Tarea 2

Sebastián Cabezas

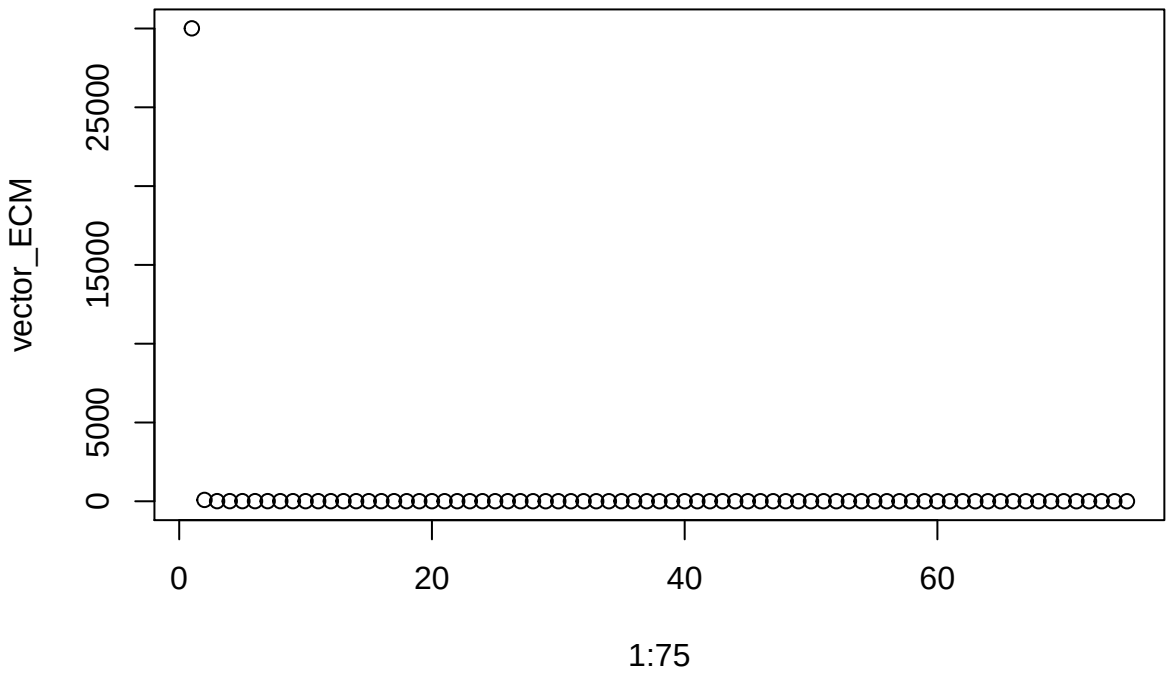
Pregunta 1

1.1

```
matriz_validacion = as.matrix(base_validacion2)

descomposicion = svd(matriz_validacion)

vector_ECM = c()


for (i in 1:75){
  if(i==1) {
    matriz_nueva = descomposicion$u[,1]*descomposicion$d[1]
    regresion = lm(adj_close~. , data = cbind(adj_close,as.data.frame(matriz_nueva)))
    error = sum(regresion$residuals^2)
    vector_ECM = c(vector_ECM, error)}
  else {
    matriz_nueva = descomposicion$u[,1:i]%*%diag(descomposicion$d[1:i])
    regresion = lm(adj_close~. , data = cbind(adj_close,as.data.frame(matriz_nueva)))
    error = sum(regresion$residuals^2)/(dim(matriz_nueva)[1])
    vector_ECM = c(vector_ECM, error)}
  }
```

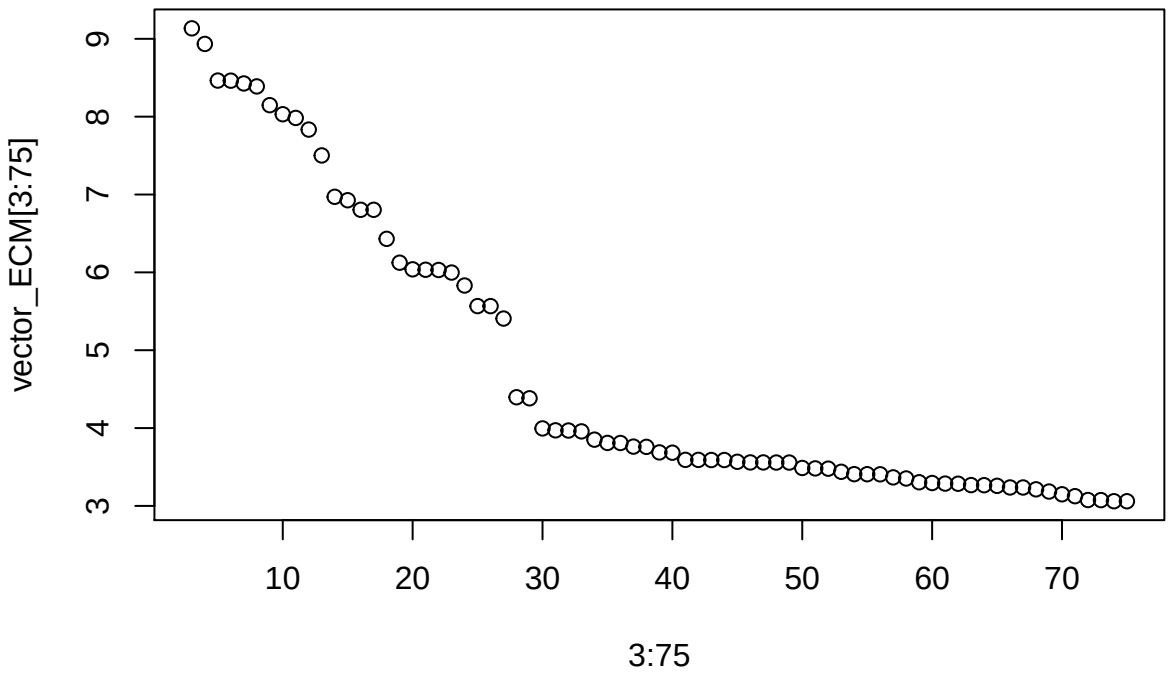
Ploteamos los 75 valores:

```
plot(1:75, vector_ECM)
```



Podemos notar que no se aprecia muy bien los valores donde escoger, por lo que eliminaremos los primeros dos valores para poder apreciar con mayor detalle

```
plot(3:75, vector_ECM[3:75])
```



Aqui elegimos el valor n=30, ya que notamos que el cambio del valor a partir de este valor no es muy notorio.

1.2

Usando r=30:

```
matriz_entrenamiento = as.matrix(base_entrenamiento2)
descomposicion2 = svd(matriz_entrenamiento)
matriz_nueva2 = descomposicion2$u[,1:30]%*%diag(descomposicion2$d[1:30])
regresion_u = lm(adj_close3~. , data = cbind(adj_close3,as.data.frame(matriz_nueva2)))


Xvalr = as.matrix(base_validacion2)%*%descomposicion2$v[,1:30]
ygorro = (Xvalr%*%(regresion_u$coefficients[2:31])) + regresion_u$coefficients[1]

ECM1_2 = mean((ygorro - adj_close)^2)

ECM1_2
```

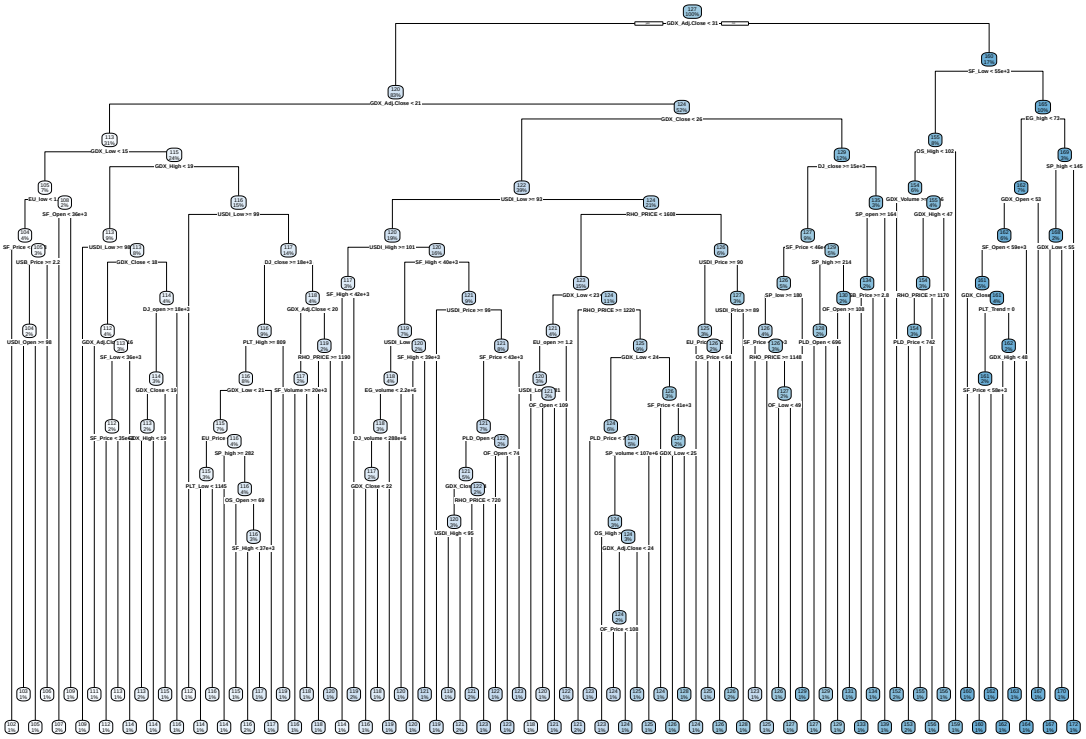
```
## [1] 7.036675
```

Pregunta 2:

2.1

```
arbol_regresion1 <- rpart(
  formula = Adj_close~ .,
  data    = base_entrenamiento,
  method = 'anova',
  cp = 0)


rpart.plot(arbol_regresion1)
```

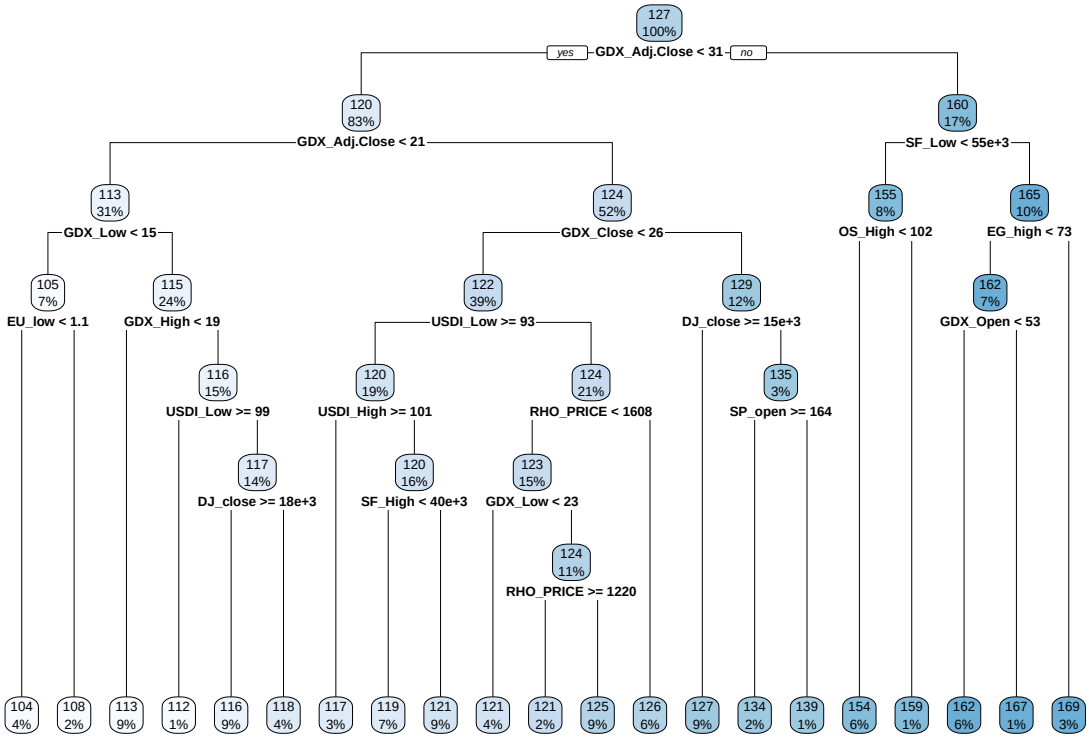
```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



Podemos notar que no se aprecia nada bien, por lo que procedemos a ocupar un valor de complejidad menor para poder apreciar el arbol

```
arbol_regresion2 <- rpart(
  formula = Adj_close~ .,
  data    = base_entrenamiento,
  method = 'anova',
  cp = 0.0005)
```

```
rpart.plot(arbol_regresion2)
```



El arbol original era el primero, pero en este segundo se puede notar de una mejor manera una parte de este.

2.2

```
minimo_valor = min(arbol_regresion1$cptable[,4])
minimo_valor = round(minimo_valor,6)
indice = which(round(as.numeric(arbol_regresion1$cptable[,4]),6) == minimo_valor)

valor_cp = arbol_regresion1$cptable[indice,1]

arbol_podado = rpart(
  formula = Adj_close~ .,
  data    = base_entrenamiento,
  method = 'anova',
  cp = valor_cp)
```

2.3

El grafico se adjunta en otro PDF ya que, en este se logro extraer toda la información desde R. Se adjunta como "grafico1.pdf"

predecimos el valor del primer registro de la base de testeo:

```
prediccion_c = predict(arbol_podado, newdata = base_testeo[1,])
```

Nos arroja un valor de 156.2525, el valor real es 157.16, lo cual es un valor bastante cercano

2.4

```
prediccion_d = predict(arbol_podado, newdata = base_testeo)

ECM2_4 = mean((base_testeo[,1]-prediccion_d)^2) # Valor de 3.682029

ECM2_4
```

```
## [1] 3.688128
```

2.5

```
random_forest  <- ranger(
  formula   = Adj_close ~ .,
  data      = base_entrenamiento,
  num.trees = 100,
  seed      = 123
)


predicciones_e = predict(
  random_forest,
  data = base_testeo
)

predicciones_e = predicciones_e$predictions
MSE_e = mean((predicciones_e - base_testeo$Adj_close)^2)

MSE_e
```

```
## [1] 1.108313
```

2.6

Comparamos los 3 valores:

```
ECM1_2
```

```
## [1] 7.036675
```

```
ECM2_4
```

```
## [1] 3.688128
```

```
MSE_e
```

```
## [1] 1.108313
```

EL valor más pequeño de ECM es el de random forest, por lo que creemos que este es el mejor.