

```
# Install pandas and scikit-learn (if not already installed)
```

```
!pip install pandas
```

```
!pip install scikit-learn
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.2
Requirement already satisfied: numpy>=1.22.4 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/di
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist
```

```
from google.colab import files
```

```
uploaded = files.upload() # This will prompt you to upload the file
```

```
import pandas as pd
```

```
# Load the uploaded CSV file (replace 'filename.csv' with your actual file name)
```

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

```
Choose Files healthcare-...ke-data.csv
```

- **healthcare-dataset-stroke-data.csv**(text/csv) - 316971 bytes, last modified: 11/1/2024 - 100% done
Saving healthcare-dataset-stroke-data.csv to healthcare-dataset-stroke-data.csv

```
import pandas as pd
```

```
# Load the dataset
```

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

```
# Display the first few rows of the dataset
```

```
df.head()
```

```
id gender age hypertension heart_disease ever_married work_type Residence
```

0	9046	Male	67.0	0	1	Yes	Private
1	51676	Female	61.0	0	0	Yes	Self-employed
2	31112	Male	80.0	0	1	Yes	Private
3	60182	Female	49.0	0	0	Yes	Private
4	1665	Female	79.0	1	0	Yes	Self-employed

Next steps: [Generate code with df](#)

[View recommended plots](#)

[New interactive sheet](#)



0

id	int64
gender	object
age	float64
hypertension	int64
heart_disease	int64
ever_married	object
work_type	object
Residence_type	object
avg_glucose_level	float64
bmi	float64
smoking_status	object
stroke	int64

dtype: object

df.describe()



	id	age	hypertension	heart_disease	avg_glucose_level	
count	5110.000000	5110.000000	5110.000000	5110.000000	5110.000000	4909
mean	36517.829354	43.226614	0.097456	0.054012	106.147677	28
std	21161.721625	22.612647	0.296607	0.226063	45.283560	7
min	67.000000	0.080000	0.000000	0.000000	55.120000	10
25%	17741.250000	25.000000	0.000000	0.000000	77.245000	23
50%	36932.000000	45.000000	0.000000	0.000000	91.885000	28
75%	54682.000000	61.000000	0.000000	0.000000	114.090000	33
max	72940.000000	82.000000	1.000000	1.000000	271.740000	97

df.isnull().sum()



	0
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	201
smoking_status	0
stroke	0

dtype: int64

```
missing_percentage = df.isnull().mean() * 100
threshold = 30
# Filter columns with more than the threshold percentage of missing values
columns_to_drop = missing_percentage[missing_percentage > threshold].index
# Drop the columns
df = df.drop(columns=columns_to_drop)
# Display the resulting dataframe after removing columns with too many missing values
print(f"Columns dropped: {list(columns_to_drop)}")
print(f"Remaining columns: {df.columns}")
```

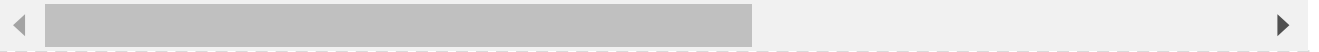


```
Columns dropped: []
Remaining columns: Index(['id', 'gender', 'age', 'hypertension', 'heart_disease', 'ever_married', 'work_type', 'Residence_type', 'avg_glucose_level', 'bmi', 'smoking_status', 'stroke'],
dtype='object')
```

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    # Indent the following line to be part of the for loop
    df[column] = label_encoder.fit_transform(df[column])
df.head()
```



	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence
0	9046	1	67.0	0	1	1	2	
1	51676	0	61.0	0	0	1	3	
2	31112	1	80.0	0	1	1	2	
3	60182	0	49.0	0	0	1	2	
4	1665	0	79.0	1	0	1	3	



Next steps:

[Generate code with df](#)[View recommended plots](#)[New interactive sheet](#)

```
# Fill missing values in columns with mode
for column in df.select_dtypes(include=['float64']).columns:
    # Indented line to be part of the for loop
    mean_value = df[column].mean() # Get the mode (most frequent value)
    df[column] = df[column].fillna(mean_value)

# Fill missing values in categorical columns with mode
for column in df.select_dtypes(include=['object']).columns:
    # Indent the following line to be part of the for loop:
    mode_value = df[column].mode()[0] # Get the mode (most frequent value)
    df[column] = df[column].fillna(mode_value)

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



	0
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

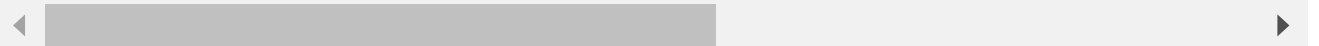
dtype: int64

df



	id	gender	age	hypertension	heart_disease	ever_married	work_type	Reside
0	9046	1	67.0	0	1	1	2	
1	51676	0	61.0	0	0	1	3	
2	31112	1	80.0	0	1	1	2	
3	60182	0	49.0	0	0	1	2	
4	1665	0	79.0	1	0	1	3	
...	...	...	...	...	...	...	...	...
5105	18234	0	80.0	1	0	1	2	
5106	44873	0	81.0	0	0	1	3	
5107	19723	0	35.0	0	0	1	3	
5108	37544	1	51.0	0	0	1	2	
5109	44679	0	44.0	0	0	1	0	

5110 rows × 12 columns



Next steps:

[Generate code with df](#)[View recommended plots](#)[New interactive sheet](#)

```
def replace_outliers_with_nan(column):
    # The following lines were not indented, causing the error
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    # Replace outliers with NaN
    return column
```

```
null_counts = df.isnull().sum() # Replace 'data' with 'df'
print("Sum of null values in each column:")
print(null_counts)
```



Sum of null values in each column:

```
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
```

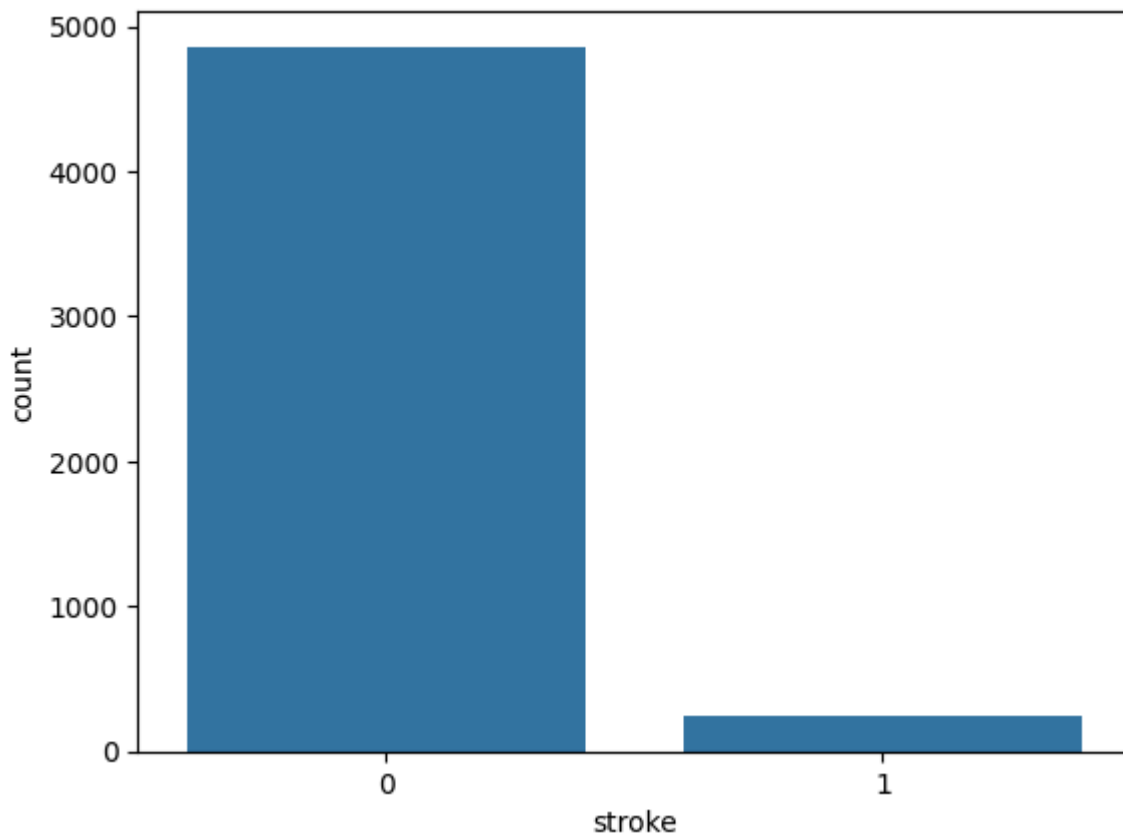
```
smoking_status    0
stroke            0
dtype: int64
```

```
# !pip install seaborn
import seaborn as sns
import matplotlib.pyplot as plt #Import matplotlib since it's used later in the code.

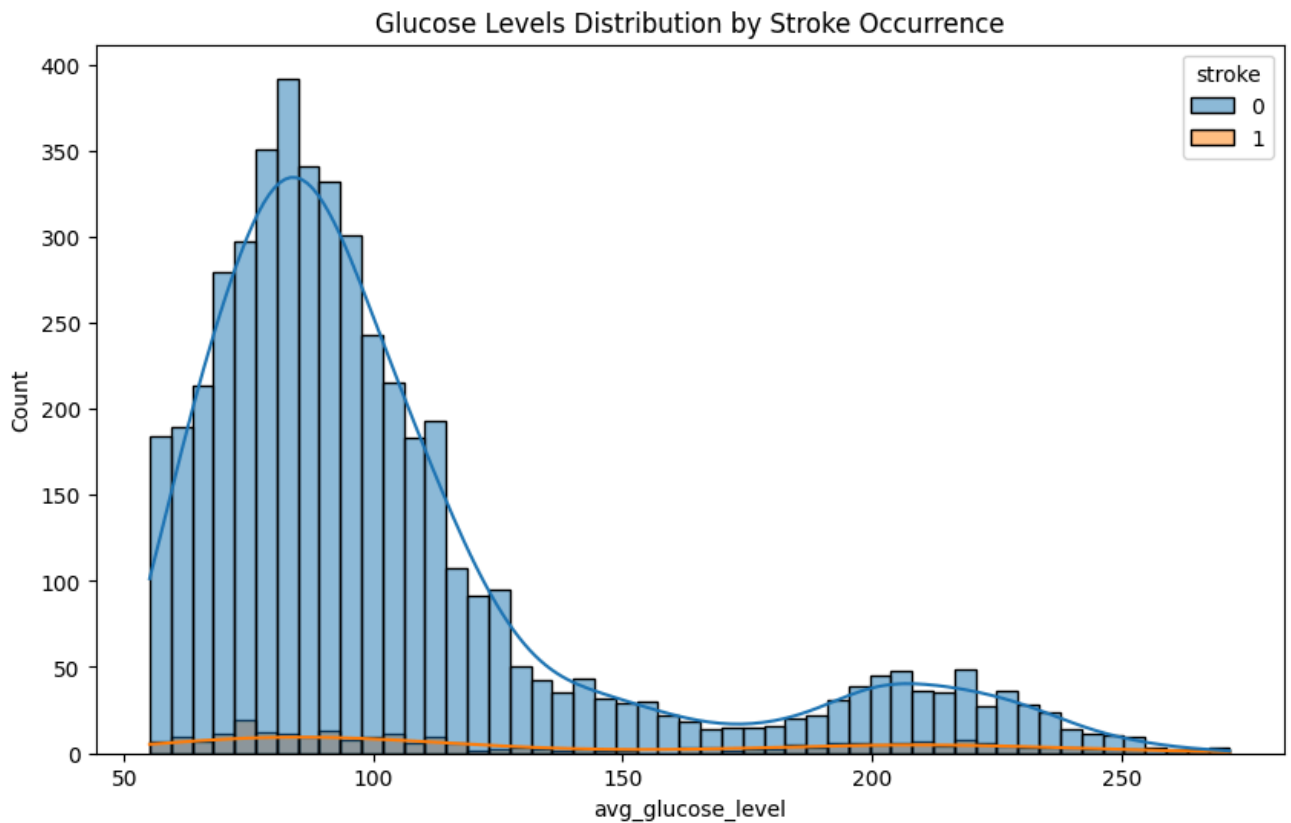
#Existing Code
df.isnull().sum()
null_counts = df.isnull().sum()
print("Sum of null values in each column:")
print(null_counts)
sns.countplot(x='stroke', data=df) #Error occurred on this line
plt.title("Stroke Distribution")
plt.show()
```

```
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (0.
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /usr/local/lib/python3.10/dist
Requirement already satisfied: pandas>=1.2 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /usr/local/lib/python3.10/d
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pack
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packag
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-pac
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-package
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packa
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
Sum of null values in each column:
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
dtype: int64
```

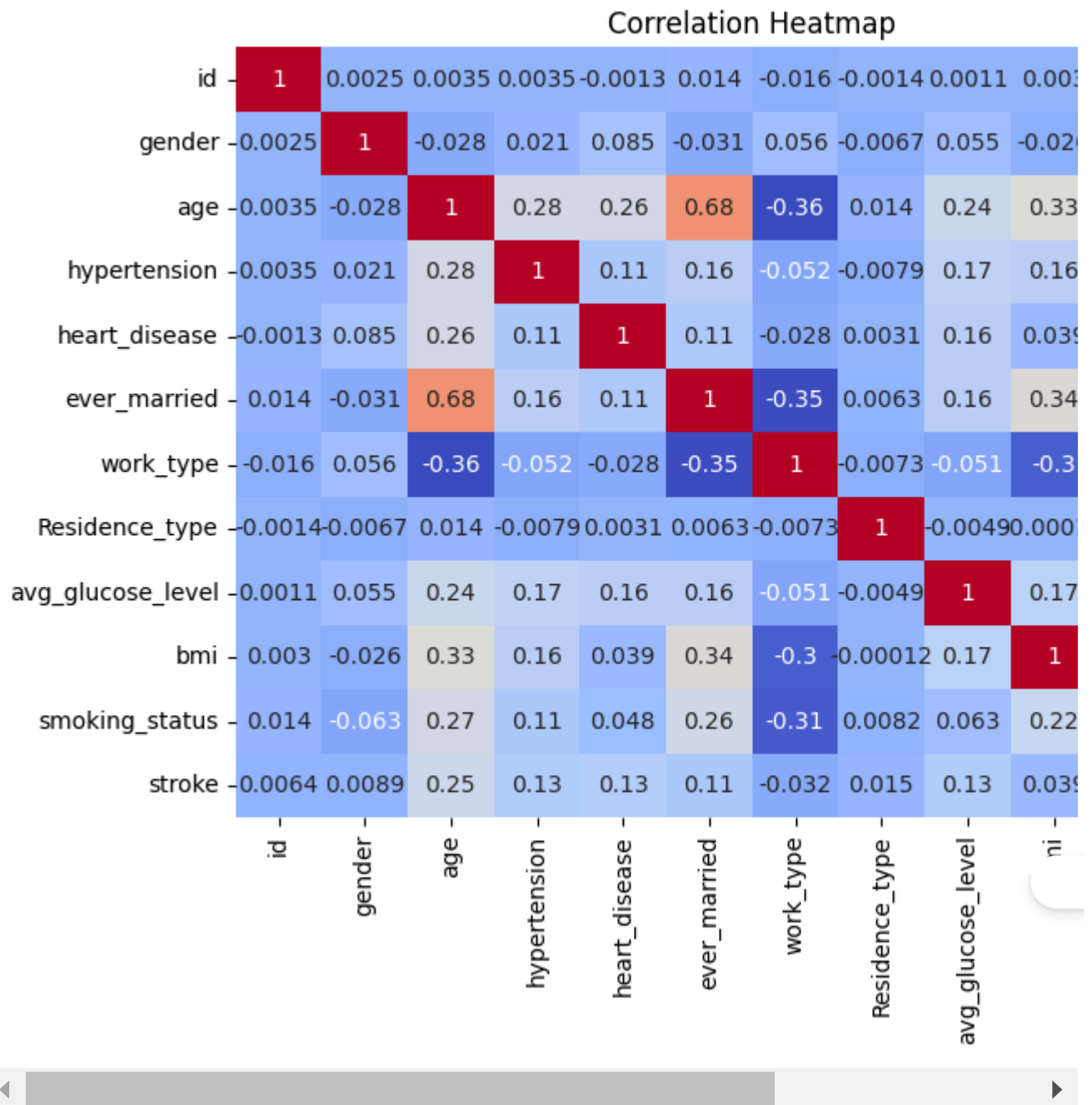
Stroke Distribution



```
plt.figure(figsize=(10,6))
sns.histplot(data=df, x='avg_glucose_level', hue='stroke', kde=True)
plt.title("Glucose Levels Distribution by Stroke Occurrence")
plt.show()
```



```
# Correlation heatmap
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```

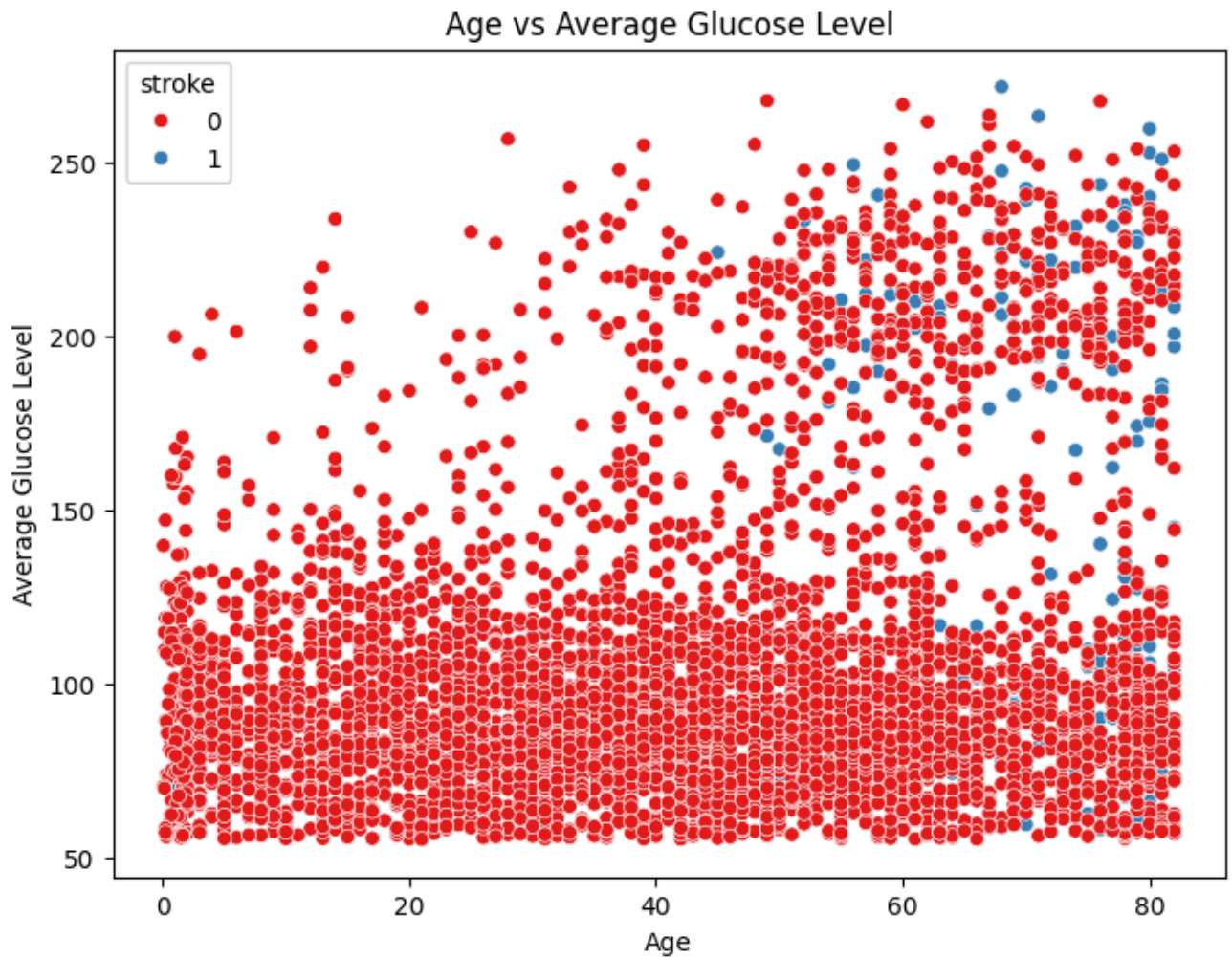


```
import matplotlib.pyplot as plt
import seaborn as sns

# Scatter plot for Age vs. Avg Glucose Level with Stroke classification
plt.figure(figsize=(8, 6))
sns.scatterplot(x='age', y='avg_glucose_level', data=df, hue='stroke', palette='Set1')

# Adding labels and title
plt.title('Age vs Average Glucose Level')
plt.xlabel('Age')
plt.ylabel('Average Glucose Level')

# Show the plot
plt.show()
```



```
import matplotlib.pyplot as plt
import seaborn as sns

# Scatter plot for Age vs BMI with Stroke classification
plt.figure(figsize=(8, 6))
sns.scatterplot(x='age', y='bmi', data=df, hue='stroke', palette='Set2')

# Adding labels and title
plt.title('Age vs BMI')
plt.xlabel('Age')
plt.ylabel('BMI')

# Show the plot
plt.show()
```

