KNN, Linear regression, and multilinear regressio

Juan Sebastian Gonzalez

2023-10-09

KNN, LINEAR REGRESSION AND MULTILINEAR REGRESSION IN A DIABETES DATASET

PART 1: Data Exploration

In this R Markdown file, I will work with a dataset comprising 22 variables and a total of 253,680 records. Through this dataset, I will demonstrate the application of data analysis techniques, including K-nearest neighbors (Knn) and various forms of regression such as linear and multilinear regression.

To begin, it is essential to import the dataset into the program, as illustrated in the subsequent section of the code.

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

parentFolder <- dirname (folder)
data_set_dia <-
  read.csv(paste0(parentFolder,"/DATASET/diabetes_012.csv"))
```

After loading our data set we must inspect and analyze the information contained in this file. This dataset encompasses a wide range of variables pertaining to individuals' health and various factors that could potentially impact the occurrence of diabetes. It incorporates information related to health indicators, lifestyle choices, and other relevant parameters, offering valuable insights into the complex interplay of factors contributing to diabetes. Subsequently, by employing the 'psych' function, we can extract a comprehensive statistical analysis of the 22 variables encompassed within the dataset. This analysis comprises metrics such as the mean, standard deviation, minimum, maximum, and various others.

In the end, we will utilize the 'mutate' function to convert all non-"= 0" values within the 'Diabetes_012' variable. Subsequently, we will present a concise table that displays the count of data classified as either "0" or "1" within this specific variable within our dataset.

```
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 section of the document, we will apply the K-nearest neighbors (KNN) predictive technique. To achieve our predictions, we will select three distinct variables. Initially, we will create a stratified sample, extracting roughly 1% of the dataset to serve as the training data for our models.

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

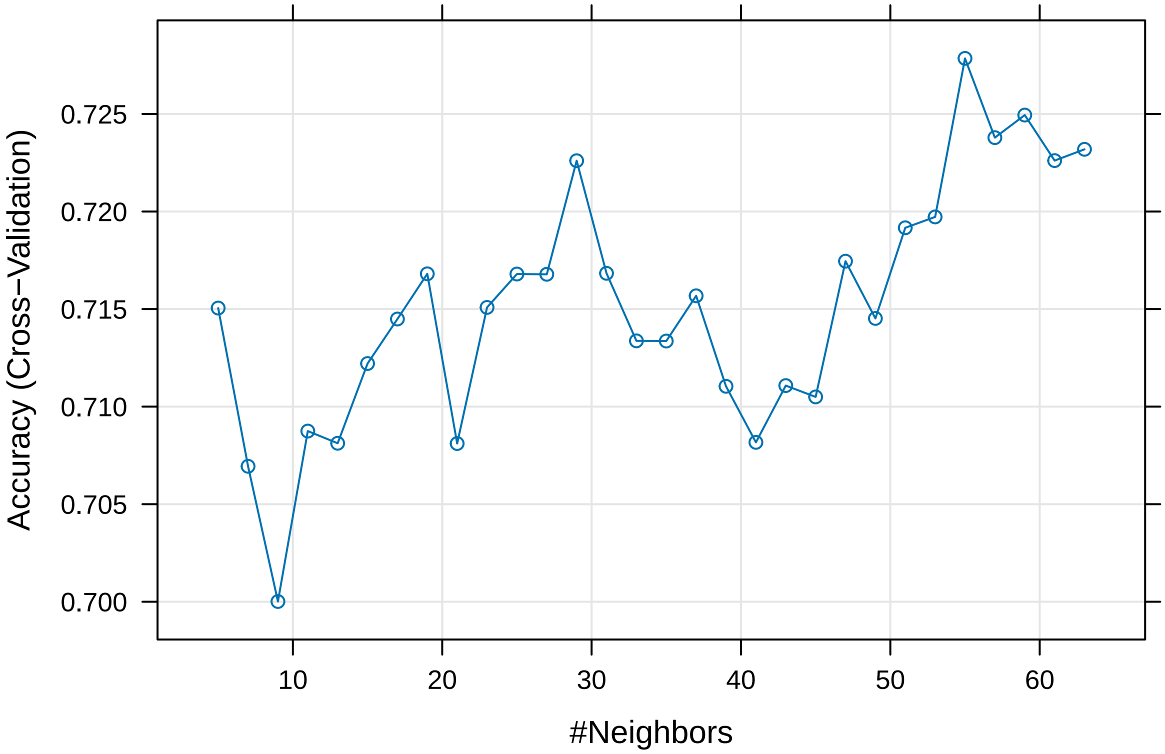
```
Conteo_ss_Diabetes
```

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

Now, in this stage, we will determine the optimal value for "K" and proceed to train the K-nearest neighbors (KNN) model for predicting diabetes.

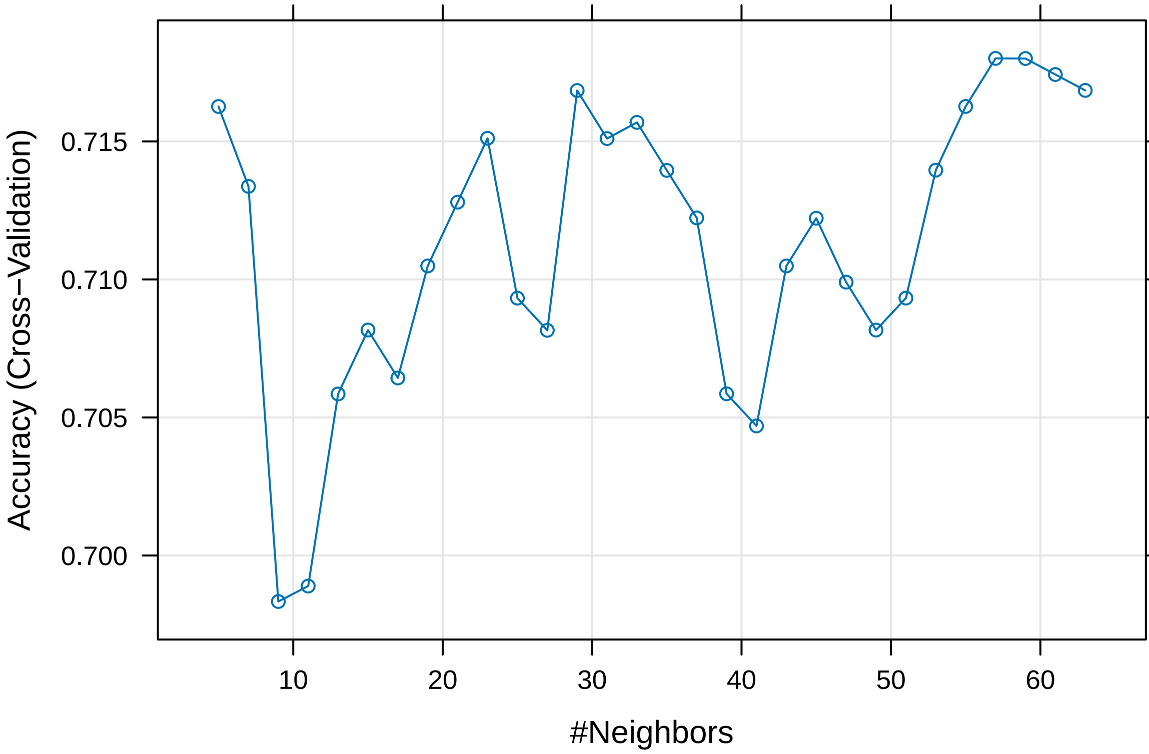
```
set.seed(24)
ss_diabetes_knn <- ss_diabetes %>%
  group_by(Diabetes_012) %>%
  sample_n(1234, replace = TRUE) %>%
  ungroup()


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", "

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 = 0.5)
knnFit <- train(Diabetes_012 ~ .
                , data = train.data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 30)

plot(knnFit)
```



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

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

```r


predictors_to_remove <- c("Smoker", "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 = 30)

plot(knnFit2)
```

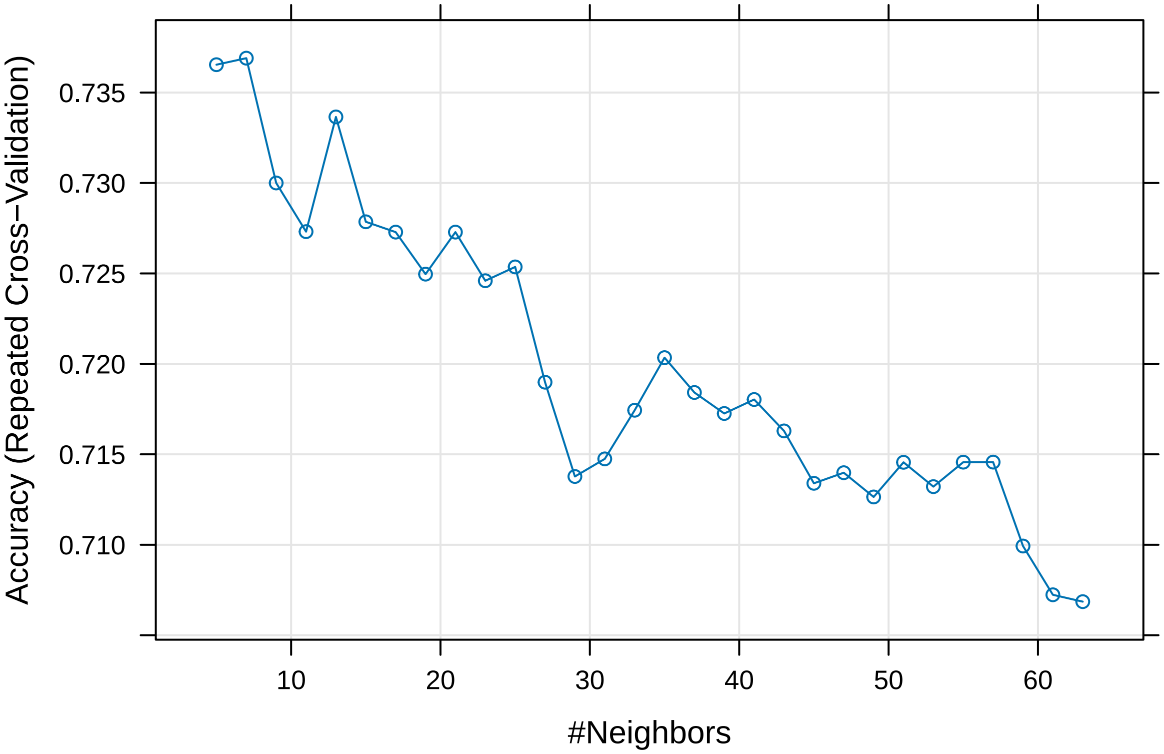


Second Prediction

```r
knnPredict2 <- predict(knnFit2, newdata = test.data2)

confusionMatrix(data = knnPredict2, reference = test.data2$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 255 115
##          1  84 287
##
##                Accuracy : 0.7314
##                  95% CI : (0.698, 0.7631)
##     No Information Rate : 0.5425
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.4628
##
##  Mcnemar's Test P-Value : 0.03345
##
##             Sensitivity : 0.7522
##             Specificity : 0.7139
##          Pos Pred Value : 0.6892
##          Neg Pred Value : 0.7736
##              Prevalence : 0.4575
##          Detection Rate : 0.3441
##    Detection Prevalence : 0.4993
##       Balanced Accuracy : 0.7331
##
##        'Positive' Class : 0
##
```

```
predictors_to_remove2 <- c("ChoclCheck", "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 = 30)

plot(knnFit3)
```



Third Prediction

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

confusionMatrix(data = knnPredict3, reference = test.data3$Diabetes_012)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 249 110
##          1  90 292
##
##                Accuracy : 0.7301
##                  95% CI : (0.6966, 0.7618)
##     No Information Rate : 0.5425
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.4588
##
##  Mcnemar’s Test P-Value : 0.1791
##
##             Sensitivity : 0.7345
##             Specificity : 0.7264
##          Pos Pred Value : 0.6936
##          Neg Pred Value : 0.7644
##              Prevalence : 0.4575
##          Detection Rate : 0.3360
##    Detection Prevalence : 0.4845
##       Balanced Accuracy : 0.7304
##
##        ’Positive’ Class : 0
##
```

KNN Heart Disease Prediction

```
set.seed(24)
ss_heartDiseaseorAttack <- ss_diabetes %>%
  group_by(HeartDiseaseorAttack) %>%
  sample_n(1234, replace = TRUE) %>%
  ungroup()

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

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)

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 = 30)
```

```
knnPredict <- predict(knnFit, newdata = test.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)

ctrl <- trainControl(method = "cv", p = 0.5)
knnFit <- train(HeartDiseaseorAttack ~ .
                , data = train.data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 30)


knnPredict <- predict(knnFit, newdata = test.data)
confusionMatrix(data = knnPredict, reference = test.data$HeartDiseaseorAttack)
```

First Prediction

```
predictors_to_remove <- c("AnyHealthcare", "HighChol", "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 = 10)
knnFit2 <- train(HeartDiseaseorAttack ~ .
                , data = train.data2
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 30)

knnPredict2 <- predict(knnFit2, newdata = test.data2)
confusionMatrix(data = knnPredict2, reference = test.data2$HeartDiseaseorAttack)
```

Second Prediction

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 221  79
##          1 118 323
##
##                Accuracy : 0.7341
##                  95% CI : (0.7008, 0.7656)
##     No Information Rate : 0.5425
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                   Kappa : 0.4595
##
##  Mcnemar’s Test P-Value : 0.006781
##
##             Sensitivity : 0.6519
##             Specificity : 0.8035
##          Pos Pred Value : 0.7367
##          Neg Pred Value : 0.7324
##              Prevalence : 0.4575
##          Detection Rate : 0.2982
##    Detection Prevalence : 0.4049
##       Balanced Accuracy : 0.7277
##
##        ’Positive’ Class : 0
##
```

```
predictors_to_remove2 <- c("ChoclCheck", "MentHlth","HvyAlcoholConsump", "Fruits", "HighChol")
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 = 15, repeats = 5)
knnFit3 <- train(HeartDiseaseorAttack ~ .
                 , data = train.data3
                 , method = "knn", trControl = ctrl2
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 30)

knnPredict3 <- predict(knnFit3, newdata = test.data3)
confusionMatrix(data = knnPredict3, reference = test.data3$HeartDiseaseorAttack)
```

Third Prediction

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 229  86
##          1 110 316
##
##                Accuracy : 0.7355
##                  95% CI : (0.7022, 0.7669)
##     No Information Rate : 0.5425
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.4642
##
##  Mcnemar’s Test P-Value : 0.1004
##
##             Sensitivity : 0.6755
##             Specificity : 0.7861
##          Pos Pred Value : 0.7270
##          Neg Pred Value : 0.7418
##              Prevalence : 0.4575
##          Detection Rate : 0.3090
##    Detection Prevalence : 0.4251
##       Balanced Accuracy : 0.7308
##
##        'Positive' Class : 0
##
```

##KNN Find Sex Prediction In the upcoming analysis, we will utilize the K-nearest neighbors (KNN) algorithm to perform sex prediction. #### First Prediction

```r
set.seed(24)
ss_sex <- ss_diabetes %>%
  group_by(Sex) %>%
  sample_n(1234, replace = TRUE) %>%
  ungroup()

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

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)

ctrl <- trainControl(method = "cv", p = 0.5)
knnFit <- train(Sex ~ .
                , data = train.data
                , method = "knn", trControl = ctrl
                , preProcess = c("range") # c("center", "scale") for z-score
                , tuneLength = 30)


knnPredict <- predict(knnFit, newdata = test.data)
confusionMatrix(data = knnPredict, reference = test.data$Sex)
```

```r


predictors_to_remove <- c("HeartDiseaseorAttack" , "NoDocbcCost", "Smoker", "Age", "PhysActivity")
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 = 10)
knnFit2 <- train(Sex ~ .
                 , data = train.data2
                 , method = "knn", trControl = ctrl
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 30)

knnPredict2 <- predict(knnFit2, newdata = test.data2)
confusionMatrix(data = knnPredict2, reference = test.data2$Sex)
```

Second Prediction

```
predictors_to_remove2 <- c("Smoker", "MentHlth","HvyAlcoholConsump", "PhysActivity", "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(Sex ~ .
                 , data = train.data3
                 , method = "knn", trControl = ctrl2
                 , preProcess = c("range") # c("center", "scale") for z-score
                 , tuneLength = 50)

knnPredict3 <- predict(knnFit3, newdata = test.data3)
confusionMatrix(data = knnPredict3, reference = test.data3$Sex)
```

Third Prediction

PART 3: Linear Regression BMI

```
folder <- dirname(rstudioapi :: getSourceEditorContext()$path)
parentFolder <- dirname (folder)
data <-
  read.csv(paste0(parentFolder,"/DATASET/diabetes_012.csv"))
data$Diabetes_012 <- ifelse(data$Diabetes_012 == 0, 0, 1)
set.seed(24)
data_estratificada2 <- data[sample(nrow(data), 3000), ]

predictors <- colnames(data_estratificada2)[-5]
sample.index <- sample(1:nrow(data_estratificada2),
                       nrow(data_estratificada2) * 0.5,
                       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
## -13.786  -3.733  -0.814   2.788  51.393
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          27.465265   1.586521  17.312  < 2e-16 ***
## Diabetes_012          2.967957   0.515737   5.755 1.05e-08 ***
## HighBP                2.429072   0.386458   6.285 4.29e-10 ***
## HighChol             -0.679890   0.369564  -1.840  0.06601 .
## CholCheck             1.519668   0.915854   1.659  0.09727 .
## Smoker               -0.736408   0.344535  -2.137  0.03273 *
## Stroke               -0.020235   0.898671  -0.023  0.98204
## HeartDiseaseorAttack -1.613690   0.622759  -2.591  0.00966 **
## PhysActivity         -2.082861   0.417878  -4.984 6.95e-07 ***
## Fruits               -0.761570   0.356482  -2.136  0.03281 *
## Veggies               0.044744   0.437292   0.102  0.91852
## HvyAlcoholConsump    -1.502100   0.693613  -2.166  0.03050 *
## AnyHealthcare        -0.020109   0.809750  -0.025  0.98019
## NoDocbcCost           0.498714   0.608165   0.820  0.41233
## GenHlth               1.015872   0.202186   5.024 5.66e-07 ***
## MentHlth             -0.003684   0.025742  -0.143  0.88621
## PhysHlth             -0.041126   0.024089  -1.707  0.08798 .
## DiffWalk              1.655782   0.546664   3.029  0.00250 **
## Sex                   0.715324   0.347145   2.061  0.03952 *
## Age                  -0.248228   0.062565  -3.968 7.61e-05 ***
## Education            -0.161352   0.199594  -0.808  0.41899
## Income                0.126681   0.094950   1.334  0.18235
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 6.378 on 1478 degrees of freedom
## Multiple R-squared:  0.1557, Adjusted R-squared:  0.1437
## F-statistic: 12.98 on 21 and 1478 DF,  p-value: < 2.2e-16
```

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

```
## Linear Regression
##
## 1500 samples
##   21 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1349, 1351, 1350, 1349, 1349, 1351, ...
## Resampling results:
##
##   RMSE      Rsquared  MAE
##   6.401683  0.141557  4.480632
##
## Tuning parameter ’intercept’ was held constant at a value of TRUE
```

```
predictors_to_remove <- c("AnyHealthcare", "PhysActivity", "MentHlth", "Education", "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(BMI ~ ., data = train.data2)

summary(ins_model)

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

print(model)
```

Second Prediction

```
predictors_to_remove <- c("ChoclCheck", "MentHlth","Smoker", "PhysActivity", "Veggies")

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

Third Prediction

Linear Regression MentHlth

```
set.seed(24)
data_estratificada2 <- data[sample(nrow(data), 3000), ]

predictors <- colnames(data_estratificada2)[-16]
sample.index <- sample(1:nrow(data_estratificada2),
                       nrow(data_estratificada2) * 0.5,
                       replace = FALSE)

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

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

summary(ins_model)

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

First Prediction

```r
predictors_to_remove <- c("AnyHealthcare", "NoDocbcCost", "Education","Smoker", "Income")
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.control <- trainControl(method = "cv", number = 5)
model <- train(MentHlth ~ ., data = train.data2, method = "lm",
               trControl = train.control)
print(model)
```

Second Prediction

```r
predictors_to_remove <- c("AnyHealthcare", "CholCheck", "NoDocbcCost", "Education", "Sex")

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

Third Prediction In conclusion, this comprehensive programming effort encompassed a series of predictive models and data analyses applied to a diverse dataset. We explored various machine learning techniques, including K-Nearest Neighbors (KNN) and Linear Regression, to predict health-related outcomes such as diabetes, heart disease, and other health indicators. Through iterative feature selection and model refinement, we gained valuable insights into the key factors influencing these health outcomes. These models provide a foundation for understanding and potentially mitigating health risks in populations, demonstrating the potential of data-driven approaches in healthcare and public health research. The code and methodologies presented here can serve as a valuable resource for future research and applications in the field of health data analysis.