Prediccion de KNN

JULIAN JAVIER GONZALEZ HERNANDEZ

2023-10-08

Parte 1: Exploracion de datos

```
folder <- dirname(rstudioapi :: getSourceEditorContext()$path)

parentFolder <- dirname (folder)
data_set_dia <-
  read.csv(paste0(parentFolder,"/diabetes_012_health_indicators_BRFSS2015.csv"))
```

Una vez que hayamos cargado nuestra colección de datos, es crucial examinar y evaluar la información que se encuentra en este archivo. En la ilustración siguiente, se presentan las variables junto con una breve descripción de su contenido.

Luego, empleando la función psych, podemos obtener un análisis estadístico de las 22 variables presentes en el conjunto de datos, abarcando medidas como la media, desviación estándar, rango mínimo y máximo, entre otros.

Por último, al emplear la función mutar, procederemos a modificar todos los datos que no sean "= 0" en la variable Diabetes_012. Después, exhibiremos en una tabla concisa la cantidad de datos clasificados como "0" o "1" en esta variable de nuestro conjunto de datos.

```
test_diabetes<- data_set_dia %>% mutate(Diabetes_012 = ifelse(Diabetes_012!= "0", "1",Diabetes_012))
```

```
Conteo_Diabetes
```

```
##
##      0      1
## 213703  39977
```

Parte 2: KNN PREDICCIÓN DE DIABETES KNN

Primera predicción En esta sección del documento, implementaremos el método predictivo KNN. Para ello, emplearemos tres variables distintas para realizar las predicciones. Inicialmente, mediante un muestreo estratificado, seleccionaremos alrededor del 1% de los datos para el entrenamiento de nuestros modelos.

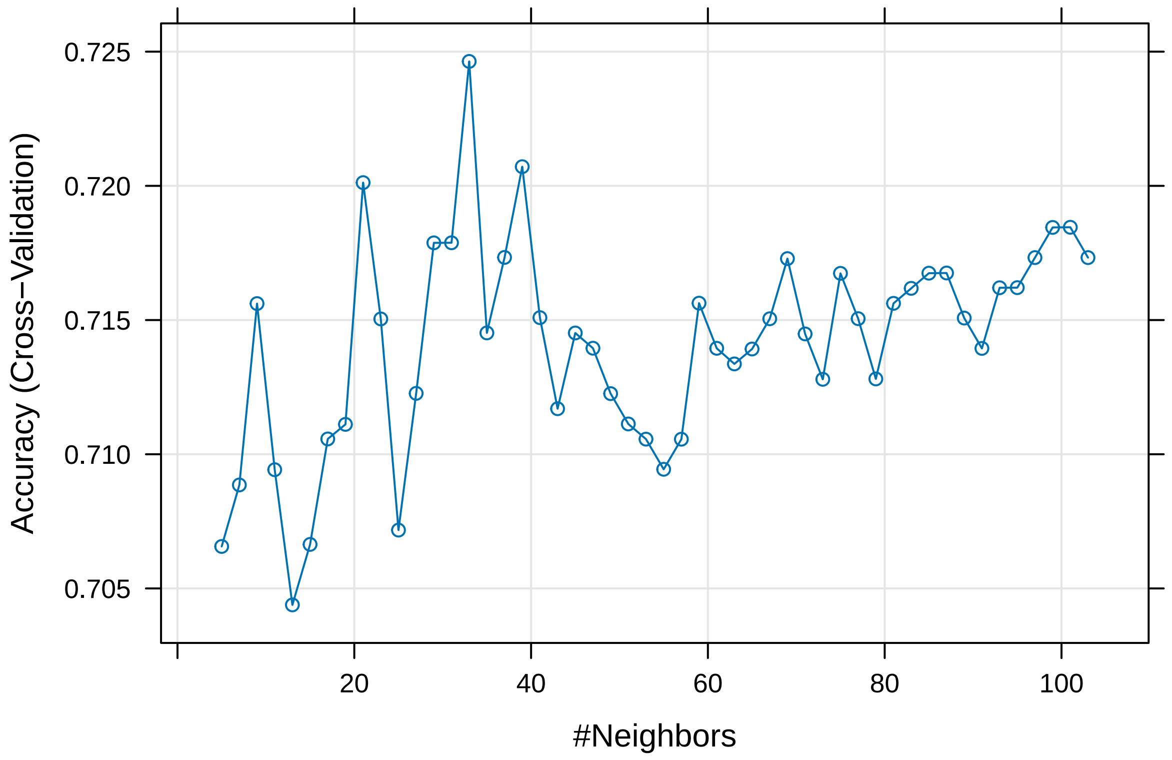
```
Sample_diabetes <- test_diabetes %>%
  group_by(Diabetes_012) %>%
  sample_n(1269, replace = TRUE) %>%
  ungroup()
```

```
Conteo_Sample_diabetes
```

```
##
##    0    1
## 1269 1269
```

```
set.seed(123)
Sample_diabetes_knn <- Sample_diabetes %>%
  group_by(Diabetes_012) %>%
  sample_n(1269, replace = TRUE) %>%
  ungroup()


S.index <- sample(1:nrow(Sample_diabetes_knn)
                       ,nrow(Sample_diabetes_knn)*0.7
                       ,replace = F)


predictors <- c("HighBP", "HighChol", "CholCheck", "BMI", "Smoker", "Stroke", "HeartDiseaseorAttack", "

TR.Data <- Sample_diabetes_knn[S.index, c(predictors, "Diabetes_012"), drop = FALSE]
TS.data <- Sample_diabetes_knn[-S.index, c(predictors, "Diabetes_012"), drop = FALSE]


TR.Data$Diabetes_012 <- factor(TR.Data$Diabetes_012)
TS.data$Diabetes_012 <- factor(TS.data$Diabetes_012)
```

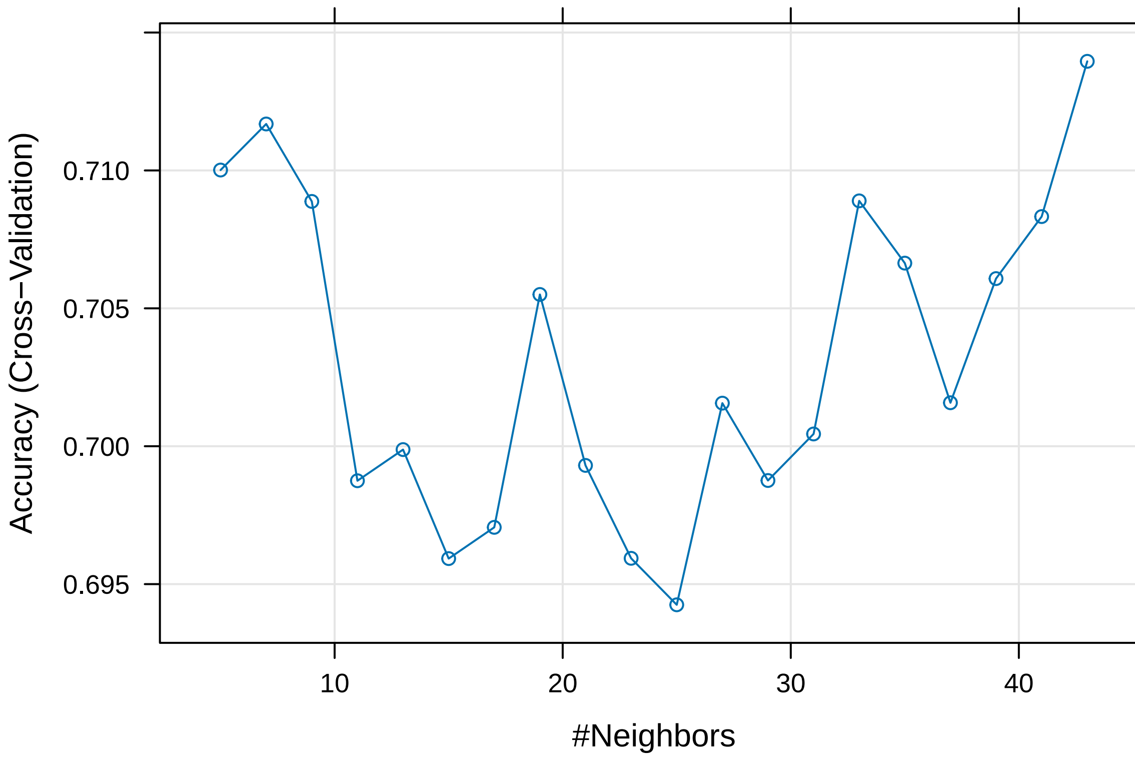
```
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(Diabetes_012 ~ .
                , data = TR.Data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 50)

plot(knnFit)
```



```r
# Hacer predicciones
knnPredict <- predict(knnFit, newdata = TS.data)

# Crea la matriiz de confusion
confusionMatrix(data = knnPredict, reference = TS.data$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 243  82
##          1 135 302
##
##                Accuracy : 0.7152
##                  95% CI : (0.6817, 0.747)
##     No Information Rate : 0.5039
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4298
##
##  Mcnemar's Test P-Value : 0.0004156
##
##             Sensitivity : 0.6429
##             Specificity : 0.7865
##          Pos Pred Value : 0.7477
##          Neg Pred Value : 0.6911
##               Prevalence : 0.4961
##           Detection Rate : 0.3189
##     Detection Prevalence : 0.4265
##        Balanced Accuracy : 0.7147
##
##         'Positive' Class : 0
##
```

```
predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Education", "Income")
TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]


ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(Diabetes_012 ~ .
                 , data = TR.Data2
                 , method = "knn", trControl = ctrl
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 20)

plot(knnFit2)
```

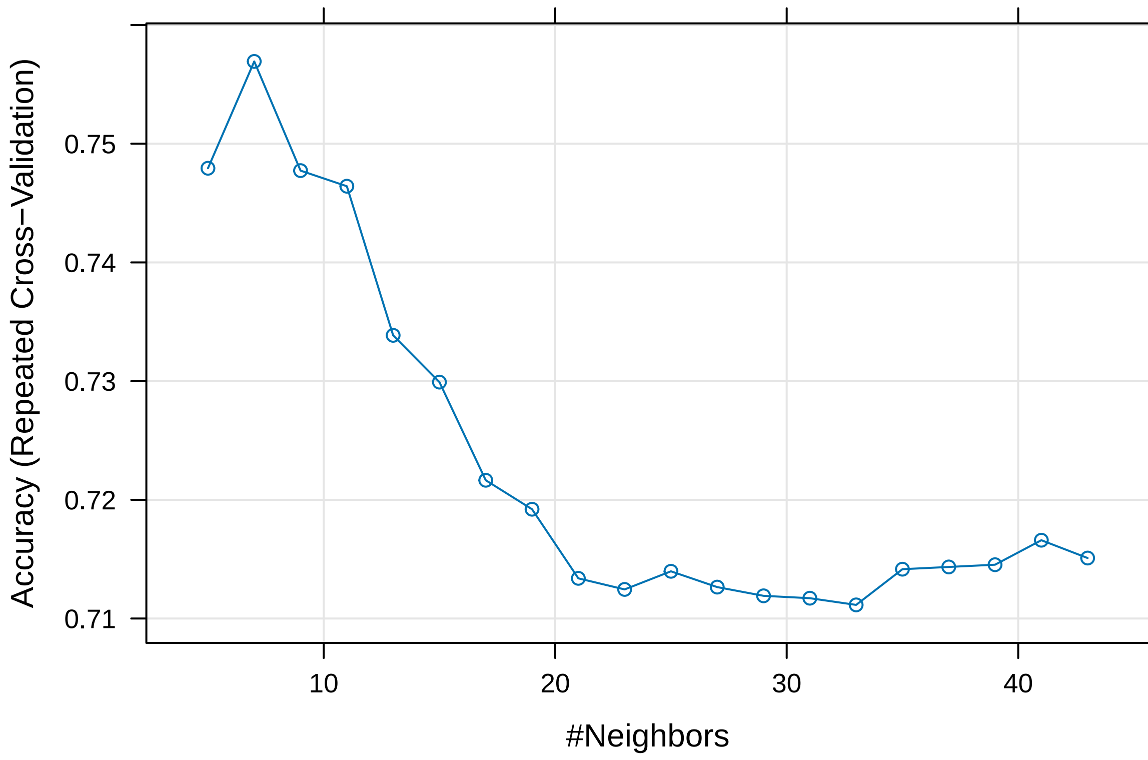


Segunda prediccion

```r
# crea predicciones
knnPredict2 <- predict(knnFit2, newdata = TS.data2)

# Crea la matriiz de confusion
confusionMatrix(data = knnPredict2, reference = TS.data2$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 239  78
##          1 139 306
##
##                Accuracy : 0.7152
##                  95% CI : (0.6817, 0.747)
##     No Information Rate : 0.5039
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4297
##
##  Mcnemar's Test P-Value : 4.64e-05
##
##             Sensitivity : 0.6323
##             Specificity : 0.7969
##          Pos Pred Value : 0.7539
##          Neg Pred Value : 0.6876
##              Prevalence : 0.4961
##          Detection Rate : 0.3136
##    Detection Prevalence : 0.4160
##       Balanced Accuracy : 0.7146
##
##        'Positive' Class : 0
##
```

```r
predictors_to_remove2 <- c("CholCheck", "MentHlth","PhysHlth", "Fruits", "Veggies")
TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove2)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove2)]

ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(Diabetes_012 ~ .
                 , data = TR.Data3
                 , method = "knn", trControl = ctrl2
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 20)

plot(knnFit3)
```



Tercera prediccion

```
knnPredict3 <- predict(knnFit3, newdata = TS.data3)

# Crea la matriiz de confusion
confusionMatrix(data = knnPredict3, reference = TS.data3$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 256  66
##          1 122 318
##
##                Accuracy : 0.7533
##                  95% CI : (0.7211, 0.7835)
##     No Information Rate : 0.5039
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.506
##
##  Mcnemar's Test P-Value : 6.039e-05
##
##             Sensitivity : 0.6772
##             Specificity : 0.8281
##          Pos Pred Value : 0.7950
##          Neg Pred Value : 0.7227
##               Prevalence : 0.4961
##           Detection Rate : 0.3360
##     Detection Prevalence : 0.4226
##        Balanced Accuracy : 0.7527
##
##         'Positive' Class : 0
##
```

KNN HeartDiseaseorAttack Prediction

```
set.seed(123)
ss_heartDiseaseorAttack <- Sample_diabetes %>%
  group_by(HeartDiseaseorAttack) %>%
  sample_n(1269, replace = TRUE) %>%
  ungroup()

predictors <- c("Diabetes_012","HighBP", "HighChol", "CholCheck", "BMI", "Smoker", "Stroke",  "PhysActi

# Datos originales
TR.Data <- ss_heartDiseaseorAttack[S.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]
TS.data <- ss_heartDiseaseorAttack[-S.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]

TR.Data$HeartDiseaseorAttack <- factor(TR.Data$HeartDiseaseorAttack)
TS.data$HeartDiseaseorAttack <- factor(TS.data$HeartDiseaseorAttack)

# Entrena el modelo KNN
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(HeartDiseaseorAttack ~ .
                , data = TR.Data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 50)

# Crear predicciones
knnPredict <- predict(knnFit, newdata = TS.data)

# Crea la matriiz de confusion
# Datos originales
TR.Data <- ss_heartDiseaseorAttack[S.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]
TS.data <- ss_heartDiseaseorAttack[-S.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]

TR.Data$HeartDiseaseorAttack <- factor(TR.Data$HeartDiseaseorAttack)
TS.data$HeartDiseaseorAttack <- factor(TS.data$HeartDiseaseorAttack)

# Entrena el modelo KNN
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(HeartDiseaseorAttack ~ .
                , data = TR.Data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 50)
```

```r
# crear predicciones
knnPredict <- predict(knnFit, newdata = TS.data)

# Crea la matriiz de confusion
confusionMatrix(data = knnPredict, reference = TS.data$HeartDiseaseorAttack)
```

Primera prediccion

```r
### Segundo modelo

predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Education", "Income")
TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]

# Entrena el modelo KNN
ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(HeartDiseaseorAttack ~ .
                 , data = TR.Data2
                 , method = "knn", trControl = ctrl
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 50)


# crear predicciones
knnPredict2 <- predict(knnFit2, newdata = TS.data2)

#Crea la matriiz de confusion
confusionMatrix(data = knnPredict2, reference = TS.data2$HeartDiseaseorAttack)
```

Segunda prediccion

```r
### Tercer Modelo

predictors_to_remove2 <- c("CholCheck", "MentHlth","HvyAlcoholConsump", "Fruits", "Veggies")
TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove2)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove2)]

# Entrena el modelo KNN
ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(HeartDiseaseorAttack ~ .
                 , data = TR.Data3
                 , method = "knn", trControl = ctrl2
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 50)


# Crear predicciones
```

```r
knnPredict3 <- predict(knnFit3, newdata = TS.data3)

# Crea la matriiz de confusion
confusionMatrix(data = knnPredict3, reference = TS.data3$HeartDiseaseorAttack)
```

Tercera prediccion

KNN Encuentra predicción de sexo

```r
### Modelos y experimentos de KNN para encontrar sexo ##################################################



## selección de 1500 muestras de cada factor del conjunto de datos#
set.seed(123)
ss_sex <- Sample_diabetes %>%
  group_by(Sex) %>%
  sample_n(1269, replace = TRUE) %>%
  ungroup()

predictors <- c("Diabetes_012","HighBP", "HighChol", "CholCheck", "BMI", "Smoker", "Stroke", "HeartDisea

# Datos Originales
TR.Data <- ss_sex[S.index, c(predictors, "Sex"), drop = FALSE]
TS.data <- ss_sex[-S.index, c(predictors, "Sex"), drop = FALSE]

TR.Data$Sex <- factor(TR.Data$Sex)
TS.data$Sex <- factor(TS.data$Sex)

# Entrena el modelo KNN
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(Sex ~ .
                , data = TR.Data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 50)


# Crear predicciones
knnPredict <- predict(knnFit, newdata = TS.data)

#Crea la matriiz de confusion
confusionMatrix(data = knnPredict, reference = TS.data$Sex)
```

Primera predicción

```r


# Segundo modelo
```

```
predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Age", "PhysActivity")
TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]

# Entrena el modelo KNN
ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(Sex ~ .
                 , data = TR.Data2
                 , method = "knn", trControl = ctrl
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 50)

#Crear predicciones
knnPredict2 <- predict(knnFit2, newdata = TS.data2)

# Crea la matriiz de confusion
  confusionMatrix(data = knnPredict2, reference = TS.data2$Sex)
```

Segunda predicción

```


### Tercer modelo

predictors_to_remove2 <- c("CholCheck", "MentHlth","HvyAlcoholConsump", "Fruits", "Veggies")
TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove2)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove2)]

# Entrena el modelo KNN
ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(Sex ~ .
                 , data = TR.Data3
                 , method = "knn", trControl = ctrl2
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 50)


#Crear predicciones
knnPredict3 <- predict(knnFit3, newdata = TS.data3)

#  Crea la matriiz de confusion
confusionMatrix(data = knnPredict3, reference = TS.data3$Sex)
```

Tercera predicción

Parte 3: Modelo de regresión lineal BM

```


### Modelo de regresión lineal BMI ###################################################################
folder <- dirname(rstudioapi :: getSourceEditorContext()$path)
```

```r
parentFolder <- dirname (folder)
data <-
  read.csv(paste0(parentFolder,"/diabetes_012_health_indicators_BRFSS2015.csv"))

data$Diabetes_012 <- ifelse(data$Diabetes_012 == 0, 0, 1)

set.seed(1)
data_estratificada2 <- data[sample(nrow(data), 3000), ]

predictors <- colnames(data_estratificada2)[-5]
S.index <- sample(1:nrow(data_estratificada2),
                       nrow(data_estratificada2) * 0.7,
                       replace = FALSE)


TR.Data <- data_estratificada2[S.index, c(predictors, "BMI"), drop = FALSE]
TS.data <- data_estratificada2[-S.index, c(predictors, "BMI"), drop = FALSE]

ins_model <- lm(BMI ~ ., data = TR.Data)

summary(ins_model)
```

Primera predicción

```
##
## Call:
## lm(formula = BMI ~ ., data = TR.Data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.218  -3.753  -0.727   2.651  59.718
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          29.700892   1.248923  23.781  < 2e-16 ***
## Diabetes_012          2.282214   0.396631   5.754 1.00e-08 ***
## HighBP                2.821500   0.297689   9.478  < 2e-16 ***
## HighChol              0.566150   0.283749   1.995 0.046146 *
## CholCheck             0.859487   0.677266   1.269 0.204564
## Smoker               -0.522257   0.272625  -1.916 0.055546 .
## Stroke               -0.813064   0.708187  -1.148 0.251063
## HeartDiseaseorAttack -1.095276   0.483551  -2.265 0.023611 *
## PhysActivity         -0.974136   0.338502  -2.878 0.004046 **
## Fruits               -0.722677   0.287499  -2.514 0.012023 *
## Veggies              -0.537476   0.351942  -1.527 0.126870
## HvyAlcoholConsump    -0.945193   0.597458  -1.582 0.113796
## AnyHealthcare         0.162150   0.612766   0.265 0.791329
## NoDocbcCost          -0.528085   0.505369  -1.045 0.296168
## GenHlth               0.621751   0.163830   3.795 0.000152 ***
## MentHlth              0.006135   0.019977   0.307 0.758794
## PhysHlth             -0.049331   0.019188  -2.571 0.010210 *
## DiffWalk              2.097624   0.436809   4.802 1.68e-06 ***
## Sex                  -0.023210   0.274379  -0.085 0.932593
## Age                   -0.414443   0.048185  -8.601  < 2e-16 ***
## Education              0.065025   0.152360   0.427 0.669579
## Income                -0.113682   0.076249  -1.491 0.136132
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.968 on 2078 degrees of freedom
## Multiple R-squared:  0.157,  Adjusted R-squared:  0.1485
## F-statistic: 18.43 on 21 and 2078 DF,  p-value: < 2.2e-16
```

```
# Entrenar el modelo
train.control <- trainControl(method = "cv", number = 10 )
model <- train(BMI ~ ., data = TR.Data, method = "lm",
               trControl = train.control)


# Resumir los resultados
print(model)
```

```
## Linear Regression
##
## 2100 samples
##   21 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1891, 1891, 1890, 1889, 1889, 1891, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   5.984649  0.1486856  4.319634
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```


#### segunda

predictors_to_remove <- c("AnyHealthcare", "CholCheck", "MentHlth", "Education", "Sex")

TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]

ins_model <- lm(BMI ~ ., data = TR.Data2)

summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "cv", number = 5)
model <- train(BMI ~ ., data = TR.Data2, method = "lm",
               trControl = train.control)
```

```r
# Resumir los resultados
print(model)
```

Segunda predicción

```r


#### Tercera
predictors_to_remove <- c("Income", "Stroke", "NoDocbcCost", "Veggies", "HvyAlcoholConsump")

TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove)]

ins_model <- lm(BMI ~ ., data = TR.Data3)

summary(ins_model)

# Entrenar el Modelo
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(BMI ~ ., data = TR.Data3, method = "lm",
               trControl = train.control)
# Resumir los resultados
print(model)
```

Tercera predicción

Modelo de regresión lineal MentHlth

```r


### Modelo de regresión lineal MentHlth ##############################################################

set.seed(1)
data_estratificada2 <- data[sample(nrow(data), 3000), ]

predictors <- colnames(data_estratificada2)[-16]
S.index <- sample(1:nrow(data_estratificada2),
                       nrow(data_estratificada2) * 0.7,
                       replace = FALSE)

### ENTRENAMIENTO
TR.Data <- data_estratificada2[S.index, c(predictors, "MentHlth"), drop = FALSE]
TS.data <- data_estratificada2[-S.index, c(predictors, "MentHlth"), drop = FALSE]

ins_model <- lm(MentHlth ~ ., data = TR.Data)
summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "cv", number = 10 )
model <- train(MentHlth ~ ., data = TR.Data, method = "lm",
               trControl = train.control)
```

```
# Resumir los resultados
print(model)
```

Primera predicción

```


#### Segunda

predictors_to_remove <- c("BMI", "HeartDiseaseorAttack", "Stroke", "PhysActivity", "CholCheck")

TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]

ins_model <- lm(MentHlth ~ ., data = TR.Data2)
summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "cv", number = 5)
model <- train(MentHlth ~ ., data = TR.Data2, method = "lm",
               trControl = train.control)


# Resumir los resultados
print(model)
```

Segunda predicción

```


#### Tercera
predictors_to_remove <- c("Diabetes_012", "HighBP", "HighChol", "Veggies", "Education")

TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove)]

ins_model <- lm(MentHlth ~ ., data = TR.Data3)
summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(MentHlth ~ ., data = TR.Data3, method = "lm",
               trControl = train.control)

# Resumir los resultados
print(model)
```

Tercera predicción

Modelo de regresión lineal PhysHlth

```
#### Modelo de regresión lineal PhysHlth #################################################################
set.seed(1)
data_estratificada3 <- data[sample(nrow(data), 3000), ]

predictors <- colnames(data_estratificada2)[-17]
S.index <- sample(1:nrow(data_estratificada3),
                        nrow(data_estratificada3) * 0.7,
                        replace = FALSE)

TR.Data <- data_estratificada2[S.index, c(predictors, "PhysHlth"), drop = FALSE]
TS.data <- data_estratificada2[-S.index, c(predictors, "PhysHlth"), drop = FALSE]

ins_model <- lm(PhysHlth ~ ., data = TR.Data)
summary(ins_model)

#Entrenar el modelo
train.control <- trainControl(method = "cv", number = 10 )
model <- train(PhysHlth ~ ., data = TR.Data, method = "lm",
               trControl = train.control)
# # Resumir los resultados
print(model)
```

Primera predicción

```
#### Segunda

predictors_to_remove <- c("Sex", "Diabetes_012", "Education", "CholCheck", "Smoker")

TR.Data2 <- TR.Data[, !(names(TR.Data) %in% predictors_to_remove)]
TS.data2 <- TS.data[, !(names(TS.data) %in% predictors_to_remove)]

ins_model <- lm(PhysHlth ~ ., data = TR.Data2)
summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "cv", number = 5)
model <- train(PhysHlth ~ ., data = TR.Data2, method = "lm",
               trControl = train.control)
# Resumir los resultados
print(model)
```

Segunda predicción

```
#### Tercera

predictors_to_remove <- c("BMI", "HeartDiseaseorAttack", "PhysActivity", "Veggies", "Stroke")

TR.Data3 <- TR.Data2[, !(names(TR.Data2) %in% predictors_to_remove)]
TS.data3 <- TS.data2[, !(names(TS.data2) %in% predictors_to_remove)]

ins_model <- lm(PhysHlth ~ ., data = TR.Data3)
summary(ins_model)

# Entrenar el modelo
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(PhysHlth ~ ., data = TR.Data3, method = "lm",
               trControl = train.control)
# Resumir los resultados
print(model)
```

Tercera predicción