

Project 1

Fatima Alkaff

```
jupyter Project Last Checkpoint: 1 minute ago
```

```
File Edit View Run Kernel Settings Help Trusted
```

```
Python [conda env:Practical]
```

```
[2]: import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
```

```
[4]: 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
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1

```
[6]: df.dtypes
```

```
[6]: 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
```

```
jupyter Project Last Checkpoint: 2 minutes ago
```

```
File Edit View Run Kernel Settings Help Trusted
```

```
Python [conda env:Practical]
```

```
[8]: df.isnull().sum()
```

```
[8]: 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
```

```
[10]: 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')
```

```
[12]: 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()
```

Jupyter Project Last Checkpoint: 2 minutes ago

File Edit View Run Kernel Settings Help Trusted

JupyterLab Python [conda env:Practical]

```
df.head()
```

```
[12]:
```

	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	36.6	1	1
1	51676	0	61.0	0	0	1	3	0	202.21	NaN	2	1
2	31112	1	80.0	0	1	1	2	0	105.92	32.5	2	1
3	60182	0	49.0	0	0	1	2	1	171.23	34.4	3	1
4	1665	0	79.0	1	0	1	3	0	174.12	24.0	2	1

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

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

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

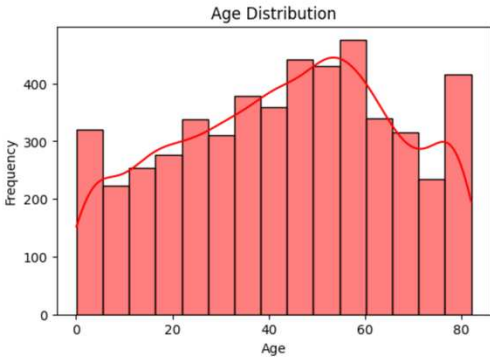
```
[14]: id                0
      gender          0
      age             0
      hypertension    0
      heart_disease   0
      ever_married    0
      work_type       0
      Residence_type  0
      avg_glucose_level 0
      bmi             0
      smoking_status  0
      stroke          0
      dtype: int64
```

Jupyter Project Last Checkpoint: 3 minutes ago

File Edit View Run Kernel Settings Help Trusted

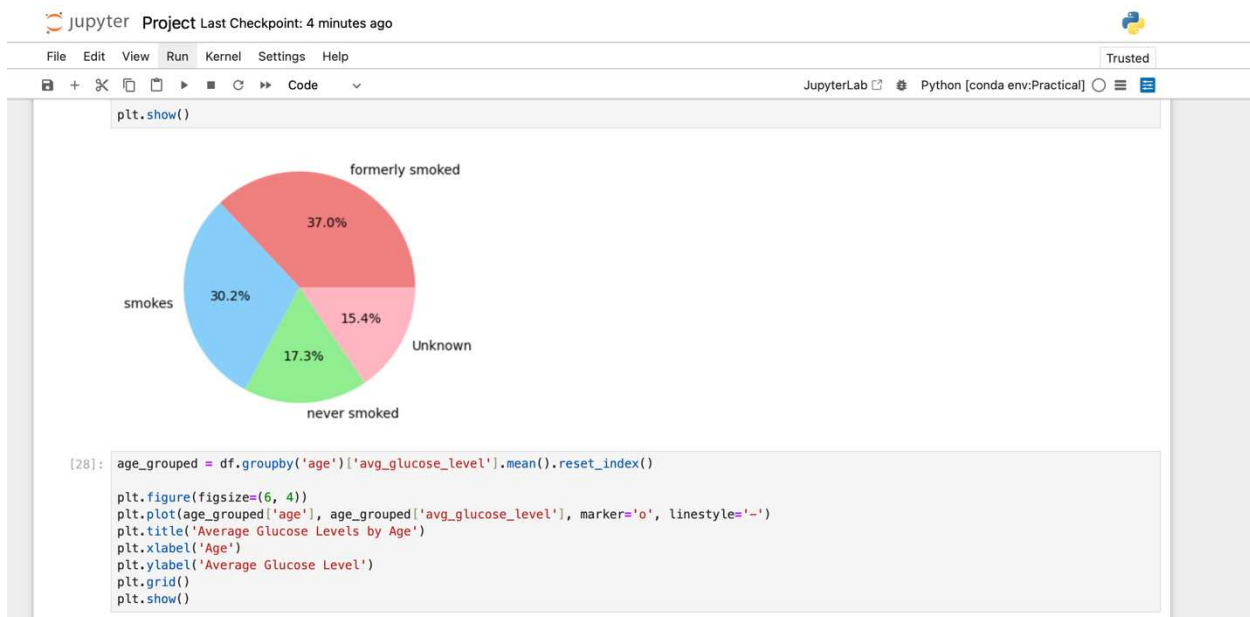
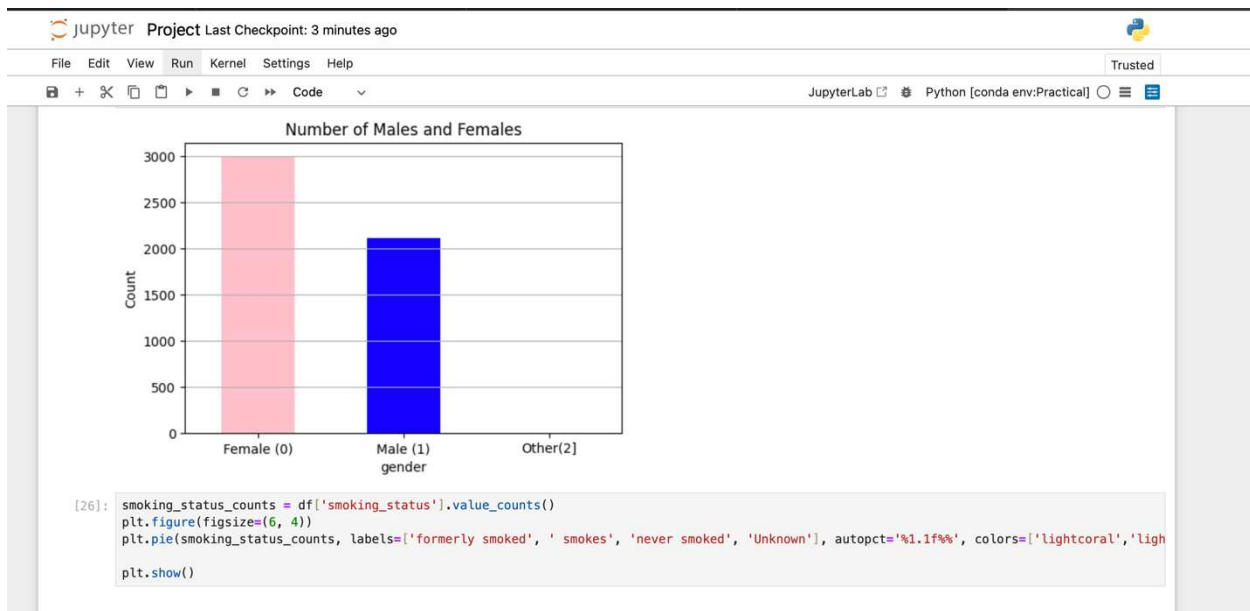
JupyterLab Python [conda env:Practical]

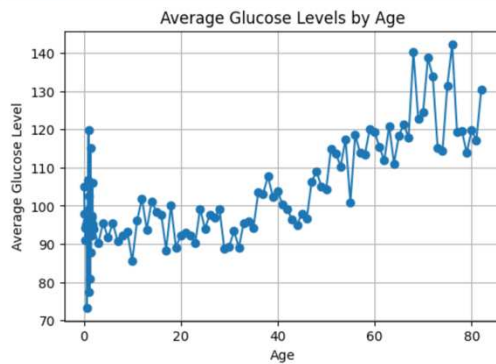
```
[16]: plt.figure(figsize=(6, 4))
      sns.histplot(df['age'], bins=15, kde=True, color='red')
      plt.title('Age Distribution')
      plt.xlabel('Age')
      plt.ylabel('Frequency')
      plt.show()
```



```
[24]: gender_counts = df['gender'].value_counts()

plt.figure(figsize=(6, 4))
gender_counts.plot(kind='bar', color=['pink', 'blue', 'green'])
plt.xticks([0, 1, 2], ['Female (0)', 'Male (1)', 'Other(2)'], rotation=0)
plt.title('Number of Males and Females')
plt.xlabel('gender')
plt.ylabel('Count')
plt.grid(axis='y')
plt.show()
```





```
[30]: plt.figure(figsize=(6, 4))
sns.scatterplot(x='age', y='avg_glucose_level', data=df, hue='gender', palette='Set1')
plt.title('Age vs Glucose')
plt.xlabel('Age')
plt.ylabel('Glucose')
plt.show()
```

Age

```
[30]: plt.figure(figsize=(6, 4))
sns.scatterplot(x='age', y='avg_glucose_level', data=df, hue='gender', palette='Set1')
plt.title('Age vs Glucose')
plt.xlabel('Age')
plt.ylabel('Glucose')
plt.show()
```

