read-me

Group No. 1
BIO545 - Mini Project

1. Purpose

Our web application aims to simplify the process of selecting the optimal statistical test for a given dataset. We achieve this by processing the user's input data and analyzing these results, combined with the user inputs, to navigate a decision tree, ultimately providing the optimal statistical test tailored to the user's input data. By leveraging a decision tree approach, the application aims to streamline the decision-making process in selecting the appropriate statistical test. This can save users time and effort in determining the most suitable test for their specific data and research questions.

2. Features

Our application simplifies the process of determining the optimal statistical analyses choice for the specific input data and user inputs:

The features of our Web applications are:

- **CSV Input**: Users can upload CSV files containing their data.
- **Data Analysis**: The website automatically checks the normality and homogeneity of the data or target column provided by the user.
- **User Interaction**: The website prompts users with questions about their requirements, presumably to understand the context or purpose of the analysis.
- **Statistical Test Recommendations**: Based on the analysis of the data and user input, the website suggests appropriate statistical tests to be used.

3. Using the Web Application

3.1 Dependencies

Frontend: HTML, CSS and Javascript
Backend: Flask framework (python)

Dependencies:

- **pandas:** A powerful data manipulation library for Python.
- **numpy:** A fundamental package for scientific computing with Python.
- **scipy:** A library used for scientific and technical computing.
- **Flask**: A lightweight WSGI web application framework for Python.
- **Flask**: The main Flask module.
- **render_template:** A function for rendering Jinja2 templates.
- **request:** An object representing the current request.
- **redirect:** A function for performing HTTP redirects.
- **url_for:** A function for generating URLs for Flask routes.
- **session:** A session object for storing user data across requests.
- **jsonify:** A function for creating a JSON response.

- **seaborn:** A Python visualization library based on matplotlib, providing a high-level interface for drawing attractive statistical graphics.
- **matplotlib.pyplot:** A plotting library for creating static, animated, and interactive visualizations in Python.
- **io:** A module providing Python's main facilities for dealing with various types of I/O.
- **base64**: A module providing functions for encoding and decoding data using Base64 encoding.

Note: Make sure to install these dependencies using *pip* before running the code.

3.2 Setting-up the Web Application

After installing all the previously mentioned modules, execute the following command:

```
python app.py
```

After running the command, navigate to the link provided in the terminal, which is typically http://127.0.0.1:5000, to start using the application.

3.3 Input files and interactive approach

We have opted for taking a **csv input** and performing statistical analyses on the backend to reduce the number of questions the user has to answer. We have also opted for a multiple choice structure to make the process faster and less prone to errors.