

Aim: Data Visualization and Storytelling

- a) Create meaningful visualizations using data visualization tools.
- b) Combine multiple visualizations to tell a compelling data story.
- c) Present the findings and insights in a clear and concise manner.

CODE:

➤ ***Importing libraries***

```
import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns
```

➤ ***Load Dataset***

```
df = pd.read_csv("healthcare-dataset-stroke-data.csv")  
df.head()
```

➤ ***Dataset Description***

```
df.info()  
df.describe(include="all")
```

➤ ***Handle Missing Values***

```
df['bmi'] = pd.to_numeric(df['bmi'], errors='coerce')  
df['bmi'].fillna(df['bmi'].mean(), inplace=True)  
df['smoking_status'].replace("Unknown", np.nan, inplace=True)  
df['smoking_status'].fillna(df['smoking_status'].mode()[0], inplace=True)  
df.isnull().sum()
```

➤ ***Bar Plot – Stroke Count by Gender***

```
plt.figure(figsize=(6,4))  
sns.countplot(data=df, x='gender', hue='stroke')  
plt.title("Stroke Count by Gender")  
plt.show()
```

➤ ***Boxplot – Age vs Stroke***

```
plt.figure(figsize=(6,4))  
sns.boxplot(data=df, x='stroke', y='age')  
plt.title("Age Distribution for Stroke vs Non-Stroke Patients")  
plt.show()
```

➤ ***Line Plot – Average Glucose Level by Age***

```
df_sorted = df.sort_values("age")  
plt.figure(figsize=(8,4))  
plt.plot(df_sorted['age'], df_sorted['avg_glucose_level'])  
plt.title("Glucose Level Trend Across Age")  
plt.xlabel("Age")  
plt.ylabel("Avg Glucose Level")  
plt.show()
```

➤ ***Heatmap – Correlation Between Numerical Variables***

```
plt.figure(figsize=(8,6))
numeric_df = df.select_dtypes(include=['int64','float64'])
sns.heatmap(numeric_df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

➤ ***Violin Plot – BMI vs Work Type***

```
plt.figure(figsize=(8,5))
sns.violinplot(data=df, x='work_type', y='bmi')
plt.xticks(rotation=45)
plt.title("BMI Distribution Across Work Types")
plt.show()
```

➤ ***Pair Plot – Numerical Variables***

```
sns.pairplot(df[['age','avg_glucose_level','bmi','stroke']], hue='stroke')
plt.show()
```

➤ ***Scatter Plot – Glucose vs BMI***

```
plt.figure(figsize=(6,4))
sns.scatterplot(data=df, x='avg_glucose_level', y='bmi', hue='stroke')
plt.title("Glucose Level vs BMI!")
plt.show()
```

➤ ***Distribution Plot – Age***

```
plt.figure(figsize=(6,4))
sns.histplot(df['age'], kde=True)
plt.title("Age Distribution")
plt.show()
```

➤ ***Pie Chart – Work Type Distribution***

```
plt.figure(figsize=(6,6))
df['work_type'].value_counts().plot.pie(autopct='%1.1f%%')
plt.title("Work Type Distribution")
plt.show()
```

➤ ***Histogram – Glucose Level***

```
plt.figure(figsize=(6,4))
plt.hist(df['avg_glucose_level'], bins=20)
plt.title("Histogram of Glucose Levels")
plt.xlabel("Glucose Level")
plt.ylabel("Frequency")
plt.show()
```

Output:

Sheth L.U.J College of Arts & Sir M.V. College of Science and Commerce

Data Science

PRACTICAL NO. 10

```
[1]: import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
[2]: df = pd.read_csv("healthcare-dataset-stroke-data.csv")  
df.head()  
  
id gender age hypertension heart_disease ever_married work_type Residence_type avg_glucose_level bmi smoking_status stroke  
0 9046 Male 67.0 0 1 Yes Private Urban 228.69 36.6 formerly smoked 1  
1 51676 Female 61.0 0 0 Yes Self-employed Rural 202.21 NaN never smoked 1  
2 31112 Male 80.0 0 1 Yes Private Rural 105.92 32.5 never smoked 1  
3 60182 Female 49.0 0 0 Yes Private Urban 171.23 34.4 smokes 1
```

```
[3]: df.info()  
df.describe(include="all")  
  
...<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 5118 entries, 0 to 5109  
Data columns (total 12 columns):  
 # Column Non-Null Count Dtype  
 # ...  
 0 id 5118 non-null int64  
 1 gender 5118 non-null object  
 2 age 5118 non-null float64  
 3 hypertension 5118 non-null int64  
 4 heart_disease 5118 non-null int64  
 5 ever_married 5118 non-null object  
 6 work_type 5118 non-null object  
 7 Residence_type 5118 non-null object  
 8 avg_glucose_level 5118 non-null float64  
 9 bmi 4909 non-null float64  
 10 smoking_status 5118 non-null object  
 11 stroke 5118 non-null int64  
dtypes: float64(3), int64(4), object(5)  
memory usage: 479.2+ KB
```

```
...  
count 5110.000000 5110 5110.000000 5110.000000 5110 5110 5110 5110.000000 4909.000000  
unique NaN 3 NaN NaN NaN 2 5 2 NaN NaN  
top NaN Female NaN NaN NaN Yes Private Urban NaN NaN  
freq NaN 2994 NaN NaN NaN 3353 2925 2596 NaN NaN  
mean 36517.829354 NaN 43.226614 0.097456 0.054012 NaN NaN NaN 106.147677 28.893237  
std 21161.721625 NaN 22.612647 0.296607 0.226063 NaN NaN NaN 45.283560 7.854067  
min 67.000000 NaN 0.080000 0.000000 0.000000 NaN NaN NaN 55.120000 10.300000  
25% 17741.250000 NaN 25.000000 0.000000 0.000000 NaN NaN NaN 77.245000 23.500000  
50% 36932.000000 NaN 45.000000 0.000000 0.000000 NaN NaN NaN 91.885000 28.100000  
75% 54682.000000 NaN 61.000000 0.000000 0.000000 NaN NaN NaN 114.090000 33.100000  
max 72940.000000 NaN 82.000000 1.000000 1.000000 NaN NaN NaN 271.740000 97.600000
```

This dataset contains healthcare records of individuals with attributes such as age, gender, lifestyle, and medical conditions. It includes a stroke column indicating whether the person experienced a stroke. Numeric attributes like age, glucose level, and BMI help understand health patterns, while categorical attributes like work type and smoking status support demographic analysis. The dataset is ideal for identifying risk factors contributing to stroke.

Sheth L.U.J College of Arts & Sir M.V. College of Science and Commerce

Data Science

PRACTICAL NO. 10

The screenshot shows a Google Colab notebook titled "Prac10-Data Visualization and Storytelling.ipynb". The code cell contains several lines of Python code for handling missing values in a DataFrame:

```
df['bmi'] = pd.to_numeric(df['bmi'], errors='coerce')
df['bmi'].fillna(df['bmi'].mean(), inplace=True)

df['smoking_status'].replace("Unknown", np.nan, inplace=True)
df['smoking_status'].fillna(df['smoking_status'].mode()[0], inplace=True)

df.isnull().sum()
```

Output of the code cell:

```
/tmp/ipython-input-2698187929.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values al
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(
df[df['bmi'].mean(), inplace=True]
/tmp/ipython-input-2698187929.py:4: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values al
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(
df['smoking_status'].replace("Unknown", np.nan, inplace=True)
/tmp/ipython-input-2698187929.py:5: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values al
```

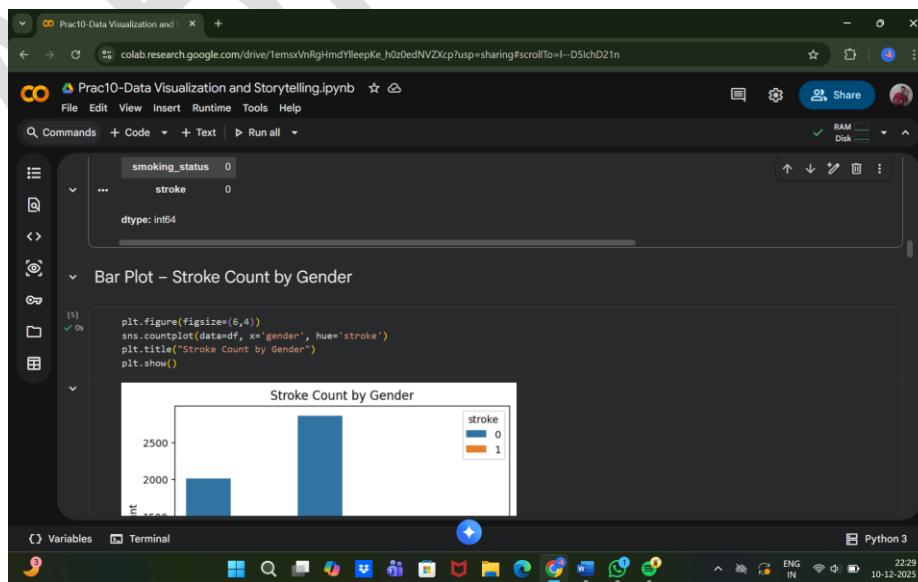
Variables and Terminal tabs are visible at the bottom.

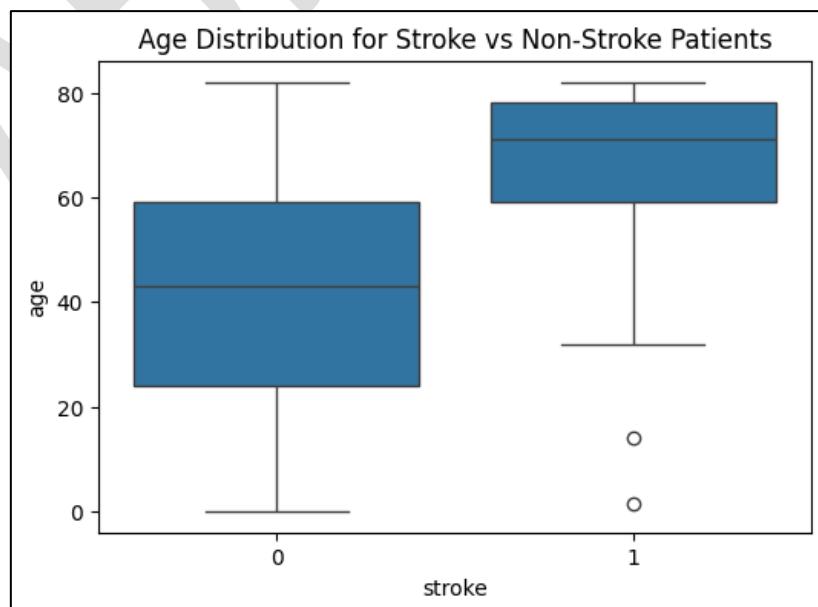
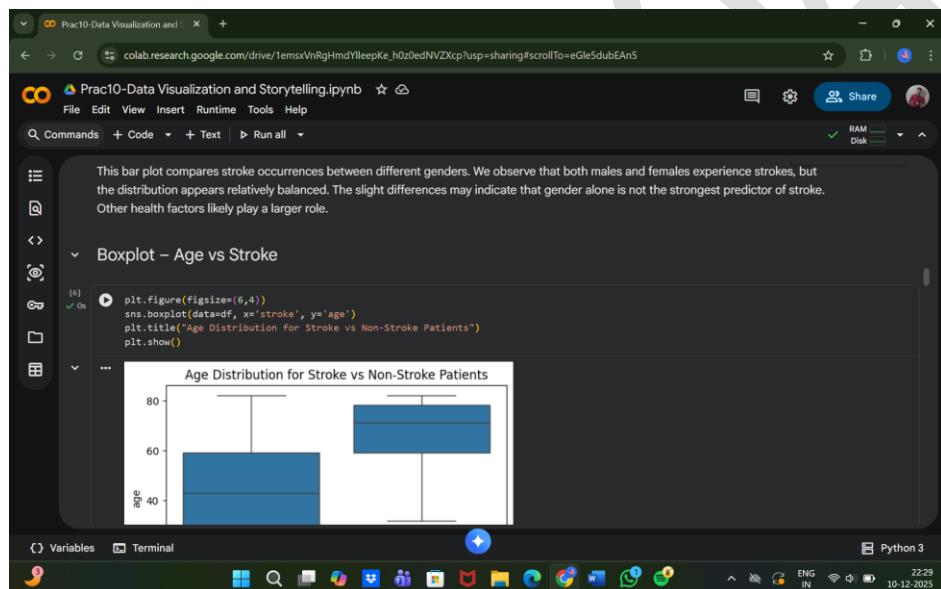
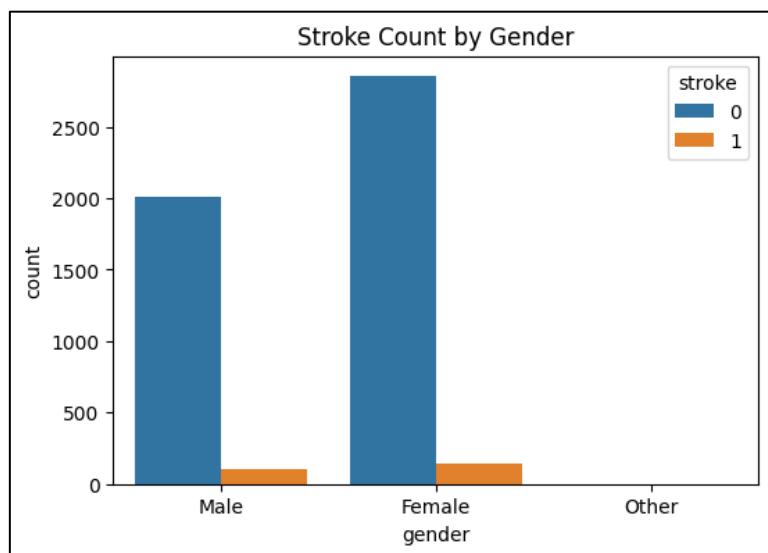
The screenshot shows a Google Colab notebook titled "Prac10-Data Visualization and Storytelling.ipynb". The code cell displays a summary of missing values (NaN) across various columns:

```
For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df' ... d(
... df['smoking_status'].fillna(df['smoking_status'].mode()[0], inplace=True)
```

Column	Value
id	0
gender	0
age	0
hypertension	0
heart_disease	0
ever_married	0
work_type	0
Residence_type	0
avg_glucose_level	0
bmi	0
smoking_status	0
stroke	0

Variables and Terminal tabs are visible at the bottom.

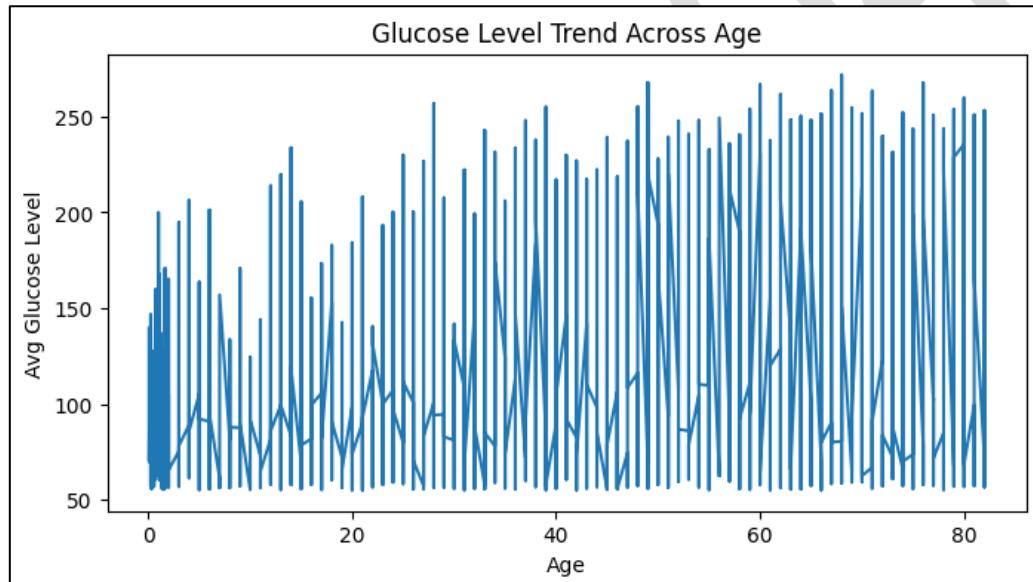




The boxplot clearly shows that stroke cases occur predominantly among older individuals. The median age for stroke patients is much higher than for non-stroke patients. This highlights age as a major risk factor for stroke. Younger individuals show almost no stroke cases.

Line Plot – Average Glucose Level by Age

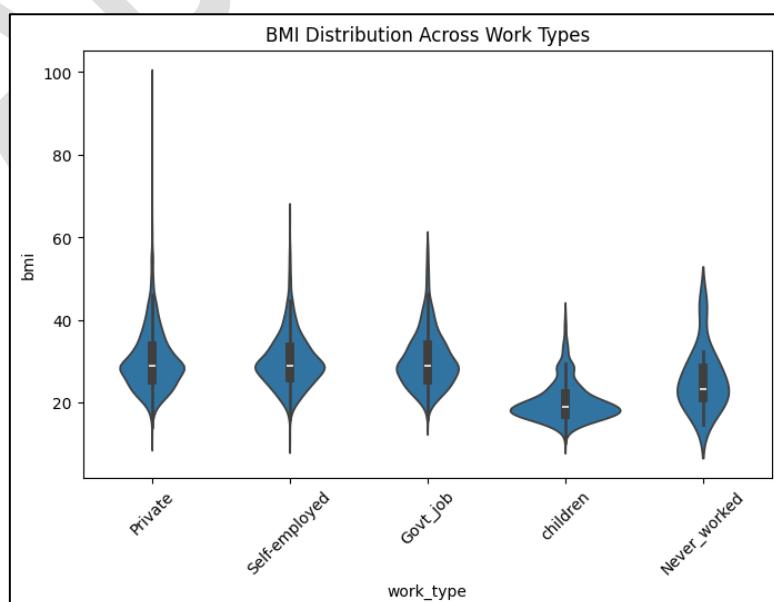
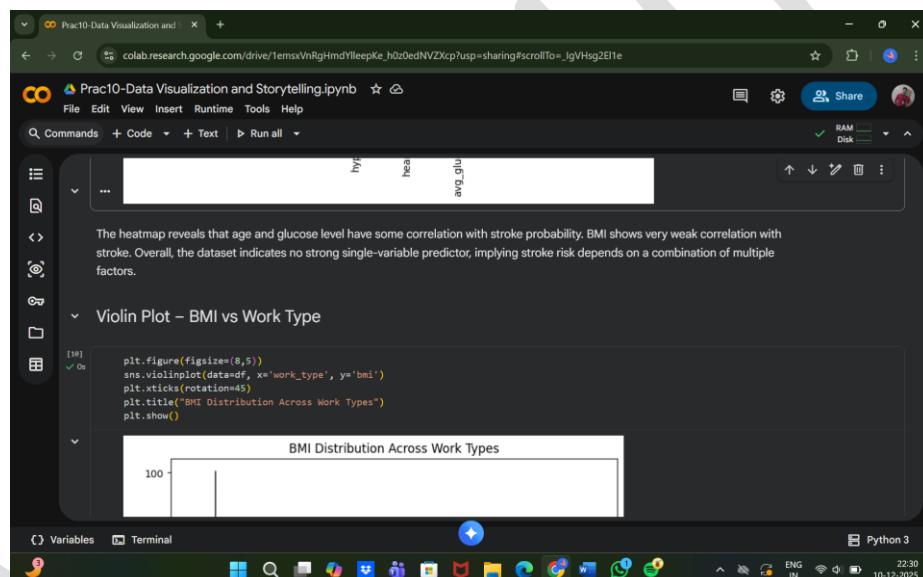
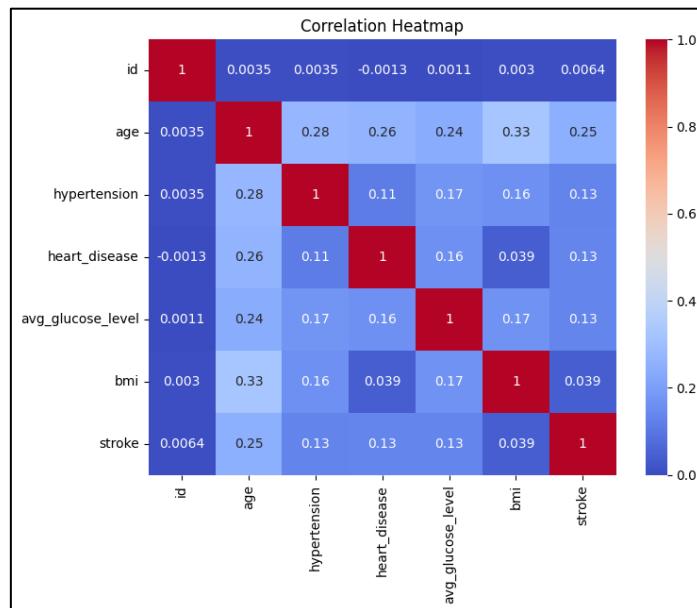
```
df_sorted = df.sort_values("age")
plt.figure(figsize=(8,4))
plt.plot(df_sorted['age'], df_sorted['avg_glucose_level'])
plt.title("Glucose Level Trend Across Age")
plt.xlabel("Age")
plt.ylabel("Avg Glucose Level")
plt.show()
```

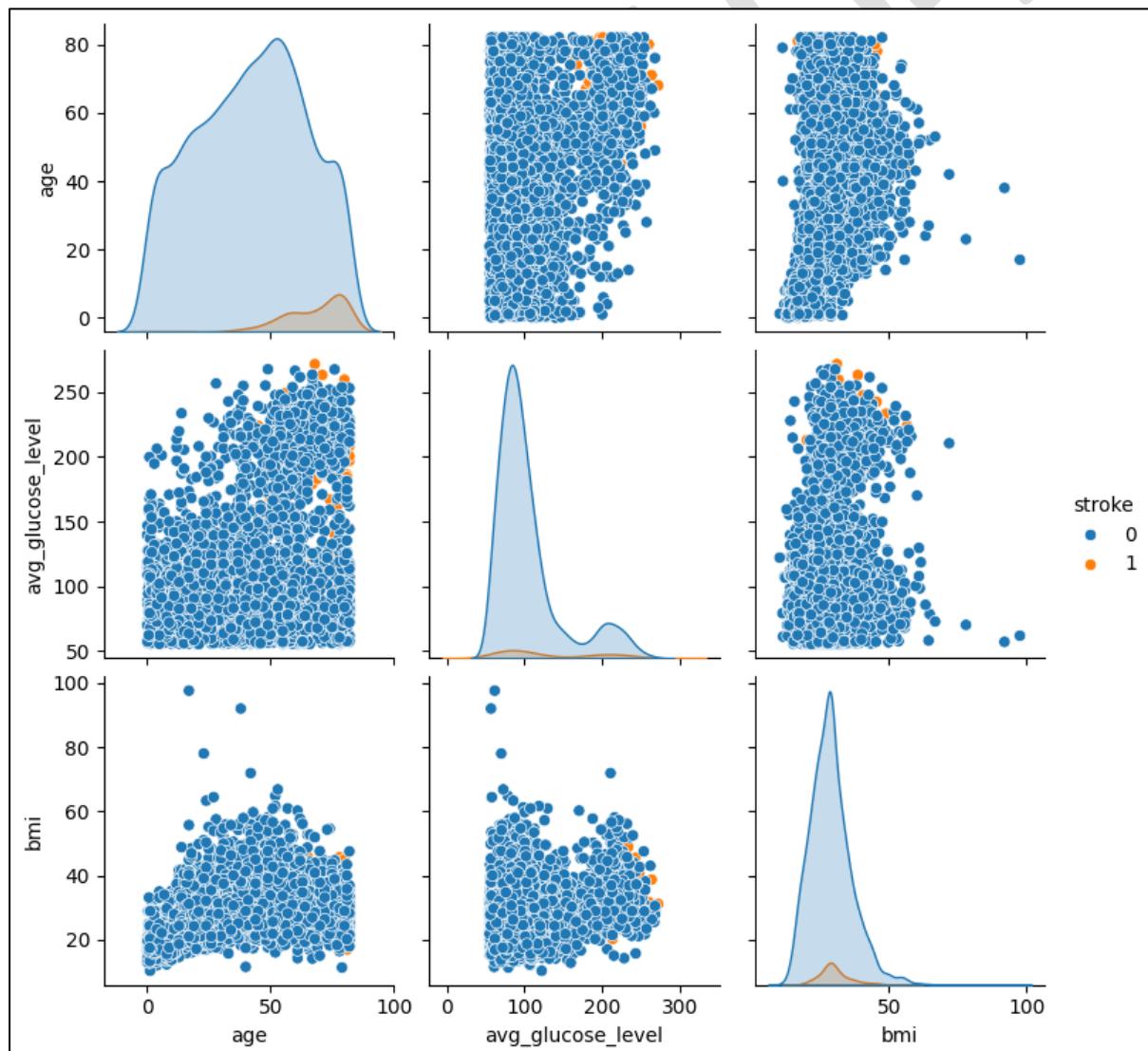
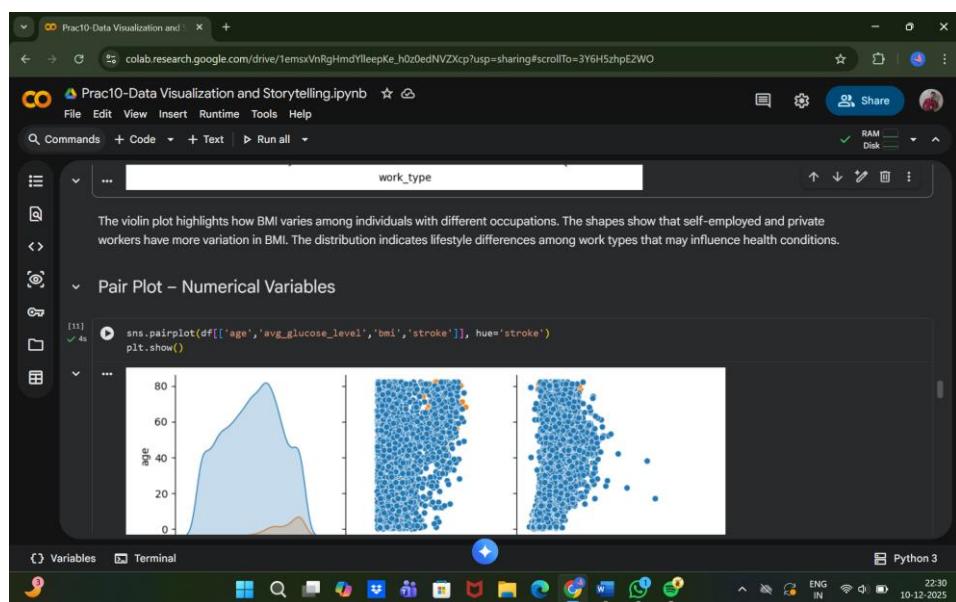


The line plot shows fluctuations in glucose levels across ages. There is a noticeable increase in glucose levels at older ages, which correlates with higher stroke cases. This suggests that glucose instability and aging jointly increase stroke risk. The trend emphasizes monitoring glucose as individuals age.

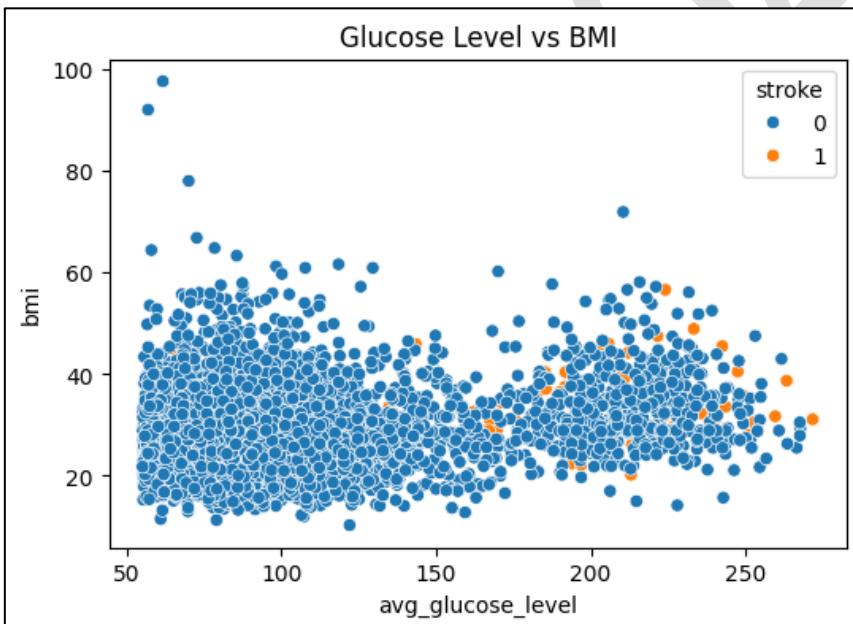
Heatmap – Correlation Between Numerical Variables

```
plt.figure(figsize=(8,6))
numeric_df = df.select_dtypes(include=['int64','float64'])
sns.heatmap(numeric_df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```

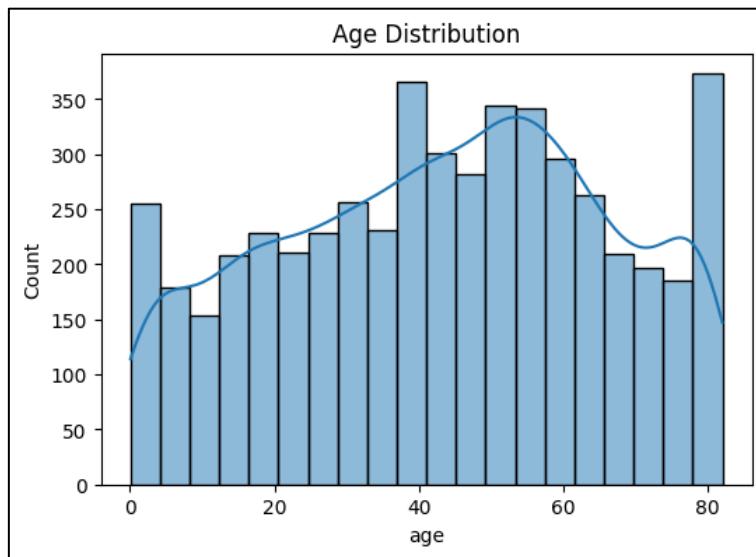




```
[12]: plt.figure(figsize=(6,4))
sns.scatterplot(data=df, x='avg_glucose_level', y='bmi', hue='stroke')
plt.title("Glucose Level vs BMI")
plt.show()
```



```
[13]: plt.figure(figsize=(6,4))
sns.histplot(df['age'], kde=True)
plt.title("Age Distribution")
plt.show()
```



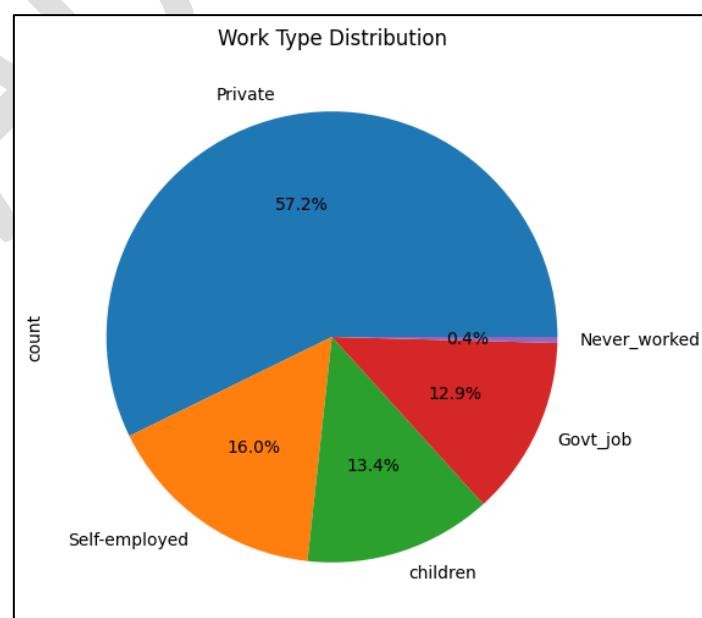
```
[14]: plt.figure(figsize=(6,6))
df['work_type'].value_counts().plot.pie(autopct='%1.1f%%')
plt.title("Work Type Distribution")
plt.show()
```

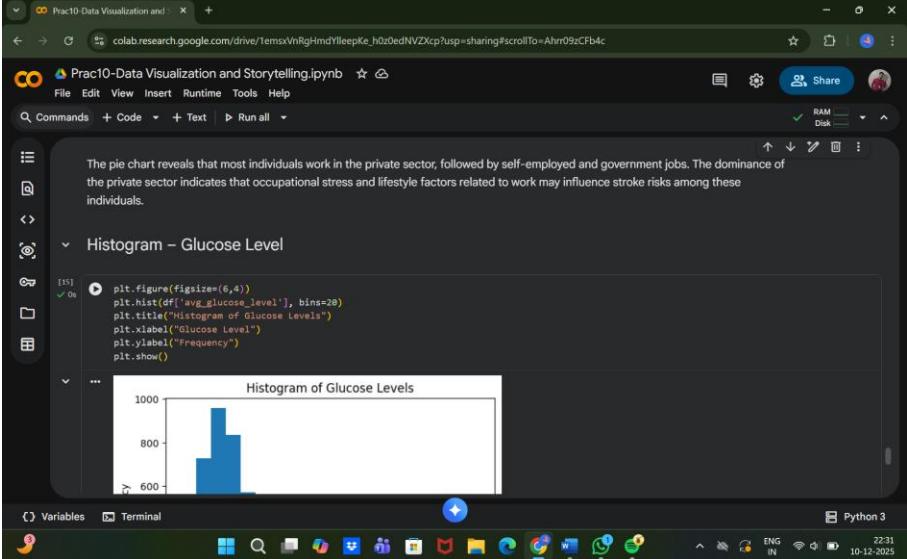
The age distribution shows a large number of middle-aged individuals but fewer elderly patients. Stroke cases in the dataset tend to cluster towards the right side (older ages). This supports the finding that stroke risk increases with age.

Pie Chart – Work Type Distribution

Work Type Distribution

Work Type	Percentage
Private	57.2%
Self-employed	16.0%
children	13.4%
Govt_job	12.9%
Never_worked	0.4%



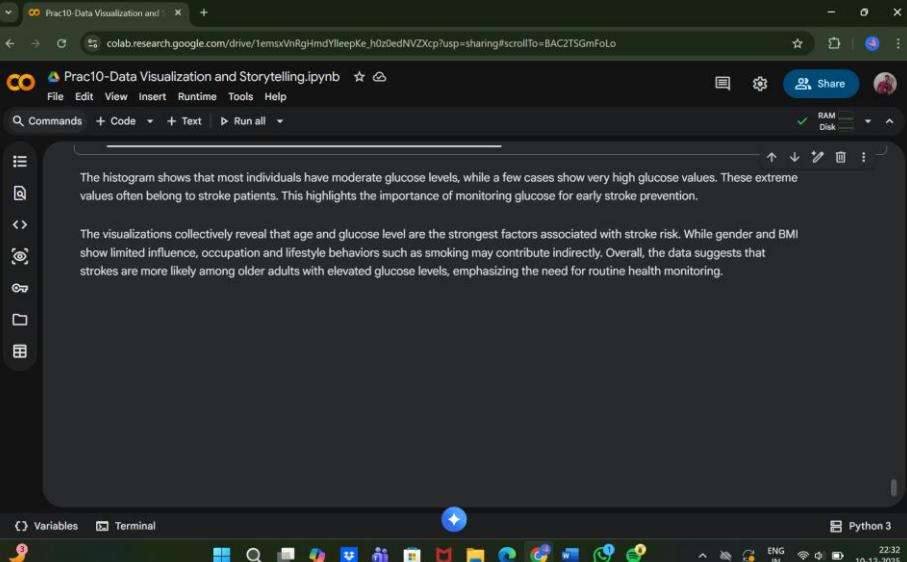
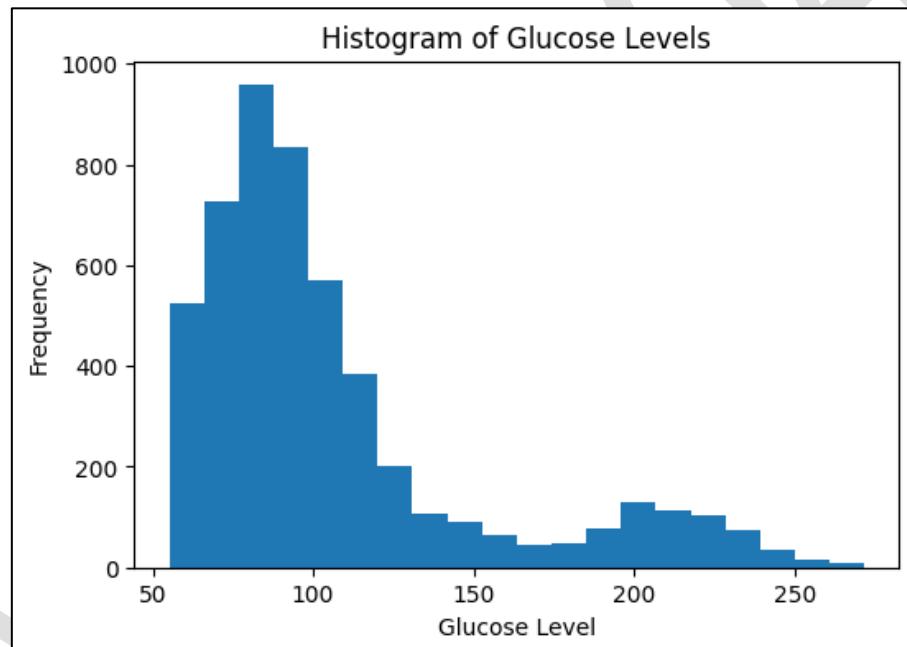


The pie chart reveals that most individuals work in the private sector, followed by self-employed and government jobs. The dominance of the private sector indicates that occupational stress and lifestyle factors related to work may influence stroke risks among these individuals.

Histogram – Glucose Level

```
[15]: plt.figure(figsize=(6,4))
plt.hist(df['avg glucose level'], bins=20)
plt.title("Histogram of Glucose Levels")
plt.xlabel("Glucose Level")
plt.ylabel("Frequency")
plt.show()
```

A histogram titled "Histogram of Glucose Levels" is displayed. The x-axis is labeled "Glucose Level" and ranges from 50 to 250. The y-axis is labeled "Frequency" and ranges from 0 to 1000. The distribution is right-skewed, with the highest frequency occurring between 75 and 85.



The histogram shows that most individuals have moderate glucose levels, while a few cases show very high glucose values. These extreme values often belong to stroke patients. This highlights the importance of monitoring glucose for early stroke prevention.

The visualizations collectively reveal that age and glucose level are the strongest factors associated with stroke risk. While gender and BMI show limited influence, occupation and lifestyle behaviors such as smoking may contribute indirectly. Overall, the data suggests that strokes are more likely among older adults with elevated glucose levels, emphasizing the need for routine health monitoring.