.08036018
## 16 1e-01   3.0 0.55365385 0.03913705
## 17 1e+00   3.0 0.39506410 0.15314896
## 18 1e+01   3.0 0.37211538 0.14680319
## 19 1e+02   3.0 0.37211538 0.14680319
## 20 1e+03   3.0 0.37211538 0.14680319
## 21 1e-01   4.0 0.55365385 0.03913705
```

```
## 22 1e+00   4.0 0.49474359 0.04938237
## 23 1e+01   4.0 0.48448718 0.05930309
## 24 1e+02   4.0 0.48448718 0.05930309
## 25 1e+03   4.0 0.48448718 0.05930309
```

Con varios valores para cost (10, 100,1000) y gamma=0.5 proporciona el menor error de validación cruzada, el cual fue 0.04596154.

kernel polinomial

```
set.seed(8527)
tune.out=tune(svm , y~., data=dat, kernel = "polynomial",
              ranges=list(cost=c(0.01,0.1,1),degree=c(2,3,4),
                          gamma=c(0.5,1,2,3) ))
```

```
summary(tune.out)
```

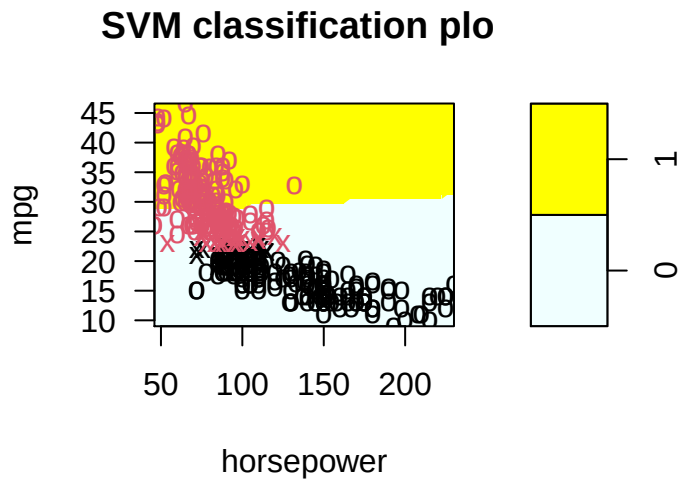
```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree gamma
##   0.1      3     2
##
## - best performance: 0.04096154
##
## - Detailed performance results:
##    cost degree gamma      error dispersion
## 1  0.01      2   0.5 0.28314103 0.05292943
## 2  0.10      2   0.5 0.18102564 0.07060678
## 3  1.00      2   0.5 0.15544872 0.06389228
## 4  0.01      3   0.5 0.07935897 0.05480159
## 5  0.10      3   0.5 0.04615385 0.03783922
## 6  1.00      3   0.5 0.04102564 0.04554842
## 7  0.01      4   0.5 0.18384615 0.07164068
## 8  0.10      4   0.5 0.18121795 0.06907102
## 9  1.00      4   0.5 0.19128205 0.04503957
## 10 0.01      2   1.0 0.27012821 0.07219579
## 11 0.10      2   1.0 0.15294872 0.06467000
## 12 1.00      2   1.0 0.16070513 0.06150881
## 13 0.01      3   1.0 0.04615385 0.03783922
## 14 0.10      3   1.0 0.04358974 0.04689185
## 15 1.00      3   1.0 0.04352564 0.04021235
## 16 0.01      4   1.0 0.18115385 0.05338905
## 17 0.10      4   1.0 0.18871795 0.03406522
## 18 1.00      4   1.0 0.18352564 0.05818565
## 19 0.01      2   2.0 0.16320513 0.07248864
## 20 0.10      2   2.0 0.15544872 0.06156313
## 21 1.00      2   2.0 0.16583333 0.04384879
## 22 0.01      3   2.0 0.04358974 0.04689185
## 23 0.10      3   2.0 0.04096154 0.03669302
## 24 1.00      3   2.0 0.04352564 0.04021235
## 25 0.01      4   2.0 0.17858974 0.04191434
## 26 0.10      4   2.0 0.18352564 0.05818565
```

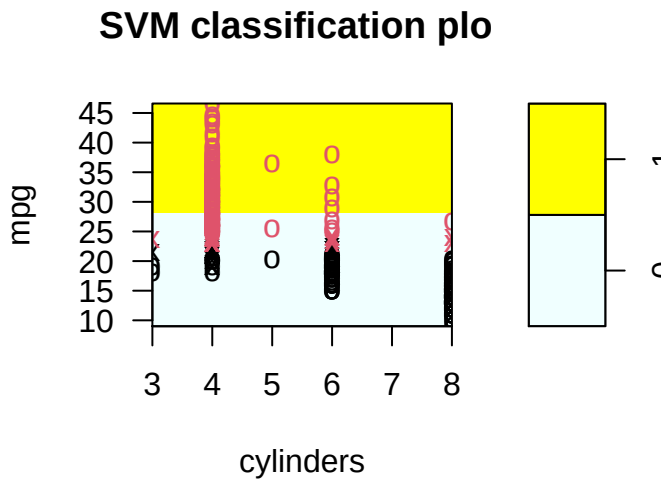
```
## 27 1.00      4   2.0 0.18352564 0.05818565
## 28 0.01      2   3.0 0.15038462 0.06513709
## 29 0.10      2   3.0 0.16070513 0.06150881
## 30 1.00      2   3.0 0.17608974 0.04908399
## 31 0.01      3   3.0 0.04352564 0.04369483
## 32 0.10      3   3.0 0.04352564 0.04021235
## 33 1.00      3   3.0 0.04352564 0.04021235
## 34 0.01      4   3.0 0.18352564 0.05818565
## 35 0.10      4   3.0 0.18352564 0.05818565
## 36 1.00      4   3.0 0.18352564 0.05818565
```

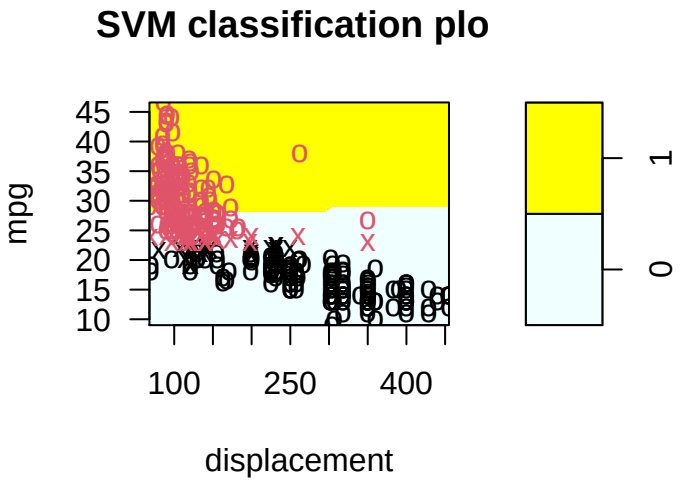
Con cost=0.1, gamma=2 y degree=3 proporciona el menor error de validación cruzada, el cual fue 0.04096154.

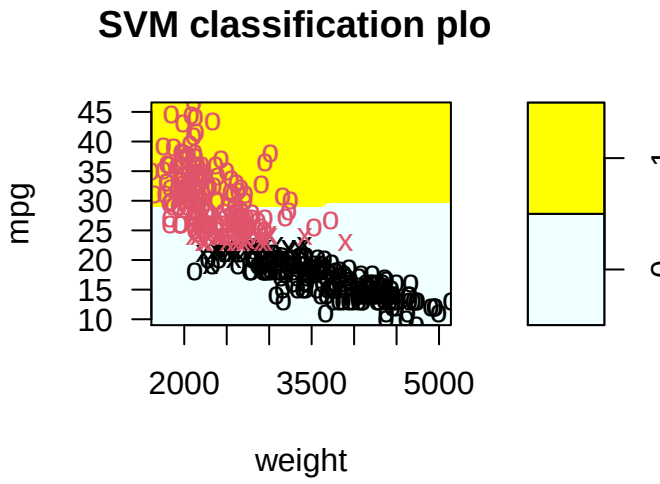
d) Realice algunos plots que sirvan de apoyo a sus afirmaciones en (b) y (c). Recomendación: En el lab, se utilizó la función plot() para objetos svm solo en casos con p = 2. Cuando p > 2, se puede utilizar la función plot() para crear gráficos que muestran pares de variables a la vez.

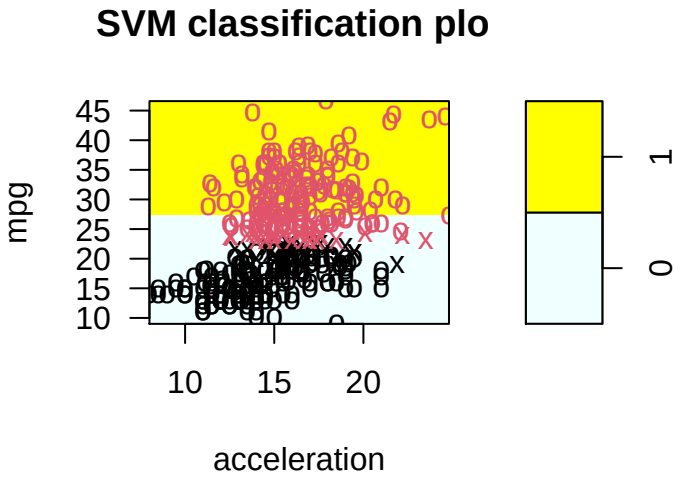
```
#modelo ajustado kernel linear
svm1 =svm(y~., data=dat, kernel = "linear",cost=1)
```

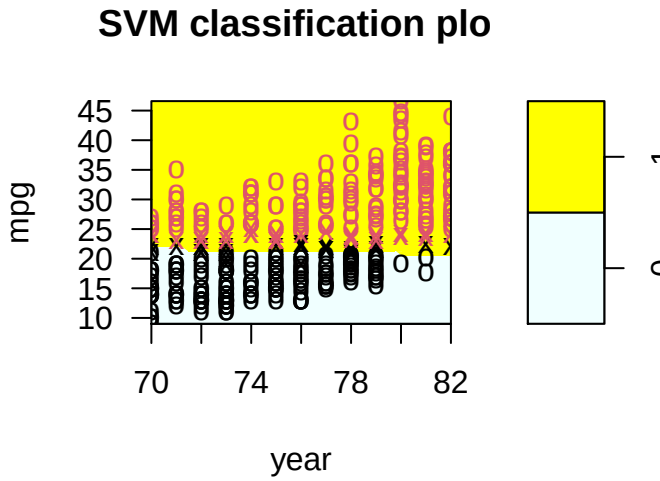


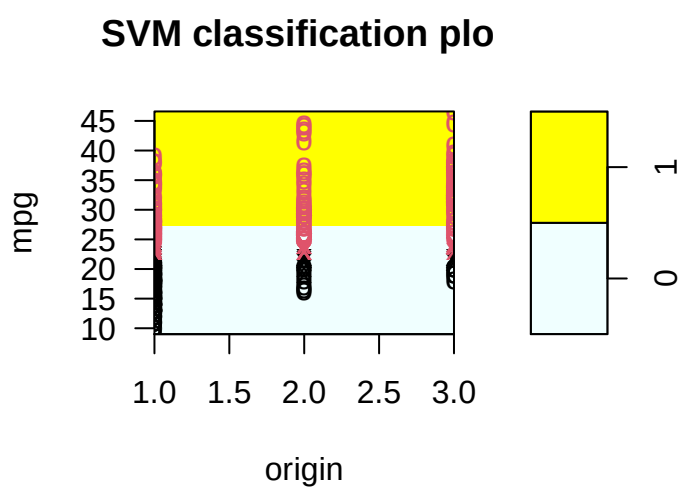



SVM classification plo
mpg
45
40
35
30
25
20
15
10
100
250
400
displacement
1
0


SVM classification plo
mpg
45
40
35
30
25
20
15
10
2000
3500
5000
weight

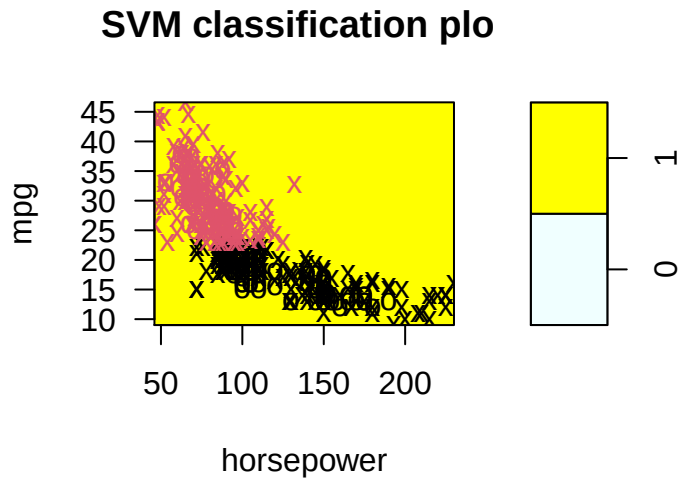

SVM classification plo
mpg
45
40
35
30
25
20
15
10
10
15
20
acceleration
1
0

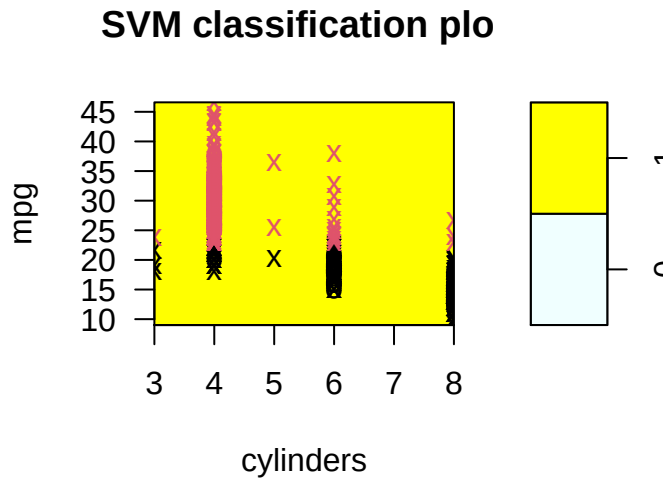

SVM classification plo
mpg
45
40
35
30
25
20
15
10
70
74
78
82
year

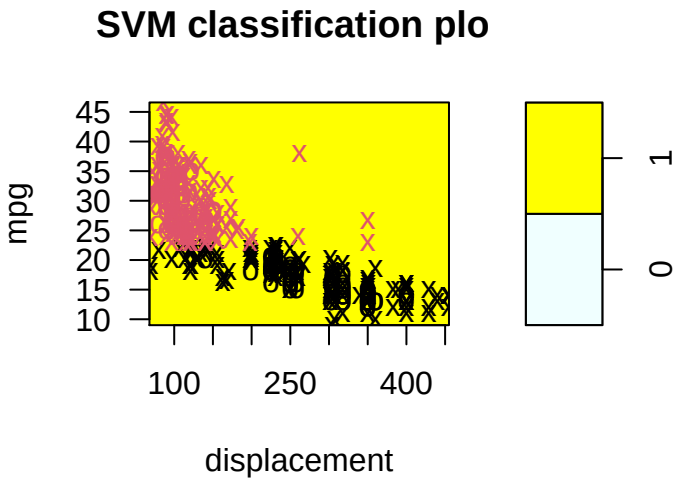


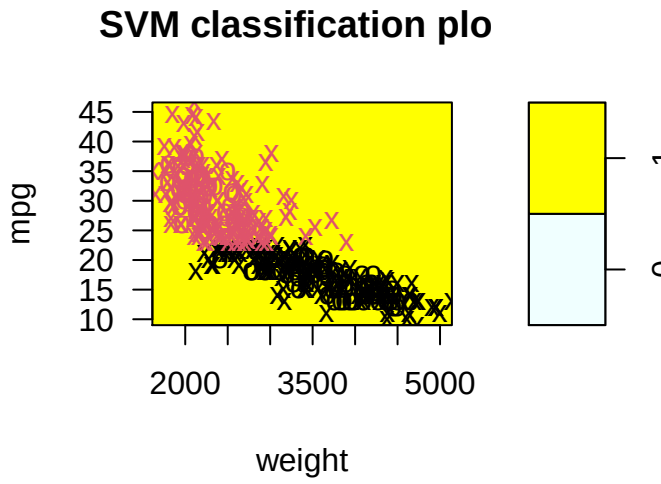
En los gráficos anteriores se observa como clasifica svm para las diferentes variables tomando de a dos, con el valor de cost que dio el error más bajo en la validación cruza.

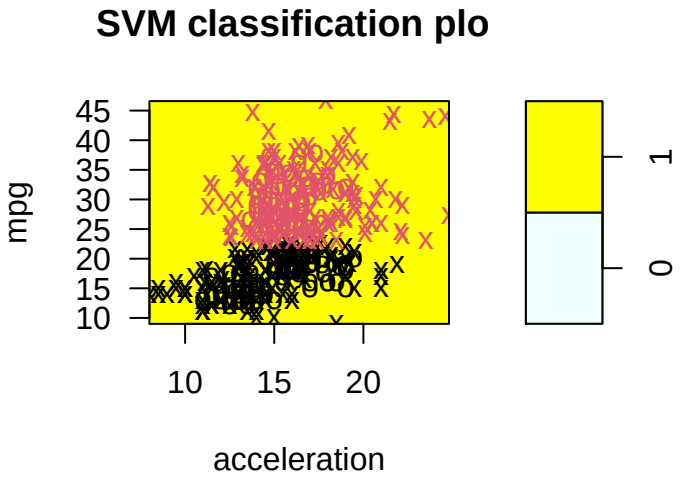
```
#modelo ajustado  kernel radial
svm2 =svm(y~., data=dat, kernel ="radial",cost=10,gamma=0.5)
```

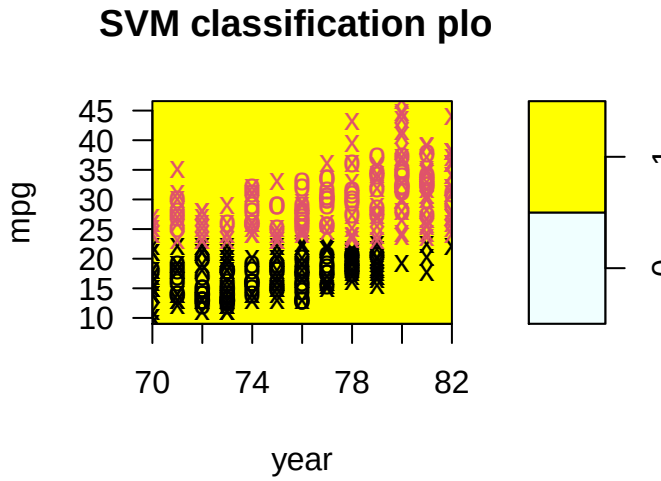


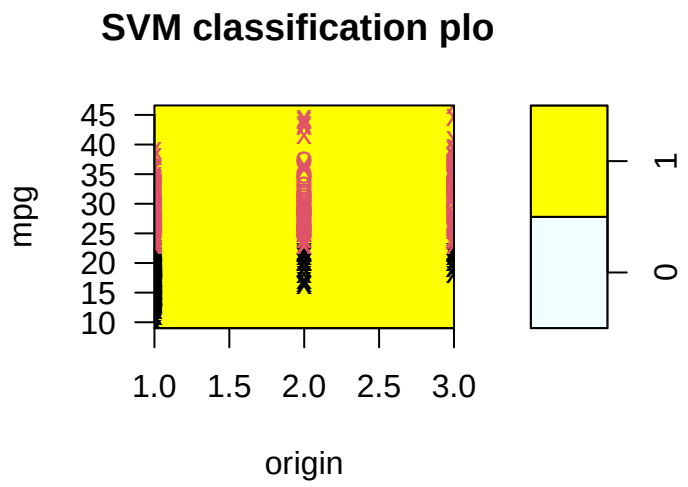



SVM classification plo
mpg
45
40
35
30
25
20
15
10
100
250
400
displacement
1
0


SVM classification plo
mpg
45
40
35
30
25
20
15
10
2000
3500
5000
weight

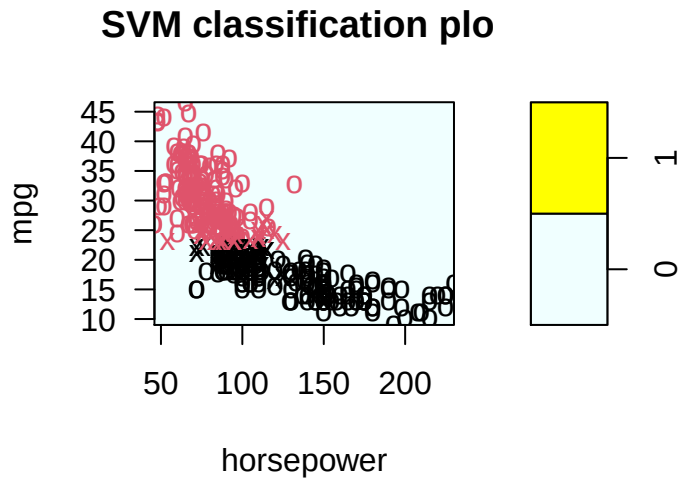

SVM classification plo
mpg
45
40
35
30
25
20
15
10
10
15
20
acceleration
1
0

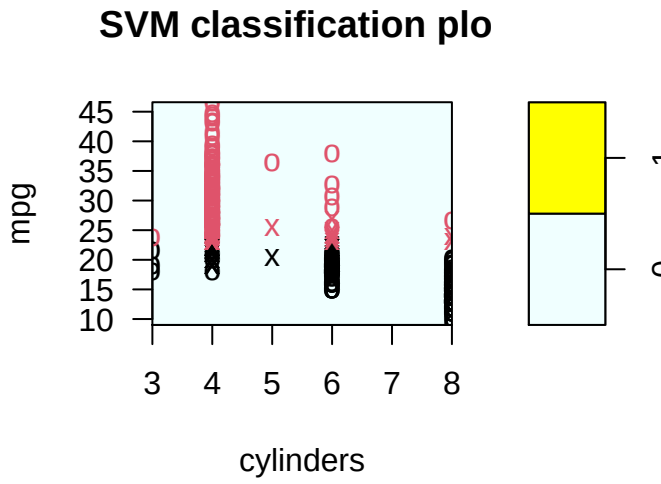

SVM classification plo
mpg
45
40
35
30
25
20
15
10
70
74
78
82
year

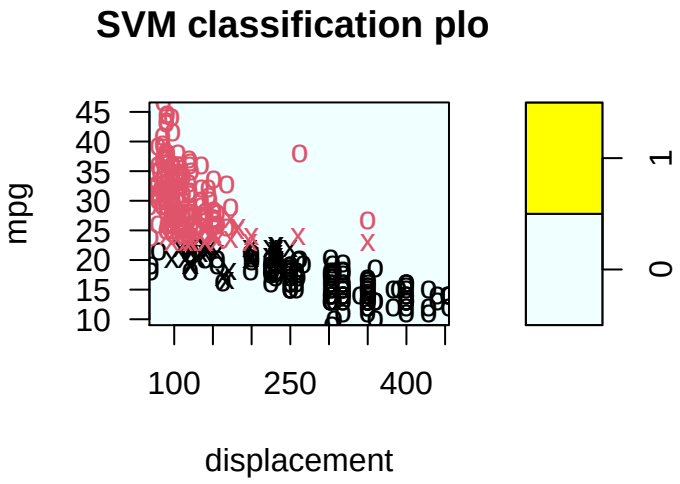


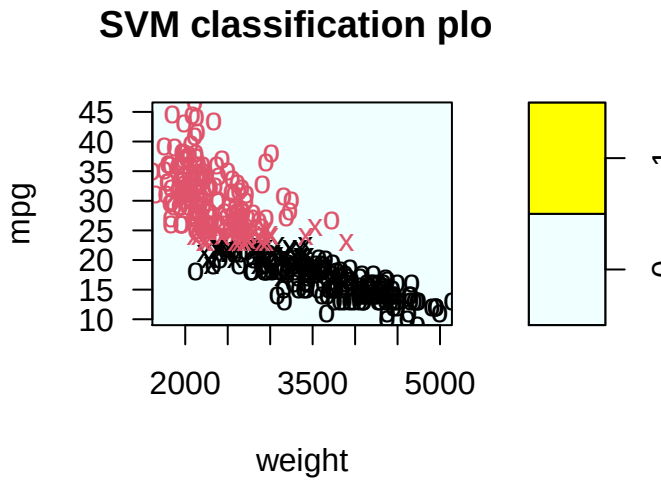
En los gráficos anteriores se observa como clasifica svm para las diferentes variables tomando de a dos, con el valor de cost=10 y gamma=0.05 que fue primero que se detectó con el error más bajo en la validación cruza.

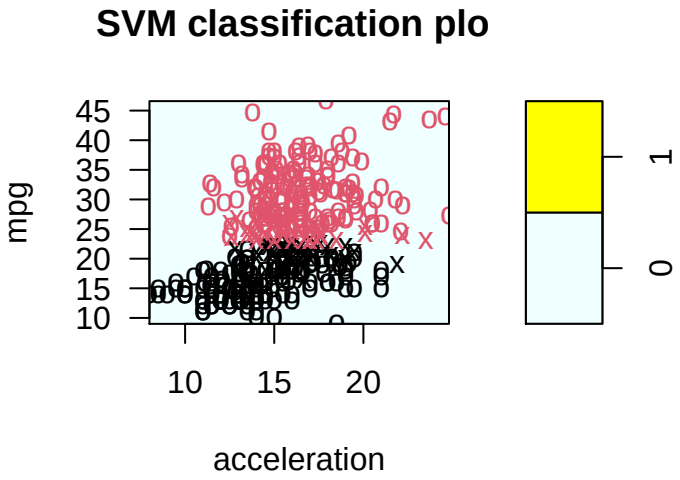
```
#modelo ajustado  kernel polinomial
svm3 =svm(y~., data=dat, kernel="polynomial",
          cost=0.1,gamma=2,degree=3)
```

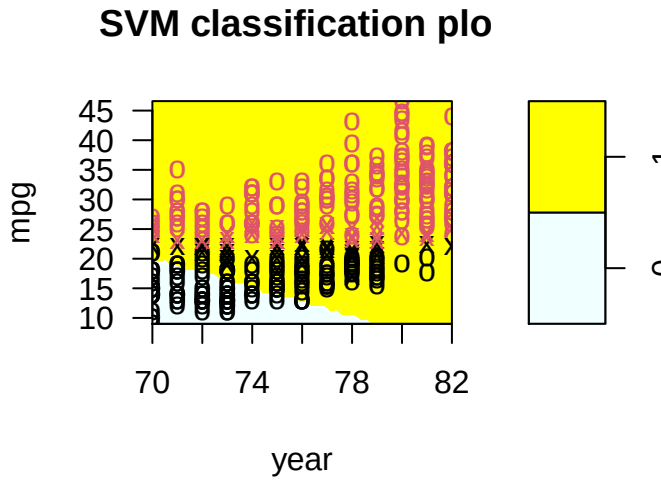


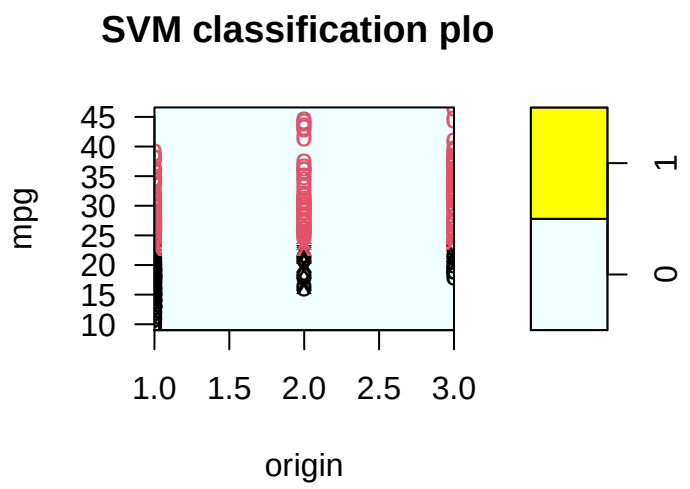



SVM classification plo
mpg
45
40
35
30
25
20
15
10
100
250
400
displacement
1
0


SVM classification plo
mpg
45
40
35
30
25
20
15
10
2000
3500
5000
weight


SVM classification plo
mpg
45
40
35
30
25
20
15
10
10
15
20
acceleration
1
0


SVM classification plo
mpg
45
40
35
30
25
20
15
10
70
74
78
82
year



En los gráficos anteriores se observa como clasifica svm para las diferentes variables tomando de a dos, con el valor de cost=0.1, gamma=2 y degree=3 que fue uno de los que se detectó con el error más bajo en la validación cruza.

4 punto

```
#librerias
library(FactoMineR)#PCA
library(factoextra)
library(gridExtra)#par
library(kableExtra)#tablas
library(dplyr)
```

[PCA, K-medias] En este ejercicio Ud va a generar un conjunto simulado de datos y entonces aplicará PCA y agrupamiento por k-medias sobre dichos datos.

a)

Genere un conjunto de datos simulados con 20 observaciones en cada una de tres clases (es decir, 60 observaciones en total) y 50 variables. Sugerencia: hay una serie de funciones en R que puede utilizar para generar datos. Un ejemplo es la función rnorm(); runif() es otra opción. Asegúrese de agregar un cambio en la media en las observaciones de cada clase a fin de obtener tres clases distintas.

Primero se fija una semilla que permita la reproductibilidad de los datos obtenidos y luego con ayuda de la función rnorm() y runif() se simula una base de datos con 50 variables, 60 observaciones y 3 clases de población:

```
set.seed(1998)
df<- data.frame(matrix(nrow=60,ncol = 50))
for(i in 1:50){
  a=rnorm(n = 20,52,3)
  b=rnorm(n = 20,72,5)
  c=runif(n = 20,min = 30,max = 55)
  df[,i] <- c(a,b,c)
}
df$clase <- c(rep("1",20),rep("2",20),rep("3",20))
```

	X1	X2	X-	X50	clase
1	46.6180062219627	53.6444371446077	.....	48.7925185604682	1
2	47.2829721816252	54.050374310435	.....	49.2866580161619	1
3	56.6169260416592	56.7227865385154	.....	53.8902883990275	1
4	.....	.....	.....	.....	.....
58	50.716506743338	46.4018209616188	.....	49.7925709595438	3
59	50.5799417803064	53.637874861015	.....	35.8686918288004	3
60	40.6623619620223	51.5669154538773	.....	41.1526024353225	3

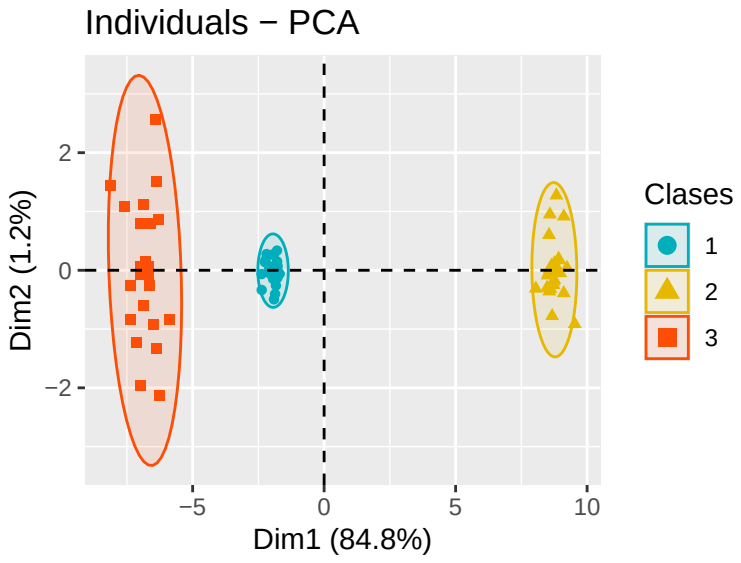
A continuación, se puede apreciar la estuctura general, con las primeras y últimas filas y también columnas:

b)

Realice PCA en las 60 observaciones y grafique las observaciones en términos de las 2 primeras variables principales Z1 y Z2. Use un color diferente para indicar las observaciones en cada una de las tres clases. Si las tres clases aparecen separados en esta gráfica, solo entonces continúe con la parte (c). Si no, vuelva al inciso a) y modifique la simulación para que haya una mayor separación entre las tres clases. No continúe con la parte (c) hasta que las tres clases muestren al menos algún grado de separación en los dos primeros vectores de scores de componentes principales.

A continuación se realiza el ajuste de componentes principales (PCA), el cual intenta reducir la dimensionalidad de la base de datos, y luego se ilustran las observaciones en las primeras 2 componentes principales, la cual logra segmentar correctamente las clases. La primera componente retiene el 84.8% de la varianza total de los datos originales, mientras la segunda el 1.2%

```
#Ajuste
res.pca <- PCA(df[,-51],graph =F)
#Gráfico
fviz_pca_ind(res.pca,
             geom.ind = "point", #mostrar puntos
             col.ind = df$clase, #clases
             palette = c("#00AFBB", "#E7B800", "#FC4E07"),
             addEllipses = TRUE, #elipses
             legend.title = "Clases"
             )+ theme_gray()
```



c)

Desarrolle agrupación de K-medias de las observaciones con K = 3. ¿Qué tan bien funcionan los clústeres que obtuvo con el algoritmo de K-medias comparado con las verdaderas etiquetas de clase?

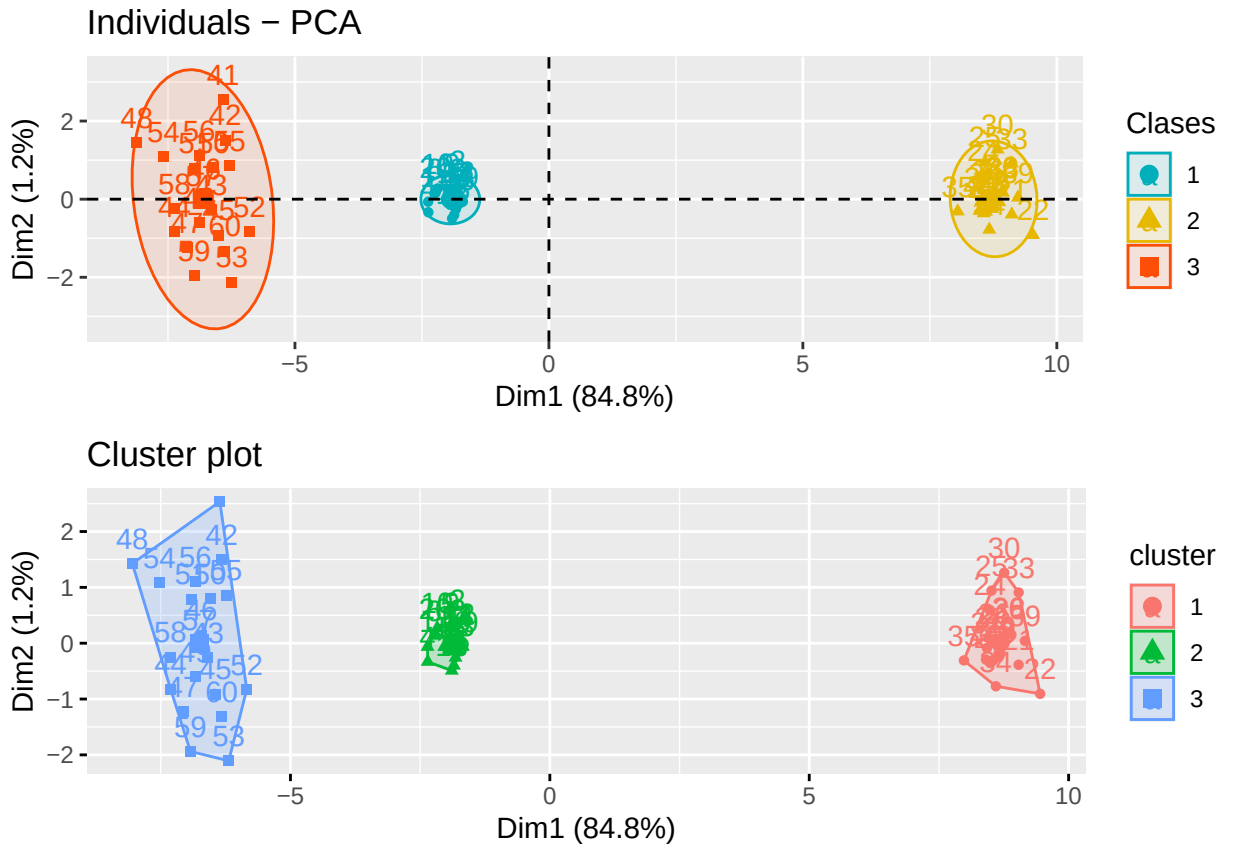
Sugerencia: puede usar la función table() en R para comparar las verdaderas etiquetas de clase con las etiquetas de clase obtenidas por agrupamiento. Tener cuidado cómo se interpretan los resultados: el agrupamiento de K-medias numera los grupos arbitrariamente, por lo que no puede simplemente comprobar si las verdaderas etiquetas de clase y las etiquetas de agrupación son las mismas.

Se realiza a continuación un análisis visual para saber como se están comportando las etiquetas que resultan de la función kmeans():

```
set.seed(123)
km.out =kmeans (df[,-51],3, nstart =20)
```

```
grid.arrange(
            fviz_pca_ind(res.pca,
             geom.ind = c("point","text"), #mostrar puntos
             col.ind = df$clase, #clases
             palette = c("#00AFBB", "#E7B800", "#FC4E07"),
             addEllipses = TRUE, #elipses
             legend.title = "Clases")+ theme_gray(),

            fviz_cluster(km.out, data = df[,-51]))
```



Así entonces, como el agrupamiento de K-medias numera las clases arbitrariamente, se verifican las clases a las que pertenecen los individuos y se puede notar que las etiquetas 1 y 2 están intercaladas. Luego, se realiza el siguiente procedimiento para intercarbiar dichos valores:

```
km2 <- km.out$cluster
for(i in 1:length(km2)){
  if(km2[i]==1){
    km2[i]<-2
  }else if(km2[i]==2){
    km2[i]<- 1
  }else{
    next
  }}; km2
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [39] 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
```

De esta manera, logramos obtener las siguiente matriz de confusión, donde todas las observaciones han sido clasificadas correctamente. Es decir, que la tasa de error fue cero, por lo cual se conlcuye que logoritmo de K-medias comparado con las verdaderas etiquetas funciona correctamente.

```
table(km2,df$clase)
```

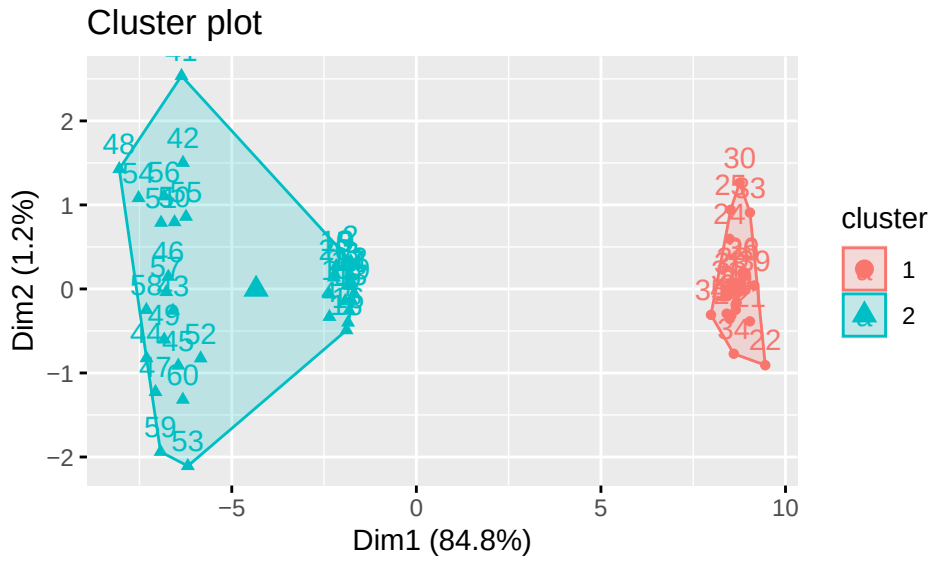
```
##
## km2  1  2  3
##   1 20  0  0
##   2  0 20  0
##   3  0  0 20
```

d)

Realice agrupamiento de K-medias con K = 2. Describa sus resultados.

En el siguiente gráfico podemos observar que realizando el agrupamiento de K-medias con K = 2, el algoritmo clasifica los datos en dos clases, para ello, en este caso unieron las 2 poblaciones más cercanas en una sola (cluster 2, azul), las cuales corresponden a la población original 1 con distribución uniforme y a la población original 3 con distribución normal.

```
set.seed(123) #semilla
km.out =kmeans (df[,-51],2, nstart =20)
fviz_cluster(km.out, data = df[,-51])
```

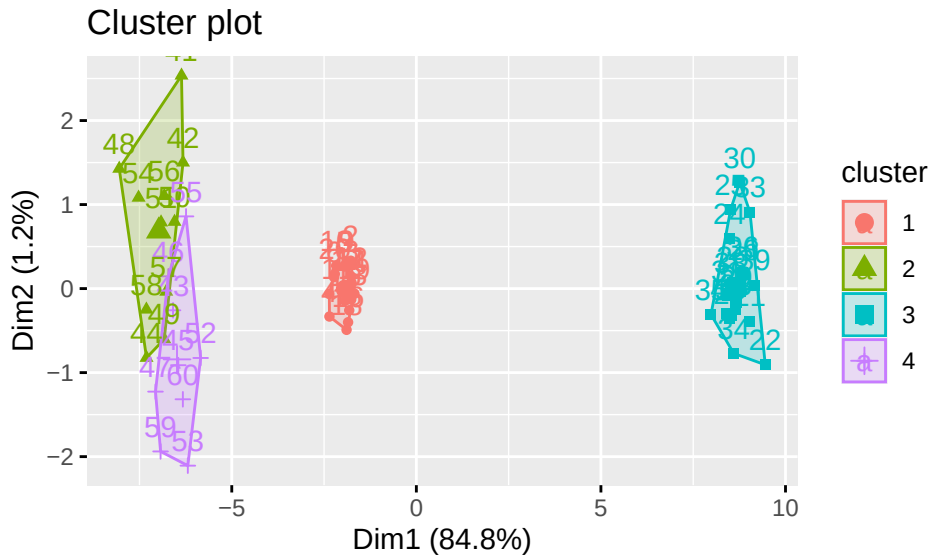


e)

Ahora realice agrupamiento de K-medias con K = 4 y describa sus resultados.

En el siguiente gráfico podemos observar que realizando el agrupamiento de K-medias con K = 4, el algoritmo clasifica los datos en cuatro clases, para ello, en este caso separó en dos la población con mayor dispersión (cluster 2 y 3) que corresponden el las verdaderas etiquetas a las poblaciones 1 y 3 respectivamente.

```
set.seed(123) #semilla
km.out =kmeans(df[,-51],4, nstart =20)
fviz_cluster(km.out, data = df[,-51])
```



f)

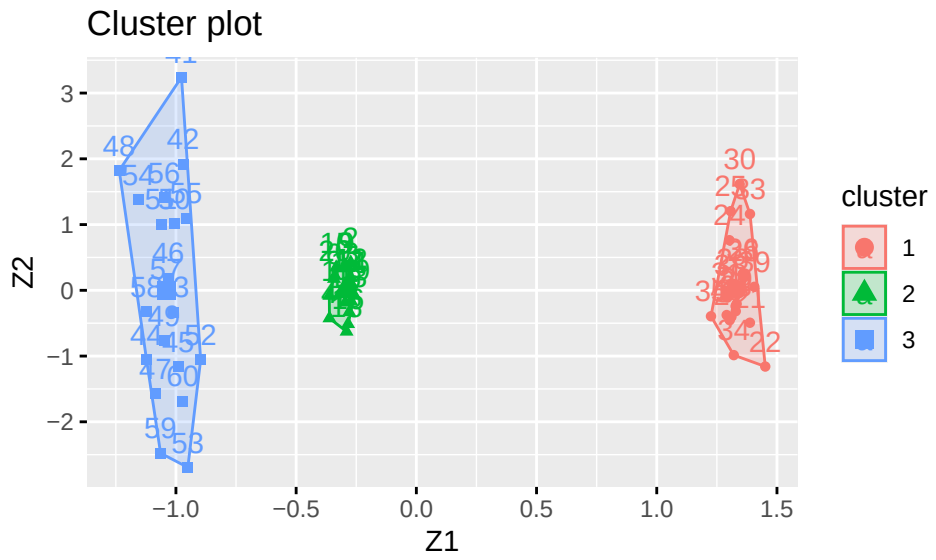
Ahora realice agrupamiento de K-medias con K = 3 en los dos primeros vectores de scores de componentes principales, en lugar de los datos en las variables originales. Es decir, realice la agrupación de K-medias en la matriz de 60 O 2, cuya primera columna es la coordenada zi1 en la primera componente principal Z1 y la segunda columna es la coordenada zi2 en la segunda componente principal Z2. Comente los resultados.

Primero se obtienen las coordenadas de las dos primeras componentes principales, de la siguiente manera:

```
Z1 <- res.pca$ind$coord[,1]
Z2 <- res.pca$ind$coord[,2]
Z<- data.frame(Z1,Z2)
```

Luego, se realiza la agrupación de K-medias con K = 3 con los datos de Z

```
set.seed(123)
km.out =kmeans(Z,3, nstart =20)
fviz_cluster(km.out, data = Z)
```



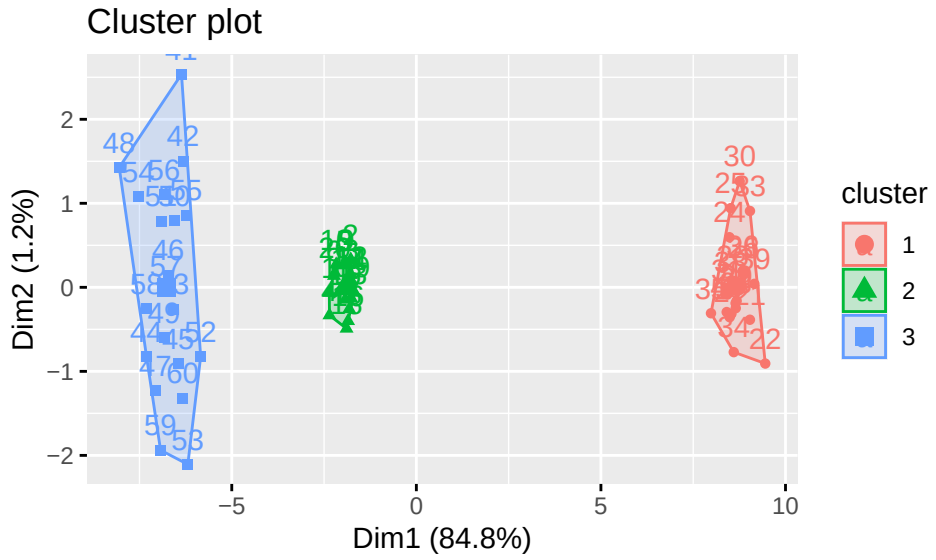
Como se puede observar, realizar agrupación de K medias con las coordenadas obtenidas en las dos primeras componenetes principales se logra también una correcta clasificación de los datos originales. Cabe mencionar que tener conocimiento acerca del número de clases, lo cual favorece los resultados del análisis y que se manifestó un cambio en escala con respercto a la original.

g)

Con la función scale(), realice agrupamiento de K-medias con K = 3 en los datos después de escalar cada variable para tener una desviación estándar de uno. ¿Cómo se comparan estos resultados con los obtenidos? en (b)? Explique.

```
set.seed(123)
sd.data=scale(df[,-51])
km.out =kmeans (sd.data,3, nstart =20)
```

```
fviz_cluster(km.out, data = sd.data)
```



En este caso, el agrupamiento de K-medias de los datos escalados, es decir con desviación estándar de uno, también logra clasificar correctamente las tres diferentes poblaciones. Es decir, con respecto a (b) donde se

hizo uso de análisis de componentes principales, ambos logran clasificar correctamente los individuos de las tres diferentes distribuciones.