

Deliverable_2

Julian Abello Orozco

2023-10-08

KNN, Linear regression, and multilinear regression, In a diabetes_012 Dataset

Part 1: Data exploration and data wrangling

In this R Markdown document i will use a data set containing 22 variables that contains 253680 objects. With this dataset I will show how to apply data analysis, Knn, linear and multilinear regression.

In order to start, it is necessary to load the data set into the program as shown in the following section of the code.

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

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

After loading our data set we must inspect and analyze the information contained in this file. In the following image we can see the variables and brief information about their content.



Summary of the variables in the "diabetes_012_health_indicators_BRFSS2015.csv" dataset

Diabetes_012: 0 = no diabetes 1 = prediabetes 2 = diabetes

HighBP: 0 = no high BP 1 = high BP

HighChol: 0 = no high cholesterol 1 = high cholesterol

CholCheck: 0 = no cholesterol check in 5 years 1 = yes cholesterol check in 5 years

BMI: Body Mass Index

Smoker: Have you smoked at least 100 cigarettes in your entire life? [Note: 5 packs = 100 cigarettes] 0 = no 1 = yes

Stroke: (Ever told) you had a stroke. 0 = no 1 = yes

HeartDiseaseorAttack: coronary heart disease (CHD) or myocardial infarction (MI) 0 = no 1 = yes

PhysActivity: physical activity in past 30 days - not including job 0 = no 1 = yes

Fruits: Consume Fruit 1 or more times per day 0 = no 1 = yes

Veggies: Consume Vegetables 1 or more times per day 0 = no 1 = yes

HvyAlcoholConsump: (adult men >=14 drinks per week and adult women >=7 drinks per week) 0 = no 1 = yes

AnyHealthcare: Have any kind of health care coverage, including health insurance, prepaid plans such as HMO, etc. 0 = no 1 = yes

NoDocbcCost: Was there a time in the past 12 months when you needed to see a doctor but could not because of cost? 0 = no 1 = yes

GenHlth: Would you say that in general your health is: scale 1-5 1 = excellent 2 = very good 3 = good 4 = fair 5 = poor

MentHlth: days of poor mental health scale 1-30 days

PhysHlth: physical illness or injury days in past 30 days scale 1-30

DiffWalk: Do you have serious difficulty walking or climbing stairs? 0 = no 1 = yes

Sex: 0 = female 1 = male

Age: 13-level age category (_AGE5YR see codebook) 1 = 18-24 9 = 60-64 13 = 80 or older

Education: Education level (EDUCA see codebook) scale 1-6 1 = Never attended school or only kindergarten 2 = elementary etc.

Income: Income scale (INCOME2 see codebook) scale 1-8 1 = less than \$10,000 5 = less than \$35,000 8 = \$75,000 or more

Figure 1: Characteristics of the data set variables

later using the function `psych` we can extract a statistical analysis of the 22 variables contained in the dataset, which include the mean, standard deviation, minimum and maximum range, among others.

Finally, using the `mutate` function we are going to transform all the data that are not "0" in the variable Diabetes_012, then we will show in a small table how many data were classified as "0" or "1" in this variable

of our set of data

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

```
Conteo_Diabetes
```

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

Part 2: KNN

KNN DIABETES PREDICTION

First Prediction In this part of the document we will use the KNN predictive method, for this we will use 3 different variables to achieve the predictions. First, through a stratified sample, we will take approximately 1% of the data to train our models.

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

```
Conteo_ss_Diabetes
```

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

At this point we will find the appropriate number of “K” and we will train the Knn model to predict Diabetes

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

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

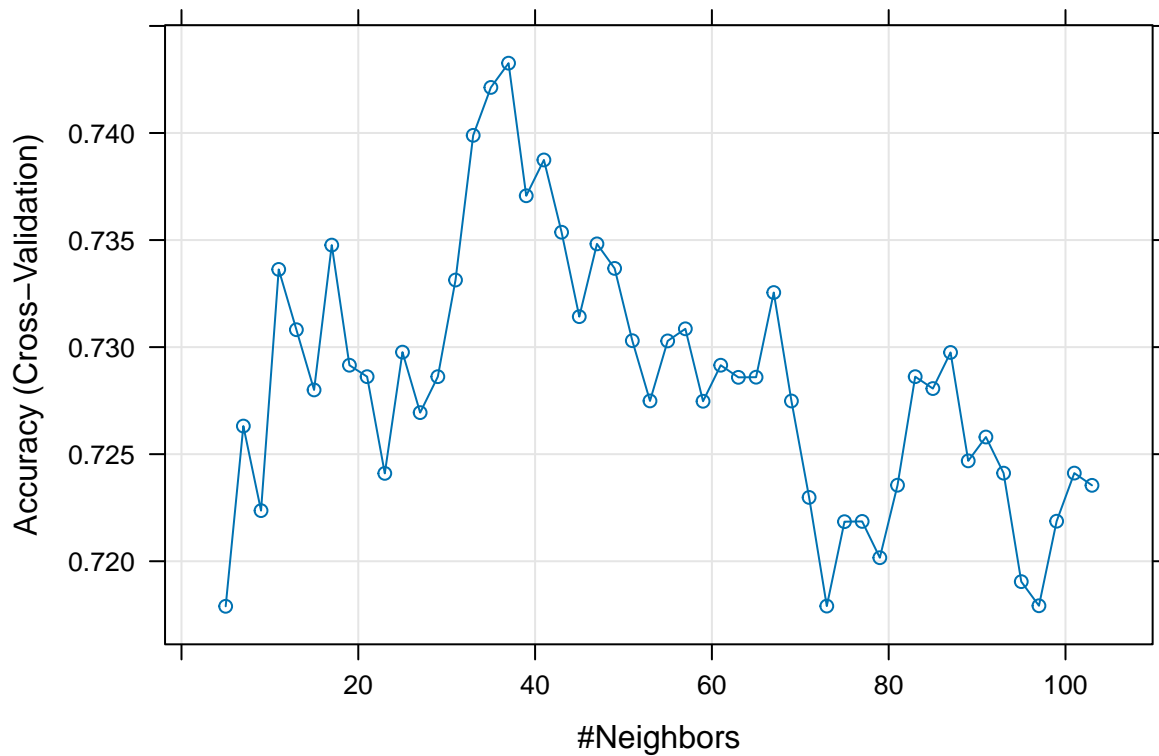
predictors <- c("HighBP", "HighChol", "CholCheck", "BMI", "Smoker", "Stroke", "HeartDiseaseorAttack", "I")

train.data <- ss_diabetes_knn[sample.index, c(predictors, "Diabetes_012"), drop = FALSE]
test.data <- ss_diabetes_knn[-sample.index, c(predictors, "Diabetes_012"), drop = FALSE]

train.data$Diabetes_012 <- factor(train.data$Diabetes_012)
test.data$Diabetes_012 <- factor(test.data$Diabetes_012)

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

```
plot(knnFit)
```



```
# Make predictions
knnPredict <- predict(knnFit, newdata = test.data)

# Creates the confusion matrix
confusionMatrix(data = knnPredict, reference = test.data$Diabetes_012)

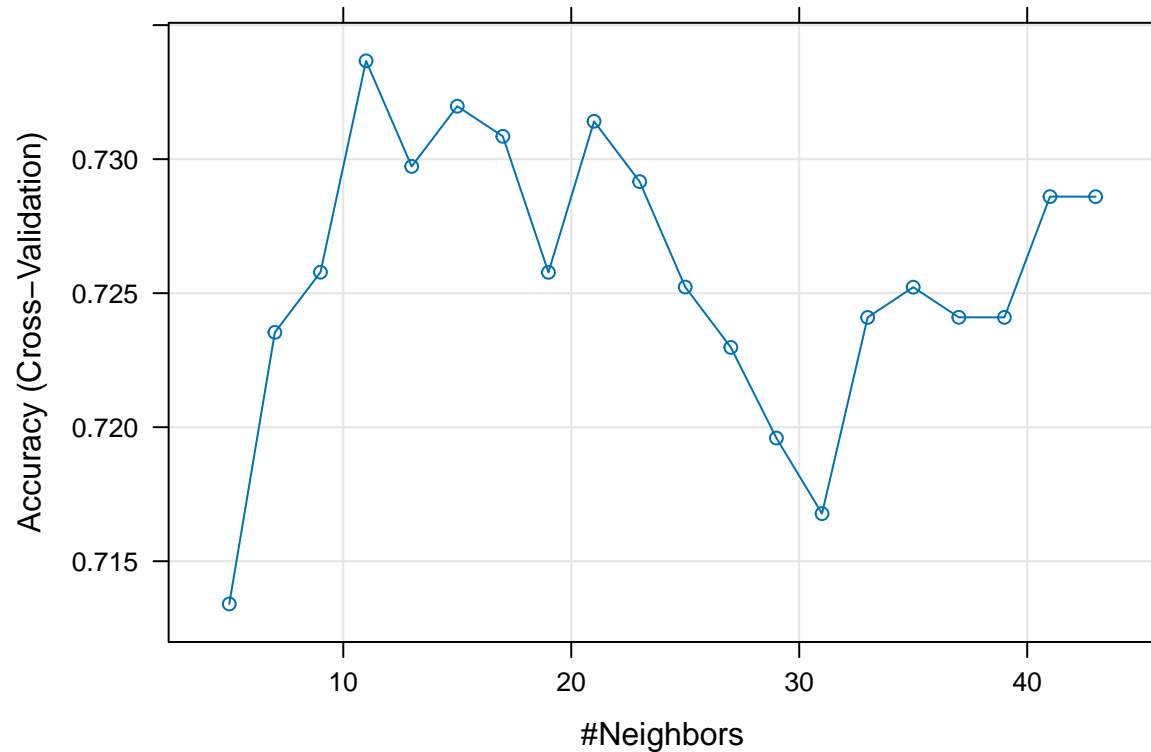
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0    1
##           0 258 108
##           1 120 276
##
##               Accuracy : 0.7008
##               95% CI : (0.6669, 0.7331)
##       No Information Rate : 0.5039
##       P-Value [Acc > NIR] : <2e-16
##
##               Kappa : 0.4014
##
##  Mcnemar's Test P-Value : 0.4663
##
##               Sensitivity : 0.6825
##               Specificity : 0.7188
```

```
##          Pos Pred Value : 0.7049
##          Neg Pred Value : 0.6970
##          Prevalence : 0.4961
##          Detection Rate : 0.3386
##          Detection Prevalence : 0.4803
##          Balanced Accuracy : 0.7006
##
##          'Positive' Class : 0
##
```

```
predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Education", "Income")
train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(Diabetes_012 ~ .
  , data = train.data2
  , method = "knn", trControl = ctrl
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 20)

plot(knnFit2)
```



Second Prediction

```

# Make predictions
knnPredict2 <- predict(knnFit2, newdata = test.data2)

# Creates the confusion matrix
confusionMatrix(data = knnPredict2, reference = test.data2$Diabetes_012)

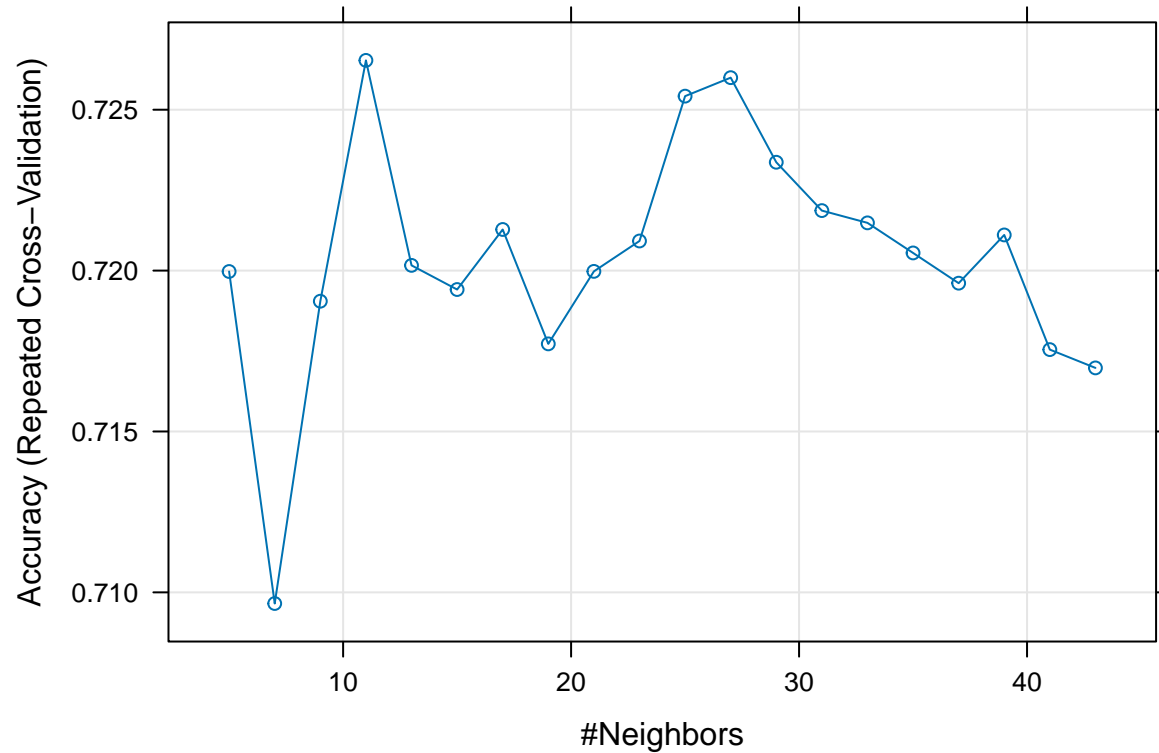
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##           0 260  99
##           1 118 285
##
##           Accuracy : 0.7152
##           95% CI : (0.6817, 0.747)
##       No Information Rate : 0.5039
##       P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.4302
##
##  Mcnemar's Test P-Value : 0.2217
##
##           Sensitivity : 0.6878
##           Specificity : 0.7422
##       Pos Pred Value : 0.7242
##       Neg Pred Value : 0.7072
##           Prevalence : 0.4961
##       Detection Rate : 0.3412
##   Detection Prevalence : 0.4711
##       Balanced Accuracy : 0.7150
##
##       'Positive' Class : 0
##

predictors_to_remove2 <- c("Choc1Check", "MentHlth", "PhysHlth", "Fruits", "Veggies")
train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove2)]
test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove2)]

ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(Diabetes_012 ~ .
  , data = train.data3
  , method = "knn", trControl = ctrl2
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 20)

plot(knnFit3)

```



Third Prediction

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

# Creates the confusion matrix
confusionMatrix(data = knnPredict3, reference = test.data3$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0    1
##      0 252  91
##      1 126 293
##
##           Accuracy : 0.7152
##           95% CI : (0.6817, 0.747)
##      No Information Rate : 0.5039
##      P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.43
##
##      Mcnemar's Test P-Value : 0.021
##
##           Sensitivity : 0.6667
##           Specificity : 0.7630
##      Pos Pred Value : 0.7347
##      Neg Pred Value : 0.6993
##           Prevalence : 0.4961
```

```
##          Detection Rate : 0.3307
##    Detection Prevalence : 0.4501
##      Balanced Accuracy : 0.7148
##
##      'Positive' Class : 0
##
```

KNN HeartDiseaseorAttack Prediction

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

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

# Original data
train.data <- ss_heartDiseaseorAttack[sample.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]
test.data <- ss_heartDiseaseorAttack[-sample.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]

train.data$HeartDiseaseorAttack <- factor(train.data$HeartDiseaseorAttack)
test.data$HeartDiseaseorAttack <- factor(test.data$HeartDiseaseorAttack)

# Train the k-NN model
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(HeartDiseaseorAttack ~ .
  , data = train.data
  , method = "knn", trControl = ctrl
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

# Make predictions
knnPredict <- predict(knnFit, newdata = test.data)

# Creates the confusion matrix
# Original data
train.data <- ss_heartDiseaseorAttack[sample.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]
test.data <- ss_heartDiseaseorAttack[-sample.index, c(predictors, "HeartDiseaseorAttack"), drop = FALSE]

train.data$HeartDiseaseorAttack <- factor(train.data$HeartDiseaseorAttack)
test.data$HeartDiseaseorAttack <- factor(test.data$HeartDiseaseorAttack)

# Train the k-NN model
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(HeartDiseaseorAttack ~ .
  , data = train.data
  , method = "knn", trControl = ctrl
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

# Make predictions
knnPredict <- predict(knnFit, newdata = test.data)
```

```
# Creates the confusion matrix
confusionMatrix(data = knnPredict, reference = test.data$HeartDiseaseorAttack)
```

First Prediction

```
### second model

predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Education", "Income")
train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

# Train the k-NN model
ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(HeartDiseaseorAttack ~ .
  , data = train.data2
  , method = "knn", trControl = ctrl
  , preprocess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

# Make predictions
knnPredict2 <- predict(knnFit2, newdata = test.data2)

# Creates the confusion matrix
confusionMatrix(data = knnPredict2, reference = test.data2$HeartDiseaseorAttack)
```

Second Prediction

```
### Third Model

predictors_to_remove2 <- c("ChoclCheck", "MentHlth", "HvyAlcoholConsump", "Fruits", "Veggies")
train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove2)]
test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove2)]

# Train the k-NN model
ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(HeartDiseaseorAttack ~ .
  , data = train.data3
  , method = "knn", trControl = ctrl2
  , preprocess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

# Make predictions
knnPredict3 <- predict(knnFit3, newdata = test.data3)

# Creates the confusion matrix
confusionMatrix(data = knnPredict3, reference = test.data3$HeartDiseaseorAttack)
```

Third Prediction

KNN Find Sex Prediction


```

####KNN Models and Experiments to Find Sex #####

## selection of 1500 samples of each factor of the dataset#
set.seed(123)
ss_sex <- ss_diabetes %>%
  group_by(Sex) %>%
  sample_n(1269, replace = TRUE) %>%
  ungroup()

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

# Original data
train.data <- ss_sex[sample.index, c(predictors, "Sex"), drop = FALSE]
test.data <- ss_sex[-sample.index, c(predictors, "Sex"), drop = FALSE]

train.data$Sex <- factor(train.data$Sex)
test.data$Sex <- factor(test.data$Sex)

# Train the k-NN model
ctrl <- trainControl(method = "cv", p = 0.7)
knnFit <- train(Sex ~ .
  , data = train.data
  , method = "knn", trControl = ctrl
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

# Make predictions
knnPredict <- predict(knnFit, newdata = test.data)

# Creates the confusion matrix
confusionMatrix(data = knnPredict, reference = test.data$Sex)

```

First Prediction

```

# second model

predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "DiffWalk", "Age", "PhysActivity")
train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

# Train the k-NN model
ctrl <- trainControl(method = "cv", number = 5)
knnFit2 <- train(Sex ~ .
  , data = train.data2
  , method = "knn", trControl = ctrl
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

#Make predictions
knnPredict2 <- predict(knnFit2, newdata = test.data2)

```

```
# Creates the confusion matrix
confusionMatrix(data = knnPredict2, reference = test.data2$Sex)
```

Second Prediction

```
### Third Model

predictors_to_remove2 <- c("Choc1Check", "MentHlth", "HvyAlcoholConsump", "Fruits", "Veggies")
train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove2)]
test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove2)]

# Train the k-NN model
ctrl2 <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
knnFit3 <- train(Sex ~ .
  , data = train.data3
  , method = "knn", trControl = ctrl2
  , preProcess = c("range") # c("center", "scale") for z-score
  , tuneLength = 50)

#Make predictions
knnPredict3 <- predict(knnFit3, newdata = test.data3)

# Creates the confusion matrix
confusionMatrix(data = knnPredict3, reference = test.data3$Sex)
```

Third Prediction

Part 3: Linear regression model BM

```
### Linear regression model BMI #####
folder <- dirname(rstudioapi :: getSourceEditorContext())$path

parentFolder <- dirname (folder)
data <-
  read.csv(paste0(parentFolder, "/dataset/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]
sample.index <- sample(1:nrow(data_estratificada2),
  nrow(data_estratificada2) * 0.7,
  replace = FALSE)

train.data <- data_estratificada2[sample.index, c(predictors, "BMI"), drop = FALSE]
test.data <- data_estratificada2[-sample.index, c(predictors, "BMI"), drop = FALSE]

ins_model <- lm(BMI ~ ., data = train.data)
```

```
summary(ins_model)
```

First Prediction

```
##
## Call:
## lm(formula = BMI ~ ., data = train.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

# Train the model
train.control <- trainControl(method = "cv", number = 10 )
model <- train(BMI ~ ., data = train.data, method = "lm",
               trControl = train.control)

# Summarize the results
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
```

```
#### second

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

train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

ins_model <- lm(BMI ~ ., data = train.data2)

summary(ins_model)

# Train the model
train.control <- trainControl(method = "cv", number = 5)
model <- train(BMI ~ ., data = train.data2, method = "lm",
               trControl = train.control)

# Summarize the results
print(model)
```

Second Prediction

```
#### Third

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

train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove)]
test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove)]

ins_model <- lm(BMI ~ ., data = train.data3)

summary(ins_model)

# Train the model
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(BMI ~ ., data = train.data3, method = "lm",
               trControl = train.control)

# Summarize the results
print(model)
```

Third Prediction

Linear regression model MentHlth

```

### Linear regression model MentHlth #####

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

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

### ENTRENAMIENTO
train.data <- data_estratificada2[sample.index, c(predictors, "MentHlth"), drop = FALSE]
test.data <- data_estratificada2[-sample.index, c(predictors, "MentHlth"), drop = FALSE]

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

# Train the model
train.control <- trainControl(method = "cv", number = 10 )
model <- train(MentHlth ~ ., data = train.data, method = "lm",
              trControl = train.control)

# Summarize the results
print(model)

```

First Prediction

```

###Second

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

train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

ins_model <- lm(MentHlth ~ ., data = train.data2)
summary(ins_model)

# Train the model
train.control <- trainControl(method = "cv", number = 5)
model <- train(MentHlth ~ ., data = train.data2, method = "lm",
              trControl = train.control)

# Summarize the results
print(model)

```

Second Prediction

```

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

train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove)]

```

```

test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove)]

ins_model <- lm(MentHlth ~ ., data = train.data3)
summary(ins_model)

# Train the model
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(MentHlth ~ ., data = train.data3, method = "lm",
               trControl = train.control)
# Summarize the results
print(model)

```

Third Prediction

Linear regression model PhysHlth

```

#### Linear regression model PhysHlth #####
set.seed(1)
data_estratificada3 <- data[sample(nrow(data), 3000), ]

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

train.data <- data_estratificada2[sample.index, c(predictors, "PhysHlth"), drop = FALSE]
test.data <- data_estratificada2[-sample.index, c(predictors, "PhysHlth"), drop = FALSE]

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

# Train the model
train.control <- trainControl(method = "cv", number = 10 )
model <- train(PhysHlth ~ ., data = train.data, method = "lm",
               trControl = train.control)
# Summarize the results
print(model)

```

First Prediction

```

###Second

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

train.data2 <- train.data[, !(names(train.data) %in% predictors_to_remove)]
test.data2 <- test.data[, !(names(test.data) %in% predictors_to_remove)]

ins_model <- lm(PhysHlth ~ ., data = train.data2)
summary(ins_model)

# Train the model
train.control <- trainControl(method = "cv", number = 5)

```

```

model <- train(PhysHlth ~ ., data = train.data2, method = "lm",
              trControl = train.control)
# Summarize the results
print(model)

```

Second Prediction

```

#### Third

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

train.data3 <- train.data2[, !(names(train.data2) %in% predictors_to_remove)]
test.data3 <- test.data2[, !(names(test.data2) %in% predictors_to_remove)]

ins_model <- lm(PhysHlth ~ ., data = train.data3)
summary(ins_model)

# Train the model
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)
model <- train(PhysHlth ~ ., data = train.data3, method = "lm",
              trControl = train.control)
# Summarize the results
print(model)

```

Third Prediction