

**Machine Learning**

**MASTER’S DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS**

Pedro Anastácio, 20180040

December 2022

Table of Contents

[1. Introduction 3](#_Toc122731001)

[2. Initial Data 3](#_Toc122731002)

[3. Data Exploration 3](#_Toc122731003)

[4. Pre-processing 4](#_Toc122731004)

[*4.1 Feature Engineering* 4](#_Toc122731005)

[*4.2 Outlier Handling Solutions* 5](#_Toc122731006)

[*4.3 Feature Selection* 5](#_Toc122731007)

[*4.3.1 Numeric features* 6](#_Toc122731008)

[*4.3.2 Categorical features* 7](#_Toc122731009)

[*4.3.3 Selecting Relevant features* 7](#_Toc122731010)

[*4.4 Feature Encoding* 7](#_Toc122731011)

[5. Modelling 7](#_Toc122731012)

[6. Performance Assessment 9](#_Toc122731013)

[9. Conclusion 10](#_Toc122731014)

[10. References 11](#_Toc122731015)

[11. Appendices 12](#_Toc122731016)

[*11.1 Tables* 12](#_Toc122731017)

[*11.2 Figures* 19](#_Toc122731018)

# 1. Introduction

This report will cover the final assessment in the Machine Learning course of the master's degree Program in Data Science and Advanced Analytics at Nova IMS, Lisbon. The aim of the project was to create a predictive model that would assist the fictional Dr. Smith in identifying people that are more likely to suffer from a newly discovered disease baptized as the Smith Parasite.

To predict if a patient will suffer, or not, from this parasite, we were to impersonate a team of data scientists and conduct an empirical data analysis on a provided small quantity of sociodemographic, health, and behavioural information obtained from the patients.

Throughout this report we will present our findings and thought process behind the decisions that lead us to a possible solution for the task at hand.

# 2. Initial Data

As mentioned in the introduction, we were given data regarding the sociodemographic (Table 1), health (Table 2) and behavioural (Table 3) information of patients. This data was split into 6 files grouped into training and testing: train\_demo.csv, train\_health.csv, train\_habits.csv, test\_demo.csv, test\_health.csv and test\_habits.csv. The training group (Figure 1) has 800 observations and 19 features including a target variable “Disease” that identifies whether a patient has contracted the parasite (1) or not (0). The testing group has 225 observations and 18 features as it excludes the target variable.

# 3. Data Exploration

Before jumping into data exploration, we created a single dataframe from the provided three tables by merging them as they share a foreign key, “Patient\_ID”.

To get an overview of the data and better understand what we were dealing with, we took advantage of the Pandas profile report. The report produced a series of insights that ranged from variable interactions and correlations to missing data (Figure 1). It was detected that the dataframe was missing 0.1% of its cells that translated to 13 missing values in the feature “Education” (Figure 2). The rest of the features had no missing values with a count of 800 data points each and there were also no duplicate rows.

Out of the 19 features 8 were numeric, 9 were categorical and 2 were Boolean (Figure 1).

All numerical features were normally distributed apart from “Physical\_Health” that skewed to the right, presenting a mean of 4.55 and a median of 3 (Table 4).

Some outliers were detected within “High\_Cholesterol” and “Blood\_Pressure” having max values that were too extreme and possibly clinically critical (Table 4). Blood pressure (Table 8) indicates how much pressure your blood is exerting against your artery walls when the heart beats. A value superior 180 mm Hg suggests a hypertensive crisis. These values would require medical support immediately. The cholesterol (Table 9) numbers show how much cholesterol is circulating in our blood and help to predict the risk of a heart problems. Together with age, smoking status, and blood pressure a doctor can evaluate a risk regarding heart problems. Has we don’t have an official number we after running through all the solutions below, it seemed that 180 would be a reasonable value.

“Birth\_Year” had a minimum value of 1855 which would lead to an individual beyond the age of 100 years old (Table 4).

According to Spearman’s correlation matrix (Figure 3) “Weight” and “Height” are positively correlated and so are “Mental\_Health” and the target “Disease”. “Physical\_Health” is negatively correlated with the target “Disease” and “Mental\_Health”.

Inserted into the categorical features was the target variable “Disease” with a balanced distribution of 51.4% 1s and 48.6% 0s (Figure 4).

Exploring each categorical feature individually, it was detected that “Region” had a duplicate value of ‘London’ in the form of ‘LONDON’ (Figure 5). Also, the category ‘I do not consume any type of alcohol’ in the feature “Drinking\_Habit” had a low count of 11 when comparing it with the 383 in ‘I consider myself a social drinker’ and 406 in ‘I usually consume alcohol every day’ (Figure 6). “Education” included some labels that offered the same level of education, for example someone with the category ‘University Incomplete (1 to 2 years)’ is in fact categorized as ‘High School Graduate’ (Figure 7). “Fruit\_Habit” like “Drinking\_Habit” had a label, ‘More than six pieces of fruit’ with a low count of 12 (Figure 8) that could possibly be aggregated to another label with similar information. “Checkup” also contained a label, ‘Less than three months’, with a reduced number of observations, 6 (Figure 9).

# 4. Pre-processing

## ***4.1 Feature Engineering***

Knowing that any change applied to the training dataset would have to be applied to testing, we created a function, feature\_engineering(), that could later on be called to test and apply the same modifications. The alterations mentioned bellow were all included inside the previously mentioned function.

Firstly, we renamed some of the features so they could be more perceptible to what they represent (Table 5).

The second step was to create 3 new features from the original ones. “Birth\_Year” was converted into “Age” that shared the same distribution and offered a more meaningful understanding of the feature. This was done by subtracting “Birth\_Year” from the current calendar year. Given that we were conducting a data analysis in a health/medical field we thought that body max index, BMI, would be a great asset and by not having it we were missing out. BMI is a value derived from “Weight” and “Height” of a person and it can be expressed numerically or categorically. We were not sure which one was the best fit, so we included both. “BMI\_metric” was a numeric feature and it originated from the division of body mass, “Weight”, by the square of the “Height”. “BMI\_noneMetric” was a categoric feature with 3 categories: ‘Underweight’, ‘Normal’ and ‘Overweight’. These categories represent a range of values from “BMI\_metric” (Table 6).

Moving forward transformations were performed on some incoherencies that had been observed in the data exploration. The duplicate value issue found in “Region” was solved by applying the method capitalize() that transforms every string value by upper casing the first character and lower casing the rest (Table 7). “Drinking\_Habit” had also been flagged for having a category with few observations, so this was solved by aggregating that specific category into another with what we considered to be a similar information and like this we also reduced the dimensionality of the feature (Table 7). In “Education” we had already observed that similar labels offered the same level of education, so the solution was to aggregate these labels (Table 7). This aggregation of labels lead to the feature “Education” having now 4 categories, reducing its dimensionality significantly. To “Fruit\_Habit” was applied the same solution used for “Drinking\_Habit” and the label with fewer observations was aggregated to the one which included similar information (Table 7). “Checkup” shared the same fate of “Drinking\_Habit” and “Fruit\_Habit” and its label ‘Less than three months’ was aggregated to ‘Less than 3 years but more than 1 year’ (Table 7).

The categorization of the original categorical features besides the target “Disease” and “Name” came next to facilitate in the observation of results (Table 7).

“Patient\_ID” was set as the index given that it was the unique identifier of each patient.

With the creation of the feature “Age” that offered the same distribution and core information of the original feature “Birth\_Year”, it was good practice to drop this last feature was together with “Name” that presented a high cardinality.

Before wrapping up the feature engineering section and returning a new and clean dataframe (Table 25), “BMI\_noneMetric” had its data type changed to object to match the rest of the categorical features (Figure 10).

## ***4.2 Outlier Handling Solutions***

When looking for outliers, 4 numerical features stand out: “Age”, “Cholesterol\_value”, “Blood\_Pressure\_value” and “Physical\_H\_bad” (Figure 11). After looking at each of them carefully we concluded that “Physical\_H\_bad” did not include outliers, but extreme values that were included within the range of days collected by that feature. The remaining 3 features had data points that were too extreme and had to be addressed.

Before moving to the proposed solutions, we defined a threshold of acceptable values for each of the three features:

“Age” < 100, “Cholesterol\_value” < 380 and “Blood\_Pressure\_value” < 180

We explored 4 possible solutions to handle these outliers. For the first solution (Figure 12) we substitute the extreme values by the previous threshold, for the second solution (Figure 13) we dropped the outliers “manually”, the third solution (Figure 14) the IQR method was to be used and our fourth created solution (Figure 15) was to use a mix of solutions 2 and 3. We also explored with outliers.

As a dynamic process, we tested all these solution in our models and we decided to proceed with the first solution, substitute the extreme values by the threshold (Table 10).

## ***4.3 Feature Selection***

After treating the outliers, we split the data into dependent and independent (“Disease”) variables and check that our dataset is balanced with 51% of patients with disease and 49% without (Figure 4). Then, used the hold out method to randomly split the dataset into training and validation, using 80:20 instead of 70:30 to avoid bias, due to our small/medium dataset. Although the dataset is balanced, we used the stratified method to keep the same structure.

### *4.3.1 Numeric features*

#### 4.3.1.1 Handling missing values

After we split the data, missing values were filled, in training and validation dataset, in feature ‘Education’ with the mode, the most frequent values, of the training dataset. It looked to be an acceptable approach as the categorical feature ‘Education’ has a normal distribution (if it was skewed it would lead in a problem), and the number of missing values is not hight, just 1.6%, and we believe it is better than drop it. And in the feature selection we ended up dropping it, as it was not relevant.

#### 4.3.1.2 Scaling

As our features were measured at different scales, they did not contribute equally to the model fitting, ended up creating bias. To deal with it, the normal approach was to scale the data and we could do it with different scalers.

1) MinMaxScaler scales numeric data features in the range [0,1] or else in the range [-1,1], if there are negative clause in the dataset. This can be very useful for some ML models like the Multi-layer Perceptrons (MLP), where the backpropagation can be more stable and even faster when input features are min-max scaled (or in general scaled) compared to using the original unscaled data. [[1](https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79)]

3) StandardScaler standardize the data following a standard normal distribution, removes the mean (mean equal to zero) and scales data to a unit variance (standard deviation equal to one), however, with outliers does not guarantee balanced features scales.

4) Robust\_scaler scales features using statistics that are robust to outliers. This method removes the median and scales the data in the range between 1st quartile and 3rd quartile, the interquartile range. If outliers are present in the dataset, then the median and the interquartile range provide better results and outperform the sample mean and variance. [2]

We test these different approaches in our models (Table 11) , as there are no significant changes in model performance, as we have previously treated outliers and as it seems theoretically more robust, we opted for MinMaxSclares with a range of 0 to 1. We run the scaler model in training dataset and based on this we applied the transformation in the training (Table 12), validation (Table 13), and lately in the test data set.

#### 4.3.1.3 Spearman Correlation and Wrapper Methods

To select the most relevant features for our model, we split the training and validation data into numeric and categorical features and perform different techniques in training dataset to give as some insights.

For numeric variables, first we check that there were no univariate variables to delete, and then performed different techniques:

1) Correlation – According to the Spearman’s correlation (Figure 16), there was not a single feature that highly correlated with the target; however, it was observed that "Weight" and "BMI\_metric" share a correlation value of 0.9. Given that "BMI\_metric" has a higher correlation towards the target, "Weight" should be discarded.

2) Wrapper Method

2.1) RFE / Backwards - Recursive Feature Elimination (RFE / Backwards) will identify the most important features to keep. The base estimator used was the Logistic Regression, and we found the Optimum number of features are six with a score of 76,88%, based on this, the less important features are wieght and “BMI\_metric”.

2.1) Lasso Regression, that is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. As Weight is present a value equal to zero, we can discard it.

2.2) Decision Tree – Based on the decision tree regressor, for both Gini and Entropy criteria, the least important variables are “Height”, “BMI\_metric” and “Weight”.

Based on the methods (Table 14), and after running our models with different criteria (Table 15), we end up discarding “Height”, “Weight” and “BMI\_metric”.

### *4.3.2 Categorical features*

For categorical features we ran the chi-square test, that is an hypothesis testing method, that checks the importance of the categoric features towards the target. With an alpha equal to 5%, we discarded the features “Region”, “Education”, “Smoking\_Habit” and “Water\_Habit”. By looking at a barplot (Figure 17) of the categorical features and the target, we can observe that these specific features do not add any information as they do not differ when the target is both positive or negative.

### *4.3.3 Selecting Relevant features*

## ***4.4 Feature Encoding***

 After having selected which categorical features were relevant, we encoded the remaining ones with One-hot encoding. This transforms categorical data into numeric data and It transforms each unique value of a categorical feature into a binary variable.

# 5. Modelling

After concluding the data pre-processing, we moved on to exploring a diverse range of models (Figure 18 and Figure 19), with default parameters, to get an idea of the best score. Logistic regression, Neural Networks and Ensemble Methods are ranked first. We also benefit from this procedure to get insights on what outlier solution to apply to the final model (Table 10), Scaler (Table 11) and Features selection (Table 15), running it multiple times with different assumptions. Based on this we decided to explore these models, and one of our biggest concerns was to keep a good balance between performance and overfitting [3,4].

Overfitting is a common problem in machine learning, where a model performs too closely on the training data but does not generalize well on the validation and test data. The reason for that is that when we use too many features, some of them are useful (signal) and others are not (noise) for the learning process. If the model learns to separate the signal from the noise, it will produce good results in the validation and test data. When overfitting occurs, the training score has good result, but the validation and test will be much lower.

A good common practice to avoid overfitting is to explore different feature combinations in the pre-processing stage and extract the best possible features.

Cross validation is another good method to include where the training data is split into N-“blocks” that are used for validation each time while the rest are used for training. This process is looped until all blocks have been used for validation. Stratified Kfold [5] was the chosen cross validation strategy to help us select the best parameters for each model and to tune them as it is the most used technique. stratified k-fold cross validation reduces the concern of utilizing random sampling and obtaining data with skewed data distributions.

We used Repeated Stratified Kfold to tune Neural Network and to choose the best parameters for the ensemble models.

Another way to control overfitting is through iteration, also known as the “early stopping” method, where each parameter is explored at a time keeping the others as default. There is an optimum number of iterations where the model learns but after a certain point the model will only learn noise.

Regularization was also explored because it is a good way of finding bias-variance trade-off by tuning the complexity of the model. Regularization is adopted universally as simple data models generalize better and are less prone to overfitting. This model was explored in more detail when Logistic Regression was applied.

Logistic Regression [6] is basically linear regression, but the output lies in a binary response instead of a continuous variable. The Input feature can be continuous or discrete, linearly scaled, and the function provides a nonlinear transformation. To obtain the estimated vector of parameters use the maximum likelihood estimation (error minimization technique), the vector that maximizes the likelihood of observing the sample, to find the best s-shaped function for the data. We can apply the regularization to this function and increase the strength of regularization [7,8].

In Logistic regression, C is our regularization parameter, C = 1 / λ. Lambda (λ) controls the trade-off between allowing the model to increase its complexity as much as it wants while trying to keep it simple. When λ is low, the model will have enough power to increase its complexity (overfit), by assuming big values to the weights for each parameter. On the other hand, when increasing λ, the model tends to underfit, as the model becomes too simple. Parameter C works the other way around, small values of C, increase the regularization strength and vice versa. We tested it in our model, to find the best C and avoid overfitting, we applied it assuming a high number of features (L1 penalty) or to deal with multicollinearity, independent variables highly correlated (L2 penalty). For each value of C there are differences between the accuracy of train and validation. Until C = 1 (the default number), both rate increase, train rate faster than validation rate (L2) (Figure 21), but for C greater than 1, both rates remain almost steady (L1) (Figure 20), or even worst, the validation rate starts to decrease. So, the best parameter regarding regularization, for our logistic regression is the default.

Neural networks imply regularization confiding the complexity (weights) of the model. Random forest, by reducing the depth of tree and branches (new features).

Keeping in mind that the optimal parameters are the ones that give the best trade-off between accuracy and overfitting reduction.

We ran neural networks, with different parameters, through the Stratified Fold cross validation method, to understand the behaviour, to tune it and to try avoiding overfitting. The hidden layer size (Table 16) increased the performance with more hidden layers, but to avoid under and overfitting, and to keep a good trade-off between the simplicity of the model and the performance accuracy, we changed the default hidden layer size from 100 to 16, the number of features. Regarding the number of iterations (Table 17), the default is set to 200 and we increased it to 300, to improve the performance but not too much to avoid overfitting. In the solver parameter (Table 18), ‘L-BFGS’ is a good option for low dimensional models and for sparse data, but in this case, it increased the train and validation scores too much and we were afraid it would lead to overfitting and ‘SGD’ lowered the performance. We ended up keeping the default ‘Adam’, that achieved good results faster, had a smaller variance between training and validation and was not as prone to overfitting as the better performing ‘L-BFGS’. The default Learning rate initialization (Table 19) is set to 0.001 and it seemed to be the optimal learning rate, that swiftly reached the minimum point. The last parameter explored was batch size (Table 20) that showed a good trade-off between accuracy and speed when kept at 50.

Ensemble learning methods are very robust against overfitting, by combining the weak models to form a stronger learner. Bagging models - Bagging and Random Forest - divide the dataset into subparts, each will feed to a model, mainly decision tree, and with the help of max-voting or by taking the average, even when the algorithm does not perform well in the original dataset, it might perform well on the given sections of that dataset. Boosting - Ada Boost and Gradient Boosting - allow every consecutive sub-dataset model to learn from the previous sub-dataset, allowing the model to learn from every section of the dataset and reduces the errors significantly.

These models are so robust, that after we tested the main parameters with different values, most performed better with the default ones (Table 21, Figure 22 - Figure 41). For instance, from all the ensemble models, decision trees ended up being the better base estimated, after comparing with K-Neighbours and Logistic Regression in Bagging and Logistic Regression in AdaBoost. Regarding the number of estimators, for most cases we decided to increase it until the score got steady. The main increase was tested in Gradient Boosting to compensate for the low learning rate and to avoid overfitting. The number of features, in Bagging told us that about 60% of the features was enough for the model to perform.

# 6. Performance Assessment

After tuning each model, to have the best parameters that would give the best performance without overfitting, we create the instance, fit it with the dependent and independent training dataset, to train the model, then use the validation dataset to get the scores, like accuracy, precision, recall and F1 score, and making our decision based on F1 Score (Table 22).

We also compared the training score with validation score, for each model (Table 23).

Finally, we also run-on test dataset, submit on Kaggle (Table 24).

Every model got good score, even logistic regression, for a simpler model, it ended with an F1 score of about 85,5%. But on Kaggle it performed worst with a score of 76,6%. The opposite occurred with Neural Networks, that had a good F1 score and a better score on Kaggle.

The ensemble methods worked as a good boost to the predictions, the boost models were slightly worse than the Bagging ones, but overall, we believe that all of them performed well regarding overfitting, even when different parameter values were tested.

# 9. Conclusion

In this project we tackled a Machine Learning classification problem with the aim of predicting whether a person was sick with the fictional disease The Smith Parasite.

We were provided with a small quantity of sociodemographic, health, and behavioural information obtained from patients. This data was missing some values but overall, it was fairly clean.

The pandas profile report gave us a versatile tool to explore the data and assisted us in planning our strategy to address the problem at hand. He observed some incoherencies with feature data that were later addressed in the feature engineering section.

In feature engineering we introduced 3 new features and dropped irrelevant ones.

Moving on to how we dealt with outliers. Out of the 5 possible ways we could have chosen, we opted to use Solution 1 that consisted of transforming extreme values and not removing any data given that the dataset was already small, and any redaction could potentially affect our findings. We also observed great results when no outliers were removed but the model that we selected as our final answer, Ensemble Bagging model, performed better when solution 1 was used.

At the end we picked Bagging instead of Random Forest, even though they performed similar, because Bagging used only 60% of the dataset leading to a reinforcement of overfitting reduction automatically when compared to Random Forest that had to use all of the data to achieve a performance that was close.

# 10. References

[1] [Everything you need to know about Min-Max normalization: A Python tutorial | by Serafeim Loukas | Towards Data Science](https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79)

[2] [How and why to Standardize your data: A python tutorial | Towards Data Science](https://towardsdatascience.com/how-and-why-to-standardize-your-data-996926c2c832)

[3] https://www.bogotobogo.com/python/scikit-learn/scikit-learn\_logistic\_regression.php

[4] <https://www.analyticssteps.com/blogs/5-machine-learning-techniques-solve-overfitting>

[5] <https://www.v7labs.com/blog/train-validation-test-set>

[6] [https://analyticsindiamag.com/7-types-classification-algorithms/

[7] https://www.bogotobogo.com/python/scikit-learn/scikit-learn\_logistic\_regression.php

[8] <https://www.kaggle.com/code/joparga3/2-tuning-parameters-for-logistic-regression>

[9] <https://cds.cern.ch/record/998831/files/9780387310732_TOC.pdf>

Last accessed - 20th December 2022

[10]<https://github.com/linux08/machine-learning-books/blob/master/Fundamentals%20of%20Machine%20Learning%20for%20Predictive%20Data%20Analytics%202015.pdf>

Last accessed – 19th December 2022

[11] <https://github.com/Apress/mastering-ml-w-python-in-six-steps>

Last accessed – 20th December 2022

[12] <https://www.geeksforgeeks.org/data-preprocessing-machine-learning-python/>

Last accessed – 21st December 2022

[13] <https://www.sas.com/en_gb/insights/articles/analytics/machine-learning-algorithms.html>

Last accessed – 15th December 2022

[14] <https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/>

Last accessed – 09 December 2022

[15] <https://www.youtube.com/watch?v=7eh4d6sabA0>

Viewed – 14th Dec 2022

[16] <https://www.youtube.com/watch?v=JxgmHe2NyeY>

Viewed – 10th Dec 2022

[17] <https://www.youtube.com/watch?v=vh525RjO6C0>

Viewed – 10th Dec 2022

# 11. Appendices

## ***11.1 Tables***

|  |  |
| --- | --- |
| Variable Name | Description |
| PatientID | The unique identifier of the patient |
| Birth\_Year | Patient Year of Birth |
| Name | Name of the patient |
| Region | Patient Living Region |
| Education | Answer to the question: What is the highest grade or year of school you have? |
| Disease | The dependent variable. If the patient has the disease (Disease = 1) or not (Disease = 0) |

Table 1 - Sociodemographic data variables

|  |  |
| --- | --- |
| Variable Name | Description |
| PatientID | The unique identifier of the patient |
| Height | Patient's height |
| Weight | Patient's weight |
| Checkup | Answer to the question: How long has it been since you last visited a doctor for a routine Checkup? [A routine Checkup is a general physical exam, not an exam for a specific injury, illness, or condition.] |
| Diabetes | Answer to the question: (Ever told) you or your direct relatives have diabetes? |
| High\_Cholesterol | Cholesterol value |
| Blood\_Pressure | Blood Pressure in rest value |
| Mental\_Health | Answer to the question: During the past 30 days, for about how many days did poor physical or mental health keep you from doing your usual activities, such as self-care, work, or recreation? |
| Physical\_Health | Answer to the question: Thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good to the point where it was difficult to walk? |

Table 2 - Health Related data variables

|  |  |
| --- | --- |
| Variable Name | Description |
| PatientID | The unique identifier of the patient |
| Smoking\_Habit | Answer to the question: Do you smoke more than 10 cigars daily? |
| Drinking\_Habit | Answer to the question: What is your behavior concerning alcohol consumption? |
| Exercise | Answer to the question: Do you exercise (more than 30 minutes) 3 times per week or more? |
| Fruit\_Habit | Answer to the question: How many portions of fruits do you consume per day? |
| Water\_Habit | Answer to the question: How much water do you drink per day? |

Table 3 - Habits Related data variables

Graphical user interface, table, Excel

Description automatically generated

Table 4 – Numeric features descriptive statistics

|  |  |
| --- | --- |
| Original Feature Name | New Feature Name |
| “High\_Cholesterol” | "Cholesterol\_value" |
| "Blood\_Pressure" | "Blood\_Pressure\_value" |
| "Mental\_Health" | Mental\_H\_bad" |
| "Physical\_Health" | "Physical\_H\_bad" |

Table 5 – Renamed Features

Graphical user interface, text

Description automatically generated

Table 6 – BMI ranges

Table

Description automatically generated

Table 7 – Feature Transformation and Categorization

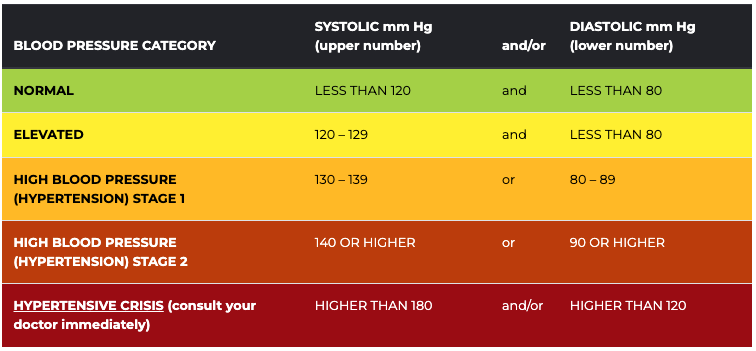


Table – Blood Pressure reference values

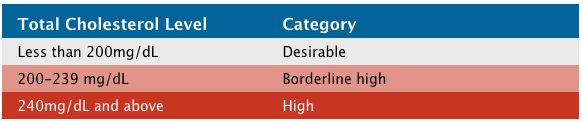


Table – Cholesterol level reference values

A picture containing calendar

Description automatically generated

Table – Exploring different outlier solutions

A picture containing table

Description automatically generated

Table - Exploring different Scalers

Graphical user interface, text, table

Description automatically generated

Table – Training dataset scaled

Text, table

Description automatically generated

Table – Validation dataset scaled

Table

Description automatically generated

Table – Feature Selection Numeric Features Decision

Table

Description automatically generated with medium confidence

Table – Model scores with different Features

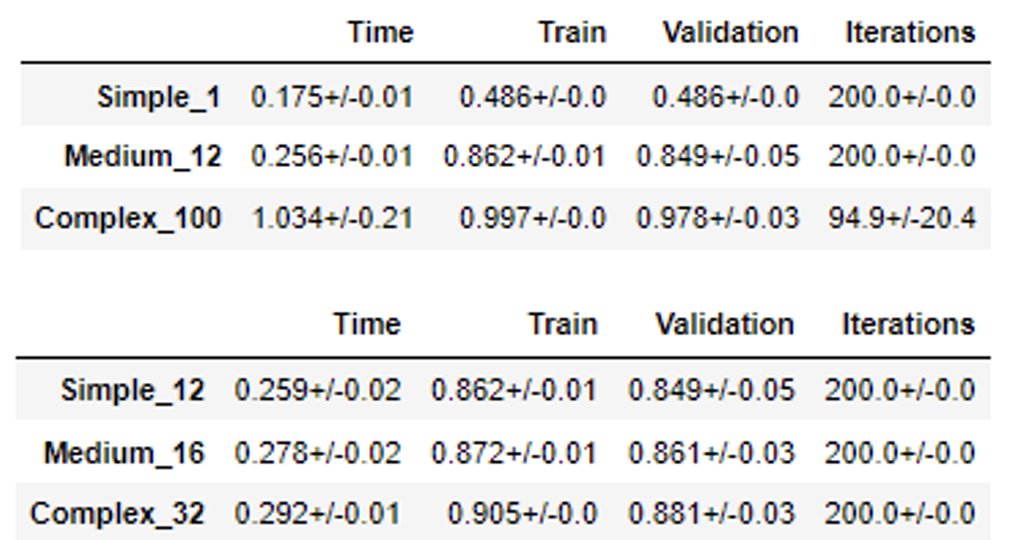


Table 16 – Performance with different Hidden Layer sizes

Table

Description automatically generated

Table 17 – Exploring different number of iterations

Graphical user interface, text, table

Description automatically generated

Table 18 – Exploring different Solver options

Table

Description automatically generated

Table 19 – Exploring different Learning Rate initializations

Table

Description automatically generated

Table 20 – Exploring different Batch sizes

Table

Description automatically generated

Table – Best Parameters for Ensemble

Table

Description automatically generated

Table 22 – Validation model Scores with the most optimal parameter combinations

Table

Description automatically generated

Table 23 – Comparing the training and validation model Scores with the most optimal parameters

Table

Description automatically generated

Table 24 – Kaggle model Score with the most optimal parameters

|  |  |
| --- | --- |
| Variable Name | Description |
| PatientID | The unique identifier of the patient |
| Height | Patient's height |
| Weight | Patient's weight |
| Checkup | Answer to the question: How long has it been since you last visited a doctor for a routine Checkup? [A routine Checkup is a general physical exam, not an exam for a specific injury, illness, or condition.] |
| Diabetes | Answer to the question: (Ever told) you or your direct relatives have diabetes? |
| Cholesterol\_value | Cholesterol value |
| Blood\_Pressure\_value | Blood Pressure in rest value |
| Mental\_H\_bad | Answer to the question: During the past 30 days, for about how many days did poor physical or mental health keep you from doing your usual activities, such as self-care, work, or recreation? |
| Physical\_H\_bad | Answer to the question: Thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good to the point where it was difficult to walk? |
| Age | Patient age |
| Region | Patient Living Region |
| Education | Answer to the question: What is the highest grade or year of school you have? |
| Disease | The dependent variable. If the patient has the disease (Disease = 1) or not (Disease = 0) |
| Smoking\_Habit | Answer to the question: Do you smoke more than 10 cigars daily? |
| Drinking\_Habit | Answer to the question: What is your behavior concerning alcohol consumption? |
| Exercise | Answer to the question: Do you exercise (more than 30 minutes) 3 times per week or more? |
| Fruit\_Habit | Answer to the question: How many portions of fruits do you consume per day? |
| Water\_Habit | Answer to the question: How much water do you drink per day? |
| BMI\_metric | BMI is a value derived from “Weight” and “Height” of a person and it can be expressed numerically or categorically. This one is numeric. |
| BMI\_noneMetric | BMI is a value derived from “Weight” and “Height” of a person and it can be expressed numerically or categorically. This one is categoric and split into the following categories: ‘Underweight’, ‘Normal’ and ‘Overweight’ |

Table – Variables of the dataframe created after feature engineering

## ***11.2 Figures***

Graphical user interface, table

Description automatically generated

Figure 1 – Overview of the raw data

Graphical user interface

Description automatically generated with medium confidence

Figure 2 – Insights of the raw data

A picture containing timeline

Description automatically generated

Figure 3 – Spearman correlation of raw data

Graphical user interface, application

Description automatically generated

Figure 4 – Target variable insights

Graphical user interface, application, table

Description automatically generated

Figure 5 – “Region” insights

Graphical user interface, table

Description automatically generated

Figure 6 – “Drinking\_Habit” insights

Graphical user interface, text, application, email

Description automatically generated

Figure 7 – “Education” insights

Table

Description automatically generated

Figure 8 – “Fruit\_Habit” insights

Graphical user interface, application

Description automatically generated

Figure 9 – “Checkup” insights

A picture containing graphical user interface

Description automatically generated

Figure 10 – Post Feature Engineering data types

Chart, bar chart

Description automatically generated

Figure – Outlier Boxplot when no solution is applied

Chart, bar chart

Description automatically generated

Figure 12 – Outlier Boxplot of solution 1

Chart

Description automatically generated

Figure 13 – Outlier Boxplot of solution 2

Chart, bar chart

Description automatically generated

Figure 14 – Outlier Bloxplot of solution 3

Chart, bar chart

Description automatically generated

Figure 15 – Outlier Boxplot of Solution 4

Application

Description automatically generated with low confidence

Figure 16 – Spearman’s Correlation Matrix heatmap

A picture containing text, cabinet, screenshot

Description automatically generated

Figure 17 - Bar Blot for Categorical Features, with the target

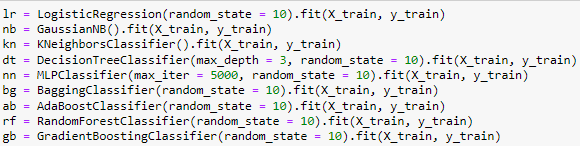


Figure 18 - Multiple models run after the data preprocessed

Chart, bar chart, histogram

Description automatically generated

Figure 19 – Model Ranking

Chart, line chart

Description automatically generated

Figure 20 - Find the best C parameter for the L1 Penalty

Chart

Description automatically generated

Figure 21 - Find the best C parameter for the L2 Penalty

Chart, box and whisker chart

Description automatically generated

Figure 22 – Bagging model base estimator

Chart, box and whisker chart

Description automatically generated

Figure 23 – Exploring different number of estimators in the Bagging model

Chart, box and whisker chart

Description automatically generated

Figure 24 – Exploring different Max Samples in the Bagging model

Chart, box and whisker chart

Description automatically generated

Figure 25 – Exploring different Max Features in the Bagging model

Chart, box and whisker chart

Description automatically generated

Figure 26 – Exploring different Bootstrap in the Bagging model

Chart, box and whisker chart

Description automatically generated

Figure 27 – Exploring different Bootstrap features in the Bagging model

Chart, line chart

Description automatically generated

Figure 28 – Exploring different OOB error rates in the Bagging model

Chart, box and whisker chart

Description automatically generated

Figure 29 – Exploring different Number of Estimators in the Random Forest model

Chart, box and whisker chart

Description automatically generated

Figure 30 - Exploring different Boostrap in the Random Forest model

Chart, box and whisker chart

Description automatically generated

Figure 31 – Exploring different Max Samples in the Random Forest model

Chart, box and whisker chart

Description automatically generated

Figure 32 – Exploring different Max Depth in the Random Forest model

Chart, line chart

Description automatically generated

Figure 33 – Exploring different OOB error rate in the Random Forest model

Chart, box and whisker chart

Description automatically generated

Figure 34 – Exploring different Base Estimators in the Adaptive Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 35 - Exploring different Algorithms in the Adaptive Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 36 - Exploring different number of estimators in the Adaptive Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 37 - Exploring different learning rates in the Adaptive Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 38 - Exploring different algorithms in the Adaptive Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 39 - Exploring different learning rates in the Gradient Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 40 - Exploring different numbers of estimators in the Gradient Boosting model

Chart, box and whisker chart

Description automatically generated

Figure 41 - Exploring different max features in the Gradient Boosting model