AI Methadology Project

Mariia Isaieva, Lin Yang

Energy consumption prediction

Introduction

Our machine learning project on energy consumption aims to develop a predictive model that accurately forecasts energy usage based on weather patterns and time of day. By leveraging machine learning algorithms and techniques, our project aims to help individuals and organizations optimize their energy usage, reduce waste, and save money.

With the increasing importance of sustainability, energy efficiency, and environmental responsibility, our project is both timely and relevant. By delivering a machine learning model that accurately predicts energy consumption, we can contribute to the broader goals of creating a more sustainable and efficient future.

In this project, our team of data scientists and machine learning engineers worked collaboratively using the Scrum project management framework. Through iterative development and experimentation, we were able to refine our model and deliver value to our end-users.

Functional part

Use case

The use case of our project is to develop a predictive model that accurately forecasts energy consumption based on weather patterns and time of day. The model can be used by individuals and organizations to optimize their energy usage, reduce waste, and save money. The use case is particularly relevant for homeowners, energy companies, city planners, and researchers who are interested in understanding energy consumption patterns or want to design more energy-efficient buildings and infrastructure.

Goal

The goal of our project is to develop a predictive model that accurately forecasts energy consumption based on weather patterns and time of day. The model aims to help individuals and organizations optimize their energy usage, reduce waste, and

save money. By achieving this goal, our project can contribute to the broader goals of sustainability, energy efficiency, and environmental responsibility.

Users

The users or customers for our project could be anyone who is interested in understanding energy consumption patterns or wants to optimize their energy usage. This could include:

- Homeowners who want to reduce their energy bills and carbon footprint.
- Energy companies who want to forecast energy demand and optimize their energy production and distribution.
- City planners who want to design more energy-efficient buildings and infrastructure.
- Researchers who want to study energy consumption trends and patterns.

By delivering a machine learning model that accurately predicts energy consumption based on weather patterns and time of day, our project could help these users or customers make more informed decisions about their energy usage, reduce waste, and save money.

Dataset

The WiDS Datathon 2022 focuses on a prediction task involving roughly 100k observations of building energy usage records collected over 7 years and a number of states within the United States. The dataset consists of building characteristics (e.g. floor area, facility type etc), weather data for the location of the building (e.g. annual average temperature, annual total precipitation etc) as well as the energy usage for the building and the given year, measured as Site Energy Usage Intensity (Site EUI). Each row in the data corresponds to the a single building observed in a given year. The task is to predict the Site EUI for each row, given the characteristics of the building and the weather data for the location of the building.

Project management framework

As a project management framework the Scrum was used.
Scrum's flexibility and adaptability allowed us to prioritize features based on our educational goals, refine our ML models iteratively, and manage unexpected challenges as they arose. By delivering value to our end-users and continually improving our processes, we were able to successfully apply machine learning techniques to real-world problems.

Team organization

- ML Engeneer - Lin Yang
    - Conducted data preprocessing and feature engineering to prepare data for model training.
    - Tested and iteratively refined ML models based on performance metrics.
    - Used MLflow to deploy and manage models in production, and monitor their performance over time.

- Data Scientist - Mariia Isaieva
    - Developed data preprocessing pipelines and feature engineering strategies to prepare data for model training.
    - Trained and evaluated machine learning models using various algorithms and techniques.
    - Used MLflow to track and manage experiments, version control models, and reproduce results.

Technical part

Requirements

For the project the following packages were used:

- jupyter==1.0.0
- pandas==1.5.3
- numpy==1.24.2
- scikit-learn==1.2.2
- shap==0.41.0
- mlflow==2.2.2
- category_encoders==2.6.0

The code was written in the Visual Studio Code IDE.

Part I

For code versioning and collaboration, we used Git and GitHub. Git is a powerful tool for version control, allowing us to track changes made to our codebase and collaborate effectively with team members. GitHub provided a platform for sharing our code and collaborating on the project with our team members.

To streamline our machine learning workflow, we separated it into different scripts placed in files helpers.py, preprocess.py, feature_selection.py, and constants.py in the folder source. By doing so, we were able to keep our code organized and modular, making it easier to maintain and update. The data loading, train-test split, preprocessing, and feature selection

were done in their respective scripts, enabling us to streamline our workflow and focus on specific tasks.

We used a custom project structure that is commonly used in machine learning projects: the data folder contains datasets, and the source folder contains all the code used for model training. Additionally, we included a Jupyter Notebook to tie everything together. This structure allowed us to keep our code organized, making it easier to understand and maintain.

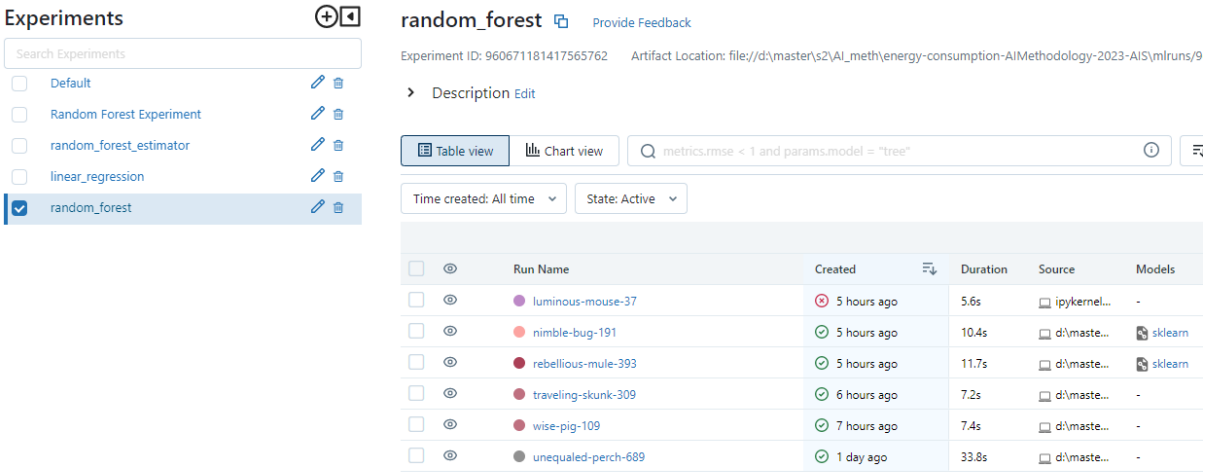
To manage our project's dependencies, we used Miniconda as a Conda environment. Miniconda is a lightweight distribution of the Conda package manager, allowing us to create a self-contained environment for our project's dependencies. This helped us avoid conflicts with other projects and kept our dependencies consistent throughout the project.
Finally, we made sure to document our project thoroughly. We documented our code, data, and project structure, making it easy for team members to understand and navigate the project. This documentation also serves as a reference for future developers who may work on the project, ensuring that our work can be continued and improved upon in the future.

Part II

To experiment with our model's configurations and parameters, we used MLflow, an open-source platform for managing the end-to-end machine learning lifecycle. For each model, we created separate experiments, enabling us to test different parameters and configurations. For our Energy Consumption regression task, we tried both the Logistic Regression and Random Forest Regressor models. While the metrics with the simple Logistic Regression were better than with the Random Forest, we continued working with the Forest model because it produced more interesting results and observations in combination with Shap. Additionally, the Tree Explainer does not work for linear regression models.

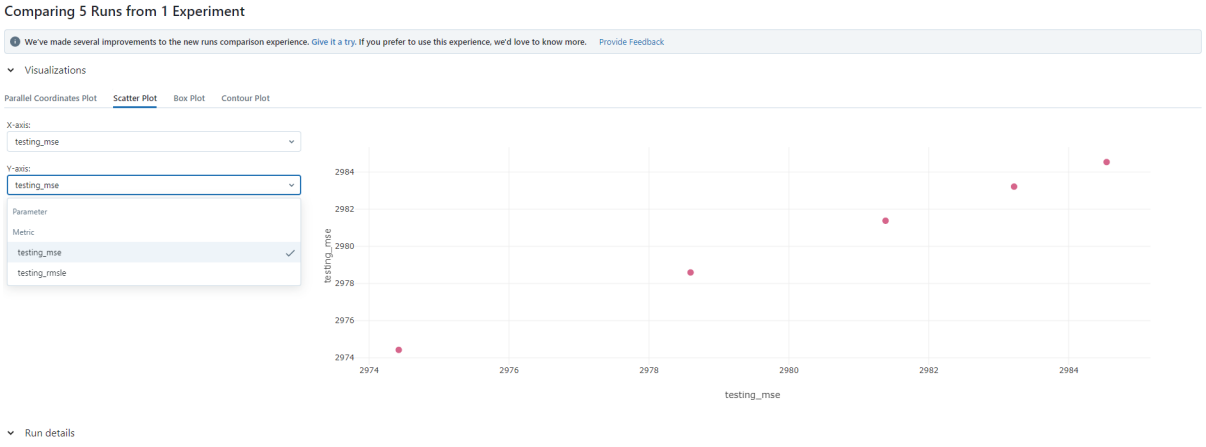
Using MLflow, we were also able to compare and select the best number, combination, and preprocessing for features. Initially, we started with only numerical features, but the model's performance wasn't very good. To address this, we performed Recursive Feature Elimination (RFE) using Logistic Regression and selected only 10 features. For preprocessing, we used the Standard Scaler for all numerical columns and the Count Encoder for the Facility_Type column, as it has many unique values. We also decided to change the values of the State_Factor column from "State_1" to 1, for example, leaving only integers. Interestingly, using One-Hot Encoder for this column resulted in these new features being important, as they were in the top 10 for the model. However, after leaving only integers, this feature wasn't as useful, but it did not decrease the model's performance.

- The experiments and runs in MLFlow UI:



We logged all experiments through MLflow, enabling us to compare results with metrics and plots. This allowed us to easily visualize and analyze the performance of different models, configurations, and preprocessing techniques.

- The visualization of metrics comparison:



Finally, we packaged our code in a reusable and reproducible model format with MLflow projects. This format enables others to reproduce our work and build on it in a consistent and organized manner. By creating a package, we ensure that our work can be continued and improved upon in the future.

Part III

As part of our ML project on energy consumption, we integrated the SHAP library into our codebase. This library was installed in our Python environment, and we made sure to include it in our lib requirements. We used SHAP to explain our model predictions, and we found it to be a powerful tool for understanding how our models were making decisions.

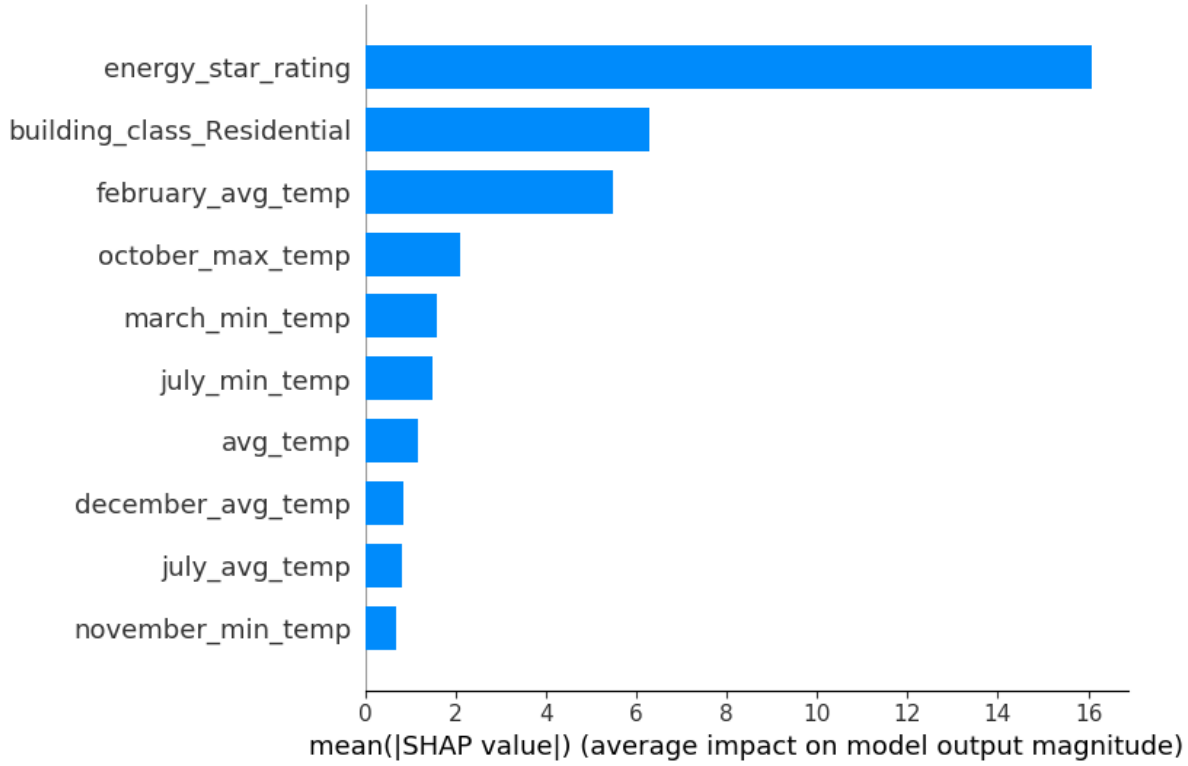
To use SHAP, we built a TreeExplainer and computed Shapley Values for our model. This allowed us to generate explanations for individual data points, as well as for the dataset as a

whole. We also used SHAP to create summary plots for each class in our dataset, which helped us to understand how our model was performing across different categories.

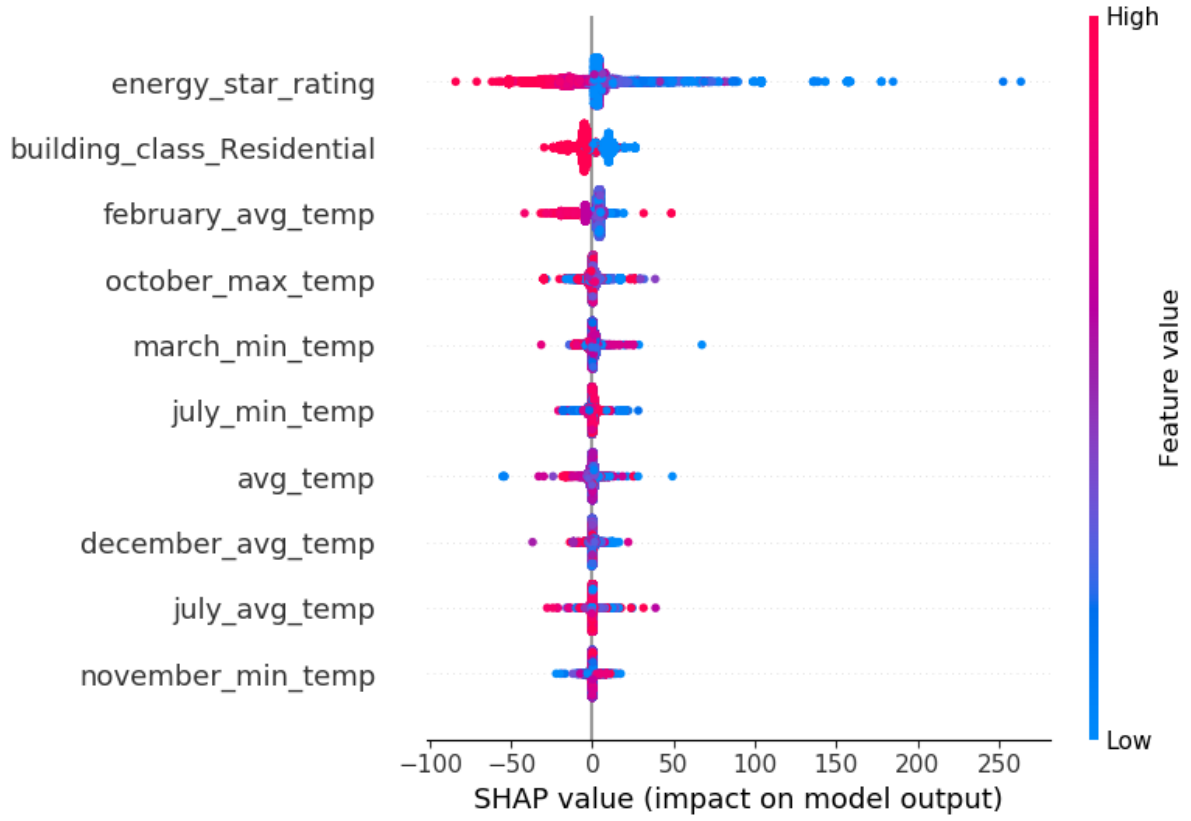
Overall, integrating SHAP into our project was a valuable step in understanding and explaining our model's behavior. It gave us a powerful tool for interpreting our results and gaining insights into our data.

- The explanation for the specific point of the dataset - Force Plot:

- Expalations for a whole dataset - Feature Importance:



- Expalations for a whole dataset - Summary Plot:



Conclusion

We were able to build an accurate and reliable regression model that can predict energy consumption with a high degree of accuracy. We used a variety of techniques and tools to preprocess our data, select features, and train our model, including Git for code versioning and collaboration, mlflow for model experimentation and comparison, and SHAP for interpreting and visualizing our model's behavior.

Our team of worked collaboratively to develop and improve our model, using a range of skills and expertise to optimize our results. We also documented our work and created a reusable and reproducible code package with MLflow projects.

We hope that our project may be valuable applications in the energy industry, helping to improve energy efficiency and reduce waste.