Predicción de la estructura secundaria de proteínas globulares

Cristina Lendinez

8/4/2021

Índice general

Ingreso las librerias que voy a usar

Algoritmo K-NN

El algoritmo K-NN utiliza la informacion de los vecinos mas cercanos para etiquetar o clasificar individuos que no tengan etiqueta o clasificación. Para ello utilizaremos un conjunto de datos para "entrenamiento" y clasificara otro conjunto de individuos que no esten clasificados. El algoritmo identificara k individuos del conjunto de datos de entrenamiento que tengan mas similitud con cada individuo a etiquetar y se asignara su clasificación. El valor k es un valor especificado de antemano.

Fortalezas	**Debilidades**
*Simple y efectivo	*No produce un modelo, lo que limita la capacidad de encontrar conocimientos en las relaciones entre las caracteristicas
*No hace suposiciones sobre la distribución de datos subyacentes	*Fase de clasificación lenta
*Fase de entrenamiento rápida	*Requiere una gran cantidad de memoria * Caracteristicas nominales y la falta de datos requiere un procesamiento adicional

Step 1: Recolección de los datos

2.Desarrollar una función en R que implemente una codificación "one-hot" (one-hot encoding) de las secuencias.

Se descargan los datos de la PEC desde la pagina de la UOC o del dataset "" . Tenemos 2 archivos que estan en formato csv y tiene como separador ",". Son 2 csv, uno de ellos lo usaremos para crear la funcion **one-hot**, y si no conseguiimos crear la funcion **one-hot**, tenemos el archivo **oh_enc** cedido por el profesor, para que podamos continuar con el análisis. Vamos a importar el primer csv, llamado ***data4*** Cargo el primer csv llamado data4.csv

```
datos <- read.csv("C:\\Users\\crist\\OneDrive\\Desktop\\machine learning\\Pec 1 Cris\\data4 .csv", head
```

Tengo que eliminar la columna V18, para poder crear mi variable de **one-hot**

```
datos2 <- datos[-18]
```

Confirmo que he eliminado la columna V18

```
str(datos2)
```

```
## 'data.frame':    10000 obs. of  17 variables:
##  $ V1 : Factor w/ 20 levels "A","C","D","E",..: 16 10 10 10 17 6 6 10 16 16 ...
##  $ V2 : Factor w/ 20 levels "A","C","D","E",..: 16 10 10 10 18 14 7 11 1 6 ...
##  $ V3 : Factor w/ 20 levels "A","C","D","E",..: 13 14 12 12 16 20 1 18 3 3 ...
##  $ V4 : Factor w/ 20 levels "A","C","D","E",..: 5 20 13 1 5 1 5 3 3 16 ...
##  $ V5 : Factor w/ 20 levels "A","C","D","E",..: 16 20 9 9 4 18 12 12 18 20 ...
##  $ V6 : Factor w/ 20 levels "A","C","D","E",..: 14 6 9 6 1 7 18 19 9 17 ...
##  $ V7 : Factor w/ 20 levels "A","C","D","E",..: 9 11 20 4 17 6 4 15 9 10 ...
##  $ V8 : Factor w/ 20 levels "A","C","D","E",..: 6 17 8 17 5 18 5 13 1 16 ...
```

```
##  $ V9 : Factor w/ 20 levels "A","C","D","E",..: 3 4 13 5 1 17 3 1 5 10 ...
##  $ V10: Factor w/ 20 levels "A","C","D","E",..: 4 11 6 4 5 16 3 14 1 16 ...
##  $ V11: Factor w/ 20 levels "A","C","D","E",..: 15 12 17 18 10 5 16 13 8 6 ...
##  $ V12: Factor w/ 20 levels "A","C","D","E",..: 20 20 9 1 8 18 4 10 8 1 ...
##  $ V13: Factor w/ 20 levels "A","C","D","E",..: 9 20 11 10 9 16 3 9 3 16 ...
##  $ V14: Factor w/ 20 levels "A","C","D","E",..: 20 17 18 16 16 15 9 12 14 20 ...
##  $ V15: Factor w/ 20 levels "A","C","D","E",..: 18 18 5 12 13 10 1 15 3 17 ...
##  $ V16: Factor w/ 20 levels "A","C","D","E",..: 3 10 13 9 3 6 18 14 9 1 ...
##  $ V17: Factor w/ 20 levels "A","C","D","E",..: 13 5 6 6 16 2 10 8 16 6 ...
```

```
#Con la uncion str, acabo de confirmar que he eliminado la columna V18,
```

Seguidamente con la funcion dummy podre crear mi función **one-hot**, y tambien con la funcion one-hot, lo hago de las 2 formas para confirmar que funciona correctamente

```
data.dummy <- dummy.data.frame(datos2, sep=".")
head(data.dummy)
```

```
##   V1.A V1.C V1.D V1.E V1.F V1.G V1.H V1.I V1.K V1.L V1.M V1.N V1.P V1.Q V1.R
## 1    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 4    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
##   V1.S V1.T V1.V V1.W V1.Y V2.A V2.C V2.D V2.E V2.F V2.G V2.H V2.I V2.K V2.L
## 1    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 5    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##   V2.M V2.N V2.P V2.Q V2.R V2.S V2.T V2.V V2.W V2.Y V3.A V3.C V3.D V3.E V3.F
## 1    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
## 6    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
##   V3.G V3.H V3.I V3.K V3.L V3.M V3.N V3.P V3.Q V3.R V3.S V3.T V3.V V3.W V3.Y
## 1    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 3    0    0    0    0    0    0    1    0    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    1    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 6    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
##   V4.A V4.C V4.D V4.E V4.F V4.G V4.H V4.I V4.K V4.L V4.M V4.N V4.P V4.Q V4.R
## 1    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0
## 4    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
## 6    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

```
##   V4.S V4.T V4.V V4.W V4.Y V5.A V5.C V5.D V5.E V5.F V5.G V5.H V5.I V5.K V5.L
## 1    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    1    0
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    1    0
## 5    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 6    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##   V5.M V5.N V5.P V5.Q V5.R V5.S V5.T V5.V V5.W V5.Y V6.A V6.C V6.D V6.E V6.F
## 1    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 6    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
##   V6.G V6.H V6.I V6.K V6.L V6.M V6.N V6.P V6.Q V6.R V6.S V6.T V6.V V6.W V6.Y
## 1    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 2    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 4    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0
##   V7.A V7.C V7.D V7.E V7.F V7.G V7.H V7.I V7.K V7.L V7.M V7.N V7.P V7.Q V7.R
## 1    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
##   V7.S V7.T V7.V V7.W V7.Y V8.A V8.C V8.D V8.E V8.F V8.G V8.H V8.I V8.K V8.L
## 1    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    1    0    0    0    0    0    0    0    1    0    0
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    1    0    0    0    0    0    0    0    1    0    0    0    0    0
## 6    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##   V8.M V8.N V8.P V8.Q V8.R V8.S V8.T V8.V V8.W V8.Y V9.A V9.C V9.D V9.E V9.F
## 1    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0
## 2    0    0    0    0    0    0    1    0    0    0    0    0    0    1    0
## 3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    1    0    0    0    0    0    0    0    1
## 5    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 6    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
##   V9.G V9.H V9.I V9.K V9.L V9.M V9.N V9.P V9.Q V9.R V9.S V9.T V9.V V9.W V9.Y
## 1    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
## 4    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0
##   V10.A V10.C V10.D V10.E V10.F V10.G V10.H V10.I V10.K V10.L V10.M V10.N V10.P
## 1     0     0     0     1     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     1     0     0
## 3     0     0     0     0     0     1     0     0     0     0     0     0     0
## 4     0     0     0     1     0     0     0     0     0     0     0     0     0
```

```
## 5     0     0     0     0     1     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     0     0
##   V10.Q V10.R V10.S V10.T V10.V V10.W V10.Y V11.A V11.C V11.D V11.E V11.F V11.G
## 1     0     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0     0
## 6     0     0     1     0     0     0     0     0     0     0     0     1     0
##   V11.H V11.I V11.K V11.L V11.M V11.N V11.P V11.Q V11.R V11.S V11.T V11.V V11.W
## 1     0     0     0     0     0     0     0     0     1     0     0     0     0
## 2     0     0     0     0     0     1     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     0     1     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     1     0
## 5     0     0     0     1     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     0     0
##   V11.Y V12.A V12.C V12.D V12.E V12.F V12.G V12.H V12.I V12.K V12.L V12.M V12.N
## 1     0     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     1     0     0     0
## 4     0     1     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     1     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     0     0
##   V12.P V12.Q V12.R V12.S V12.T V12.V V12.W V12.Y V13.A V13.C V13.D V13.E V13.F
## 1     0     0     0     0     0     0     0     1     0     0     0     0     0
## 2     0     0     0     0     0     0     0     1     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     1     0     0     0     0     0     0     0
##   V13.G V13.H V13.I V13.K V13.L V13.M V13.N V13.P V13.Q V13.R V13.S V13.T V13.V
## 1     0     0     0     1     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     1     0     0     0     0     0     0     0
## 4     0     0     0     0     1     0     0     0     0     0     0     0     0
## 5     0     0     0     1     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     1     0     0
##   V13.W V13.Y V14.A V14.C V14.D V14.E V14.F V14.G V14.H V14.I V14.K V14.L V14.M
## 1     0     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     1     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     0     0
##   V14.N V14.P V14.Q V14.R V14.S V14.T V14.V V14.W V14.Y V15.A V15.C V15.D V15.E
## 1     0     0     0     0     0     0     0     0     1     0     0     0     0
## 2     0     0     0     0     0     1     0     0     0     0     0     0     0
## 3     0     0     0     0     0     0     1     0     0     0     0     0     0
## 4     0     0     0     0     1     0     0     0     0     0     0     0     0
## 5     0     0     0     0     1     0     0     0     0     0     0     0     0
## 6     0     0     0     1     0     0     0     0     0     0     0     0     0
##   V15.F V15.G V15.H V15.I V15.K V15.L V15.M V15.N V15.P V15.Q V15.R V15.S V15.T
## 1     0     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0     0
```

```
## 3     1     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     1     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     1     0     0     0     0
## 6     0     0     0     0     0     1     0     0     0     0     0     0     0
##   V15.V V15.W V15.Y V16.A V16.C V16.D V16.E V16.F V16.G V16.H V16.I V16.K V16.L
## 1     1     0     0     0     0     1     0     0     0     0     0     0     0
## 2     1     0     0     0     0     0     0     0     0     0     0     0     1
## 3     0     0     0     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     1     0
## 5     0     0     0     0     0     1     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     1     0     0     0     0
##   V16.M V16.N V16.P V16.Q V16.R V16.S V16.T V16.V V16.W V16.Y V17.A V17.C V17.D
## 1     0     0     0     0     0     0     0     0     0     0     0     0     0
## 2     0     0     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     1     0     0     0     0     0     0     0     0     0     0
## 4     0     0     0     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0     0
## 6     0     0     0     0     0     0     0     0     0     0     0     1     0
##   V17.E V17.F V17.G V17.H V17.I V17.K V17.L V17.M V17.N V17.P V17.Q V17.R V17.S
## 1     0     0     0     0     0     0     0     0     0     1     0     0     0
## 2     0     1     0     0     0     0     0     0     0     0     0     0     0
## 3     0     0     1     0     0     0     0     0     0     0     0     0     0
## 4     0     0     1     0     0     0     0     0     0     0     0     0     0
## 5     0     0     0     0     0     0     0     0     0     0     0     0     1
## 6     0     0     0     0     0     0     0     0     0     0     0     0     0
##   V17.T V17.V V17.W V17.Y
## 1     0     0     0     0
## 2     0     0     0     0
## 3     0     0     0     0
## 4     0     0     0     0
## 5     0     0     0     0
## 6     0     0     0     0
```

```
datos$V18<-factor(datos$V18,levels=c("h","e","_"),
labels=c("a_helix","b_sheet","coil"))
```

Ahora pruebo con la funcion ***one__hote***

```
One_hot_data = one_hot(as.data.table(datos2))
##Puedo observar que las 2 funciones funcionan perfectamente, como hay varias formas de hacer el ejerci
```

3. Desarrollar un script en R que implemente un clasificador knn. El script debe realizar los siguientes apartados:

(a) Leer el fichero data4.csv. Cada registro contiene una secuencia de 17 aminoácidos y la clase de estructura secundaria correspondiente al aminoácido central (posición 9), donde los caracteres 'h', 'e' y '_' representan -helix, -sheet y coil, respectivamente. Después de cargar los datos, crear una tabla donde se muestre el número de secuencias de cada clase.

Vuelvo a importar el csv data4.csv, ya que al crear la funcion con dummy y one hot he podido comprobar que funciona y las variables se transofrman en 1 y 0 según donde cae el aminoacido

```
datos <- read.csv("C:\\Users\\crist\\OneDrive\\Desktop\\machine learning\\Pec 1 Cris\\data4 .csv", head
```

Haggo la tabla donde los caracterea sean -helis, -sheet, -coil, lo hare de dos formasm una centrandome en la posicion 9 y otra sin centrarme en ellos

```
datos$V18<-factor(datos$V18,levels=c("h","e","_"),
labels=c("a_helix","b_sheet","coil"))
table(factor(datos$V18), factor(datos$V9))
```

```
##
##              A   C   D   E   F   G   H   I   K   L   M   N   P   Q   R   S   T
##   a_helix 296  33 151 160 145 100  49 119 255 265  40 105  59  88 108 127  89
##   b_sheet 116  47  52  46  98 111  26 160  86 218  39  43  49  67  55 144 167
##   coil    456 111 405 256 154 747 127 222 337 376  67 309 355 172 151 494 286
##
##              V   W   Y
##   a_helix 190  34  95
##   b_sheet 276  51  84
##   coil    301  66 165
```

```
##Como podemos ver he creado una tabla en la que me encuentro los 17 aminoacidos  en su estructura secu
##hare otra tabla diferente tambien
```

Ahora hago la tabla sin centrarme en la columna V9

```
table(datos$V18)
```

```
##
## a_helix b_sheet    coil
##    2508    1935    5557
```

Lo que puedo observar, es que tenemos 5557 que son coil, 1935 que son sheet, y 2508 que son helix.

##(b) Utilizar la función de codificación "one-hot" para representar las secuencias NOTA: En casode no poder hacer la función, se puede descargar el fichero oh_enc.csv con las secuencias ya transformados

```
##quito la columna V18 para poder codificar lo datos
datos2 <- datos[-18]
data.dummy <- dummy.data.frame(datos2, sep=".")
One_hot_data = one_hot(as.data.table(datos2))
##Al tener los datos codificados, ya puedo ponerme con el apartado 3, donde hare el training y el test
```

Creo la funcion One_hote_data, en todo momento estoy siguiendo el libro

```
One_hot_data_label<-cbind(One_hot_data,datos$V18)
data.dummy.label <- cbind(data.dummy,datos$V18)
```

(c) Utilizando la semilla aleatoria 123, separar los datos en dos partes, una parte para training (67%) y una parte para test (33%).

Voy a eparar los datos de 2 formas con dummy y con one-hot, para seguir asegurandome que todo funciona correctamente.

Primero lo hago con la funcion One_hote_data_label

```
library(purrr)
library(caret)
```

```
##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(123)
ind <- sample(1:nrow(One_hot_data_label),round(2*nrow(One_hot_data_label)/3))
training_one_hot <-One_hot_data_label[ind]
test_one_hot <- One_hot_data_label[-ind,]
```

Compruebo en que he creado el training y el testing

```
dim(training_one_hot)
```

```
## [1] 6667  341
```

```
dim(test_one_hot)
```

```
## [1] 3333  341
```

Ahora hago lo mismo con la función dummy, ya que me gusta ver que podemos resolver ls PEC de varias formas

```
library(caret)
set.seed(123)
ind <- sample(1:nrow(data.dummy.label),round(2*nrow(data.dummy.label)/3))
training_dummy <-data.dummy.label[ind,]
test_dummy <- data.dummy[-ind,]
```

Compruebo con la funcion dim que se dividio mi dummy en 67% de training y 33 % de test

```
dim(training_dummy)
```

```
## [1] 6667  341
```

```
dim(test_dummy)
```

```
## [1] 3333  340
```

```
##Como puedo ver al ver la longitud de training y test con la funcion dummy veo que la funcion test tie
```

(d) Utilizar un knn (k = 1, 3, 5, 7, 11) basado en el training para predecir la estructura secundaria de las secuencias del test.

Mientras estuve haciendo la Pec me encontre problemas con el vector que contienen las etiquetas,lo que tengo que hacer es pasarlo a factor, seguire haciendolo todo tanto en onehot como en dummy

```
head(training_one_hot$V2)
```

```
## [1] coil    a_helix coil    a_helix coil    coil
## Levels: a_helix b_sheet coil
```

Convierto a integer el vector de las etiquetas de one-hot

```
training_one_hot$V2<-as.integer(training_one_hot$V2)
head(training_one_hot$V2)
```

```
## [1] 3 1 3 1 3 3
```

miro los 5 primeros valores de test_one_hot

```
head(test_one_hot)
```

```
##    V1_A V1_C V1_D V1_E V1_F V1_G V1_H V1_I V1_K V1_L V1_M V1_N V1_P V1_Q V1_R
## 1:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 2:    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 3:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 5:    0    0    1    0    0    0    0    0    0    0    0    0    0    0    0
## 6:    0    0    1    0    0    0    0    0    0    0    0    0    0    0    0
##    V1_S V1_T V1_V V1_W V1_Y V2_A V2_C V2_D V2_E V2_F V2_G V2_H V2_I V2_K V2_L
## 1:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 2:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0
## 6:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##    V2_M V2_N V2_P V2_Q V2_R V2_S V2_T V2_V V2_W V2_Y V3_A V3_C V3_D V3_E V3_F
## 1:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

```
## 2:    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6:    0    0    0    0    0    0    1    0    0    0    0    0    0    0    0
##    V3_G V3_H V3_I V3_K V3_L V3_M V3_N V3_P V3_Q V3_R V3_S V3_T V3_V V3_W V3_Y
## 1:    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 2:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 3:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0
## 6:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##    V4_A V4_C V4_D V4_E V4_F V4_G V4_H V4_I V4_K V4_L V4_M V4_N V4_P V4_Q V4_R
## 1:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    1
## 4:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
##    V4_S V4_T V4_V V4_W V4_Y V5_A V5_C V5_D V5_E V5_F V5_G V5_H V5_I V5_K V5_L
## 1:    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
## 2:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 5:    1    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 6:    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0
##    V5_M V5_N V5_P V5_Q V5_R V5_S V5_T V5_V V5_W V5_Y V6_A V6_C V6_D V6_E V6_F
## 1:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 2:    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 6:    0    0    0    0    0    0    1    0    0    0    0    0    0    0    0
##    V6_G V6_H V6_I V6_K V6_L V6_M V6_N V6_P V6_Q V6_R V6_S V6_T V6_V V6_W V6_Y
## 1:    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2:    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6:    0    0    0    0    0    0    0    0    0    0    0    0    1    0    0
##    V7_A V7_C V7_D V7_E V7_F V7_G V7_H V7_I V7_K V7_L V7_M V7_N V7_P V7_Q V7_R
## 1:    0    0    0    0    0    0    0    0    0    0    1    0    0    0    0
## 2:    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0
## 6:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
##    V7_S V7_T V7_V V7_W V7_Y V8_A V8_C V8_D V8_E V8_F V8_G V8_H V8_I V8_K V8_L
## 1:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    1    0    0    0    0    0    0    0    0    0
## 6:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
```

```
##    V8_M V8_N V8_P V8_Q V8_R V8_S V8_T V8_V V8_W V8_Y V9_A V9_C V9_D V9_E V9_F
## 1:    0    0    0    0    0    0    1    0    0    0    0    0    0    1    0
## 2:    0    0    0    0    0    0    0    1    0    0    0    0    0    0    0
## 3:    0    0    0    0    0    0    1    0    0    0    0    0    0    0    0
## 4:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 6:    0    0    0    0    0    0    0    0    0    1    0    0    0    0    0
##    V9_G V9_H V9_I V9_K V9_L V9_M V9_N V9_P V9_Q V9_R V9_S V9_T V9_V V9_W V9_Y
## 1:    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
## 2:    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0
## 3:    0    0    0    0    0    0    0    0    1    0    0    0    0    0    0
## 4:    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0
## 5:    0    0    0    0    0    0    0    0    0    0    0    1    0    0    0
## 6:    0    0    0    0    1    0    0    0    0    0    0    0    0    0    0
##    V10_A V10_C V10_D V10_E V10_F V10_G V10_H V10_I V10_K V10_L V10_M V10_N
## 1:     0     0     0     0     0     0     0     0     0     0     1     0
## 2:     0     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     1     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V10_P V10_Q V10_R V10_S V10_T V10_V V10_W V10_Y V11_A V11_C V11_D V11_E
## 1:     0     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     1     0     0     0     0     0     0     0     0
## 3:     1     0     0     0     0     0     0     0     0     0     0     1
## 4:     0     0     0     0     0     0     0     0     0     0     0     0
## 5:     0     0     0     0     0     1     0     0     0     0     0     0
## 6:     1     0     0     0     0     0     0     0     0     0     0     0
##    V11_F V11_G V11_H V11_I V11_K V11_L V11_M V11_N V11_P V11_Q V11_R V11_S
## 1:     0     0     0     0     0     0     0     1     0     0     0     0
## 2:     1     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     1     0     0     0     0     0     0     0
## 5:     0     1     0     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V11_T V11_V V11_W V11_Y V12_A V12_C V12_D V12_E V12_F V12_G V12_H V12_I
## 1:     0     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     0     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     0     0
## 6:     1     0     0     0     0     0     0     0     0     0     0     0
##    V12_K V12_L V12_M V12_N V12_P V12_Q V12_R V12_S V12_T V12_V V12_W V12_Y
## 1:     0     0     0     0     0     0     0     0     0     0     0     1
## 2:     0     0     0     0     0     0     0     0     0     1     0     0
## 3:     0     0     0     0     1     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     0     1     0     0
## 5:     0     0     1     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     1     0     0     0     0
##    V13_A V13_C V13_D V13_E V13_F V13_G V13_H V13_I V13_K V13_L V13_M V13_N
## 1:     0     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     1     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     1     0     0     0
```

```
## 5:     1     0     0     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V13_P V13_Q V13_R V13_S V13_T V13_V V13_W V13_Y V14_A V14_C V14_D V14_E
## 1:     0     0     0     0     0     0     0     1     0     0     0     0
## 2:     0     0     0     1     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     1     0     0     0
## 4:     0     0     0     0     0     0     0     0     1     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     1     0     0     0     0     0     0
##    V14_F V14_G V14_H V14_I V14_K V14_L V14_M V14_N V14_P V14_Q V14_R V14_S
## 1:     0     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     0     1     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     0     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V14_T V14_V V14_W V14_Y V15_A V15_C V15_D V15_E V15_F V15_G V15_H V15_I
## 1:     1     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     1     0     0     0     0     0     0     0
## 5:     0     1     0     0     0     0     0     0     0     0     0     0
## 6:     0     1     0     0     0     0     0     0     0     0     0     0
##    V15_K V15_L V15_M V15_N V15_P V15_Q V15_R V15_S V15_T V15_V V15_W V15_Y
## 1:     0     0     0     0     0     0     0     0     0     1     0     0
## 2:     0     1     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     0     0     0     0     1     0     0
## 4:     0     0     0     0     0     0     0     0     0     0     0     0
## 5:     0     0     0     0     0     0     0     0     1     0     0     0
## 6:     0     0     0     0     0     0     0     1     0     0     0     0
##    V16_A V16_C V16_D V16_E V16_F V16_G V16_H V16_I V16_K V16_L V16_M V16_N
## 1:     0     0     0     0     0     0     0     0     0     1     0     0
## 2:     0     0     0     0     0     1     0     0     0     0     0     0
## 3:     0     0     0     0     0     1     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     1     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     0     0
## 6:     1     0     0     0     0     0     0     0     0     0     0     0
##    V16_P V16_Q V16_R V16_S V16_T V16_V V16_W V16_Y V17_A V17_C V17_D V17_E
## 1:     0     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     1     0     0
## 3:     0     0     0     0     0     0     0     0     0     0     0     0
## 4:     0     0     0     0     0     0     0     0     0     0     0     0
## 5:     0     0     1     0     0     0     0     0     0     0     0     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V17_F V17_G V17_H V17_I V17_K V17_L V17_M V17_N V17_P V17_Q V17_R V17_S
## 1:     1     0     0     0     0     0     0     0     0     0     0     0
## 2:     0     0     0     0     0     0     0     0     0     0     0     0
## 3:     0     0     0     0     0     1     0     0     0     0     0     0
## 4:     0     0     0     0     1     0     0     0     0     0     0     0
## 5:     0     0     0     0     0     0     0     0     0     0     1     0
## 6:     0     0     0     0     0     0     0     0     0     0     0     0
##    V17_T V17_V V17_W V17_Y      V2
## 1:     0     0     0     0    coil
## 2:     0     0     0     0 b_sheet
```

```
## 3:     0     0     0     0    coil
## 4:     0     0     0     0 a_helix
## 5:     0     0     0     0    coil
## 6:     0     0     0     1 b_sheet
```

Paso el vector a integer, para no tener problemas al realizar los knn vecinos cercanos

```
test_one_hot$V2<-as.integer(test_one_hot$V2)
head(test_one_hot$V2)
```

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

```
table(training_one_hot$V2)
```

```
##
##    1    2    3
## 1668 1282 3717
```

```
table(test_one_hot$V2)
```

```
##
##    1    2    3
##  840  653 1840
```

(d) Utilizar un knn (k = 1, 3, 5, 7, 11) basado en el training para predecir la estructura secundaria de las secuencias del test.

Vamos a hacer los KNN los vecinos cercanos con los datos que hemos separado en el apartado anterior

K1

```
library(class)
library(gmodels)
predicion1<-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$V2, k=1)
k1<-CrossTable(x=test_one_hot$V2, y=predicion1,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
```

```
##
##
##                  | predicion1
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       766 |        60 |        14 |       840 |
##                 |     0.912 |     0.071 |     0.017 |     0.252 |
##                 |     0.954 |     0.088 |     0.008 |           |
##                 |     0.230 |     0.018 |     0.004 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        32 |       519 |       102 |       653 |
##                 |     0.049 |     0.795 |     0.156 |     0.196 |
##                 |     0.040 |     0.759 |     0.055 |           |
##                 |     0.010 |     0.156 |     0.031 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         5 |       105 |      1730 |      1840 |
##                 |     0.003 |     0.057 |     0.940 |     0.552 |
##                 |     0.006 |     0.154 |     0.937 |           |
##                 |     0.002 |     0.032 |     0.519 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       803 |       684 |      1846 |      3333 |
##                 |     0.241 |     0.205 |     0.554 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

Ahora hago el ***k3***

```
predicion3<-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$V2, k=3)
k1<-CrossTable(x=test_one_hot$V2, y=predicion3,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion3
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       769 |        63 |         8 |       840 |
##                 |     0.915 |     0.075 |     0.010 |     0.252 |
##                 |     0.942 |     0.105 |     0.004 |           |
```

```
##                 |     0.231 |     0.019 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        45 |       442 |       166 |       653 |
##                 |     0.069 |     0.677 |     0.254 |     0.196 |
##                 |     0.055 |     0.734 |     0.087 |           |
##                 |     0.014 |     0.133 |     0.050 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         2 |        97 |      1741 |      1840 |
##                 |     0.001 |     0.053 |     0.946 |     0.552 |
##                 |     0.002 |     0.161 |     0.909 |           |
##                 |     0.001 |     0.029 |     0.522 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       816 |       602 |      1915 |      3333 |
##                 |     0.245 |     0.181 |     0.575 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

k5

```
predicion5<-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$V2, k=5)
k1<-CrossTable(x=test_one_hot$V2, y=predicion5,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                 | predicion5
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       758 |        76 |         6 |       840 |
##                 |     0.902 |     0.090 |     0.007 |     0.252 |
##                 |     0.940 |     0.136 |     0.003 |           |
##                 |     0.227 |     0.023 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        47 |       391 |       215 |       653 |
##                 |     0.072 |     0.599 |     0.329 |     0.196 |
##                 |     0.058 |     0.701 |     0.109 |           |
##                 |     0.014 |     0.117 |     0.065 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         1 |        91 |      1748 |      1840 |
```

```
##                 |     0.001 |     0.049 |     0.950 |     0.552 |
##                 |     0.001 |     0.163 |     0.888 |           |
##                 |     0.000 |     0.027 |     0.524 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       806 |       558 |      1969 |      3333 |
##                 |     0.242 |     0.167 |     0.591 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

k7

```
predicion7<-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$V2, k=7)
k1<-CrossTable(x=test_one_hot$V2, y=predicion7,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                 | predicion7
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       745 |        90 |         5 |       840 |
##                 |     0.887 |     0.107 |     0.006 |     0.252 |
##                 |     0.936 |     0.174 |     0.002 |           |
##                 |     0.224 |     0.027 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        50 |       348 |       255 |       653 |
##                 |     0.077 |     0.533 |     0.391 |     0.196 |
##                 |     0.063 |     0.672 |     0.126 |           |
##                 |     0.015 |     0.104 |     0.077 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         1 |        80 |      1759 |      1840 |
##                 |     0.001 |     0.043 |     0.956 |     0.552 |
##                 |     0.001 |     0.154 |     0.871 |           |
##                 |     0.000 |     0.024 |     0.528 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       796 |       518 |      2019 |      3333 |
##                 |     0.239 |     0.155 |     0.606 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

k11

```
predicion11<-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$V2, k=11)
k1<-CrossTable(x=test_one_hot$V2, y=predicion11,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion11
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       768 |        71 |         1 |       840 |
##                 |     0.914 |     0.085 |     0.001 |     0.252 |
##                 |     0.941 |     0.171 |     0.000 |           |
##                 |     0.230 |     0.021 |     0.000 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        48 |       300 |       305 |       653 |
##                 |     0.074 |     0.459 |     0.467 |     0.196 |
##                 |     0.059 |     0.721 |     0.145 |           |
##                 |     0.014 |     0.090 |     0.092 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         0 |        45 |      1795 |      1840 |
##                 |     0.000 |     0.024 |     0.976 |     0.552 |
##                 |     0.000 |     0.108 |     0.854 |           |
##                 |     0.000 |     0.014 |     0.539 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       816 |       416 |      2101 |      3333 |
##                 |     0.245 |     0.125 |     0.630 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

##(e) Por otra parte, sabemos que las clases -helix y -sheet son del tipo non-coil. Realizar otro knn (k = 1, 3, 5, 7, 11) para esta nueva clasificación, coil y non-coil. Además, realizar una curva ROC para cada k y calcular su área bajo la curva (AUC).

**Haremos el mismo proceso que en apartado anterior, pero diferenciaremos entre las que no son coil

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

```
training_one_hot$coil <- recode(training_one_hot$V2,"1:2=0; 3=1")
test_one_hot$coil <- recode(test_one_hot$V2, "1:2=0; 3=1")
table(training_one_hot$coil)
```

```
##
##    0    1
## 2950 3717
```

```
table(test_one_hot$coil)
```

```
##
##    0    1
## 1493 1840
```

Vamos a realizar el knn para (1,3,5,7,11), pero lo haremos en los datos coil, el apartado será muy parecido al anterior, pero usaremos lo s datos coil

```
library(class)
library(gmodels)
predicion_coil_1 <-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$coil, k=1)
predicion_coil_3 <-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$coil, k=3)
predicion_coil_5 <-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$coil, k=5)
predicion_coil_7 <-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$coil, k=7)
predicion_coil_11 <-knn(train=training_one_hot, test=test_one_hot,
cl=training_one_hot$coil, k=11)
```

Tenemos que crear la curva ROC, y elAUC, para las 5 k que hemos realizado (k=1,3,5,7,11)

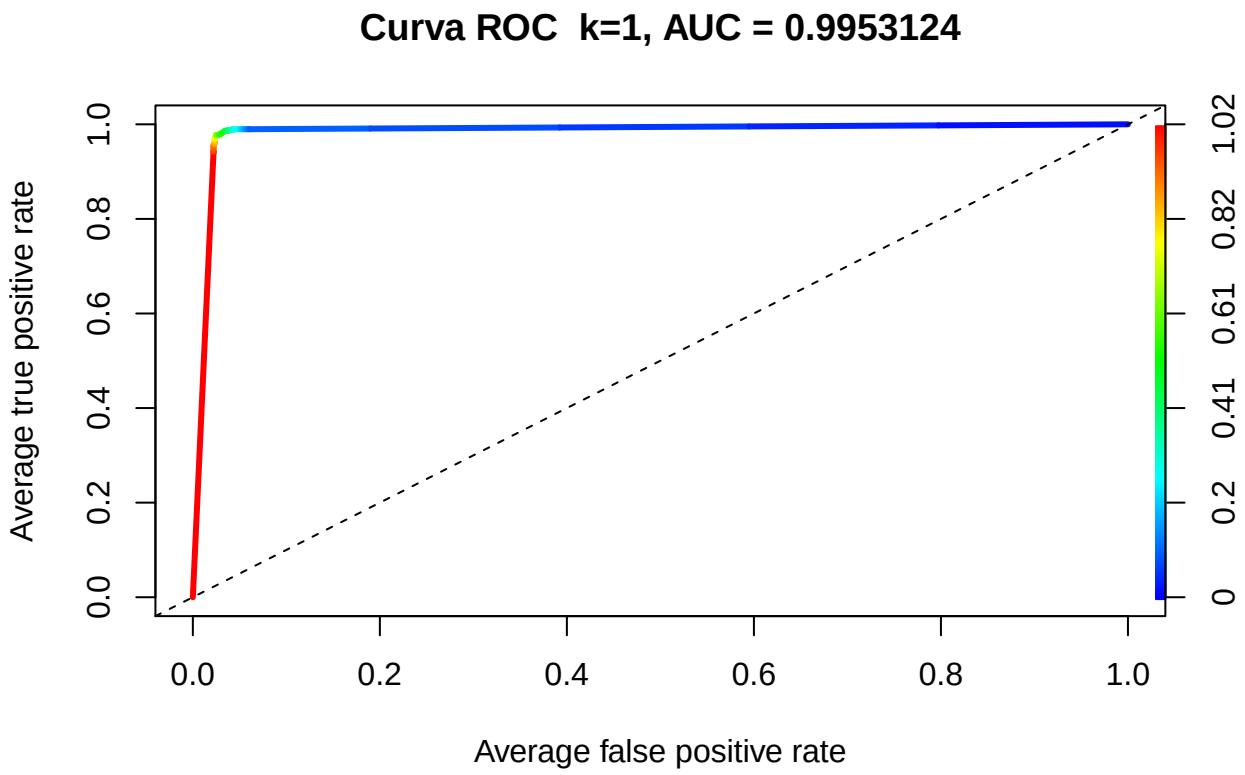
```
library(ROCR)
#K=1
test_predicion_1<-knn(train=training_one_hot, test=test_one_hot,cl=training_one_hot$coil, k=1,prob=T)
test_probabilidad_1<-attr(test_predicion_1,"prob")
t_1_probabilidad<-ifelse(predicion_coil_1==1,test_probabilidad_1,1-test_probabilidad_1)
test_1nc_probabilidad<-1-t_1_probabilidad
resultados_coil_1<-data.frame(test_one_hot$coil,predicion_coil_1, t_1_probabilidad,test_1nc_probabilidad
head(resultados_coil_1)
```

```
##   test_one_hot.coil predicion_coil_1 t_1_probabilidad test_1nc_probabilidad
## 1                 1                1                1                      0
## 2                 0                0                0                      1
## 3                 1                1                1                      0
## 4                 0                0                0                      1
## 5                 1                1                1                      0
## 6                 0                0                0                      1
```

```
library(ROCR)
predicion.k1<-prediction(predictions= resultados_coil_1$t_1_probabilidad,labels=resultados_coil_1$test_
performance1k<-performance(predicion.k1,measure="tpr",x.measure="fpr")
auc.performance1k<-performance(predicion.k1,measure="auc")
auc.performance1k<-unlist(auc.performance1k@y.values)
auc.performance1k
```

```
## [1] 0.9828701
```

```
plot(performance1k,avg="threshold",colorize=T,lwd=3,
main=("Curva ROC  k=1, AUC = 0.9953124"))
abline(a=0,b=1,lwd=1,lty=2)
```



Voy ahora con los otros 4 knn que me faltan.

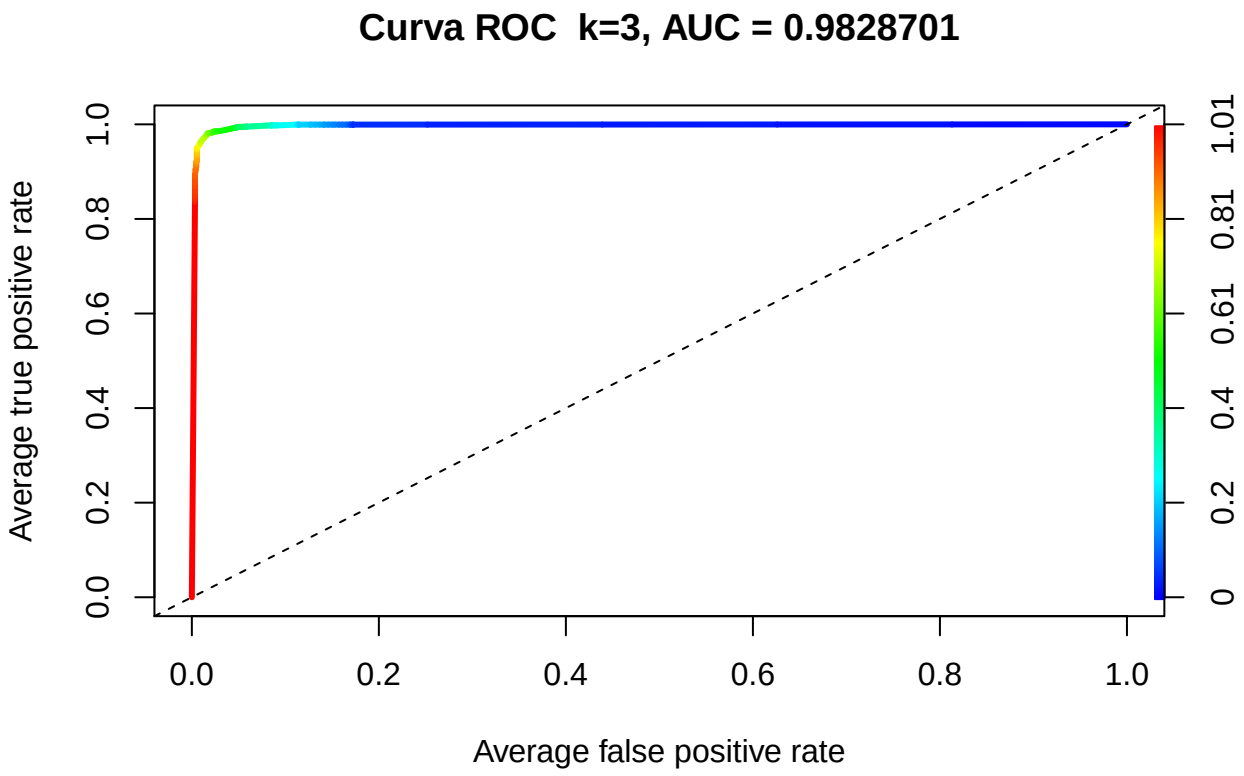
```
library(ROCR)
#K=3
test_predicion_3<-knn(train=training_one_hot, test=test_one_hot,cl=training_one_hot$coil, k=3,prob=T)
test_probabilidad_3<-attr(test_predicion_3,"prob")
t_3_probabilidad<-ifelse(predicion_coil_3==1,test_probabilidad_3,1-test_probabilidad_3)
test_3nc_probabilidad<-1-t_3_probabilidad
resultados_coil_3<-data.frame(test_one_hot$coil,predicion_coil_3, t_3_probabilidad,test_3nc_probabilidad
head(resultados_coil_3)
```

```
##   test_one_hot.coil predicion_coil_3 t_3_probabilidad test_3nc_probabilidad
## 1                 1                1        1.0000000            0.00000000
## 2                 0                0        0.0000000            1.00000000
## 3                 1                1        1.0000000            0.00000000
## 4                 0                0        0.0000000            1.00000000
## 5                 1                1        0.9545455            0.04545455
## 6                 0                0        0.0000000            1.00000000
```

```
library(ROCR)
predicion.k3<-prediction(predictions= resultados_coil_3$t_3_probabilidad,labels=resultados_coil_3$test_
performance3k<-performance(predicion.k3,measure="tpr",x.measure="fpr")
auc.performance3k<-performance(predicion.k3,measure="auc")
auc.performance3k<-unlist(auc.performance3k@y.values)
auc.performance3k
```

```
## [1] 0.9966216
```

```
plot(performance3k,avg="threshold",colorize=T,lwd=3,
main=("Curva ROC  k=3, AUC = 0.9828701"))
abline(a=0,b=1,lwd=1,lty=2)
```



k5

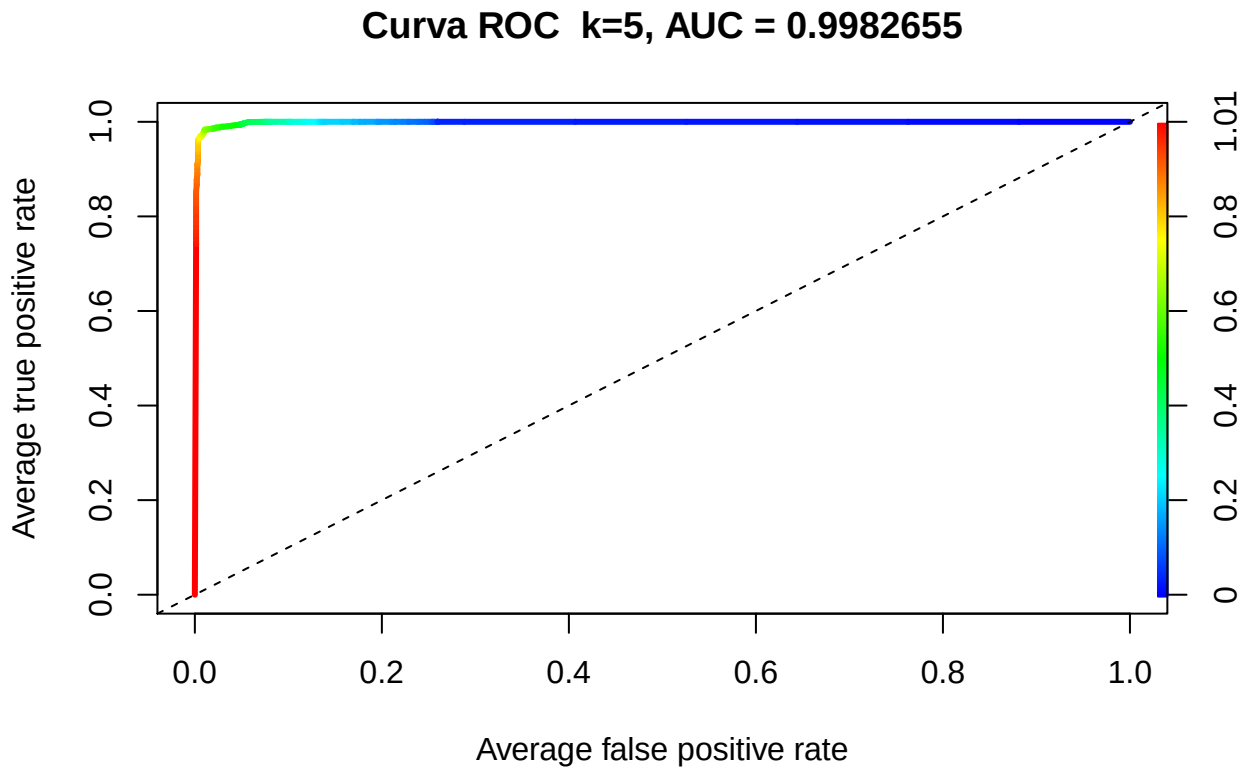
```
library(ROCR)
#K=5
test_predicion_5<-knn(train=training_one_hot, test=test_one_hot,cl=training_one_hot$coil, k=5,prob=T)
test_probabilidad_5<-attr(test_predicion_5,"prob")
t_5_probabilidad<-ifelse(predicion_coil_5==1,test_probabilidad_5,1-test_probabilidad_5)
test_5nc_probabilidad<-1-t_5_probabilidad
resultados_coil_5<-data.frame(test_one_hot$coil,predicion_coil_5, t_5_probabilidad,test_5nc_probabilidad
head(resultados_coil_5)
```

```
##   test_one_hot.coil predicion_coil_5 t_5_probabilidad test_5nc_probabilidad
## 1                 1                1        1.0000000            0.00000000
## 2                 0                0        0.0000000            1.00000000
## 3                 1                1        1.0000000            0.00000000
## 4                 0                0        0.0000000            1.00000000
## 5                 1                1        0.9545455            0.04545455
## 6                 0                0        0.2500000            0.75000000
```

```
library(ROCR)
predicion.k5<-prediction(predictions= resultados_coil_5$t_5_probabilidad,labels=resultados_coil_5$test_
performance5k<-performance(predicion.k5,measure="tpr",x.measure="fpr")
auc.performance5k<-performance(predicion.k5,measure="auc")
auc.performance5k<-unlist(auc.performance5k@y.values)
auc.performance5k
```

```
## [1] 0.9982655
```

```
plot(performance5k,avg="threshold",colorize=T,lwd=3,
main=("Curva ROC  k=5, AUC = 0.9982655"))
abline(a=0,b=1,lwd=1,lty=2)
```



k7

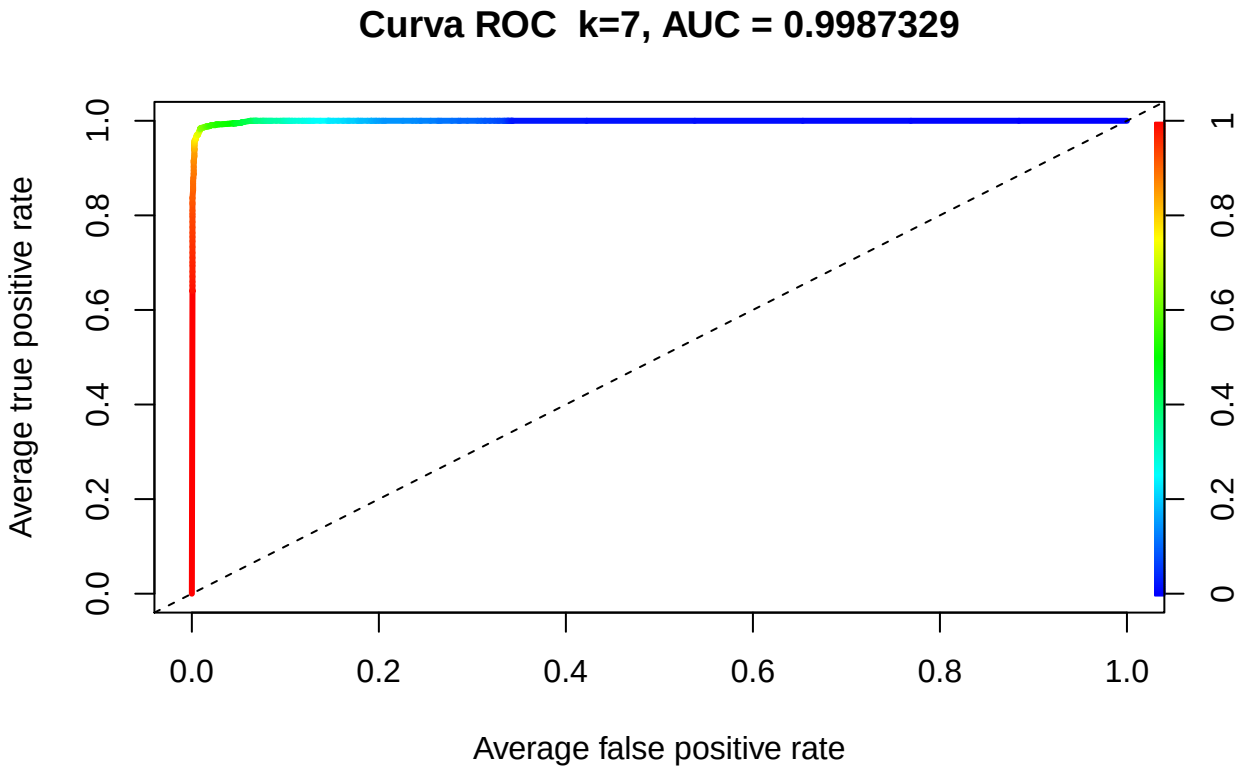
```
library(ROCR)
#K=7
test_predicion_7<-knn(train=training_one_hot, test=test_one_hot,cl=training_one_hot$coil, k=7,prob=T)
test_probabilidad_7<-attr(test_predicion_7,"prob")
t_7_probabilidad<-ifelse(predicion_coil_7==1,test_probabilidad_7,1-test_probabilidad_7)
test_7nc_probabilidad<-1-t_7_probabilidad
resultados_coil_7<-data.frame(test_one_hot$coil,predicion_coil_7, t_7_probabilidad,test_7nc_probabilidad
head(resultados_coil_7)
```

```
##   test_one_hot.coil predicion_coil_7 t_7_probabilidad test_7nc_probabilidad
## 1                 1                1        1.0000000            0.00000000
## 2                 0                0        0.1000000            0.90000000
## 3                 1                1        1.0000000            0.00000000
## 4                 0                0        0.0000000            1.00000000
## 5                 1                1        0.9545455            0.04545455
## 6                 0                0        0.2500000            0.75000000
```

```
library(ROCR)
predicion.k7<-prediction(predictions= resultados_coil_7$t_7_probabilidad,labels=resultados_coil_7$test_
performance7k<-performance(predicion.k7,measure="tpr",x.measure="fpr")
auc.performance7k<-performance(predicion.k7,measure="auc")
auc.performance7k<-unlist(auc.performance7k@y.values)
auc.performance7k
```

```
## [1] 0.9987329
```

```
plot(performance7k,avg="threshold",colorize=T,lwd=3,
main=("Curva ROC  k=7, AUC = 0.9987329"))
abline(a=0,b=1,lwd=1,lty=2)
```



k11

```
library(ROCR)
#K=11
test_predicion_11<-knn(train=training_one_hot, test=test_one_hot,cl=training_one_hot$coil, k=11,prob=T)
test_probabilidad_11<-attr(test_predicion_11,"prob")
t_11_probabilidad<-ifelse(predicion_coil_11==1,test_probabilidad_11,1-test_probabilidad_11)
test_11nc_probabilidad<-1-t_11_probabilidad
resultados_coil_11<-data.frame(test_one_hot$coil,predicion_coil_11, t_11_probabilidad,test_11nc_probabil
head(resultados_coil_11)
```

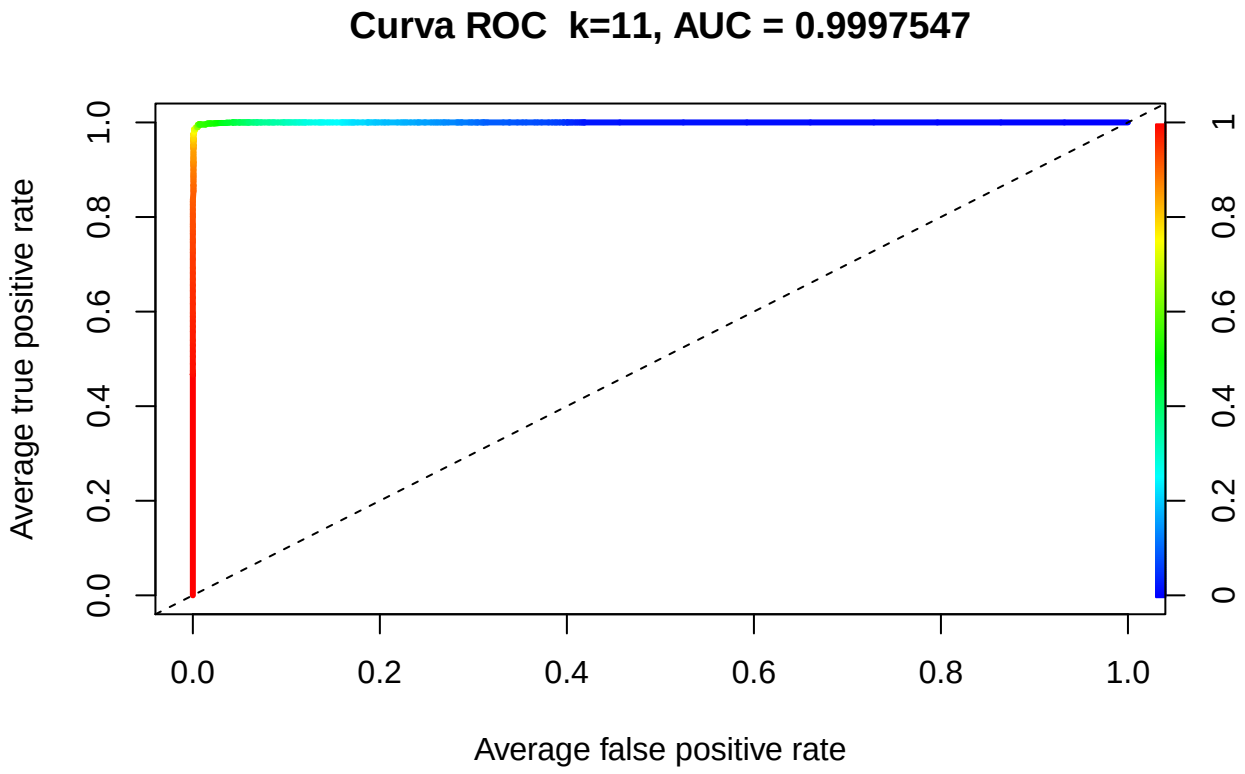
```
##   test_one_hot.coil predicion_coil_11 t_11_probabilidad test_11nc_probabilidad
```

```
## 1                1                1         0.9545455          0.04545455
## 2                0                0         0.2500000          0.75000000
## 3                1                1         1.0000000          0.00000000
## 4                0                0         0.0000000          1.00000000
## 5                1                1         0.9545455          0.04545455
## 6                0                0         0.2500000          0.75000000
```

```
library(ROCR)
predicion.k11<-prediction(predictions= resultados_coil_11$t_11_probabilidad,labels=resultados_coil_11$t
performance11k<-performance(predicion.k11,measure="tpr",x.measure="fpr")
auc.performance11k<-performance(predicion.k11,measure="auc")
auc.performance11k<-unlist(auc.performance11k@y.values)
auc.performance11k
```

```
## [1] 0.9997547
```

```
plot(performance11k,avg="threshold",colorize=T,lwd=3,
main=("Curva ROC  k=11, AUC = 0.9997547"))
abline(a=0,b=1,lwd=1,lty=2)
```



##(f) Comentar los resultados de la clasificación para las tres clases de estructuras secundarias basado,como mínimo, en el error de clasificación y el valor de kappa. Además, comentar los resultados para las clases coil y non-coil en función del AUC, número de falsos positivos, falsos negativos y error de clasificación obtenidos para los diferentes valores de k.

```
library(caret)
Matriz_1<-CrossTable(x=test_one_hot$V2, y=predicion1,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion1
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##               1 |       766 |        60 |        14 |       840 |
##                 |     0.912 |     0.071 |     0.017 |     0.252 |
##                 |     0.954 |     0.088 |     0.008 |           |
##                 |     0.230 |     0.018 |     0.004 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               2 |        32 |       519 |       102 |       653 |
##                 |     0.049 |     0.795 |     0.156 |     0.196 |
##                 |     0.040 |     0.759 |     0.055 |           |
##                 |     0.010 |     0.156 |     0.031 |           |
## ----------------|-----------|-----------|-----------|-----------|
##               3 |         5 |       105 |      1730 |      1840 |
##                 |     0.003 |     0.057 |     0.940 |     0.552 |
##                 |     0.006 |     0.154 |     0.937 |           |
##                 |     0.002 |     0.032 |     0.519 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       803 |       684 |      1846 |      3333 |
##                 |     0.241 |     0.205 |     0.554 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

```
table(test_one_hot$V2)
```

```
##
##    1    2    3
##  840  653 1840
```

Pasamos a factor

```
test_one_hot$V2 <- factor(test_one_hot$V2, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(test_one_hot$V2)
```

```
##
## a_helix b_sheet    coil
##     840     653    1840
```

```
predicion1 <- factor(predicion1, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(predicion1)
```

```
## predicion1
## a_helix b_sheet    coil
##     803     684    1846
```

Creo la matriz de confusion, para poder obtener el valor kappa y el erro

```
confusionMatrix(predicion1,test_one_hot$V2, positive="non_coil")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction a_helix b_sheet coil
##    a_helix     766      32    5
##    b_sheet      60     519  105
##    coil         14     102 1730
##
## Overall Statistics
##
##                Accuracy : 0.9046
##                  95% CI : (0.8941, 0.9144)
##     No Information Rate : 0.5521
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8392
##
##  Mcnemar's Test P-Value : 0.005023
##
## Statistics by Class:
##
##                      Class: a_helix Class: b_sheet Class: coil
## Sensitivity                  0.9119         0.7948      0.9402
## Specificity                  0.9852         0.9384      0.9223
## Pos Pred Value               0.9539         0.7588      0.9372
## Neg Pred Value               0.9708         0.9494      0.9260
## Prevalence                   0.2520         0.1959      0.5521
## Detection Rate               0.2298         0.1557      0.5191
## Detection Prevalence         0.2409         0.2052      0.5539
## Balanced Accuracy            0.9485         0.8666      0.9313
```

Calculo el error

```
print(error_1 <- 1-0.9046)
```

```
## [1] 0.0954
```

Podemos ver que el error que obtengo es de 0.0954 y estoy obteniendo un valor de kappa de 0.8392, son buenos valores, pero voy aprobar en las siguientes ***k=3,5,7,11***

```
library(caret)
Matriz_3<-CrossTable(x=test_one_hot$V2, y=predicion3,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion3
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##         a_helix |       769 |        63 |         8 |       840 |
##                 |     0.915 |     0.075 |     0.010 |     0.252 |
##                 |     0.942 |     0.105 |     0.004 |           |
##                 |     0.231 |     0.019 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##         b_sheet |        45 |       442 |       166 |       653 |
##                 |     0.069 |     0.677 |     0.254 |     0.196 |
##                 |     0.055 |     0.734 |     0.087 |           |
##                 |     0.014 |     0.133 |     0.050 |           |
## ----------------|-----------|-----------|-----------|-----------|
##            coil |         2 |        97 |      1741 |      1840 |
##                 |     0.001 |     0.053 |     0.946 |     0.552 |
##                 |     0.002 |     0.161 |     0.909 |           |
##                 |     0.001 |     0.029 |     0.522 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       816 |       602 |      1915 |      3333 |
##                 |     0.245 |     0.181 |     0.575 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

```
predicion3 <- factor(predicion3, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(predicion3)
```

```
## predicion3
## a_helix b_sheet    coil
##     816     602    1915
```

```
confusionMatrix(predicion3,test_one_hot$V2, positive="non_coil")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction a_helix b_sheet coil
##    a_helix     769      45    2
##    b_sheet      63     442   97
##    coil          8     166 1741
##
## Overall Statistics
##
##                Accuracy : 0.8857
##                  95% CI : (0.8744, 0.8963)
##     No Information Rate : 0.5521
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8048
##
##  Mcnemar's Test P-Value : 1.782e-05
##
## Statistics by Class:
##
##                      Class: a_helix Class: b_sheet Class: coil
## Sensitivity                  0.9155         0.6769      0.9462
## Specificity                  0.9811         0.9403      0.8835
## Pos Pred Value               0.9424         0.7342      0.9091
## Neg Pred Value               0.9718         0.9227      0.9302
## Prevalence                   0.2520         0.1959      0.5521
## Detection Rate               0.2307         0.1326      0.5224
## Detection Prevalence         0.2448         0.1806      0.5746
## Balanced Accuracy            0.9483         0.8086      0.9148
```

Calculamos el error

```
print(error_3 <- 1-0.8048)
```

```
## [1] 0.1952
```

Sigo con el ***k5***

```
library(caret)
Matriz_5<-CrossTable(x=test_one_hot$V2, y=predicion5,
prop.chisq = FALSE)
```

```
##
##
```

```
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion5
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##         a_helix |       758 |        76 |         6 |       840 |
##                 |     0.902 |     0.090 |     0.007 |     0.252 |
##                 |     0.940 |     0.136 |     0.003 |           |
##                 |     0.227 |     0.023 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##         b_sheet |        47 |       391 |       215 |       653 |
##                 |     0.072 |     0.599 |     0.329 |     0.196 |
##                 |     0.058 |     0.701 |     0.109 |           |
##                 |     0.014 |     0.117 |     0.065 |           |
## ----------------|-----------|-----------|-----------|-----------|
##            coil |         1 |        91 |      1748 |      1840 |
##                 |     0.001 |     0.049 |     0.950 |     0.552 |
##                 |     0.001 |     0.163 |     0.888 |           |
##                 |     0.000 |     0.027 |     0.524 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       806 |       558 |      1969 |      3333 |
##                 |     0.242 |     0.167 |     0.591 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

```
predicion5 <- factor(predicion5, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(predicion5)
```

```
## predicion5
## a_helix b_sheet    coil
##     806     558    1969
```

```
confusionMatrix(predicion5,test_one_hot$V2, positive="non_coil")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction a_helix b_sheet coil
##    a_helix     758      47    1
##    b_sheet      76     391   91
##    coil          6     215 1748
```

```
##
## Overall Statistics
##
##                Accuracy : 0.8692
##                  95% CI : (0.8573, 0.8805)
##     No Information Rate : 0.5521
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7745
##
##  Mcnemar's Test P-Value : 4.254e-13
##
## Statistics by Class:
##
##                      Class: a_helix Class: b_sheet Class: coil
## Sensitivity                  0.9024         0.5988      0.9500
## Specificity                  0.9807         0.9377      0.8520
## Pos Pred Value               0.9404         0.7007      0.8878
## Neg Pred Value               0.9676         0.9056      0.9326
## Prevalence                   0.2520         0.1959      0.5521
## Detection Rate               0.2274         0.1173      0.5245
## Detection Prevalence         0.2418         0.1674      0.5908
## Balanced Accuracy            0.9416         0.7682      0.9010
```

Hago el error para ***k5***

```
print(error_5 <- 1-0.7745)
```

```
## [1] 0.2255
```

Este valor es peor

```
library(caret)
Matriz_7<-CrossTable(x=test_one_hot$V2, y=predicion7,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion7
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
```

```
## ----------------|-----------|-----------|-----------|-----------|
##         a_helix |       745 |        90 |         5 |       840 |
##                 |     0.887 |     0.107 |     0.006 |     0.252 |
##                 |     0.936 |     0.174 |     0.002 |           |
##                 |     0.224 |     0.027 |     0.002 |           |
## ----------------|-----------|-----------|-----------|-----------|
##         b_sheet |        50 |       348 |       255 |       653 |
##                 |     0.077 |     0.533 |     0.391 |     0.196 |
##                 |     0.063 |     0.672 |     0.126 |           |
##                 |     0.015 |     0.104 |     0.077 |           |
## ----------------|-----------|-----------|-----------|-----------|
##            coil |         1 |        80 |      1759 |      1840 |
##                 |     0.001 |     0.043 |     0.956 |     0.552 |
##                 |     0.001 |     0.154 |     0.871 |           |
##                 |     0.000 |     0.024 |     0.528 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       796 |       518 |      2019 |      3333 |
##                 |     0.239 |     0.155 |     0.606 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

```
predicion7 <- factor(predicion7, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(predicion7)
```

```
## predicion7
## a_helix b_sheet    coil
##     796     518    2019
```

```
confusionMatrix(predicion7,test_one_hot$V2, positive="non_coil")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction a_helix b_sheet coil
##    a_helix     745      50    1
##    b_sheet      90     348   80
##    coil          5     255 1759
##
## Overall Statistics
##
##                Accuracy : 0.8557
##                  95% CI : (0.8433, 0.8674)
##     No Information Rate : 0.5521
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.749
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
```

```
##                      Class: a_helix Class: b_sheet Class: coil
## Sensitivity                  0.8869         0.5329      0.9560
## Specificity                  0.9795         0.9366      0.8259
## Pos Pred Value               0.9359         0.6718      0.8712
## Neg Pred Value               0.9626         0.8917      0.9384
## Prevalence                   0.2520         0.1959      0.5521
## Detection Rate               0.2235         0.1044      0.5278
## Detection Prevalence         0.2388         0.1554      0.6058
## Balanced Accuracy            0.9332         0.7347      0.8909
```

Hago el error a partir del valor de kappa

```
print(error_7 <- 1-0.749)
```

```
## [1] 0.251
```

Hago el ultimo el valor de ***k11***

```
library(caret)
Matriz_11<-CrossTable(x=test_one_hot$V2, y=predicion11,
prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  3333
##
##
##                  | predicion11
## test_one_hot$V2 |         1 |         2 |         3 | Row Total |
## ----------------|-----------|-----------|-----------|-----------|
##         a_helix |       768 |        71 |         1 |       840 |
##                 |     0.914 |     0.085 |     0.001 |     0.252 |
##                 |     0.941 |     0.171 |     0.000 |           |
##                 |     0.230 |     0.021 |     0.000 |           |
## ----------------|-----------|-----------|-----------|-----------|
##         b_sheet |        48 |       300 |       305 |       653 |
##                 |     0.074 |     0.459 |     0.467 |     0.196 |
##                 |     0.059 |     0.721 |     0.145 |           |
##                 |     0.014 |     0.090 |     0.092 |           |
## ----------------|-----------|-----------|-----------|-----------|
##            coil |         0 |        45 |      1795 |      1840 |
##                 |     0.000 |     0.024 |     0.976 |     0.552 |
##                 |     0.000 |     0.108 |     0.854 |           |
```

```
##                 |     0.000 |     0.014 |     0.539 |           |
## ----------------|-----------|-----------|-----------|-----------|
##    Column Total |       816 |       416 |      2101 |      3333 |
##                 |     0.245 |     0.125 |     0.630 |           |
## ----------------|-----------|-----------|-----------|-----------|
##
##
```

```
predicion11 <- factor(predicion11, levels=c(1,2,3),
labels=c("a_helix","b_sheet","coil"))
table(predicion11)
```

```
## predicion11
## a_helix b_sheet    coil
##     816     416    2101
```

hago la ultima matriz de confusion con el valor de k11

```
confusionMatrix(predicion11,test_one_hot$V2, positive="non_coil")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction a_helix b_sheet coil
##    a_helix     768      48    0
##    b_sheet      71     300   45
##    coil          1     305 1795
##
## Overall Statistics
##
##                Accuracy : 0.859
##                  95% CI : (0.8467, 0.8706)
##     No Information Rate : 0.5521
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7508
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: a_helix Class: b_sheet Class: coil
## Sensitivity                  0.9143        0.45942      0.9755
## Specificity                  0.9807        0.95672      0.7950
## Pos Pred Value               0.9412        0.72115      0.8544
## Neg Pred Value               0.9714        0.87899      0.9635
## Prevalence                   0.2520        0.19592      0.5521
## Detection Rate               0.2304        0.09001      0.5386
## Detection Prevalence         0.2448        0.12481      0.6304
## Balanced Accuracy            0.9475        0.70807      0.8853
```

Hago el ultimo error a partir del valor kappa

```r
print(error_11 <- 1-0.7508)
```

```
## [1] 0.2492
```

Como podemos observar el mejor ha sido el k1 con un valor ***kappa= 0.8392, y un error= 0.0954***, los modelos van empeorando entre ellos el k3 y k11, pero los peores son el k5 y k7, Podemos ver que los datos sacados en las curvas ROC son buenos estan muy proximos a 1, el mejor AUC en las curvas ROOC es el k11 con un valor de AUC = 0.9997547.