The Smith Parasite

An Unknown Parasitic Disease

Machine Learning Project

23.12.2022

Leonhard Allgaier	20220635
David Halder	20220632
Hubert Oberhauser	20220628
Rita Soares	20220616
Lukas Stark	20220626

Table of Contents

1. Introduction **2**

2. Data Exploration **2**

3. Pre-Processing **3**

3.1. Methodology 3

3.2. Function: “Preprocessing” 3

3.3. Function: “encode_scale” and scaling pipelines 4

3.4. Feature Selection 4

4. Modelling **5**

4.1. Model Selection 5

4.2. Optimization 6

5. Assessment **7**

5.1. Assessment Methods 7

5.2. Results 8

6. Conclusion **8**

7. Annex - Self Study **10**

7.1. Annex 1 - Density-Based Spatial Clustering of Applications with Noise (DBScan) 10

7.2. Annex 2 - Stochastic Gradient Descent Classifier 10

7.3. Annex 3 - Cumulative Gain- and Lift Charts 11

8. Annex - Figures & Tables **13**

Figure 1 - Project Plan 13

Figure 2 - Spread Map Smith Parasite - United Kingdom 13

Figure 3 - Boxplots 14

Figure 4 - Pairplot 15

Figure 5 - Barplots 16

Figure 6 - Correlation Heatmap 21

Table 1 - Feature Selection 22

Figure 7 - Feature Selection - SGDClassifier 23

Figure 8 - Feature Selection - DecisionTreeClassifier 24

Figure 9 - Feature Selection - RandomForestClassifier 25

Figure 10 - ROC Curves 26

Figure 11 - Precision-Recall Curves 27

Table 2 - Optimal Hyperparameters - RandomForest 27

Figure 12 - Results 28

9. References **29**

1. Introduction

In the year 2022, the smith parasite was discovered in England. It has yet to be discovered how it's transmitted, and which group of people are more vulnerable to it. However, it is clear that an infection can have severe short- and long-term-consequences. This parasite represents a danger to public health and could possibly be a new pandemic. Therefore it is crucial to get a deep understanding of this parasite and find reliable methods to identify infected individuals.

This report summarises a study that aims to classify the group of individuals more likely to suffer from the parasite using machine learning algorithms to predict, based on a set of information, how likely a patient is to be infected or been infected in the past – a value above 0.5 will be taken as a positive for the infection. Besides predicting infections this project also aims to find out what characteristics make individuals susceptible to an infection.

Figure 1 shows the process of the project, on which the structure of the report is based on. First the data exploration, the pre-processing and the feature selection steps are presented. Next the model selection and optimization is discussed. Finally, the assessment methods, the final results are elaborated and concluded.

2. Data Exploration

For this study three datasets were provided, one with mainly demographic and personal data of the patients and the other two composed of health habits and medical information. Since a description of the variables was given with the project, it will not be repeated here.

Figure 2 shows how this disease is affecting the different regions of England with the respective population density. Examining the map, it seems like the parasite spread from the South to the comparatively less populated North of England. In comparison to other viruses it does not seem like it spreads faster in highly populated areas than in rural areas, as the absolute number of cases is indeed higher in London but the relative number of cases seems comparatively stable.

After checking the data types and the descriptive statistics of the data, it was analysed for empty strings and NaN-values. Education is the only column with missing values, 13 in total. To detect outliers, boxplots and DBScan were used (See Figure 3)

The balance of the target variable and the gender was analysed as well. While the target variable is not severely unbalanced (1: 51%) the gender shows that we have significantly more male observations (Male: 71%). This could potentially have implications on the prediction of infection of the disease on females, but it was decided to not process this since it could cause other variables to gain importance artificially.

For univariate and bivariate analysis and to get a better understanding of the relationship between each independent and the dependent variable we plotted pairplots and barcharts (See Figure 4 & 5). It shows how each column is distributed by the possible target values, 0 (no disease) or 1 (presence of disease), additionally the proportional plot was also considered to assess that the distribution is not biassed by a higher or lower number of records of a certain value. A good feature is expected to have an uneven spread between values, as this shows a good indicator to infer the target value. Thus we can already conclude that Water_Habit, Smoking_Habit, Education and Region will not be relevant to our models.(See Figure 5)

3. Pre-Processing

3.1. Methodology

In order to ensure reproducibility two functions were created for the largest part of the Pre_Processing. In the first function data transformations, missing and inconsistent values as well as outliers and ordinal encoding is covered. The second function includes the scaling of data and one-hot encoding of nominally scaled data. This is due to the reason that different scaler/classifier combinations are tested. Finally, we utilise pipelines to scale the data before every k-fold iteration.

3.2. Function: "Preprocessing"

First the faulty values in the birth year are corrected. It was assumed that there had been a mistake during the gathering of the data. As the numbers eight and nine are very close to each other on the keyboard it was presumed that the birth year was meant to be in the 20th century. Therefore 100 years were added to every birth year which fitted this pattern. To account for the wrong values in the "Region" column all values were set to lowercase, which corrected the observations with the characteristic "LONDON".

Next variables were added, first we calculated the age with the current year and the corrected birth year column. Next, a "Gender" variable was created, by extracting the title out of the name-variable of the person. Moreover,a BMI column was calculated by dividing the weight by the height to the power of two. A categorical BMI with four characteristics was added as well, to review if it adds more value respectively less noise.

Using the interquartile-range we found outliers in five different variables (See Figure 3). The outliers in the variable "Birth Year" were processed during the treatment of the incoherences. Since the number of outliers was too high to delete them, we tried different methods, like imputing them with a linear model or the median and ultimately flooring and capping. We decided to impute the values with the value of the closest whisker. This method was chosen, since it removes the extreme values, however the rank of the observation remains the same. Since all missing values were only found in one categorical variable ("Education"), we decided to impute them with the mode. Finally we imputed the

categorical variables which had a clear rank in their characteristic with a manual ordinal encoder and the binary variables with a binary encoder, to change their values to zero and one.

Subsequently in order to find multidimensional outliers we followed a clustering-based approach like described by Han et. al (2011, p 567 pp.). A DBScan algorithm was used to achieve this. (see Annex 1) However, there were not many multidimensional outliers found and excluding them deteriorated the F1-Score.Thus, we reincluded the multidimensional outliers found.

3.3. Function: "encode_scale" and scaling pipelines

The second function has two main purposes: scale the data and encode the variables which are nominally scaled. However, it also ensures that every test/validation set is scaled in the same way as the training set.

The function has three inputs: a dataframe, a scaler (default= StandardScaler()) and reset_fit (default=False). While the first two inputs are relatively straightforward, the third input is more complex but also crucial. If it is set to "True", the scaler will not only transform the dataset but also fit the chosen scaler to the provided dataset and export the fitted scaler as a global variable. Consequently, the function can be used on the training- as well as on the validation-set, given that the training set was scaled first.

However, this function is not suitable for K-Fold splits, since for this method the data has to be scaled based on the training set before every iteration. To ensure this, three pipelines are utilised. First a pipeline which scales the data, next one which fits the data to a classifier and calculates a score based on the predictions and finally a pipeline that puts the first two together. That way it is ensured that the data is always scaled based on the current training set.

3.4. Feature Selection

One last important step before starting with modelling is feature selection. The main goal of this procedure is the performance improvement of the machine learning models. Feature selection can help to eliminate unnecessary or irrelevant features from the dataset which also can boost efficiency and lead to a less overfitting model. Furthermore it helps to reduce the complexity of a model which consequently supports interpretability.

The first approach was to get a rough picture by means of statistical tests. Since the nature of the input and target variables is important in univariate feature selection, we first divided the data into numerical and categorical variables. Subsequently, the numerical columns were used to visualise the correlation matrix by means of a heatmap. (See Figure 6) Due to the categorical property of the target variable, Kendall was used as the method of choice. With regard to the categorical features, a chi-squared test was performed (significance level 0.05).

Since Univariate methods evaluate the importance of each feature independently, they lack to consider the relationships between features. This can be useful for identifying the most important features in a dataset, for example to set countermeasures to slow the spread. However, it may not capture the complex relationships between features that can be important for prediction. To tackle this problem we also wanted to confirm the filter methods results with feature selection methods that are not model independent, hence recursive feature selection with cross validation (RFECV) and SelectFromModel, an embedded method from scikit-learn. (scikit-learn [1], 2022)

Examining the results of the three different feature selection models, some patterns could be detected. As one can see in the table “Feature Selection” there are nine variables which are considered important to predict an infection by all methods (See Table 1). Blood Pressure and High Cholesterol were only selected by the Embedded Method and the Wrapper Method. The gender was only considered useful by the Filter- and the final Embedded-Method. For optimization purposes we decided to use SelectFromModel to get the best selection for each model. In case the optimising classifier isn’t compatible with SelectFromModel, for example the MLPClassifier, we will use the univariate feature set.

To get an even better understanding what variables were most important we can assess the feature importance of the SelectFromModel function of our final model. Here one can see that “Diabetes”, “Checkup”, “Fruit_Habit and “Mental_Health” are important for the model. In combination with the coefficient values of our baseline model and the correlation matrix the direction of the relationship to the target variable can be examined. (See Figure 6 & 7) Regarding the other two classifiers (See Figure 8 & 9).

In order to reduce dimensionality, principal components were calculated. They were first added to the variables and then it was tried to replace variables with them. Since both of these methods led to worse prediction scores and decreased interpretability, they were not further used.

4. Modelling

4.1. Model Selection

In order to evaluate the performance of different machine learning models and select the best one for our dataset, we utilise the hold-out method to split the data into training and validation sets. We use stratified sampling to ensure that the distribution of classes in the dataset is reflected in both sets, with a 80:20 ((640,160) ratio of training to validation data. Splitting the data a third time was not performed since the test dataset and the corresponding kaggle submission provide another possibility to check the generalisation of each model.

As a starting point, we have chosen to use the Stochastic Gradient Descent Classifier (SGDClassifier) (See Annex 2) as our baseline model, as it is a simple linear model that can serve as a reference

point for comparison with more complex models. Before applying any preprocessing steps, we have obtained a baseline score for the SGDClassifier to provide a sense of the model's potential performance. We will also apply feature selection and hyperparameter tuning techniques to optimise the performance of the SGDClassifier throughout the project.

In order to select the most appropriate classification algorithms for our dataset, we employed two evaluation metrics. The first metric was the receiver operating characteristic (ROC) curve and its corresponding area under the curve (AUC). This is a valid approach, particularly for datasets that do not exhibit imbalanced class distributions. Additionally, considering the performance evaluation based on the F1 score, the harmonic mean of precision and recall, we implemented the precision-recall curve as well. With the fact that making wrong predictions for infected people could be really problematic, the predictions should be more recall-driven. In that case, considering the precision-recall curve with its calculated average precision (AP) score has to be viewed with caution. Not only should the models with the highest AP scores be selected, but the curve shape should also be taken into account. A more right-lying curve, indicating a higher recall, is therefore more suitable for our recall-driven approach and that's why we opted for the DecisionTreeClassifier instead of Support Vector Classification (SVC). Ultimately, we utilised both the ROC curve and Precision-Recall curve in order to select the optimal model. (See Figure 10 & 11)

Based on these metrics we chose the best three classifiers which resulted in MLPClassifier (MLP), DecisionTreeClassifier (DT) and an ensemble classifier - RandomForestClassifier (RF).

4.2. Optimization

Finally we ended up with four different classifiers to fine-tune. The initial baseline classifier SGDClassifier, and the three selected via the ROC/AUC & Precision-Recall curves. As previously described in the "Feature Selection" chapter, we began by using SelectFromModel to perform classifier-specific feature selection, as different classifiers may prioritise and assign varying levels of importance to different features. We then calculated the initial F1-score for our train-validation split and plotted a cross-validated learning curve to assess the training and validation performance. To optimise the hyperparameters, we first rolled out a randomised search to approximate the optimal parameters and consequently did a more precise GridSearch to identify the optimal parameter settings. In order not to go beyond the scope of this report, only the fine-tuning of our final model will be discussed in detail. The GridSearch process resulted in the identification of the following optimal combination of hyperparameters for our random forest model (See Table 2). The *max_depth* parameter was used to prune the tree to a specific degree which helps to decrease overfitting problems by prohibiting nodes to fully expand till only pure leaves exist. As for *max_features*, which determines the number of features to consider when searching for the best split, the GridSearch results in "None" and therefore considers all available features. With the *min_samples_leaf* hyperparameter, we ensure that each leaf node contains at least one sample and *min_samples_split* is making sure that nodes don't get split before containing a minimum of 4 samples. Last but not least

the optimal amount of individual trees resulted in an *n_estimators* value of 450. After all, we recalculated the F1-score and plotted the learning curve to evaluate the effect of the GridSearchCV on the model's performance. With the exception of the SGDCLassifier, the learning curves of the other three fine-tuned models still indicate a degree of overfitting. Given the fact that the individual validation scores and kaggle scores resulted in a good degree of generalisation, we made the decision to not further focus too strongly on the cross-validated learning curves. (See Figure 12)

5. Assessment

5.1. Assessment Methods

As previously mentioned, the main evaluation metric of this project is the F1-Score. The metric is not only calculated for the train and validation set but can also be obtained at submission which offers another opportunity to test the obtained models on unseen data.

On top of the F1-Score multiple other assessment methods were used. As previously mentioned, the ROC/AUC Curve and the Precision-Recall Curve were applied for model selection. For optimization learning curves, based on a stratified K-Fold average of the F1-score were used. As the learning curves were examined within model optimization, they will not be furtherly discussed. For the final model evaluation, the confusion matrix, the recall and Lift and Cumulative Gain Curves were evaluated.

An explanation of Lift and Cumulative Gain Curves can be found in the Annex. They are most used for marketing opportunities, to minimise costs and maximise the return by focusing only on the most promising groups. However, as shown by Joloudari et. al. (2019) in their study about predicting liver disease, these charts can also be used in the medical context. In our case, testing and treatment probabilities will be scarce and expensive, therefore it would make sense to focus on the deciles with the highest probabilities to ensure efficient identification, testing and treatment.

Since the objective of this project is to classify if a person is infected, the recall score is also given a high weighting within our assessment. A high recall means that the model is able to correctly identify most of the positive cases, which is particularly important in a medical setting.

In our project, we were trying to classify whether a person has the Smith parasite, so it is important to have a high recall. A high recall means that the model does not produce a lot of false negatives, which is particularly important in a medical setting where it is crucial to identify all positive cases of a particular disease. However, as elaborated before, resources for testing or treatment might be scarce. Therefore, the recall is only considered in combination with either the precision or the F1-Score.

5.2. Results

The first results were retrieved from the baseline model, a stochastic gradient descent classifier, prior to pre-processing the data. It achieved a F1-Score of 0.67. After pre-processing the model was used again and reached a cross validation score (F1) of 0.85 for the “RobustScaler”. Examining at the other classifiers two stand out: The Decision Tree and the Random Forest with cross validation scores of 0.95 and 0.97. This result can also be seen for the ROC/AUC Curve, since the Random Forest achieved an AUC of 1 and the Decision Tree of 0.96. However, here the MLP-Classifier scores even higher, with an AUC score of 0.99. The Precision/Recall Curve behaves similarly, with a score of 1 for the Random Forest. However, it can be examined that the SRD and the SVM fall short on recall, which, as explained before, is an important metric for this use-case. Lastly the four final models were chosen, trained, and optimised.

Looking at the different scores and curves within Figure 12 it gets clear that the Random Forest is superior to all the other classifiers. While the SRD and the Neural Network produces 13 and 2 False Negatives the RF has a recall score of 1. This is also given for the decision tree, however the number False Negatives produced by the Decision Tree are considerably higher than of the Random Forest. Hence, the Random Forest achieved the highest F1-Score of 0.99 on the validation set.

The Random Forest also shows a great effectiveness. All observations which fall under the 0.4-percentile (sorted descending by prediction probability) contain 79% of all True Positive values. That means that the model classifies the disease 1.9 times better than random choice would. (See Figure 12 and Annex 3) Finally, the model was used to classify the unseen observations in the test set. After submitting the predictions to Kaggle, the predictions achieved a F1-Score of 1, which means that 100% of the observations were classified correctly.

6. Conclusion

This project used supervised learning to classify individuals which are infected with the smith parasite, based on the sociodemographic, health and behavioural data provided. This is highly significant, since the origin, the way it spreads, and the long-term consequences can be severe. On top of that there are also asymptomatic cases and, since it has only recently been discovered, testing infrastructures or vaccines might not be available yet..

The objective was to identify positive cases with the highest possible F1-Score. Therefore, four models were trained, optimised, and tested (SRD, MLP, DT, RF). The highest performing model, the Random Forest, identified that the variables which contain individual information about Diabetes, the habit regarding regular check ups and regarding the habit of eating fruit regularly have the highest importance for predicting whether an individual is infected or not. Surprisingly the region does not seem to have a high impact, even though there are various population densities represented in the dataset.

In addition to its value in the specific context of predicting infection with the smith parasite, this project highlights the potential for machine learning and artificial intelligence (AI) to revolutionise the field of medicine. By leveraging the power of these technologies, we can gain insights and make predictions that would not be possible through traditional methods alone. This has the potential to improve patient care diagnostics, and outcomes, as well as streamline and optimise various aspects of the healthcare system.

7. Annex - Self Study

7.1. Annex 1 - Density-Based Spatial Clustering of Applications with Noise (DBScan)

DBScan is a density based clustering algorithm, which creates clusters based on the number of objects close to point o. The algorithm has two input parameters: ϵ, which specifies the neighbourhood of point o, and MinPts, which specifies how many points need to be within ϵ so that point o can be marked as a core object.

The algorithm randomly selects one point in the dataset, and checks its ϵ-neighbourhood if there are enough points within it to make it a core object. If that is not the case, it is marked as noise and the algorithm moves on to the next point. If it is the case, the point is declared a central point and a cluster C is created. All the points within this cluster are then saved in a candidate set N and are visited by the algorithm, to see if they are core points as well. This continues until the algorithm can't find anymore core points and cluster C is completed. (See Han et. al., 2011, p. 472 pp.)

In the case of this project, DBScan was used to find the noise points which do not fall within any of the ϵ-neighbourhoods respective clusters. To find the optimal value for ϵ we used the K-Nearest Neighbour Algorithm like it was introduced by Rahmah and Sukaesih Sitanggang (2016): The distances are calculated, sorted, plotted and at the point of the maximum curvature we can extract the optimal ϵ. Finally the DBScan algorithm provided by scikit-learn was used to detect multidimensional outliers, but since removing them worsened the F1-Score it was decided to keep them in the model. (See scikit-learn [2], 2022)

7.2. Annex 2 - Stochastic Gradient Descent Classifier

The Stochastic Gradient Descent (SGD) Classifier is a linear classification function which can apply stochastic gradient descent to different loss functions and apply penalties in order to classify observations. (See scikit-learn [3], 2022)

Gradient Descent in general trains a model by computing the partial derivatives, or in other words the gradient, of every parameter of the model and minimising a loss function in order to fit the intercept as well as the slope of the function to the available data. It uses a predefined training rate which decreases the learning steps as it gets closer to the optimum. (See Géron 2019 p. 123 pp.)

Stochastic gradient descent works in a similar way but doesn't pick the whole training-set to calculate the partial-derivatives but picks a random instance at every step and computes the gradient only

based on the chosen observation. This makes the stochastic gradient descent significantly faster than the batch version, since it doesn't calculate all the gradients for all the observations, which is especially useful for large training datasets. (See Géron 2019, p. 125 pp.)

In order to choose the optimal features for the classifier we try different feature selection methods on a baseline SGD-Classifier from Scikit-Learn. From all the different methods used, Filter-, Wrapper- and Embedded-Method. Finally the embedded method was chosen, utilising the function SelectFromModel.

In order to improve this score at first a Randomised Search Algorithm was applied in order to find the region of the optimal values for the hyperparameters of the SGD-Classifier. The first parameter to optimise is the selection of the used loss function. The default value is "hinge" which applies a linear support vector machine. On top of that we are trying Logistic Regression, a quadratically penalised support vector machine, a modified Huber loss function and the linear loss function which is used by the perceptron algorithm. (See scikit-learn [4], 2022)

Next we define the penalty for the model. The penalty defines the regularisation term which is used to prevent overfitting the model. The added parameters to the randomised search are the l1-, l2-norms, the elastic net (a convex combination of L1 and L2) and "none", which doesn't add a regularisation term. We optimise the alpha-value which controls the regularisation term and the learning rate. A higher alpha results in stronger regularisation which helps to prevent overfitting but also makes the model less complex and therefore potentially less precise. Finally we regulate the elasticnet with the l_1 ratio. The l_1 ratio indicates how the l1 and the l2 penalization are mixed together. The value 0 corresponds to a l2 penalty and the value 1 to a l1 penalty. (scikit-learn [3] & [4], 2022).
After the randomised search was concluded, we utilised a grid search to find the optimal values for the parameters. Finally the parameter values were chosen as follows: For alpha a value of 0.1, for the l1_ratio a value 0.95, for the loss"'modified_huber" and for the penalty "elasticnet" was chosen.

7.3. Annex 3 - Cumulative Gain- and Lift Charts

Lift and cumulative gain charts are measures of model effectiveness and model performance for binary classifiers, as they show the difference between the results which have been obtained with and without the model. They both consist of one baseline curve, which represents a random pick classifier, and at least one curve provided by the used classifier. (See scikit-plot, 2022)

Both charts work in a similar way. First the classification-probabilities need to be retrieved from the classifier. Then a table is created from the validation data, which contains the unique identifier, the actual class and the probabilities of affiliation to one class. This data is sorted in descending order by

the probabilities. The data is then grouped, for example into deciles. Consequently, the cumulative gain chart shows what percentage y of the actual true positive values could be identified with only x% of the data, which shows the highest probability according to our model. (See Figure 12) For the random classifier x=y, therefore the y for the chosen classifier should always be higher than the x. (See Miha, p. 97 pp., 2006)

The lift chart shows the same data from a different perspective.(See Figure 12) Rather than telling us how much percent more of the target group can be found by using the classifier, it tells us how many times more observations which are associated with the target group are in the respective group, in comparison to random choice. So if the lift is 1.8 for the top 20% by probability, it means that we are able to identify 1.8 times more patients than identifying them without any classifier. (scikit-plot, 2022)

8. Annex - Figures & Tables

Figure 1 - Project Plan

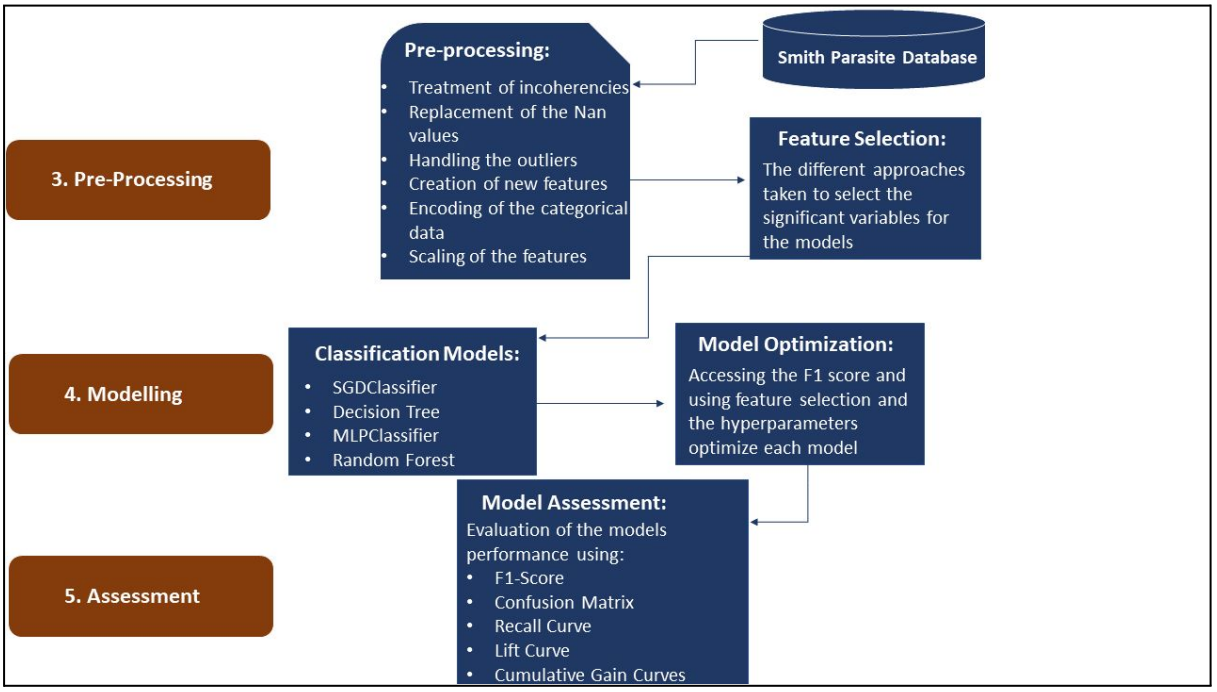


Figure 2 - Spread Map Smith Parasite - United Kingdom

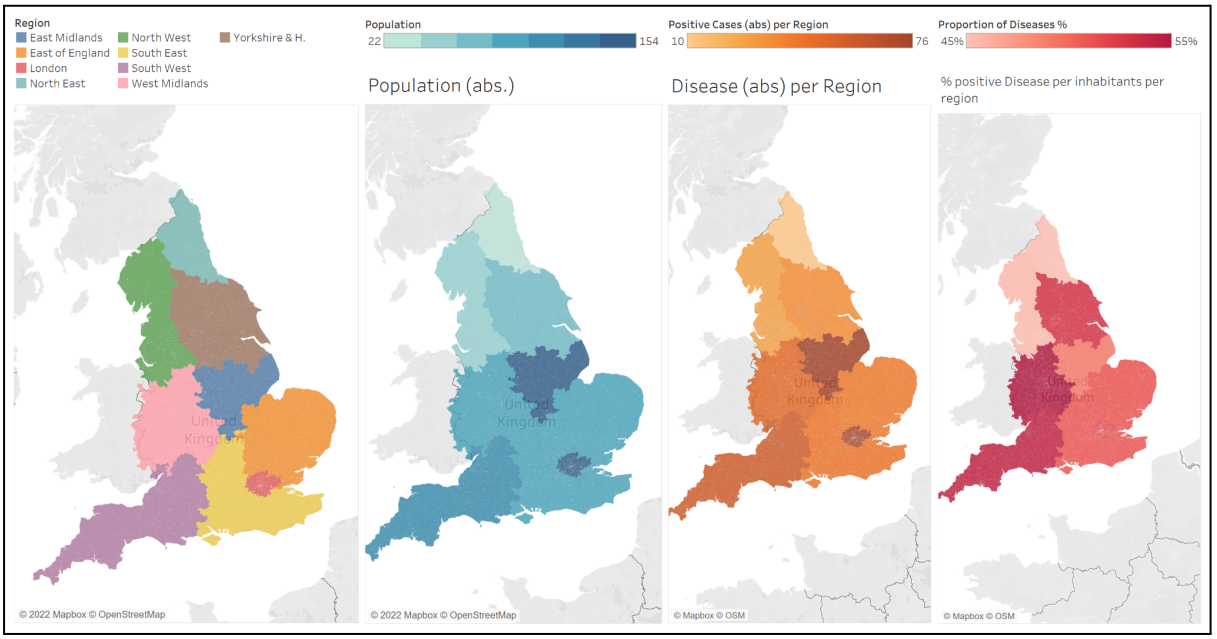


Figure 3 - Boxplots

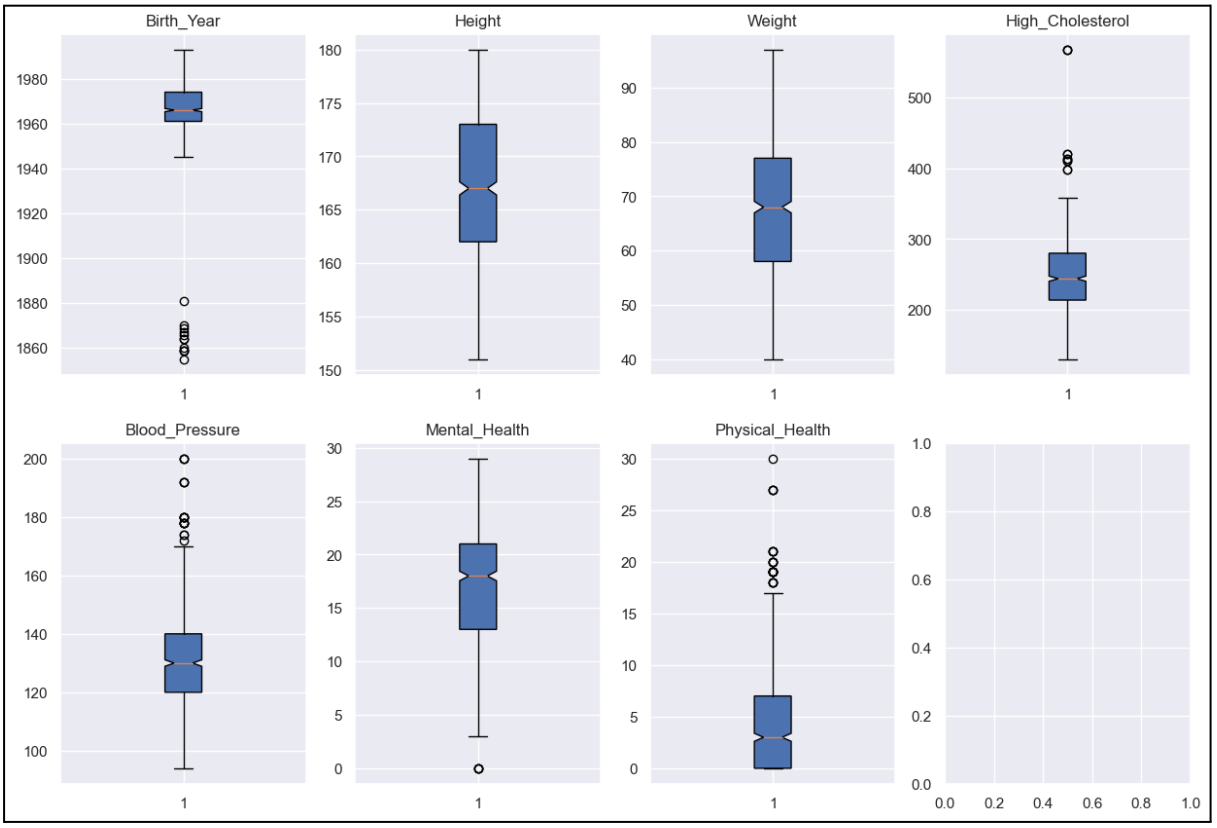


Figure 4 - Pairplot

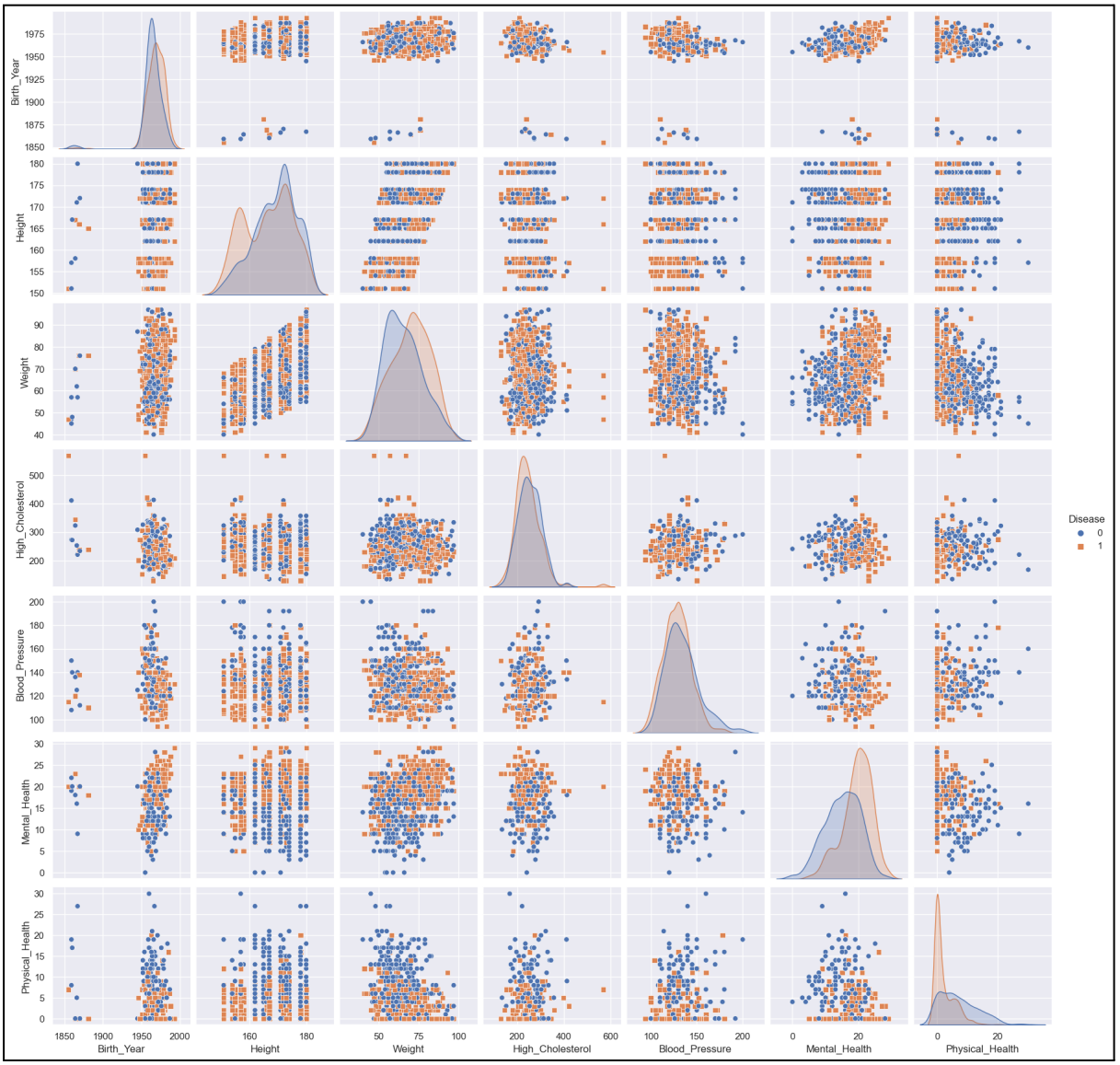
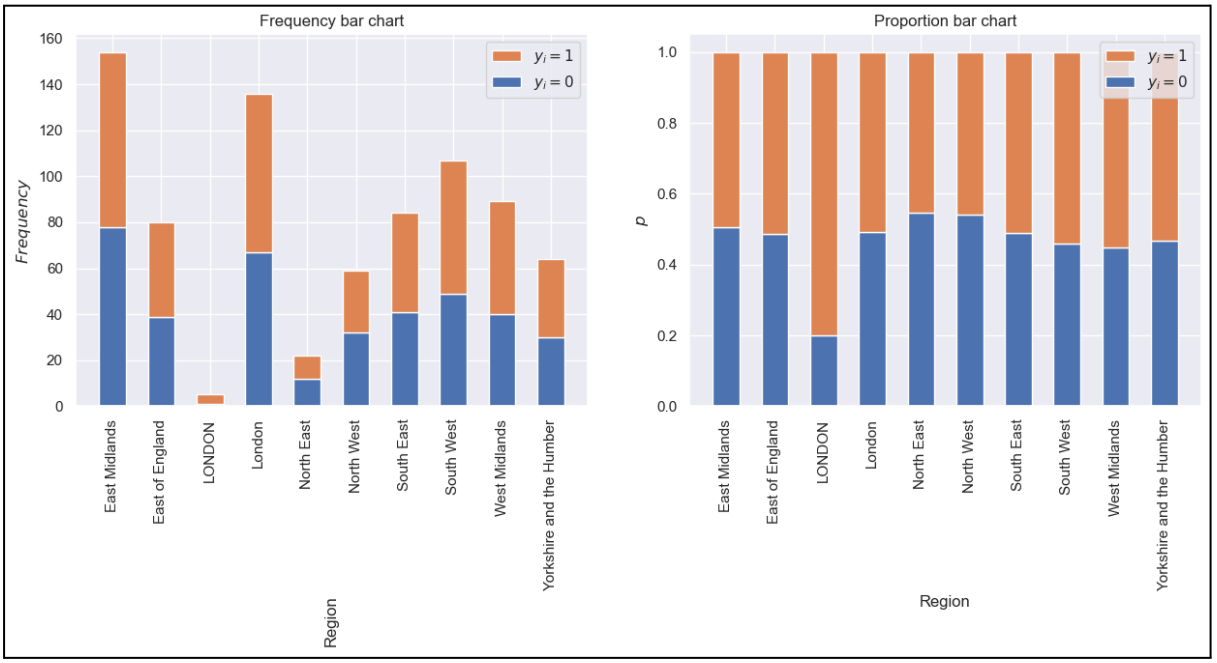
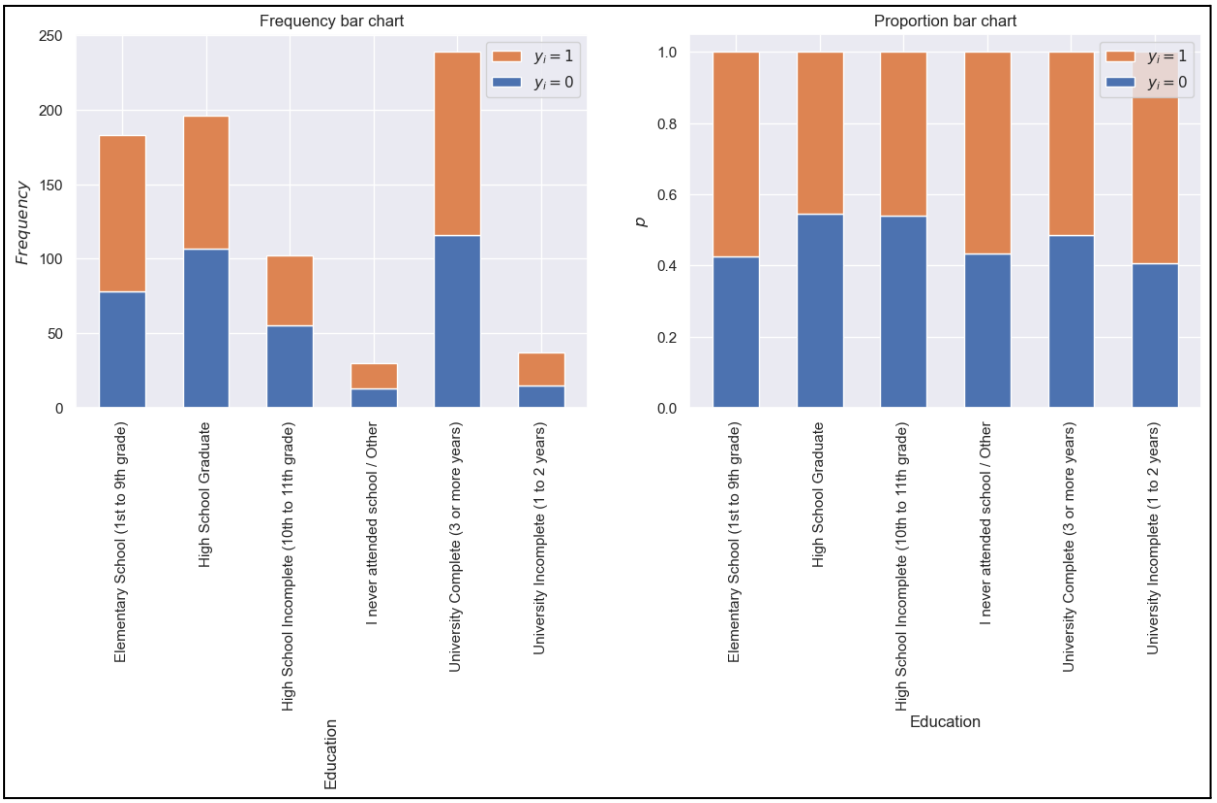
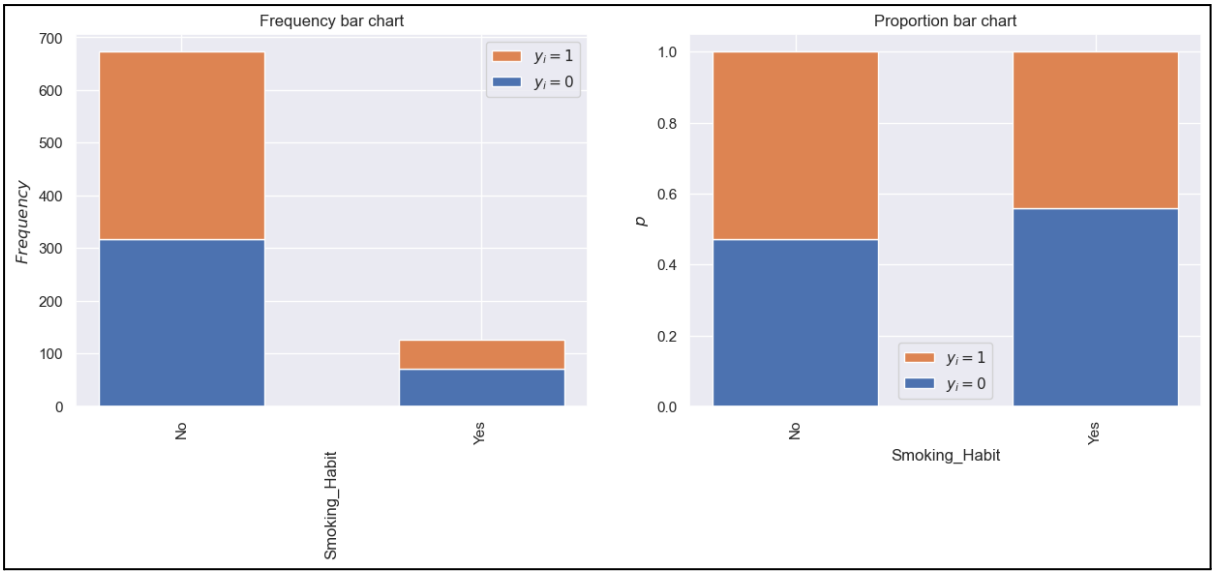
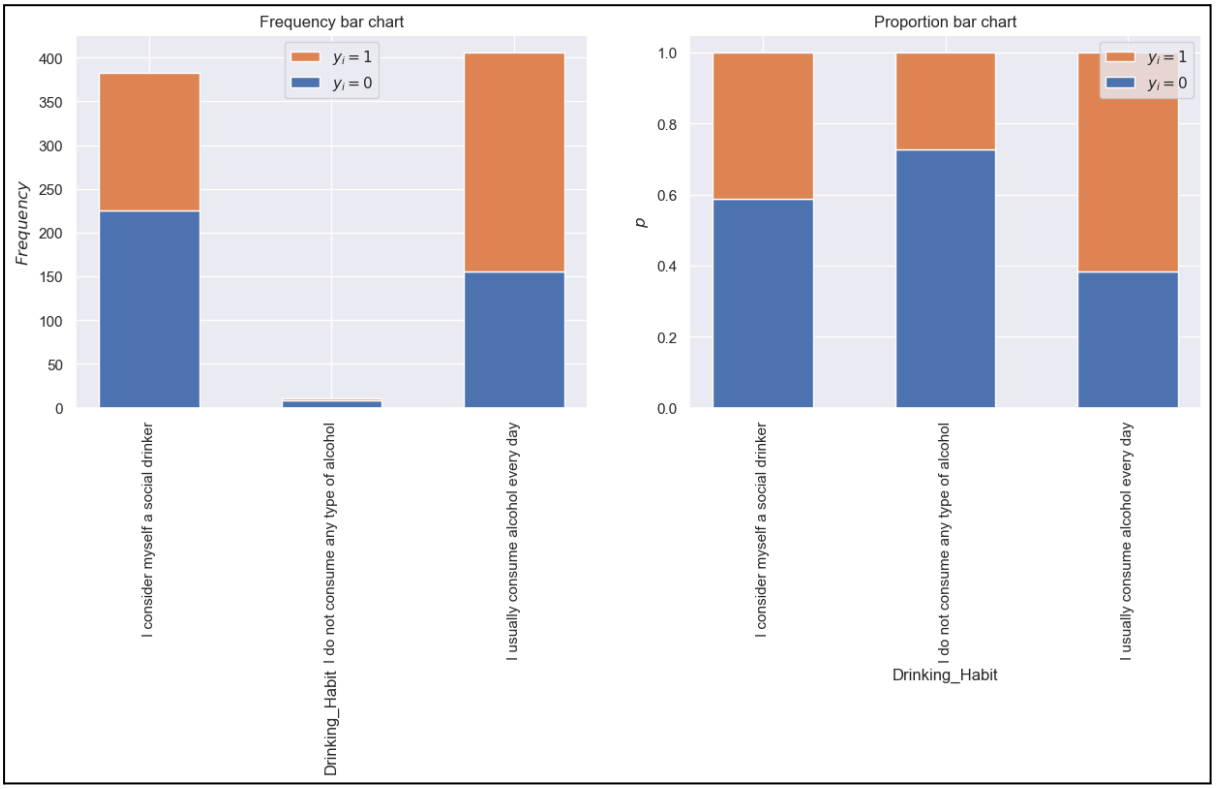


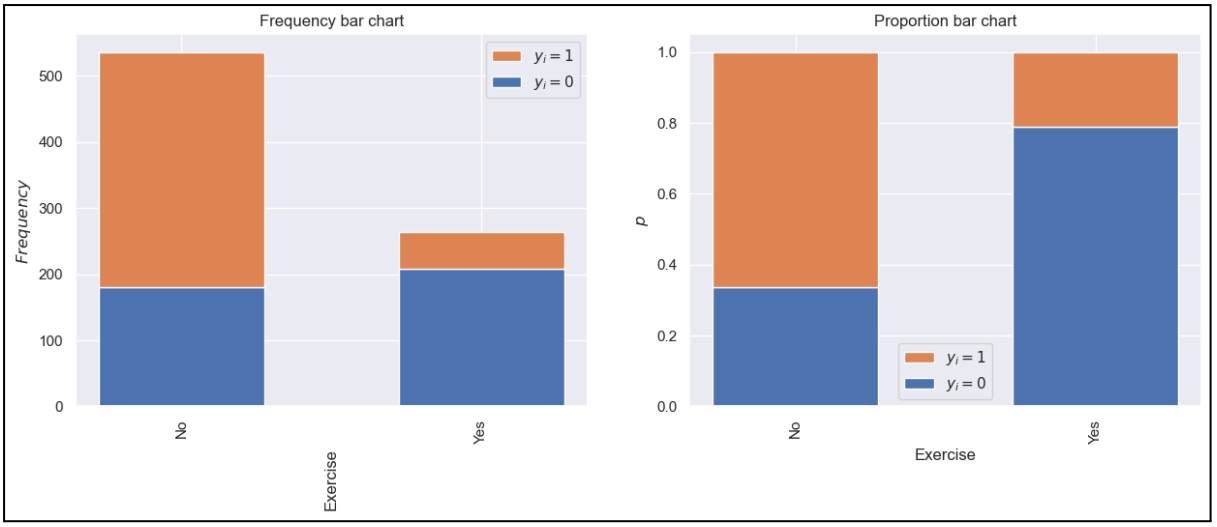
Figure 5 - Barplots

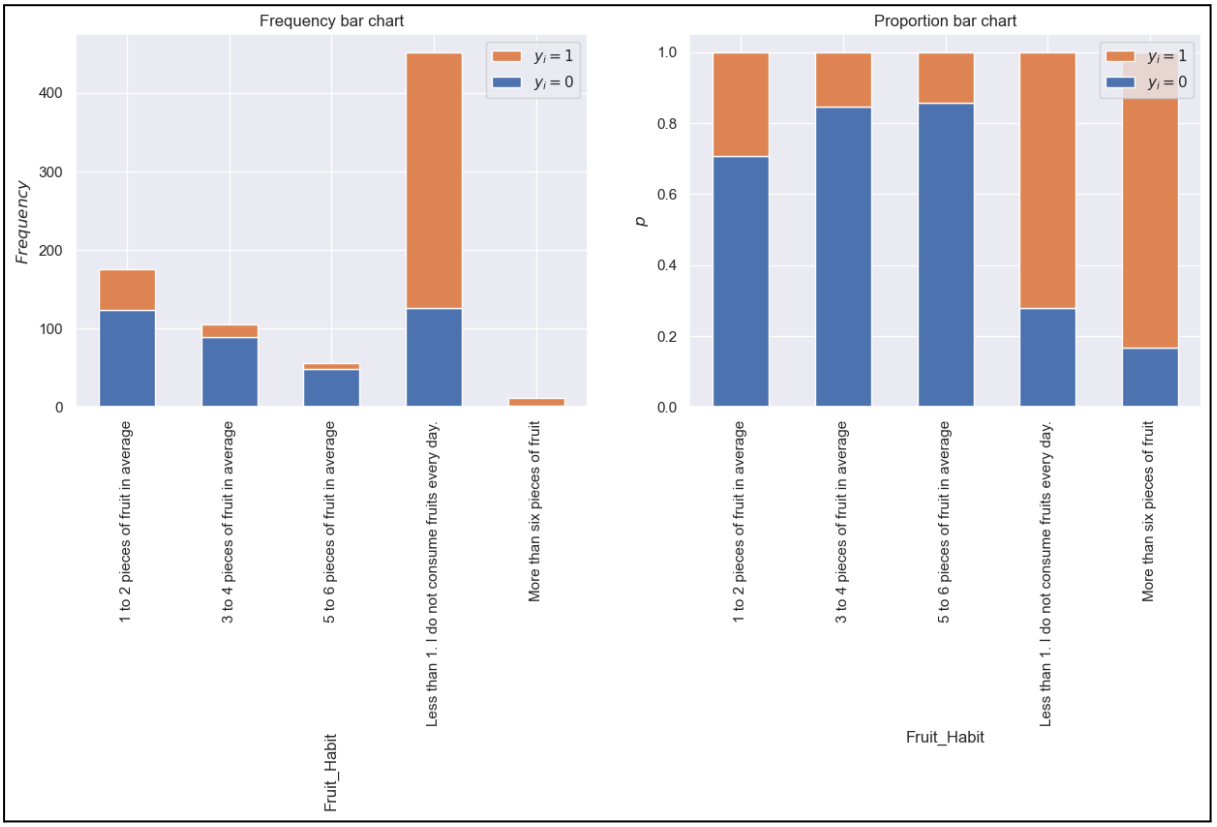


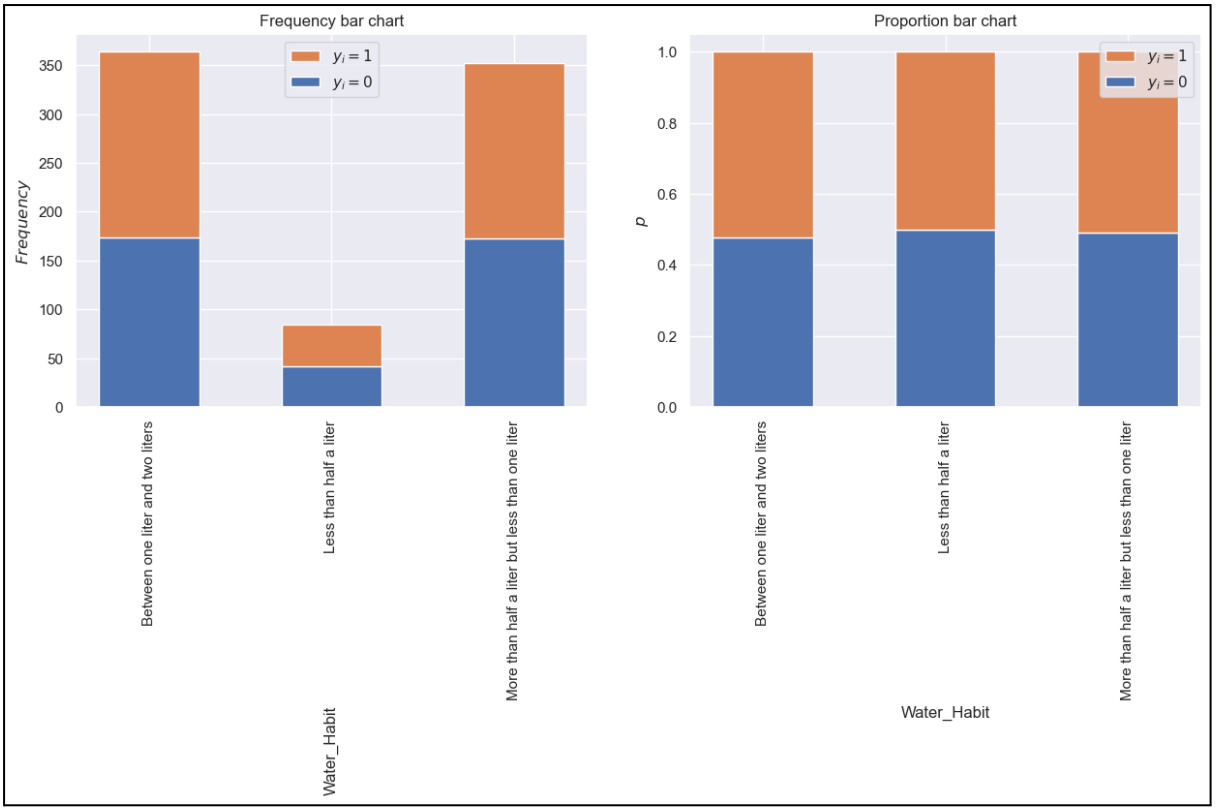


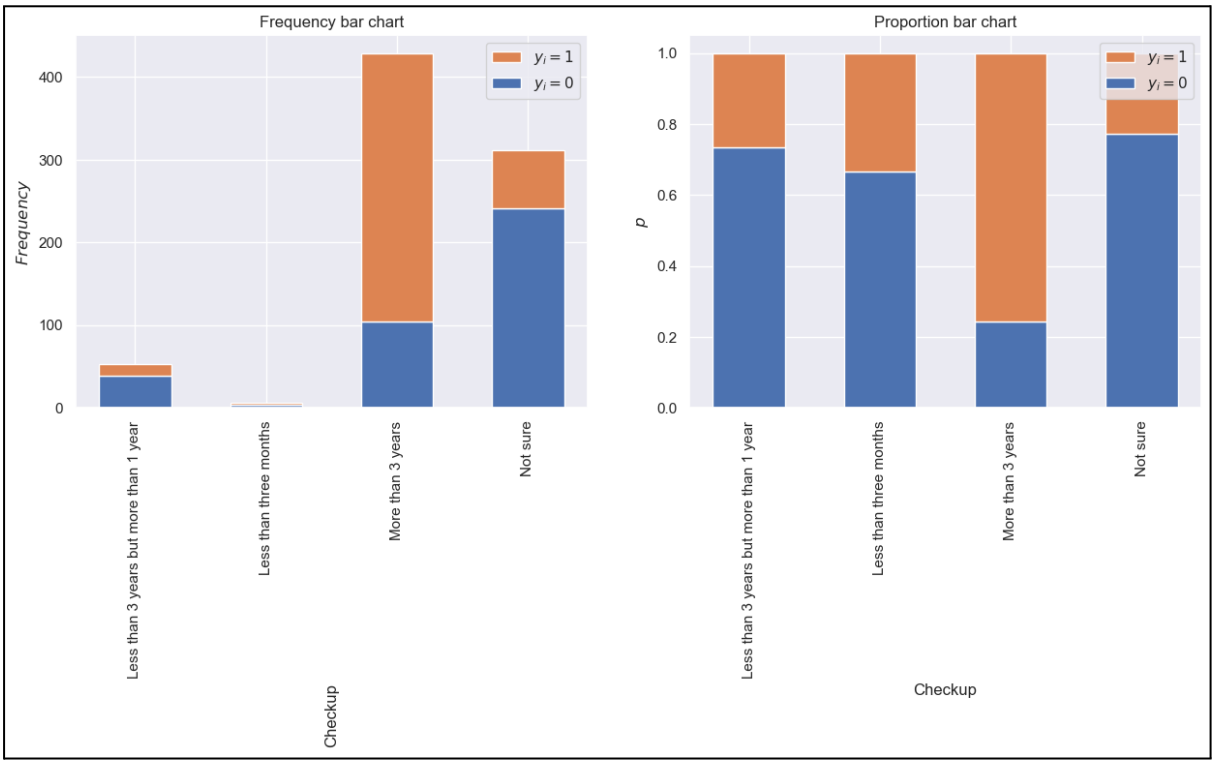

Frequency bar chart
Frequency
700
600
500
400
300
200
100
0
$y_i = 1$
$y_i = 0$
No
Yes
Smoking_Habit
Proportion bar chart
p
1.0
0.8
0.6
0.4
0.2
0.0
$y_i = 1$
$y_i = 0$
No
Yes
Smoking_Habit


Frequency bar chart
Frequency
400
350
300
250
200
150
100
50
0
$y_i = 1$
$y_i = 0$
I consider myself a social drinker
I do not consume any type of alcohol
I usually consume alcohol every day
Drinking_Habit
Proportion bar chart
p
1.0
0.8
0.6
0.4
0.2
0.0
$y_i = 1$
$y_i = 0$
I consider myself a social drinker
I do not consume any type of alcohol
I usually consume alcohol every day
Drinking_Habit


Frequency bar chart
Frequency
Exercise
No
Yes
$y_i = 1$
$y_i = 0$
Proportion bar chart
p
Exercise
No
Yes
$y_i = 1$
$y_i = 0$


Frequency bar chart
Frequency
Fruit_Habit
1 to 2 pieces of fruit in average
3 to 4 pieces of fruit in average
5 to 6 pieces of fruit in average
Less than 1. I do not consume fruits every day.
More than six pieces of fruit
$y_i = 1$
$y_i = 0$
Proportion bar chart
p
Fruit_Habit
1 to 2 pieces of fruit in average
3 to 4 pieces of fruit in average
5 to 6 pieces of fruit in average
Less than 1. I do not consume fruits every day.
More than six pieces of fruit
$y_i = 1$
$y_i = 0$


Frequency bar chart
$y_i = 1$
$y_i = 0$
Frequency
0
50
100
150
200
250
300
350
Between one liter and two liters
Less than half a liter
More than half a liter but less than one liter
Water_Habit
Proportion bar chart
$y_i = 1$
$y_i = 0$
p
0.0
0.2
0.4
0.6
0.8
1.0
Between one liter and two liters
Less than half a liter
More than half a liter but less than one liter
Water_Habit


Frequency bar chart
$y_i = 1$
$y_i = 0$
Frequency
0
100
200
300
400
Less than 3 years but more than 1 year
Less than three months
More than 3 years
Not sure
Checkup
Proportion bar chart
$y_i = 1$
$y_i = 0$
p
0.0
0.2
0.4
0.6
0.8
1.0
Less than 3 years but more than 1 year
Less than three months
More than 3 years
Not sure
Checkup

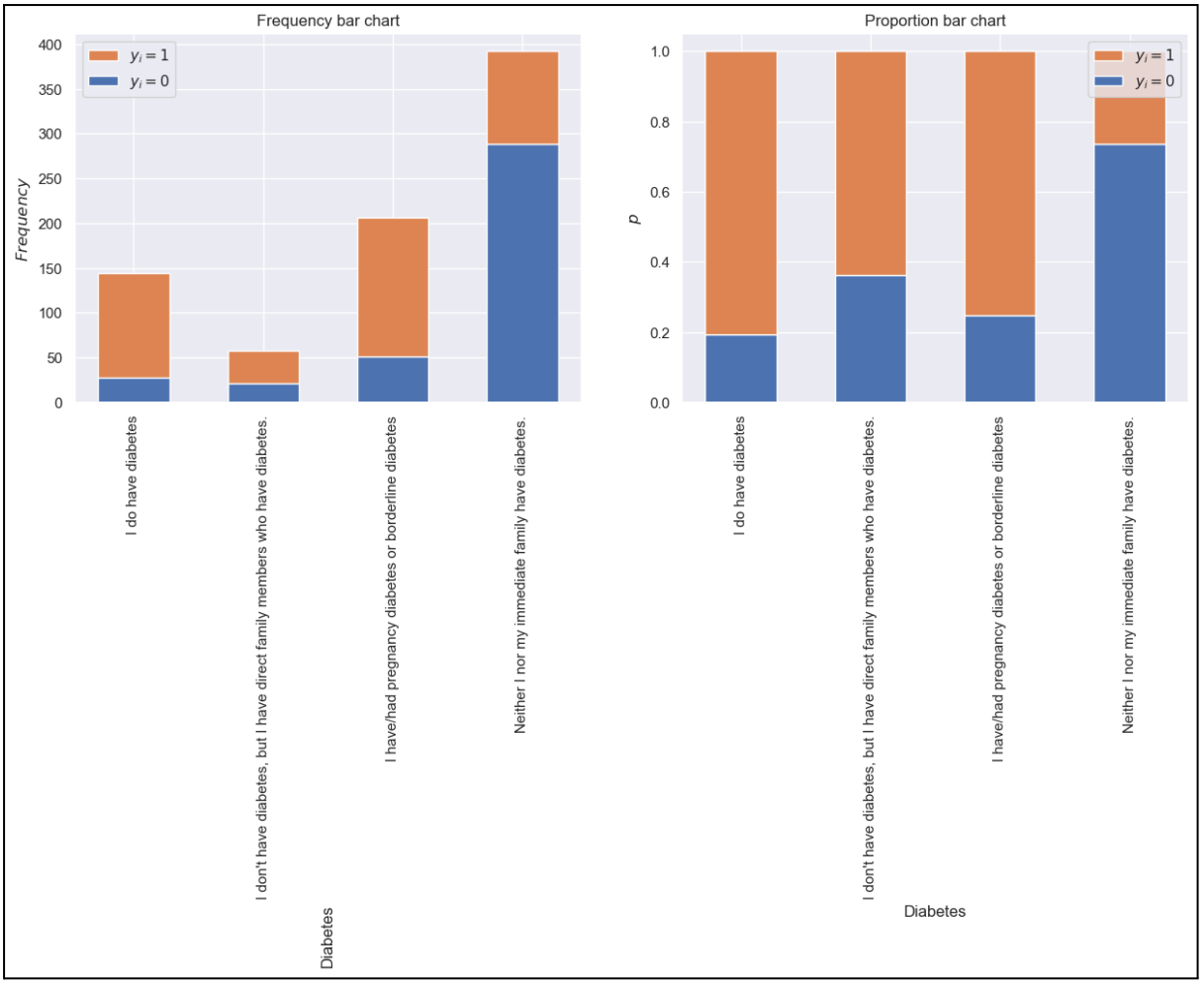

Frequency bar chart
Frequency
400
350
300
250
200
150
100
50
0
$y_i = 1$
$y_i = 0$
I do have diabetes
I don't have diabetes, but I have direct family members who have diabetes.
I have/had pregnancy diabetes or borderline diabetes
Neither I nor my immediate family have diabetes.
Diabetes
Proportion bar chart
p
1.0
0.8
0.6
0.4
0.2
0.0
$y_i = 1$
$y_i = 0$
I do have diabetes
I don't have diabetes, but I have direct family members who have diabetes.
I have/had pregnancy diabetes or borderline diabetes
Neither I nor my immediate family have diabetes.
Diabetes

Figure 6 - Correlation Heatmap

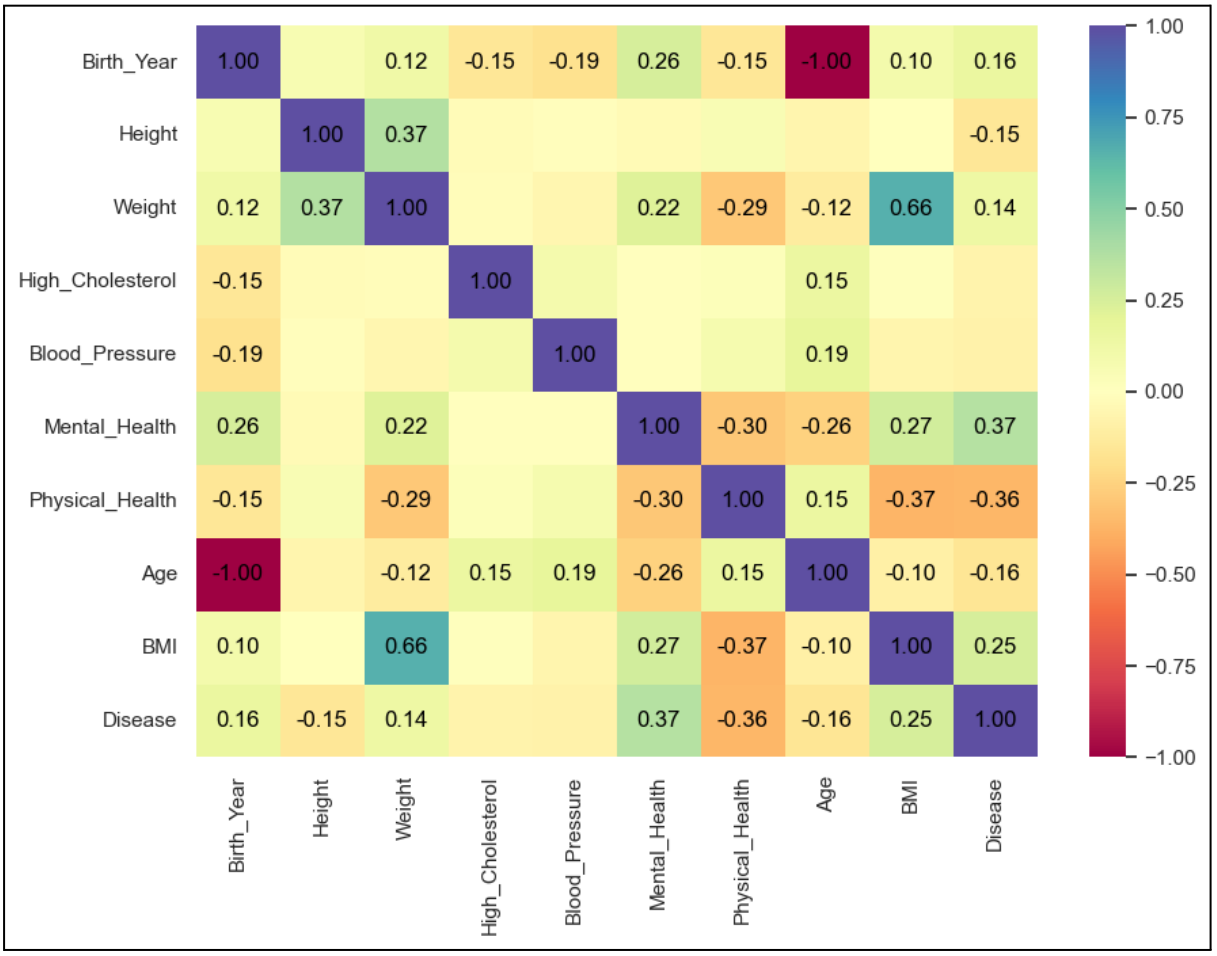


Table 1 - Feature Selection

Filter Method	Wrapper Method (RandomForest)	Embedded Method (RandomForest)
Age	*Age*	*Age*
Weight	*Weight*	*Weight*
Mental_Health	*Mental_Health*	*Mental_Health*
Physical_Health	*Physical_Health*	*Physical_Health*
BMI	*BMI*	*BMI*
Exercise	Exercise	Exercise
Fruit_Habit	Fruit_Habit	Fruit_Habit
Checkup	Checkup	Checkup
Diabetes	Diabetes	Diabetes
Height	*High_Cholesterol*	*High_Cholesterol*
Gender	*Blood_Pressure*	*Blood_Pressure*
BMI_Category		Gender
Drinking_Habit		

(numeric features - italic/cursive)

Figure 7 - Feature Selection - SGDClassifier

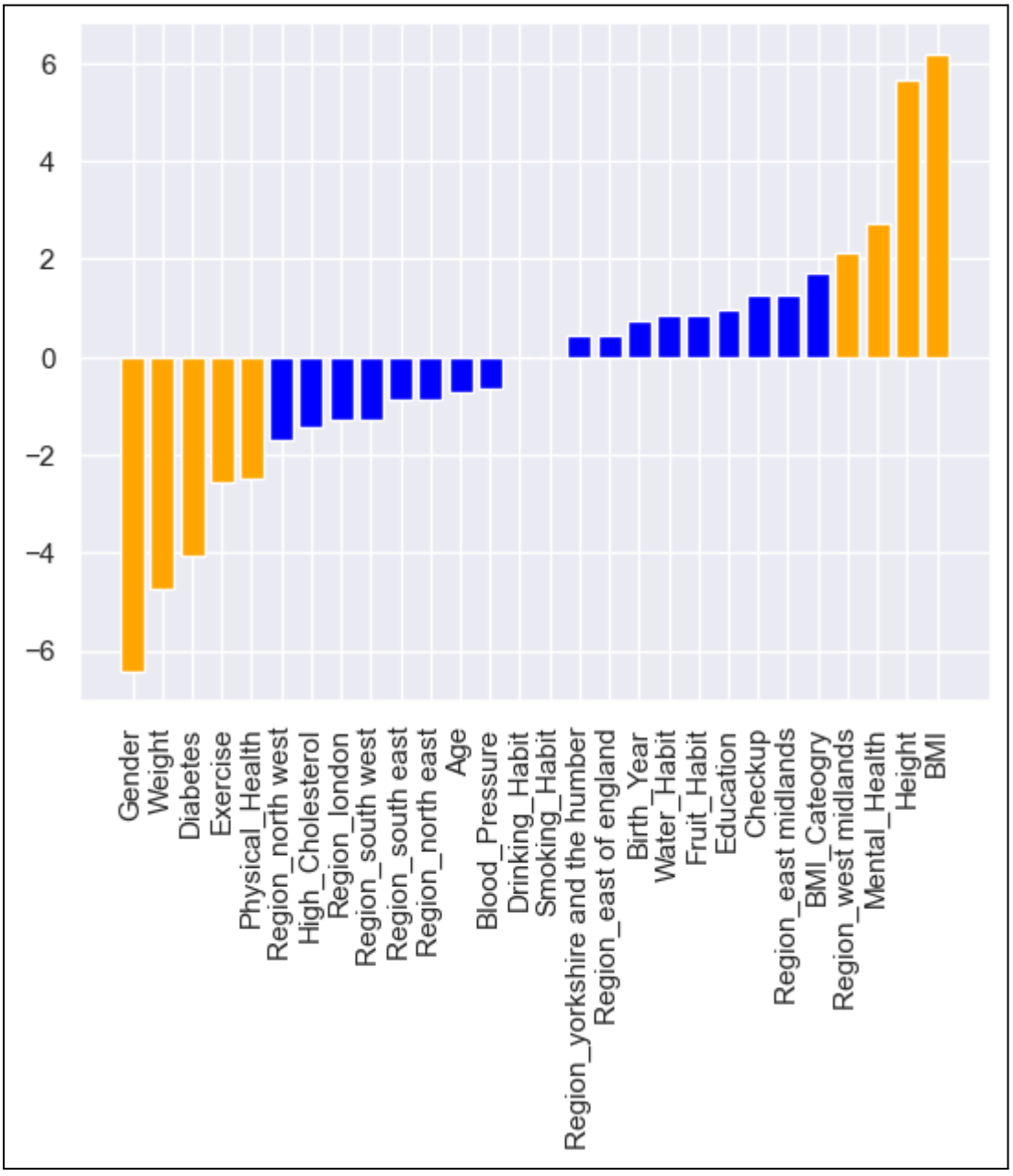


Figure 8 - Feature Selection - DecisionTreeClassifier

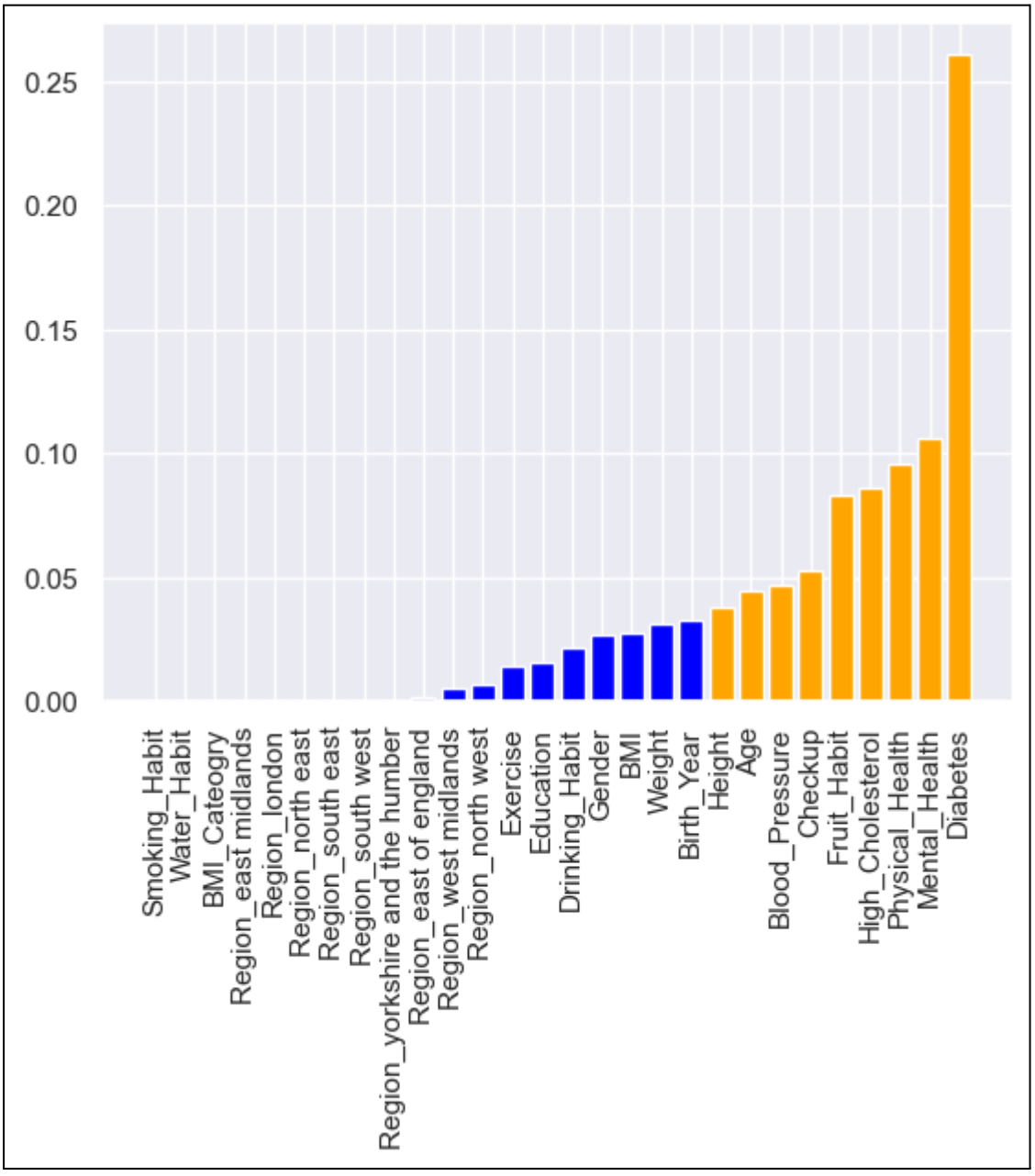


Figure 9 - Feature Selection - RandomForestClassifier

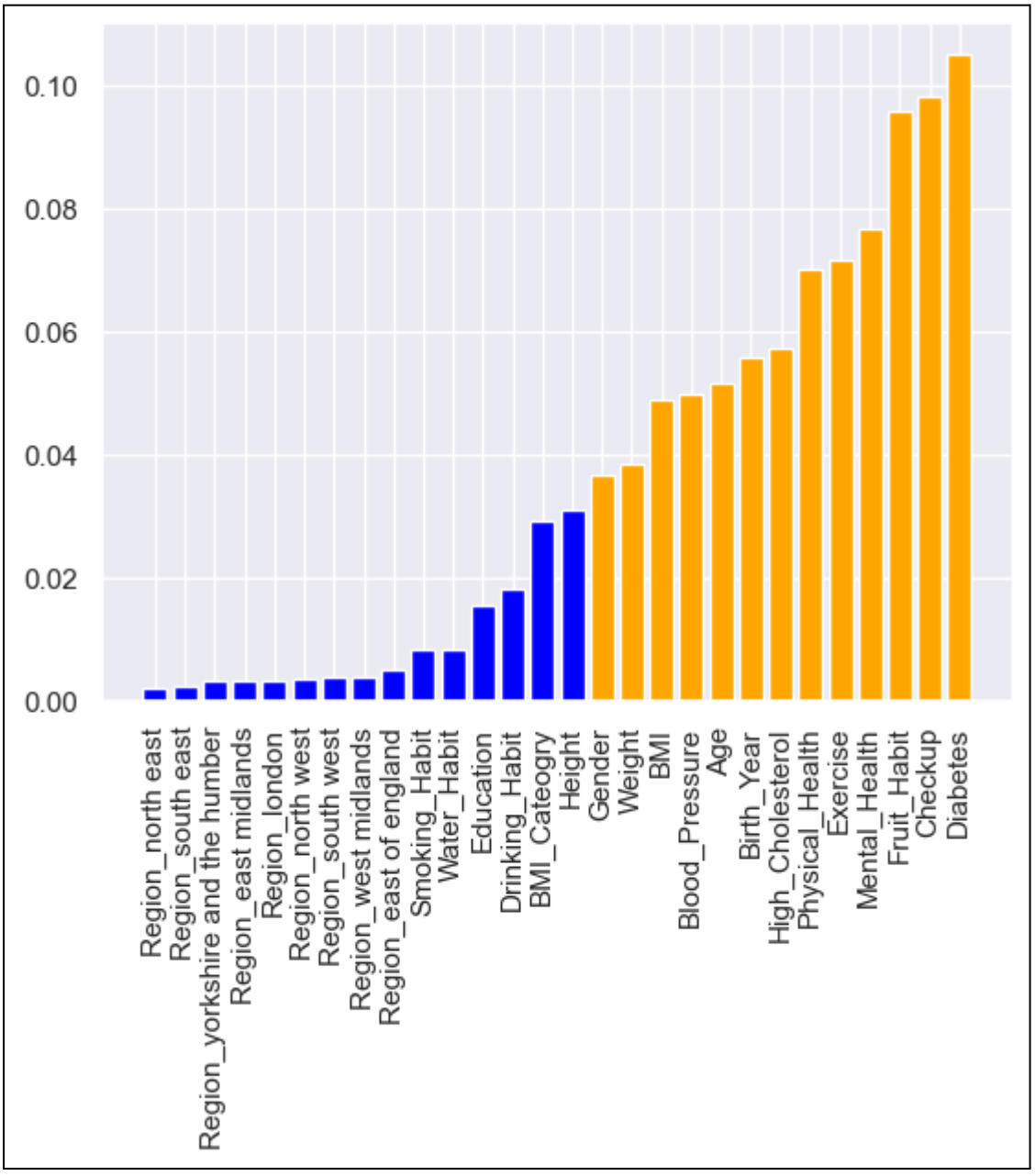


Figure 10 - ROC Curves

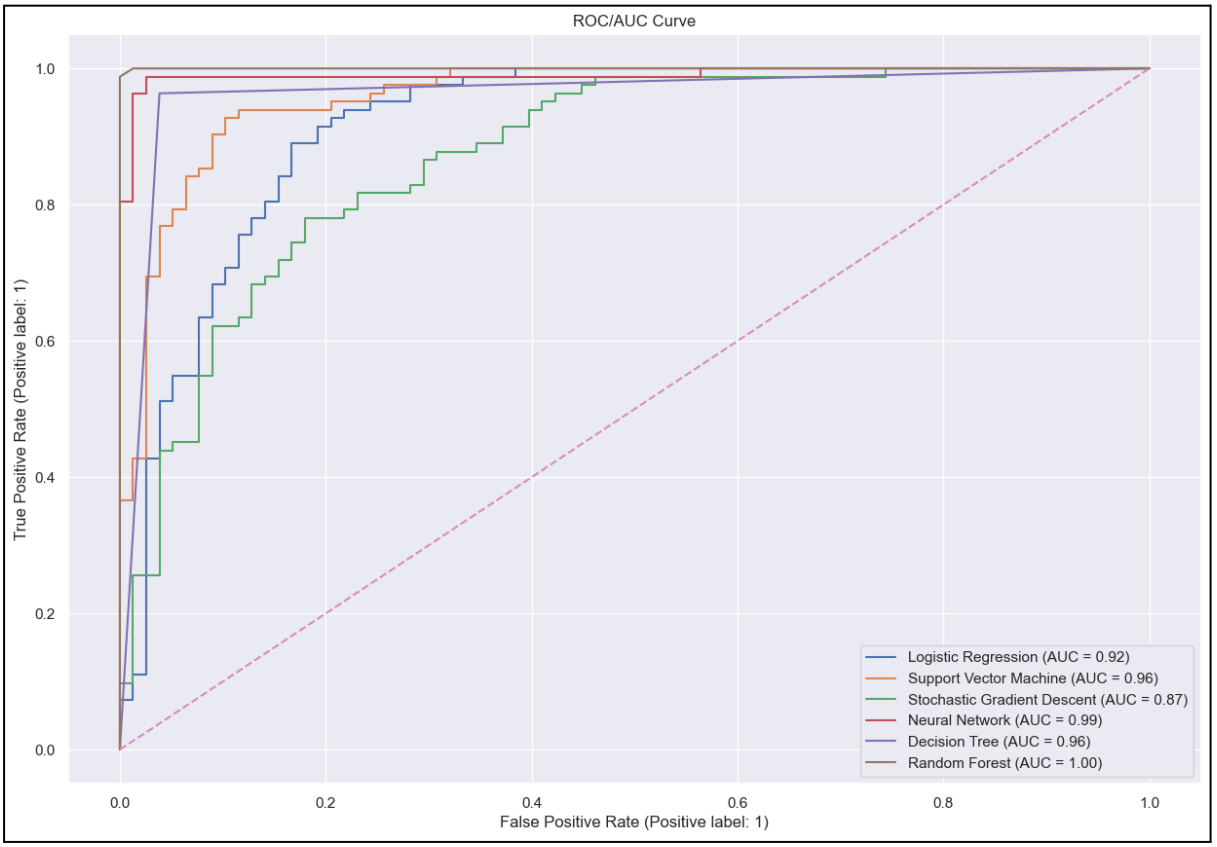


Figure 11 - Precision-Recall Curves

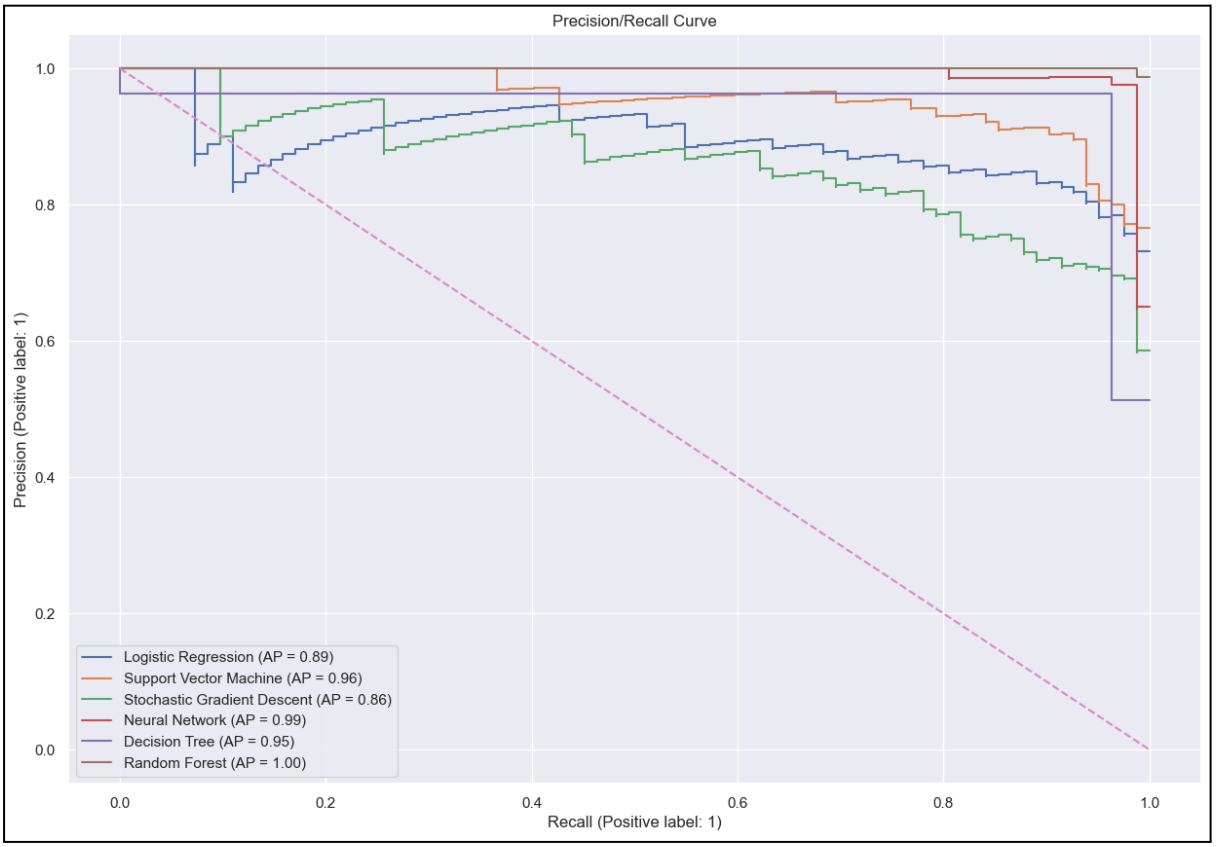
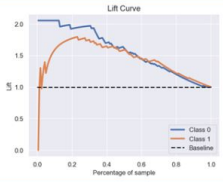
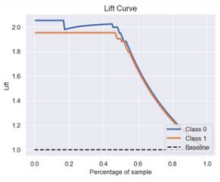
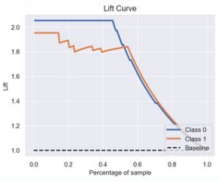
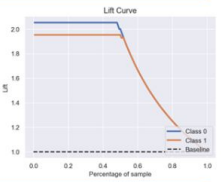
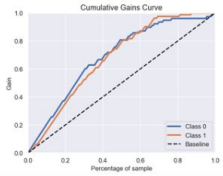
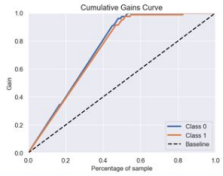
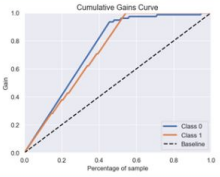
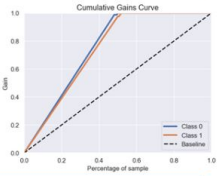


Table 2 - Optimal Hyperparameters - RandomForest

Hyperparameter	Value
max_depth	20
max_features	None
min_samples_leaf	1
min_samples_split	4
n_estimators	450

Figure 12 - Results

	Stochastic Gradient Descent	Neural Network	Decision Tree	Random Forests
F1 - Score	0.7904	0.9467	0.9704	0.9939
Lift				
Cumulative Gains				
Recall Score	0.8049	0.9756	1.0	1.0

Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	59	19
Actual Positive	16	66

	Predicted Negative	Predicted Positive
Actual Negative	71	7
Actual Positive	2	80

	Predicted Negative	Predicted Positive
Actual Negative	73	5
Actual Positive	0	82

	Predicted Negative	Predicted Positive
Actual Negative	77	1
Actual Positive	0	82

9. References

Géron, Aurélien (2022): Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. Concepts, tools, and techniques to build intelligent systems. 3rd ed. Beijing, Boston, Farnham, Sebastopol, Tokyo: O'Reilly.

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011): Data mining. Concepts and techniques. 3rd ed (Online-Ausg.). s.l.: Elsevier professional. Available online at https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=729031
.

Joloudari, Javad Hassannataj; Saadatfar, Hamid; Dehzangi, Abdollah; Shamshirband, Shahaboddin (2019): Computer-aided decision-making for predicting liver disease using PSO-based optimized SVM with feature selection. In Informatics in Medicine Unlocked 17, p. 100255. DOI: 10.1016/j.imu.2019.100255.

Miha Vuk (2006): ROC curve, lift chart and calibration plot. In Advances in Methodology and Statistics. Available online at https://www.semanticscholar.org/paper/ROC-curve%2C-lift-chart-and-calibration-plot-Vuk/ebe9a6b158bb20275e78a6bf35371de6b0523344.

Rahmah, Nadia; Sitanggang, Imas Sukaesih (2016): Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra. In IOP Conf. Ser.: Earth Environ. Sci. 31, p. 12012. DOI: 10.1088/1755-1315/31/1/012012.

scikit-learn [1] (2022): sklearn.feature_selection.SelectFromModel. Available online at https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectFromModel.html, updated on 12/22/2022, checked on 12/23/2022.

scikit-learn [2] (2022): sklearn.cluster.DBSCAN. Available online at https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html, updated on 12/21/2022, checked on 12/21/2022.

Scikit-learn [3] (2022): sklearn.linear_model.SGDClassifier. Available online at https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html#sklearn.linear_model.SGDClassifier, updated on 12/9/2022, checked on 12/11/2022.

scikit-learn [4] (2022) - Stochastic Gradient Descent), updated on 11/20/2022, checked on 12/6/2022. Available online at: https://scikit-learn.org/stable/modules/sgd.html

Scikit-plot (2021): Metrics Module (API Reference) — Scikit-plot documentation. Available online at https://scikit-plot.readthedocs.io/en/stable/metrics.html, updated on 1/29/2021, checked on 12/22/2022.