# Project Report: Energy Efficiency Analysis Web Application

## Executive Summary

This web-based data analysis application provides an in-depth exploration of an energy efficiency dataset through advanced statistical and machine learning techniques. Leveraging Python’s powerful data science libraries, the project offers comprehensive insights via an interactive web interface.

## Project Overview

The application is a sophisticated data analysis tool designed to perform detailed statistical and machine learning analysis on an energy efficiency dataset, utilizing a modern web framework with robust data science methodologies.

Members: 23K-0030 Abser Mansoor ,23K-0032 Saif Ur Rehaman,23K-0052 Muhammad Ali

## Technical Stack

* **Web Framework**: Flask
* **Data Analysis**: Pandas, NumPy
* **Machine Learning**: Scikit-learn
* **Visualization**: Matplotlib, Seaborn
* **Frontend**: HTML templates

## Data Source

Dataset: Energy Efficiency Dataset - Source: [Kaggle Energy Efficiency Dataset](https://www.kaggle.com/datasets/elikplim/eergy-efficiency-dataset)

## Detailed Project Components

### 1. Web Application Architecture (app.py)

#### Key Features

* Modular route design
* Comprehensive data analysis integration
* Dynamic template rendering

#### Route Definitions

1. **Home Route (/)**
   * Renders main dashboard
   * Provides navigation to key analysis sections
2. **Graphs Route (/graphs)**
   * Generates comprehensive data visualizations
   * Includes:
     + Box Plots
     + Distribution Plots
     + Correlation Heatmap
     + Regression Plots
3. **Regression Route (/regression)**
   * Performs K-Nearest Neighbors Regression
   * Analyzes two target variables (Y1 and Y2)
   * Computes key performance metrics
4. **Statistics Route (/statistics)**
   * Conducts in-depth statistical analysis
   * Generates statistical visualizations

### 2. Data Visualization (analysis/Graphs.py)

#### Visualization Techniques

1. **Box Plots**
   * Covers all features (X1-X8, Y1, Y2)
   * Identifies data distribution and outliers
2. **Distribution Plots**
   * Individual feature probability density analysis
   * Reveals underlying data characteristics
3. **Correlation Heatmap**
   * Visualizes inter-feature relationships
   * Detects potential multicollinearity
4. **Regression Plots**
   * Explores feature pair relationships
   * Includes trend-indicating regression lines

### 3. Regression Analysis (analysis/KNeighboursRegressor.py)

#### Methodology

* K-Nearest Neighbors (KNN) Regression
* Dual target variable analysis (Y1 and Y2)

#### Analysis Process

1. Data Preprocessing
2. Feature Scaling
3. Performance Evaluation

#### Performance Metrics

* Mean Squared Error (MSE)
* R² Score
* Optimal K-value determination

### 4. Statistical Analysis (analysis/statistical\_analysis.py)

#### Statistical Metrics Computed

* Standard Deviation
* Mean
* Median
* Mode
* Variance

### 5. HTML Templates

#### index.html

* Project dashboard
* Navigation to key analysis sections
* Dataset source link

#### graphs.html

* Comprehensive visualization display
* Grid-based graph layout
* Dynamically rendered visualizations

#### regression.html

* Detailed regression analysis results
* Metrics for Y1 and Y2
* Regression comparison plot

#### statistics.html

* Statistical metrics visualization
* Comprehensive statistical plot

## Project Visualization Outputs

Generated visualizations stored in static folder: - Box Plot - Distribution Plots - Correlation Heatmap - Regression Comparison Plot - Statistical Analysis Plot

## Implementation and Deployment

### Prerequisites

* Python 3.x
* Required libraries:
  + Flask
  + Pandas
  + NumPy
  + Scikit-learn
  + Matplotlib
  + Seaborn

### Deployment Steps

1. Install dependencies
2. Run app.py
3. Access analysis pages via web interface

## Key Insights

* Comprehensive multi-dimensional analysis of energy efficiency dataset
* Advanced regression and statistical techniques
* Interactive web-based visualization

## Potential Improvements

1. Implement more advanced regression techniques
2. Add interactive data exploration features
3. Enhance error handling
4. Develop more sophisticated visualization options

## Technical Challenges and Solutions

* Handled data preprocessing and cleaning
* Implemented robust scaling techniques
* For UI had to change the python scripts into functions for importing them in the app.py code