**Household Energy Consumption Analysis**

***Purpose:***

The purpose of the *PowerPulse: Household Energy Usage Forecast* project is to develop a machine learning model that predicts household energy consumption based on historical data. Accurate predictions enable consumers to optimize energy usage, reduce costs, and promote efficient habits. Additionally, energy providers can use the model to forecast demand, optimize resource allocation, and improve load management, while contributing to environmental conservation efforts by reducing energy wastage.

***Need:***

With rising energy demand and increasing environmental concerns, it is crucial for both households and energy providers to manage energy consumption effectively. A reliable predictive model can assist households in monitoring and reducing energy usage, while providing energy providers with the ability to forecast demand and optimize supply chains. This can lead to lower costs, efficient energy distribution, and a reduced environmental impact.

***STEPS***

1.Importing needed Libraries

**Libraries and Their Purpose**

**zipfile** – Handles ZIP archive extraction and compression.

**pandas** – Provides data structures like DataFrames for data manipulation and analysis.

**numpy** – Supports numerical operations, arrays, and matrix computations.

**matplotlib.pyplot** – Used for creating static, animated, and interactive visualizations.

**seaborn** – Enhances data visualization with statistical plotting capabilities.

**sklearn.preprocessing.MinMaxScaler** – Scales features to a given range (0 to 1) to normalize data.

**sklearn.model\_selection.train\_test\_split** – Splits dataset into training and testing sets.

**sklearn.linear\_model.LinearRegression** – Implements linear regression for predictive modeling.

**sklearn.metrics** – Provides performance evaluation metrics such as:

* + mean\_squared\_error (MSE) – Measures average squared error of predictions.
  + mean\_absolute\_error (MAE) – Calculates average absolute difference between predicted and actual values.
  + r2\_score – Evaluates model fit by measuring variance explained.

**sklearn.ensemble.RandomForestRegressor** – A machine learning algorithm that uses multiple decision trees for regression tasks.

**sklearn.model\_selection.RandomizedSearchCV** – Performs hyperparameter tuning using random search for model optimization.

2.File Extraction

**Extract ZIP File**

* Opens and extracts the dataset from the ZIP archive.

**Load Dataset**

* Reads the extracted .txt file as a DataFrame using ; as the separator.

**Display Data Structure**

* data.info() – Shows column details and data types.
* data.head() – Displays the first five rows.

2.Data Preprocessing

**Handle Missing Values**

* Replaces ? with pd.NA for proper missing value handling.

**Convert Numeric Columns**

* Converts specified columns to numeric type, coercing invalid values to NaN.

**Check Missing Values**

* Prints the count of missing values in each column.

4. Date and Time Parsing

**Convert 'Date' and 'Time' to Datetime Format**

* Ensures 'Time' is a string before merging with 'Date'.
* Uses pd.to\_datetime() to create a new 'Datetime' column with correct format (dayfirst=True).

**Extract Date & Time Components**

* Splits 'Datetime' into separate columns:
  + Date, Time – Extracts date and time.
  + Year, Month, Day – Extracts date components.
  + Hour, Minute, Second – Extracts time components.

**Verify DataFrame Updates**

* Prints the first five rows to check the changes.

5. Handling missing data

**Remove Missing Datetime Rows**

* Drops rows where 'Datetime' is NaN.

**Select Numeric Columns**

* Identifies columns with numeric data types.

**Fill Missing Numeric Values**

* Replaces NaN in numeric columns with the column mean.

**Verify Missing Values**

* Prints the count of remaining missing values.

6. Feature Engineering

**Extract Date, Hour, and Day of the Week**

* Creates new columns for Date, Hour, and DayOfWeek from 'Datetime'.

**Calculate Daily Averages**

* Computes the mean of key energy metrics per day.

**Add Rolling Averages**

* Computes 1-day and 7-day rolling averages for Global\_active\_power.

**Identify Peak Usage Hour**

* Finds the hour with the highest average Global\_active\_power consumption.
* Adds a flag (IsPeakHour) for peak hours.

**Summarize Energy Usage by Day of the Week**

* Computes average Global\_active\_power and Global\_reactive\_power by weekday.

**Calculate Daily Total Energy Consumption**

* Sums Global\_active\_power for each day.

**Normalize Features**

* Uses MinMaxScaler to scale key numerical columns between 0 and 1.

**Save Processed Dataset**

* Saves the feature-engineered dataset as a CSV file.

7. Exploratory Data Analysis (EDA)

**Ensure Correct Datetime Format**

* + Converts 'Datetime' to a proper datetime format, handling errors.

**Select Numeric Columns**

* + Filters only float64 and int64 columns for analysis.

**Correlation Heatmap**

* + Visualizes feature relationships using a heatmap.

**Distribution Analysis (Histograms)**

* + Plots histograms for key energy variables to analyze their distribution.

**Time Series Analysis**

* + **Daily Trend**: Plots daily average Global\_active\_power.
  + **Monthly Trend**: Computes and plots the monthly average trend.
  + **Yearly Trend**: Displays a bar chart of yearly average consumption.

These visualizations help in understanding energy consumption patterns over time.

8. Modelling

**Define Features & Target Variable**

* + X: Selected input features.
  + y: Target variable (Global\_active\_power).

**Train-Test Split**

* + Splits data into **80% training** and **20% testing** sets.

**Train Linear Regression Model**

* + Initializes and fits a LinearRegression model.

**Make Predictions**

* + Predicts y\_pred on the test set.

**Evaluate Model Performance**

* + **RMSE**: Measures prediction error magnitude.
  + **MAE**: Shows average absolute error.
  + **R²**: Indicates how well the model explains variance.

This establishes a baseline for energy consumption prediction using linear regression.

9. Random Forest Regressor

**Initialize Random Forest Model**

* + Creates a RandomForestRegressor with 100 estimators and a maximum depth of 20.

**Train the Model**

* + Fits the Random Forest model on the training data.

**Make Predictions**

* + Predicts y\_pred\_rf on the test set.

**Evaluate Model Performance**

* + **RMSE**: Measures prediction error magnitude.
  + **MAE**: Calculates average absolute error.
  + **R²**: Indicates model's variance explanation.

This model is used to assess the prediction performance of energy consumption using Random Forest.

10. Feature Importance

**Extract Feature Importances**

* + Retrieves feature importance scores from the trained Random Forest model.

**Create DataFrame for Feature Importance**

* + Creates a DataFrame (importance\_df) with feature names and their corresponding importance scores.
  + Sorts the DataFrame in descending order of importance.

**Display Feature Importances**

* + Prints the feature importances in sorted order.

**Plot Feature Importances**

* + Plots a horizontal bar chart to visualize the importance of each feature, with higher importance at the top.

This helps in understanding which features contribute most to the model's predictions.

11. Summary and Insights

**Create Performance Comparison DataFrame**

* + A DataFrame (performance\_comparison) is created to compare the performance of the **Linear Regression** and **Random Forest** models.
  + It includes columns for RMSE, MAE, and R² for both models.

**Display Model Performance Comparison**

* + Prints the DataFrame showing the comparison of model evaluation metrics.

This provides a summary to assess and compare the performance of the two models.

12. Time-series visualization of actual vs predicted values

**Plot Actual vs Predicted Values**

* + Plots the first 100 values of the actual (y\_test.values[:100]) and predicted (y\_pred\_rf[:100]) energy usage for comparison.
  + **Actual values** are shown in blue, and **predicted values** are shown in orange.

**Customize Plot**

* + Adds title, axis labels, and a legend for clarity.
  + Enables grid for better readability.

This visualization helps in assessing how well the Random Forest model's predictions match the actual energy consumption values over a sample of the test data.

***Conclusion:***

The *PowerPulse* project successfully provides a predictive model that helps in understanding household energy consumption patterns. By leveraging machine learning techniques such as regression modeling, feature engineering, and hyperparameter tuning, the model delivers accurate predictions, valuable insights into energy usage trends, and practical recommendations for optimization. These insights can guide both consumers and energy providers toward better energy management, cost reduction, and sustainability efforts, contributing to a more efficient and eco-friendly future.