Ahmad Mehmood DS-D Ahmad Luqman

I22-1915 I22-2018

Energy Demand Forecasting Documentation

Project Overview

This application provides advanced energy demand forecasting capabilities using machine learning and clustering techniques. It allows users to analyze historical energy consumption patterns, identify similar usage clusters, and predict future demand based on weather conditions and temporal factors.

The system combines multiple forecasting models with interactive visualizations to help energy planners, utilities, and researchers make data-driven decisions.

Key Features

- City-specific energy demand analysis
- K-means clustering of energy consumption patterns
- Multiple forecasting models (Random Forest, XGBoost, ARIMA, SARIMA)
- Interactive visualizations of clusters and forecasts
- Adjustable parameters for customized analysis
- Comprehensive performance metrics

Methodology

Data Preprocessing

Raw energy demand and weather data undergo several preprocessing steps:

- Missing value imputation using mean/median values
- Feature engineering to extract temporal patterns (hour, day, month, season)
- Standardization of numerical features
- Anomaly detection and removal using Z-scores and Isolation Forest

Clustering Analysis

K-means clustering is applied to identify distinct energy usage patterns:

- Features include temperature, humidity, wind speed, and demand
- Optimal k determined using elbow method and silhouette scores
- PCA used for dimensionality reduction and visualization
- Cluster analysis reveals weather-dependent consumption patterns

Forecasting Models

Multiple forecasting approaches are implemented:

- **Random Forest:** Ensemble of decision trees with hyperparameter tuning
- **XGBoost:** Gradient boosting optimized for performance
- **ARIMA:** AutoRegressive Integrated Moving Average for time series forecasting
- **SARIMA:** Seasonal ARIMA that captures seasonal patterns in time series data
- **Baseline:** Naive forecast using previous day's same hour

Using the Application

Input Parameters

To begin your analysis:

1. Select a city from the dropdown menu
2. Choose a start date and end date for your analysis period
3. Click "Analyze Data" to process your selection

Model Controls

Customize your analysis with these controls:

- **Number of Clusters (k):** Adjust the slider to change the number of clusters in the K-means algorithm
- **Look-back Window:** Set how many previous hours to consider for time series forecasting
- **Forecasting Model:** Choose between Random Forest, XGBoost, ARIMA, and SARIMA algorithms

Interpreting Results

The application provides two main visualizations:

- **Cluster Visualization:** Shows how data points group together based on similar characteristics. Points of the same color belong to the same cluster.
- **Forecast Plot:** Compares actual energy demand (blue line) with predicted demand (red dashed line). The closer these lines, the more accurate the forecast.

Performance metrics (MSE, MAE, R^2) help you evaluate forecast accuracy. Lower MSE and MAE values and higher R^2 values indicate better performance.

2. Backend (FLASK)

2.1 Data Generation

- The application uses generic data generation functions that generate synthetic time series data with seasonal patterns and noise.
- Seed-based randomization is used for consistency in generated results.

2.2 API Endpoints

- **/api/cluster**: Performs K-means clustering on city-specific data.
- **/api/overall-cluster**: Clusters data across all cities.
- **/api/forecast**: Provides energy demand forecasting using four models (Random Forest, XGBoost, ARIMA, SARIMA).
- **/api/cities**: Returns the list of available cities.

2.3 Machine Learning Implementation

- K-means clustering is used for data segmentation, with PCA for dimensionality reduction.
- Random Forest and XGBoost are used for regression tasks.
- ARIMA and SARIMA provide time series forecasting with simplified implementations using lagged values.
- Feature scaling is applied for improved model performance, with standard metric calculation.

3. Frontend (index.html)

3.1 User Interface

- The frontend provides a city selection dropdown and date range inputs.

- Users can control clustering parameters (e.g., k-value) and forecasting settings (e.g., lookback window).
- Navigation is organized through tab-based menus.

3.2 Visualization Components

- City clustering visualization with PCA-based scatter plots.
- Cross-city cluster visualization for overall data patterns.
- Time series forecasting plot for actual vs. predicted values.
- Model comparison bar chart showing performance metrics.

3.3 JavaScript Functionality

- Chart.js is used for interactive data visualization.
- AJAX requests enable real-time interaction with backend APIs.
- Dynamic UI updates are based on user interactions.

4. Key Interactions

4.1 Data Flow

- Users select parameters in the frontend.
- Frontend sends AJAX requests to backend API endpoints.
- Backend processes data and returns JSON responses.
- Frontend visualizes the results dynamically.

4.2 Model Execution

- Machine learning computations are handled on the backend.
- Results are displayed on the frontend with real-time updates.
- Users can adjust parameters to see updated predictions and cluster visualizations.

The Energy Demand Forecasting Application efficiently combines machine learning, data visualization, and a user-friendly interface to provide accurate and interactive energy demand forecasting. The backend handles complex model computations, while the frontend ensures intuitive visualization and user interaction.