Zelestra X AWS ML Ascend Challenge - 2nd Edition

# Tools and Technologies Used

To tackle this challenge, we leveraged a well-rounded data science stack:

1. Programming Language: Python 3
2. Development Environment: Jupyter Notebooks (local setup)
3. Core Libraries:
   * Data Handling: pandas, numpy
   * Visualization: matplotlib, seaborn
   * Machine Learning: scikit-learn, xgboost, lightgbm, catboost

# Methodology

We followed a structured machine learning workflow to guide our solution development:

1. **Data Preprocessing**  
   We started by cleaning the data, handling missing values, and removing outliers. Features were transformed, categorical variables were encoded, and numerical data was scaled using StandardScaler or MinMaxScaler.
2. **Model Development**  
   We built baseline models such as Linear Regression and Decision Trees before moving to more powerful ensemble models like XGBoost, LightGBM, and CatBoost. Eventually, we implemented a stacked regressor to combine the strengths of the best individual models.
3. **Evaluation Metrics**  
   We evaluated our models using RMSE, MAE, and R² Score, with a target of achieving over 90% accuracy on test data.
4. **Model Validation**  
   We used K-Fold Cross-Validation (K=5) to ensure our model generalizes well and validated performance using leaderboard feedback.

# Our Approach

#### ****1. Data Understanding & Preprocessing****

* Performed exploratory analysis on the dataset.
* Cleaned missing or inconsistent values.
* Derived new features such as power ratio and panel efficiency.
* Applied scaling to normalize feature distributions.

#### ****2. Feature Engineering****

* Encoded categorical variables using label encoding.
* Generated new interaction features based on environmental variables like temperature, irradiance, and humidity.
* Removed features with high multicollinearity and low variance.

#### ****3. Model Selection & Training****

* Trained and evaluated baseline models to set benchmarks.
* Implemented and fine-tuned advanced models: XGBoost, LightGBM, CatBoost.
* Created a stacked regressor that aggregated predictions from the top models.
* Used grid search for hyperparameter optimization.

#### ****4. Evaluation Strategy****

* Prioritized RMSE and MAE to measure model precision.
* Used K-Fold Cross-Validation to avoid overfitting.
* Monitored performance on public leaderboards.

# File Mapping

|  |  |
| --- | --- |
| File Name | Description |
| Solar\_V2.ipynb | Intermediate enriched model development |
| Solar\_v3\_tus (1).ipynb | Final version with stacking and performance tuning |
| solar\_v1\_accuracy\_89.90197.ipynb | Notebook associated with the first successful version |

# Steps to Reproduce

* To reproduce the results locally:
  + Install the required packages:  
    pip install -r requirements.txt
* Execute the notebooks in the following order:
  + Solar\_V2.ipynb
  + solar\_v1\_accuracy\_89.90197.ipynb
  + Solar\_v3\_tus (1).ipynb

# Final Model

Our final submission was powered by a **Model ensemble (weighted averaging)** that combined predictions from XGBoost, CatBoost, and LightGBM. This approach yielded the most consistent and accurate predictions, outperforming individual models in validation metrics.

# Key Learnings and Innovation

- Stacking multiple models led to a notable performance boost.  
- Feature engineering played a critical role in improving accuracy.  
- Careful data cleaning and validation ensured generalizable predictions.

# Conclusion

This challenge allowed us to apply a systematic ML approach to an impactful real-world problem. Our efforts led to the development of a high-accuracy predictive model that can be scaled and deployed for practical solar energy forecasting applications.