**Performance Assessment**

NVM2 TASK 2: Predictive Analysis

Bader Ale

Department of Information Technology, Western Governors University

D209 Data Mining I

September 2023

Table of Contents

[Part I: Research Question 3](#_Toc146915908)

[Part II: Method Justification 3](#_Toc146915909)

[Part III: Data Preparation 4](#_Toc146915910)

[Part IV: Analysis 7](#_Toc146915911)

[Part V: Data Summary and Implications 8](#_Toc146915912)

# Part I: Research Question

For this performance assessment, our research question is: **Given certain patient characteristics, can we classify whether a patient is hypertensive or not**. Using decision trees, the goal for this data analysis is to be able to classify whether a patient is hypertensive or not taking into consideration other patient-specific variables contained in the dataset.

# Part II: Method Justification

Decision trees is a supervised learning algorithm that uses a question-based structure that poses several choices for data. Its’ name is based on the tree-like structure that it forms while creating the decision forks. Decision trees can be used for both regression and classification problems; the latter will be used for this assessment (Geeks for Geeks, 2023). The decision tree creates several nodes from a primary or “root” node and several decisions are posed by the algorithm. Depending on the answer (Yes/No, or a numerical comparison), it finds its way down the branches to the final answer. For this assessment, the last node (‘terminal node’) will contain the answer to the question, in this case whether the patient has high blood pressure or not (HighBlood\_Yes).

One of the assumptions of the decision tree classifier algorithm is that the output variable (output of last node) should be discrete, meaning a yes/no (Geeks for Geeks, 2023).

In order to run this algorithm, several packages were imported. Ther Pandas library was used to manipulate the dataset appropriately. Seaborn was used to visualize the data. Sci-kit Learn was used as the main library that contained all the machine learning modules. For example, the preprocessing module was used to split the dataset into its train and test data sets and the model\_selection module was used to import the decision trees algorithm.

# Part III: Data Preparation

In order to successfully run the model, some data preparation needs to be performed. This step, usually called data cleaning, involves looking errors, duplicates, and missing values among others, in preparation to pass the data into the model (Crabtree & Nehme, 2023).

One important step in the data preparation step is scaling. Scaling is critical for the model because it levels the scale of the variables so that the algorithm can perform better predictions (SciKit Learn, 2023).

The variables that will be used for the model are shown in the figure below.

A screen shot of a computer code

Description automatically generated

The steps to prepare the data were as follows: first, the dataset was scanned for any null values. This was accomplished by using the .*null().sum()* command. The output shown below shows all the columns with null values; in this case all columns showed zero.

A screenshot of a computer screen

Description automatically generated

Secondly, duplicated values were detected and treated, if any. This was further accomplished using the *.duplicated().value\_counts().* Here, the output shows *False 10000* which indicates that there are no duplicated values.

A screenshot of a computer code

Description automatically generated

The next step was to detect and treat outliers. The dataset was first scaled in order to facilitate visualization of the boxplots for the variables. The *MinMaxScaler* module from Sklearn was used for this and afterwards a boxplot using Seaborn was produced.

A close-up of a computer screen

Description automatically generatedA screenshot of a graph

Description automatically generated

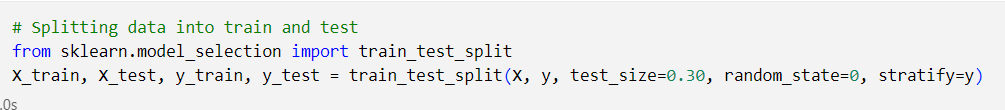
In order to treat the outliers, Z-scores were first calculated using the SciPy library – a new column was created that housed the z-scores for *Vitamin-D* levels and *Income*. All records with a z-score of less than -3 and more than 3 were removed. After this process, the z-score column was dropped as it will not be needed any more for the algorithm (Sharma, 2023).

A computer code on a white background

Description automatically generated

# Part IV: Analysis

Once the data preparation phase is complete, it was then split into training and testing data – the training model, as the name suggests, is used to train the model and the test portion is used to see how well the model performs by comparing it to the actual target values.



The model was fitted on the training data and then using the *.predict()* command on the test data. An accuracy score, using SKLearn’s *accuracy\_score*  was used to determine how well the model performed.

A screenshot of a computer program

Description automatically generated

# Part V: Data Summary and Implications

The model was then scored using accuracy and the mean-squared-error (MSE). The SKLearn library has a *metrics*  module which can compute the accuracy score for the model. It also contains the *mean\_squared\_error* module for computing the MSE.



For our model, the accuracy score was 0.5972 and the MSE was 0.4084. The accuracy score calculates how well, in percentage, the model classified the observations (Zach, 2021). In this case, the model correctly classified the observations ~60% of the time. On the other hand, the MSE, which is the average squared error between the actual values and the predicted value, was ~41%. Although technically there is no general rule for how to interpret the MSE within the context of the specific model, the general notion is the smaller the number the better your model performs (Allwright, 2022). The MSE can be used for future iterations of the model in order to measure how well the model performed using different parameters.

A screenshot of a computer error

Description automatically generated

The model was able to predict whether the patient was hypertensive or not about 60% of the time. It shows that the model does require some tuning in order to perform better – it only was able to guess correctly slightly more that half of the time. This can be remedied by parameter tuning.

One of the limitations of the dataset was that it was not based on real world data. Real world clinical data should include more clinical factors about the patient that more closely correspond to their health rather that their demographics. In order to achieve better results, a bigger dataset comprising of more explanatory variables could help the results – more data means a bigger training set could be used so the model can have better predictive power. Moreover, there are other clinical parameters that can help in predicting hypertensive status – kidney disease (or lack thereof), activity level and other metabolic parameters can provide better real-world value rather than some of the variables shown in the dataset.

Based on the results of the model, a recommended course of action for the real world would be to modify the information being recorded for the dataset. Moreover, more hyperparameter tuning would be necessary with the addition of more clinical factors, as stated above, in order to increase the accuracy score and better the model performance.

Works Cited

Allwright, S. (2022, August 26). *How to interpret MSE (simply explained)*. Retrieved from Stephen Allwright: https://stephenallwright.com/interpret-mse/

Crabtree, M., & Nehme, A. (2023, July). *What is Data Analysis? An Expert Guide With Examples*. Retrieved September 2023, from Datacamp: https://www.datacamp.com/blog/what-is-data-analysis-expert-guide

*Geeks for Geeks*. (2023, August 20). Retrieved September 2023, from Decision Tree: https://www.geeksforgeeks.org/decision-tree/

SciKit Learn. (2023). *6.3. Preprocessing data*. Retrieved September 2023, from https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing

Sharma, R. (2023, May 10). *Detect and Remove the Outliers using Python*. Retrieved September 2023, from Geeks for Geeks: https://www.geeksforgeeks.org/detect-and-remove-the-outliers-using-python/