10 Academy: Artificial Intelligence Mastery

Week 0 Challenge Document

Date: 06 December - 08 December 2024

Table of Contents

Challenge Overview

This week's challenge is focused on understanding, exploring, and analyzing solar farm data found in Benin, Sierra Leone, and Togo. The challenge aims to evaluate candidates for the 12-week training program in Data Engineering (DE), Financial Analytics (FA), and Machine Learning Engineering (MLE).

Applicants who proceed to the next level by demonstrating sufficient performance in this week's challenge will have a clear picture of the required discipline, resilience, proactivity, talent diversity, and other essential elements of the 10 Academy training. Those who can not make it to the limited spots available will gain a clear understanding of the direction they should improve to prepare for FA, DE & MLE job positions in the future. Everyone will gain project experience to showcase in their professional profile.

This week, therefore, is a **win-win** for everyone. We advise you to put your best effort into completing as many tasks as possible. We know that the number of tasks you are required to complete is a lot, and you **will not** have time to build intuition or to be comfortable with the new concepts and skills you are exposed to with this week's challenge. Please **note that** building a deeper understanding is not the purpose for this week's project. Moreover, you may have never done or attempted to do some of the tasks before this training. If you are confused and **overwhelmed**, know that it **is expected**.

The tutors, community managers, and all other teams are there to support you as best as they can. Be proactive in asking questions, provide resources that may help others, and above all **persist**!

Business Objective

MoonLight Energy Solutions aims to develop a strategic approach to significantly enhance its operational efficiency and sustainability through targeted solar investments. As an Analytics Engineer at MoonLight Energy Solutions, your task is to perform a quick analysis of an environmental measurement provided by the engineering team and translate your observation as a strategy report. Your analysis should focus on identifying key trends and learn valuable insights that will support your data-driven case - your recommendation based on the statistical analysis and EDA. In particular, your analysis and recommendation must present a strategy focusing on identifying high-potential regions for solar installation that align with the company's long-term sustainability goals. Your report should provide an insight to help realize the overarching objectives of MoonLight Energy Solutions.

Dataset Overview

Solar Radiation Measurement Data

The data for this week's challenge is extracted and aggregated from Solar Radiation Measurement Data. Each row in the data contains the values for solar radiation, air temperature, relative humidity, barometric pressure, precipitation, wind speed, and wind direction, cleaned and soiled radiance sensor (soiling measurement) and cleaning events.

. The structure of the data is as follows

- **Timestamp (yyyy-mm-dd hh:mm)**: Date and time of each observation.
- **GHI (W/m^2)**: Global Horizontal Irradiance, the total solar radiation received per square meter on a horizontal surface.
- **DNI (W/m^2)**: Direct Normal Irradiance, the amount of solar radiation received per square meter on a surface perpendicular to the rays of the sun.
- **DHI (W/m^2)**: Diffuse Horizontal Irradiance, solar radiation received per square meter on a horizontal surface that does not arrive on a direct path from the sun.
- **ModA (W/m^2)**: Measurements from a module or sensor (A), similar to irradiance.
- **ModB (W/m^2)**: Measurements from a module or sensor (B), similar to irradiance.
- **Tamb (°C)**: Ambient Temperature in degrees Celsius.
- **RH (%)**: Relative Humidity as a percentage of moisture in the air.
- **WS (m/s)**: Wind Speed in meters per second.
- **WSgust (m/s)**: Maximum Wind Gust Speed in meters per second.
- **WSstdev (m/s)**: Standard Deviation of Wind Speed, indicating variability.
- **WD (°N (to east))**: Wind Direction in degrees from north.
- **WDstdev**: Standard Deviation of Wind Direction, showing directional variability.
- **BP (hPa)**: Barometric Pressure in hectopascals.
- **Cleaning (1 or 0)**: Signifying whether cleaning (possibly of the modules or sensors) occurred.
- **Precipitation (mm/min)**: Precipitation rate measured in millimeters per minute.
- **TModA (°C)**: Temperature of Module A in degrees Celsius.
- **TModB (°C)**: Temperature of Module B in degrees Celsius.
- **Comments**: This column is designed for any additional notes.

Week's Topics Covered

1. **Python Programming:**
   - Task-specific programming assignments.
2. **GitHub Commands:**
   - Continuous committing and repository management.
3. **Data Understanding and Exploration:**
   - Applying exploratory data analysis techniques.
4. **CI/CD:**
   - Understanding continuous integration and continuous deployment.
5. **Streamlit**
   - Creating a dashboard using Streamlit.

Team

Facilitators :

- Yabebal
- Mahlet
- Kerod
- Rediet
- Rehmet

Key Dates

- **Challenge Introduction** - 8:30 AM UTC time on Friday 06 Dec 2024.
- **Submission** - 8:00 PM UTC time on Sunday 08 Dec 2024.

Deliverables and Tasks to be done

Task 1:

Git and GitHub

- Tasks:
    - Setting up Python environment
    - Git version control
    - CI/CD
- Key Performance Indicators (KPIs):
    - Dev Environment Setup.
    - Relevant skill in the area demonstrated.
- Suggested folder structure:

```
├── .vscode/
│   └── settings.json
├── .github/
│   └── workflows
│       ├── unittests.yml
├── .gitignore
├── requirements.txt
├── README.md
|------ src/
├── notebooks/
│   ├── __init__.py
│   └── README.md
├── tests/
│   ├── __init__.py
└── scripts/
    ├── __init__.py
    └── README.md
```

Project Planning - EDA & Stats

- Tasks:
    - Data Understanding
    - Exploratory Data Analysis (EDA)
    - Statistical thinking
- KPIs:
    - Proactivity to self-learn - sharing references.
    - EDA techniques to understand data and discover insights,
    - Demonstrating Stats understanding by using suitable statistical distributions and plots to provide evidence for actionable insights gained from EDA.

Minimum Essential To Do

- Create a github repository that you will be using to host all the code for this week.
- Create at least one new branch called "task-1" for your analysis of day 1
- Commit your work at least three times a day with a descriptive commit message
- Perform Exploratory Data Analysis (EDA) analysis on the following:
    - **Summary Statistics**: Calculate the mean, median, standard deviation, and other statistical measures for each numeric column to understand data distribution.
    - **Data Quality Check**: Look for missing values, outliers, or incorrect entries (e.g., negative values where only positive should exist), especially in columns like GHI, DNI, and DHI and check for outliers, especially in sensor readings (ModA, ModB) and wind speed data (WS, WSgust).
    - **Time Series Analysis**: Plot **bar charts or line charts** of GHI, DNI, DHI, and Tamb over time to observe patterns by month, trends throughout day, or anomalies, such as peaks in solar irradiance or temperature fluctuations.
    - Evaluate the impact of cleaning (using the 'Cleaning' column) on the sensor readings (ModA, ModB) over time.
    - **Correlation Analysis**: Use **correlation matrices** or **pair plots** to visualize the correlations between solar radiation components (GHI, DNI, DHI) and temperature measures (TModA, TModB). Investigate the relationship between wind conditions (WS, WSgust, WD) and solar irradiance using **scatter matrices**.
    - **Wind Analysis**: Use **radial bar plots** or **wind roses** Identify trends and significant wind events by showing the distribution of wind speed and direction, along with how variable the wind direction tends to be.
    - **Temperature Analysis**: Examine how relative humidity (RH) might influence temperature readings and solar radiation.
    - **Histograms**: Create histograms for variables like GHI, DNI, DHI, WS, and temperatures to visualize the frequency distribution of these variables.
    - **Z-Score Analysis:** Calculate Z-scores to flag data points that are significantly different from the mean
    - **Bubble charts** to explore complex relationships between variables, such as GHI vs. Tamb vs. WS, with bubble size representing an additional variable like RH or BP (Barometric Pressure)
    - **Data Cleaning**: Based on the initial analysis, clean the dataset by handling anomalies and missing values, especially in columns like Comments which appear entirely null.

Bonus Task:

Dashboard Development Using Streamlit

- Tasks:
    - Designing and developing a dashboard using Streamlit to visualize data insights.
    - Integrating Python scripts to fetch and process data dynamically.
    - Implementing interactive features (e.g., sliders, buttons) to allow users to customize visualizations.

  - Deploying the Streamlit dashboard to Streamlit Community Cloud.
- KPIs
  - **Dashboard Usability**: Ease of use, with intuitive navigation and clear labels.
  - **Interactive Elements**: Effective use of Streamlit widgets to enhance user engagement.
  - **Visual Appeal**: Clean and professional design that effectively communicates data insights.
  - **Deployment Success**: Fully functional deployment, accessible via a public URL.
- Suggested Folder Structure:

```
├── .streamlit
│   └── config.toml
├── .vscode
│   └── settings.json
├── .github
│   └── workflows
│       ├── unittests.yml
├── .gitignore
├── requirements.txt
├── README.md
├── notebooks
│   ├── __init__.py
│   ├── example.ipynb
│   └── README.md
├── tests
│   ├── __init__.py
├── app
│   ├── __init__.py
│   ├── main.py  # main Streamlit application script
│   ├── utils.py  # utility functions for data processing and visualization
└── scripts
    ├── __init__.py
    └── README.md
```

Minimum Essential To Do:

- Merge the necessary branches from task-1 into the main branch using a Pull Request (PR)
- Create at least one new branch called "dashboard-dev" for the ongoing development of the dashboard.
- Commit your work with a descriptive commit message.
- Design and develop the Streamlit dashboard to visualize the dataset with interactive elements
- Document the development process and usage instructions in the README.md file.

Due Date (Submission)

Sunday: (Dec, 08, 20248:00 PM (UTC)

- GitHub Link to your main branch
- Final report - covering all week-0 work.
  - Write it in the format that you could post it as a Blog on Medium.
- Include your dashboard screenshot into github dashboard folder

Other Considerations:

- **Documentation:** Encourage detailed documentation in code and report writing.
- **Collaboration:** Emphasise collaboration through Github issues and projects.
- **Communication**: Regular check-ins, Q&A sessions, and a supportive community atmosphere.
- **Flexibility:** Acknowledge potential challenges and encourage proactive communication.
- **Professionalism:** Emphasise work ethics and professional behavior.
- **Time Management:** Stress the importance of punctuality and managing time effectively.

Tutorials Schedule

In the following, the color **purple** indicates morning sessions, and non-purple indicates afternoon sessions.

- Day 1 (Friday 06 Dec 2024):
    - Introduction to the Challenge
    - Python Environment Setup, Git and Github, CI/CD (Kerod)
- Day 2 (Saturday 07 Dec 2024):
    - Project Planning & EDA techniques (Rediet)
    - Dashboard development using Streamlit (Rehmet)

Feedback

You will receive comments/feedback in addition to a grade.

References

- **Python Testing**
  - https://machinelearningmastery.com/a-gentle-introduction-to-unit-testing-in-python/
  - https://docs.python-guide.org/writing/tests/
  - https://realpython.com/python-testing/
- **Dashboard design and implementation**
  - Get started - Streamlit Docs
  - Streamlit 101: An in-depth introduction | by Shail Deliwala | Towards Data Science
  - StreamLit - Data Scientists tool for developing web apps | by INSAID | Medium
  - Streamlit Community Cloud.
- **Python Programming:**
  - Object Oriented Programming
  - Python Courses and Tutorials: Online and On Site (python-course.eu)
- **Data Engineering**
  - What is Data Engineer: Role Description, Skills, and Background | AltexSoft
- **Version control – Git**
  - What is version control | Atlassian
  - Learn Git branching -- interactive way to learn Git
  - Git with large files
  - Which files to not track and how to not track them? | Atlassian
  - .gitignore docs
  - Conventional commits -- lightweight convention on top of commit messages.
- **CI/CD**
  - What is Continuous Integration | Atlassian
  - DevOps Pipeline | Atlassian
  - 7 Popular Open Source CI/CD Tools - DevOps.com
  - Setting up a CI/CD pipeline on Github