Project Methodology

Mitchell Hornsby

The goal of this project is to forecast the imbalances in energy demand in the 13 regional areas of power generation as defined by the US Energy Information Administration [9]. We seek to identify regions that could benefit from the increased adoption of renewable energy, specifically solar and wind power. We will compare regional climate and energy production and demand data with a goal of showing that for a given region with an energy deficit, they could benefit based on climate characteristics to investing more in solar or wind power.

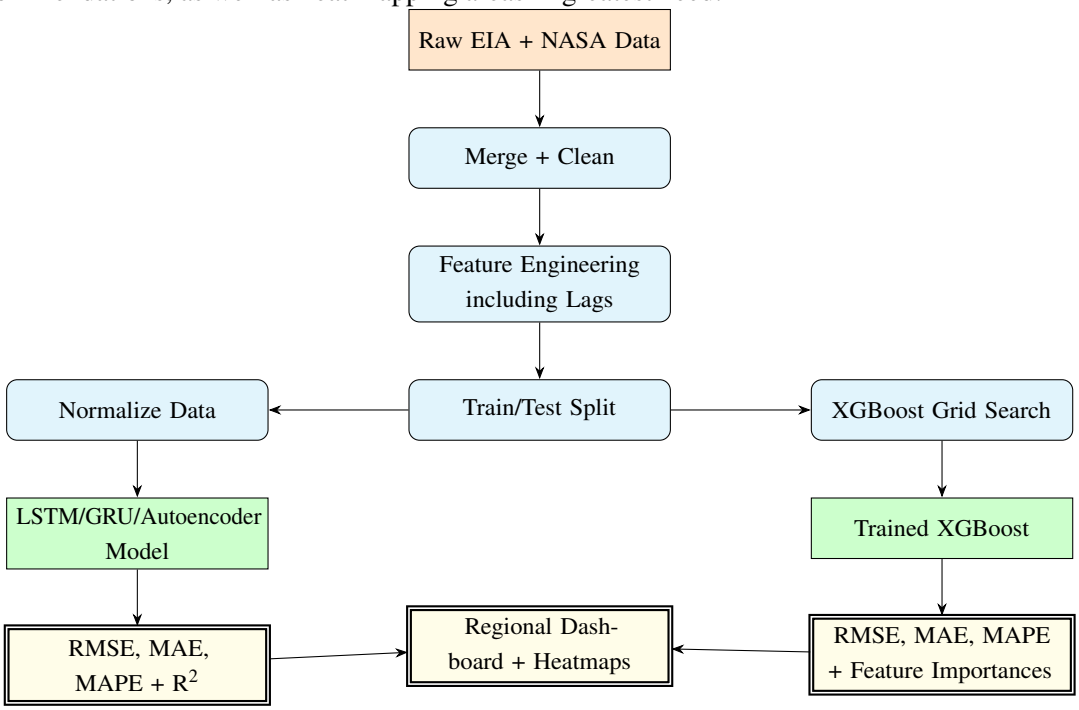
To undertake this task we will implement several well known machine learning approaches to forecast via regression the amount of energy a region will either contribute to the wider grid, or need to ingest to fund a deficit. The below chart illustrates our approach at a high level. Using available API's from both the EIA and NASA's Power project [6] we obtain hourly observation data for each region over the past 5 years. We perform some preprocessing steps to merge and clean the data set and also create some new data called features through feature engineering to calculate percentages, rolling averages or other derived metrics that may assist is revealing patterns in the data. We then split the data into training/validation/testing data sets where we include a few years in the training data for the model, verify the performance of the model versus the validation set for tuning and improving performance and then lastly generalize over the test set of unseen data.

We need to consider two tracks for our modeling. Our first track is to use a gradient boosting module known as XGBoost to train the model. This model works very well on tabular data and is tree based where splits are made based on fit attributes to follow the data to the most accurate representation of the training set. This is accomplished by combining many simple decision trees sequentially and improving the model with each sequential iteration. This model is very flexible and can be tuned very effectively for different parameter combinations. This model will serve as our performance baseline due to it's straightforward training and flexibility as well as it not requiring any features to be scaled or normalized.

We will also train a few types of Recurrent Neural Networks, RNN's that have specific use cases for time series via the concept of a sequence. While XGBoost may have features that influence it's understanding of the data's order, it is not necessarily understanding that it is training on an ordered sequence of observations. RNN's can take the "state" of a previous observation and incorporate that information into the next observation's training state for the length of a sequence window. Two such models we will utilize are called Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). Both models perform a type of gating between the prior state and current state in the learning and have differing complexities (with

GRU being simpler) which can contribute to greater understanding of our model. As a final layer, we will implement an autoencoder to compress some of our engineering features into latent vectors and then use this updated ata to retrain the RNN's. This can help us address some noise and multicollinearity (related features) that could be causing performance issues.

Lastly, we will compare the results using various standard regression metrics and select the best model for each region, creating a dashboard to show the results and any needed recommendations, as well as heat mapping areas in greatest need.



1. Purpose of Methodology

Our chosen approaches provide us with flexibility and depth for seeking consistent and reliable models in areas where we may see considerable differences due to size of the region, population and demand, as well as climatic idiosyncrasies.

XGBoost is suitable as a baseline as it does not require normalization, has a few hyperparameters to help tune and can handle many features at once. We are able to engineer many features to try to capture longer term patterns and cyclical behavior that it may not otherwise capture since it is not recognizing sequences.

RNN's are suitable as they are able to remember sequences and thus can help with longer term and seasonal patterns. For LSTM and GRU the Keras models provide us with flexibility to add layers to the RNN in order to help capture these longer term and higher level interactions [2].

Adding an Autoencoder can help reduce the data the RNN's need to utilize without sacrificing information and will further assist in allowing the RNN's to focus on information rather than being overwhelmed by features.

We will compare the end states of each type of model per region and hope to gain insight on which work best in forecasting for that region's specific needs.

2. Problem Statement

For this type of problem we will be performing regression analysis on data that is very important for the health of our planet, national security and regional reliability for all persons living in the contiguous United States. Having energy independence in a reliable and renewable way could relieve stress on the total energy grid as well as energy markets while utilizing already available resources. As an economy built on innovation, this is an under-served area domestically with many foreign suppliers being used. Finding empirical evidence that regions that previously had only been served by traditional oil and gas supply could help spur local investment and domestic innovation in these areas.

Some related work in this area includes:

* Machine learning and the renewable energy revolution: Exploring solar and wind energy solutions for a sustainable future including innovations in energy storage[8]

* Development of a Hybrid Artificial Intelligence Framework for accurate forecasting of solar power generation using machine learning algorithms and time-series analysis[3]

3. Data Collection and Preparation

Data Sources

The data used in this project originates from two primary sources: the U.S. Energy Information Administration's (EIA) Open Data Electricity API and NASA's POWER Project API. These datasets were collected with the goal of creating a comprehensive view of energy supply and demand across the contiguous 48 U.S. states. This foundation supports analysis of shortfalls and the need for inter regional energy transfers.

Both datasets are publicly available and freely accessible from U.S. government entities, though use of the EIA API requires a free API key. The data can be accessed at `https://www.eia.gov/opendata/` and `https://power.larc.nasa.gov/`. We additionally use Wikipedia and OurAirports to locate the largest cities and airports in each region to obtain the latitude and longitude coordinates to feed into NASA's system to obtain the location based weather and solar data.

Data Description

Our data includes hourly data across 13 regions for 5 years. This makes for approximately 43,000 rows for each of the 13 regions. We use features from NASA for the temperature 2 meters above the surface, wind speed at 50 meters high and solar irradiance. Features obtained from the EIA includes day-ahead demand, demand, net generation total interchange,

as well as decomposed solar and wind contributions to generation. The weather data are used to determine drivers of demand like temperature as well as potential drivers to supply such as wind speed and sunlight. The energy data includes our target, net interchange, as well as other components related to the hourly status of the grid for that region. There are some regional differences due to size that likely will show some predictive errors for the same model across regions. For example, regions that generate less power and have less demand may see smaller relative needs for net interchange and lower resulting metrics as a result. We aim to normalize the measurement results when needed to make even comparisons

Some issues include NASA's dataset marking missing values as -999. A known issue involves missing GHI values beginning in August 2024 due to a satellite communication failure [7]. To mitigate this, the project sets this date as the cutoff and uses historical seasonal averages to fill intermittent gaps. EIA data has shown no missing values thus far.

Preprocessing Steps

Our first step in preprocessing the data is to define the states in each region. This was performed manually using maps on the EIA website. Once that was compiled, we use Wikipedia [10] to locate the largest airport in each region and get the related coordinates via OurAirports [5]. We then go out to the EIA site via their API and gather the previously described data using pagination of 5,000 rows at a time to not overwhelm the API. We follow a similar strategy getting NASA's data using six month increments. We then time align and merge the EIA and NASA data and finally merge the coordinates.

We do some feature engineering and feature selection at this point to create variables relating to the percent contribution of solar generation, percent contribution of wind generation, we encode some temporal features like day of week, hour, month, day of year, if it's a weekend. We also establish some rolling means and lag features to allow for seasonality to be diagnosed by the model and help with predicting the next hour based on these lags.

Lastly, we normalize only the continuous numerical features ahead of applying our RNN models. This step is not necessary for XGBoost as it is robust to outliers since it is tree based and does not use activation functions that could squish results into smaller numerical spaces.

4. Selection of Machine Learning Models

Model Consideration

For the selection of our models we had a strategy to incorporate models that would not solely be predicting based on the time series of the target but could include exogenous factors into the model. We wanted the model to understand why there was a need for interchange as a deficit in energy generation in addition to why factors such as temperature and time of day/year could play a role. As such, we eliminated statistical based historical models such as ARIMA as a pure time series analysis model.

We selected XGBoost for its fast interpretable forecasting on tabular data. Given its flexibility we added features to incorporate its understanding of time as much as possible.

We selected RNN's as they were able to remember prior states which could also assist in the understanding of the temporal component and location within the day/year for the observation state.

We selected the usage of an autoencoder to allow for some feature compression to reduce the complexity of the RNN and then be able to add multiple layers of learning to the RNN models we selected.

Final Model Selection

Our criteria for selecting the final models will be to compare the Mean Squared Error, Mean Absolute Error, Mean Absolute Percentage Error, and R-Squared for each model and each region. Our goal is to find the best model for each region. We want to compare the magnitude of the errors, the average deviation for each model, the interpretable percentage rate across regions, and the amount of variability explained by each model. Ideally, we will find a model for each region with low Mean Absolute Percentage Error. this is because the RMSE, MAE are absolute figures that will vary widely depending on how much power each region can generate. The percentage error will normalize our measurement on a per region basis.

5. Model Development and Training

Architecture and Configuration

All models are trained per region. Results are compiled for the various metrics and compared graphically.

For XGBoost we parse through each region and fit using grid search cross validation to determine which hyperparameters will lead to the best generalization of the model. We store those parameters for later so we can compare across regions and implement the model for predictions.

For the RNN's we normalize the data and then similarly parse through each region and build sequences for a 24 hour period to analyze. Given the added complexity of neural networks versus XGBoost we use optimization methods to do grid search of the hyperparameters utilizer Keras tuning methods [1].

Training Process

We use a time based train/test split where 4 years are the training data with the most recent year as the test data. This is a fairly simple strategy that makes sense since we would be making predictions in the future, the test set should be gauged for generalization and represent the most recent available data while using approximately 80 percent of our data for training [4]. For the RNN, there are validation splits carved out in the fitting of the model using the

sequential build. We also need to add dropout to the RNN when adding layers in a specific manner to maintain the previous state and avoid issue with gradient explosion/vanishing [2].

Hyperparameter Tuning

For tuning hyperparameters on XGBoost we use GridSearch CV (cross validation) in concert with Time Series Split as the cross validation framework. This allows for the cross validation to not violate the sequencing order of the original data set for the cross validation. Our parameters for grid search include the number of base estimators, max depth of each tree and the learning rate for the gradient.

For the RNN's we use KerasTuner which allows us to replace hard coded hyperparameter values with a range of possible choices. We will use Bayesian Optimization for the tuning which attempts to make predictions for which new hyperparameter values are likely to perform best given the outcomes of previous choices [1].

The autoencoder will likely require some tuning but it will also allow us to further tune the RNN's by simplying the model and allowing us computing power to add layers and go deeper.

References

[1] François Chollet. *Deep Learning with Python*. 2nd. https://learning.oreilly.com/library/view/deep-learning-with/9781617296864/Text/title.htm. Manning Publications, 2021. ISBN: 9781617296864. (Visited on 06/07/2025).

[2] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. 3rd. https://learning.oreilly.com/library/view/hands-on-machine-learning/9781098125967/preface01.html. O'Reilly Media, 2022. ISBN: 9781098125974. (Visited on 06/07/2025).

[3] Bushra Mehmood et al. *DEVELOPMENT OF A HYBRID ARTIFICIAL INTELLIGENCE FRAMEWORK FOR ACCURATE FORECASTING OF SOLAR POWER GENERATION USING MACHINE LEARNING ALGORITHMS AND TIME-SERIES ANALYSIS*. Spectrum of Enginerring Sciences Journal. https://sesjournal.com/index.php/1/article/view/396. 2025. (Visited on 05/23/2025).

[4] Aileen Nielsen. *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. https://learning.oreilly.com/library/view/practical-time-series/9781492041641/preface01.html. O'Reilly Media, 2019. ISBN: 9781492041658. (Visited on 06/07/2025).

[5] OurAirports contributors. *OurAirports: Open data for the world's airports*. OurAirports.com. https://ourairports.com/data/airports.csv. 2024. (Visited on 05/23/2025).

[6] NASA POWER Project. *NASA POWER Project API Getting Started Guide*. https://power.larc.nasa.gov. https://power.larc.nasa.gov/docs/tutorials/api-getting-started/. (Visited on 05/23/2025).

[7] NASA POWER Project. *POWER Data Sources and Methodology*. https://power.larc.nasa.gov. https://power.larc.nasa.gov/docs/methodology/data/sources/. (Visited on 05/23/2025).

[8] Abu Danish Aiman Bin Abu Sofian et al. *Machine learning and the renewable energy revolution: Exploring solar and wind energy solutions for a sustainable future including innovations in energy storage*. Sustainable Development. https://www.semanticscholar.org/paper/Machine-learning-and-the-renewable-energy-Exploring-Sofian-Lim/3a07972cb00d6da5c8641cee5e0266d86ee031ca. 2024. (Visited on 05/23/2025).

[9] U.S. Energy Information Administration. *EIA Open Data Frequently Asked Questions*. https://www.eia.gov. https://www.eia.gov/opendata/faqs.php. (Visited on 05/23/2025).

[10] Wikipedia contributors. *List of airports in the United States – Wikipedia, The Free Encyclopedia*. Wikipedia. https://en.wikipedia.org/wiki/List_of_airports_in_the_United_States. 2024. (Visited on 05/23/2025).