RIGA TECHNICAL UNIVERSITY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

INSTITUTE OF APPLIED COMPUTER SYSTEMS

**Practical assignment 2: Applying methods of machine learning**

**Created by: Umut Kandemir**

**ID:201ADB101**

**GitHub:** https://github.com/UmutKandemir/PracticalAssignment2-Diabetes

**INFORMATION ON DATASET**

**Description:**

In this assignment,” Applying methods of machine learning”, the dataset should be selected according to the specified criteria. While choosing the data set, I paid attention to compliance with the requirements and chose a data set suitable for my area of ​​interest. I also believe it is important to choose a dataset that I will enjoy working on.

Accordingly, I have chosen a dataset that is about predicting diabetes. The dataset is taken from “Kaggle”.

Here is the link for the dataset:

<https://www.kaggle.com/datasets/whenamancodes/predict-diabities>

**Author of the dataset:** AMAN CHAUHAN

**Problem Domain:** The purpose of this dataset is to predict whether a patient has diabetes, based on certain diagnostic measurements included in the dataset. Since diabetes is an important threat to human life, it has to be investigated deeply. Briefly, diabetes is related to the amount of insulin in the human body. If the body cannot use insulin in a proper way or the pancreas cannot produce enough insulin, people can face diabetes disease.

**License:** [CC0: Public Domain](https://creativecommons.org/publicdomain/zero/1.0/)

**Number of data objects in the dataset:** 768

**Dataset Collection:** This dataset is collected from the National Institute of Diabetes and Digestive and Kidney.

Table 1.1 shows the details of the dataset

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type/Range | Role | Description |
| Pregnancies | Numeric / (0-17) | feature | Expression for the Number of pregnancies |
| Glucose | Numeric / (0-199) | feature | Expression for the Glucose level in the blood |
| Blood Pressure | Numeric / (0-122) | feature | Expression for the Blood pressure measurement |
| Skin Thickness | Numeric / (0-99) | feature | Expression for the thickness of the skin |
| Insulin | Numeric / (0-846) | feature | Expression for the Insulin level in the blood |
| BMI | Numeric / (0-67.1) | feature | Expression for the Body mass index |
| Diabetes Pedigree Function | Numeric / (0.08-2.42) | feature | Expression for the Diabetes percentage |
| Age | Numeric / (21-81) | feature | Expression for the age |
| Outcome | Categorical / (0-1)  0 = No  1 = Yes | **target** | Expression for the final result |

Figure 1.1 shows the classes and labels of the dataset:

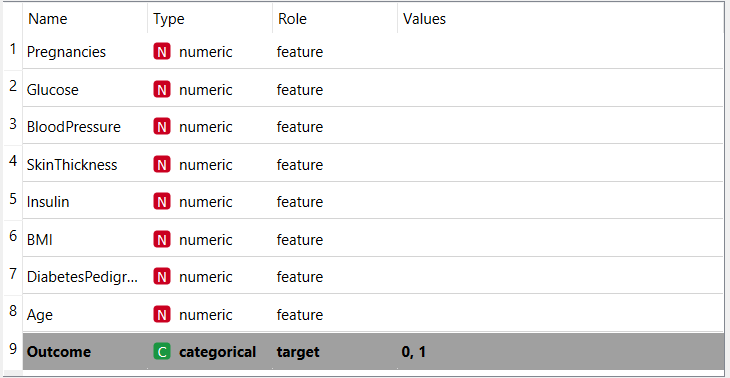
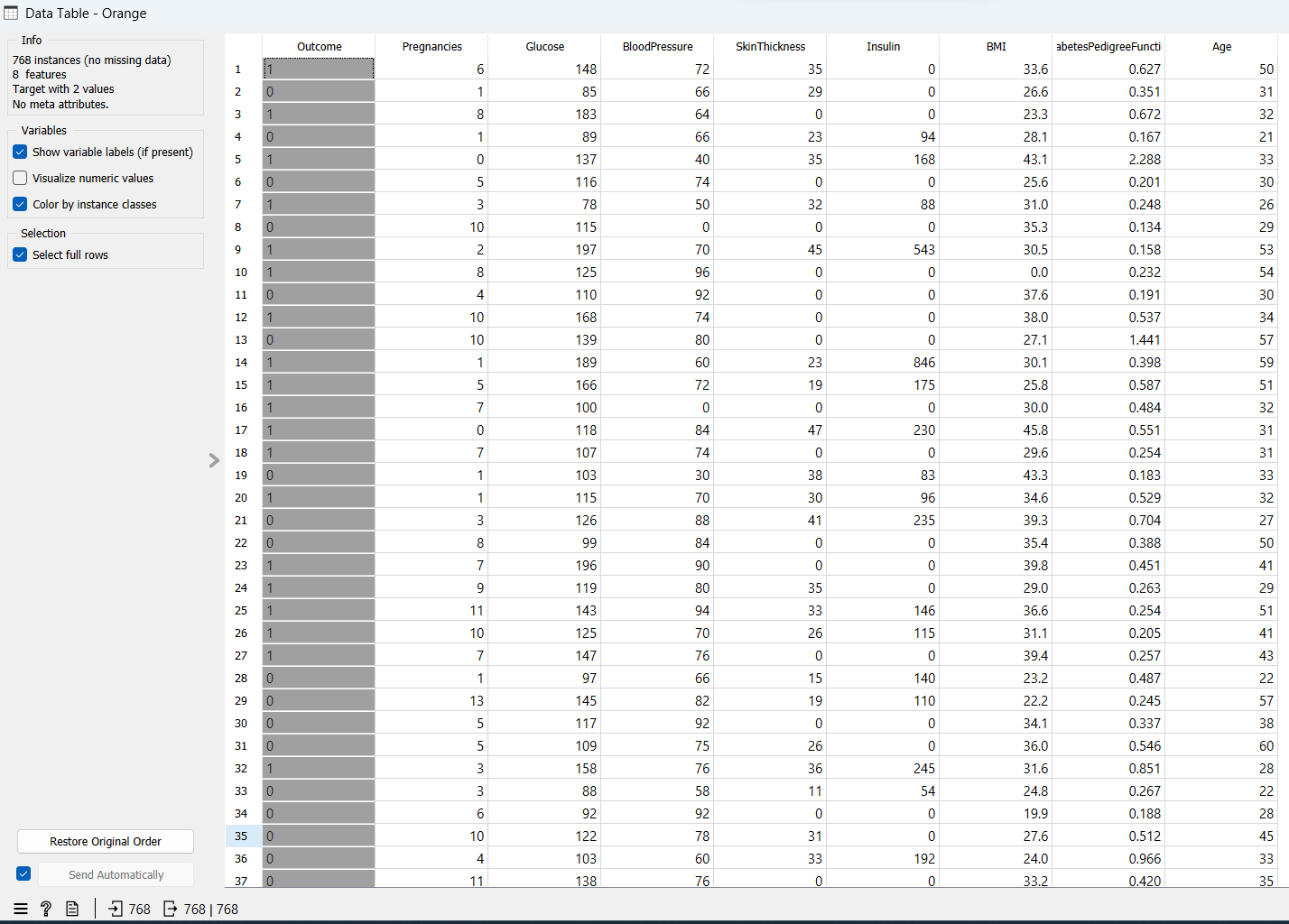


Figure 1.2 shows the part of the data table

****

**REPRESENTING DATASET VISUALLY AND STATISTICALLY**

**Scatter Plot:**

Scatter plots combine the values of two quantitative variables and present them in a graph which will help us to analyze the relationship between two continuous variables.

Figure 2.1 shows us a scatter plot using age and BMI(Body Mass Index) to examine whether a patient has diabetes. Here, red dots indicate that the patient is diabetic and blue dots indicate the patient is not diabetic. We can see that younger patients are more likely to have diabetes. Also, we can observe that Patients with a body mass index of 20-30 who are aged between 20-30 tend to have diabetes. Based on this figure, we can say that having a body mass index in the range of 20-30 may increase the likelihood of patients with diabetes.

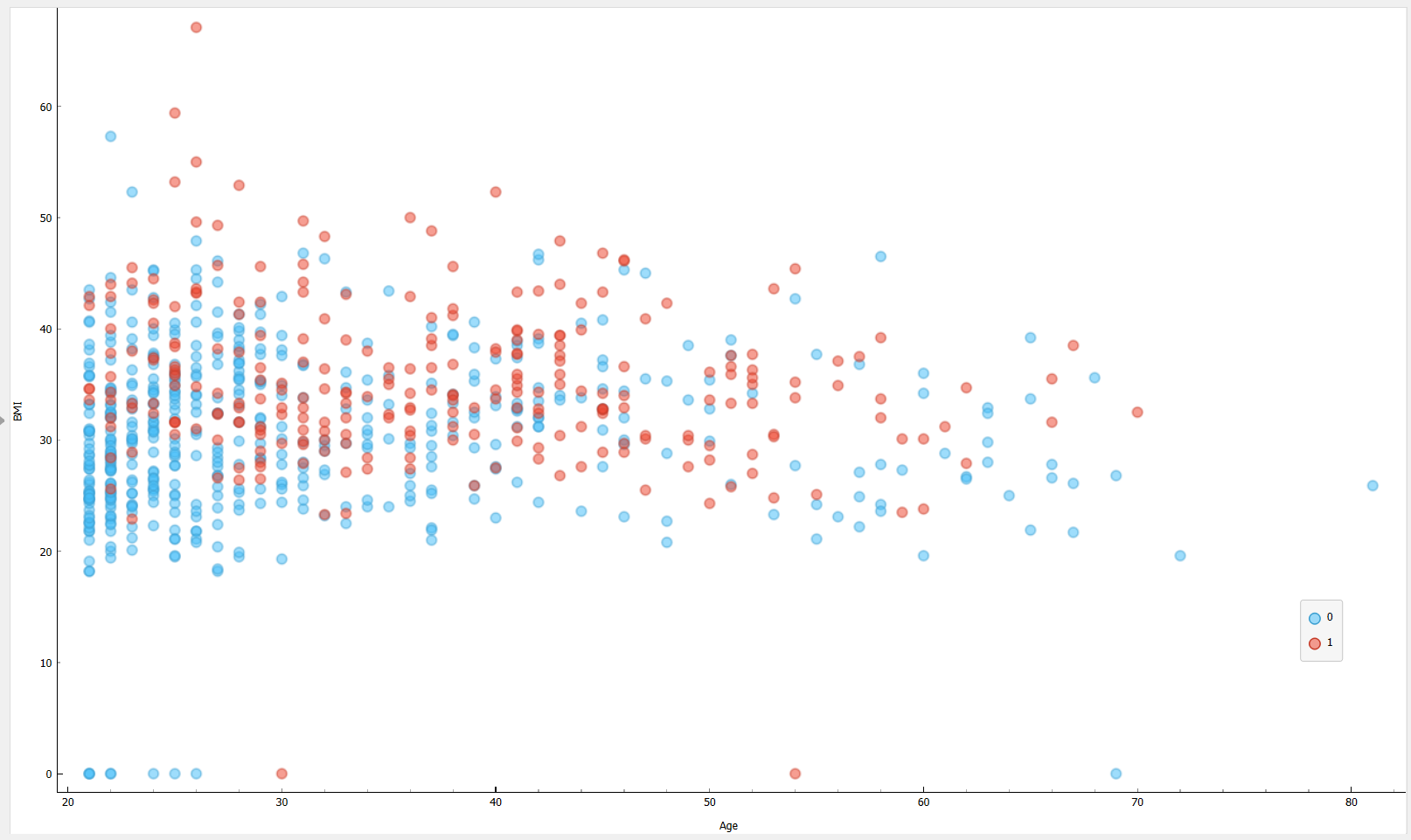


Figure 2.2 shows us a scatter plot using age and glucose to examine whether a patient has diabetes. We can clearly see that patients who has high glucose value tend to have diabetes. Like in Figure 2.1, we can say that younger patients are more likely to have diabetes especially if they have high glucose in their blood. According to Orange’s “Find Informative Projections” feature, the Age-Glucose combination in a plot gives us the best class separability.

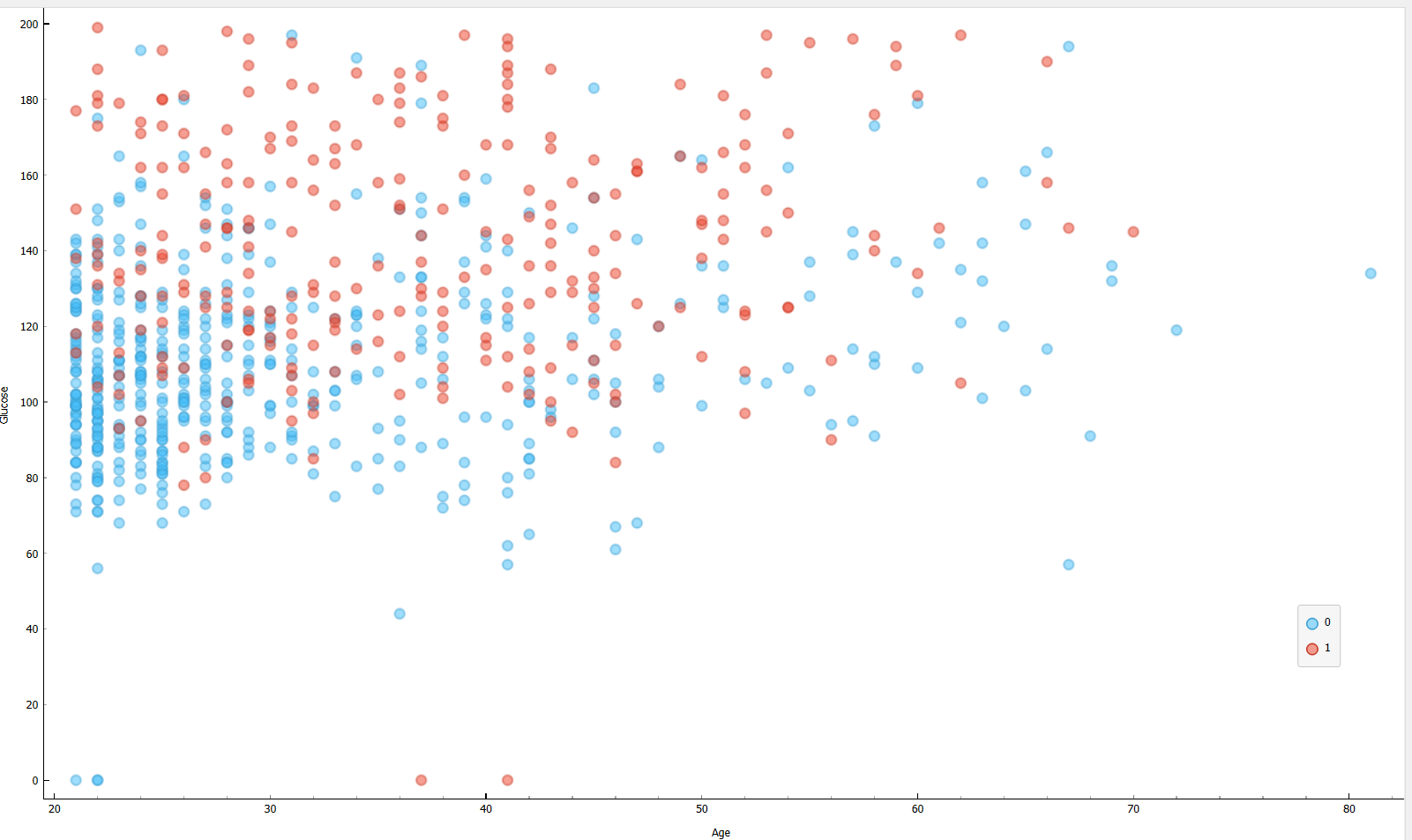
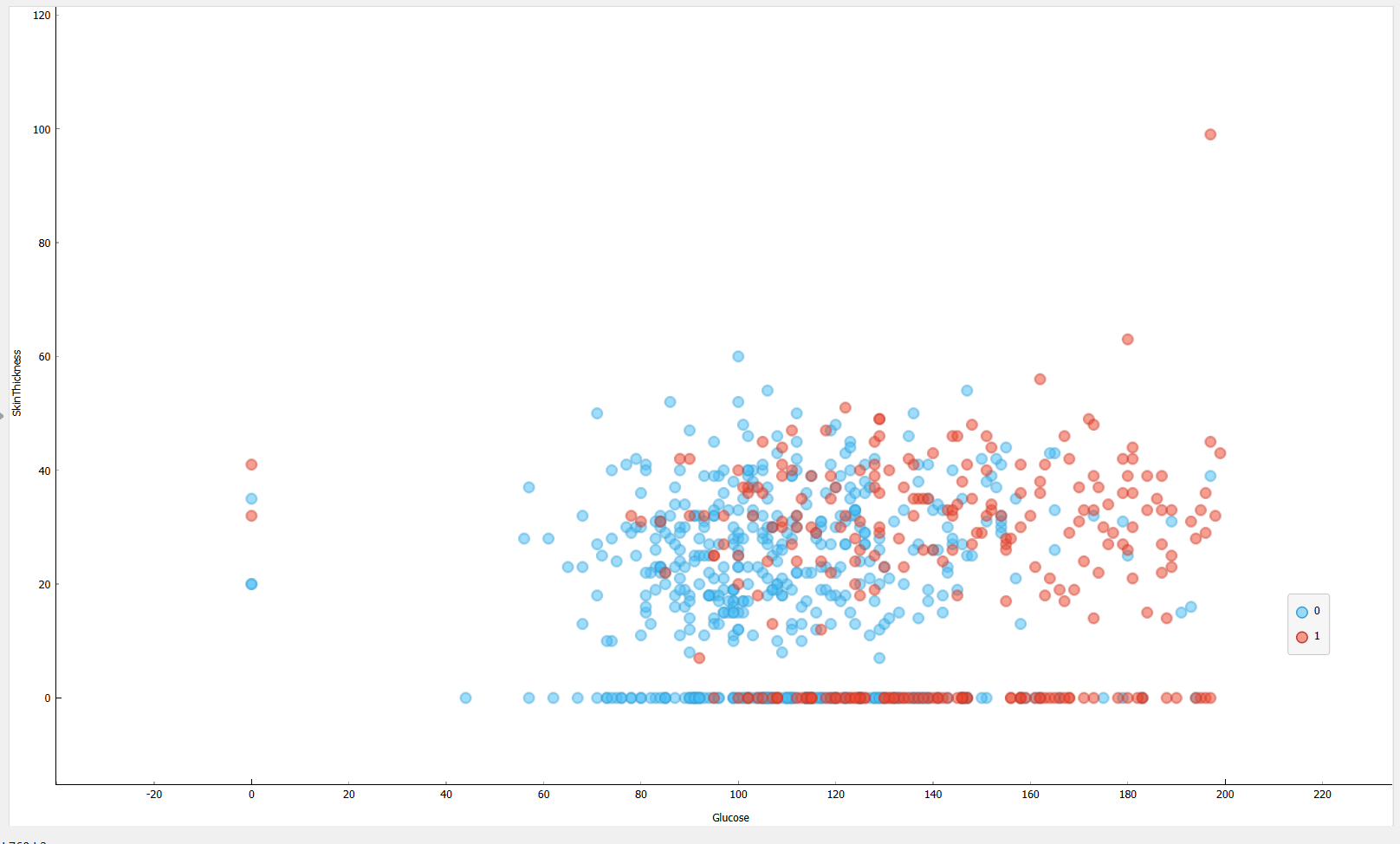


Figure 2.3 shows us a scatter plot using Skin Thickness and Glucose to examine whether a patient has diabetes. We already said before that when there is a high value of glucose, it is more likely to have diabetes. On the other hand, patients who have skin thickness in a range of values 2-40 tend to have diabetes.

**Histograms:**

By using histogram charts in Orangetool, we can examine the continuous data easily.

we will see the representation of the other variables versus the frequency.

Figure 3.1 shows the frequency of having diabetes by age. When we look at the proportions, those between the ages of 28-30, tend to have diabetes with a %36 chance which is the highest chance among other ranges.

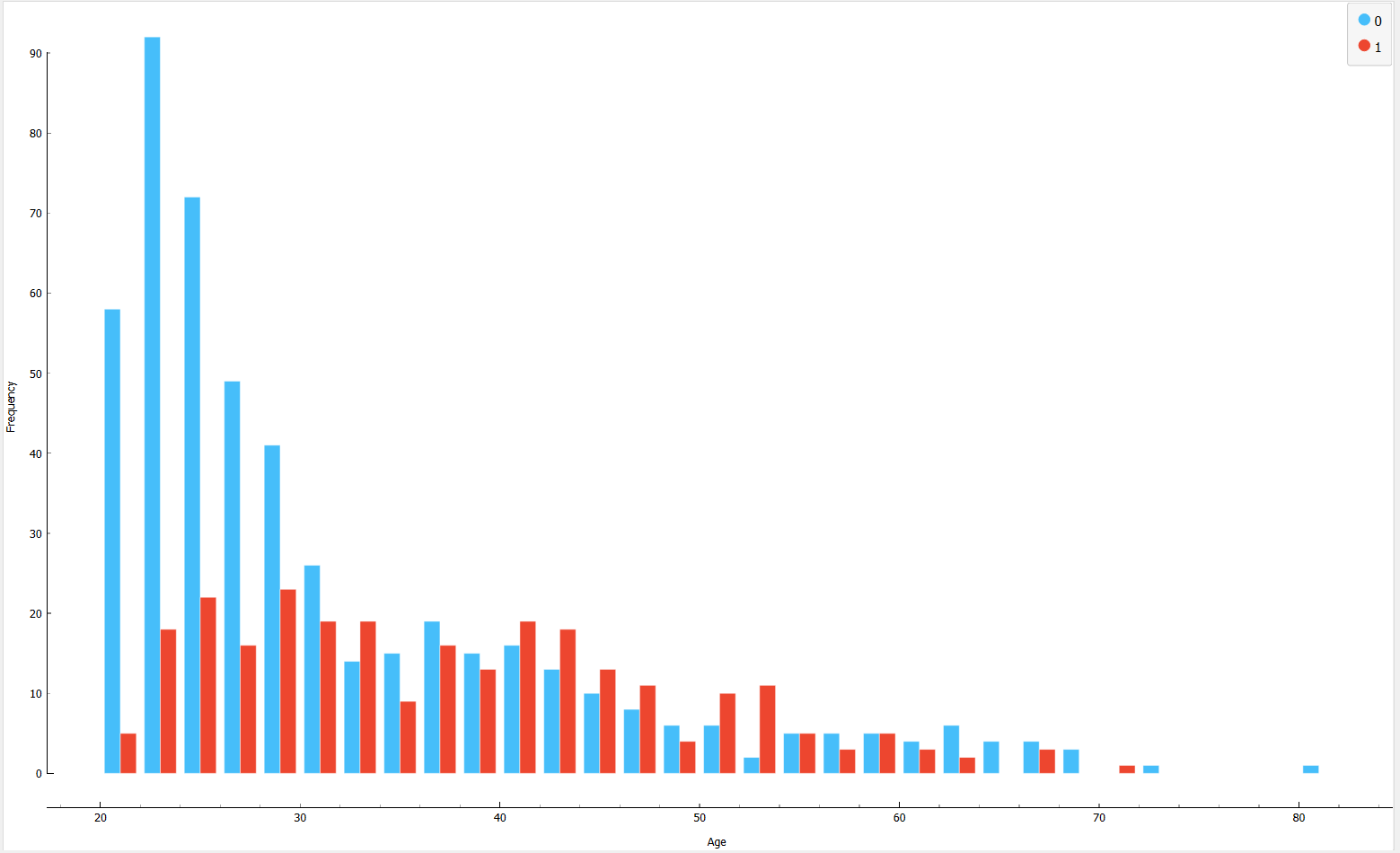


Figure 3.1

Figure 3.2 shows the frequency of glucose amount in the blood. Glucose amount is a huge factor to determine whether a patient will have diabetes or not. In this case, the range 125-130 has the highest chance to have diabetes. %40 patients has diabetes in this range which shows that 125-130 value Range is critical for patients.

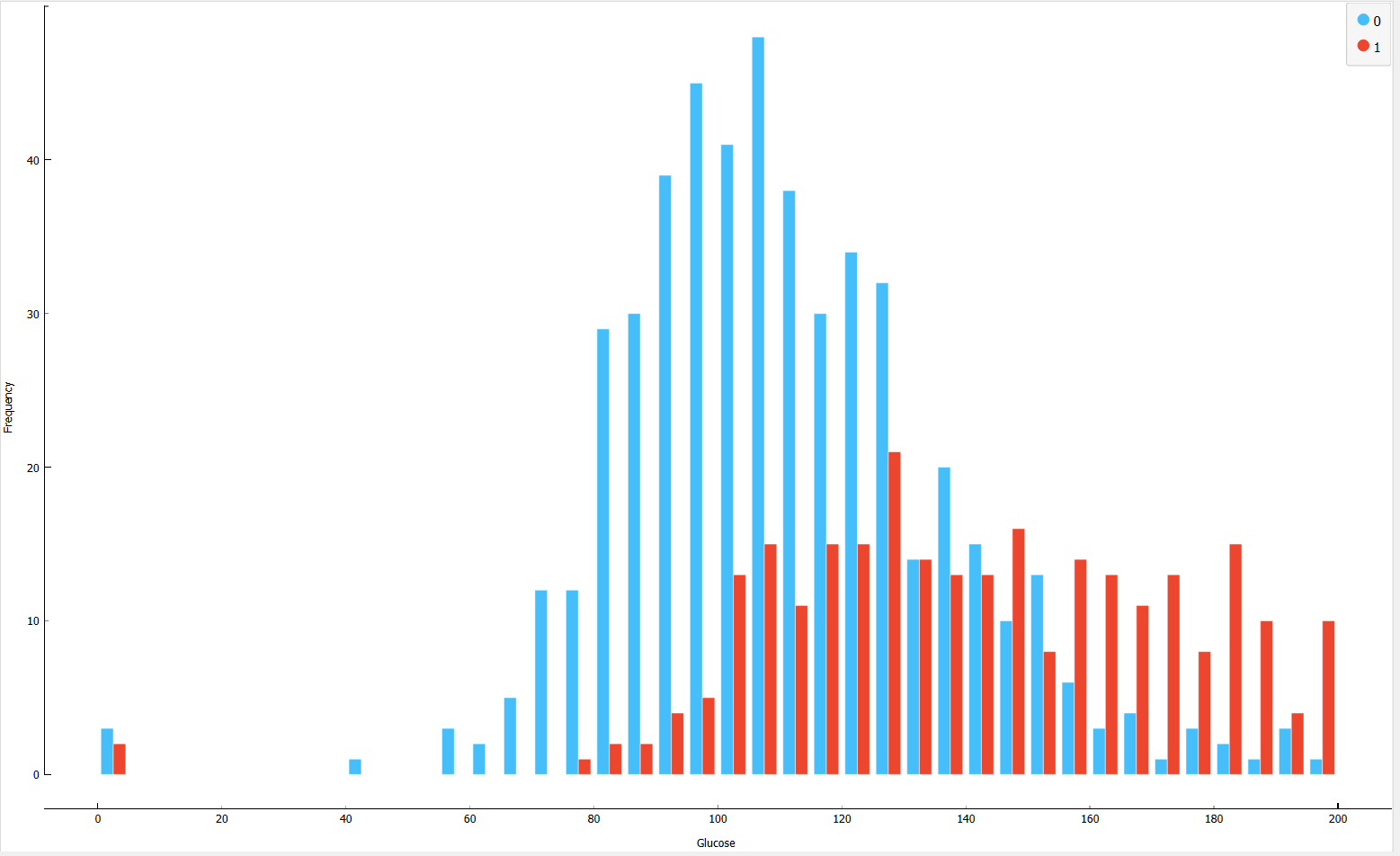


Figure 3.2

Figure 3.3 shows the effect of BMI on the frequency of having diabetes. At a BMI range of 30-36, %45 of patients are having diabetes and that means this range is critical for patients.

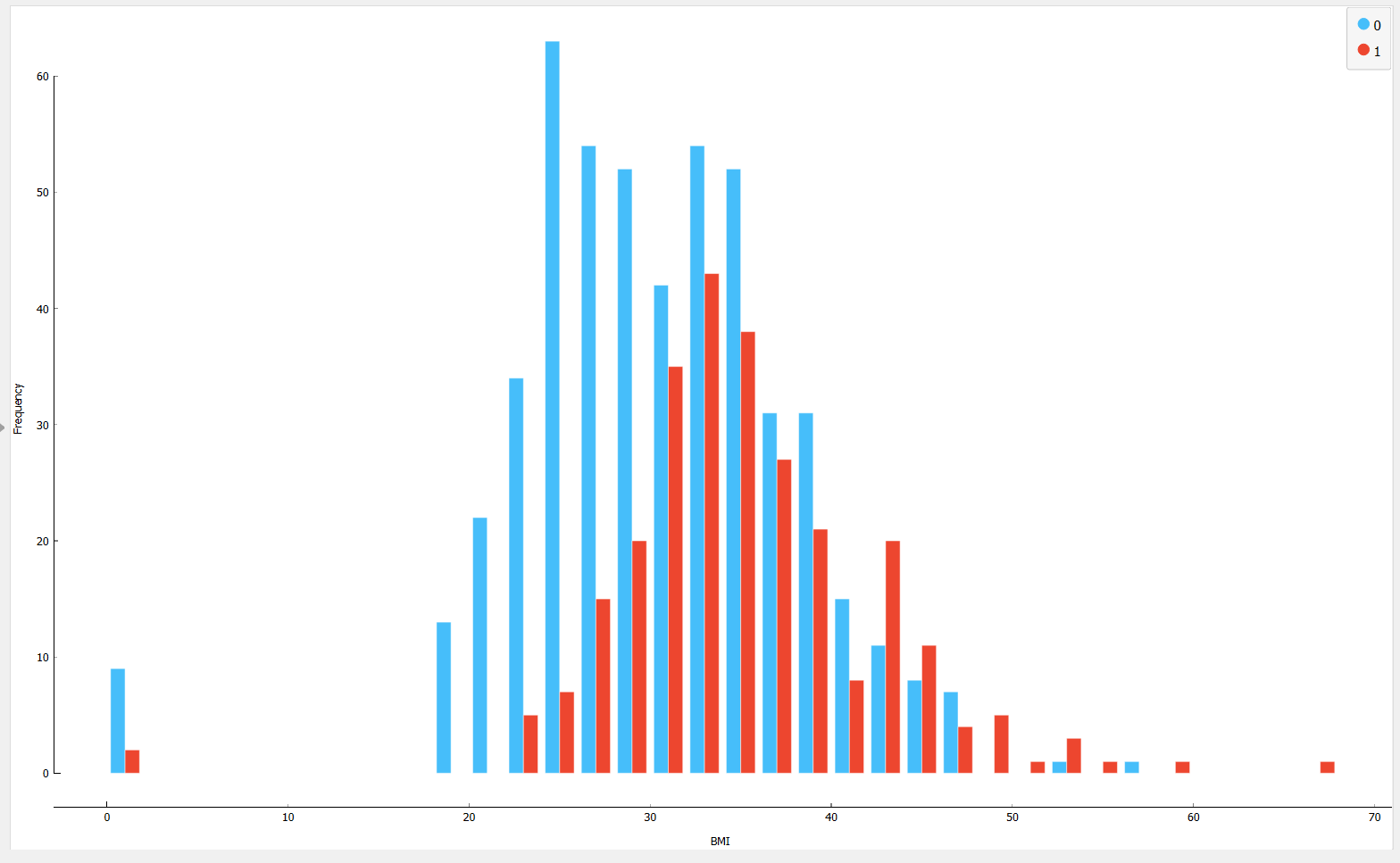


Figure 3.3

After examining all graphs, I came up to the conclusion that Age, Glucose, and BMI are good predictors to predict whether a patient has diabetes. Below we can see the full distributions from features, all of them not mentioned above because some of them don’t provides better separability compared to the ones that I mentioned.



Figure 3.4

**Distributions:**

Distributions will help us to see the value distribution of data as in the previous examples. Here we will see some additional values in the graphs such as mode, mean values, and dispersion value which shows the distribution of data around a central value.

There are many distribution options in the Orange tool, but we will use normal distribution in this case. Figure 4.1 shows the normal distribution for age features. We can again examine the age distribution like in the previous examples but in this case, we can see the mean values and standard deviation also. The mean value of patients who has diabetes is 37.07 and for patients who don’t have diabetes is 31.19. Their standard deviation is 11.16 and 10.95.

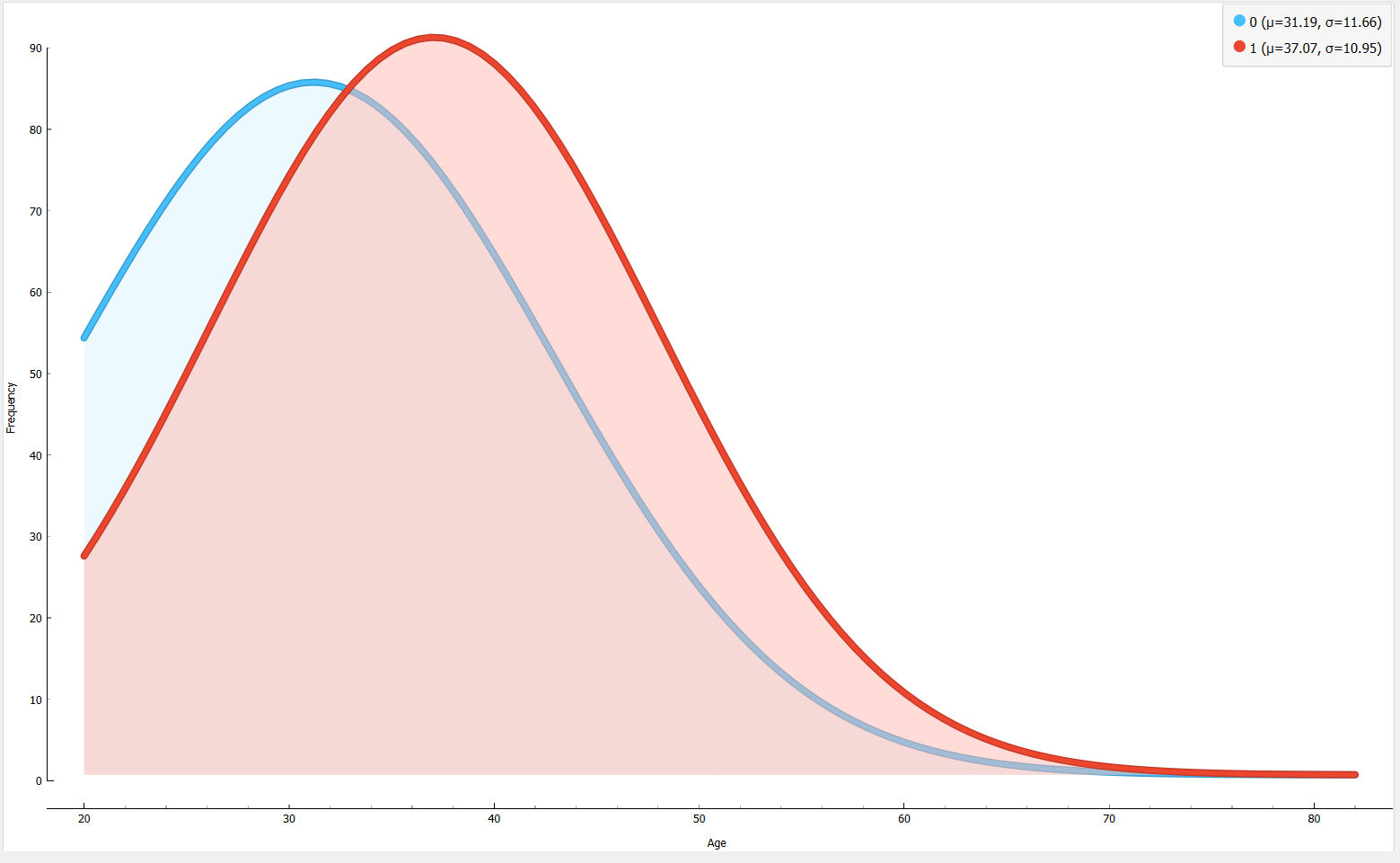


Figure 4.1

Figure 4.2 shows the normal distribution for BMI features. The mean values are similar to Figure 4.1. One thing to mention is that; mean, median and mode values are almost the same when we check Figure 3.4. We can say it is an almost perfect normal distribution

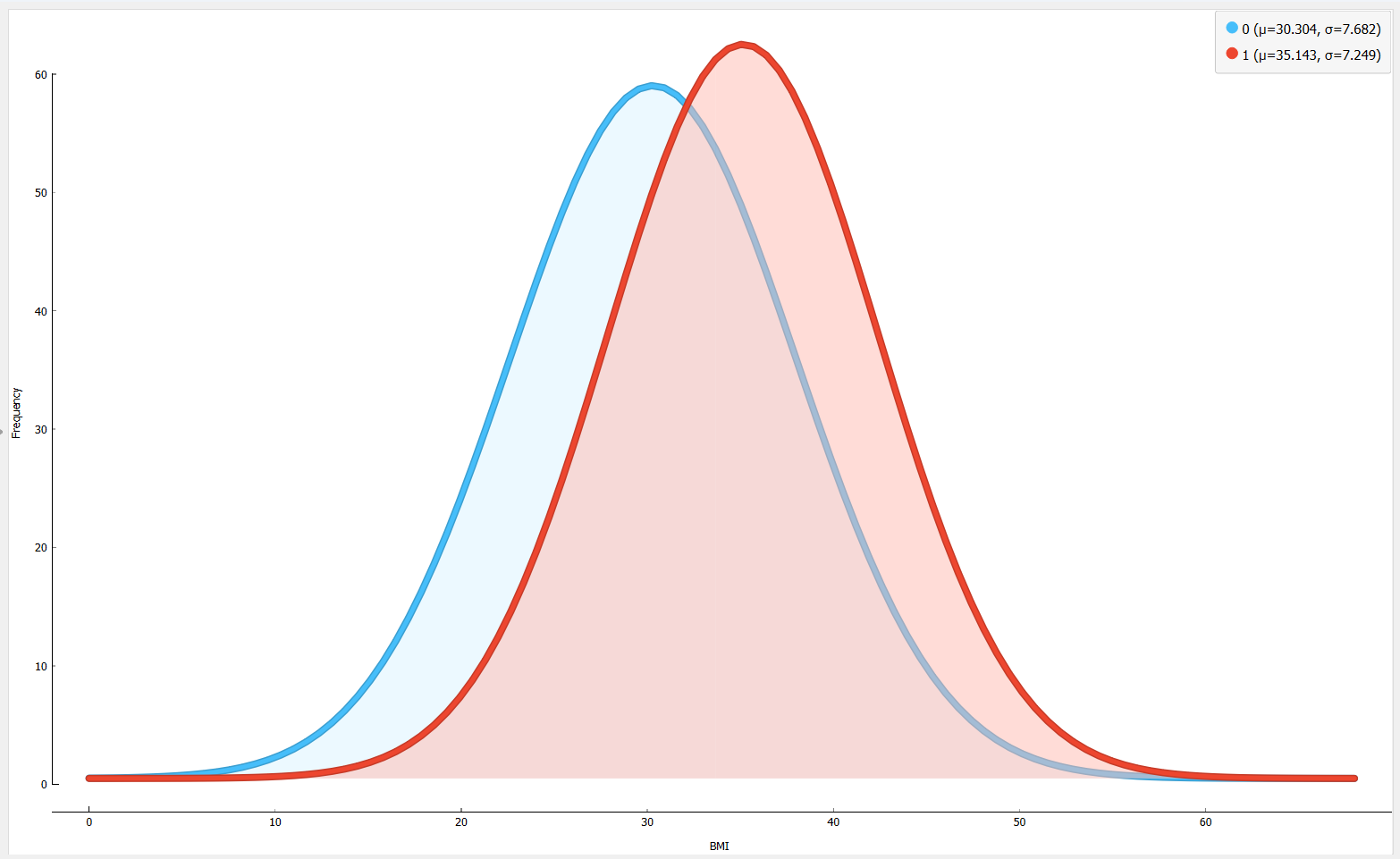


Figure 4.2

**Correlations:**

Here we will examine the correlations between features, the goal is to see how our explored data relates to each other. Previously I mentioned that Age, Glucose, and BMI have significant affect to predict whether a patient has diabetes. By using the correlation feature of Orange, we will have the opportunity to check their correlations with each other.







After checking the correlations data, we can say that Age-Glucose, BMU-Glucose and Age-BMI have a correlation with each other.

**UNSUPERVISED LEARNING**

**Hierarchical Clustering:**

After the data is explored above, we can use unsupervised algorithms to better understand the structure of the data. In this case, we will use hierarchical clustering. It is an machine learning algorithm that will determine the optimal number of clusters. Each data point will be placed in its own cluster. By calculating the distances of the distance matrix, the algorithm is combining two points that are closest to each other. In each step, the distance between clusters is calculated and stored.

There are some hyperparameters we have to consider while applying the clustering algorithm. In order to calculate distance, these parameters are essential.

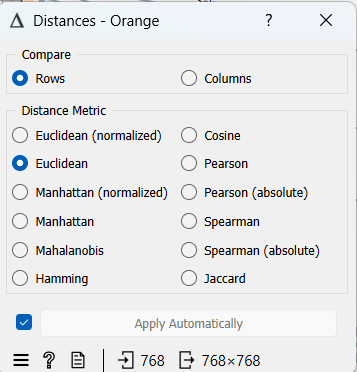
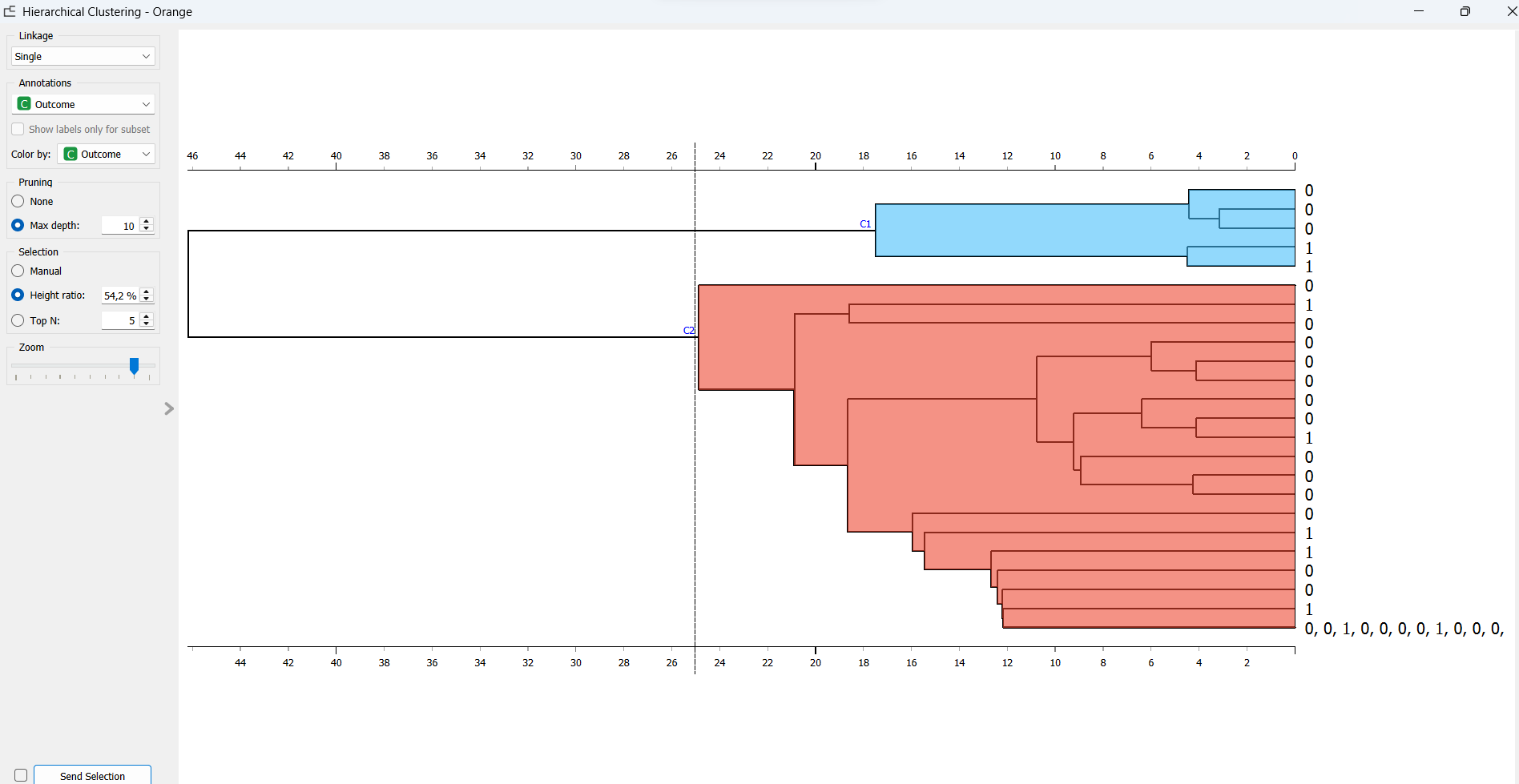


Figure 5.1

**Compare:** We can choose where we want to calculate the distance between (rows or columns).

**Distance metric:** Will choose which method we want to apply

Figure 5.2 shows clustering for our outcome(diabetes or not). As I set below, it measures the distances between rows (depth=10). The height ratio is %54.4 in this case. I tried to find better separability of my classes by moving the line. By looking at this dendrogram, we can say that the cluster didn’t separate well enough since there are some mixed outcomes on the right-hand side.



**Figure 5.2**

**Hyperparameters**

**Single linkage:** calculates the distance between the two closest elements.

**Average linkage:** calculates the distance between two clusters.

**Weighted linkage:** calculates the distance between pair groups, example:A1-A2-A3 and creates a new cluster called A4 by combining pairs.

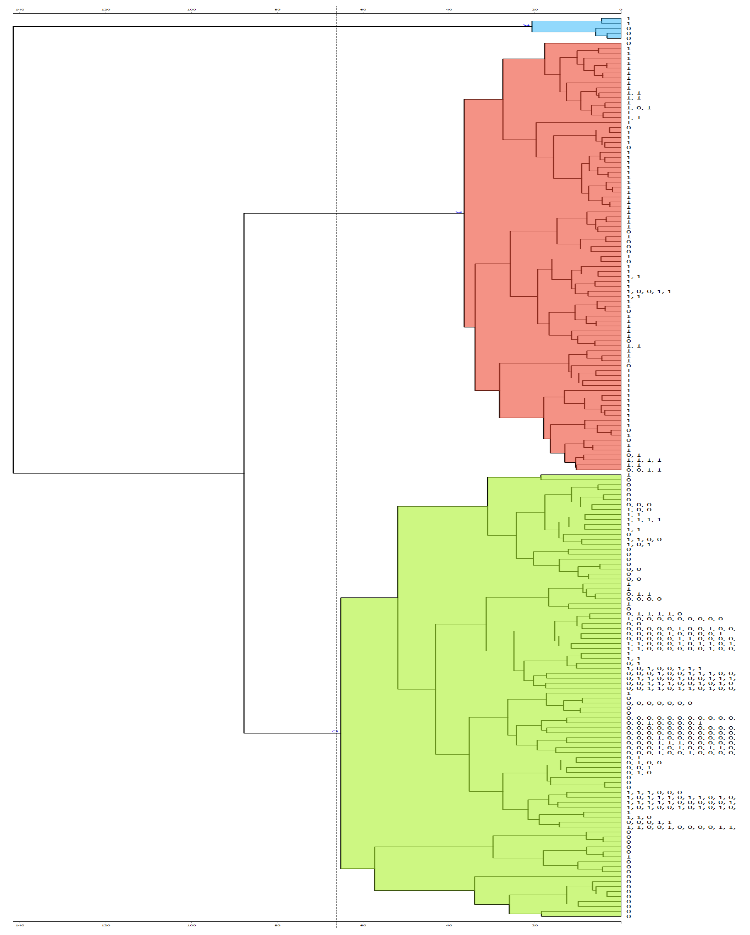
**Complete linkage:** calculates the distance between the farthest clusters.

**Ward linkage:** calculates error sum.

**Experiments:**

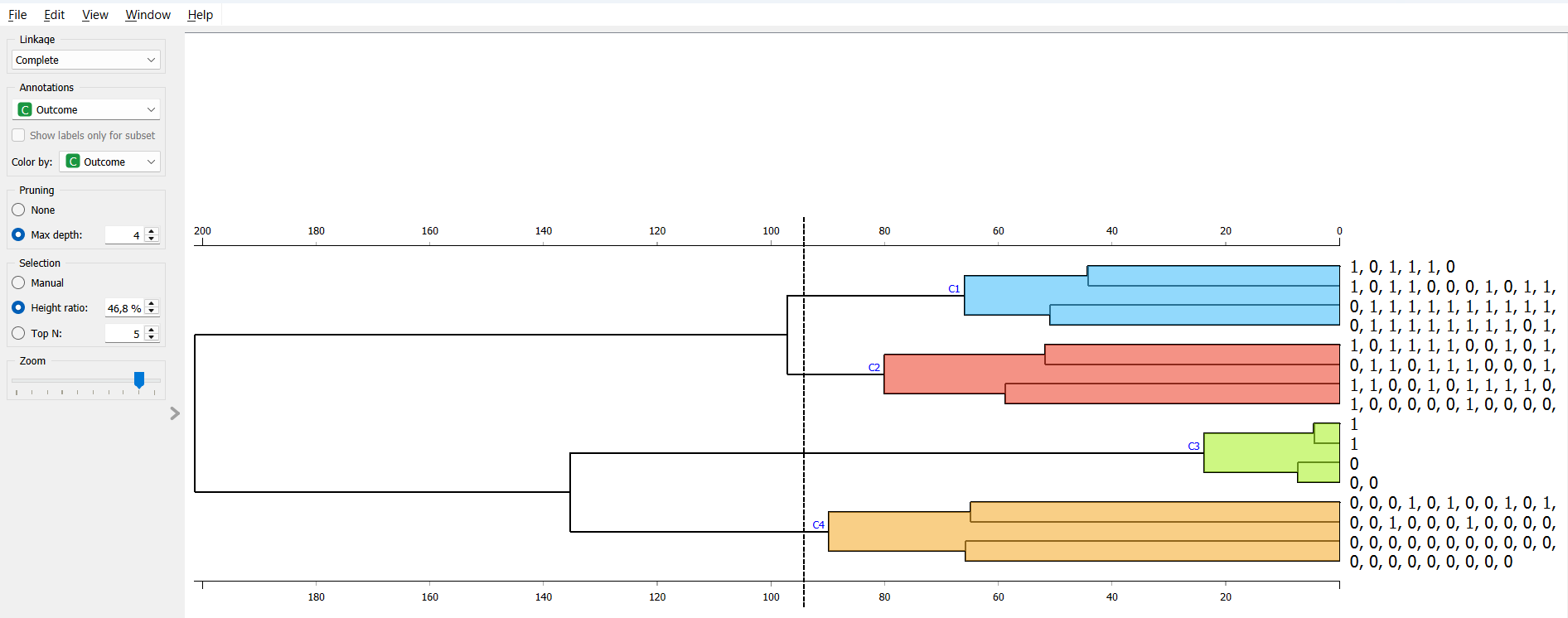
1. Linkage: Weighted , max depth:10

Here, we can see that when we select “weighted” linkage, we will have one more additional cluster, and height ratio moved to %46.8

****

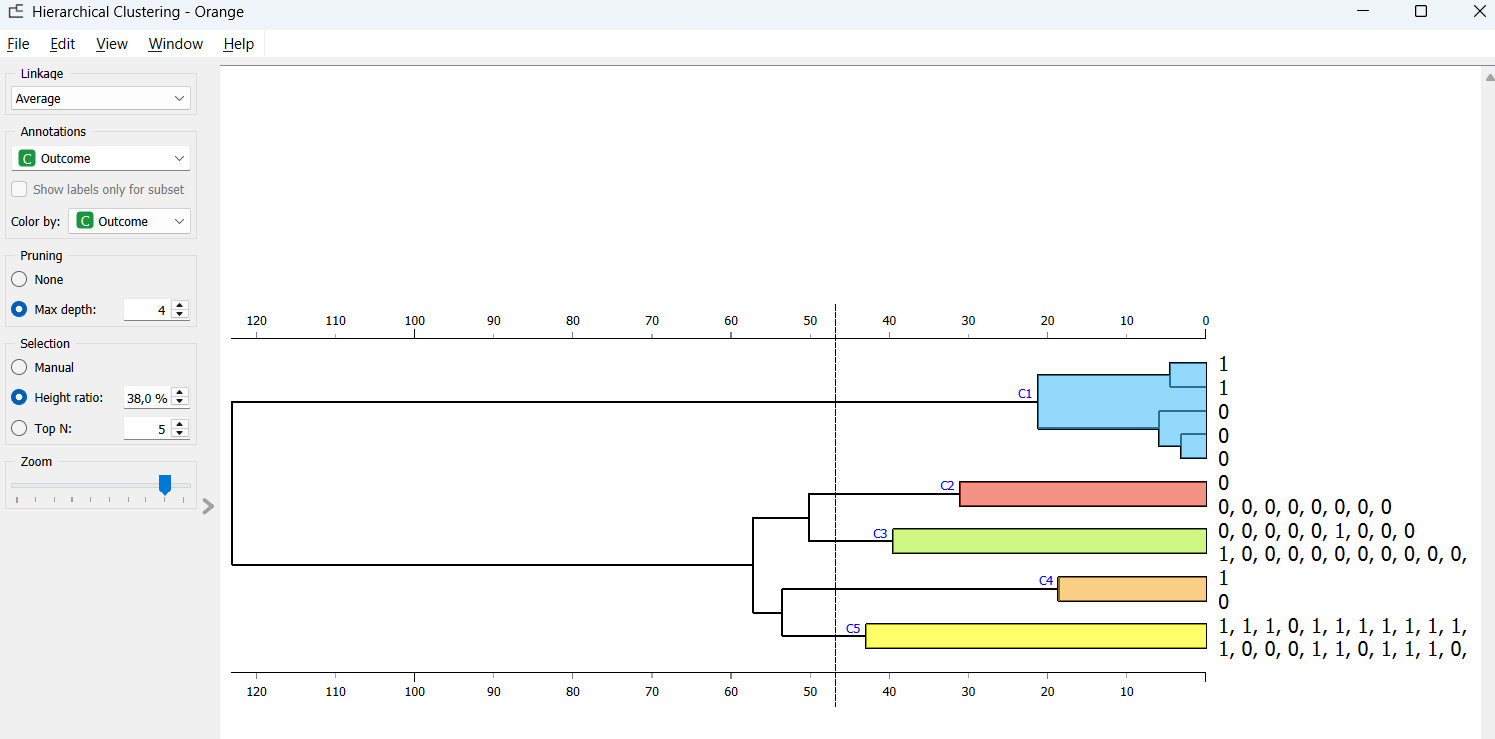
1. Linkage: Complete , Max depth:4

Here, we can see that we changed the linkage to Complete and we get 1 more additional cluster. We have 4 now. In order to see it easily, I set the max depth as 4



1. Linkage: Average, Max depth: 4

Here, I set the linkage to average and additionally, I moved the line to the right side to show what it looks like. In this case clusters divided itself and we have 5 small clusters.

****

**K-means clustering:**

We can define it as can be characterized as the process of locating subgroups within data because data points within the same subgroup (cluster) are quite similar while those inside different clusters are very different. In this case, we will use the silhouette score in order to evaluate the clusters. This score will change within the range -1,1 and a higher score will mean better clustering. According to these informations, I will make experiments.

I will specify a range for the number of clusters to be created. The algorithm will calculate the silhouette scores accordingly. A maximum number of iterations will be set.

The below figure shows us how many clusters are needed. We can see in the figure 5.1 that the second cluster has the highest value so that means I have to use it.

**Informations about hypermeters:**

**Fixed:** We will define fix clusters, which means specific numbers will be defined.

**From:** It represents the cluster range which will be defined by me (below).

**Normalization:** If we didn’t normalize our columns before, we have to do it here.

**Random initialization:** Initially, all clusters in the range will be assigned randomly.

**Re-runs:** It shows how many times we want to run the algorithm

**Maximal iterations:** It Showsiterations per run

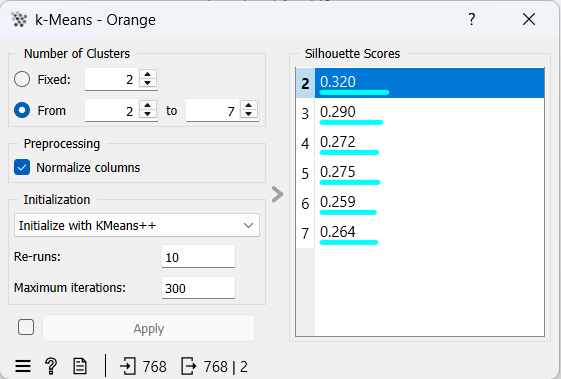


Figure 5.2 shows us the silhouette plot. We can observe the clustering of diabase patients and not diabase patients.

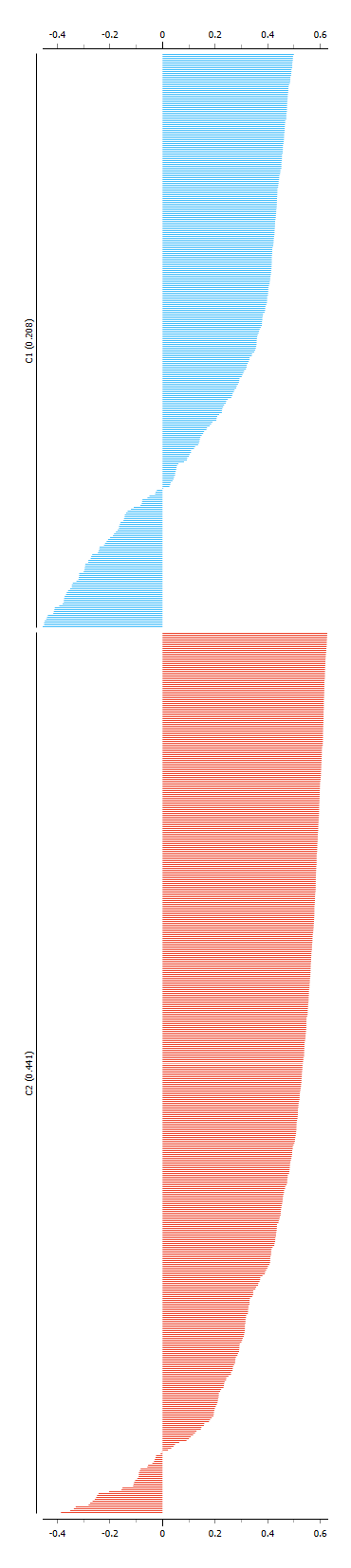
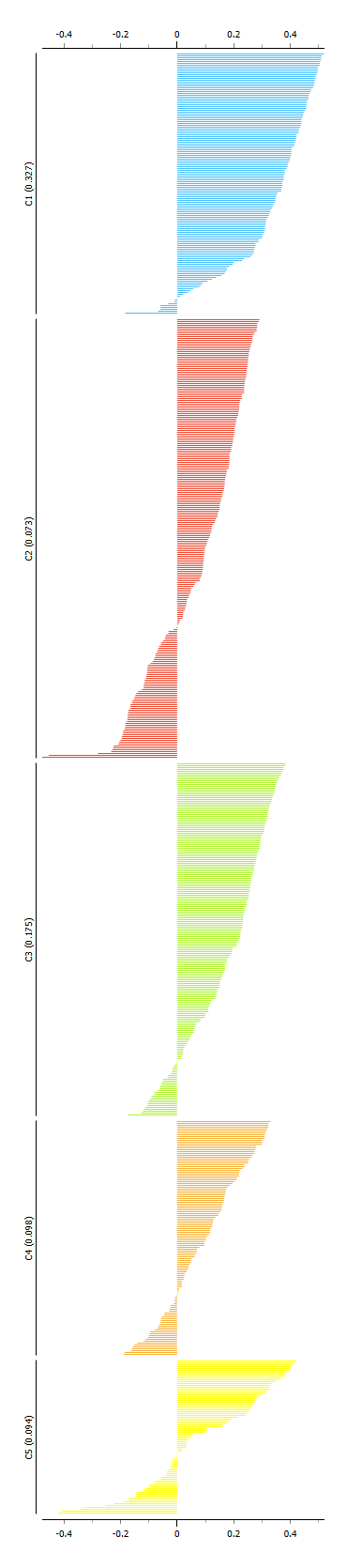
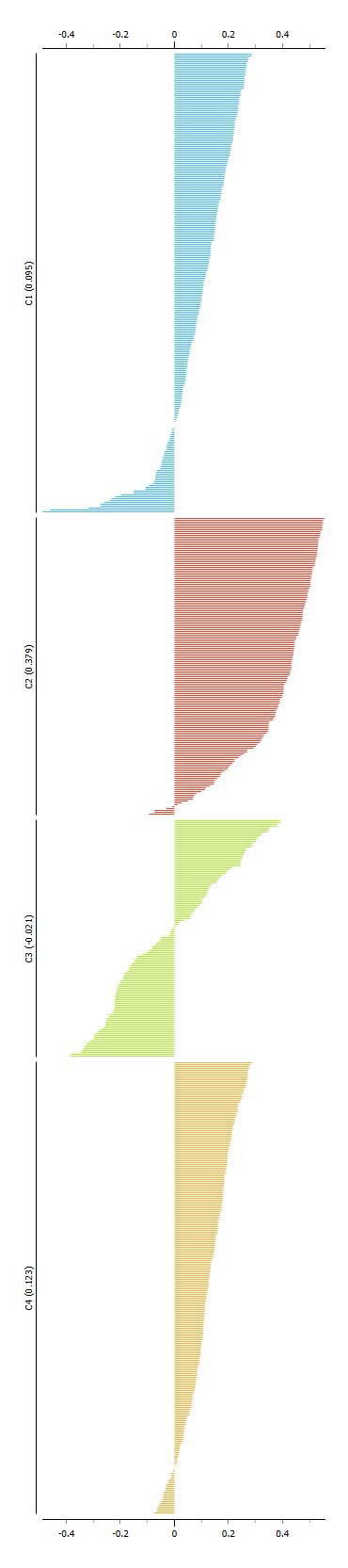
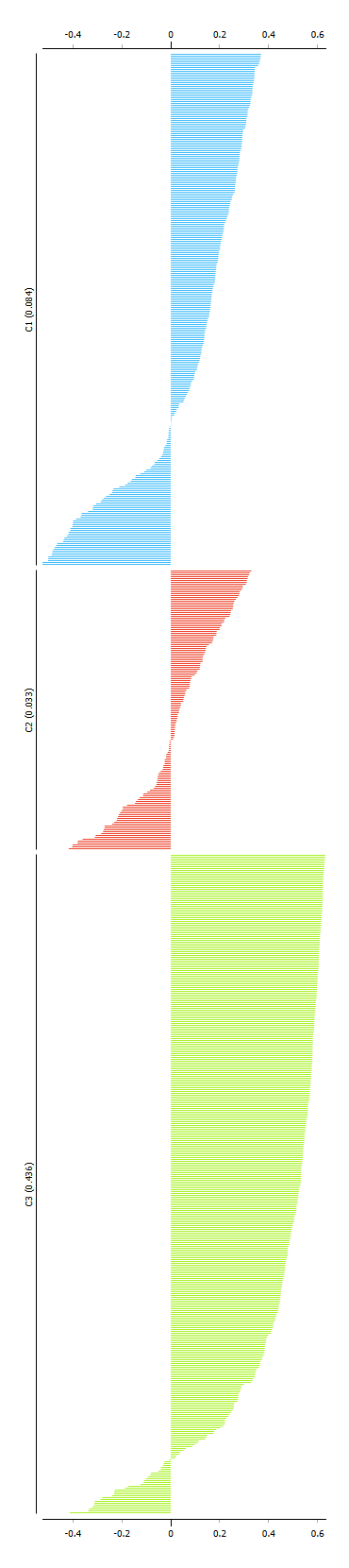


Figure 5.2

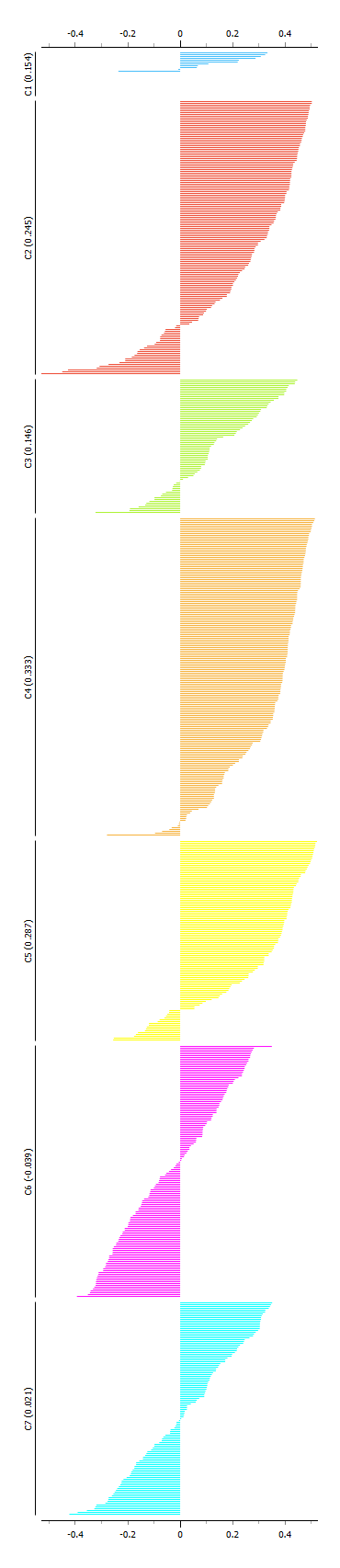
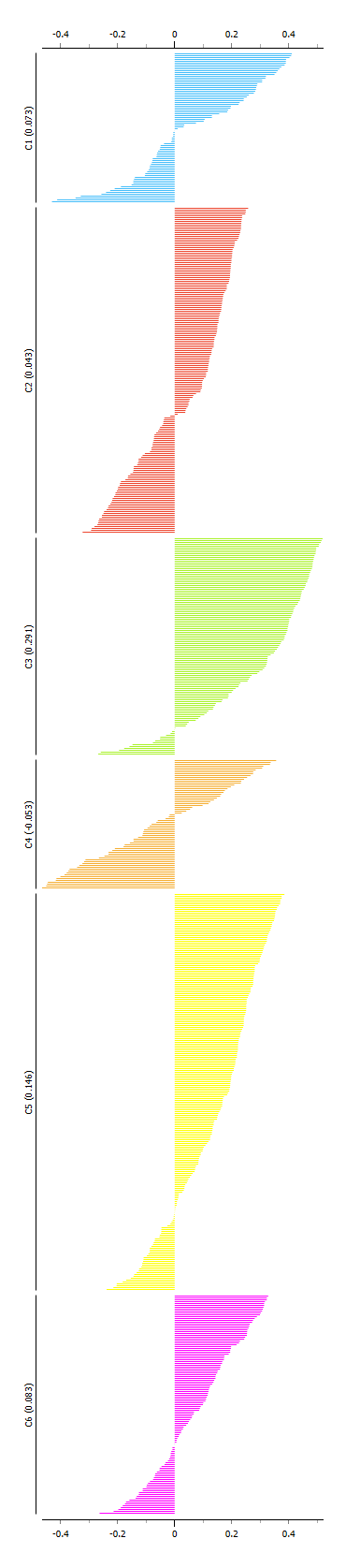
**Experiments:**

Experiments will be done by calculating 5 different k values.

1. k = 3 2) k = 4 3)k = 5



4) k = 6 5) k = 7

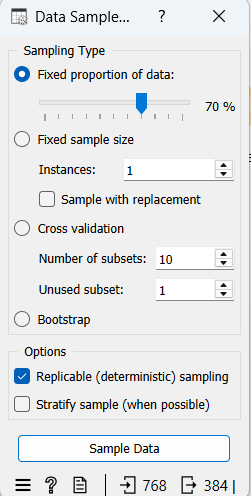


As we can see from the silhouettes, scores are close to “0” in every case. This means distances between clusters are not significant. We also have negative values in each experiment and this means there is the possibility that those are assigned to the wrong cluster. The farthest value from zero is 0.6, which is nice, it represents clusters located far from other clusters. To conclude, the results are not as effective/sufficient as expected. We obtained these results by changing parameters each round, changing the k value, re-run value, and max iteration value.

**SUPERVISED LEARNING**

In order to split the dataset for training and testing, we need to use the “data sampler” element at the Orange tool.

Below, I specified the proportion of data objects at %70, which means %70 of data objects will be the training dataset and %30 of the data objects will be the test dataset.



I will choose 2 supervised methods: kNN and Neural Networks since we already discussed them in class and in the demonstration videos.

1. **kNN:**

It is used for classification tasks and in this case it will be used to define closest input to the output.

**Hyperparameters:**

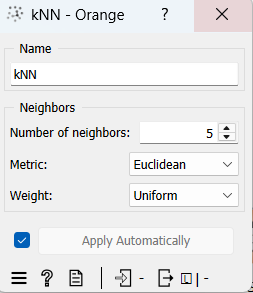
**Name:** name of the widget.

**Number of neighbors:** It represents the nearest number of neighbors.

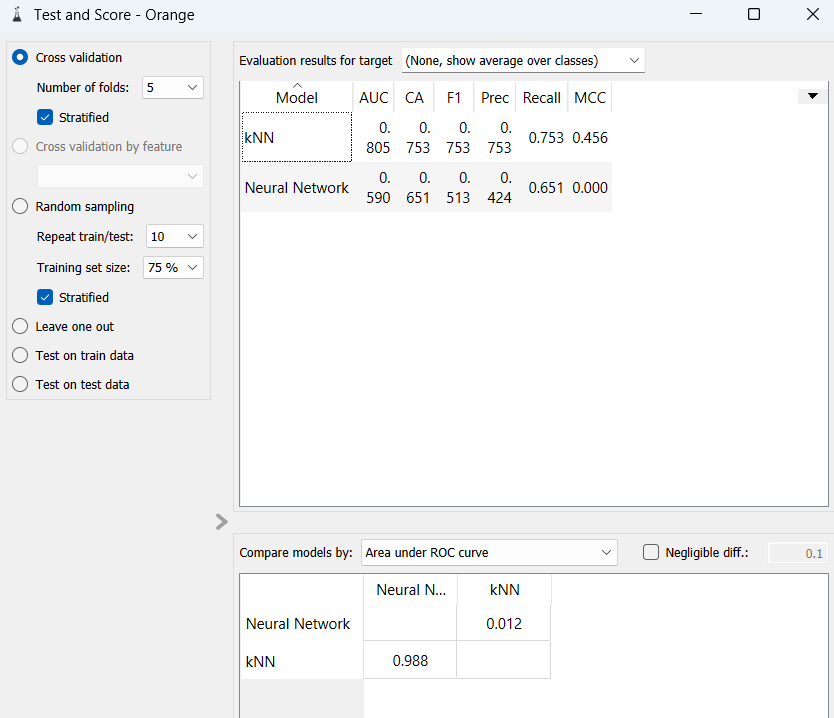
**Metric:** Shows the distance parameter.

**Weight:** Represents model criteria.

Initially, I set the hyperparameters as follows:

****

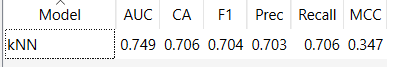
After I defined the algorithm hyperparameters, I added the element “Test and Score” to compare the performance. of both algorithms. After making the adjustments , I reached the result below. This result shows us a comparison of the performance of algorithms. As we can see, the Knn algorithm has better results.

****

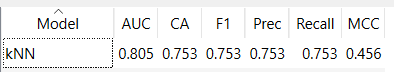
Now, I will examine the performance of kNN separately by changing hyperparameters.

**Experiments:**

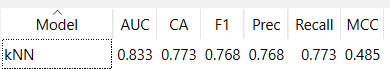
1. Number of neighbors: 3, Metric: Euclidean, Weight: Uniform



1. Number of neighbors: 5, Metric: Euclidean, Weight: Uniform

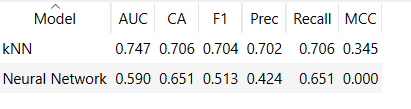


1. Number of neighbors: 10, Metric: Euclidean, Weight: Uniform



From these 3 experiments; we can observe that when the number of neighbors increases algorithm provides best accuracy.

1. Number of neighbors: 3, Metric: Euclidean, Weight: by distances



Here, we can see that changing Weight “uniform” to “by distances” does not change anything.

1. **Neural Network:**

**Hyperparameters:**

**Name:** name of the widget.

**Neurons in hidden layers:** shows the number of neurons.

**Activation:** defines applied function.

**Solver:** sets weight.

**Maximal number of iterations:** possible number of iterations which will be set by me.

In this case, I will try to understand the relation between the neurons and layers. So, other hyperparameters will remain the same such as :

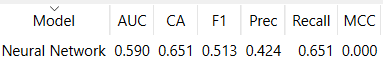
Activation: Logistic

Solver: SGD

Maximum number of iterations: 1000

**Experiments:**

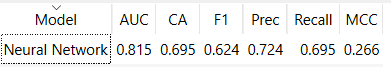
1. Hidden layer: 100,100



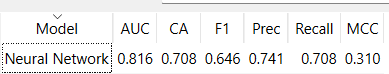
1. Hidden layer: 100,100,100



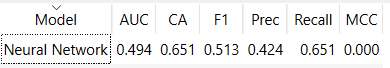
1. Hidden layer: 100



1. Hidden Layer 120

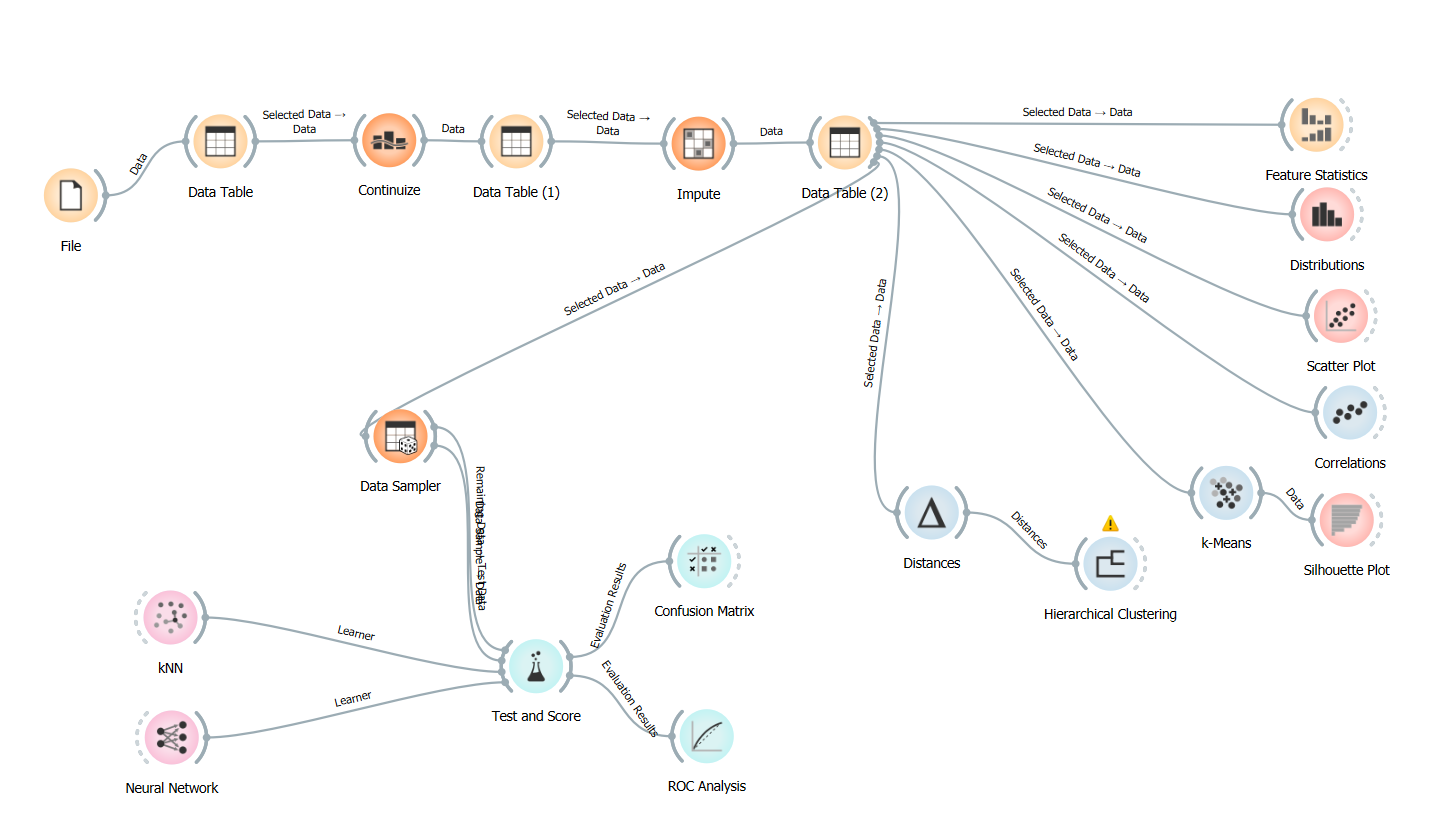


1. Hidden layer: 100,50,70



I changed the number of neurons and layers in each experiment and examine the outcomes. Nothing significantly happens when we have 2 layers or 3 layers. But when we have 1 layer, the algorithm works well. In this case, if we increase the neuron inside that single layer, the output becomes more accurate. If we will change the neurons in 3 layer case, nothing will change.

Final Output:



References:

<https://orangedatamining.com/widget-catalog/unsupervised/hierarchicalclustering/>

<https://orangedatamining.com/widget-catalog/unsupervised/kmeans/>

<https://medium.com/deep-learning-turkiye/k-means-algoritmas%C4%B1-b460620dd02a>

<https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html#:~:text=The%20silhouette%20plot%20displays%20a,like%20number%20of%20clusters%20visually>.

<https://www.javatpoint.com/hierarchical-clustering-in-machine-learning>

<https://www.javatpoint.com/hierarchical-clustering-in-machine-learning>

<https://orange3.readthedocs.io/projects/orange-visual-programming/en/latest/widgets/model/knn.html>

<https://orange3.readthedocs.io/projects/orange-visual-programming/en/latest/widgets/model/neuralnetwork.html>