Visualize Space Science

High-level summary

Objective

Major Requirement of the Project: Development of Interactive Web Application

The ability to read in input URLs for datasets, clean the datasets, and produce effective visualizations of data is enabled by this application. Additionally, the application produces a 3D model of a rocket and communicates with the user using a friendly interface.

Key Components

User Input for Datasets

Inputting the URL for the dataset that you need is allowed in this application. This input will serve as the starting point for data analysis.

Data Cleaning and Processing:

As soon as the application receives the URL of the dataset, it fetches the data set and performs some basic cleaning and preprocessing operations before proceeding with any analytical operation.

It includes dealing with missing values, normalization, and filtering of data from the concerned dataset to suit the analysis.

Data Visualization:

After data cleansing, the application creates visualizations that meaningfully represent the data. This is done through charts, graphs, and the use of other visual aids to make it possible for users to interpret trends, patterns, and insights that may exist in the dataset.

3D Rocket Model:

Another feature introduced in the web application is a 3D model of a rocket. The model provides visual interest that makes the website user-friendly; it ties well with the space exploration themes.

Web Development

The basic website was built using HTML, CSS, and JavaScript. This includes;

Input Box: Users input the URL where the dataset is held

Submit Button: This works to activate the workflow for data retrieval and processing.

This 3D model integrates into the web site to create an interactive interface.

Flask Backend Integration:

Flask backend (app.py): This is the driving force behind this web application, which handles data retrieval through a given URL and manages workflow concerning cleaning and visualization of data. It produces output.

This project successfully delivers a web application to the users for a submission of dataset URLs for cleaning, which visualizes the results. The integration of a 3D rocket model also increases user engagement while valuable insights are still extracted from the streamlined data processing workflow.

Applications:

Various stakeholders, such as researchers, students, and data enthusiasts, can make use of the platform to provide an arena for hands-on exploration and analysis of datasets in specific fields of space and scientific research.

Final Project Details

This project will require a web application that would accept URLs of different datasets that users could input into the application to clean up the data and later visualize it. A 3D rocket model is used to add some aesthetic appeal to the user interface.

Project Goals

Input: Accept dataset URLs from users.
Process: Cleaning and preprocessing of datasets.

Visualization: Interactive visualization of the cleaned data.
User Interaction: The site should have a 3D rocket model.

Datasets Employed:

- Compulsory Datasets:
  OSD-379
  OSD-665
- Supplementary Datasets:
  OSD-678
  OSD-702
  OSD-516
  21 others from NASA's open-source

Design Phase:

1. User Interface:
   a. Dataset URL input box
   b. Submit button to process the dataset
2. Data Processing:
   a. Fetch the dataset from the URL
   b. Clean dataset using Python (Pandas)
   c. Data Visualization:

Use libraries such as Matplotlib or Plotly to make data visualizations

1. Technical Implementation:
   a. Technologies Applied:
      i. Front-end: HTML, CSS, JavaScript
      ii. Back-end: Flask (Python), Pandas, Matplotlib/Plotly

```
@app.route('/submit', methods=['POST'])
def submit():
    dataset_url = request.form['dataset_url']
    data = pd.read_csv(dataset_url)  # Load dataset from URL
    cleaned_data = clean_data(data)  # Clean dataset
    img = visualize_data(cleaned_data)  # Generate visualization
    return render_template('results.html', plot_url=img)
```

Project breakdowns

Technical Stack:

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** Flask (Python), Pandas, Matplotlib/Plotly
- **3D Model Rendering:** Three.js or similar library

Application Workflow:

1. **User Input:**
    - Users enter dataset URLs via an HTML form.
2. **Data Processing:**
    - Flask backend receives the URL and loads the dataset.
    - The dataset is cleaned using a defined data cleaning function.
3. **Data Visualization:**
    - Cleaned data is visualized using Matplotlib/Plotly.
    - Results are displayed back to the user.

Key Code Components:

- **Flask Route for Submission:**
    - Handles user requests and data processing.

- **Data Cleaning Function:**
    - Cleans the input dataset (e.g., removes NaN values, duplicates).
- **Visualization Function:**
    - Generates plots based on cleaned data.

User Experience:

- Engaging homepage with a 3D rocket model.
- Simple input form for dataset URLs.
- Interactive results page displaying visualizations.

Space agency data used

- Compulsory Datasets:
  OSD-379
  OSD-665
- Supplementary Datasets:
  OSD-678
  OSD-702
  OSD-516
  21 others from NASA's open-source

References

1. https://www.spaceappschallenge.org/nasa-space-apps-2024/challenges/
2. https://www.spaceappschallenge.org/nasa-space-apps-2024/challenges/visualize-space-science/
3. https://www.spaceappschallenge.org/nasa-space-apps-2024/challenges/visualize-space-science/?tab=resources
4. https://osdr.nasa.gov/bio/repo/data/studies/OSD-379
5. https://osdr.nasa.gov/bio/repo/data/studies/OSD-665
6. https://osdr.nasa.gov/bio/repo/data/studies/OSD-678
7. https://osdr.nasa.gov/bio/repo/data/studies/OSD-516
8. https://huggingface.co/docs/evaluate/en/package_reference/visualization_methods

9. https://medium.com/@sooryanarayan_5231/end-to-end-machine-learning-deployment-from-model-training-to-web-service-integration-using-flask-4263d96b9479
10. https://youtu.be/wSR2SsRsDq0?si=BZxBtnLJXfSaOZ6_
11. https://www.geeksforgeeks.org/k-means-clustering-introduction/
12. https://scikit-learn.org/dev/modules/generated/sklearn.preprocessing.LabelEncoder.html