**Project 1 : Stroke Prediction Analysis**

**Source Code :**

```
#Importing libraries
import pandas as pd
#Importing files
from google.colab import files
#Uploading files
uploaded = files.upload()
#Reading files
df = pd.read_csv("Stroke.csv")
#Printing the head
df.head()

#Finds the information of the columns
df.info()

#Find out the size
df.size

#Find out the shape
df.shape

#Drop the NaN values
df.dropna

#Find out the numerical columns
numerical_cols = df.columns[df.dtypes != object]
numerical_cols

#Find out the categorical columns
categorical_cols = df.columns[df.dtypes == object]
categorical_cols

#Find out the uniqe values of a column
df['work_type'].unique()

#Find out the correlation of the "stroke" column with other columns
correlation_matrix = df.corr()
correlation_matrix["stroke"]

#Dropping some columns
```

```
stroke_labels = df["stroke"].copy()
df_drop = df.drop(["work_type","Residence_type", "ever_married"],axis = 1)
df_drop.head()

#Applying One-Hot-Encoder
from sklearn.preprocessing import OneHotEncoder
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['age']])
print(result)

from sklearn.preprocessing import OneHotEncoder
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['smoking_status']])
print(result)

from sklearn.preprocessing import OneHotEncoder
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['heart_disease']])
print(result)

from sklearn.preprocessing import OneHotEncoder
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['gender']])
print(result)

from sklearn.preprocessing import OneHotEncoder
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['avg_glucose_level']])
print(result)

import matplotlib.pyplot as plt

# Read the CSV file into a DataFrame
df = pd.read_csv('Stroke.csv')  # Replace 'your_uploaded_file.csv' with the actual file name

# Plot a histogram
plt.hist(df['stroke'], bins=10, color='blue', edgecolor='black')

# Add labels and title
plt.xlabel('Data Values')
plt.ylabel('Frequency')
plt.title('Histogram of Data')

# Show the plot
plt.show()

#Find out the percentage population with hypertension. 90% people has no hypertension issue.
df.groupby('hypertension')['hypertension'].count().apply(lambda x: x* 100/ len(df))

# Choose the specific column you want to plot
column_to_plot = 'hypertension'

# Plot the data
plt.figure(figsize=(10, 6))
plt.plot(df[column_to_plot])
plt.title('Plot of hypertension')
plt.xlabel('Index')
plt.ylabel(column_to_plot)
plt.grid(True)
plt.show()

#Find out the percentage population with heart diseases. Almost 95% has no heart disease.
df.groupby('heart_disease')['heart_disease'].count().apply(lambda x: x* 100/ len(df))

column_to_plot = 'heart_disease'

# Plot the data
plt.figure(figsize=(10, 6))
plt.plot(df[column_to_plot])
plt.title('Plot of heart disease')
plt.xlabel('Index')
plt.ylabel(column_to_plot)
plt.grid(True)
plt.show()

#Find out the percentage of occuring stroke
df.groupby('stroke')['stroke'].count().apply(lambda x: x* 100/ len(df))
```

```
#Plotting the data for stroke
column_to_plot = 'stroke'

# Plot the data
plt.figure(figsize=(10, 6))
plt.plot(df[column_to_plot])
plt.title('Plot of stroke')
plt.xlabel('Index')
plt.ylabel(column_to_plot)
plt.grid(True)
plt.show()

#Plotting the categorical features
categorical_columns = df.select_dtypes(include=['object']).columns

# Plot data for each categorical column
for column in categorical_columns:
    plt.figure(figsize=(10, 6))
    df[column].value_counts().plot(kind='bar', color='skyblue')
    plt.title(f'Distribution of {column}')
    plt.xlabel(column)
    plt.ylabel('Count')
    plt.show()

#Applying BiVariate Analysis
import seaborn as sns
# Example: Scatter plot for two numerical variables
sns.scatterplot(x='hypertension', y='stroke', data=df)
plt.title('Bivariate Analysis : hypertension vs stroke')
plt.show()

#Find out which age group has more heart disease
sns.histplot(x ='age', data=df, bins = 10, hue = 'heart_disease', multiple="stack")

#Find out which age group has more hypertension
sns.histplot(x ='age', data=df, bins = 10, hue = 'hypertension', multiple="stack")

#Find out which age group has high glucose level
sns.lineplot(x ='age', data=df, y = 'avg_glucose_level')
```

```
#Find out which age group has high BMI
sns.lineplot(x ='age', data=df, y = 'bmi')

#Applying Pearson Correlation
corr = df.iloc[:, 1:].corr()
corr

#Encoding the categorical columns
categorical_cols

#Plotting correlation in graph
plt.figure(figsize=(20,15))
sns.heatmap(corr, vmin=-1, vmax=1, annot=True)
```

**Project Explnation :**

This code is a data analysis and visualization script written in Python, utilizing several libraries and frameworks. Let's break down the code step by step :

1. **Importing Libraries :**

```
import pandas as pd
from google.colab import files
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import OneHotEncoder
```

**Explanation :**

- **pandas** : Used for data manipulation and analysis.
- **files** from **google.colab** : Used for uploading files in a Google Colab environment.
- **matplotlib.pyplot** : Used for creating visualizations like histograms and line plots.
- **seaborn** : A statistical data visualization library based on Matplotlib.
- **OneHotEncoder** from **sklearn.preprocessing** : Used for one-hot encoding categorical features.

2. **File Upload and Data Reading :**

```
uploaded = files.upload()
df = pd.read_csv("Stroke.csv")
```

**Explanation :**

- **files.upload()** : Allows users to upload files in a Google Colab environment.
- **pd.read_csv("Stroke.csv")** : Reads a CSV file named "Stroke.csv" into a Pandas DataFrame called **df**.

3. **Basic DataFrame Operations :**

```
df.head()
df.info()
df.size
df.shape
df.dropna
numerical_cols = df.columns[df.dtypes != object]
categorical_cols = df.columns[df.dtypes == object]
```

**Explanation :**

- **df.head()** : Displays the first few rows of the DataFrame.
- **df.info()** : Provides information about the DataFrame, including data types and missing values.
- **df.size** : Returns the total number of elements in the DataFrame.
- **df.shape** : Returns the dimensions (rows, columns) of the DataFrame.
- **df.dropna** : This seems to be an error. It should be **df.dropna()** to actually drop NaN values.
- **numerical_cols** : Extracts columns with non-object (numerical) data types.
- **categorical_cols**: Extracts columns with object (categorical) data types.

4. **One-Hot Encoding :**

```
onehot = OneHotEncoder()
result = onehot.fit_transform(df[['age']])
```

**Explanation :**

- One-hot encodes specific columns (**'age'**, **'smoking_status'**, **'heart_disease'**, **'gender'**, **'avg_glucose_level'**) using **OneHotEncoder** from scikit-learn.

5. **Histogram Plotting :**

```
plt.hist(df['stroke'], bins=10, color='blue', edgecolor='black')
plt.xlabel('Data Values')
plt.ylabel('Frequency')
plt.title('Histogram of Data')
plt.show()
```

**Explanation :**

- Plots a histogram of the 'stroke' column with 10 bins using Matplotlib.

6. **Grouping and Percentage Calculations :**

```
df.groupby('hypertension')['hypertension'].count().apply(lambda x: x * 100 / len(df))
```

**Explanation :**

- Groups the DataFrame by 'hypertension' and calculates the percentage of each group.

7. **Line Plots and Bar Charts :**

```
sns.lineplot(x='age', data=df, y='avg_glucose_level')
df[column].value_counts().plot(kind='bar', color='skyblue')
```

**Explanation :**

- Utilizes Seaborn and Matplotlib for line plots and bar charts based on specific columns.

8. **Scatter Plots and Heatmaps :**

```
sns.scatterplot(x='hypertension', y='stroke', data=df)
sns.heatmap(corr, vmin=-1, vmax=1, annot=True)
```

**Explanation :**

- Uses Seaborn for scatter plots and a heatmap of the correlation matrix.

This code showcases data analysis and visualization techniques using popular Python libraries and frameworks, including Pandas, Matplotlib, Seaborn, and scikit-learn. The programming language used is Python.