

handling missing values

```
In [59]: import pandas as pd
import numpy as np

df = pd.read_csv('amarr.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
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

```
In [47]: df.dtypes
```

```
Out[47]: 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
```

```
In [42]: df.describe()
```

	id	age	hypertension	heart_disease	avg_glucose_level	bmi	stroke
count	5110.000000	5110.000000	5110.000000	5110.000000	5110.000000	4909.000000	5110.000000
mean	36517.829354	43.226614	0.097456	0.054012	106.147677	28.893237	0.048728
std	21161.721625	22.612647	0.296607	0.226063	45.283560	7.854067	0.215320
min	67.000000	0.080000	0.000000	0.000000	55.120000	10.300000	0.000000
25%	17741.250000	25.000000	0.000000	0.000000	77.245000	23.500000	0.000000
50%	36932.000000	45.000000	0.000000	0.000000	91.885000	28.100000	0.000000
75%	54682.000000	61.000000	0.000000	0.000000	114.090000	33.100000	0.000000
max	72940.000000	82.000000	1.000000	1.000000	271.740000	97.600000	1.000000

```
In [43]: df.isnull().sum()
```

```
Out[43]: id                0
gender              0
age                201
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
```

```
In [8]: len(df)
```

```
Out[8]: 5110
```

```
In [34]: missing_percentage = df.isnull().mean() * 100
threshold = 30
columns_to_drop = missing_percentage[missing_percentage > threshold].index
df = df.drop(columns=columns_to_drop)
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')

```
In [35]: missing_age = df.isnull().mean() * 100
columns_with_missing_30 = missing_age[missing_age > 30]
print("Columns with more than 30% missing values:")
print(columns_with_missing_30)
```

Columns with more than 30% missing values:
Series([], dtype: float64)

```
In [92]: from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for column in df.select_dtypes(include='object').columns:
    df[column] = label_encoder.fit_transform(df[column])
df.head()
```

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	1	67.0	0	1	1	2	1	228.69	36.600000	1	1
1	51676	0	61.0	0	0	1	3	0	202.21	28.893237	2	1
2	31112	1	80.0	0	1	1	2	0	105.92	32.500000	2	1
3	60182	0	49.0	0	0	1	2	1	171.23	34.400000	3	1
4	1665	0	79.0	1	0	1	3	0	174.12	24.000000	2	1

```
In [40]: for column in df.select_dtypes(include='float64').columns:
mean_value = df[column].mean()
df[column] = df[column].fillna(mean_value)
```

```
In [42]: for column in df.select_dtypes(include='object').columns:
mode_value = df[column].mode()[0]
df[column] = df[column].fillna(mode_value)
```

```
In [83]: df.isnull().sum()
```

```
Out[83]: 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
```

```
In [43]: df
```

	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	36.600000	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	28.893237	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.500000	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.400000	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.000000	never smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	28.893237	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.000000	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.600000	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.600000	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_Job	Urban	85.28	26.200000	Unknown	0

5110 rows x 12 columns

Handling outliers

```
In [53]: def replace_outliers_with_nan(column):
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

return column.where((column >= lower_bound) & (column <= upper_bound), np.nan)

for col in df.select_dtypes(include=[np.number]).columns:
    df[col] = replace_outliers_with_nan(df[col])
```

```
In [54]: null_counts = df.isnull().sum()
print("Sum of null values in each column:")
print(null_counts)
```

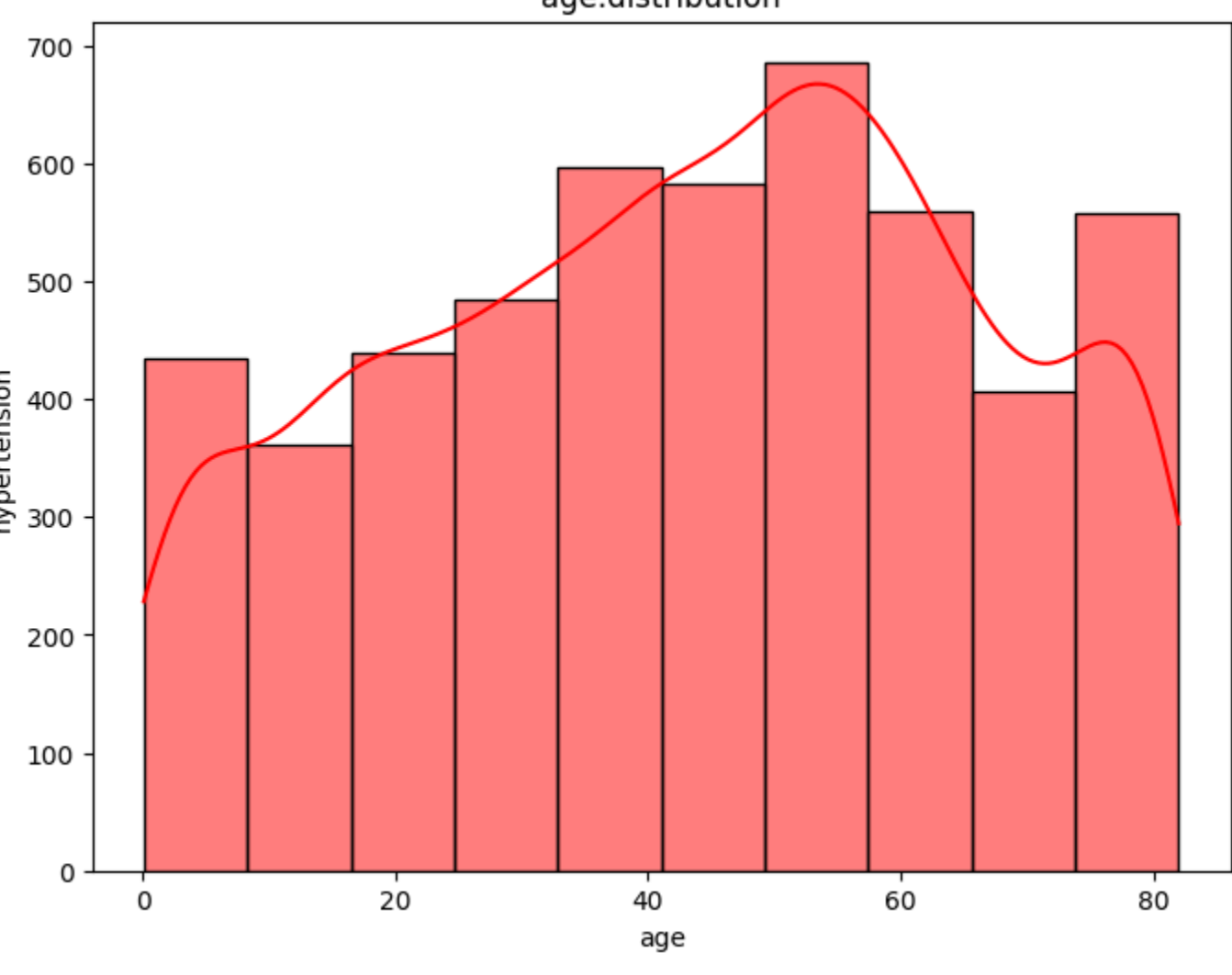
Sum of null values in each column:
id 0
gender 0
age 0
hypertension 498
heart_disease 276
ever_married 0
work_type 0
Residence_type 0
avg_glucose_level 765
bmi 147
smoking_status 0
stroke 249
dtype: int64

```
In [26]: import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv('amarr.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
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

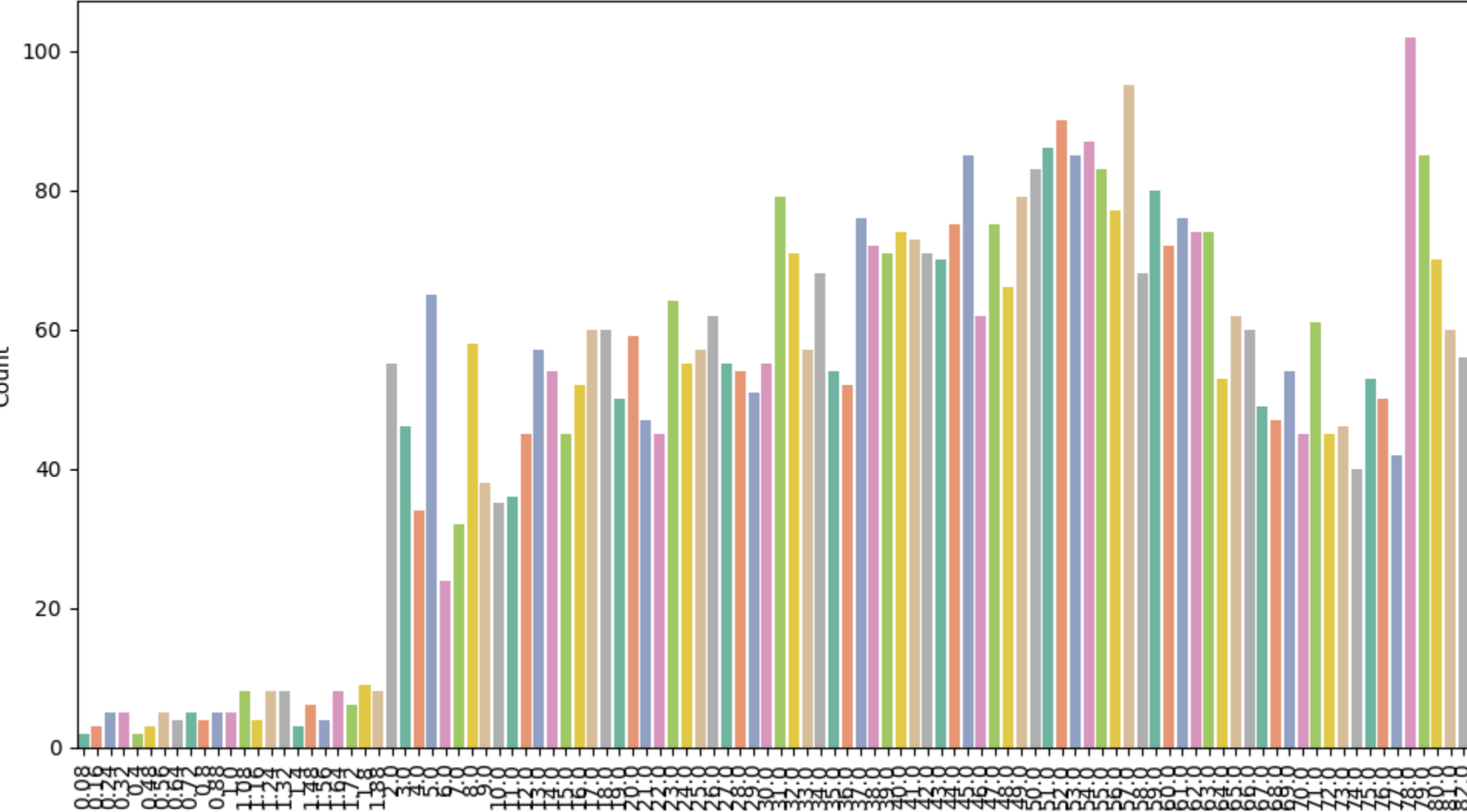
```
In [20]: plt.figure(figsize=(8,6))
sns.histplot(df['age'], bins=10, kde=True, color='red')
```

age.distribution



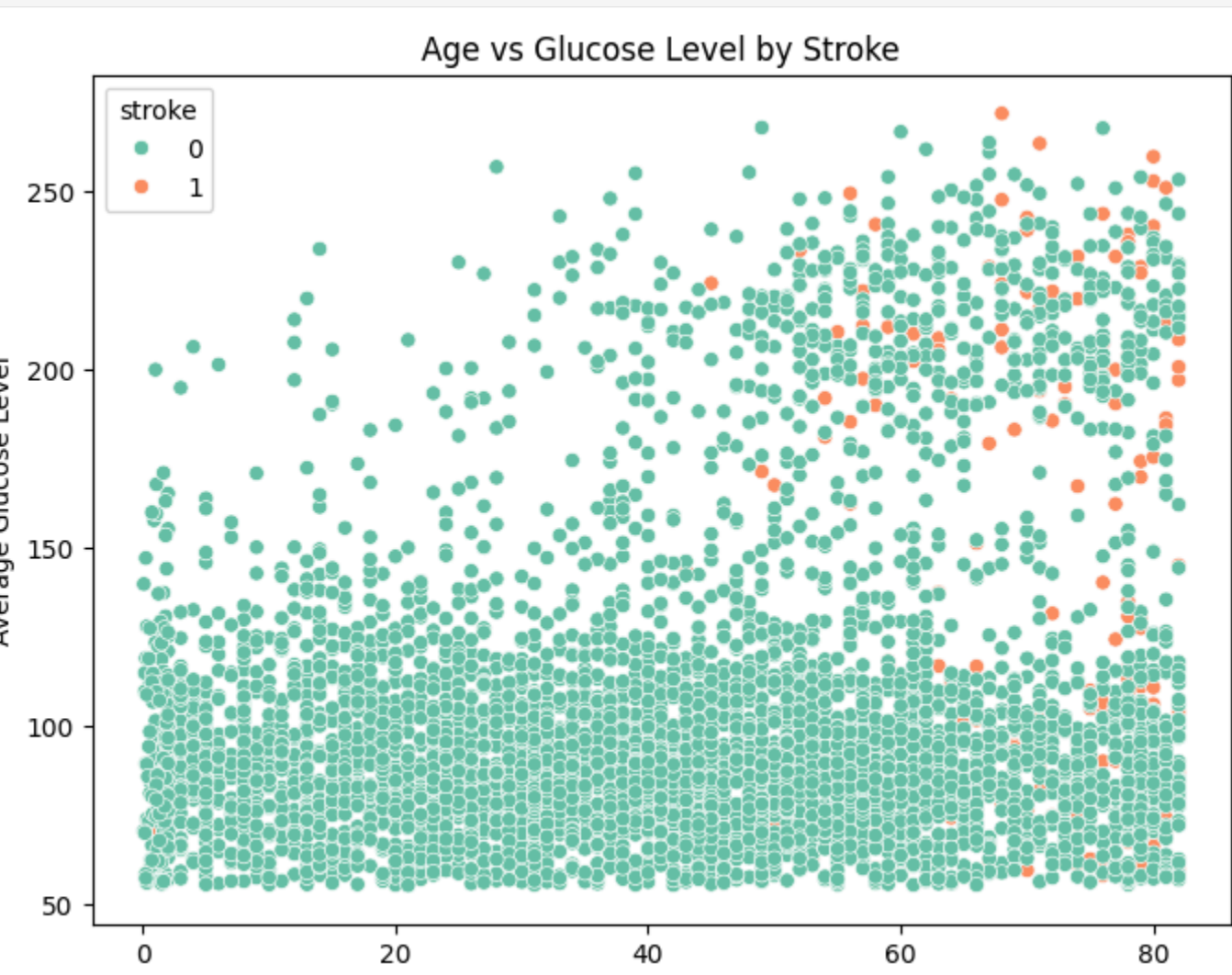
```
In [55]: plt.figure(figsize=(10, 6))
sns.countplot(x='age', data=df, palette='Set2')
```

Age Distribution



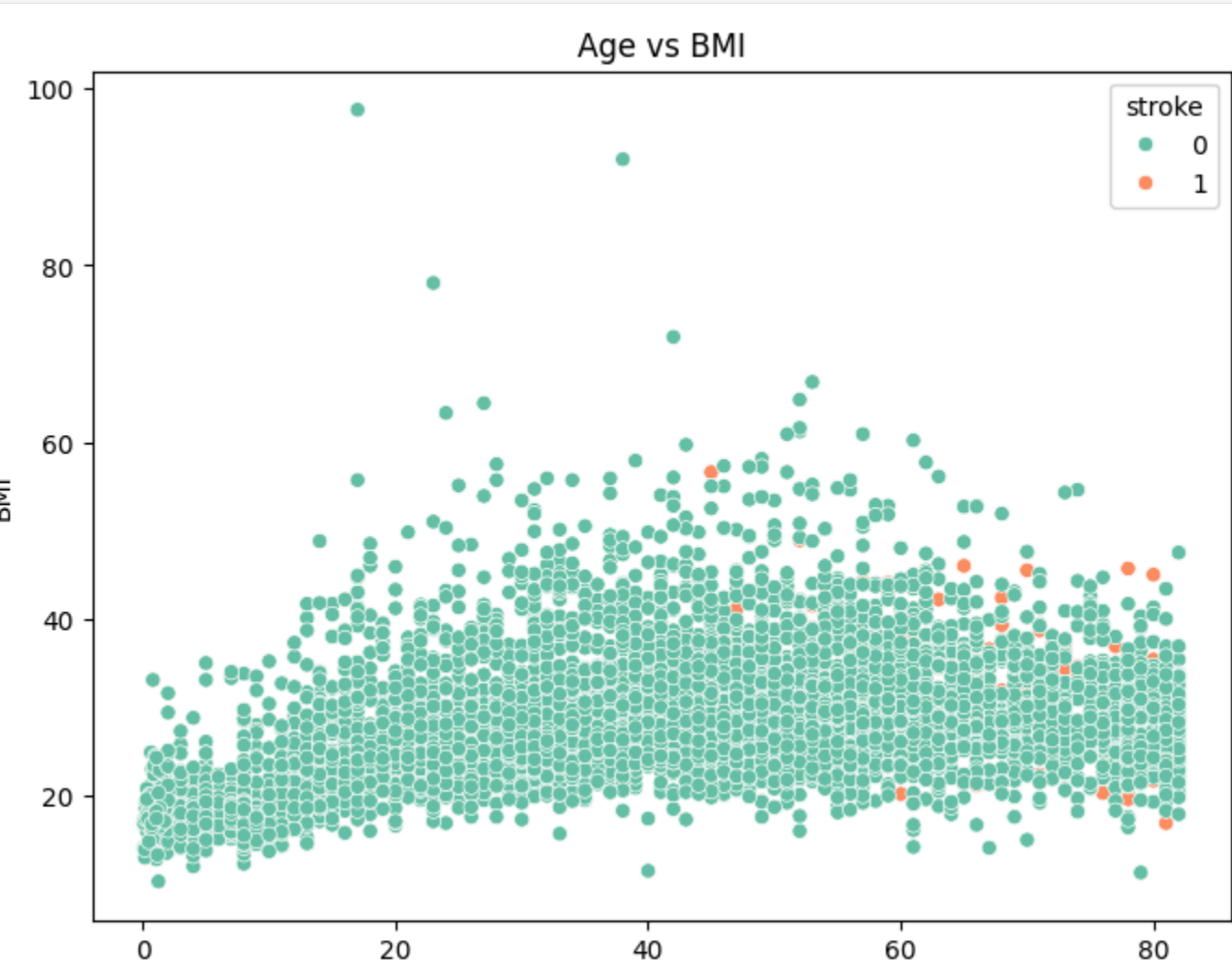
```
In [27]: plt.figure(figsize=(8,6))
sns.scatterplot(x='age', y='avg_glucose_level', data=df, hue='stroke', palette='Set2')
```

Age vs Glucose Level by Stroke



```
In [33]: plt.figure(figsize=(8, 6))
sns.scatterplot(x='age', y='bmi', data=df, hue='stroke', palette='Set2')
```

Age vs BMI



```
In [31]: plt.figure(figsize=(8, 6))
sns.boxplot(x='stroke', y='bmi', data=df, palette='Set1')
```

BMI Levels by Stroke Classification

