MACHINE LEARNING FOR PREDICTING DIABETES MELLITUS

A project report submitted in partial fulfilment of the requirement for the degree of

Bachelor in Science
In
Computer Science
by

Alexander Roque Rodrigues

under the supervision of

Ashweta Fondekar
PARVATIBAI CHOWGULE COLLEGE OF ARTS & SCIENCE AUTONOMOUS

June 2019

Declaration by Candidate

I declare that this project report has been prepared by me and to the best of my knowledge, it has not previously formed the basis for the award of any diploma or degree by any other University.

Certificate by Supervisor

Certified that the Project Report is a record of work done by the candidate himself/herself/themselves under my guidance during the period of study and that to the best of my knowledge, it has not previously formed the basis of the award of any degree or diploma of any other University.

Ashweta Fondekar
Project Supervisor

Work Record

1 Acknowledgements

Contents

List of Figures

List of Tables

Abstract

Diabetes Mellitus is a disease that prevents the body from properly expanding the energy stored from the food consumed. The purpose of this project was to select machine learning algorithms that are able to predict or classify a person as diabetic or healthy based on the legacy data. The algorithms compared were KNN Classifier, Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Gradient Boosting Classifier, Support Vector Classifier and the Multi-Layered Perceptron. From all the above the Multi-Layered Perceptron gave an accuracy of prediction on the dataset as 79.70%. To improve the performance of the classifier, I have considered new features deduced from the currently existing feature set a re-trained the classifier on the new dataset that I generated, which is now able to classify the subject as diabetic or healthy with a new accuracy of 93.10%. A significant change which boosted the accuracy by 13.4%. After selection of the algorithm I further advanced the platform of cluster computing to deploy the algorithm onto and generate predictions in without any human interference (apart from entering the data itself) and also made the data available to the users via an easy to use web application which gives them access to the observations then stored in the database after being predicted by the algorithm deployed on the nodes.

2 Introduction

In recent days, there has been a sharp increase in the cases of diabetes mellitus. Diabetes mellitus is on the rise amongst many people and the rate of contracting this lifestyle disease could be reduced significantly if proper measures and precautions were to be instilled amongst people the number of people can be reduced.

Machine learning is a growing field in computer science. With the development and introduction of many algorithms the prediction and accuracy of the predictions itself has improved substantially. Machine learning and healthcare systems are also becoming increasingly popular in the healthcare sector.

The project encompasses the qualities of Remote Patient Monitoring (RPM) and Clinical Decision Support (CDS). RPM provides medical facilities that have the ability to transmit patient data to healthcare professionals who might very well be halfway around the world. RPM can monitor blood glucose levels and blood pressure. It is particularly helpful for patients with chronic conditions such as type 2 diabetes, hypertension, or cardiac disease. Data collected and transmitted via PRM can be used by a healthcare professional or a healthcare team to detect medical events such as stroke or heart attack that require immediate and aggressive medical intervention. Data collected may be used as part of a research project or health study. RPM is a life-saving system for patients in remote areas who cannot access face-to-face health care. CDS analyzes data from clinical and administrative systems. The aim is to assist healthcare providers in making informed clinical decisions. Data available can provide information to medical professions who are preparing diagnoses or predicting medical conditions like drug interactions and reactions. CDS tools filter information to assist healthcare professionals in caring for individual clients.

The objective of this project is to create a system that is able to use the machine learning algorithms and predict the outcome of the parameters entered into the algorithm and help the patient draw a conclusion whether or not he/she has the same traits exhibited by similar patients that have diabetes. Also the system should have a UI that is capable of displaying the data of the patients to the doctor and to the patients themselves for further interpretation.

3 Software Requirements

The main function of this project is to enable the doctors to advise their patients with the help of the prediction software. The system should be accessible to the patient as well as the doctor, therefore it should have a web interface for the two parties to interact with.

3.1 Functionalities for Doctors

As an owner of a doctors account a doctor should be able to:

- Add new observations for the machine learning algorithm to predict.
- Analyse the patients previous records.
- Have a dashboard for patient performance monitoring.

3.2 Functionalities for Patients

As the patient, one should be able to:

- Should be able to see the predicted risk of developing diabetes.
- Should be able to view historic data.

3.3 Functionalities for Master Nodes

As the master nodes:

- Control chunk size.
- Remotely update the slave nodes.
- Check the status of nodes.
- Control the SQL database.

3.4 Functionalities for Slave Nodes

- Should have the machine learning algorithm tuned as per specifications.
- Remote update should be possible.
- Remote access should be possible.

4 Hardware Requirements

4.1 Raspberry Pi

According to raspberrypi.org, the Raspberry Pi 3 Model B is the earliest model of the third-generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016. Some of the key features of this single board computer or SBC are:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU.
- 1GB RAM.
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board.
- 100 Base Ethernet.
- 40-pin extended GPIO.
- 4 USB 2 ports.
- Full size HDMI.
- Micro SD port for loading the operating system and storing data.

In this project I will be using 3 Raspberry Pi's to implement a cluster and deploy the machine learning algorithm on.

4.2 Switch

A network switch was used in the project to make up for the lack of ethernet ports available on the router.

4.3 Router

A router is required to assign internet protocol addresses to the nodes using dynamic host control protocol (DHCP). The router also is responsible for displaying the nodes connected to the network thereby displaying host names and making it more easier to capture the addresses of each node.

4.4 Master Node

The master node is assigned the task of managing the the slave nodes connected to the network. The master node and the slave nodes should be connected to the same database to run and execute queries. The master node shall also be assigned with the task of killing a particular process or shutting down a particular node if required.

4.5 Slave Node

The slave nodes are to be configured with the selected algorithm and are the most vital part of the structure. The slave nodes will be responsible for receiving instructions from the master node and will be responsible for learning from the dataset and then can generate predicted outcomes for the patient records received from the master.

4.6 Database

The database will store all the records for the patients and doctors. The database is a SQL database that will not be subjected to a change in the database schema structure for consistency.

4.7 Web Server

The web server provides an interface for the doctor and the patients to read the predictions and legacy data of the patient from the website.

5 Technology Stack

5.1 Python

5.1.1 Pandas

Pandas, is a library that is required for loading the comma separated value file into python. Pandas is a package for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

5.1.2 Sklearn

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to inter-operate with the Python numerical and scientific libraries NumPy and SciPy.

5.1.3 Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy is open-source software and has many contributors.

5.1.4 Itertools

The itertools module standardizes a core set of fast, memory efficient tools that are useful by themselves or in combination. Together, they form an "iterator algebra" making it possible to construct specialized tools succinctly and efficiently in pure Python.

5.2 MYSQL

MySQL is an open-source relational database management system. MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter and YouTube.

5.3 Apache Web Server

The Apache HTTP Server, colloquially called Apache, is free and open-source cross-platform web server software, released under the terms of Apache License 2.0. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The vast majority of Apache HTTP Server instances run on a Linux distribution, but current versions also run on Microsoft Windows and a wide variety of Unix-like systems. Past versions also ran on OpenVMS, NetWare, OS/2 and other operating systems, including ports to mainframes.

5.4 PHP

PHP is a general-purpose programming language originally designed for web development. PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor. PHP code may be executed with a command line interface (CLI), embedded into HTML code, or used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in a web server or as a Common Gateway Interface (CGI) executable. The web server outputs the results of the interpreted and executed PHP code, which may be any type of data, such as generated HTML code or binary image data. PHP can be used for many programming tasks outside of the web context, such as standalone graphical applications and robotic drone control.

5.5 AJAX

Ajax is a set of web development techniques using many web technologies on the client side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously (in the background) without interfering with the display and behavior of the existing page. By decoupling the data interchange layer from the presentation layer, Ajax allows web pages and, by extension, web applications, to change content dynamically without the need to reload the entire page.[3] In practice, modern implementations commonly utilize JSON instead of XML.

Ajax is not a single technology, but rather a group of technologies. HTML and CSS can be used in combination to mark up and style information. The webpage can then be modified by JavaScript to dynamically display—and allow the user to interact with—the new information. The built-in XMLHttpRequest object, or since 2017 the new ”fetch()” function within JavaScript, is commonly used to execute Ajax on web pages allowing websites to load content onto the screen without refreshing the page. Ajax is not a new technology, or different language, just existing technologies used in new ways.

5.6 GitHub

GitHub is a global company that provides hosting for software development version control using Git. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

5.7 SSH

Secure Shell is a cryptography network protocol for operating network services securely over an unsecured network. Typical applications include remote command-line, login, and remote command execution, but any network service can be secured with SSH. The SSH is used to communicate between the master and slave nodes.

6 Selecting an Algorithm

The dataset used for this project is from the UCI Machine learning repository and can be found at their website. Table 2 is an outcome of the accuracy of the algorithms subjected to train and predict the outcomes for given sets of patients. After analysis of multiple papers, I have seen that many data scientists have trained and tested the algorithms with the raw, untreated datasets, thereby causing the model to perform poorly. After much inspection I was able to remove the null values from the dataset which were leading it to reduce the model performance. After

7 The Multi Layered Perceptron

7.1 Introduction

The multi layer perceptron (MLP) is the one of the most commonly used artificial neural networks. A multilayer perceptron is not a single perceptron with multiple layers, but rather multiple layers of artificial neurons that can be perceptrons. The layers of the MLP form a directed, acyclic graph. Generally, each layer is fully connected to the subsequent layer; the output of each artificial neuron in a layer is an input to every artificial neuron in the next layer towards the output. MLPs have three or more layers of artificial neurons.

8 Coding Process

The concept of the system remains simple. The system can be described using the following diagrams.

8.1 Designing the System

- The system firstly reads the records from the database that are not classified as 0 or 1. The master node then converts all the results obtained from the database into chunks as specified by the code.

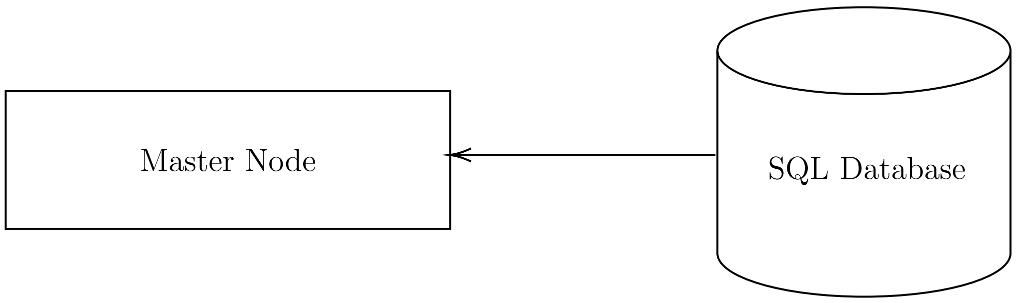
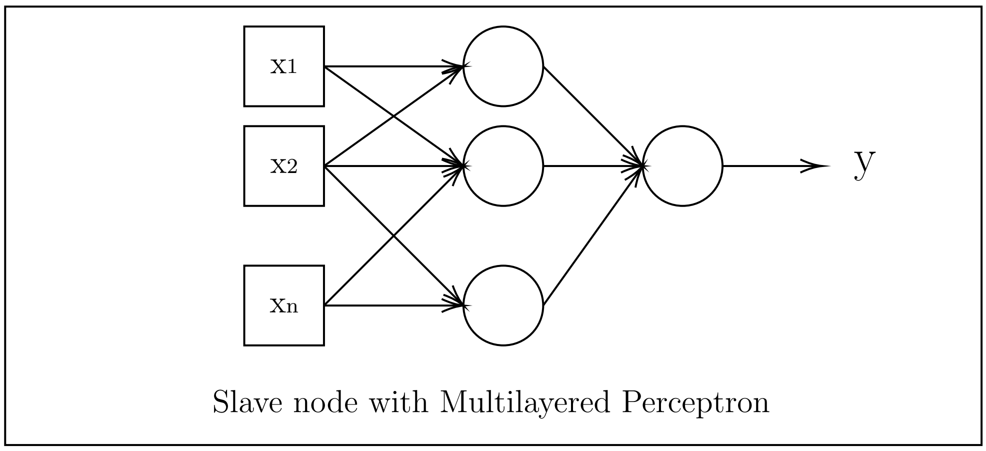
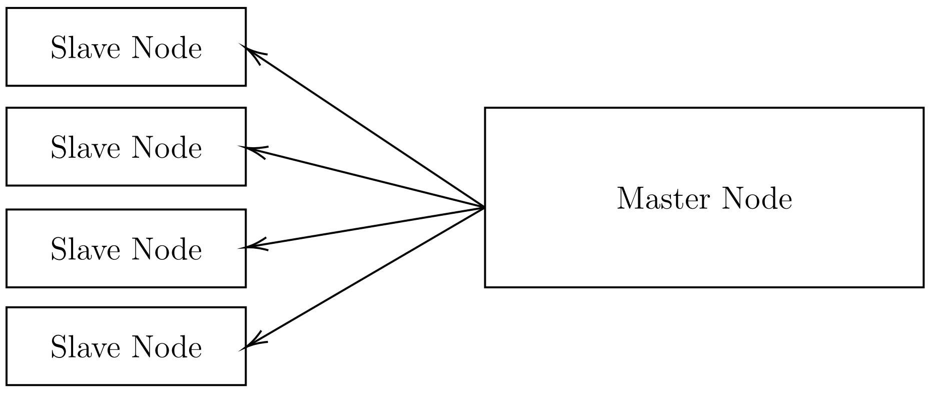


Figure 1: Master Node reading data from the SQL Database.

- odei



- The chunks containing the data of the queries that were left unpredicted via the cluster are broken up via the chunk size set in the master environment and then are sent to the cluster in a round robin fashion. The method of sending the queries can be set to either round robin fashion or all at once. The



9 Deploying to Cluster

A cluster is a group of two or more servers connected to each other in such a way that they behave like a single server. Each machine in the cluster is called a node. Because each machine in the cluster runs the same services as other machines in the cluster, any machine can stand in for any other machine in the cluster. This becomes important when one machine goes down or must be taken out of service for a time. The remaining machines in the cluster can seamlessly take over the work of the downed machine, providing users with uninterrupted access to services and data.

9.1 Benefits of Clustering

- Increased resource availability: If one Intelligence Server in a cluster fails, the other Intelligence Servers in the cluster can pick up the workload. This prevents the loss of valuable time and information if a server fails. Strategic resource usage: You can distribute projects across nodes in whatever configuration you prefer. This reduces overhead because not all machines need to be running all projects, and allows you to use your resources flexibly.
    - Increased performance: Multiple machines provide greater processing power.
    - Greater scalability: As your user base grows and report complexity increases, your resources can grow.
    - Simplified management: Clustering simplifies the management of large or rapidly growing systems.
    - Fail-over support ensures that a business intelligence system remains available for use if an application or hardware failure occurs. Clustering provides failover support in two ways:

      Load redistribution: When a node fails, the work for which it is responsible is directed to another node or set of nodes. Request recovery: When a node fails, the system attempts to reconnect MicroStrategy Web users with queued or processing requests to another node. Users must log in again to be authenticated on the new node. The user is prompted to resubmit job requests.

Load Balancing Load balancing is a strategy aimed at achieving even distribution of user sessions across Intelligence Servers, so that no single machine is overwhelmed. This strategy is especially valuable when it is difficult to predict the number of requests a server will receive. MicroStrategy achieves four-tier load balancing by incorporating load balancers into the MicroStrategy Web and Web products.

Load is calculated as the number of user sessions connected to a node. The load balancers collect information on the number of user sessions each node is carrying. Using this information at the time a user logs in to a project, MicroStrategy Web connects them to the Intelligence Server node that is carrying the lightest session load. All requests by that user are routed to the node to which they are connected until the user disconnects from the MicroStrategy Web product.

Project Distribution and Project Fail over When you set up several server machines in a cluster, you can distribute projects across those clustered machines or nodes in any configuration, in both Windows and Linux environments. All servers in a cluster do not need to be running all projects. Each node in the cluster can host a different set of projects, which means only a subset of projects need to be loaded on a specific Intelligence Server machine. This feature provides you with flexibility in using your resources, and it provides better scalability and performance because of less overhead on each Intelligence Server machine.

Distributing projects across nodes also provides project fail-over support. For example, one server is hosting project A and another server is hosting projects B and C. If the first server fails, the other server can host all three projects to ensure project availability.

Project creation, duplication, and deletion in a three-tier, or server, connection are automatically broadcast to all nodes during run-time to ensure synchronization across the cluster.

Work Fencing User fences and workload fences allow you to reserve nodes of a cluster for either users or a project subscriptions.

10 Future Scope

11 Tables

Column Name	Description
Pregnancies	Number of times pregnant.
Glucose	Plasma glucose concentration a 2 hours in an oral glucose tolerance test.
Blood Pressure	Diastolic blood pressure (mm Hg)
Skin Thickness	Triceps skin fold thickness (mm)
Insulin	2-Hour serum insulin (muU/ml)
Body Mass Index	Body mass index (weight in kg/(height in m)ˆ2)
Diabetes Pedigree Function	Diabetes pedigree function
Age	Age (years)
Outcome	Class variable (0 or 1) 268 of 768 are 1, the others are 0

Table 1: Pima Indians dataset header.

Algorithm	Additional Parameters	Train Set Accuracy
K Nearest Neighbour	-	0.79
Logistic Regression	C = 1	0.781
Logistic Regression	C = 0.01	0.700
Logistic Regression	C = 100	0.785
Decision Tree	-	1.00
Decision Tree	Max Depth = 3	0.773
Random Forest	Estimators = 100	1.000
Random Forest	Estimators = 100; Max Depth = 3	0.800
Gradient Boosting	-	0.917
Gradient Boosting	Max Depth = 1	0.804
Gradient Boosting	Learning Rate = 0.01	0.802
Support Vector Machine	-	1.00
Support Vector Machine	Train and Test set scaled using MinMaxScaler	0.77
Support Vector Machine	C = 1000	0.790
MLP Classifier	Random State = 42	0.73
MLP Classifier	Random State = 0	0.823
MLP Classifier	Max Iterations = 1000	0.908
MLP Classifier	Max Iterations = 1000; Alpha = 1; Random State = 0	0.806

Table 2: Test and Train accuracy's using various machine learning algorithms with various parameters.

Algorithm	Additional Parameters	Test Set Accuracy
K Nearest Neighbour	-	0.78
Logistic Regression	C = 1	0.771
Logistic Regression	C = 0.01	0.703
Logistic Regression	C = 100	0.766
Decision Tree	-	0.714
Decision Tree	Max Depth = 3	0.740
Random Forest	Estimators = 100	0.786
Random Forest	Estimators = 100; Max Depth = 3	0.755
Gradient Boosting	-	0.792
Gradient Boosting	Max Depth = 1	0.781
Gradient Boosting	Learning Rate = 0.01	0.776
Support Vector Machine	-	0.65
Support Vector Machine	Train and Test set scaled using MinMaxScaler	0.77
Support Vector Machine	C = 1000	0.797
MLP Classifier	Random State = 42	0.72
MLP Classifier	Random State = 0	0.802
MLP Classifier	Max Iterations = 1000	0.792
MLP Classifier	Max Iterations = 1000; Alpha = 1; Random State = 0	0.797

Table 3: Test and Train accuracy's using various machine learning algorithms with various parameters.

12 Appendix I

12.1 K Nearest Neighbours

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='
    uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski',
     metric_params=None, n_jobs=None, **kwargs)
```

12.2 Logistic Regression

```
class sklearn.linear_model.LogisticRegression(penalty='l2', dual=False
    , tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1,
    class_weight=None, random_state=None, solver='lbfgs', max_iter
    =100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None
    , l1_ratio=None)
```

12.3 Decision Tree

```
class sklearn.tree.DecisionTreeClassifier(criterion='gini', splitter='
    best', max_depth=None, min_samples_split=2, min_samples_leaf=1,
    min_weight_fraction_leaf=0.0, max_features=None, random_state=None
    , max_leaf_nodes=None, min_impurity_decrease=0.0,
    min_impurity_split=None, class_weight=None, presort='deprecated',
    ccp_alpha=0.0)
```

12.4 Random Forest Classifier

```
class sklearn.ensemble.RandomForestClassifier(n_estimators=100,
    criterion='gini', max_depth=None, min_samples_split=2,
    min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='
    auto', max_leaf_nodes=None, min_impurity_decrease=0.0,
    min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=
    None, random_state=None, verbose=0, warm_start=False, class_weight
    =None, ccp_alpha=0.0, max_samples=None)
```

12.5 Gradient Boosting

```
class sklearn.ensemble.GradientBoostingClassifier(loss='deviance',
    learning_rate=0.1, n_estimators=100, subsample=1.0, criterion='
    friedman_mse', min_samples_split=2, min_samples_leaf=1,
    min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease
    =0.0, min_impurity_split=None, init=None, random_state=None,
    max_features=None, verbose=0, max_leaf_nodes=None, warm_start=
    False, presort='deprecated', validation_fraction=0.1,
    n_iter_no_change=None, tol=0.0001, ccp_alpha=0.0)
```

12.6 Multi Layered Perceptron

```
class sklearn.neural_network.MLPClassifier(hidden_layer_sizes=(100, ),
     activation='relu', solver='adam', alpha=0.0001, batch_size='auto
    ', learning_rate='constant', learning_rate_init=0.001, power_t
    =0.5, max_iter=200, shuffle=True, random_state=None, tol=0.0001,
    verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=
    True, early_stopping=False, validation_fraction=0.1, beta_1=0.9,
    beta_2=0.999, epsilon=1e-08, n_iter_no_change=10, max_fun=15000)
```

12.7 SVM

```
class sklearn.svm.SVC(C=1.0, kernel='rbf', degree=3, gamma='scale',
    coef0=0.0, shrinking=True, probability=False, tol=0.001,
    cache_size=200, class_weight=None, verbose=False, max_iter=-1,
    decision_function_shape='ovr', break_ties=False, random_state=None
    )
```

13 Appendix II

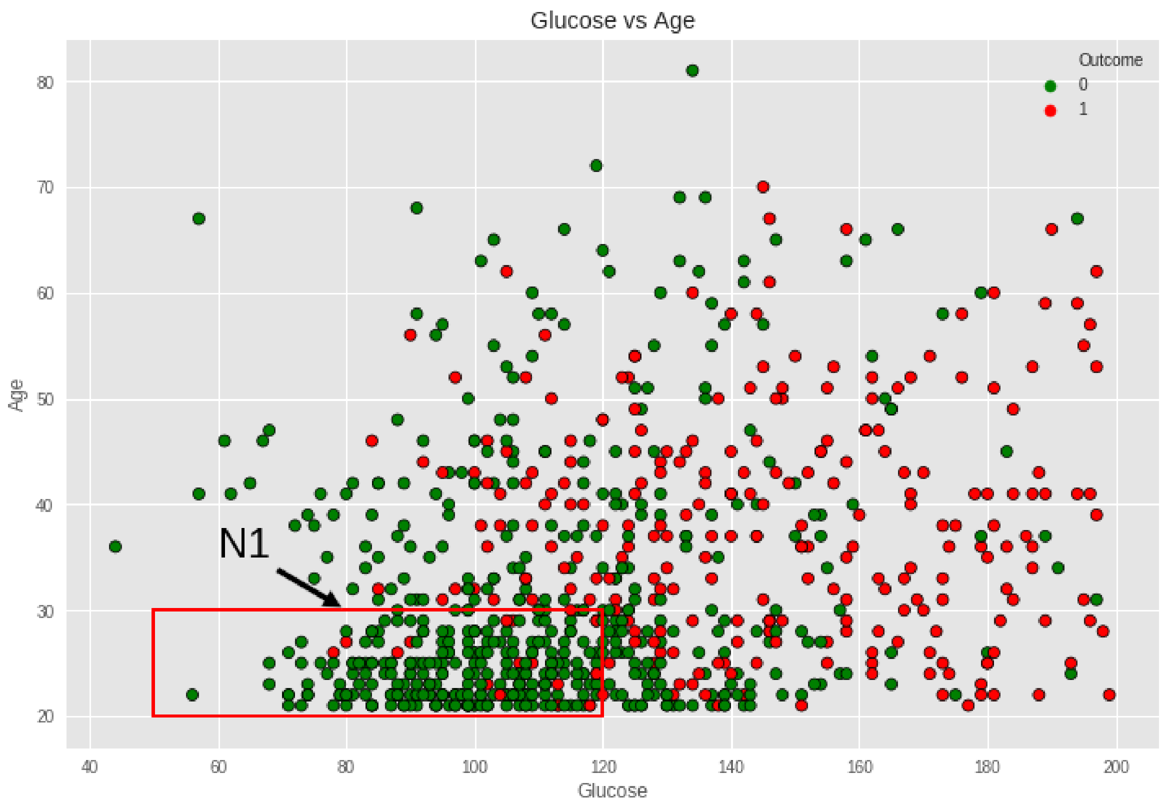


Figure 2: Master Node reading data from the SQL Database.

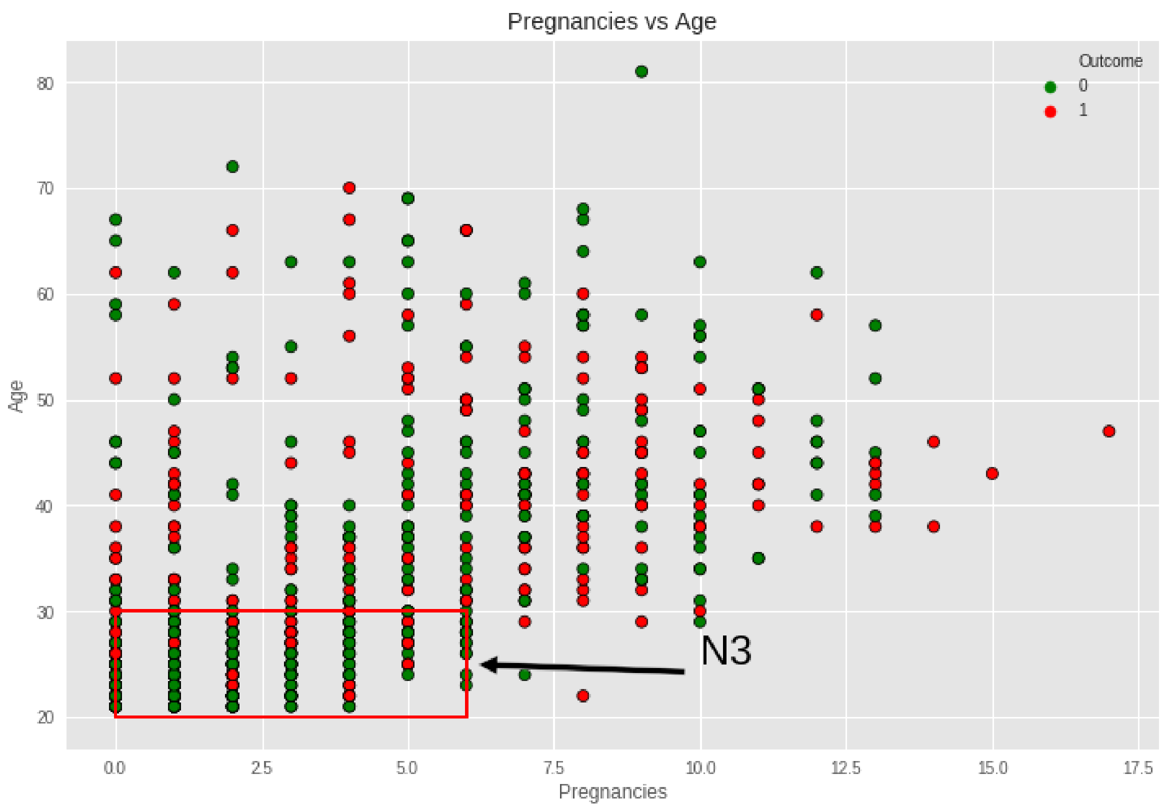


Figure 3: Master Node reading data from the SQL Database.

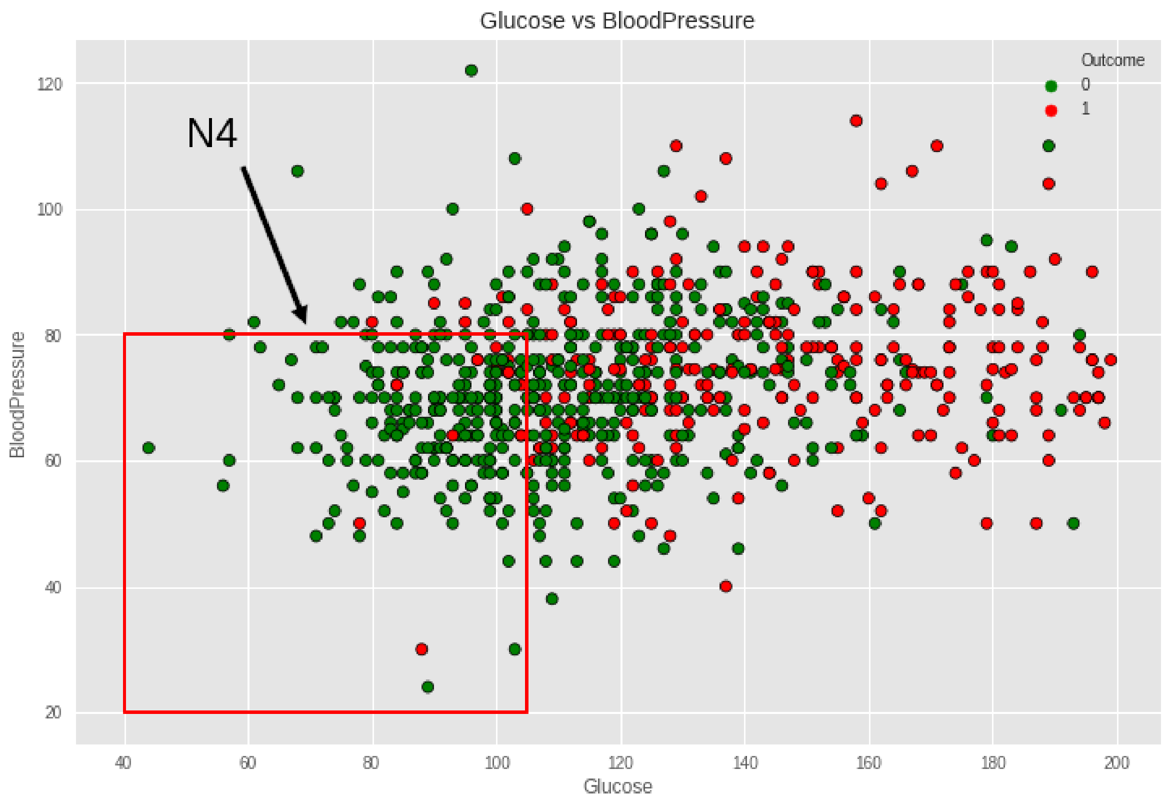


Figure 4: Master Node reading data from the SQL Database.

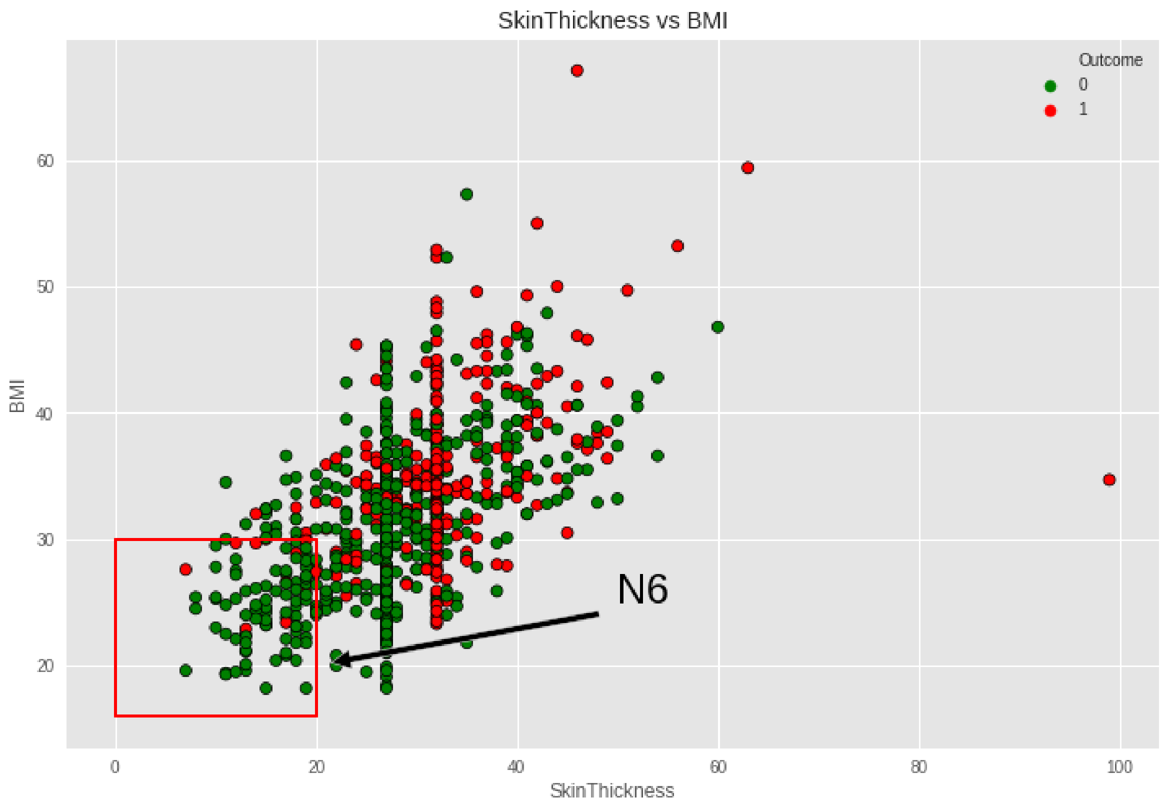


Figure 5: Master Node reading data from the SQL Database.

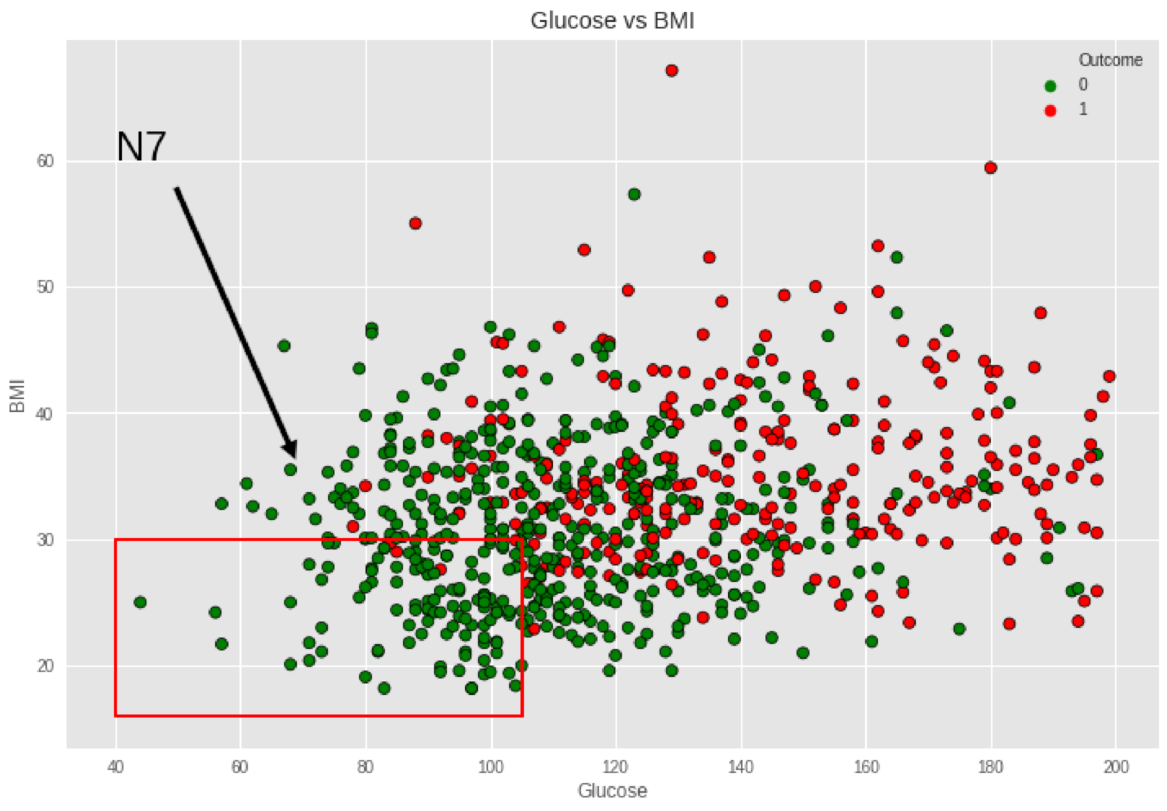


Figure 6: Master Node reading data from the SQL Database.

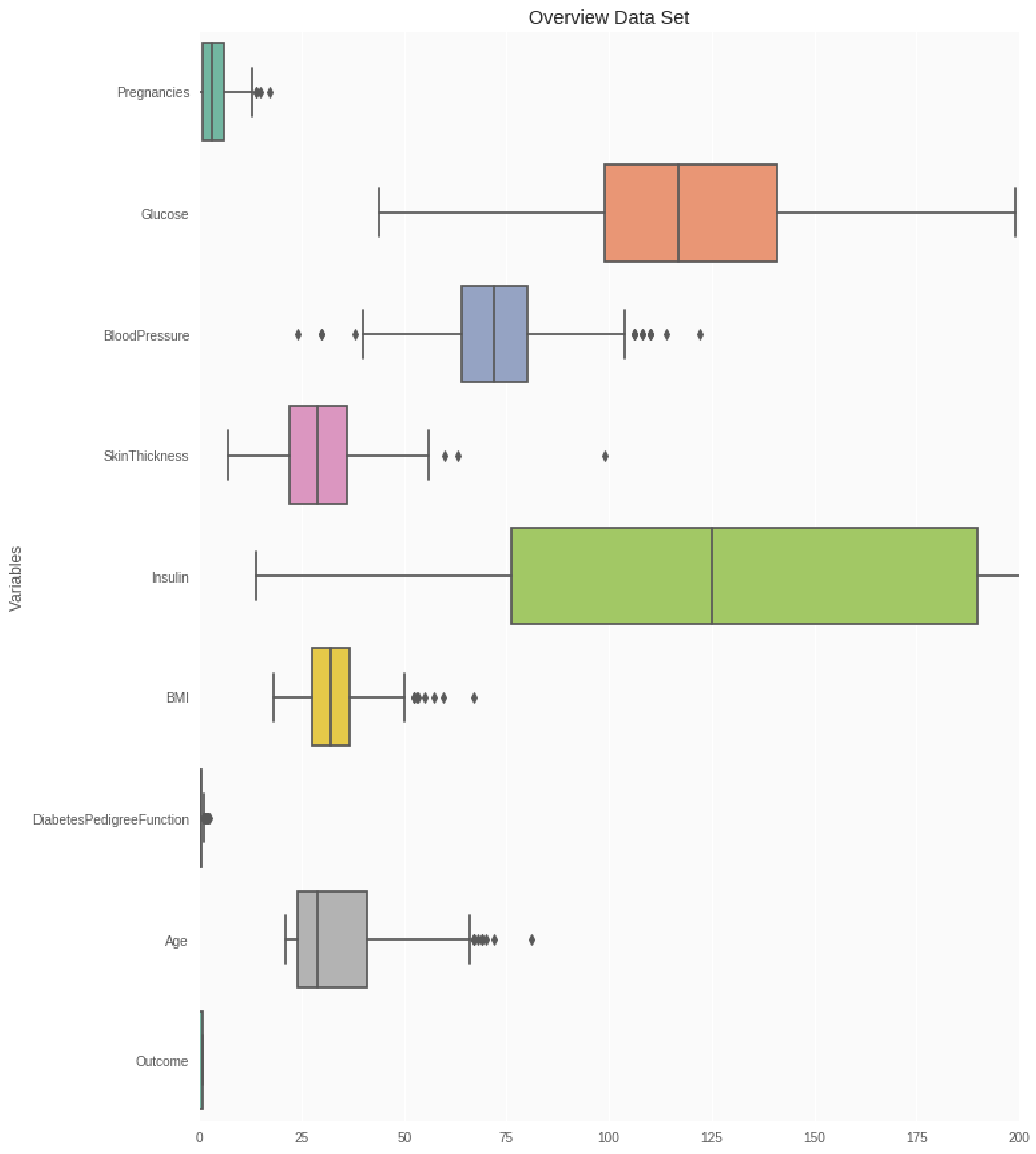


Figure 7: Master Node reading data from the SQL Database.

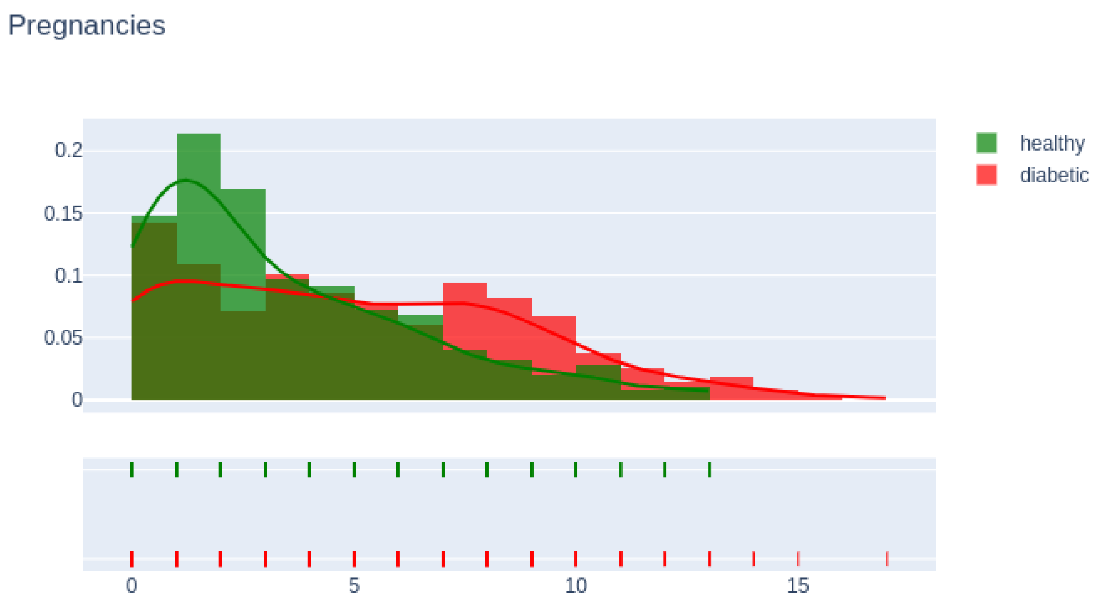


Figure 8: Master Node reading data from the SQL Database.

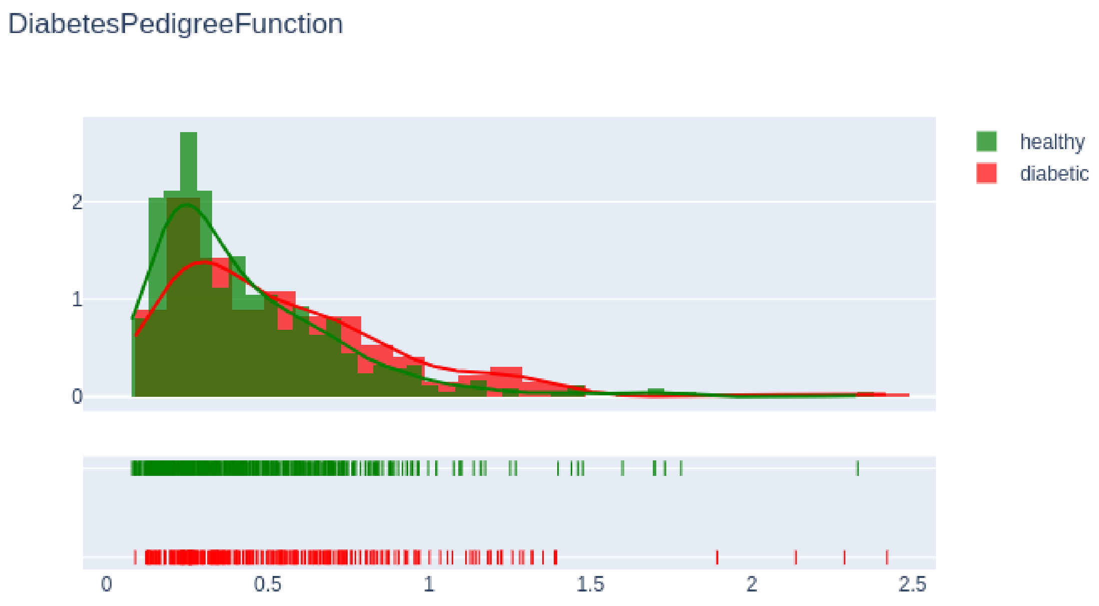


Figure 9: Master Node reading data from the SQL Database.

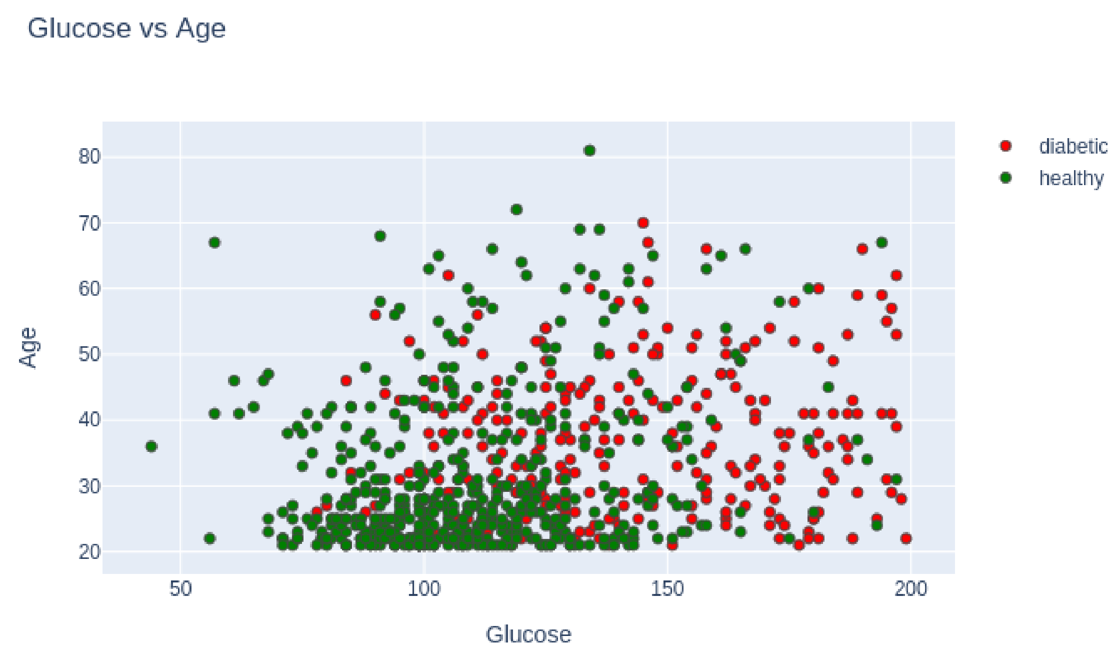


Figure 10: Master Node reading data from the SQL Database.

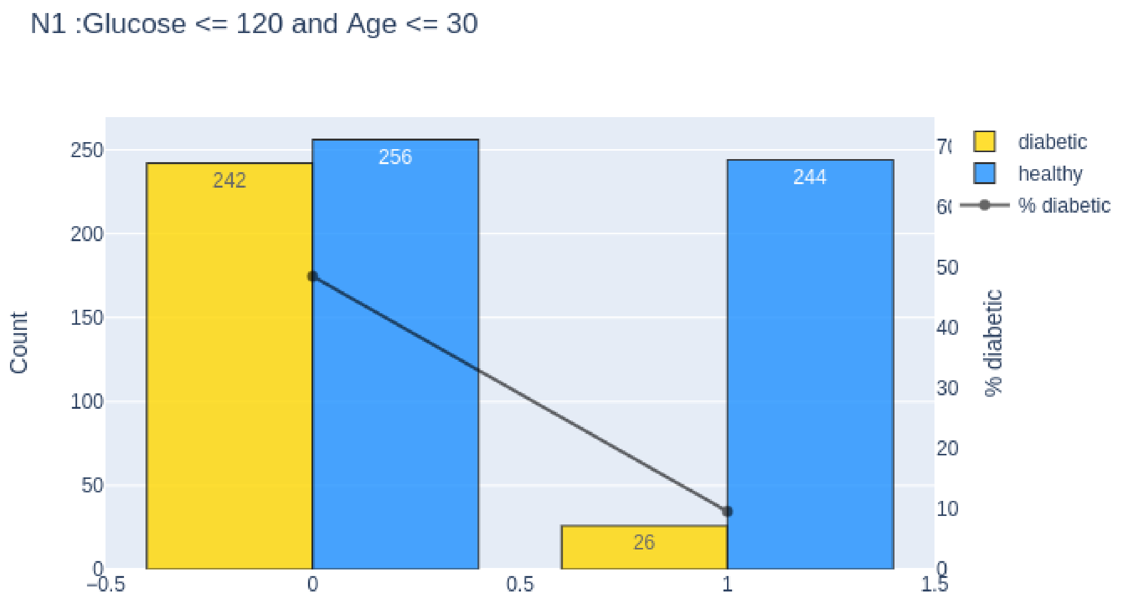


Figure 11: Master Node reading data from the SQL Database.

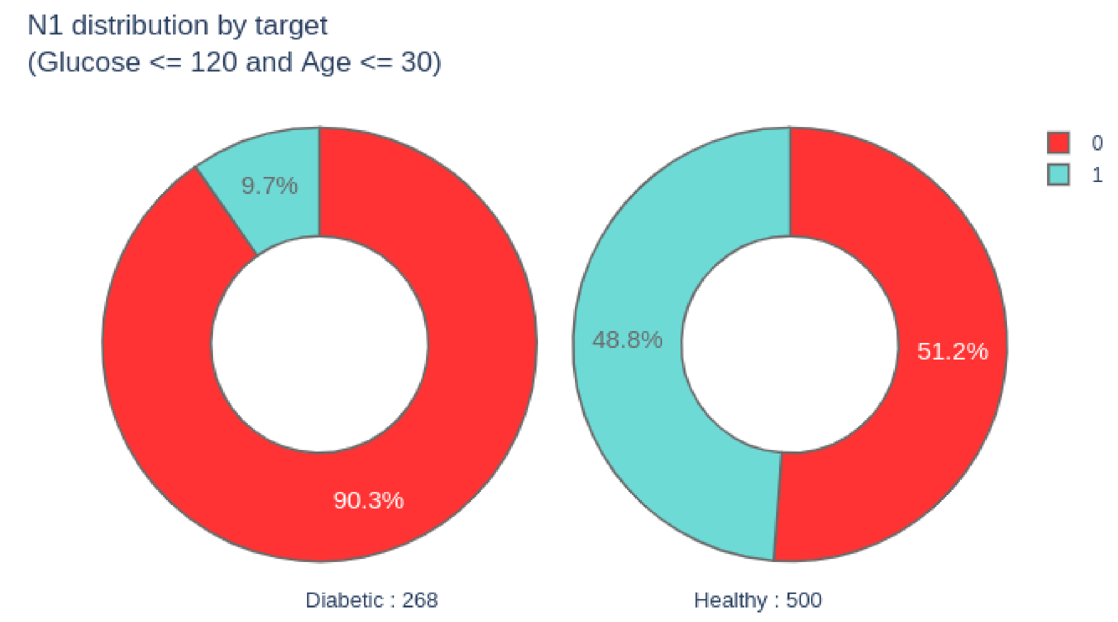


Figure 12: Master Node reading data from the SQL Database.

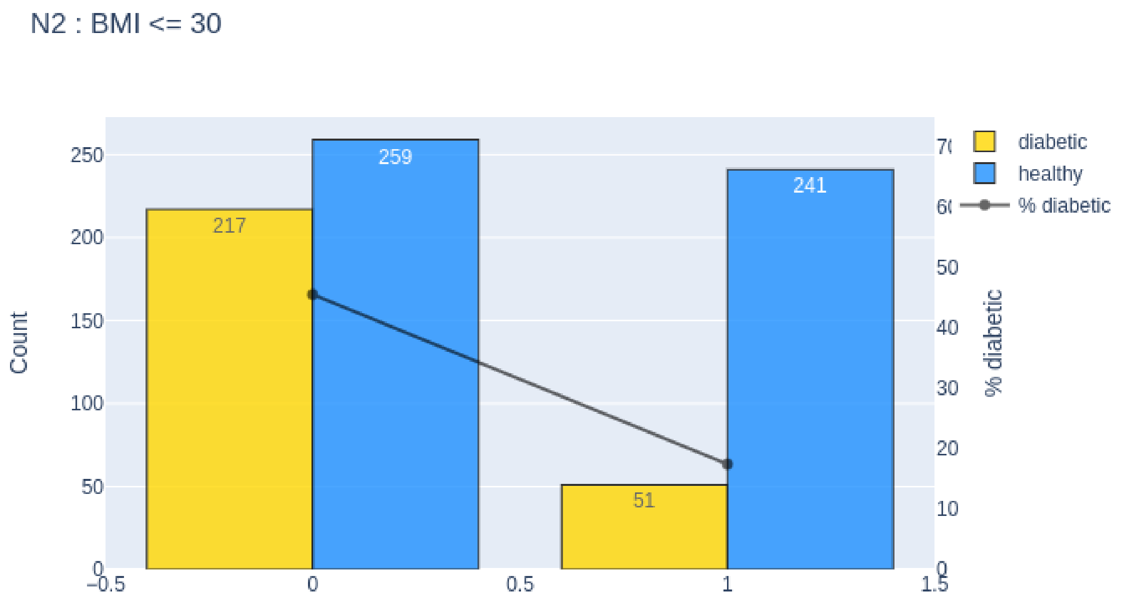


Figure 13: Master Node reading data from the SQL Database.

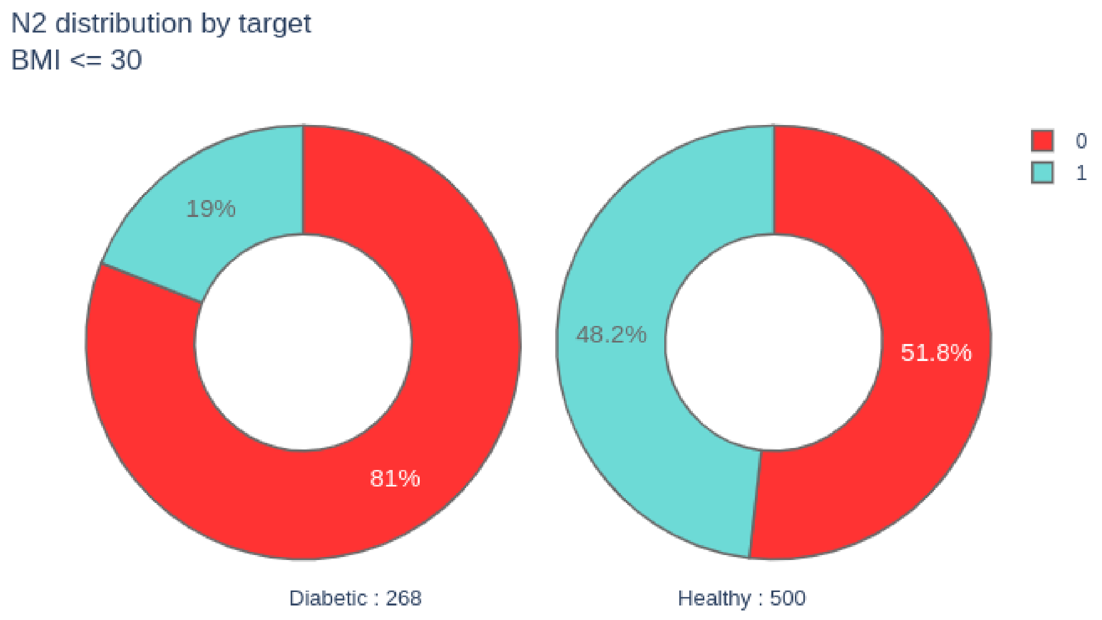


Figure 14: Master Node reading data from the SQL Database.

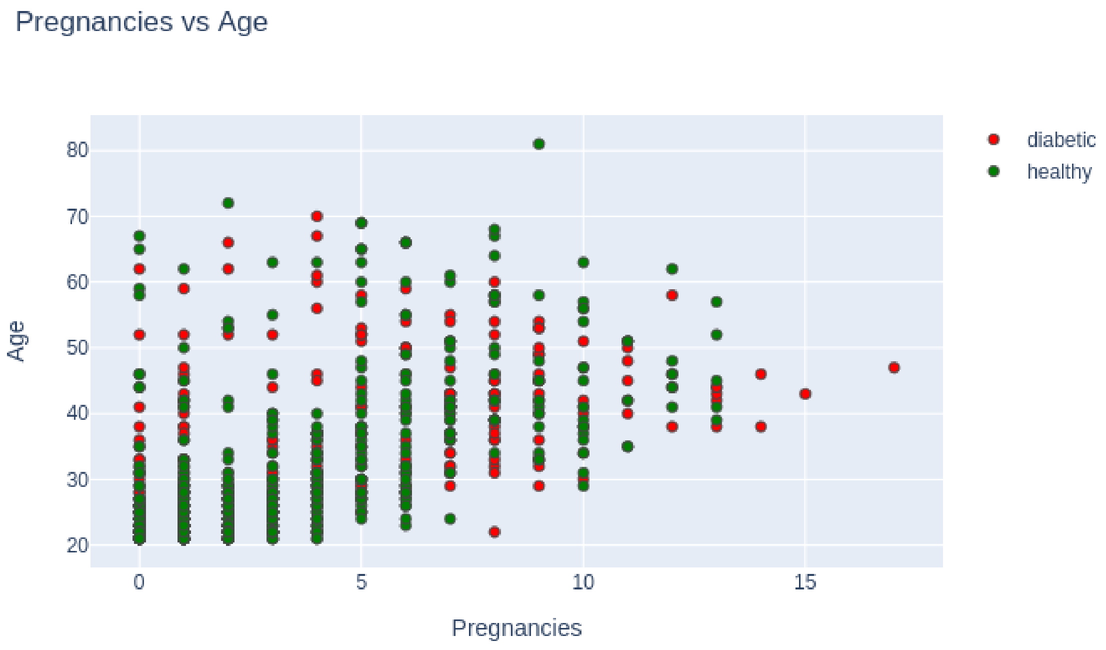


Figure 15: Master Node reading data from the SQL Database.

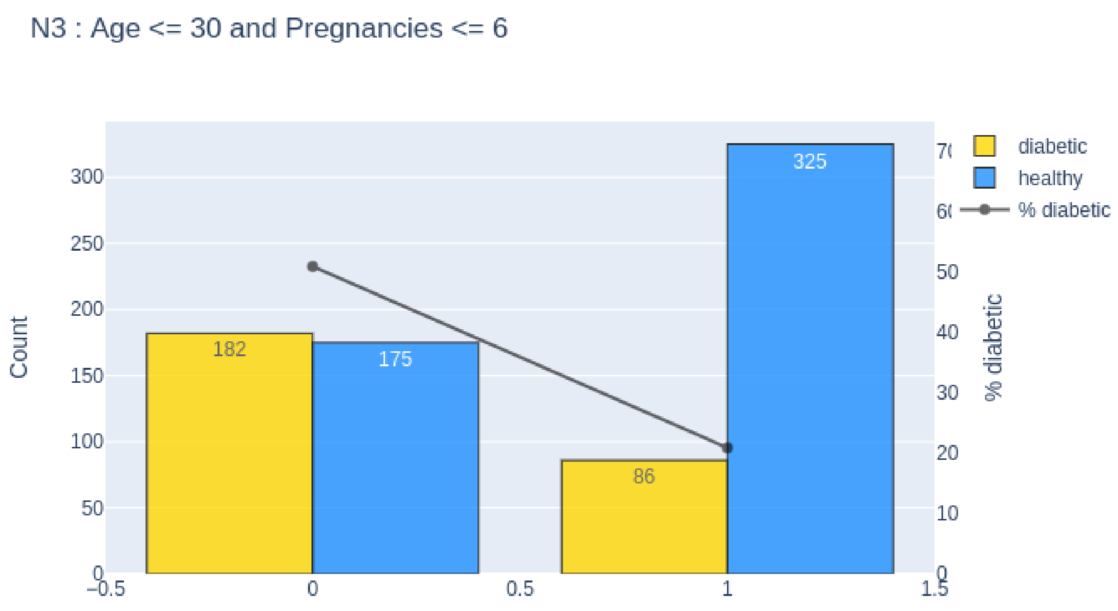


Figure 16: Master Node reading data from the SQL Database.

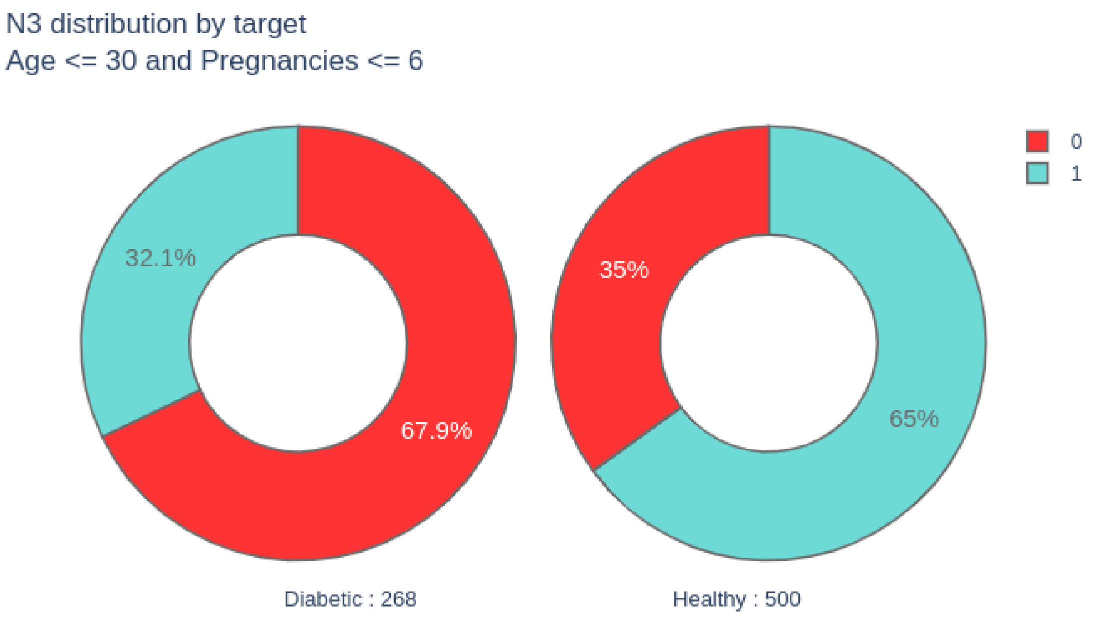


Figure 17: Master Node reading data from the SQL Database.

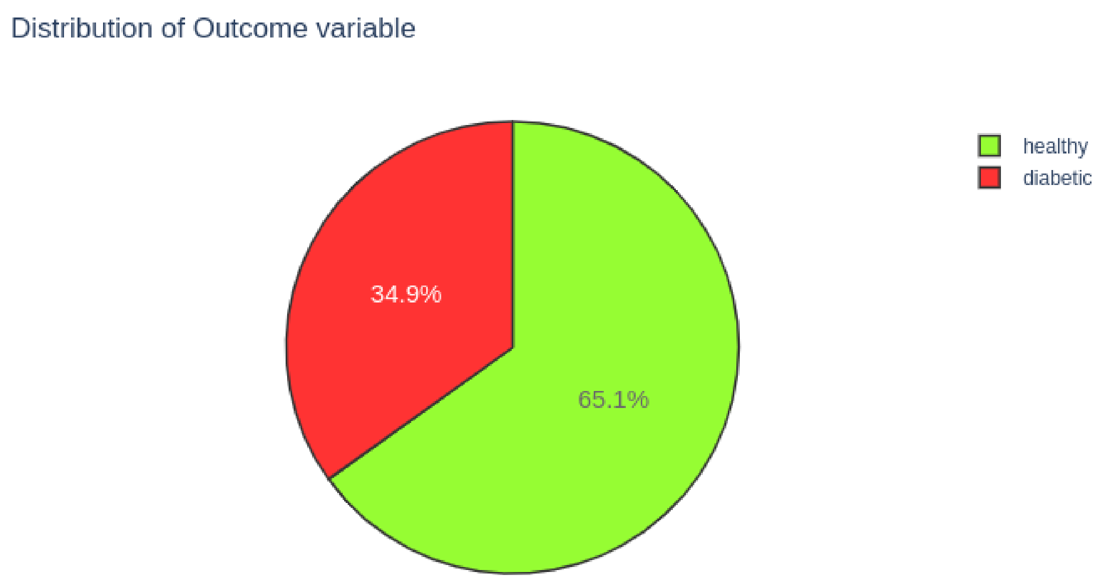


Figure 18: Master Node reading data from the SQL Database.

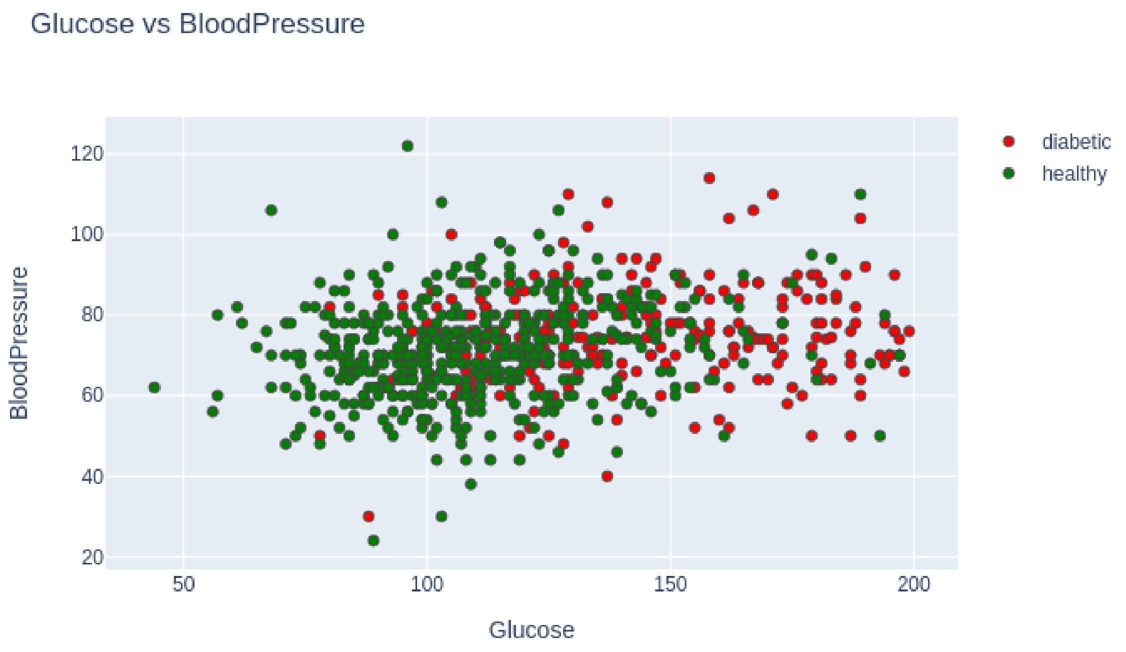


Figure 19: Master Node reading data from the SQL Database.

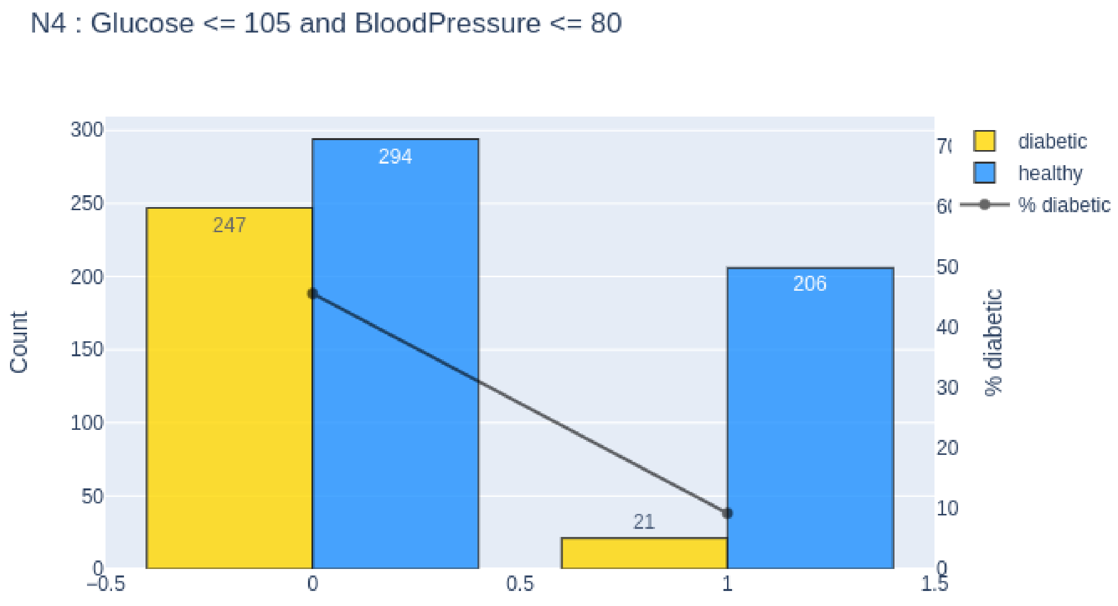


Figure 20: Master Node reading data from the SQL Database.

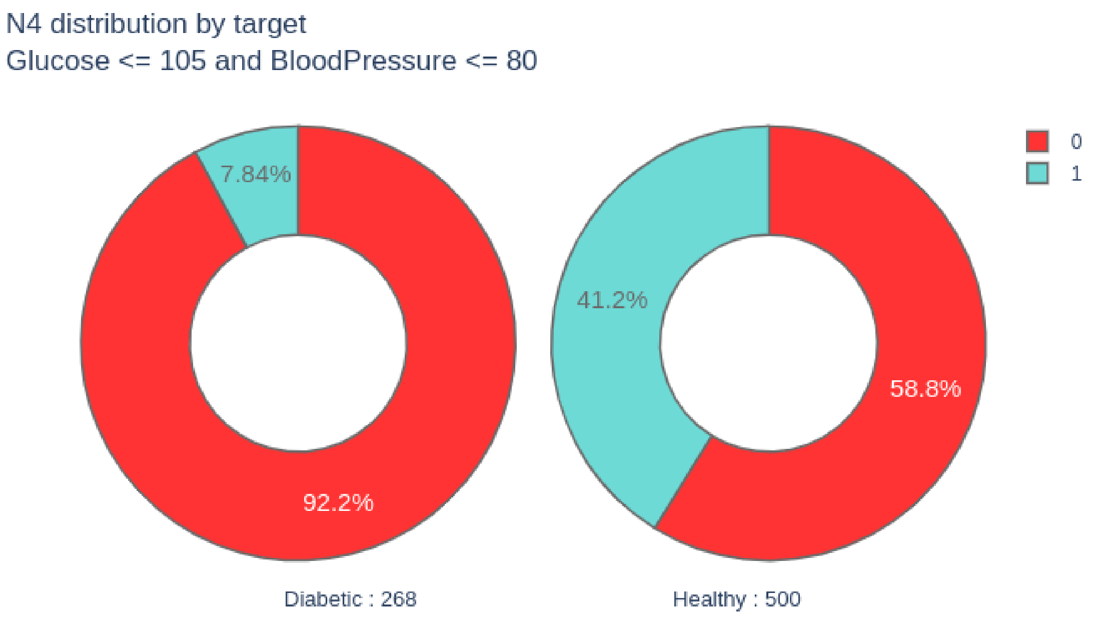


Figure 21: Master Node reading data from the SQL Database.

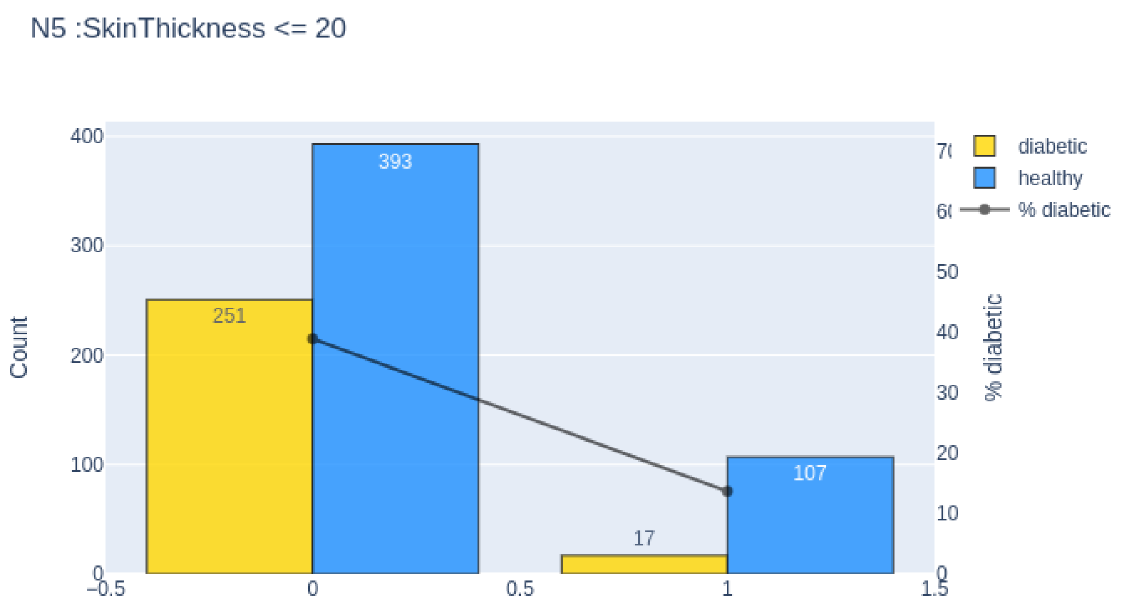


Figure 22: Master Node reading data from the SQL Database.

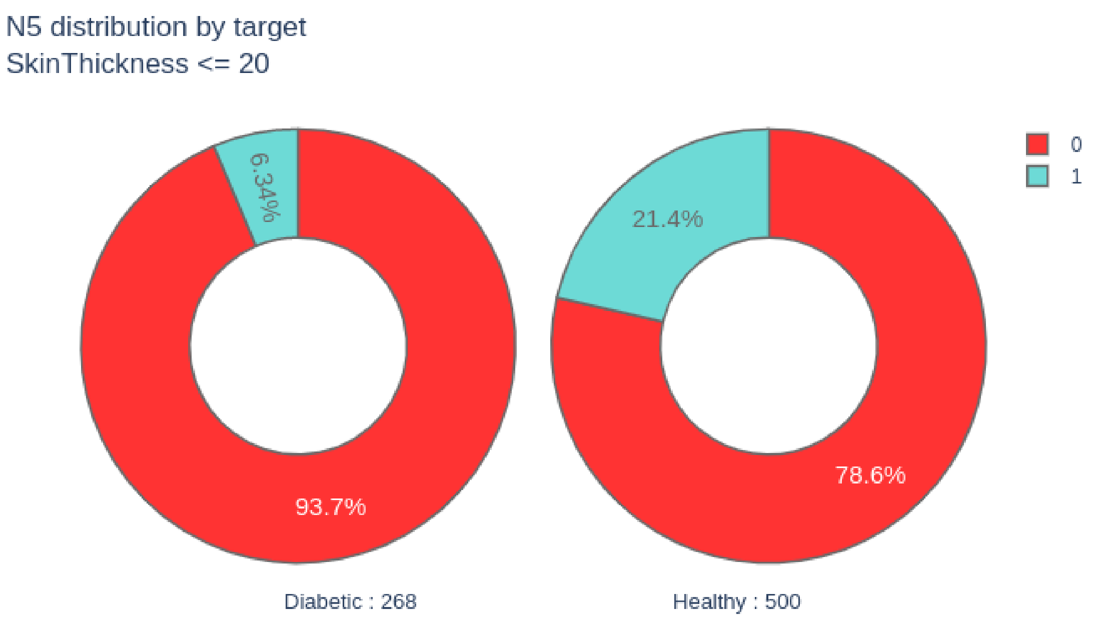


Figure 23: Master Node reading data from the SQL Database.

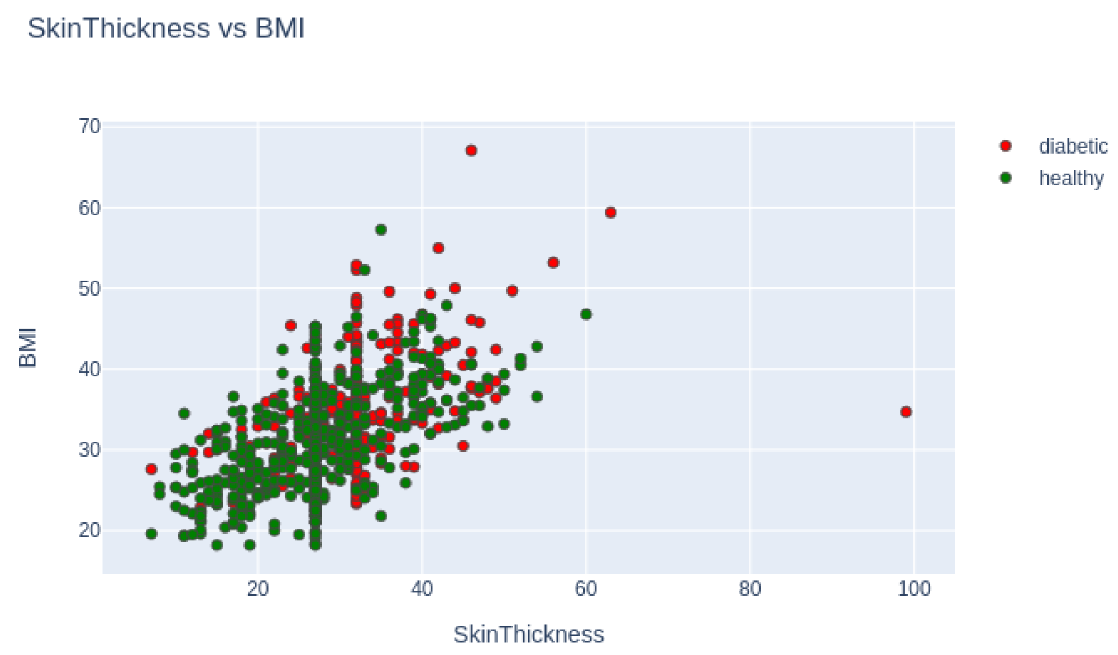


Figure 24: Master Node reading data from the SQL Database.

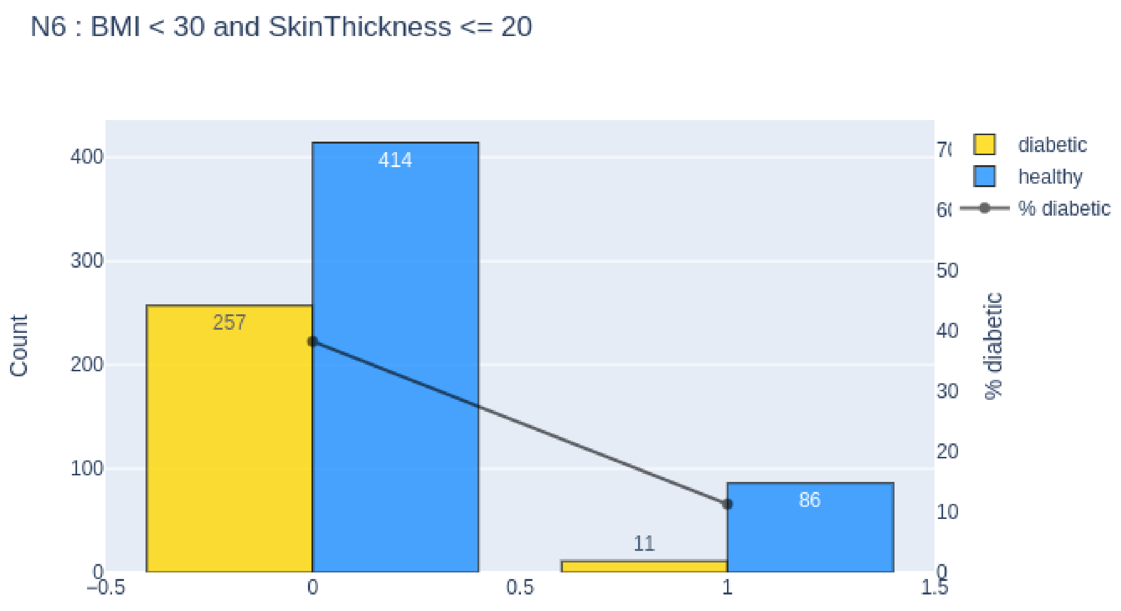


Figure 25: N6 Distribution by target.

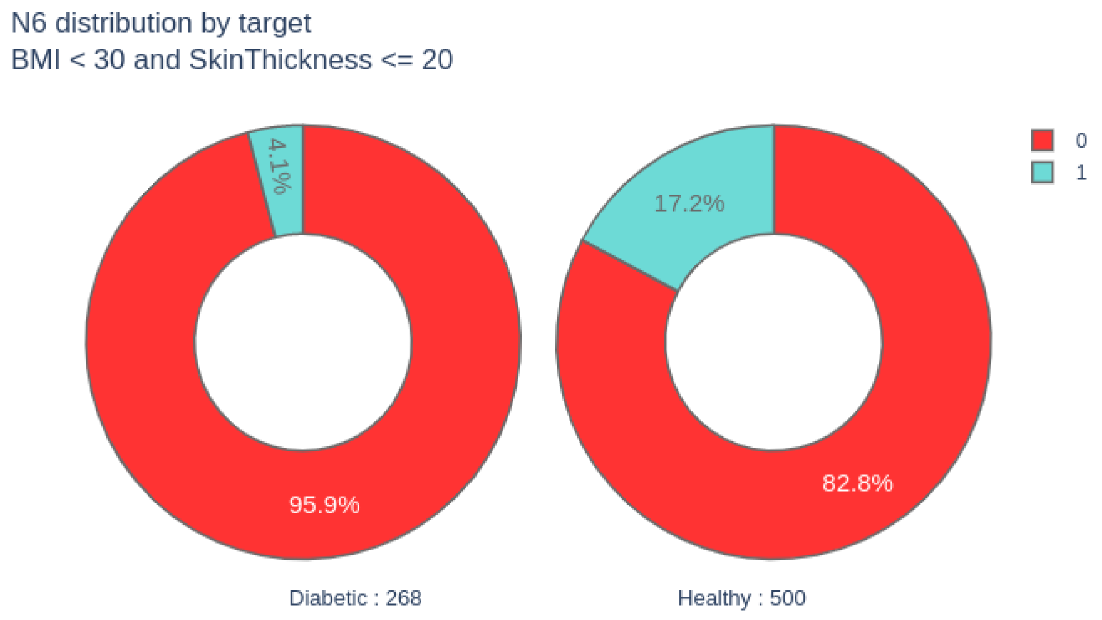


Figure 26: Glucose v/s BMI

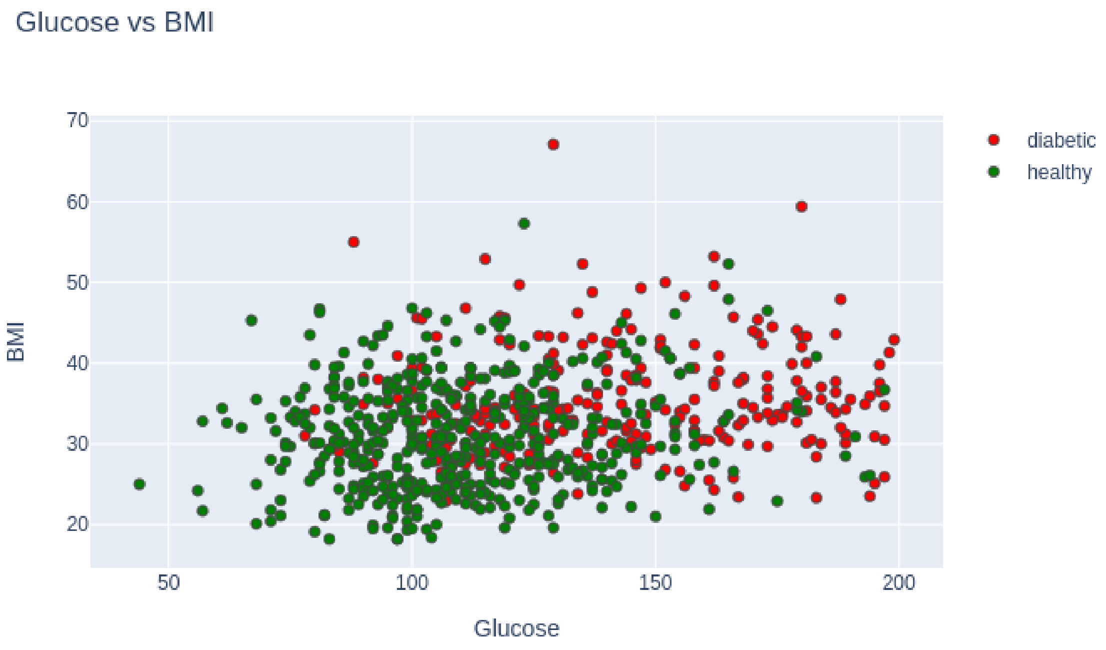


Figure 27: N7

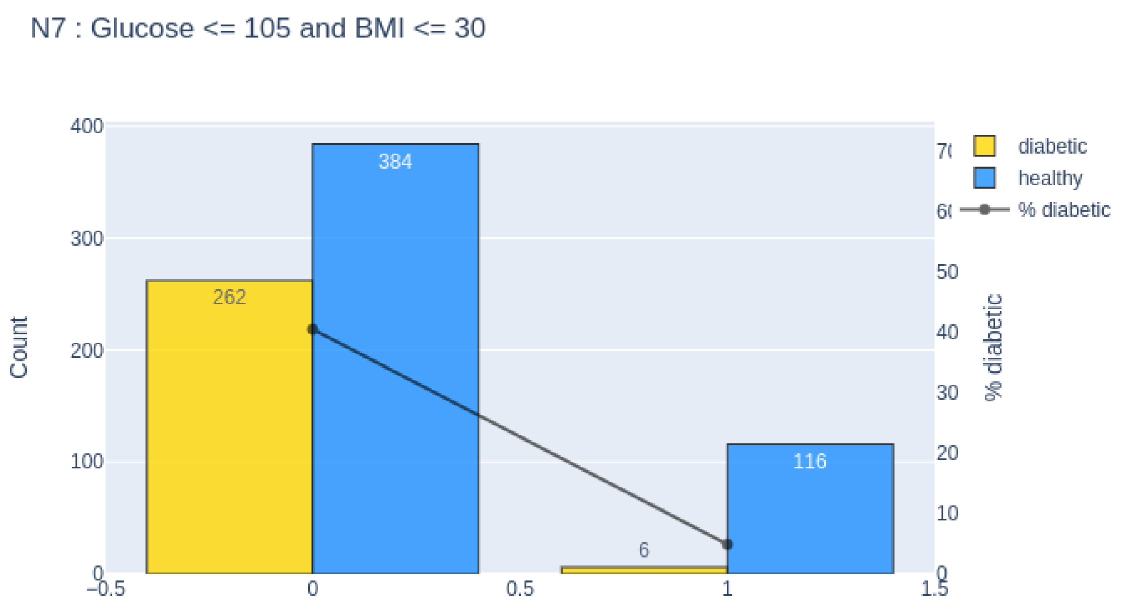


Figure 28: Missing values in count and percentage.

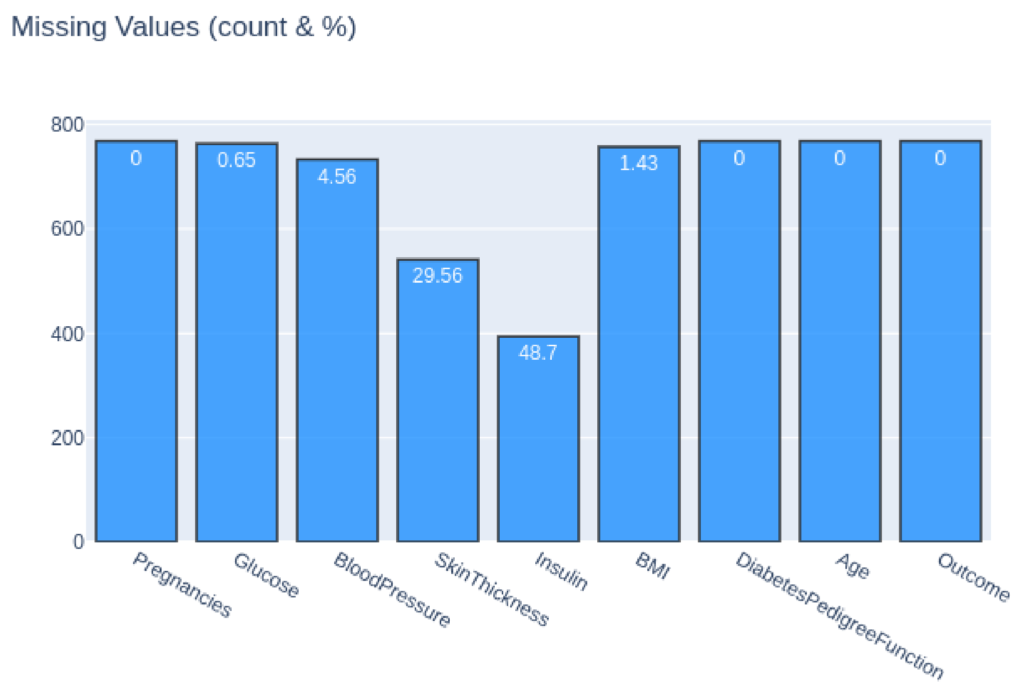


Figure 29: N7 distribution by target.

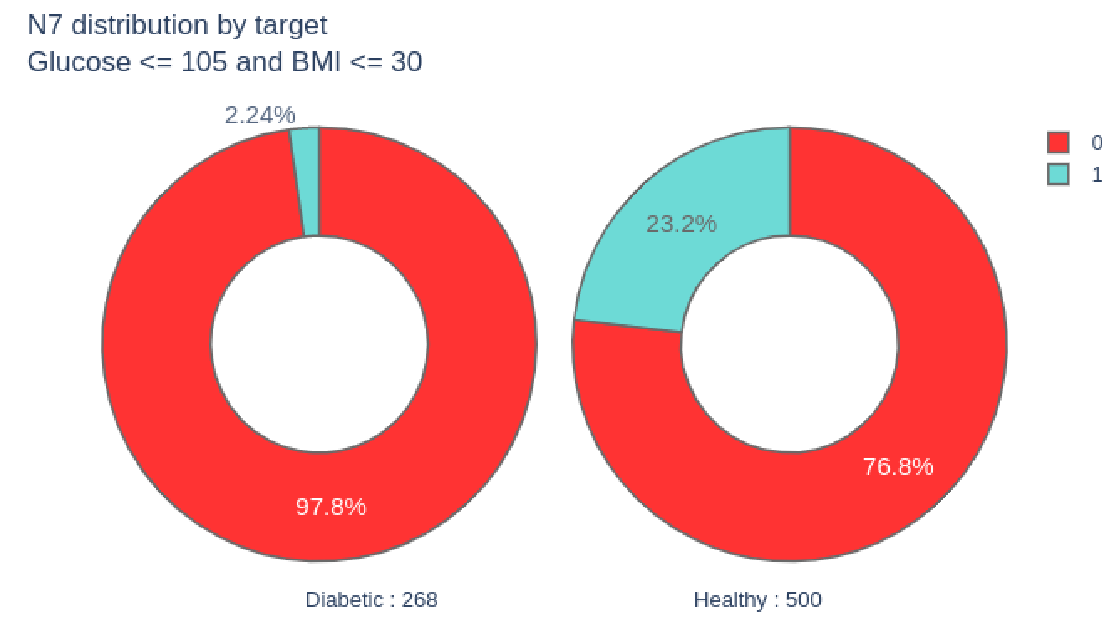


Figure 30: N7.

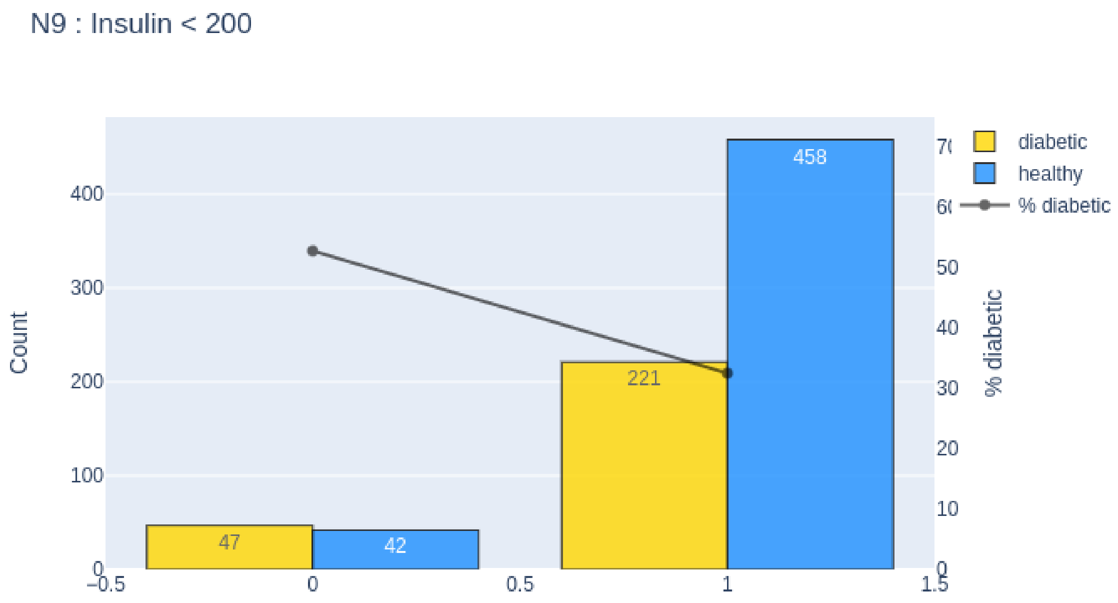


Figure 31: N9 where $insulin < 200$.

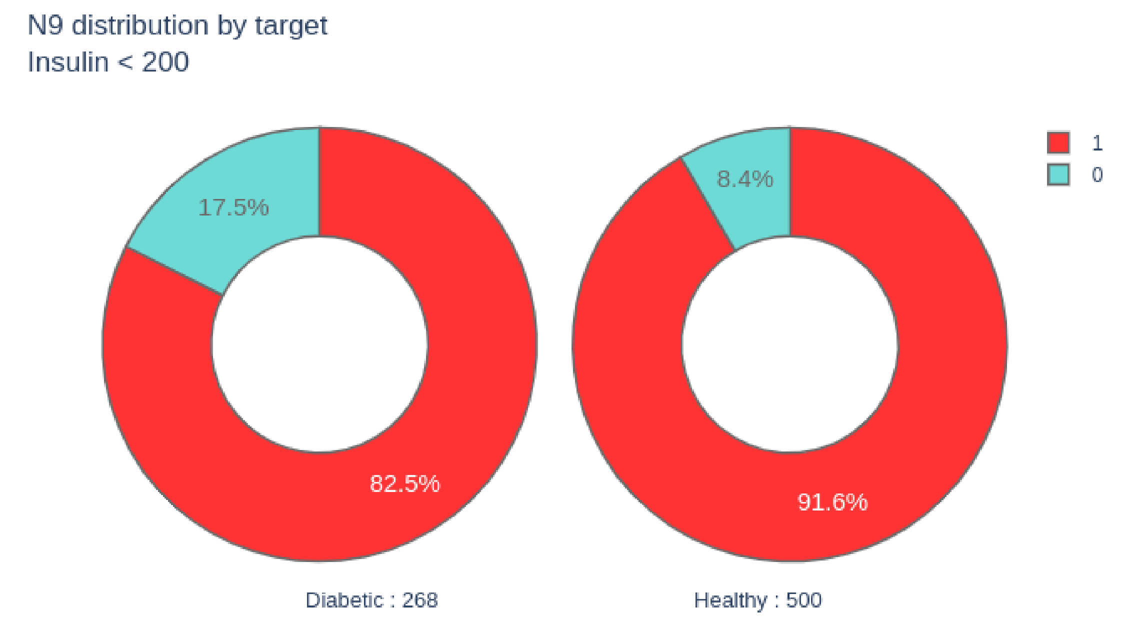


Figure 32: N9 distribution by target.

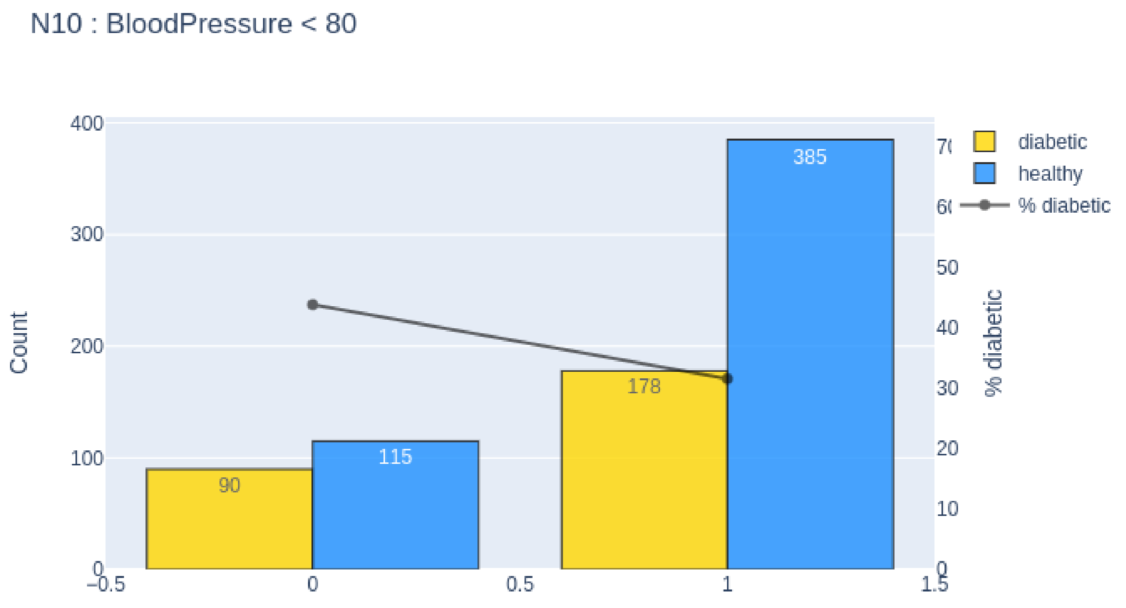


Figure 33: N10 distribution for diabetic and healthy people bp¡80.

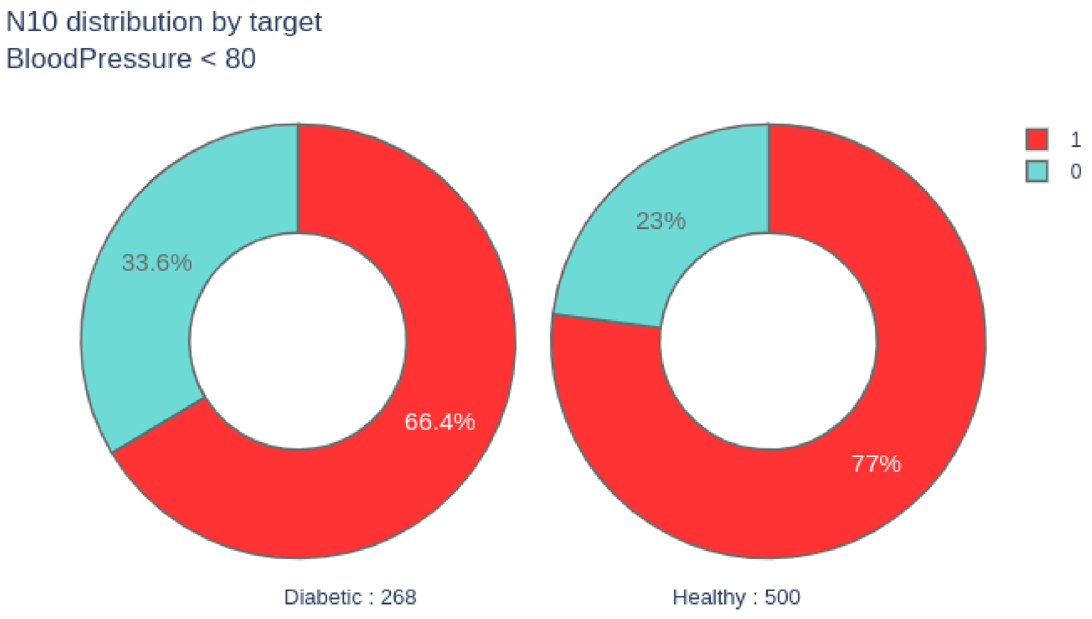


Figure 34: N10 distribution for diabetic and healthy people.

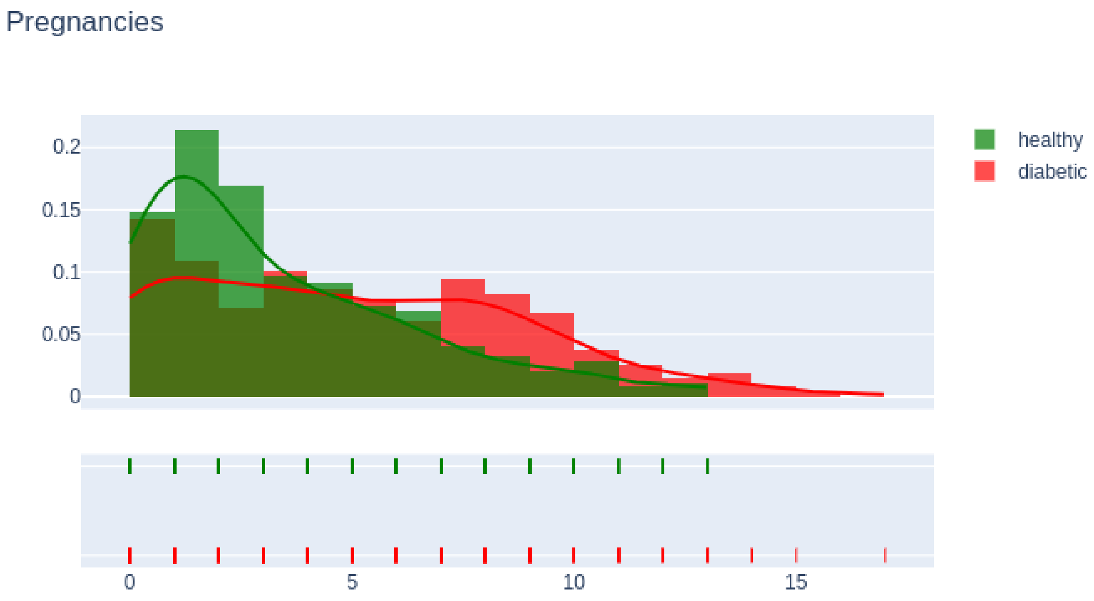


Figure 35: Pregnancies distribution for diabetic and healthy people.

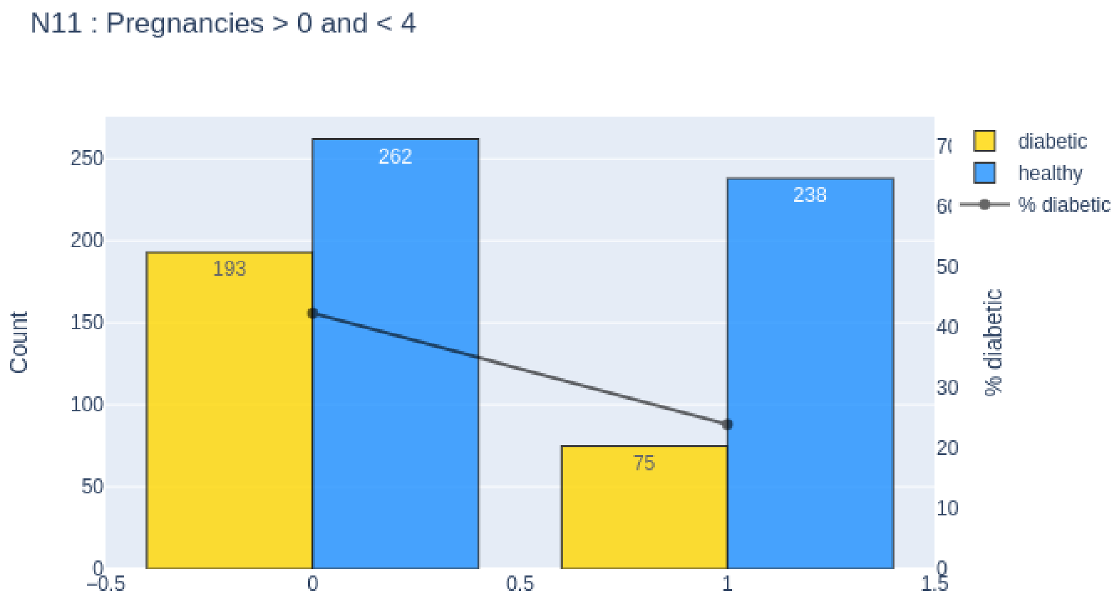


Figure 36: N11 feature obtained with patients ¿ 0 and ¡ 4.

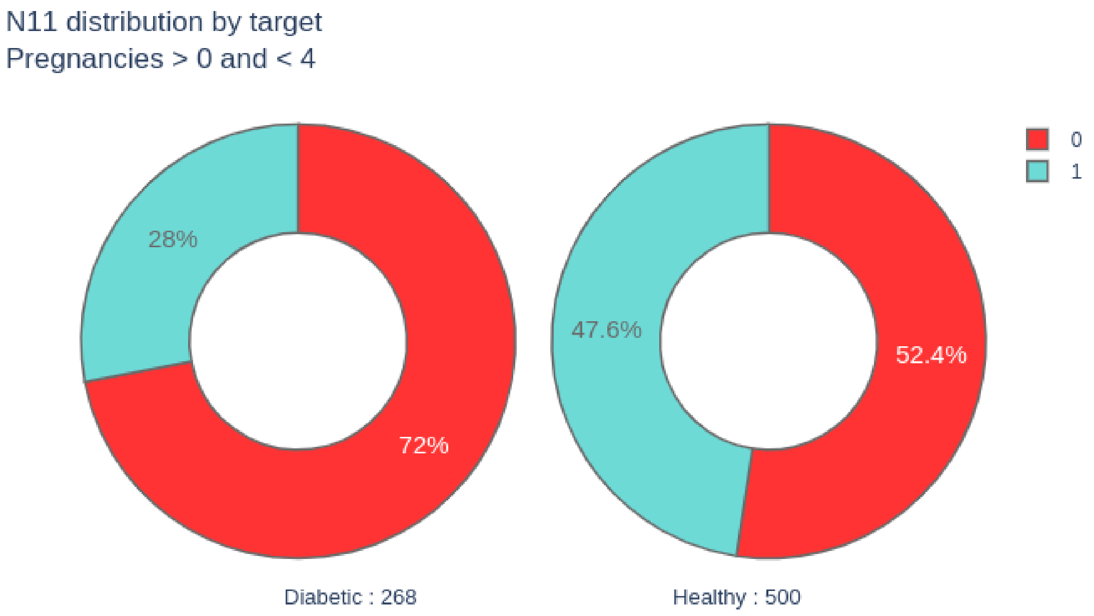


Figure 37: N11 distribution via target labels.

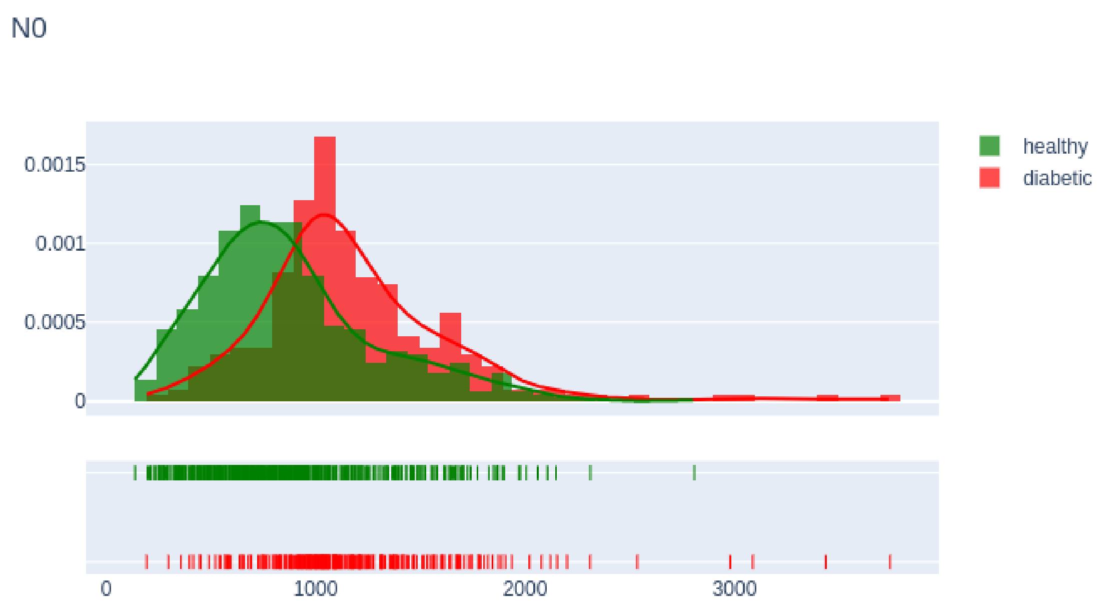


Figure 38: N0 Distribution.

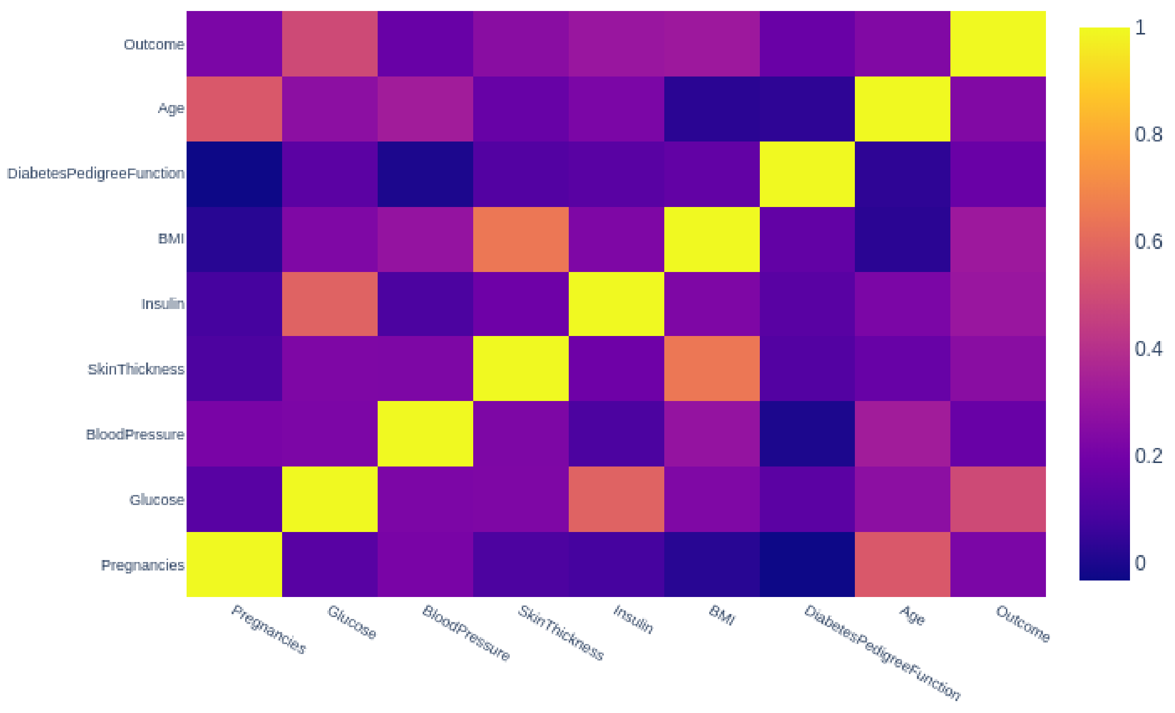


Figure 39: Correlation plot of raw dataset.

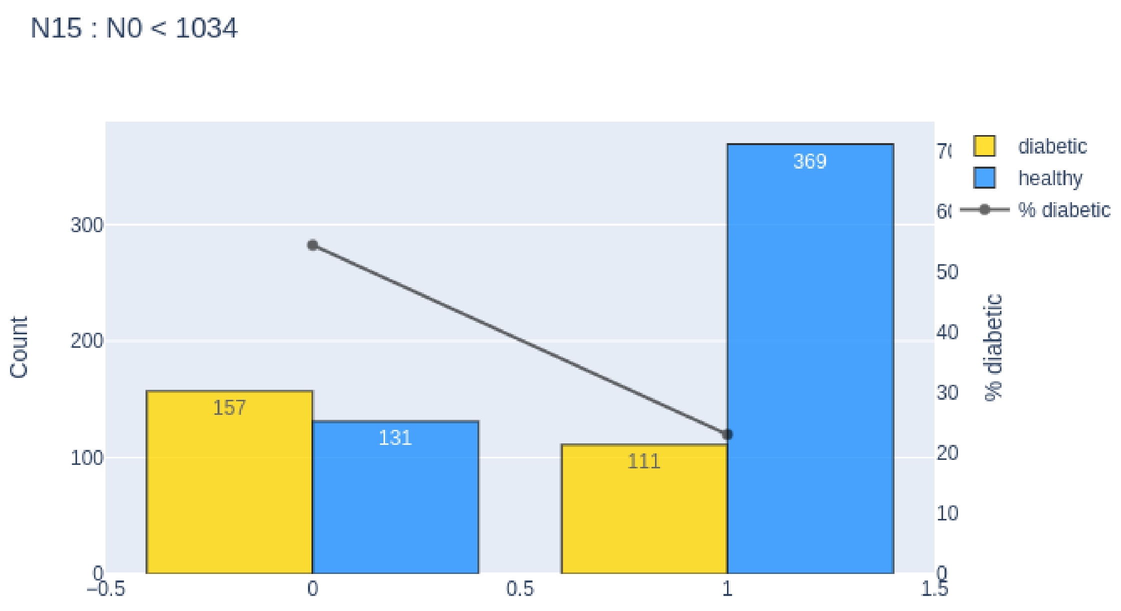


Figure 40: N15 where N0 is less than 1034.

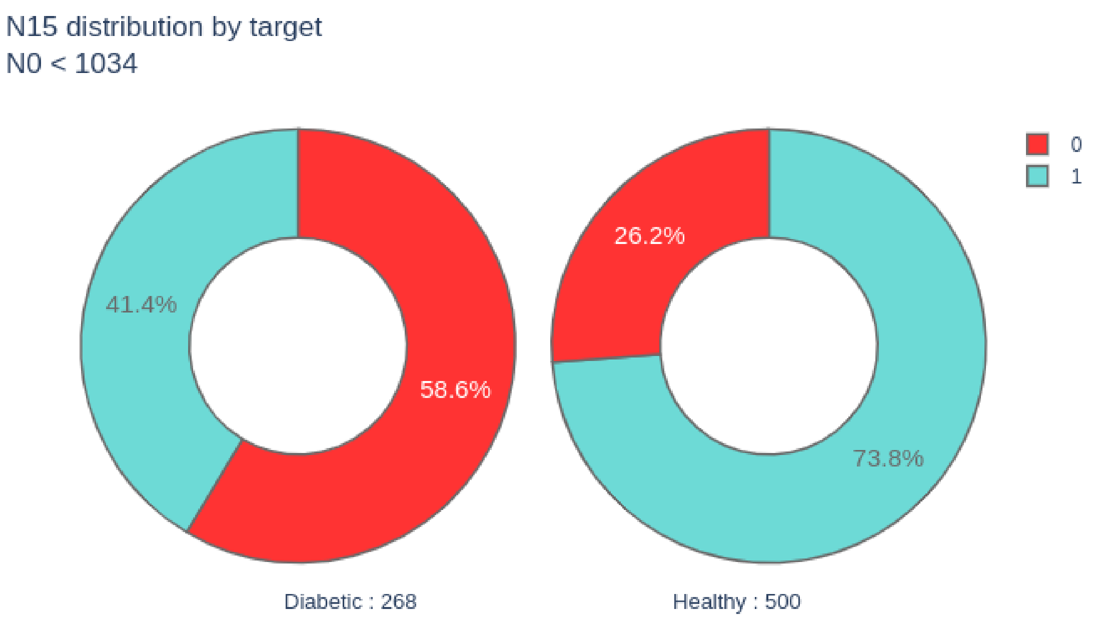


Figure 41: N15 distribution by target labels.

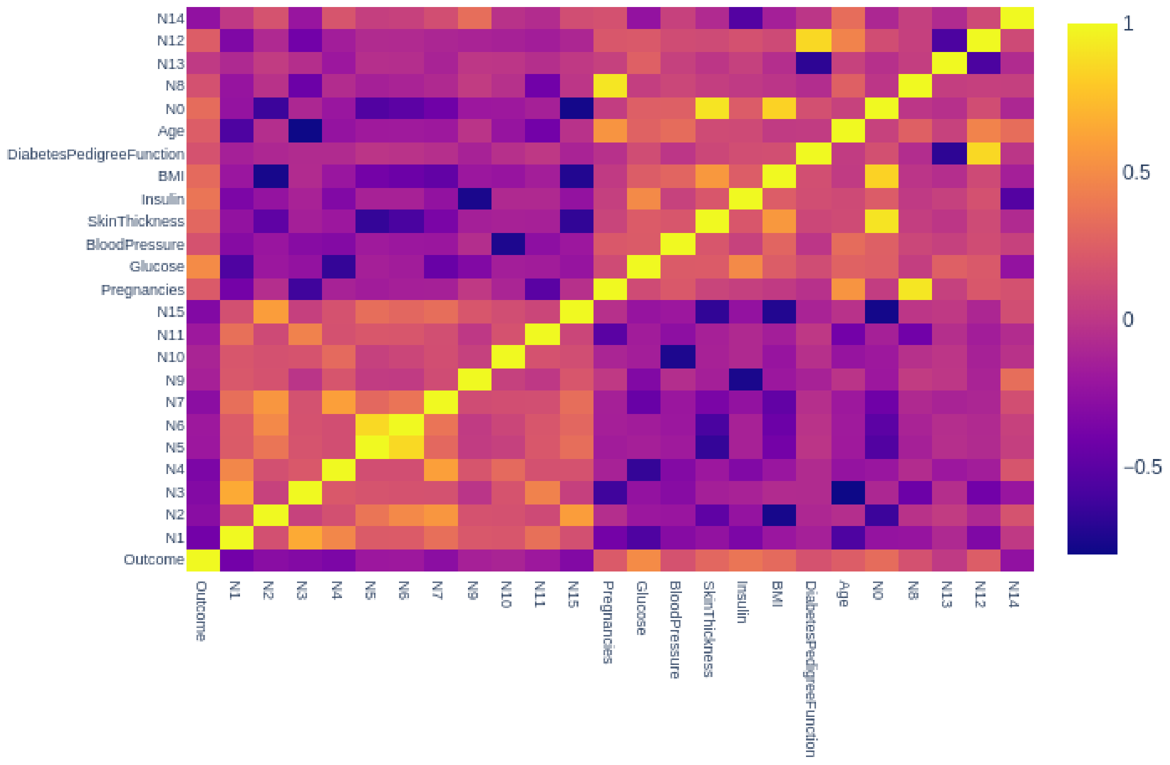


Figure 42: Correlation plot with newly added features.

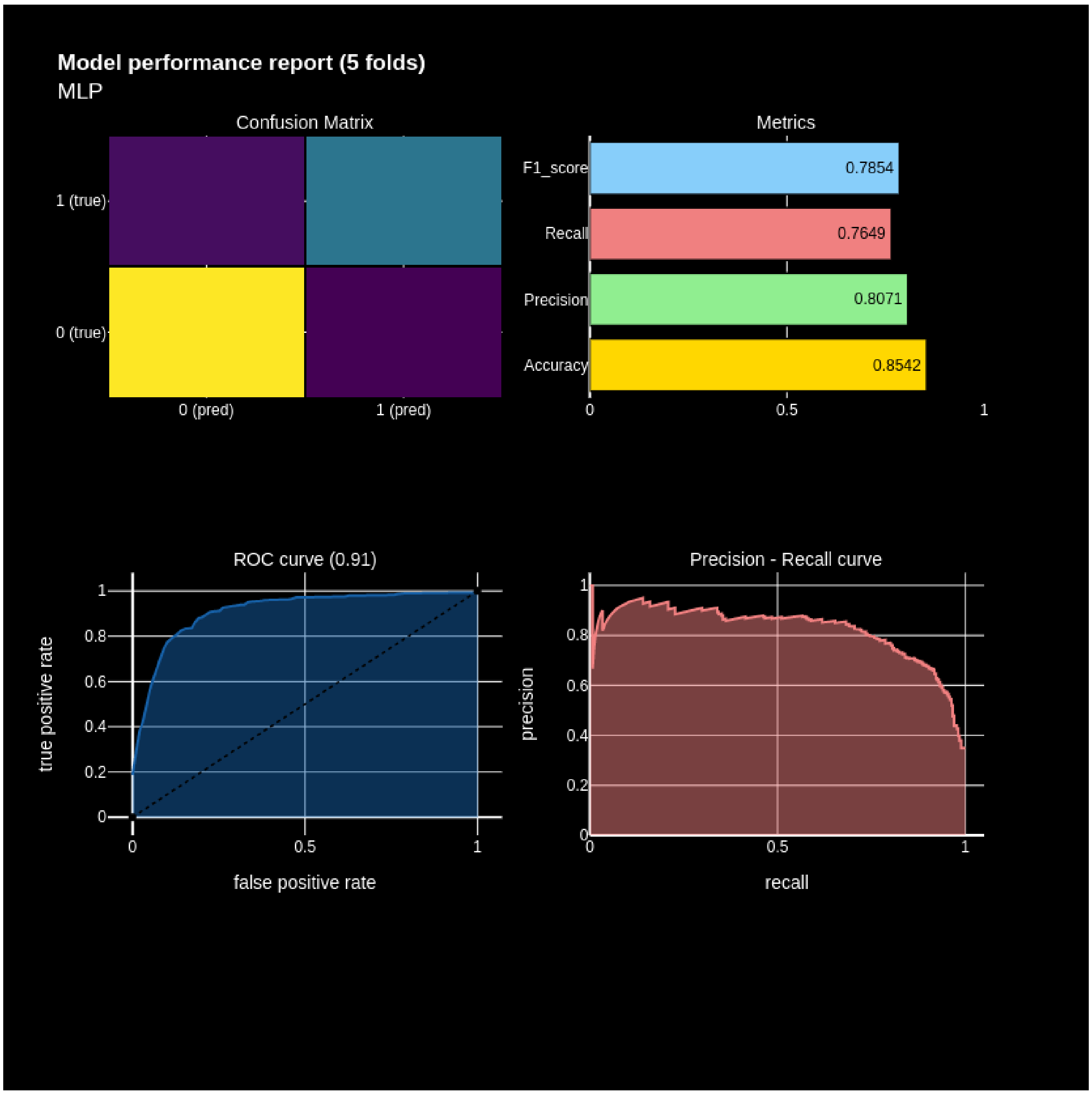
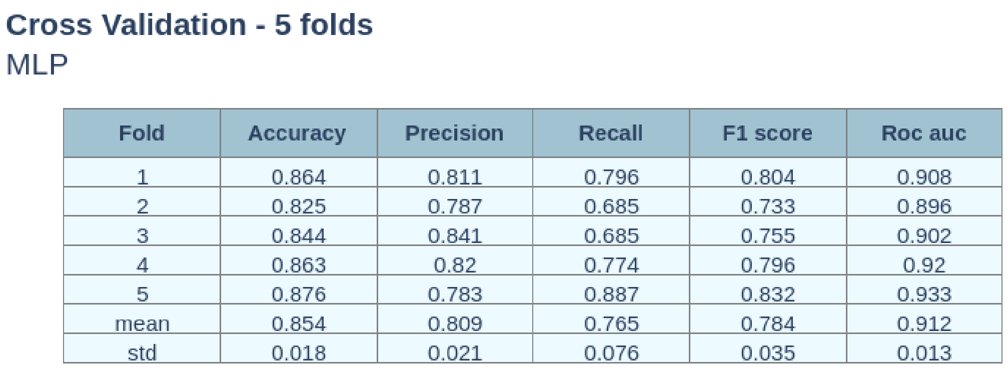


Figure 43: Model Performance report measured with 5 fold iterations.



Cross Validation - 5 folds

MLP

Fold	Accuracy	Precision	Recall	F1 score	Roc auc
1	0.864	0.811	0.796	0.804	0.908
2	0.825	0.787	0.685	0.733	0.896
3	0.844	0.841	0.685	0.755	0.902
4	0.863	0.82	0.774	0.796	0.92
5	0.876	0.783	0.887	0.832	0.933
mean	0.854	0.809	0.765	0.784	0.912
std	0.018	0.021	0.076	0.035	0.013

Figure 44: 5 Fold Cross Validation score for the MLP Classifier.

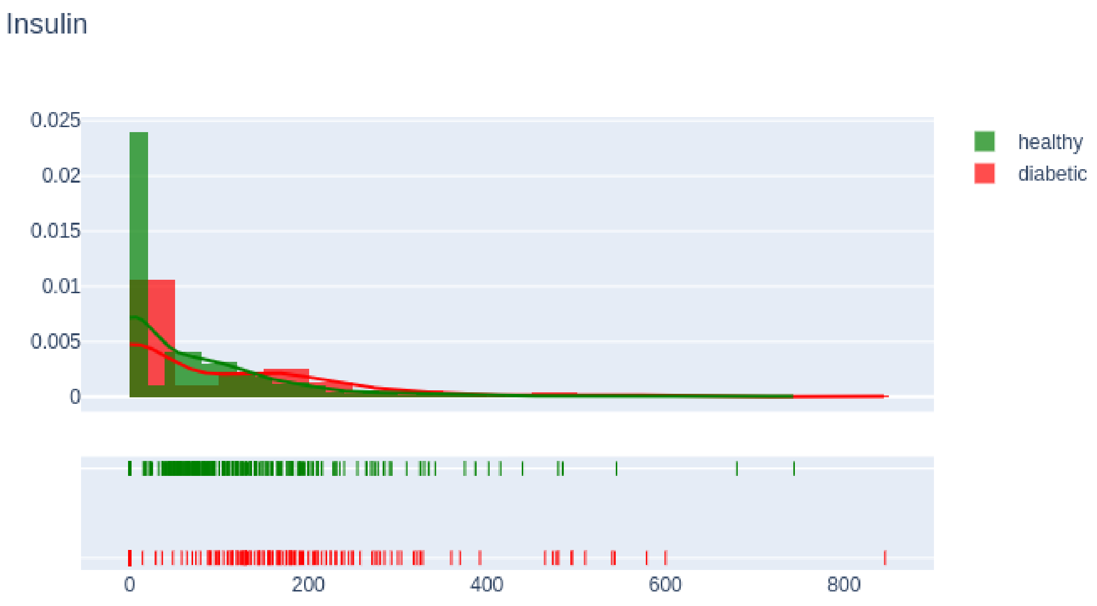


Figure 45: Insulin distribution for diabetic and healthy people.

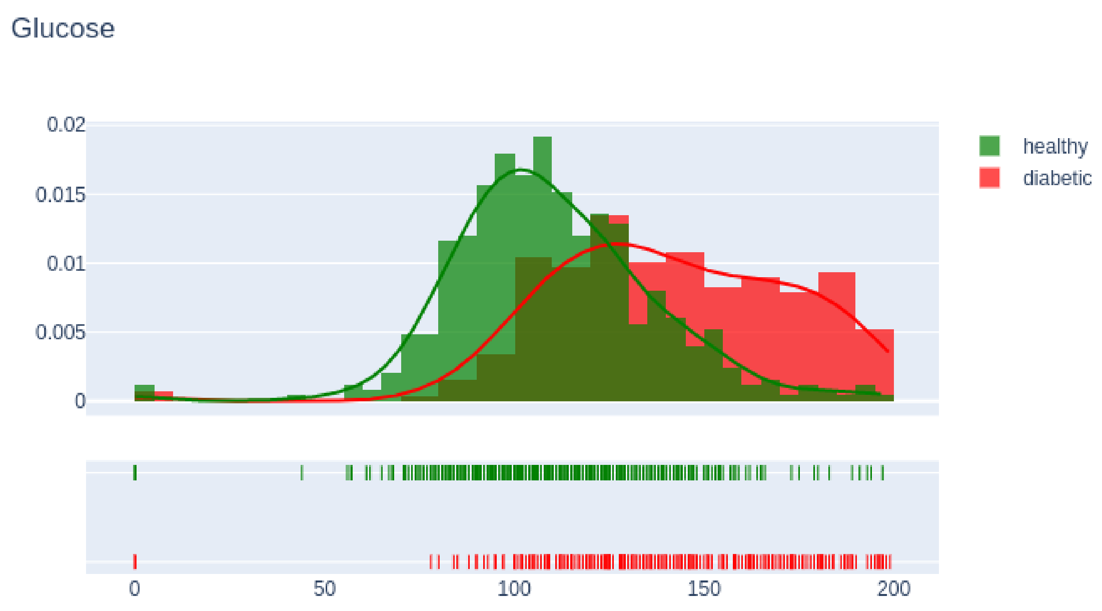


Figure 46: Glucose distribution for diabetic and healthy people.

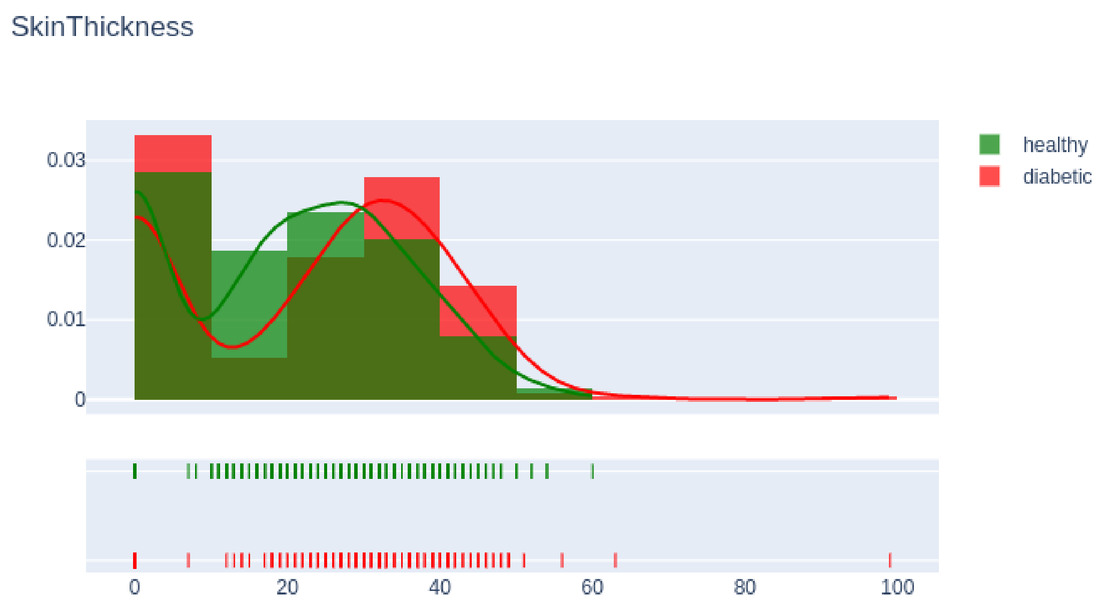


Figure 47: Skin Thickness distribution for diabetic and healthy people.

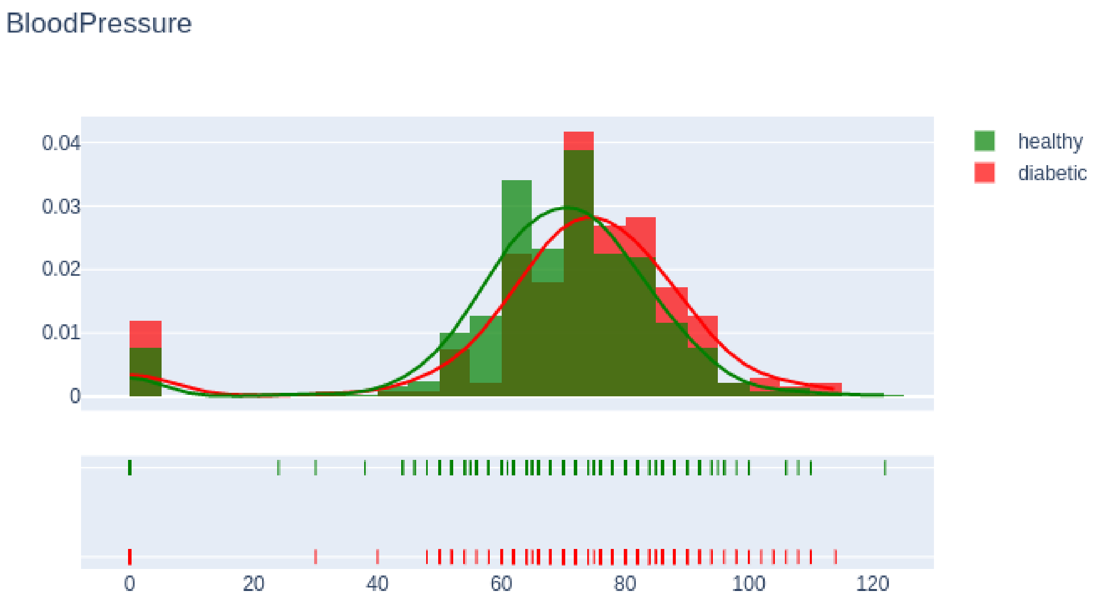


Figure 48: Blood pressure distribution for diabetic and healthy people.

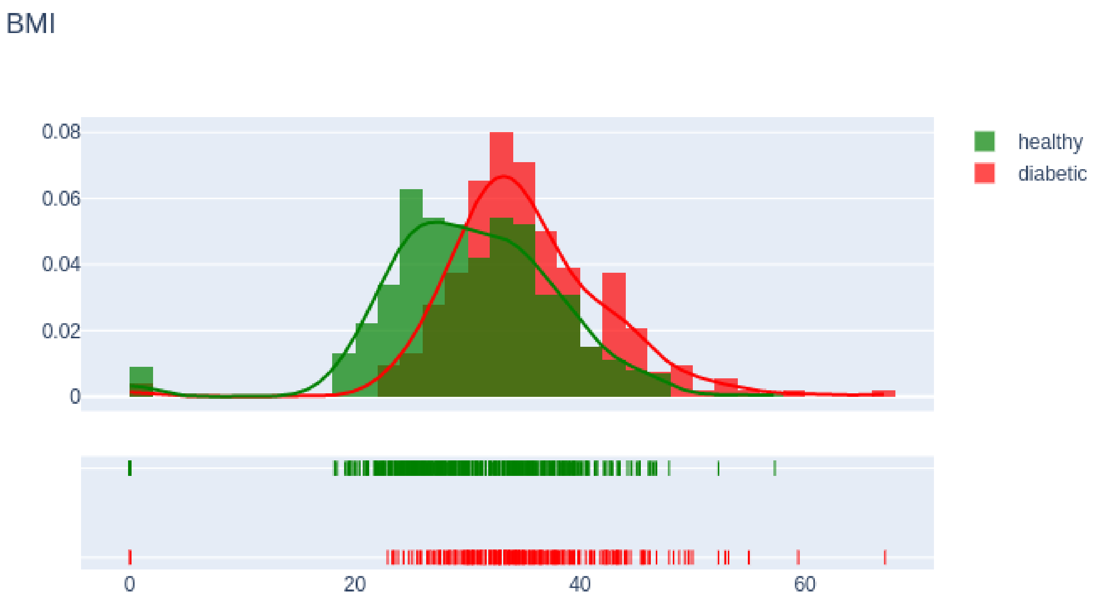


Figure 49: BMI distribution for diabetic and healthy people.

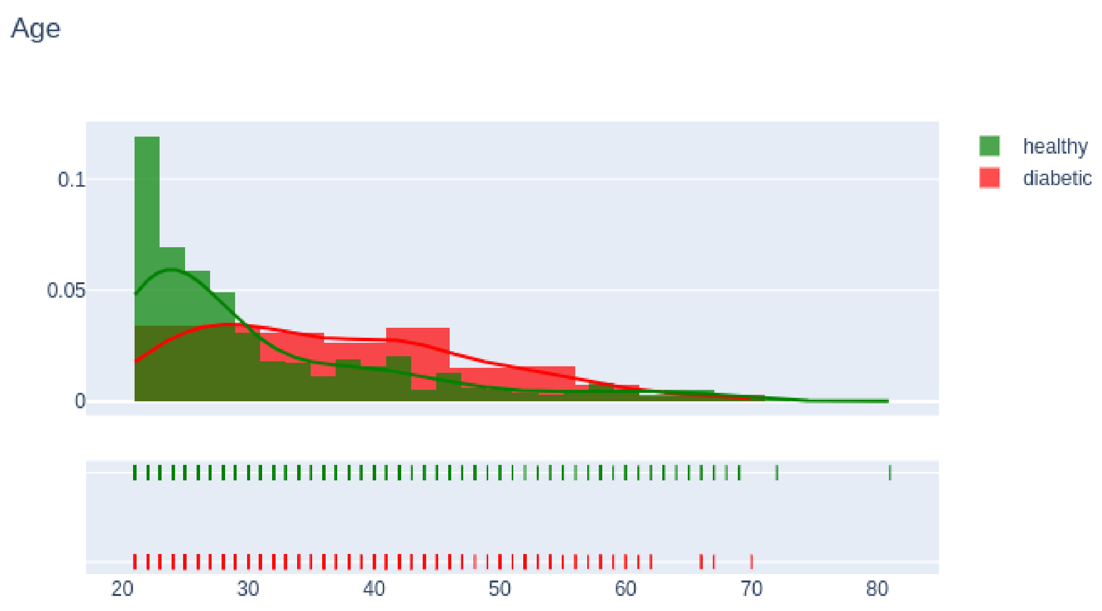


Figure 50: Age distribution for diabetic and healthy people.

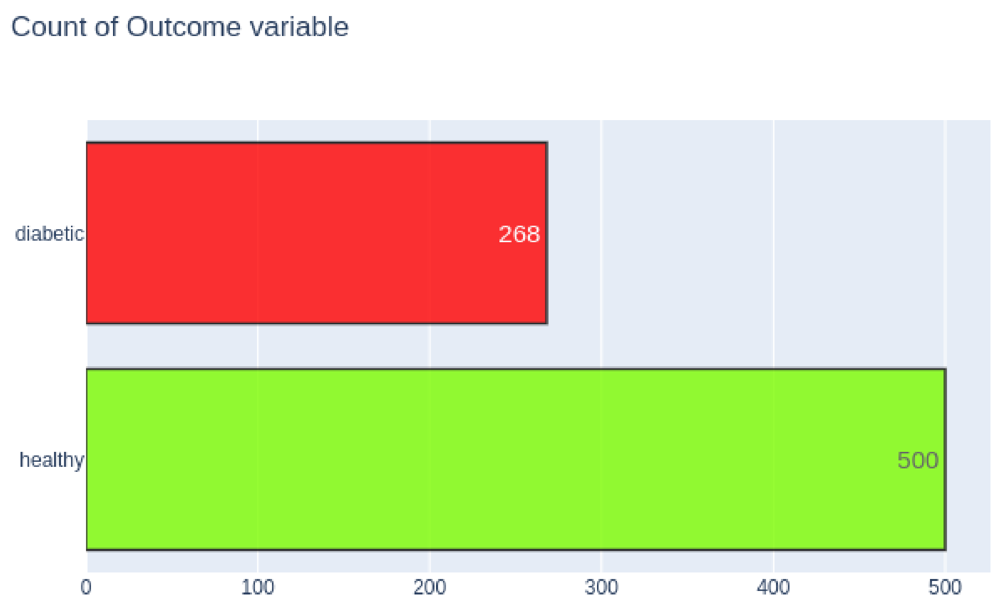


Figure 51: Count of Outcome Variable.

14 Student Profile

15 Conclusion

References

[1] K. Alberti and P. Zimmet, "Definition, diagnosis and classification of diabetes mellitus and its complications. part 1: diagnosis and classification of diabetes mellitus. provisional report of a who consultation," Jul 2004.

[2] S. Watson, "Diabetes: Symptoms, causes, treatment, prevention, and more," Mar 2019.

[3] "1.1. linear models¶."

[4] D. Datta, "Bibliography with the bibtex program," *LaTeX in 24 Hours*, p. 141–151, 2017.

[5] C. Kral, "Github-tutorial," *Modelica - Objektorientierte Modellbildung von Drehfeldmaschinen*, p. 295–317, Dec 2018.

[6]

[7] S. P. Chatrati, G. Hossain, A. Goyal, A. Bhan, S. Bhattacharya, D. Gaurav, and S. M. Tiwari, "Smart home health monitoring system for predicting type 2 diabetes and hypertension," *Journal of King Saud University - Computer and Information Sciences*, Jan 2020.

[8] D. Deva, "Qcon rio - machine learning for everyone," *LinkedIn SlideShare*, Aug 2015.

[9]

[10] P. Kaur and R. Kaur, "Comparative analysis of classification techniques for diagnosis of diabetes," *SpringerLink*, Jan 1970.

[11] L. G. Lisa, L. Gladman, and Lisa, "Types of healthcare information systems - scott-clark medical," *Scott*, Nov 2019.

[12] K. Maladkar, "Why is random search better than grid search for machine learning," *Analytics India Magazine*, Sep 2019.

[13] A. Mandal, "What is diabetes?," *News*, Feb 2019.

[14] P. Mandot, "What is lightgbm, how to implement it? how to fine tune the parameters?," *Medium*, Dec 2018.

[15] S. Norena, "Python model tuning methods using cross validation and grid search," *Medium*, Jun 2018.

[16]