Case Study 2 - K Nearest Neighbors

Today's Goal

Welcome to the second case study in our series focusing on the K-Nearest Neighbors(KNN) algorithm. In this case study, we will continue our exploration of addressing the prevalent health concern of obesity through predictive modeling. Our main goal today is to leverage the KNN algorithm to classify individuals into obesity categories based on their unique characteristics.

Dataset Overview

For this case study, we'll continue working with the dataset used in our first case study on neural networks. This dataset contains extensive information on individuals' demographic characteristics, physical attributes, and lifestyle habits. To streamline our analysis, we'll utilize the tidy version of the dataset, where numerical variables have been rounded and a Body Mass Index (BMI) variable has been added.

Before we start our analysis, please download the dataset 'obesity_cleaned.csv' from the GitHub folder 'case study 2'.

The original dataset source: https://www.kaggle.com/datasets/mrsimple07/obesity-prediction/data.

1. Getting Started

A. Initial Setups

(i) To begin, make sure you have the following libraries installed and loaded in your R environment: tidyverse, ggplot2, dplyr, caret and class.

(ii) Set the seed to '123' to ensure the reproducibility of your results.

Recap: Why is it important to set a seed at the beginning of our analysis?

(iii) Now, please load the dataset that you've downloaded into your R environment. Assign this data to an object named 'knn_data'.

B. Data Cleanings

(i) Once again, familiarize yourself with the structure of the dataset.

Hint: You can use functions like `head()` and `glimpse()` to get an overview of the data.

(ii) Generate a statistical summary for the numerical variables in the dataset.

Hint: Try using the function `summary()`.

(iii) What are the minimum, maximum, mean, and median values of sleeping hours for the individuals in the dataset?

(iv) Transform all categorical variables into their factor formats. By doing so, we ensure that R recognizes and treats these variables as categorical in the visualization the modeling tasks.

Hint: You can use the function `as.factor()`.

2. Data Visualizations

(i) Create a bar plot to visualize the distribution of obesity categories among different marital statuses in the dataset. Make sure you generate a plot with clear title and axis label.

Hint: To differentiate between different marital statuses, use distinct colors for the bars to represent different marital status by including `fill = MaritalStatus` within the `aes()` argument.

(ii) Briefly describe the distributions you observed from the generated bar plot.

(iii) Create a scatter plot to visualize the correlation between daily calorie intake and weight, differentiated by gender. Again, make sure you generate a plot with clear title and axis label.

(iv) What can you infer from the scatter plot?

3. Model Building

(i) Before building our KNN models, we need to first scale all numerical variables in our dataset to ensure they all contribute equally to the analysis. Use the `scale()` function to normalize these variables.

(ii) Similar to what we've done in case study 1, convert 'Gender', 'Race', and 'MaritalStatus' into their numeric formats.

(iii) Divide the dataset into 70% training and 30% testing sets.

(iv) Remove the response variable 'ObesityCategory' from the training and testing datasets, and store the resulting datasets into new objects titled 'train_scaled' and 'test_scaled'.

Note: Don't change the original traing data 'train_data' and testing data 'test_data'.

Discussion: Why do we need to remove the respones variable before we apply the KNN algorithm to make predictions on our dataset?

(v) Constructing the KNN Model

- Use the `knn()` function to build the KNN model and make predictions. Recall that our goal is to classify individuals into specific obesity categories.
- Use all available covariates in the dataset.
- Set the number of neighbors k to 1.
- Store the predictions in a variable named test_pred.

(vi) Evaluate the model's performance by generating the confusion matrix for the KNN predictions using the function confusionMatrix().

(vii) What do the accuracy, sensitivity, and specificity rates indicate about the predictive power of the model?

(viii) Use the KNN algorithm to make predictions by setting the number of nearest neighbors to 5, 10, 20, and 100. For each model, generate the corresponding confusion matrices.

(ix) Based on the performance metrics obtained for different values of 'k' in the KNN algorithm, what can you conclude about the impact of 'k' on the model's predictive accuracy?

(x) Discuss the advantages and disadvantages of applying KNN versus Neural Networks for our classification tasks.