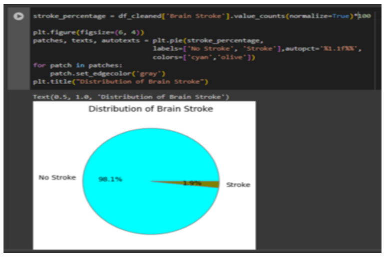
1 Organise Data

Go to the Previous section

2 Data Visualization

1. How is the distribution of stroke observed across the dataset?



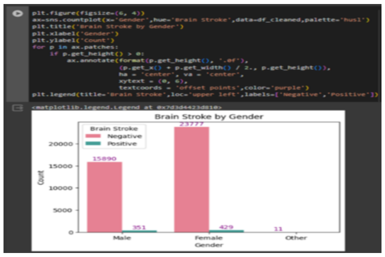
```
stroke_percentage = df_cleaned['Brain Stroke'].value_counts(normalize=True)*100

plt.figure(figsize=(6, 4))
patches, texts, autotexts = plt.pie(stroke_percentage,
                                    labels=['No Stroke', 'Stroke'],autopct='%1.1f%%',
                                    colors=['cyan','olive'])
for patch in patches:
    patch.set_edgecolor('gray')
plt.title("Distribution of Brain Stroke")
```

```
Text(0.5, 1.0, 'Distribution of Brain Stroke')
```

Figure 1: Pie Chart showing Distribution

In our dataset, stroke occurrence is highly prevalent, with 98.1% of records indicating stroke, while only 1.9% show no stroke. This significant imbalance necessitates careful consideration in predictive modelling to ensure accurate and reliable analysis.

2. How does the occurrence of stroke vary between genders within the dataset?



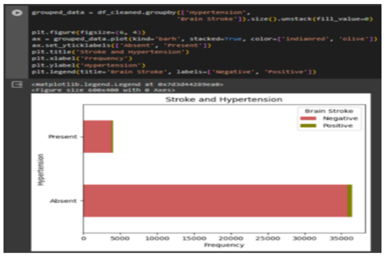
```
plt.figure(figsize=(6, 4))
ax=sns.countplot(x='Gender',hue='Brain Stroke',data=df_cleaned,palette='husl')
plt.title('Brain Stroke by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
for p in ax.patches:
    if p.get_height() > 0:
        ax.annotate(format(p.get_height(), '.0f'),
                    (p.get_x() + p.get_width() / 2., p.get_height()),
                    ha = 'center', va = 'center',
                    xytext = (0, 6),
                    textcoords = 'offset points',color='purple')
plt.legend(title='Brain Stroke',loc='upper left',labels=['Negative','Positive'])
```

```
<matplotlib.legend.Legend at 0x7d3d4423d810>
```

Figure 2: Vertical Bar Graph showing *Gender* with *Brain Stoke*

In our dataset, the occurrence of stroke varies between genders. Among males, 351 records indicate stroke, while 15,890 show no stroke. In comparison, 429 females have experienced stroke, with 23,777 records showing no stroke. This distribution suggests that females are slightly more affected by stroke in our dataset. Understanding such gender disparities is crucial for tailored preventive strategies and healthcare interventions. Further analysis can delve into the factors contributing to this discrepancy, potentially uncovering insights into gender-specific risk factors for stroke.

3. What is the total number of individuals who had experienced a stroke and also have hypertension?



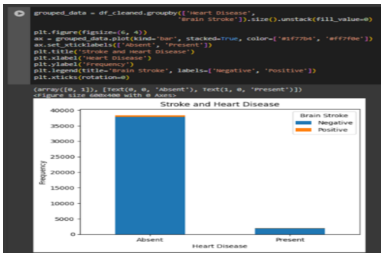
```
grouped_data = df_cleaned.groupby(['Hypertension',
                                   'Brain Stroke']).size().unstack(fill_value=0)

plt.figure(figsize=(6, 4))
ax = grouped_data.plot(kind='barh', stacked=True, color=['indianred', 'olive'])
ax.set_yticklabels(['Absent', 'Present'])
plt.title('Stroke and Hypertension')
plt.xlabel('Frequency')
plt.ylabel('Hypertension')
plt.legend(title='Brain Stroke', labels=['Negative', 'Positive'])
```

```
<matplotlib.legend.Legend at 0x7d3d44289ea0>
<Figure size 600x400 with 0 Axes>
```

Figure 3: Horizontal Bar graph showing *Hypertension* with *Brain Stroke*

The total number of individuals who experienced a stroke and have hypertension is approximately 5000. Notably, there's a higher frequency of stroke occurrence in individuals without hypertension, hinting at a potential inverse relationship between stroke and hypertension presence.

4. Find the total number of people who beard the stroke have heart disease?



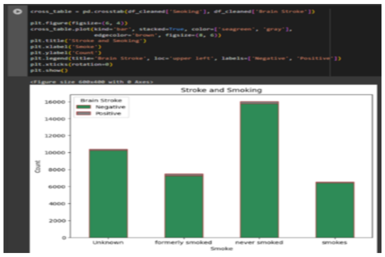
```
grouped_data = df_cleaned.groupby(['Heart Disease',
                                   'Brain Stroke']).size().unstack(fill_value=0)

plt.figure(figsize=(6, 4))
ax = grouped_data.plot(kind='bar', stacked=True, color=['#1f77b4', '#ff7f0e'])
ax.set_xticklabels(['Absent', 'Present'])
plt.title('Stroke and Heart Disease')
plt.xlabel('Heart Disease')
plt.ylabel('Frequency')
plt.legend(title='Brain Stroke', labels=['Negative', 'Positive'])
plt.xticks(rotation=0)
```

```
(array([0, 1]), [Text(0, 0, 'Absent'), Text(1, 0, 'Present')])
<Figure size 600x400 with 0 Axes>
```

Figure 4: Vertical Bar graph showing *Heart Disease* with *Brain Stroke*

The data indicates fewer instances of heart disease among individuals who have experienced stroke compared to those without heart disease. Furthermore, individuals with stroke outnumber those without, suggesting a higher prevalence of stroke regardless of heart disease status.

5. How does the distribution of strokes differ between smokers non-smokers?



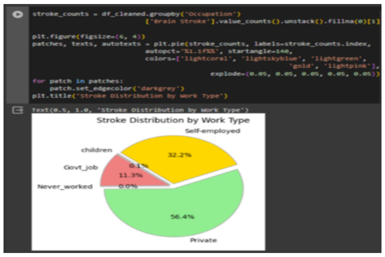
```
cross_table = pd.crosstab(df_cleaned['Smoking'], df_cleaned['Brain Stroke'])

plt.figure(figsize=(6, 4))
cross_table.plot(kind='bar', stacked=True, color=['seagreen', 'gray'],
                 edgecolor='brown', figsize=(8, 6))
plt.title('Stroke and Smoking')
plt.xlabel('Smoke')
plt.ylabel('Count')
plt.legend(title='Brain Stroke', loc='upper left', labels=['Negative', 'Positive'])
plt.xticks(rotation=0)
plt.show()
```

```
<Figure size 600x400 with 0 Axes>
```

Figure 5: Vertical Bar graph showing *Smoking* with *Brain Stroke*

The distribution of strokes differs between smokers and non-smokers. The graph illustrates a higher count of individuals who never smoked, with a significant portion experiencing stroke. Conversely, fewer individuals who smoked are depicted, with a lower incidence of stroke. This suggests a potential association between non-smoking status and increased stroke occurrence.

6. What is the occurrence of stoke based on occupation?



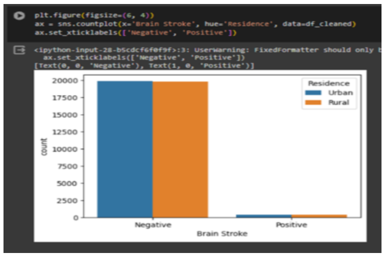
```
stroke_counts = df_cleaned.groupby('Occupation')
                                ['Brain Stroke'].value_counts().unstack().fillna(0)[1]

plt.figure(figsize=(6, 4))
patches, texts, autotexts = plt.pie(stroke_counts, labels=stroke_counts.index,
                                autopct='%1.1f%%', startangle=140,
                                colors=['lightcoral', 'lightskyblue', 'lightgreen',
                                                                 'gold', 'lightpink'],
                                                   explode=(0.05, 0.05, 0.05, 0.05, 0.05))
for patch in patches:
    patch.set_edgecolor('darkgrey')
plt.title('Stroke Distribution by Work Type')
```

```
Text(0.5, 1.0, 'Stroke Distribution by Work Type')
```

Figure 6: Pie Chart showing Distribution of *Brain Stroke* among *Occupation*

The graph displays the distribution of stroke occurrence across various occupation types. The majority (56.4%) are employed in the private sector, followed by self-employed individuals (32.2%). However, no records are present for individuals who have never worked, suggesting a potential data gap or exclusion from the dataset.

7. How does the occurrence of stroke vary based on the places individuals live?



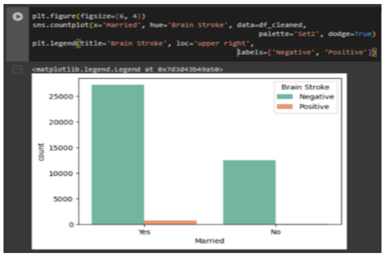
```
plt.figure(figsize=(6, 4))
ax = sns.countplot(x='Brain Stroke', hue='Residence', data=df_cleaned)
ax.set_xticklabels(['Negative', 'Positive'])
```

```
<ipython-input-28-b5cdcf6f0f9f>:3: UserWarning: FixedFormatter should only b
  ax.set_xticklabels(['Negative', 'Positive'])
[Text(0, 0, 'Negative'), Text(1, 0, 'Positive')]
```

Figure 7: Vertical Bar Graph showing *Brain Stroke* with *Residence Type*

The graph depicts a notable discrepancy in stroke occurrence based on individuals' places of residence. A higher count of individuals in both urban and rural areas is observed to have not experienced brain stroke, compared to those who have. This finding suggests a lower incidence of stroke among residents of both urban and rural areas, highlighting potential differences in lifestyle factors, healthcare access, or environmental influences between urban and rural populations.

8. What is the relationship between martial status and the occurrence of brain stroke?



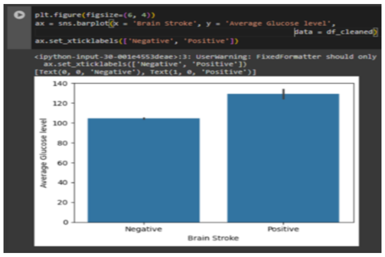
```
plt.figure(figsize=(6, 4))
sns.countplot(x='Married', hue='Brain Stroke', data=df_cleaned,
                                         palette='Set2', dodge=True)
plt.legend(title='Brain Stroke', loc='upper right',
                                     labels=['Negative', 'Positive'])
```

```
<matplotlib.legend.Legend at 0x7d3d43b49a50>
```

Figure 8: Clustered Bar Graph showing *Brain Stroke* with *Martial Status*

The graph highlights a higher count of married individuals who have experienced stroke compared to those who have not. This suggests a potential association between marital status and increased risk of stroke occurrence among married individuals.

9. How does the level of glucose relate to the occurrence of stroke?



```
plt.figure(figsize=(6, 4))
ax = sns.barplot(x = 'Brain Stroke', y = 'Average Glucose level',
                                                   data = df_cleaned)
ax.set_xticklabels(['Negative', 'Positive'])
```

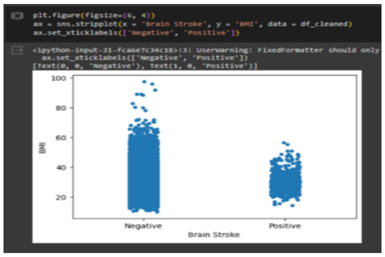
```
<ipython-input-30-001e4553deae>:3: UserWarning: FixedFormatter should only
  ax.set_xticklabels(['Negative', 'Positive'])
[Text(0, 0, 'Negative'), Text(1, 0, 'Positive')]
```

Figure 9: Vertical Bar Graph showing *Brain Stroke* with *Glucose Level*

The graph reveals a clear trend indicating a correlation between high levels of glucose and brain stroke occurrence. Individuals with elevated glucose levels are depicted as more likely to have experienced brain stroke, suggesting a potential association between high glucose levels and increased risk of stroke incidence. This finding underscores the importance of monitoring and managing glucose levels as part of stroke prevention strategies.

10. What is the association between body mass index (BMI) and the occurrence of stroke?

The graph illustrates a consistent prevalence of brain stroke across BMI values ranging from 20 to 45. This observation suggests a potential association between BMI within this range and the occurrence of brain stroke. Understanding this relationship could inform preventive measures targeting individuals within specific BMI ranges to mitigate stroke risk.

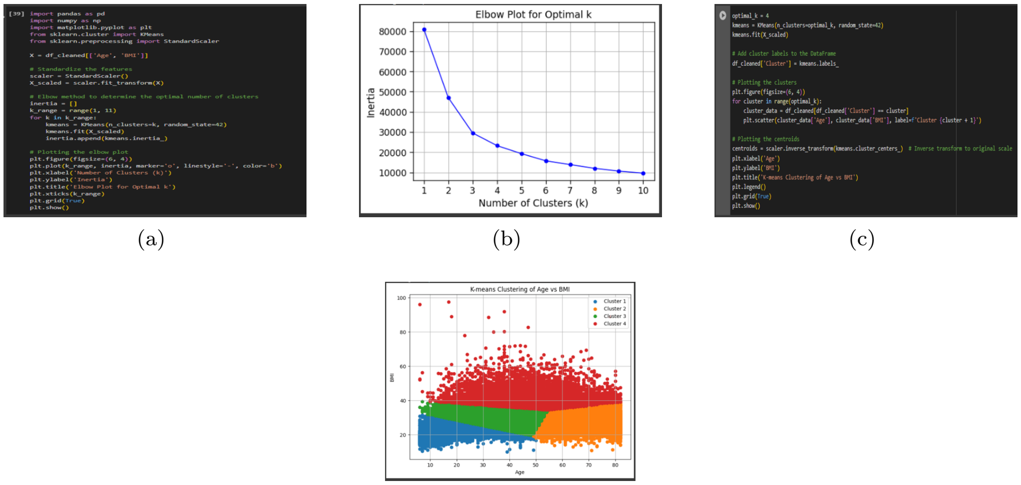


```
plt.figure(figsize=(6, 4))
ax = sns.stripplot(x = 'Brain Stroke', y = 'BMI', data = df_cleaned)
ax.set_xticklabels(['Negative', 'Positive'])
```

```
<ipython-input-31-fca6e7c34c18>:3: UserWarning: FixedFormatter should only
  ax.set_xticklabels(['Negative', 'Positive'])
[Text(0, 0, 'Negative'), Text(1, 0, 'Positive')]
```

Figure 10: Strip Plot showing *Brain Stroke* with *BMI*

11. Is there any relation between Age and BMI?



```
[39] import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     from sklearn.cluster import KMeans
     from sklearn.preprocessing import StandardScaler

     X = df_cleaned[['Age', 'BMI']]

     # Standardize the features
     scaler = StandardScaler()
     X_scaled = scaler.fit_transform(X)

     # Elbow method to determine the optimal number of clusters
     inertia = []
     k_range = range(1, 11)
     for k in k_range:
         kmeans = KMeans(n_clusters=k, random_state=42)
         kmeans.fit(X_scaled)
         inertia.append(kmeans.inertia_)

     # Plotting the elbow plot
     plt.figure(figsize=(6, 4))
     plt.plot(k_range, inertia, marker='o', linestyle='-', color='b')
     plt.xlabel('Number of Clusters (k)')
     plt.ylabel('Inertia')
     plt.title('Elbow Plot for Optimal k')
     plt.xticks(k_range)
     plt.grid(True)
     plt.show()
```

(a)

(b)

```
optimal_k = 4
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
kmeans.fit(X_scaled)

# Add cluster labels to the DataFrame
df_cleaned['Cluster'] = kmeans.labels_

# Plotting the clusters
plt.figure(figsize=(6, 4))
for cluster in range(optimal_k):
    cluster_data = df_cleaned[df_cleaned['Cluster'] == cluster]
    plt.scatter(cluster_data['Age'], cluster_data['BMI'], label=f'Cluster {cluster + 1}')

# Plotting the centroids
centroids = scaler.inverse_transform(kmeans.cluster_centers_)  # Inverse transform to original scale
plt.xlabel('Age')
plt.ylabel('BMI')
plt.title('K-means Clustering of Age vs BMI')
plt.legend()
plt.grid(True)
plt.show()
```

(c)

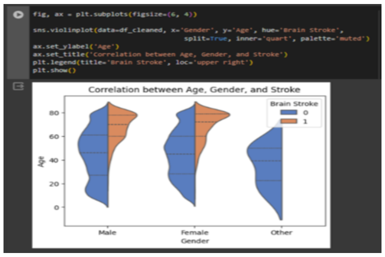
(d)

Figure 11: K-Means

K-means clustering analysis was performed to explore the relationship between Age and BMI. The analysis revealed the presence of four distinct clusters, each representing a group of data points with similarities in Age and BMI values. By examining these clusters, we can gain insights into potential patterns or relationships between Age and BMI within our dataset

12. How does the occurrence of brain stroke vary with age and gender?

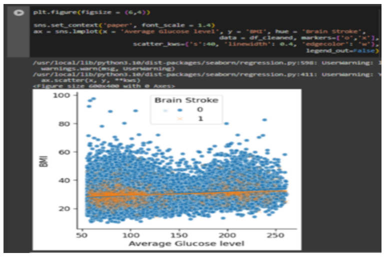
Analysis of the data reveals that female individuals show a higher susceptibility to brain stroke compared to males. However, when examining the relationship between age and brain stroke occurrence, no clear correlation emerges. This suggests that while gender may play a significant role in stroke risk, age alone may not be the sole determinant, indicating the influence of other factors in stroke occurrence among different demographic groups.



```
fig, ax = plt.subplots(figsize=(6, 4))

sns.violinplot(data=df_cleaned, x='Gender', y='Age', hue='Brain Stroke',
                                          split=True, inner='quart', palette='muted')
ax.set_ylabel('Age')
ax.set_title('Correlation between Age, Gender, and Stroke')
plt.legend(title='Brain Stroke', loc='upper right')
plt.show()
```

Figure 12: Violin Plot showing *Brain Stroke* with *Age* and *BMI*

13. How does the occurrence of stroke relate to both glucose level and body mass index (BMI)?



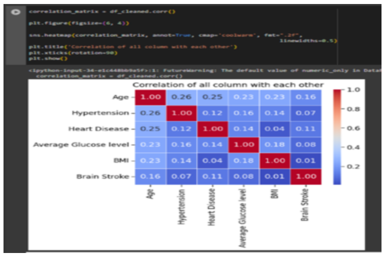
```
plt.figure(figsize = (6,4))

sns.set_context('paper', font_scale = 1.4)
ax = sns.lmplot(x = 'Average Glucose level', y = 'BMI', hue = 'Brain Stroke',
                                          data = df_cleaned, markers=['o','x'],
                          scatter_kws={'s':40, 'linewidth': 0.4, 'edgecolor': 'w'},
                                                                legend_out=False)
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/regression.py:598: UserWarning: 
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.10/dist-packages/seaborn/regression.py:411: UserWarning: 
  ax.scatter(x, y, **kws)
<Figure size 600x400 with 0 Axes>
```

Figure 13: Scatter Plot showing *Glucose Level* with *BMI* based on *Brain Stroke*

The graph highlights that individuals with a BMI ranging from 25 to 35 and average glucose levels between 50 and 120 are particularly at risk of brain stroke. This pattern suggests a potential association between these factors and stroke occurrence. Understanding this relationship could inform targeted preventive strategies aimed at managing both glucose levels and BMI to reduce the risk of stroke.

14. How are all columns correlated with each other within the dataset?



```
correlation_matrix = df_cleaned.corr()

plt.figure(figsize=(6, 4))

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f",
                                                          linewidths=0.5)
plt.title('Correlation of all column with each other')
plt.xticks(rotation=90)
plt.show()
```

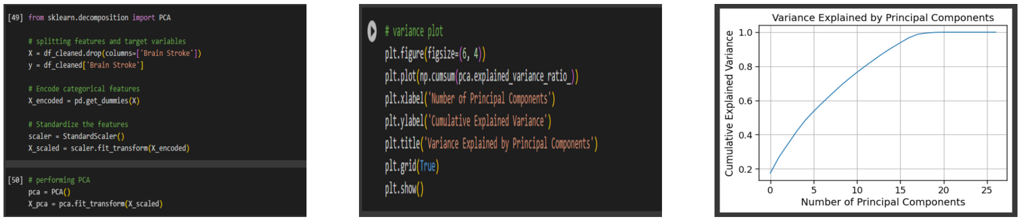
```
<ipython-input-34-e1c448bb9a5f>:1: FutureWarning: The default value of numeric_only in Data
  correlation_matrix = df_cleaned.corr()
```

Correlation of all column with each other
Age 1.00 0.26 0.25 0.23 0.23 0.16
Hypertension 0.26 1.00 0.12 0.16 0.14 0.07
Heart Disease 0.25 0.12 1.00 0.14 0.04 0.11
Average Glucose level 0.23 0.16 0.14 1.00 0.18 0.08
BMI 0.23 0.14 0.04 0.18 1.00 0.01
Brain Stroke 0.16 0.07 0.11 0.08 0.01 1.00
Age
Hypertension
Heart Disease
Average Glucose level
BMI
Brain Stroke
1.0
0.8
0.6
0.4
0.2

Figure 14: HeatMap showing correlation among different variables

The analysis of correlations within the dataset reveals several notable associations. There is a significant correlation between brain stroke and age, indicating that age plays a crucial role in stroke occurrence. Additionally, a correlation is observed between brain stroke and heart disease, suggesting that individuals with heart disease may be more predisposed to stroke. However, the correlation between BMI and brain stroke is weak, implying that while BMI may influence stroke risk, its impact is less pronounced compared to age and heart disease.

15. Will performing Principal Component Analysis (PCA) speed up the prediction process?



```
[49] from sklearn.decomposition import PCA

     # splitting features and target variables
     X = df_cleaned.drop(columns=['Brain Stroke'])
     y = df_cleaned['Brain Stroke']

     # Encode categorical features
     X_encoded = pd.get_dummies(X)

     # Standardize the features
     scaler = StandardScaler()
     X_scaled = scaler.fit_transform(X_encoded)
```

```
[50] # performing PCA
     pca = PCA()
     X_pca = pca.fit_transform(X_scaled)
```

(a)

```
# variance plot
plt.figure(figsize=(6, 4))
plt.plot(np.cumsum(pca.explained_variance_ratio_))
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('Variance Explained by Principal Components')
plt.grid(True)
plt.show()
```

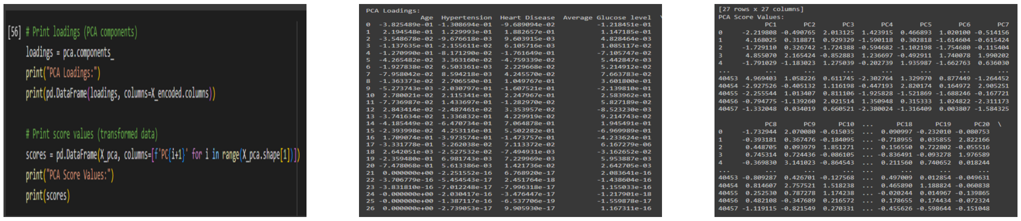
(b)

Variance Explained by Principal Components
Cumulative Explained Variance
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
Number of Principal Components

(c)

Figure 15: Principal Component Analysis

After performing Principal Component Analysis (PCA), it was observed that to achieve 80% variance explained, 11 principal components should be selected, as indicated by the variance plot. However, considering that the original dataset comprises only 10 features, the use of PCA in this context may not be particularly advantageous. In fact, selecting 11 principal components would result in an increase in the number of features rather than reducing dimensionality.



```
[56] # Print loadings (PCA components)
     loadings = pca.components_
     print("PCA Loadings:")
     print(pd.DataFrame(loadings, columns=X_encoded.columns))

     # Print score values (transformed data)
     scores = pd.DataFrame(X_pca, columns=[f'PC{i+1}' for i in range(X_pca.shape[1])])
     print("PCA Score Values:")
     print(scores)
```

(a)

```
PCA Loadings:
             Age  Hypertension  Heart Disease  Average Glucose level
0  -3.825489e-01 -1.308694e-01  -9.689094e-02          -1.218451e-01
1   2.194548e-01  1.229993e-01   1.882657e-01           1.147185e-01
2  -3.548678e-02 -9.676618e-03   9.603915e-03           4.828464e-03
3  -1.137635e-01 -2.155611e-02   6.105716e-03           1.085117e-02
4  -1.270990e-01 -8.171290e-02  -1.761649e-01          -7.105747e-02
5  -4.265482e-02  3.363160e-02  -4.759339e-02           5.442847e-03
6  -1.927838e-02  6.503361e-03   2.229668e-02           5.214912e-02
7  -7.958042e-02  8.594218e-03   4.245570e-02           7.663783e-02
8  -1.363373e-02  2.706550e-01   1.049767e-01           3.601800e-01
9  -5.273743e-03 -2.030797e-01  -1.607521e-01          -2.139810e-01
10  2.780021e-02  2.115341e-01   2.247967e-01           2.583962e-01
11 -7.736987e-02  1.433697e-01  -1.282970e-02           5.827189e-02
12  2.843414e-02 -2.487461e-02   3.353957e-02          -8.523230e-03
13 -3.741634e-02  1.336832e-01   4.229919e-02           9.214743e-02
14 -4.185449e-02 -6.470734e-01   7.064878e-01           1.945491e-01
15 -2.393998e-02  4.253116e-01   5.502282e-01          -6.969989e-01
16  1.709074e-01 -3.973574e-01  -1.473757e-01          -4.233624e-01
17 -3.331778e-01  5.262038e-02   7.113372e-02           6.167279e-06
18  2.642051e-03 -2.527532e-02  -7.494931e-03          -3.162652e-02
19 -2.359480e-01  6.981743e-03   7.229969e-03           5.953887e-03
20 -7.478068e-01  5.613386e-03   1.421736e-02           2.642705e-03
21  0.000000e+00 -2.251552e-16   6.768920e-17           2.083641e-16
22 -3.706779e-16 -5.454543e-17   2.451764e-18          -1.438604e-16
23 -3.831810e-16 -7.012248e-17  -7.996318e-17           1.155033e-16
24 -0.000000e+00 -2.030417e-16  -3.476447e-17          -1.217901e-18
25 -0.000000e+00 -1.387117e-16  -6.537706e-19          -1.559878e-17
26  0.000000e+00 -2.739053e-17   9.905930e-17           1.167311e-16
```

(b)

```
[27 rows x 27 columns]
PCA Score Values:
            PC1       PC2       PC3       PC4       PC5       PC6       PC7
0     -2.219808 -0.490765  2.013125  1.423915  0.466893  1.020100 -0.514156
1      4.168025  0.318871  0.929329 -1.590118  0.302818 -1.614604 -0.615424
2     -1.729110  0.326742 -1.724388 -0.594682 -1.102198 -1.754680 -0.115404
3      4.855070  2.165424 -0.852883  1.236697 -0.492911  1.740078  1.990202
4     -1.791029 -1.183023  1.275039 -0.202739  1.935987 -1.662763  0.636030
...         ...       ...       ...       ...       ...       ...       ...
40453  4.969403  1.058226  0.611745 -2.302764  1.329970  0.877449 -1.264452
40454 -2.927526 -0.405132  1.116198 -0.447193  2.820174  0.164972  2.905251
40455 -2.255544  1.013407  0.811106 -1.925828 -1.521869 -1.688246 -0.167721
40456 -0.794775 -1.139260  2.021514  1.350948  0.315333  1.024822 -2.311173
40457 -1.332048  0.034019  0.660521 -2.380024 -1.316409  0.003807 -1.584325

            PC8       PC9      PC10  ...      PC18      PC19      PC20  \
0     -1.732944  2.070080 -0.615035  ...  0.090997 -0.232010 -0.080753
1     -0.393181  0.367476 -0.184095  ... -0.718955  0.035855  2.822166
2      0.448705  0.093979  1.851271  ...  0.156550  0.722802 -0.055516
3      0.745314  0.724436 -0.086105  ... -0.836491 -0.093278  1.976589
4     -0.369830  3.141023 -0.864543  ...  0.211560  0.740652  0.018244
...         ...       ...       ...  ...       ...       ...       ...
40453 -0.809287  0.426701 -0.127568  ...  0.497009  0.012854 -0.049631
40454  0.814607  2.757521  1.518238  ...  0.465890  1.188824 -0.060838
40455  0.252530  0.787278  1.174238  ... -0.020244  0.014967 -0.139865
40456  0.482108 -0.347689  0.216572  ...  0.178655  0.174434 -0.072324
40457 -1.119115 -0.821549  0.270331  ... -0.455626 -0.598644 -0.151048
```

(c)

Figure 16: Principal Component Analysis

Therefore, while PCA is often utilized for dimensionality reduction and speeding up the prediction process, in this scenario, it may not provide any significant benefits and could potentially complicate the analysis without improving prediction speed.