Course : Business Intelligence and Data Mining

Title : 2023 Fall Semester – Final Term

Team Members:

Benoit Kham 112015007

楊玉英 112578408

余珮綺 112598028

詹端安 111749407

STROKE RISK PREDICTION

TABLE OF CONTENTS

Data source:

https://www.kaggle.com/datasets/teamincribo/stroke-prediction?select=stroke_prediction_dataset.csv

Code: Stroke risk prediction.ipynb

1. Introduction of Data Source

The Stroke Prediction dataset is designed to predict the likelihood of a stroke occurrence in individuals. The dataset is available on Kaggle and consists of 15,000 records with 22 features as shown in Table 1.1. It aims to provide comprehensive information necessary for the development of predictive models related to stroke detection.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Patient ID	Patient Name	Age	Gender	Hyperten	Heart D	Marital St	Work Type	Residence	Average Glucose Level	Body Mass	Smoking Status	Alcohol Intake	Physical Activity	Stroke Hi	Family Hi	Dietary Habits	Stress Levels	Blood Pressu	Cholesterol Levels	Symptoms	Diagnosis
2	18153	Mamooty Khu	56	Male	0	1	Married	Self-employed	Rural	130,91	22,37	Non-smoker	Social Drinker	Moderate	0	Yes	Vegan	3,48	140/108	HDL: 68, LDL: 133	Difficulty Speaking, Headache	Stroke
3	62749	Kaira Subrami	80	Male	0	0	Single	Self-employed	Urban	183,73	32,57	Non-smoker	Never	Low	0	No	Paleo	1,73	146/91	HDL: 63, LDL: 70	Loss of Balance, Headache, Di	Stroke
4	32145	Dhanush Bala	26	Male	1	1	Married	Never Worked	Rural	189	20,32	Formerly Smoked	Rarely	High	0	Yes	Paleo	7,31	154/97	HDL: 59, LDL: 95	Seizures, Dizziness	Stroke
5	6154	Ivana Baral	73	Male	0	0	Married	Never Worked	Urban	185,29	27,5	Non-smoker	Frequent Drinker	Moderate	0	No	Paleo	5,35	174/81	HDL: 70, LDL: 137	Seizures, Blurred Vision, Sever	No Stroke
6	48973	Darshit Jayara	51	Male	1	1	Divorced	Self-employed	Urban	177,34	29,06	Currently Smokes	Rarely	Low	0	Yes	Pescatarian	6,84	121/95	HDL: 65, LDL: 68	Difficulty Speaking	Stroke
7	29307	Advika Kota	62	Female	0	0	Single	Private	Urban	91,6	37,47	Currently Smokes	Social Drinker	High	0	No	Gluten-Free	4,85	132/64	HDL: 80, LDL: 69	Severe Fatigue	Stroke
8	25525	Elakshi Karan	40	Female	1	0	Married	Private	Urban	77,83	28,2	Currently Smokes	Never	Low	1	No	Vegetarian	6,38	178/105	HDL: 31, LDL: 134		No Stroke
9	4809	Shalv Dugar	61	Female	0	1	Divorced	Government Job	Rural	194,73	26,44	Formerly Smoked	Rarely	Moderate	1	No	Vegan	5,85	179/72	HDL: 66, LDL: 134	Loss of Balance	Stroke
10	7372	Raghav Hands	72	Female	1	1	Married	Self-employed	Rural	72,99	30,1	Formerly Smoked	Rarely	High	1	No	Vegetarian	0,73	141/106	HDL: 71, LDL: 140	Loss of Balance, Dizziness, Diff	Stroke
11	37504	Krish Kulkarni	82	Male	0	0	Divorced	Self-employed	Urban	111,23	28,83	Currently Smokes	Rarely	High	1	Yes	Non-Vegetarian	8,39	119/88	HDL: 32, LDL: 88		No Stroke
12	15298	Neelofar Devi	41	Male	0	1	Divorced	Government Job	Urban	94,9	36,74	Formerly Smoked	Frequent Drinker	Low	1	Yes	Pescatarian	1,56	91/80	HDL: 32, LDL: 114	Difficulty Speaking, Numbness	No Stroke
13	36017	Anaya Koshy	72	Female	0	0	Divorced	Private	Urban	155,32	30,87	Currently Smokes	Frequent Drinker	Moderate	0	Yes	Paleo	8,71	127/110	HDL: 55, LDL: 96	Loss of Balance, Blurred Visior	Stroke
14	66924	Ahana Lalla	30	Female	0	1	Divorced	Government Job	Urban	163,15	19,36	Formerly Smoked	Frequent Drinker	Moderate	0	Yes	Non-Vegetarian	9,19	114/67	HDL: 80, LDL: 83	Loss of Balance, Numbness	Stroke
15	46821	Zaina Chaudh	80	Female	0	0	Single	Private	Urban	136,06	25,19	Formerly Smoked	Never	High	1	No	Gluten-Free	4,14	97/81	HDL: 40, LDL: 141	Loss of Balance	No Stroke
16	54426	Tara Swamina	42	Male	0	1	Married	Self-employed	Rural	181,02	19,35	Formerly Smoked	Frequent Drinker	Low	1	No	Paleo	2,58	170/102	HDL: 70, LDL: 70	Dizziness, Blurred Vision, Wea	Stroke
17	86093	Azad Krishnan	86	Male	0	1	Married	Government Job	Rural	130,71	31,83	Non-smoker	Social Drinker	High	0	No	Non-Vegetarian	3,77	151/97	HDL: 52, LDL: 128	Confusion, Difficulty Speaking,	Stroke
18	9062	Mehul Rangar	31	Female	1	1	Single	Private	Rural	64,91	16,9	Formerly Smoked	Frequent Drinker	High	0	No	Vegetarian	3,26	175/84	HDL: 30, LDL: 154	Numbness, Numbness, Blurre	No Stroke
19	29940	Nishith Bhatt	63	Female	1	1	Divorced	Self-employed	Rural	88,43	32,45	Currently Smokes	Rarely	High	1	No	Non-Vegetarian	8,29	103/86	HDL: 56, LDL: 82	Weakness, Weakness, Blurred	No Stroke
20	53292	Vritika Lala	40	Female	0	1	Single	Government Job	Urban	199,01	31,22	Non-smoker	Rarely	Moderate	1	Yes	Vegan	0,82	120/60	HDL: 58, LDL: 92	Confusion, Seizures	Stroke
21	23954	Taran Khatri	25	Male	0	0	Married	Private	Urban	71,38	39	Non-smoker	Rarely	Moderate	0	Yes	Gluten-Free	0,46	170/64	HDL: 72, LDL: 174	Seizures	Stroke
22	73140	Nitara Kapadi	40	Female	1	1	Divorced	Government Job	Rural	171,67	18,12	Formerly Smoked	Social Drinker	Low	1	No	Vegetarian	9,7	98/106	HDL: 41, LDL: 114	Severe Fatigue, Difficulty Spea	No Stroke
23	62785	Manjari Dhalr	33	Female	0	0	Divorced	Government Job	Urban	72,85	37,58	Formerly Smoked	Never	Low	0	No	Non-Vegetarian	5,65	119/105	HDL: 68, LDL: 75	Weakness, Seizures, Numbnes	No Stroke
24	44810	Saira Loyal	64	Female	1	1	Divorced	Never Worked	Rural	72,66	24,82	Currently Smokes	Frequent Drinker	Moderate	1	No	Non-Vegetarian	2,27	134/70	HDL: 42, LDL: 162	Loss of Balance, Seizures, Con	Stroke
25	35165	Emir Rajan	69	Male	0	1	Married	Private	Rural	149,46	27,2	Non-smoker	Frequent Drinker	Low	0	Yes	Vegan	8,4	163/94	HDL: 50, LDL: 69	Numbness, Seizures, Weaknes	No Stroke
26	79771	Nayantara Iss	80	Female	0	1	Divorced	Never Worked	Urban	154,25	15,42	Non-smoker	Never	High	0	Yes	Vegan	9,56	178/74	HDL: 54, LDL: 115	Dizziness	No Stroke
27	36975	Jhanvi Brar	24	Female	0	0	Married	Self-employed	Urban	79,89	17,58	Currently Smokes	Social Drinker	High	1	No	Vegetarian	6,48	151/65	HDL: 73, LDL: 111	Numbness, Loss of Balance, Ni	Stroke
28	81347	Gokul Bhakta	47	Female	0	1	Divorced	Government Job	Rural	83,76	20,14	Non-smoker	Never	Low	1	No	Gluten-Free	9,15	102/85	HDL: 30, LDL: 137	Numbness, Weakness	No Stroke
29	6743	Yasmin Vala	83	Male	0	1	Divorced	Never Worked	Urban	180,31	22,12	Non-smoker	Rarely	High	1	No	Vegan	7,21	151/64	HDL: 69, LDL: 109	Severe Fatigue, Weakness, Dif	Stroke
30	30237	Aayush Choud	68	Female	0	0	Single	Private	Rural	108,49	21,33	Non-smoker	Frequent Drinker	Moderate	0	Yes	Non-Vegetarian	5,35	119/110	HDL: 67, LDL: 185	Weakness, Dizziness, Headacl	Stroke
31	28610	Miraya Kaur	63	Male	0	1	Married	Never Worked	Urban	107,84	23,42	Formerly Smoked	Never	Low	1	No	Paleo	4,69	123/63	HDL: 67, LDL: 102		Stroke
32	39392	Aayush Bora	72	Female	0	0	Single	Never Worked	Rural	161,83	33,97	Formerly Smoked	Frequent Drinker	High	1	No	Non-Vegetarian	9,26	109/104	HDL: 53, LDL: 93	Loss of Balance, Confusion, Bl	No Stroke
33	8202	Nitya Garg	57	Male	0	1	Married	Government Job	Rural	76,3	29,44	Non-smoker	Rarely	Moderate	0	Yes	Vegan	2,85	101/69	HDL: 33, LDL: 143	Confusion, Severe Fatigue, Los	Stroke
34	28323	Zain Kari	71	Male	0	1	Divorced	Self-employed	Urban	162,15	39,43	Currently Smokes	Frequent Drinker	Moderate	0	No	Pescatarian	9,8	108/80	HDL: 46, LDL: 60		No Stroke
35	59056	Abram Samra	69	Female	0	1	Single	Never Worked	Rural	75,92	18,32	Currently Smokes	Rarely	High	1	Yes	Vegetarian	6,81	161/65	HDL: 40, LDL: 112	Loss of Balance	No Stroke
36	73489	Lavanya Karpa	62	Female	0	1	Divorced	Private	Rural	175	29,35	Currently Smokes	Social Drinker	High	0	No	Non-Vegetarian	6,74	146/103	HDL: 46, LDL: 94	Seizures, Seizures	Stroke
37	5940	Arnav Bansal	58	Female	0	0	Married	Never Worked	Urban	123,2	25,35	Non-smoker	Rarely	Moderate	1	No	Keto	9,38	90/109	HDL: 33, LDL: 174	Blurred Vision, Difficulty Speal	No Stroke
38	96568	Ahana Shenoy	80	Female	0	1	Married	Government Job	Urban	166,91	31,68	Non-smoker	Social Drinker	High	0	No	Vegetarian	8	137/109	HDL: 68, LDL: 101	Dizziness, Headache	Stroke

Table 1.1. Dataset of Stroke Prediction research

(Source: Kaggle, 2023)

The dataset appears to cover a wide range of factors that are suspected to influence the likelihood of a stroke, such as Gender, Marital Status, Work Type, Residence Type, Smoking Status, Alcohol Intake, Physical Activity, Family History, Dietary Habits, Symptoms. It provides a diverse set of variables that can be utilized for training and evaluating machine learning models.

2. Explanation of Data Variables

The details of variables would be described as Table 2.1.

Variables	Name	Definition	Types	Uses
X1	Patient ID	*Unique identifier for each patient*	1-15,000	Enables the tracking and differentiation of individual patients within the dataset
X2	Patient Name	*Name of the patient*	Names	Provides personal identification
X3	Age	*Age of the patient*	Numbers	Indicates the patient's age for demographic and age-related health assessments
X4	Gender	*Gender of the patient*	Male Female	Incorporates gender information for gender-specific health analyses and considerations
X5	Hypertension	*Presence of hypertension*	0: No 1: Yes	Flags whether the patient has a history of hypertension, a critical factor in cardiovascular health
X6	Heart Disease	*Presence of heart disease*	0: No 1: Yes	Indicates whether the patient has a history of heart disease, an essential cardiovascular health indicator
X7	Marital Status	*Marital status of the patient*	Single Married	Incorporates socio-demographic information for potential

			Divorced	correlations with health outcomes
X8	Work Type	*Type of work the patient is engaged in*	Private Government Never worked Self-employed	Considers occupational factors that may impact health and lifestyle.
X9	Residence Type	*Type of residence*	Urban Rural	Reflects the patient's living environment, which can influence health behaviors and outcomes
X10	Average Glucose Level (*)	*Average glucose level in the patient's blood.*	Numbers	A key indicator of blood sugar control and diabetes risk
X11	Body Mass Index (BMI) (**)	*Body Mass Index of the patient*	Numbers	Evaluates the patient's weight status, a crucial factor in overall health assessment
X12	Smoking Status	*Smoking status of the patient*	Non-smoker Formerly-smoked Currently Smokes	Captures information on tobacco use, a significant factor in respiratory and health
X13	Alcohol Intake	*Alcohol consumption status*	Social Drinker Never Rarely Frequent Drinker	Considers lifestyle factors related to alcohol consumption and potential impacts on health.

X14	Physical Activity	*Level of physical activity*	High Moderate Low	Assesses the patient's activity level, which is crucial for overall health and well-being
X15	Stroke History	*History of Stroke*	0: No 1: Yes	Indicates whether the patient has a history of stroke, a critical neurological event
X16	Family History	*Family History of Stroke*	Yes No	Considers genetic factors and family history related to stroke risk.
X17	Dietary Habits	*Dietary habits of the patient*	Vegan Paleo Pescatarian Gluten-Free Vegatarian Non-vegatarian	Captures information about the patient's dietary preferences, which can impact health outcomes
X18	Stress Levels (***)	*Stress levels of the patient.*	Numbers	Considers the patient's perceived stress levels, which can affect overall health.
X19	Blood Pressure Levels (****)	*Blood pressure levels of the patient*	Numbers	Essential for assessing health and risk factors.

X20	Cholesterol Levels (*****)	*Cholesterol levels of the patient*	Numbers	A key indicator of health
X21	Symptoms	*Symptoms reported by the patient*	Difficulty Speaking, Headache, Loss of Balance, Dizziness, Confusion, and others	Captures additional health-related information.
Y	Diagnosis	*The target variable indicating the diagnosis*	Stroke/No Stroke	

Table 2.1. Explanation of variables

(*) **Average Glucose Level**

The average glucose level refers to the average amount of glucose (sugar) in the blood over a specific period, providing an indication of how well blood sugar is controlled over time.

(**) **Body Mass Index (BMI)**

BMI is a measure of body fat based on height and weight.

$$\text{BMI} = \frac{\text{Weight (in kilograms)}}{\text{Height}^2 \text{ (in meters)}}$$

The World Health Organization (WHO) provides the following general BMI categories for adults:

- Underweight: BMI less than 18.5

- Normal weight: BMI between 18.5 and 24.9

- Overweight: BMI between 25 and 29.9

- Obesity: BMI 30 or greater

(***) Stress Level

Stress level refers to the perceived or experienced amount of mental or emotional pressure and strain an individual feels in response to various life events, situations, or demands.

(****) Blood Pressure Level

Blood pressure is a measure of the force of blood against the walls of the arteries as the heart pumps it around the body. Blood pressure is often written as systolic over diastolic, for example, 120/80 mmHg. The American Heart Association provides the following general categories for blood pressure levels in adults:

- Normal: Systolic less than 120 mmHg and diastolic less than 80 mmHg
- Elevated: Systolic 120-129 mmHg and diastolic less than 80 mmHg
- Hypertension Stage 1: Systolic 130-139 mmHg or diastolic 80-89 mmHg
- Hypertension Stage 2: Systolic 140 mmHg or higher or diastolic 90 mmHg or higher
- Hypertensive Crisis: Systolic higher than 180 mmHg and/or diastolic higher than 120 mmHg

(*****) Cholesterol Level

Cholesterol is a fatty substance that is essential for building cells and producing certain hormones. The two main types of cholesterol that are measured are Low-Density Lipoprotein (LDL) and High-Density Lipoprotein (HDL).

- Low-Density Lipoprotein (LDL) Cholesterol: Often referred to as "bad" cholesterol.

- High levels of LDL cholesterol are associated with an increased risk of atherosclerosis and heart disease.

- Normal levels are generally considered to be less than 100 mg/dL, but optimal levels may vary based on individual health factors.

- High-Density Lipoprotein (HDL) Cholesterol: Often referred to as "good" cholesterol.

- HDL helps remove LDL cholesterol from the bloodstream.

- Higher levels of HDL cholesterol are generally considered beneficial and associated with a lower risk of heart disease.

- Normal levels are typically above 40 mg/dL for men and above 50 mg/dL for women.

Elevated levels of total cholesterol (which includes both LDL and HDL cholesterol) and high levels of LDL cholesterol, in particular, are risk factors for strokes.

3. Explanation of Research Topic

3.1. The objectives of research

The aim of predicting strokes using 22 variables is to identify and understand the factors that contribute to the likelihood of a patient experiencing a stroke. By analyzing this dataset, we could gain the understanding of the following information.

- **Risk assessment:** Evaluate the risk of stroke for an individual based on their demographic information (age, gender), lifestyle factors (smoking, alcohol intake, physical activity), or medical history (hypertension, heart disease).

- **Early detection**: Identify patterns or combinations of factors that may indicate an increased risk of stroke, allowing for early intervention and preventive measures.

- **Treatment planning**: Provide insights into the potential impact of specific health indicators (average glucose level, BMI, blood pressure, cholesterol levels) on stroke risk, assisting in the development of personalized treatment plans.

- **Public health strategies**: Understand the prevalence of stroke risk factors in different populations, helping to develop targeted public health campaigns and interventions.

- **Patient education**: Educate individuals about modifiable risk factors (such as lifestyle choices) to empower them to make informed decisions and adopt healthier behaviors.

- **Resource allocation**: Assist healthcare providers in allocating resources more efficiently by focusing on high-risk populations and tailoring interventions based on individual characteristics.

- **Genetic and environmental factors**: Explore the impact of family history, dietary habits, stress levels, and residence type on stroke risk, considering both genetic and environmental influences.

- **Monitoring and follow-up:** Establish a system for regular monitoring and follow-up of individuals identified as high-risk, ensuring timely interventions and adjustments to their healthcare plans.

3.2. Research Methods

With the purpose of predicting the target variable Y (Diagnosis) using the variables X1 to X21 in this dataset, we follow these steps:

a. Data Preprocessing

Handle any missing or incomplete data.

b. Variable Selection Process

Identify important variables that contribute to the prediction of the diagnosis.

c. Choose a Cluster Model

Select a cluster model suitable by comparing the calculated center positions of various clusters under different clustering methods

d. Split the Data

Split the dataset into training and testing sets. This helps to train models on one subset of data and evaluate its performance on another.

e. Building a predict model

4. Data Cleaning Process

Let's take a look at our dataset:

	Patient ID	Patient Name	Age	Gender	Hypertension	Heart Disease	Marital Status	Work Type	Residence Type	Average Glucose Level	...	Alcohol Intake	Physical Activity	Stroke History	Family History of Stroke	Dietary Habits	Stress Levels	Blood Pressure Levels	Cholesterol Levels	Symptoms	Diagnosis
0	18153	Mamooty Khurana	56	Male	0	1	Married	Self-employed	Rural	130.91	...	Social Drinker	Moderate	0	Yes	Vegan	3.48	140/108	HDL: 68, LDL: 133	Difficulty Speaking, Headache	Stroke
1	62749	Kaira Subramaniam	80	Male	0	0	Single	Self-employed	Urban	183.73	...	Never	Low	0	No	Paleo	1.73	146/91	HDL: 63, LDL: 70	Loss of Balance, Headache, Dizziness, Confusion	Stroke
2	32145	Dhanush Balan	26	Male	1	1	Married	Never Worked	Rural	189.00	...	Rarely	High	0	Yes	Paleo	7.31	154/97	HDL: 59, LDL: 95	Seizures, Dizziness	Stroke
3	6154	Ivana Baral	73	Male	0	0	Married	Never Worked	Urban	185.29	...	Frequent Drinker	Moderate	0	No	Paleo	5.35	174/81	HDL: 70, LDL: 137	Seizures, Blurred Vision, Severe Fatigue, Head...	No Stroke
4	48973	Darshit Jayaraman	51	Male	1	1	Divorced	Self-employed	Urban	177.34	...	Rarely	Low	0	Yes	Pescatarian	6.84	121/95	HDL: 65, LDL: 68	Difficulty Speaking	Stroke

5 rows × 22 columns

We can observe that our dataset has 22 columns and multiple types of data, such as numeric and nominal.

As it is, it will be challenging to use it so we need to clean it.

First, we can get rid of the columns *Patient ID* and *Patient Name*.

Then, let's take a look at the last columns: *Blood Pressure Levels, Cholesterol Levels* and *Symptoms*.

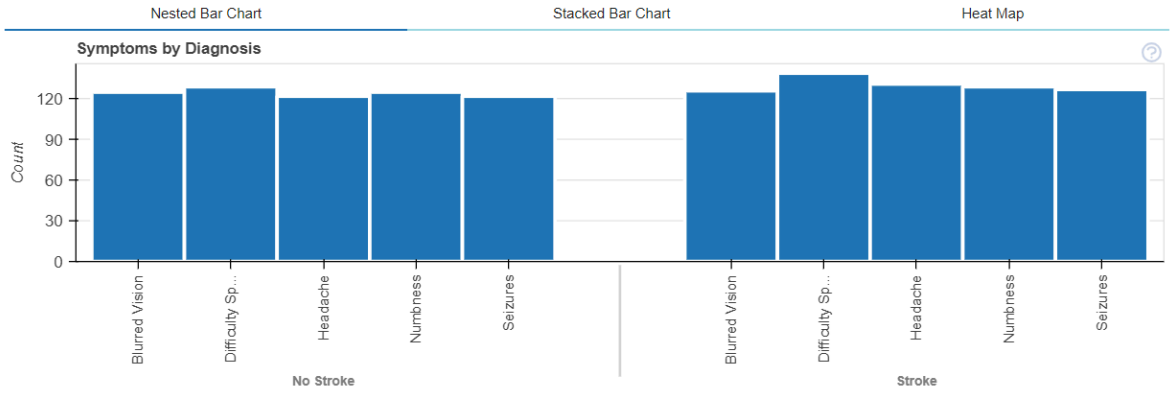
For *Blood Pressure Levels*, we need to make the division so that we get numerical values.

For *Cholesterol Levels*, we separated this column into two: *HDL* and *LDL*, then we dropped this one.

Lastly, for *Symptoms*, this column is quite unique because it traces all symptoms the patient had and sometimes, those symptoms are appearing twice with the same patient. We now have

two choices, either we try to separate this column as we did with the *Cholesterol Levels* column, or we drop it.

Of course, we can't drop it just like that so let's plot this column over diagnosis to know whether or not it seems to be relevant to keep.



As we can see in this chart, there is no striking difference between the symptoms of people diagnosed without stroke and people diagnosed with stroke.

Considering how this column could pollute our dataset if we separate it and how relevant this column seems to be, we decided to drop it.

Finally, to get a usable dataset, we transform all the nominal values using the pandas.categorical.codes function which will encode all nominal values into numerical ones in alphabetical order.

For example, in the *Gender* column, Female will be replaced by 0 and Male by 1 or in the *Diagnosis* column, No stroke becomes 0 and Stroke becomes 1. Here are all the nominal values that got transformed.

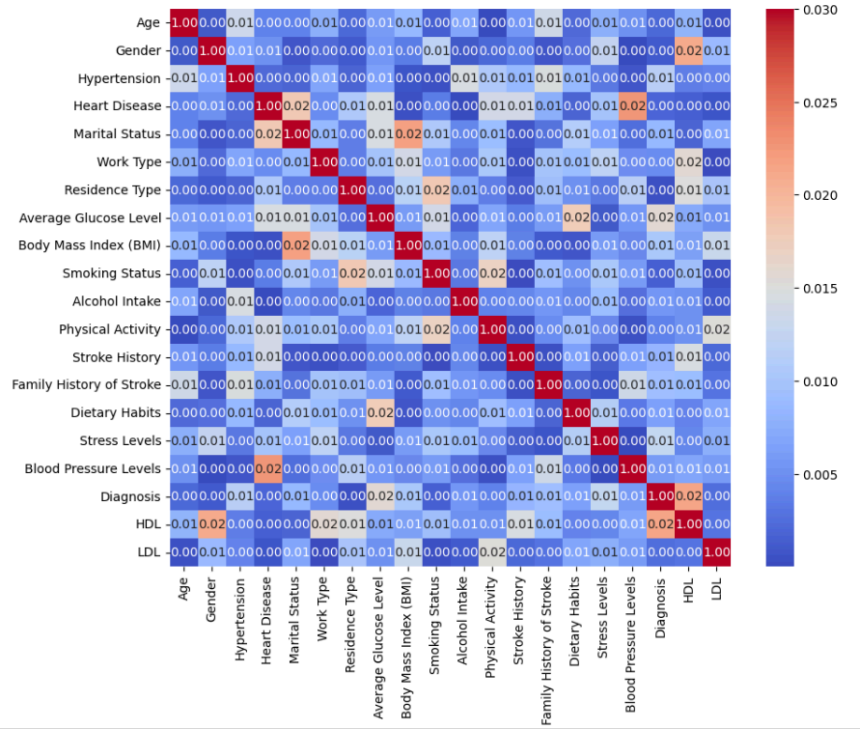
```
Index(['Female', 'Male'], dtype='object')
Index(['Divorced', 'Married', 'Single'], dtype='object')
Index(['Government Job', 'Never Worked', 'Private', 'Self-employed'], dtype='object')
Index(['Rural', 'Urban'], dtype='object')
Index(['Currently Smokes', 'Formerly Smoked', 'Non-smoker'], dtype='object')
Index(['Frequent Drinker', 'Never', 'Rarely', 'Social Drinker'], dtype='object')
Index(['High', 'Low', 'Moderate'], dtype='object')
Index(['No', 'Yes'], dtype='object')
Index(['Gluten-Free', 'Keto', 'Non-Vegetarian', 'Paleo', 'Pescatarian',
       'Vegan', 'Vegetarian'],
      dtype='object')
Index(['No Stroke', 'Stroke'], dtype='object')
```

This is what our dataset looks like now that it has been cleaned:

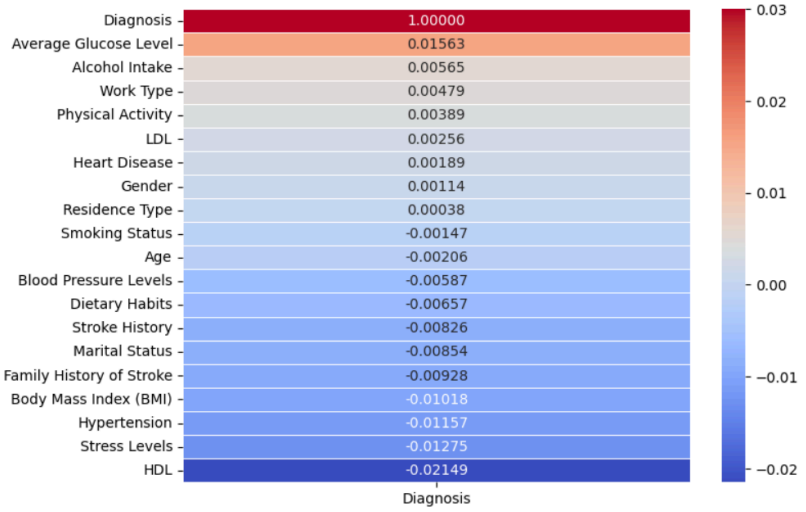
	Age	Gender	Hypertension	Heart Disease	Marital Status	Work Type	Residence Type	Average Glucose Level	Body Mass Index (BMI)	Smoking Status	Alcohol Intake	Physical Activity	Stroke History	Family History of Stroke	Dietary Habits	Stress Levels	Blood Pressure Levels	Diagnosis	HDL	LDL
0	56	1	0	1	1	3	0	130.91	22.37	2	3	2	0	1	5	3.48	1.30	1	68	133
1	80	1	0	0	2	3	1	183.73	32.57	2	1	1	0	0	3	1.73	1.60	1	63	70
2	26	1	1	1	1	1	0	189.00	20.32	1	2	0	0	1	3	7.31	1.59	1	59	95
3	73	1	0	0	1	1	1	185.29	27.50	2	0	2	0	0	3	5.35	2.15	0	70	137
4	51	1	1	1	0	3	1	177.34	29.06	0	2	1	0	1	4	6.84	1.27	1	65	68

5. Variable Selection Process

Now that we have a clean dataset, let's decide which features we want to keep.
In order to do that, we will create the correlation matrix of our dataset:



There is a lot of information in this matrix so let's take a look at the values that interests us, the ones over *Diagnosis*:



We can see that the correlation values are very low but we will keep the ones with the highest values which are: *Average Glucose Level, HDL, LDL, Stress Levels, Hypertension* and *Body Mass Index (BMI)*. (We selected LDL because it was at first in the same column as HDL)

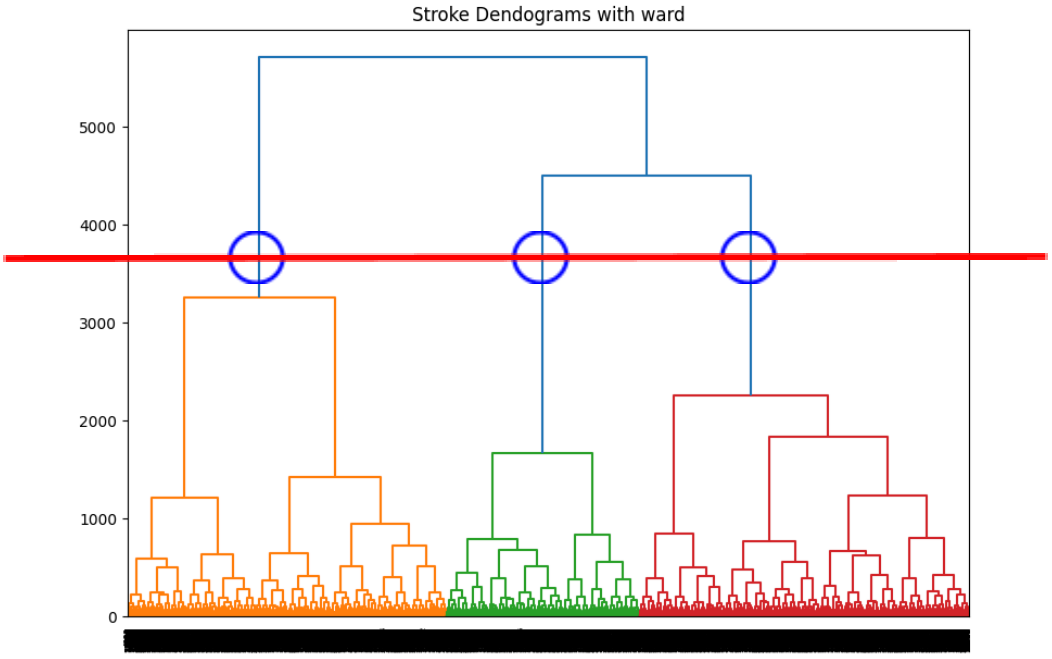
6. Cluster

Based on the clustering methods we learned in class, we will use the Hierarchical and Non-hierarichical methods to find the most appropriate clustering method and the number of clusters.

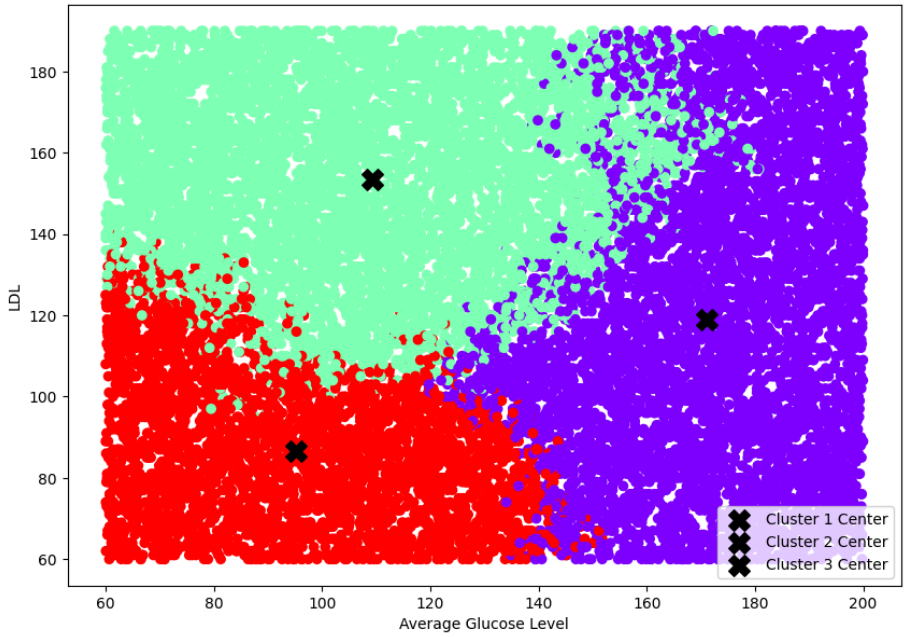
Hierarchical

Ward

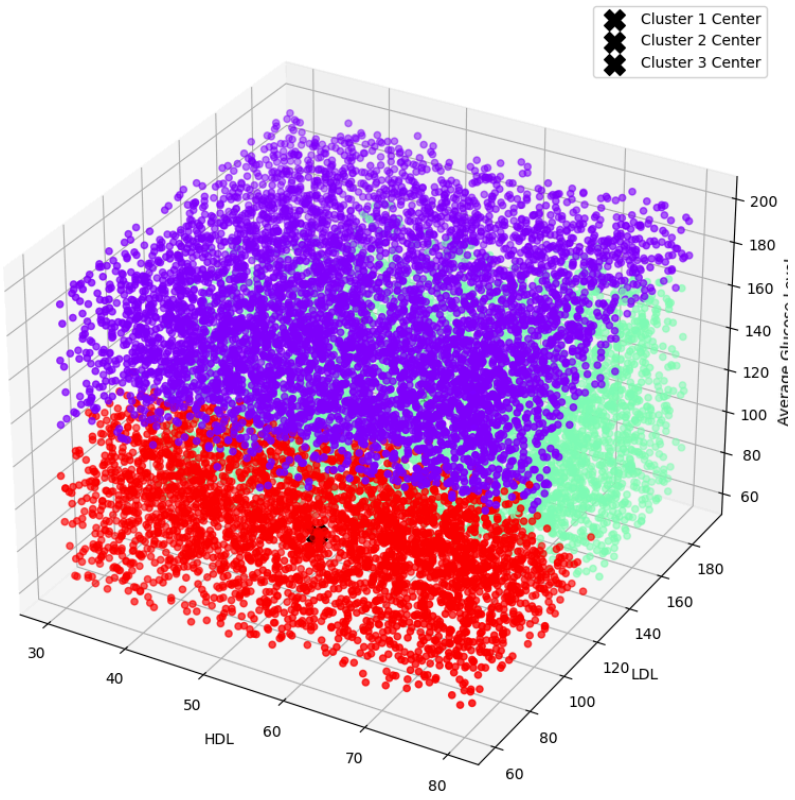
First, after reviewing Ward's dendrogram, we decided to divide the number of clusters into 3 clusters based on the following chart



The following picture shows the 2D clustering results and the location of the center of the cluster. (Average Glucose Level and LDL)



The following picture shows the 3D clustering results and the location of the center of the cluster (Average Glucose Level, HDL and LDL).



The following picture shows the number of data for each cluster.

```
Cluster 2: 5865 samples
Cluster 1: 5691 samples
Cluster 3: 3444 samples
```

The distance from each data point to the center of the cluster averaged over all values (this will be used as an average judgment of the concentration within the cluster).

```
Average Distance for Cluster 1: 40.75922445358474
Average Distance for Cluster 2: 37.37520984571152
Average Distance for Cluster 3: 32.006508340210026
AVG of total distance:36.713647546502095
```

Average distance between clusters (this will be used as an average judgment of the degree of dispersion between clusters)

```
Average Inter-Cluster Distance: 73.8875988780487
```

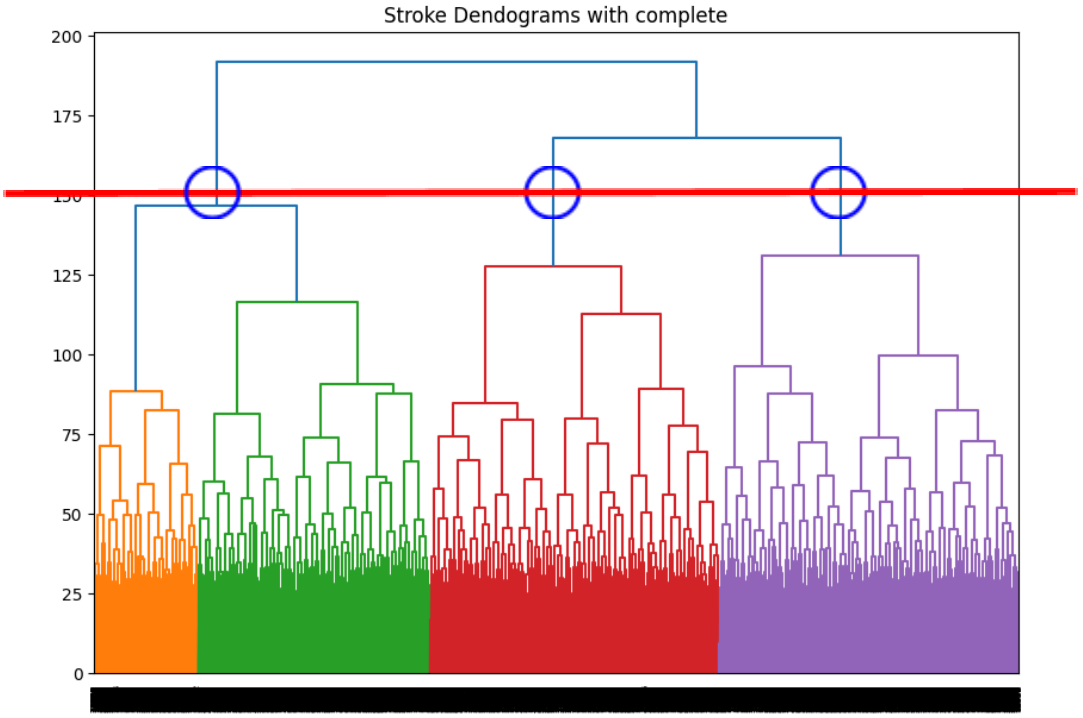
And the average status of the data across clusters (which will be used as a judgment of whether the differences across clusters are sufficiently significant)

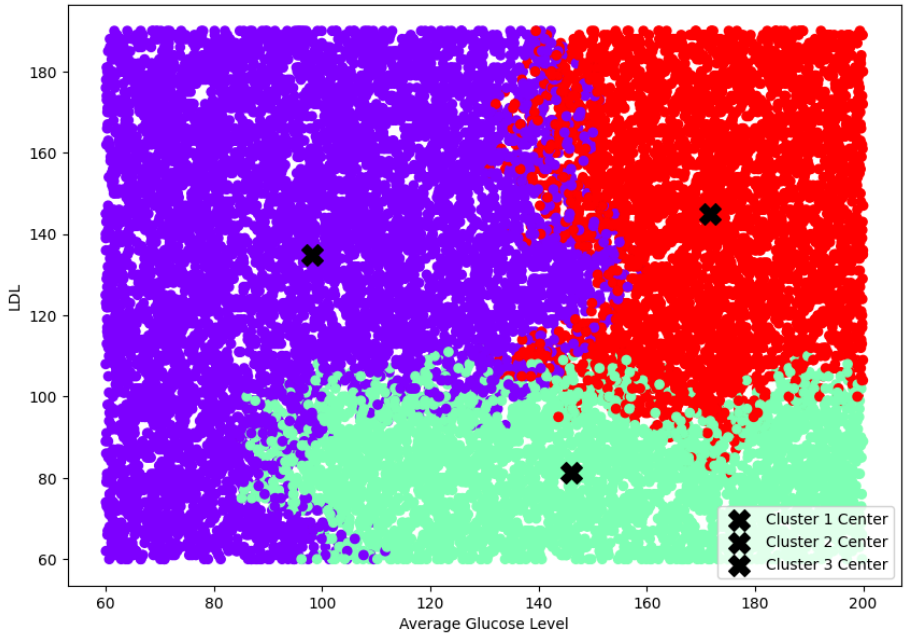
Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	171.118296	54.324899	119.047970	5.043889	0.241082	27.453270
1	109.189321	55.264621	153.404604	5.028010	0.253026	27.553507
2	95.077973	55.947735	86.699768	4.978618	0.255226	27.374172

Complete Linkage

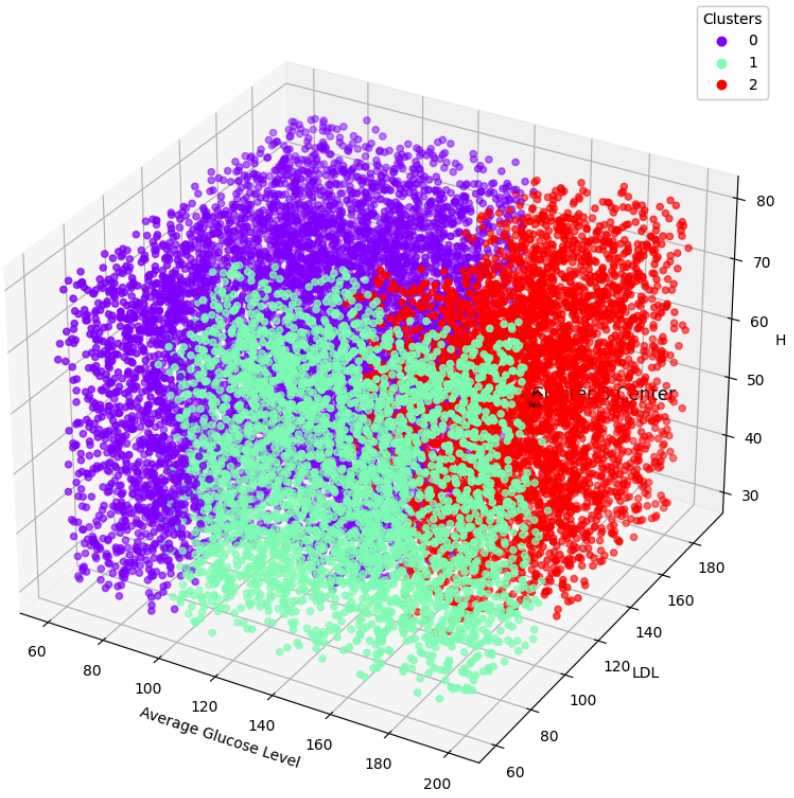
First, after reviewing Complete Linkage's dendrogram, we decided to divide the number of clusters into 3 clusters based on the following chart

The following picture shows the 2D clustering results and the location of the center of the cluster (Average Glucose Level and LDL).





The following picture shows the 3D clustering results and the location of the center of the cluster (Average Glucose Level, HDL and LDL).



The following picture shows the number of data for each cluster.

```
Cluster 1: 7397 samples
Cluster 2: 3505 samples
Cluster 3: 4098 samples
```

The distance from each data point to the center of the cluster averaged over all values (this will be used as an average judgment of the concentration within the cluster).

```
Average Distance for Cluster 1: 42.98218168244123
Average Distance for Cluster 2: 35.19781927331319
Average Distance for Cluster 3: 34.007669489101886
AVG of total distance:37.39589014828544
```

Average distance between clusters (this will be used as an average judgment of the degree of dispersion between clusters)

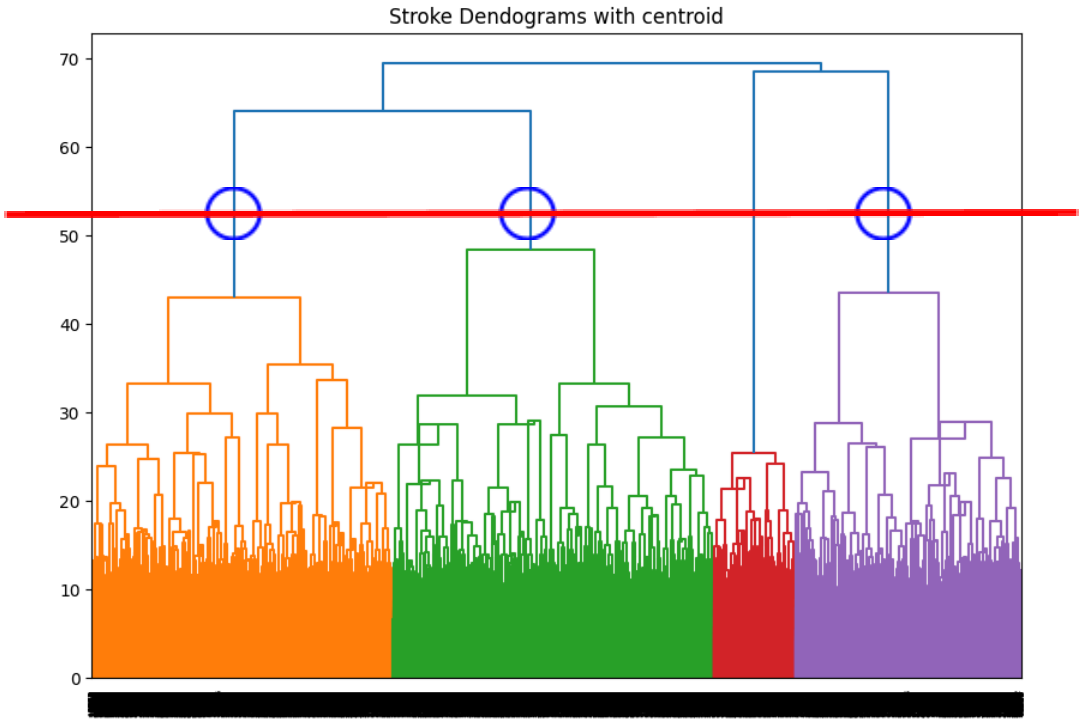
```
Average Inter-Cluster Distance: 71.50101706820841
```

And the average status of the data across clusters (which will be used as a judgment of whether the differences across clusters are sufficiently significant)

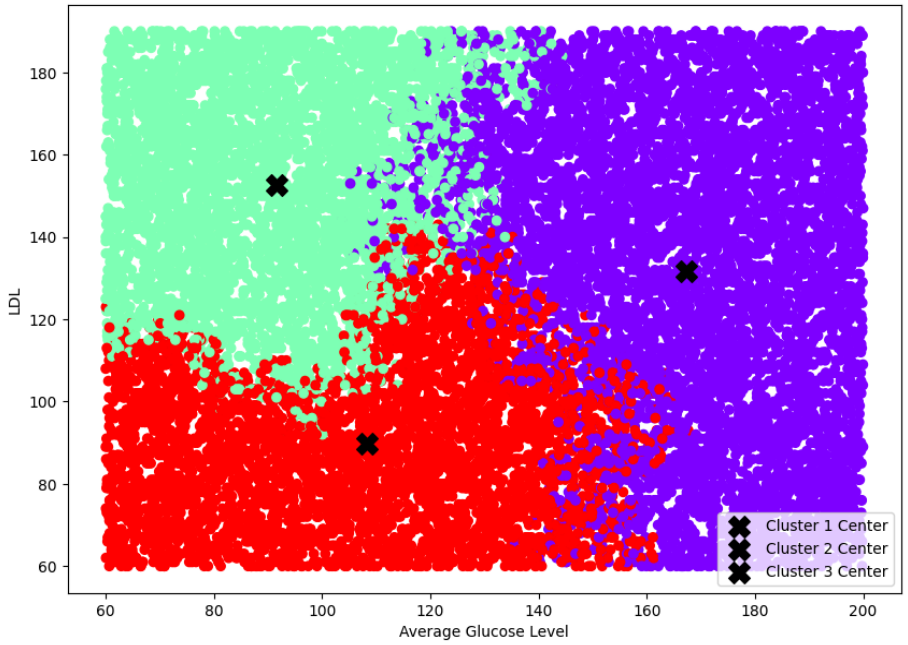
Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	98.183462	55.243072	134.764905	5.017081	0.250913	27.555401
1	145.985330	54.870471	81.352068	5.016245	0.247361	27.124565
2	171.726813	54.909712	144.904588	5.038341	0.246950	27.627045

Centroid

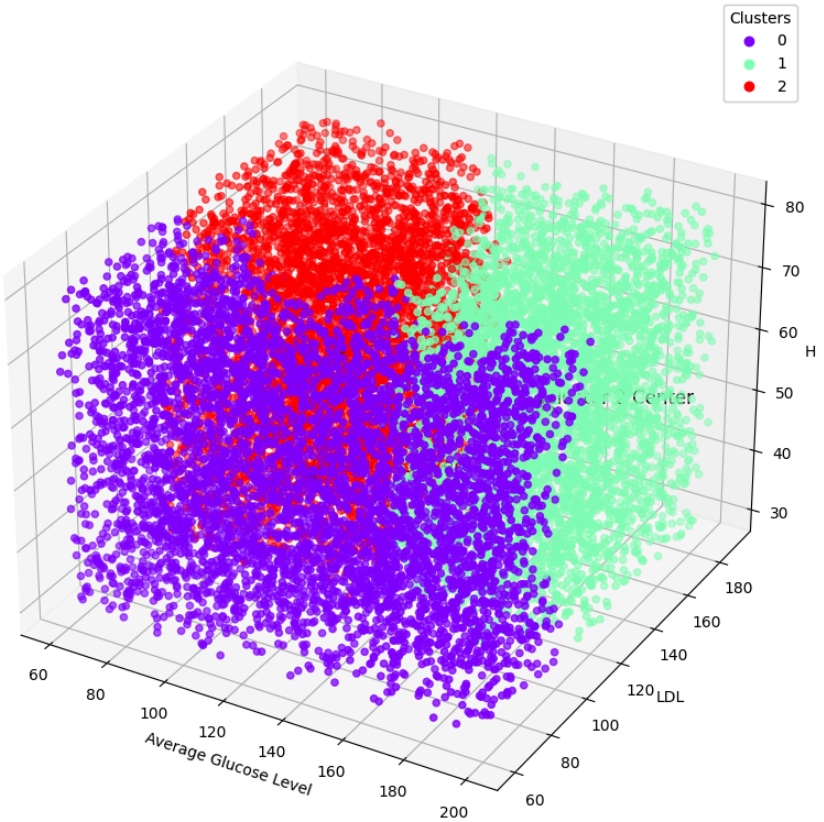
First, after reviewing Centroid's dendrogram, we decided to divide the number of clusters into 3 clusters based on the following chart



The following picture shows the 2D clustering results and the location of the center of the cluster (Average Glucose Level and LDL).



The following picture shows the 3D clustering results and the location of the center of the cluster (Average Glucose Level, HDL and LDL).



The following picture shows the number of data for each cluster.

```
Cluster 3: 5112 samples
Cluster 1: 5417 samples
Cluster 2: 4471 samples
```

The distance from each data point to the center of the cluster averaged over all values (this will be used as an average judgment of the concentration within the cluster).

```
Average Distance for Cluster 1: 43.33657490428665
Average Distance for Cluster 2: 33.868691793429015
Average Distance for Cluster 3: 35.35522016800246
AVG of total distance:37.52016228857271
```

Average distance between clusters (this will be used as an average judgment of the degree of dispersion between clusters)

```
Average Inter-Cluster Distance: 71.81586552908892
```

And the average status of the data across clusters (which will be used as a judgment of whether the differences across clusters are sufficiently significant)

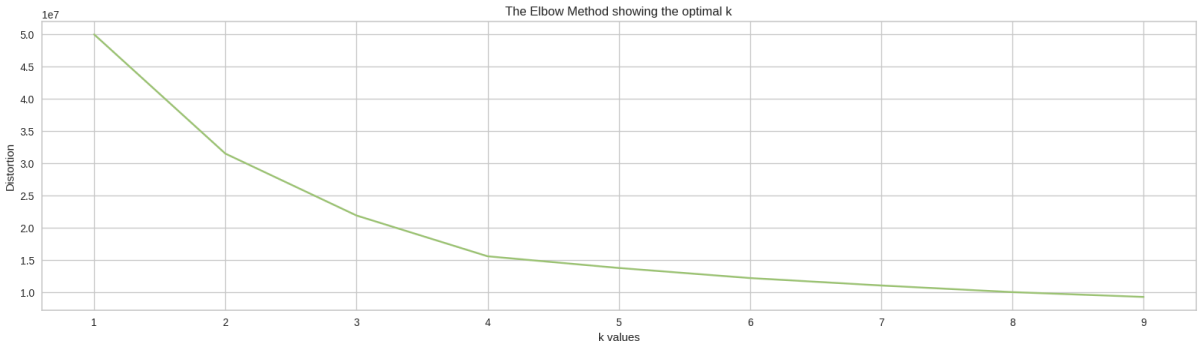
Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	128.654453	55.290013	83.991877	5.002073	0.249400	27.208252
1	166.823158	54.848356	148.780362	5.054437	0.248714	27.610027
2	97.592062	55.015845	147.815532	5.016782	0.248826	27.637520

Non-hierarichical

K-means

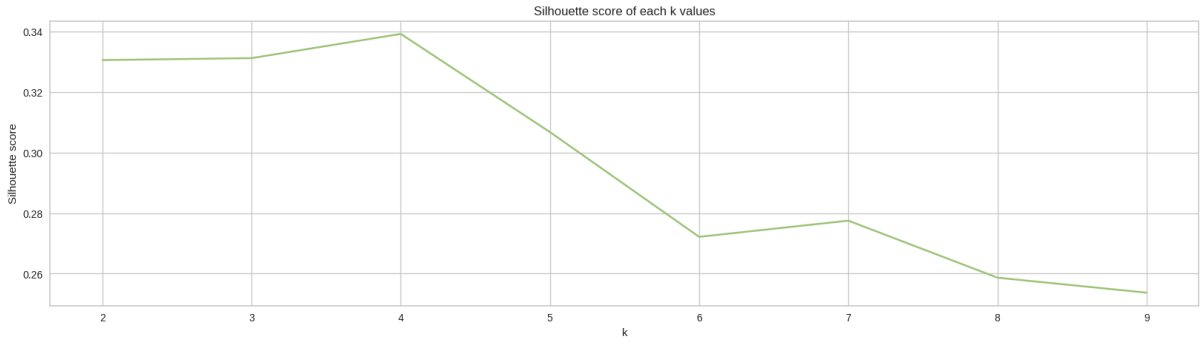
To determine the optimal number of clusters (K) in a K-means clustering analysis,

First, we check the elbow diagram of K-means. It helps in finding the point where the rate of improvement (decrease in within-cluster variance) sharply changes, resembling an "elbow."

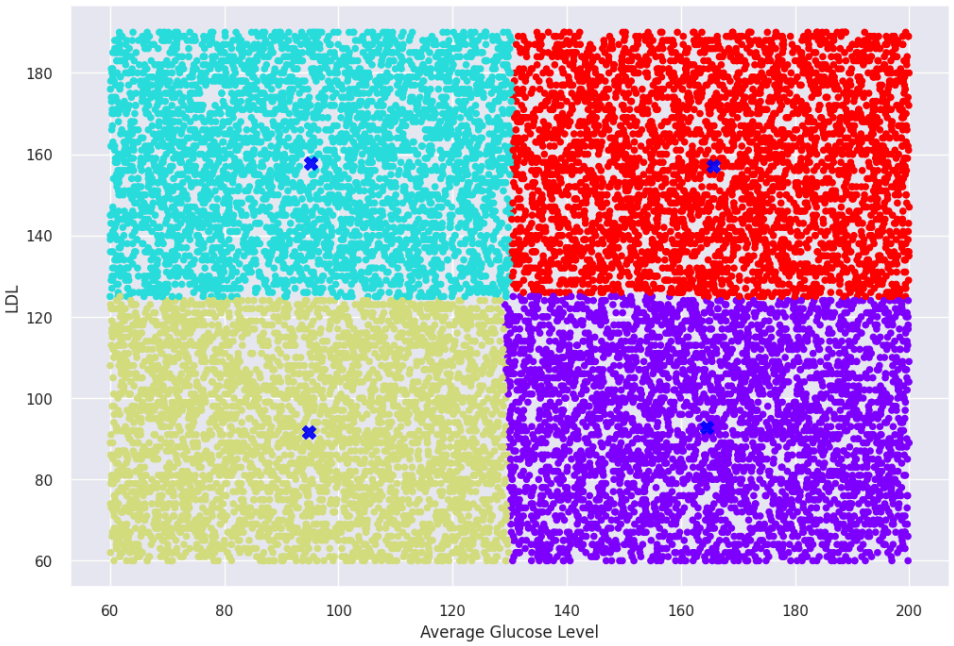


Then, we check the Silhouette score graph. The silhouette score measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette score ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

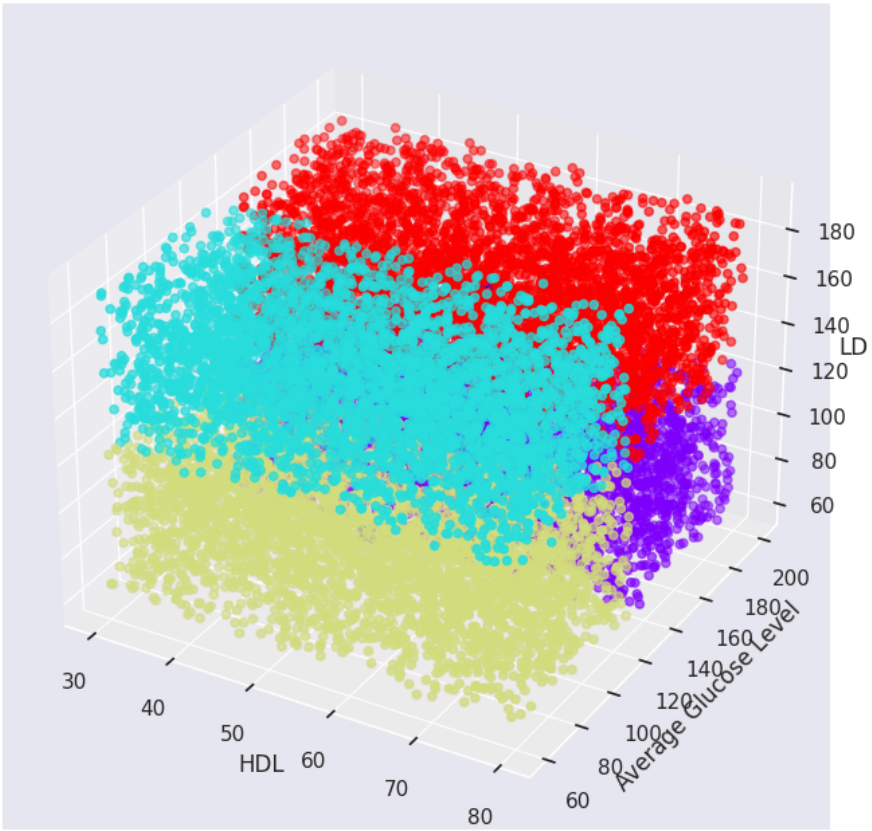
In the silhouette score graph, we look for the highest silhouette score, as it suggests the optimal number of clusters.



The following picture shows the 2D clustering results and the location of the center of the cluster.



The following picture shows the 3D clustering results and the location of the center of the cluster.



The following picture shows the number of data for each cluster.

```
Cluster 4: 3676 samples
Cluster 1: 3703 samples
Cluster 3: 3744 samples
Cluster 2: 3877 samples
```

The distance from each data point to the center of the cluster averaged over all values (this will be used as an average judgment of the concentration within the cluster).

```
Average Distance for Cluster 1: 30.962791519469658
Average Distance for Cluster 2: 31.302064215848933
Average Distance for Cluster 3: 31.156968849864445
Average Distance for Cluster 4: 30.966362447154108
AVG of total distance:31.097046758084286
```

Average distance between clusters (this will be used as an average judgment of the degree of dispersion between clusters)

```
Average Inter-Cluster Distance: 77.03856685857
```

And the average status of the data across clusters (which will be used as a judgment of whether the differences across clusters are sufficiently significant)

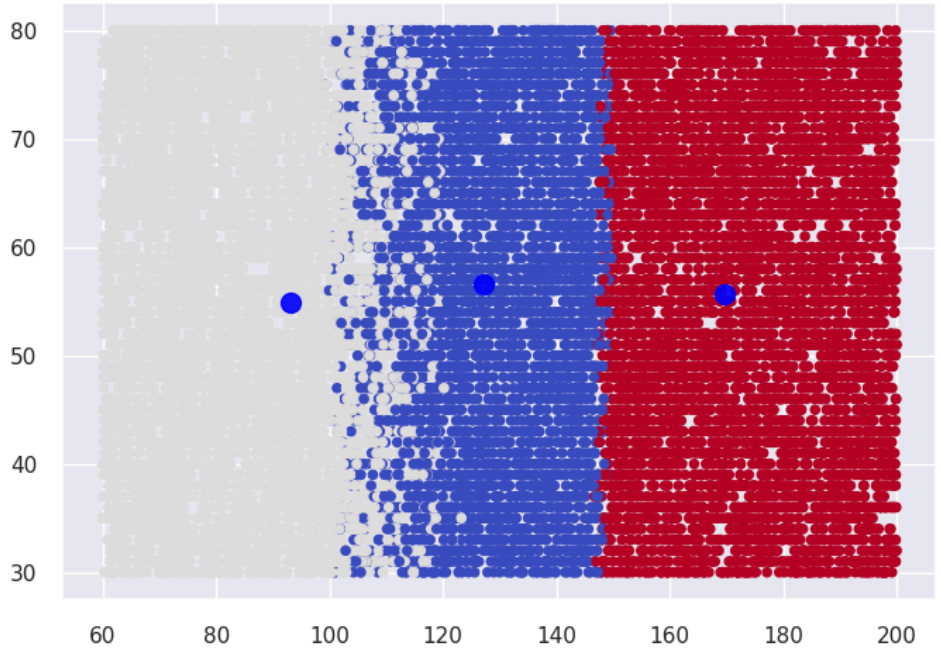
Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	164.467707	55.115312	92.850122	5.008844	0.243316	27.340945
1	95.143260	55.046428	157.739489	5.037908	0.247098	27.579123
2	94.678745	54.808494	91.654380	4.989228	0.255342	27.384479
3	165.752603	55.294886	157.040261	5.054684	0.250272	27.589570

Self-Organizing Map, SOM

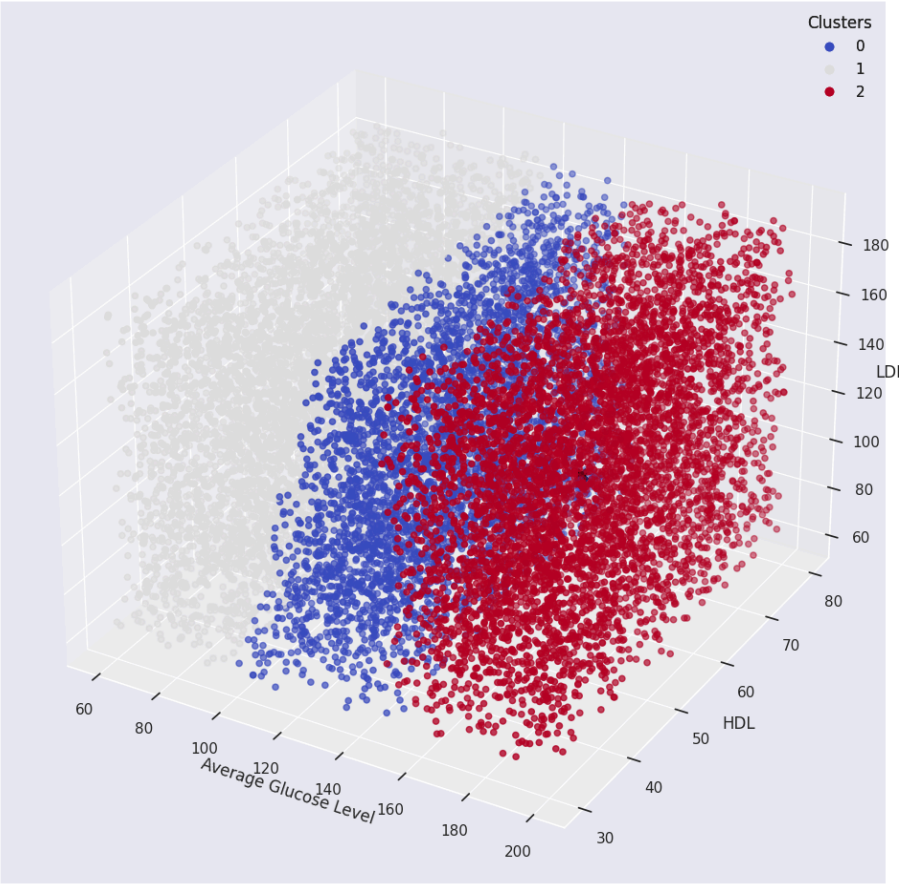
The SOM model is first trained with the parameters shown below

```
s = som.SOM(neurons=(1,3), dimentions=6, n_iter=500, learning_rate=0.2)
s.train(samples)
print("SOM Cluster centres:", s.weights_)
print("SOM labels:", s.labels_)
result_SOM = s.predict(samples)
```

The following picture shows the 2D clustering results and the location of the center of the cluster.



The following picture shows the 3D clustering results and the location of the center of the cluster.



The following picture shows the number of data for each cluster.

```
Cluster 1: 4005 samples
Cluster 3: 5489 samples
Cluster 2: 5506 samples
```

The distance from each data point to the center of the cluster averaged over all values (this will be used as an average judgment of the concentration within the cluster)

```
Average Distance to Center 0: 55.17339773750041
Average Distance to Center 1: 63.01354875044978
Average Distance to Center 2: 64.96604788618527
Total Average Distance to Centers: 61.05099812471182
```

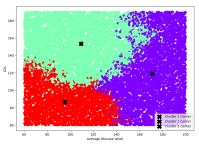
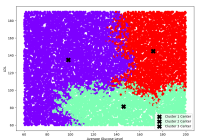
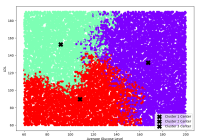
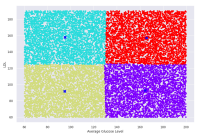
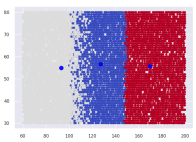
Average distance between clusters (this will be used as an average judgment of the degree of dispersion between clusters)

```
Average Inter-Cluster Distance: 51.349130099128274
```

And the average status of the data across clusters (which will be used as a judgment of whether the differences across clusters are sufficiently significant)

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	128.539615	55.48814	116.514357	5.006342	0.250437	27.569164
1	85.669797	54.72866	130.276062	5.020053	0.250999	27.463440
2	174.016956	55.09346	126.047367	5.037275	0.245946	27.415983

In order to facilitate the selection of the best cluster method, the following is a comparison of all the clustering methods, the meaning of the columns from the top to the bottom are: the number of data points of each cluster, the average value of the distance of each data point of each cluster from the center of the cluster and the combined average value, and the average value of the distance between the clusters.

Ward	Complete Linkage	Centroid	K-means	SOM
				
Cluster 1: 5691 Cluster 2: 5865 Cluster 3: 3444	Cluster 1: 7397 Cluster 2: 3505 Cluster 3: 4098	Cluster 1: 5417 Cluster 2: 4471 Cluster 3: 5112	Cluster 1: 3703 Cluster 2: 3877 Cluster 3: 3744 Cluster 4: 3676	Cluster 1: 4005 Cluster 2: 5506 Cluster 3: 5489
Avg dis C1: 40.759 Avg dis C2: 37.375 Avg dis C3: 32.006 total avg: 36.714	Avg dis C1: 42.982 Avg dis C2: 35.198 Avg dis C3: 34.007 total avg: 37.396	Avg dis C1: 43.348 Avg dis C2: 33.893 Avg dis C3: 35.780 total avg: 37.540	Avg dis C1: 30.967 Avg dis C2: 31.153 Avg dis C3: 30.967 Avg dis C4: 31.302 total avg:31.097	Avg dis C1: 55.173 Avg dis C2: 63.013 Avg dis C3: 64.966 total avg: 61.051
Average Inter-Cluster Distance: 73.887	Average Inter-Cluster Distance: 71.501	Average Inter-Cluster Distance: 71.816	Average Inter-Cluster Distance: 77.038	Average Inter-Cluster Distance: 51.349

And the average of the values of each cluster after each clustering method.

Ward:

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	92.635343	56.195970	121.977804	5.006958	0.254235	27.452198
1	160.478920	54.762030	150.687699	5.040640	0.244336	27.596113
2	160.154176	52.885094	83.844539	5.026930	0.245109	27.296706

Complete Linkage:

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	169.552329	54.749921	122.136320	5.043443	0.245313	27.417078
1	100.434993	54.489590	156.630798	5.040835	0.246341	27.611684
2	100.468384	56.303243	90.083528	4.965990	0.258366	27.394853

Centroid:

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	173.648519	56.491117	126.312886	5.051769	0.244997	27.493314
1	100.434993	54.489590	156.630798	5.040835	0.246341	27.611684
2	109.531310	54.058101	91.754061	4.971993	0.256143	27.314348

K-means:

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	164.467707	55.115312	92.850122	5.008844	0.243316	27.340945
1	95.143260	55.046428	157.739489	5.037908	0.247098	27.579123
2	94.678745	54.808494	91.654380	4.989228	0.255342	27.384479
3	165.752603	55.294886	157.040261	5.054684	0.250272	27.589570

SOM:

Cluster	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	Body Mass Index (BMI)
0	128.539615	55.48814	116.514357	5.006342	0.250437	27.569164
1	85.669797	54.72866	130.276062	5.020053	0.250999	27.463440
2	174.016956	55.09346	126.047367	5.037275	0.245946	27.415983

It is known that to find the optimal cluster method, the following conditions are required: whether the data points are close enough to the center of the cluster, whether the clusters are

sufficiently dispersed, and whether the average value of the clusters is sufficiently different from each other.

Therefore, based on the above conditions, we choose K-means for the subsequent classification model prediction.

Below is the meaning of each Cluster based on their average values:

Cluster 0 (3703 samples):

- Average Glucose Level: 164.47
- HDL (High-Density Lipoprotein): 55.12
- LDL (Low-Density Lipoprotein): 92.85
- Stress Levels: 5.01
- Hypertension: 0.24
- Body Mass Index (BMI): 27.34

This cluster is characterized by moderate to high values in Average Glucose Level, LDL, and Stress Levels. The prevalence of hypertension is relatively low, and the BMI falls within the overweight range.

Health Indices:

- High: Average Glucose Level.
- Moderate: HDL, Stress Levels, Hypertension, BMI.
- Low: LDL.

Cluster 1 (3877 samples):

- Average Glucose Level: 95.14
- HDL: 55.05
- LDL: 157.74
- Stress Levels: 5.04
- Hypertension: 0.25
- BMI: 27.58

Cluster 1 exhibits lower Average Glucose Levels, normal HDL, but elevated LDL. Stress levels are moderate, and the prevalence of hypertension is slightly higher than Cluster 0. The BMI is indicative of being overweight.

Health Indices:

- High: LDL
- Moderate: HDL, Stress Levels, Hypertension, BMI.
- Low: Average Glucose Level.

Cluster 2 (3744 samples):

- Average Glucose Level: 94.68
- HDL: 54.81
- LDL: 91.65
- Stress Levels: 4.99
- Hypertension: 0.26
- BMI: 27.38

This cluster has relatively low Average Glucose Levels and LDL, normal HDL, and lower stress levels. The prevalence of hypertension is slightly higher, and the BMI falls within the overweight range.

Health Indices:

- High: None.
- Moderate: HDL, Stress Levels, Hypertension, BMI.
- Low: Average Glucose Level, LDL.

Cluster 3 (3676 samples):

- Average Glucose Level: 165.75
- HDL: 55.29
- LDL: 157.04
- Stress Levels: 5.05
- Hypertension: 0.25
- BMI: 27.59

Similar to Cluster 1, Cluster 3 has higher Average Glucose Levels and LDL, normal HDL, and moderate stress levels. The prevalence of hypertension is slightly higher, and the BMI indicates overweight.

Health Indices:

- High: Average Glucose Level, LDL.
- Moderate: HDL, Stress Levels, Hypertension, BMI.

- Low: None.

In summary, each cluster represents a group of individuals with distinct patterns in key health indicators, such as glucose levels, lipid profiles, stress levels, hypertension prevalence, and BMI. These clusters provide valuable insights into the diversity of health profiles within the dataset, enabling a more targeted and personalized approach to healthcare interventions.

7. Data Splitting

Therefore, merge the Diagnosis column that was dropped before the cluster, and the following picture shows the result of the merged data.

	Average Glucose Level	HDL	LDL	Stress Levels	Hypertension	\
0	130.91	68	133	3.48	0	
1	183.73	63	70	1.73	0	
2	189.00	59	95	7.31	1	
3	185.29	70	137	5.35	0	
4	177.34	65	68	6.84	1	

	Body Mass Index (BMI)	Diagnosis
0	22.37	1
1	32.57	1
2	20.32	1
3	27.50	0
4	29.06	1

Then the data of each group will be separated into df1, df2, df3, df4, so that the training data and testing data can be divided for each cluster in the following step.

```
df1 = split_used_df[split_used_df['Cluster'] == 0]
df2 = split_used_df[split_used_df['Cluster'] == 1]
df3 = split_used_df[split_used_df['Cluster'] == 2]
df4 = split_used_df[split_used_df['Cluster'] == 3]
```

The following diagrams show the code and the output of the data segmentation to check whether the segmentation is successful (we splitted into 80% training and 20% testing).

```
#1
X1 = df1.drop("Diagnosis", axis=1)
y1 = df1["Diagnosis"]
X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.2, random_state=42)

# 2
X2 = df2.drop("Diagnosis", axis=1)
y2 = df2["Diagnosis"]
X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.2, random_state=42)


# 3
X3 = df3.drop("Diagnosis", axis=1)
y3 = df3["Diagnosis"]
X3_train, X3_test, y3_train, y3_test = train_test_split(X3, y3, test_size=0.2, random_state=42)

# 4
X4 = df4.drop("Diagnosis", axis=1)
y4 = df4["Diagnosis"]
X4_train, X4_test, y4_train, y4_test = train_test_split(X4, y4, test_size=0.2, random_state=42)

print(X1_train.head())
print("==============")
print(y1_train.head())
```

```
       Average Glucose Level  HDL  LDL  Stress Levels  Hypertension  \
12223                 131.56   38   73           1.23             0
1899                  177.77   79   63           8.88             0
20                    171.67   41  114           9.70             1
8395                  141.36   37   74           3.96             0
11487                 155.52   65  109           6.04             0

       Body Mass Index (BMI)
12223                  19.54
1899                   25.32
20                     18.12
8395                   27.08
11487                  28.62
==============
12223    1
1899     0
20       0
8395     1
11487    0
Name: Diagnosis, dtype: int8
```

Then we can proceed to Building a Predict Model.

8. Building a Predict Model

According to 4 groups of testing - training data we got from 4 Clusters. Our group decided to try 3 methods to build a predict model for Stroke Prediction.

We did the same thing for the first step of 3 methods of prediction:

- First, we flattened the target variable and counted the occurrences of each class of "Diagnosis" using Counter. It provides an initial overview of the class distribution in the unbalanced training data.
- Then, we applied SMOTE to the training data to oversample the minority class, making the class distribution more balanced.
- *After running several times, we discovered that with different random_state, we'll get different ratios of accuracy. So for the Decision Tree and Neural Network method, we use a code to run the random_state randomly each time, this way can get a better result.*

The next steps of each method was proceed as below:

a. Discriminant:

- Initialized an instance of the Linear Discriminant Analysis (LDA) classifier, fit the LDA model to the oversampled training data and generated predictions for the test set using the trained LDA model.
- Show the confusion matrix and calculate the accuracy.

Cluster 1

```
[174] temp_discriminant_1 = y1_train.values.flatten()
      print(Counter(temp_discriminant_1))
      ### smote : balance training data
      from imblearn.over_sampling import SMOTE
      random_state = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state = random_state)
      print(f"random_state used for this run: {random_state}")

      X1_train_SM, y1_train_SM = sm.fit_resample(X1_train, y1_train)
      print(Counter(y1_train_SM))
      clf1 = LinearDiscriminantAnalysis()
      clf1.fit(X1_train_SM, y1_train_SM)

      #Result of y prediction
      y1_predicted = clf1.predict(X1_test)

      ##Confusion matrix
      from sklearn.metrics import confusion_matrix
      confusion_matrix = confusion_matrix(y1_test, y1_predicted)
      print(confusion_matrix)

      accuracy_discriminant_1 = accuracy_score(y1_test, y1_predicted)
      print(f"Accuracy: {accuracy_discriminant_1}")

      Counter({1: 1496, 0: 1444})
      random_state used for this run: 2091479674
      Counter({1: 1496, 0: 1496})
      [[170 190]
       [192 184]]
      Accuracy: 0.48097826086956524
```

Cluster 2

```
[176] temp_discriminant_2 = y2_train.values.flatten()
      print(Counter(temp_discriminant_2))
      ### smote : balance training data
      from imblearn.over_sampling import SMOTE
      random_state = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state = random_state)
      print(f"random_state used for this run: {random_state}")

      X2_train_SM, y2_train_SM = sm.fit_resample(X2_train, y2_train)
      print(Counter(y2_train_SM))
      clf2 = LinearDiscriminantAnalysis()
      clf2.fit(X2_train_SM, y2_train_SM)

      #Result of y prediction
      y2_predicted = clf2.predict(X2_test)

      ##Confusion matrix
      from sklearn.metrics import confusion_matrix
      confusion_matrix = confusion_matrix(y2_test, y2_predicted)
      print(confusion_matrix)

      accuracy_discriminant_2 = accuracy_score(y2_test, y2_predicted)
      print(f"Accuracy: {accuracy_discriminant_2}")

      Counter({0: 1562, 1: 1432})
      random_state used for this run: 1239884809
      Counter({0: 1562, 1: 1562})
      [[217 176]
       [180 176]]
      Accuracy: 0.5246995994659546
```

Cluster 3

```
[180] temp_discriminant_3 = y3_train.values.flatten()
      print(Counter(temp_discriminant_3))
      ### smote : balance training data
      from imblearn.over_sampling import SMOTE
      random_state = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state = random_state)
      print(f"random_state used for this run: {random_state}")

      X3_train_SM, y3_train_SM = sm.fit_resample(X3_train, y3_train)
      print(Counter(y3_train_SM))
      clf3 = LinearDiscriminantAnalysis()
      clf3.fit(X3_train_SM, y3_train_SM)

      #Result of y prediction
      y3_predicted = clf3.predict(X3_test)

      ##Confusion matrix
      from sklearn.metrics import confusion_matrix
      confusion_matrix = confusion_matrix(y3_test, y3_predicted)
      print(confusion_matrix)

      accuracy_discriminant_3 = accuracy_score(y3_test, y3_predicted)
      print(f"Accuracy: {accuracy_discriminant_3}")

      Counter({1: 1512, 0: 1451})
      random_state used for this run: 9299199
      Counter({1: 1512, 0: 1512})
      [[185 187]
       [174 195]]
      Accuracy: 0.5128205128205128
```

Cluster 4

```
[181] temp_discriminant_4 = y4_train.values.flatten()
      print(Counter(temp_discriminant_4))
      ### smote : balance training data
      from imblearn.over_sampling import SMOTE
      random_state = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state = random_state)
      print(f"random_state used for this run: {random_state}")

      X4_train_SM, y4_train_SM = sm.fit_resample(X4_train, y4_train)
      print(Counter(y4_train_SM))
      clf4 = LinearDiscriminantAnalysis()
      clf4.fit(X4_train_SM, y4_train_SM)

      #Result of y prediction
      y4_predicted = clf4.predict(X4_test)

      ##Confusion matrix
      from sklearn.metrics import confusion_matrix
      confusion_matrix = confusion_matrix(y4_test, y4_predicted)
      print(confusion_matrix)

      accuracy_discriminant_4 = accuracy_score(y4_test, y4_predicted)
      print(f"Accuracy: {accuracy_discriminant_4}")

      Counter({0: 1564, 1: 1537})
      random_state used for this run: 1221880741
      Counter({0: 1564, 1: 1564})
      [[188 198]
       [203 187]]
      Accuracy: 0.4832474226804124
```

b. Decision Tree:

- Generated a random state for reproducibility.
- Initialized Decision Tree classifier with a specified maximum depth and random state, fit the Decision Tree model to the oversampled training data and generated predictions for the test set using the trained Decision Tree model.

- Show the confusion matrix and calculate the accuracy.

Cluster 1

```
[214] temp_DecisionTree_1=y1_train.values.flatten()
      print(Counter(temp_DecisionTree_1))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X1_train_SM, y1_train_SM = sm.fit_resample(X1_train, y1_train)
      print(Counter(y1_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a CART template
      clf1= DecisionTreeClassifier(max_depth =5, random_state = random_state)
      # Train the model on the training data
      clf1.fit(X1_train_SM, y1_train_SM)
      y1_predicted = clf1.predict(X1_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y1_test, y1_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_DecisionTree_1 = accuracy_score(y1_test, y1_predicted)
      print("Accuracy:", accuracy_DecisionTree_1)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({1: 1496, 0: 1444})
random_state_smote used for this run: 2146404317
Counter({1: 1496, 0: 1496})
[[ 94 266]
 [ 81 295]]
Accuracy: 0.5285326086956522
random_state used for this run: 3878323306
```

Cluster 2

```
[215] temp_DecisionTree_2=y2_train.values.flatten()
      print(Counter(temp_DecisionTree_2))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X2_train_SM, y2_train_SM = sm.fit_resample(X2_train, y2_train)
      print(Counter(y2_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a CART template
      clf2= DecisionTreeClassifier(max_depth =5, random_state = random_state)
      # Train the model on the training data
      clf2.fit(X2_train_SM, y2_train_SM)
      y2_predicted = clf2.predict(X2_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y2_test, y2_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_DecisionTree_2 = accuracy_score(y2_test, y2_predicted)
      print("Accuracy:", accuracy_DecisionTree_2)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({0: 1562, 1: 1432})
random_state_smote used for this run: 1258806308
Counter({0: 1562, 1: 1562})
[[175 218]
 [153 203]]
Accuracy: 0.5046728971962616
random_state used for this run: 3038621215
```

Cluster 3

```
[216] temp_DecisionTree_3=y3_train.values.flatten()
      print(Counter(temp_DecisionTree_3))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X3_train_SM, y3_train_SM = sm.fit_resample(X3_train, y3_train)
      print(Counter(y3_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a CART template
      clf3= DecisionTreeClassifier(max_depth =5, random_state = random_state)
      # Train the model on the training data
      clf3.fit(X3_train_SM, y3_train_SM)
      y3_predicted = clf3.predict(X3_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y3_test, y3_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_DecisionTree_3 = accuracy_score(y3_test, y3_predicted)
      print("Accuracy:", accuracy_DecisionTree_3)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({1: 1512, 0: 1451})
random_state_smote used for this run: 397182490
Counter({1: 1512, 0: 1512})
[[215 157]
 [202 167]]
Accuracy: 0.5155195681511471
random_state used for this run: 2320120879
```

Cluster 4

```
[218] temp_DecisionTree_4=y4_train.values.flatten()
      print(Counter(temp_DecisionTree_4))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X4_train_SM, y4_train_SM = sm.fit_resample(X4_train, y4_train)
      print(Counter(y4_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a CART template
      clf4= DecisionTreeClassifier(max_depth =5, random_state = random_state)
      # Train the model on the training data
      clf4.fit(X4_train_SM, y4_train_SM)
      y4_predicted = clf4.predict(X4_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y4_test, y4_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_DecisionTree_4 = accuracy_score(y4_test, y4_predicted)
      print("Accuracy:", accuracy_DecisionTree_4)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({0: 1564, 1: 1537})
random_state_smote used for this run: 3322916266
Counter({0: 1564, 1: 1564})
[[129 257]
 [100 290]]
Accuracy: 0.5399484536082474
random_state used for this run: 2046463791
```

c. Neural Network:

- Generated a random state for reproducibility.
- Initialized an MLP (Neural Network) classifier with a specified alpha value (L2 penalty term), maximum number of iterations, and random state, fit the MLP model to

the oversampled training data and generated predictions for the test set using the trained MLP model.

- Show the confusion matrix and calculate the accuracy.

Cluster 1

```
[228] temp_neural_network_1=y1_train.values.flatten()
      print(Counter(temp_neural_network_1))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X1_train_SM, y1_train_SM = sm.fit_resample(X1_train, y1_train)
      print(Counter(y1_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a MLP template
      clf1= MLPClassifier(alpha=0.5, max_iter=1000, random_state=random_state)
      # Train the model on the training data
      clf1.fit(X1_train_SM, y1_train_SM)
      y1_predicted = clf1.predict(X1_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y1_test, y1_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_neural_network_1 = accuracy_score(y1_test, y1_predicted)
      print("Accuracy:", accuracy_neural_network_1)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({1: 1496, 0: 1444})
random_state_smote used for this run: 599961219
Counter({1: 1496, 0: 1496})
[[129 231]
 [127 249]]
Accuracy: 0.5135869565217391
random_state used for this run: 4203679593
```

Cluster 2

```
[237] temp_neural_network_2=y2_train.values.flatten()
      print(Counter(temp_neural_network_2))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X2_train_SM, y2_train_SM = sm.fit_resample(X2_train, y2_train)
      print(Counter(y2_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a MLP template
      clf2= MLPClassifier(alpha=0.5, max_iter=1000, random_state=random_state)
      # Train the model on the training data
      clf2.fit(X2_train_SM, y2_train_SM)
      y2_predicted = clf2.predict(X2_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y2_test, y2_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_neural_network_2 = accuracy_score(y2_test, y2_predicted)
      print("Accuracy:", accuracy_neural_network_2)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({0: 1562, 1: 1432})
random_state_smote used for this run: 1656325048
Counter({0: 1562, 1: 1562})
[[389   4]
 [343  13]]
Accuracy: 0.5367156208277704
random_state used for this run: 3284806749
```

Cluster 3

```
[253] temp_neural_network_3=y3_train.values.flatten()
      print(Counter(temp_neural_network_2))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X3_train_SM, y3_train_SM = sm.fit_resample(X3_train, y3_train)
      print(Counter(y2_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a MLP template
      clf3= MLPClassifier(alpha=0.5, max_iter=1000, random_state=random_state)
      # Train the model on the training data
      clf3.fit(X3_train_SM, y3_train_SM)
      y3_predicted = clf3.predict(X3_test)

      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y3_test, y3_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_neural_network_3 = accuracy_score(y3_test, y3_predicted)
      print("Accuracy:", accuracy_neural_network_3)
      print(f"random_state used for this run: {random_state}")
```

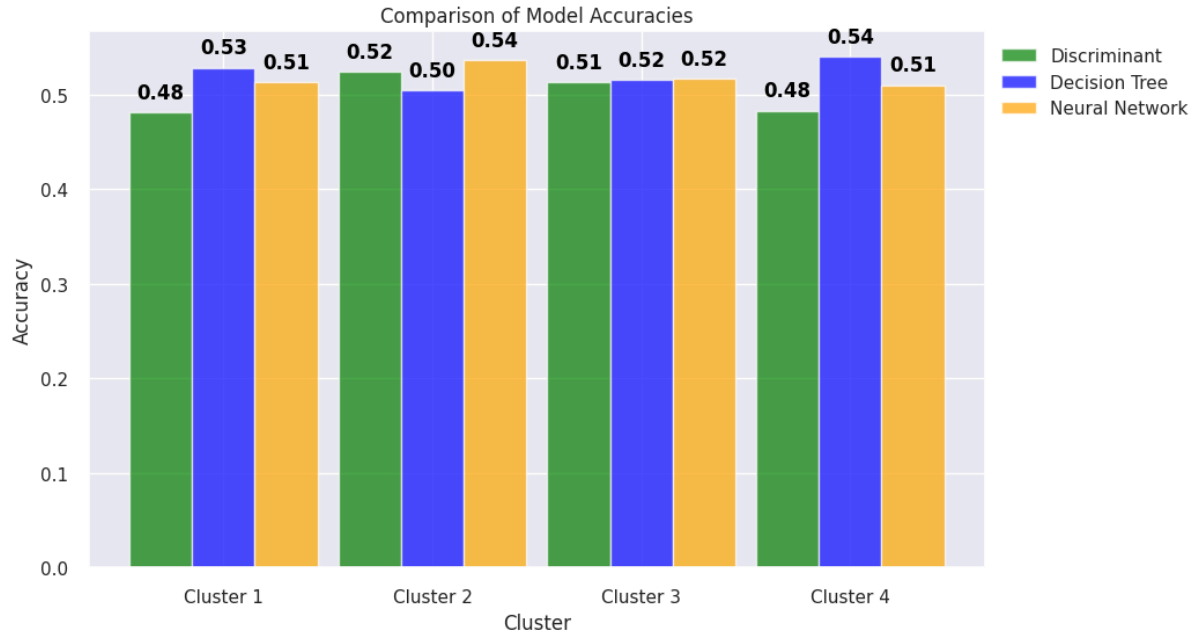
```
Counter({0: 1562, 1: 1432})
random_state_smote used for this run: 32814661
Counter({0: 1562, 1: 1562})
[[333  39]
 [319  50]]
Accuracy: 0.5168690958164642
random_state used for this run: 4059090695
```

Cluster 4

```
[256] temp_neural_network_4=y4_train.values.flatten()
      print(Counter(temp_neural_network_2))
      from imblearn.over_sampling import SMOTE
      random_state_smote = np.random.randint(0, 4294967295)
      sm = SMOTE(random_state=random_state_smote)
      print(f"random_state_smote used for this run: {random_state_smote}")

      X4_train_SM, y4_train_SM = sm.fit_resample(X4_train, y4_train)
      print(Counter(y4_train_SM))

      random_state = np.random.randint(0, 4294967295)
      # Create a MLP template
      clf4= MLPClassifier(alpha=0.5, max_iter=1000, random_state=random_state)
      # Train the model on the training data
      clf4.fit(X4_train_SM, y4_train_SM)
      y4_predicted = clf4.predict(X4_test)
      ##Confusion matrix
      from sklearn.metrics import accuracy_score, confusion_matrix
      confusion_matrix = confusion_matrix(y4_test, y4_predicted)
      print(confusion_matrix)

      # Calculate the accuracy of the model
      accuracy_neural_network_4 = accuracy_score(y4_test, y4_predicted)
      print("Accuracy:", accuracy_neural_network_4)
      print(f"random_state used for this run: {random_state}")
```

```
Counter({0: 1562, 1: 1432})
random_state_smote used for this run: 1365002274
Counter({0: 1564, 1: 1564})
[[225 161]
 [219 171]]
Accuracy: 0.5103092783505154
random_state used for this run: 1188473773
```

To visualize the comparison of the result of 3 methods, we draw a chart below:




```
mean accuracy discriminant: 0.5004364489591112
mean accuracy decision_tree: 0.5221683819128271
mean accuracy neural_network: 0.5193702378791223
```

Upon comparing the results visually through a chart, it was evident that all three methods yielded accuracies in the range of 48% to 54%. Notably, the Decision Tree method demonstrated a slightly superior performance, achieving accuracies of 53%, 50%, 52% and 54% across the four clusters.

Decision Tree model also because it helps us figure out which health factors are most important for predicting strokes. The model gives a score to each factor, showing its importance. This is helpful for us to focus on the key things that influence whether someone might have a stroke. These features match well with our goal of predicting strokes based on various health indicators.

Consequently, based on these reasons, **the group decided to adopt the Decision Tree method for further analysis and predictions.**

9. Summary

Our Stroke Prediction project aimed to develop a predictive model utilizing a Kaggle dataset with 22 features and 15,000 records. Key phases included data preprocessing, variable

selection, clustering, and application of three prediction models: Linear Discriminant Analysis (LDA), Decision Tree, and Neural Network.

Key Findings and Insights:

Variable Importance: Essential features influencing stroke prediction, such as Average Glucose Level, HDL, LDL, Stress Levels, Hypertension, and BMI, were identified through thorough analysis. *(But in our analysis, these features have very very small differences between people with stroke and without stroke, only Average Glucose Level, HDL, LDL are slightly higher).*

Cluster Analysis: K-means clustering provided distinct health indicator-based clusters, offering insights into diverse health profiles within the dataset.

	Cluster 0	Cluster 1	Cluster 2	Cluster 3
Average Glucose Level	High	Low	Low	High
LDL	Low	High	Low	High

```
Mean diagnosis_c1: 0.5092491838955386
1    1872
0    1804
=====
Mean diagnosis_c2: 0.4776916911568261
0    1955
1    1788
=====
Mean diagnosis_c3: 0.5078293736501079
1    1881
0    1823
=====
Mean diagnosis_c4: 0.49703378901212275
0    1950
1    1927
```

Arrange the stroke rate of each cluster from high to low:

- Cluster 1: ~50.9%
- Cluster 3: ~50.7%
- Cluster 4: ~49.7%
- Cluster 2: ~.47.8%

Model Performance: The Decision Tree did a bit better, getting around 50-54% accuracy. However, it's not quite up to the ideal standard, showing there's room to make it better.

Conclusion:

Even though we learned valuable things from the project, the model's accuracy is just a bit above 50%. It needs some improvements. We can make it better by getting more detailed data and looking at other things. The clustering part helped us understand health patterns, but we need stronger info.

Future Improvements:

- Since the accuracy is not great right now, we may need to get more and better data to fix that.
- Look into genes, the environment, and more health details to make the model stronger.
- Experiment with different ways of doing things, change settings, and see how it affects the model.
- Make the model work better for certain groups or types of people.

This study gives us a good start for predicting strokes. But, we need to keep making it better. We should look more at the details, try different things, and make sure it fits with what doctors really do. The goal is to make it more accurate and useful for all sorts of people.

10. Appendix

Check the mean of each feature in people with stroke (Diagnosis = 1) and without stroke (Diagnosis = 0)

```
[265] Diagnosis_features_means = df_corr.groupby('Diagnosis').mean()
      print(Diagnosis_features_means)
```

```
                    Age    Gender  Hypertension  Heart Disease  Marital Status  \
Diagnosis
0             54.078864  0.507568      0.253983       0.501992        1.018720
1             53.992100  0.508704      0.243974       0.503883        1.004687

              Work Type  Residence Type  Average Glucose Level  \
Diagnosis
0              1.518853        0.497876             128.815228
1              1.529593        0.498259             130.080588

              Body Mass Index (BMI)  Smoking Status  Alcohol Intake  \
Diagnosis
0                         27.547621        1.001062        1.500797
1                         27.400355        0.998661        1.513390

              Physical Activity  Stroke History  Family History of Stroke  \
Diagnosis
0                      0.992698        0.504381                  0.510754
1                      0.999063        0.496117                  0.501473

              Dietary Habits  Stress Levels  Blood Pressure Levels        HDL  \
Diagnosis
0                   3.037573       5.059169               1.643554  55.382634
1                   3.011382       4.985907               1.638404  54.744510

                     LDL
Diagnosis
0             124.958444
1             125.150911
```