Comparison of Machine Learning Model Performances for Prosumer Energy Behaviour

Thomas Deckers 32659757
University of British Columbia

Jakob Khalil 86176674
University of British Columbia

Lewis Mason 52118163
University of British Columbia

Abstract

Residential solar panels provide an immense opportunity for renewable energy, but also present a significant challenge for utility companies, who must accurately forecast energy production and consumption of thousands of consumers to provide reliable service. In this paper, we compare several linear regression models and XGBoost in their ability to predict the energy production and consumption of Estonian houses and businesses. We perform manual knowledge-based feature selection and transformation, and conclude that, with restrictively-selected features, linear regression models perform similarly on this dataset regardless of regularization. We find that XGBoost performs worse than linear models in predicting energy consumption, but better in predicting production. We hope that these results provide transferable knowledge for other stochastic temporal weather-based problems.

1 Introduction

Residential solar panels are a popular way for users to improve their sustainability and reduce power costs. Many utility companies allow houses and businesses with solar panels to sell their surplus energy back to the grid. This allows users to install solar panels without needing large and costly batteries. Since homes and businesses with solar panels both produce and consume energy, they are known as "prosumers." However, prosumers also pose a key challenge to utility companies, since their energy usage and production can be highly variable. For electricity companies, having an inaccurate prediction of energy production leads to inefficient energy balancing, higher operational costs, and grid instability. In this project, we assess a variety of common machine learning (ML) algorithms on their ability to predict both the energy consumption and production of prosumers based on information about the prosumer and the weather forecast. This data is provided by Eestia Energia [1], an Estonian energy company, on Kaggle. Prior research has tested several algorithms on similar problems, however, past work has used datasets spanning a shorter time period, or methods that do not scale well. In this paper, we will establish the performance of several ML algorithms on this dataset and seek to understand the strengths and limitations of these algorithms. Furthermore, predicting the energy of prosumers is one of many highly variable, time-based, weather-dependent problems which are of great relevance to society, among others such as predicting crop production, understanding ecosystem health, and forecasting vehicle traffic. We hope that by establishing the performance of easily replicable algorithms on an example problem of this type, we can establish trends that will inform future usage of ML models.

Preprint. Under review.

2 Related Work

Recently there has been an increasing number of publications related to prosumer influences on energy grids, but none tackle the exact problem posed here, as they lack data or use an approach with scalability flaws.

Zhou et al. [5] performs a production and consumption prediction based on a method of hierarchical clustering with adaptive k-means for clients based on their individual client parameters. The method then uses a deep belief network to extract nonlinear features between the different energy types. The model predicts electrical, thermal, and gas net load. One downside of this method is that client clustering is performed at the start, and thus changing client parameters, such as upgrades to their systems, will require that initial step be re-done.

Rojek et al. [4] uses a multi-layer perceptron to predict prosumer energy consumption and production 24 hours in advance. They test XGBoost, convolutional networks, and deep learning while also tuning hyperparameters. The results state that deep neural networks have the greatest performance after cross validation. This paper, however, only contains a dataset with 30 days of data from April to May. This is likely not enough variation in weather and temperature to determine the effectiveness of the model over a full year.

Popławski et al. [3] uses lasso regression and random forests to test on prosumer data generated from Poland. They use a very small dataset consisting of a few days in November and a few days in April. Additionally, their training data is scaled to be normalized but cyclical features were not handled any differently compared to their other floating point data.

Zou et al. [6] uses edge assisted attention-based federated learning. This dataset only consisted of 20 prosumers and has a separate model trained for each of their clients. This limits the scalability because it requires significant resources to train a model for every client, which then must be retrained for changes to pre-existing clients, and will not work at all for clients that do not have the required dataset.

Extensive research has been done to predict energy usage for the entire grid, such as in Inteha et al. [2] which used a genetic algorithm-bidirectional gated recurrent unit (GA-biGRU) on a dataset extending from January 2016 to March 2020. However, these predictions were not made for individual customers and are thus less useful for gaining insight into customer behavior, and predicting the role of prosumers.

By comparing the performance of scalable models on a prosumer-level dataset that spans more than a year, we will provide new insight into energy forecasting.

3 Description and Justification

3.1 Data-set

3.1.1 Manual Feature Selection

The dataset given by Estia Energia consists of multiple csv files, with variables shown in Table 1. Of these, the most important are client.csv, train.csv, and weather_forecast.csv. Client.csv includes client specific features such as power capacity, which has the potential to change over time. Train.csv includes the hourly target variables, production and consumption, for each client over the span of 16 months from September 2021 to January 2023 which results in about 1,050,000 data points. Weather_forecast.csv contains the weather forecasts for specific counties that can be mapped to clients. Every 24 hours, a single forecast was taken to predict weather variables for every hour of the next 48 hours. Within all csv's, a variety of feature types are observed, including time-series data, numeric data, binary data, and categorical data. Due to a highly stochastic nature, the large dimensionality of the problem, and the inclusion of multiple data types, machine learning is a great application for this dataset.

A substantial part of our design process was manual knowledge-driven feature selection. We had two main reasons for omitting features provided by Eesti Energia. First, we wanted to avoid redundant features, which can circumvent regularization and lead to improper variable weighting. These features generally came in the form of duplicate data between different csv's, however, they often required

client.csv	**forecast_weather.csv**	**train.csv**	station_to_county.csv
product_type	latitude	product_type	county_name
county	longitude	county	county
eic_count	origin_datetime	datetime	latitude
installed_capacity	hours_ahead	data_block_id	longitude
is_business	**temperature**	is_business	
date	dewpoint	prediction_unit_id	
data_block_id	**cloudcover_high**	**consumption**	
	cloudcover_low	**production**	
	cloudcover_mid		
	cloudcover_total		
	10_metre_u_wind		
	10_metre_v_wind		
	data_block_id		
	forecast_datetime		
	direct_solar_radiation		
	surface_solar_radiation_downwards		
	snowfall		
	total_precipitation		

Table 1: The raw data

special care to ensure that time was correctly accounted for, which is explained later in this section. Next, features were omitted based on relevance to the target variables. We first removed the entire sets of data gas_prices.csv, electricity_prices.csv, as well as historical_weather_data.csv. These were not shown in Table 1 as they were removed. We removed gas_prices.csv and electricity_prices.csv because we assume there is a low causation between the price of gas, electricity and the amount of energy a client's solar panel generates or how much electricity they use. Generally, clients have daily routines for electricity and do not modify them based on electricity prices. Electricity consumption is largely driven by heating costs, which has a stronger correlation with seasonal shifts such as winter, and thus should be captured by the date and weather forecasts. We omitted historical_weather_data.csv because there is a high chance that the model would learn to give this an extremely high weight and predict purely based on the previous day's weather. This essentially creates a "lag" machine that learns the best loss can be achieved by predicting the results of old time-series data. In fact, in figure 10 of [5] we can see this effect, it seems that prediction is just the target data shifted by a few hours, or lagged, with some additional slope information. Finally, we removed variables from our feature set that served as linker keys to different csv's, as well as variables we deemed to not be dominant contributors to the target parameters, or those that may not generalize well to all prosumers in a given category.

To summarize our feature selection, all variables kept as inputs for our model are bolded in table 1. At a high level, these include the type of business, type of product, county of client, county weather, date, and time.

3.1.2 Feature Matrix Generation

We then combined the variables into one feature matrix. For client_info.csv, we were able to match clients to their electricity production and consumption based on the provided variables. However, transforming the weather forecast into features had a further complication; while we only knew the county of the clients, many counties have multiple weather stations with separate forecasts. To address this, we only used the weather forecast from the most geographically central weather station in each county, measured by latitude. We also transformed the provided date and time into three features: the day of the year (0-364), day of the week (0-6), and hour (0-23).

We chose to predict energy consumption and production each with their own model, as these two targets are largely independent from one another. By doing so, we are simplifying the problem, which enables simpler models to perform better, reducing our chances of overfitting. Additionally, for models that are slower than O(n) for fitting or testing on n examples, splitting the data will improve computational time.

Variable	Encoding	Number of features
Hour	Cyclical	2
Day of week	Cyclical	2
Day of year	Cyclical	2
County	One hot	16
Is business	Binary	1
Product type	One-hot	4
EIC count	Numerical	1
Installed capacity	Numerical	1
Temperature	Numerical	1
Dewpoint	Numerical	1
Cloud cover low	Numerical	1
Cloud cover mid	Numerical	1
Cloud cover high	Numerical	1
Direct solar radiation	Numerical	1
Surface solar radiation downwards	Numerical	1
Snowfall	Numerical	1
Total precipitation	Numerical	1

Table 2: Feature transformations

The provided training dataset contained entries with missing information. Furthermore, the data did not include a one-day weather forecast for the last day of the training set, and the first 5850 rows of the training dataset did not have a corresponding entry in client_information.csv. Since each entry is independent and we have no lag features, we chose to discard these subsets of data. This still left us with just over one million examples.

We further transformed the temporal features of the matrix so that they would integrate better with our models. For cyclical features (day of year, hour, and day of week) we encoded them into two new features by taking the sine and cosine. For categorical features (county and product type), we used a one-hot encoding. For the remaining numerical features (including our wave-encoded temporal features), we standardized to a mean of 0 and standard deviation of 1. In cross validation, this standardization was always done for the training dataset, with the training mean and standard deviation then applied to the validation set. This creates a total of 38 features, shown in Table 2.

3.2 Methods

We hid data from October 1st, 2022 to January 18th, 2023, representing about 30% of the dataset, to be used as a test set.

Using this test set, we compared the performance of 5 separate ML models: ridge regression, lasso regression, Huber regression, elastic net, and XGBoost (Table 3). These were implemented in python using the scikit-learn and XGBoost packages, and trained on servers provided by Google Colaboratory. The four different linear regression models allowed us to test how performance is affected by regularization. Since these are well-established and well-understood parametric models, they were easy to implement and relatively quick to train. XGBoost is a commonly used tool in Kaggle competitions, and served as a valuable point of comparison to a completely separate but very standard non-linear approach. Finally we used the mean and median of the full training dataset as a baseline prediction to compare our models with.

The approach for evaluating machine learning models involved a careful 4-fold cross-validation process that emphasized the preservation of the temporal order in the training data. This method entailed a step-by-step training and validation procedure, starting with training on the first fold and validating on the rest, and then progressively including more folds in the training set while reducing the validation set correspondingly. This approach was crucial for ensuring that models were trained on historical data and validated on future data. This aligns with the goal of forecasting future power consumption and production based on observed historical trends. Additionally, we employed a randomized search over a wide range of hyper-parameter configurations for each model, rather than an exhaustive search, in order to efficiently explore the configuration space of each model. This process involved 20 iterations of randomized cross-validation searches, resulting in 80 trained instances per model. The time taken for this process was varied; linear models were quick to train, while

Algorithm	Loss	Reguralization	Tuned Hyperparameters
Ridge	L2	L2	Alpha
Lasso	L2	L1	Alpha
Huber	L1 (Huber approx.)	L2	Alpha, epsilon
Elastic net	L2	L2+L1	Alpha, L2/L1 ratio
XGBoost	L2	N/A	Number of trees, depth, learning rate

Table 3: Models and tuned hyper-parameters

models such as XGBoost required much more computational time due to the complexity of training many trees. The best configurations for each model were then identified, and the corresponding models were retrained on the entire training set before being saved for evaluation against our hidden test dataset. Certain models that we were interested in exploring, such as kernel-based models and neural networks, were excluded due to the lack of GPU and TPU acceleration support in SKLearn's implementations. Training these models on our CPU rendered the training and cross-validation process impractically time consuming and did not align with our time constraints, which eventually led to their omission from the final results.

4 Analysis

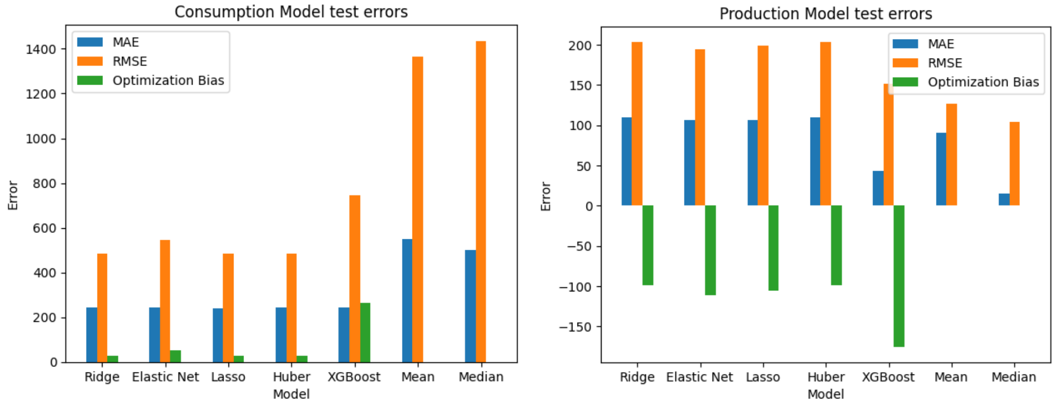

Figure 1: Test error plots

For the electricity consumption dataset, our main results are shown in Figure 1. All 5 ML models performed significantly better than the baseline. All four linear regression models had very similar performance, with an MAE within 3% of one another. This shows that the regularization used had a minimal effect on performance, suggesting that all features were meaningful for making predictions. The one notable exception to this trend was that elastic net had a 13% worse RSME than the other linear models. Additionally, elastic net had the highest optimization bias of the linear models, which is reasonable since it had an additional hyperparameter to tune. In this case, it appears that it led to overfitting to the validation set. XGBoost performed substantially worse than the linear models, and had a very high optimization bias. XGBoost has three hyperparameters, so it is not surprising to see such a high optimization bias, which in this case, was detrimental to test performance. This could have been combated by using more folds in our cross validation, however we were restricted by our hardware capacity.

Our results from the electricity production dataset showed two surprising results. First, we found that all of our models had a negative optimization bias, which is to say that they performed better in the test set in comparison to the cross validation scores. We suspect that this was due to the temporal nature of the data, and the fact that it does not span much more than a year - since our folds are temporally restricted, the first training fold would only contain only data from e.g. september to october, which would create a model that performs poorly on data from the rest of the year, contributing to a high validation error. Since electricity production through solar panels is more seasonally dependent than electricity consumption, this effect was magnified in the production case. The second surprising result was that all our models performed considerably worse than both baseline cases. In fact, the best entries in the Kaggle competition as of the time of writing have a mean average error of 62.2 across consumption and production combined, whereas predicting the median for energy production has a mean average error of just 14.6. The low baseline error shows that energy production has a low overall variance. Otherwise, the results follow the exact opposite trends to the consumption dataset: all linear models perform about the same, with elastic net doing slightly better, owing to a lower optimization bias, whileXGBoost outperforms all of the linear models considerably. One reason that we might expect XGBoost to perform well here is that many target values for electricity production are zero, namely at night and when there's heavy cloud cover. This data structure is very naturally represented by a tree, and is much harder to capture with linear models. This effect may have been strengthened by the fact that our test set spans October to January, where we expect more samples with zero production due to shorter days. However, we also note that tree-based models do not perform well in applications with unbounded variables. For instance, with improving technologies, we expect that the average installed solar panel capacity will increase over time. When encountering larger values than those that exist in the training set, a tree-based model will be unable to extrapolate.

5 Discussion and Future Work

This paper provides a real world application of predicting current estimations for a target variable (consumption and production) based on stochastic but cyclic variables (day, time) and estimations of other variables for the specific time (weather forecast). It has been shown that through careful feature selection, a variety of linear models perform nearly identical, despite having different regularizations. Thus, the method of manual feature selection based on intuition was warranted, as feature removal methods like L1 regularization could not find useless enough features to remove.

While it is nice that all features are being used, it may be the case that adding more features would decrease the test loss achieved. This is something that could be tested with more time as additional data requires careful consideration for how it is introduced and normalized

For our dataset, linear models have a lower generalization gap than non-linear models for the consumption model, whereas the opposite is true for the production model.Trying a more complex model that is able to perform changes of the data basis, such as neural networks, could be beneficial. With more time we could perform hyperparameter tuning of such models for comparison.

We hope that our approach can serve as an example for future investigations into similar problems.

References

[1] Kristjn Eljand et al. *Enefit - Predict Energy Behavior of Prosumers*. 2023.

[2] Azfar Inteha et al. "A Data Driven Approach for Day Ahead Short Term Load Forecasting". In: *IEEE Access* 10 (2022), pp. 84227–84243. DOI: `10.1109/ACCESS.2022.3197609`.

[3] Tomasz Popławski, Sebastian Dudzik, and Piotr Szeląg. "Forecasting of Energy Balance in Prosumer Micro-Installations Using Machine Learning Models". In: *Energies* 16.18 (2023). ISSN: 1996-1073. DOI: `10.3390/en16186726`. URL: `https://www.mdpi.com/1996-1073/16/18/6726`.

[4] Izabela Rojek et al. "Machine Learning- and Artificial Intelligence-Derived Prediction for Home Smart Energy Systems with PV Installation and Battery Energy Storage". In: *Energies* 16.18 (2023). ISSN: 1996-1073. DOI: `10.3390/en16186613`. URL: `https://www.mdpi.com/1996-1073/16/18/6613`.

[5] Bin Zhou et al. "Multi-energy net load forecasting for integrated local energy systems with heterogeneous prosumers". In: *International Journal of Electrical Power Energy Systems* 126 (2021), p. 106542. ISSN: 0142-0615. DOI: `https://doi.org/10.1016/j.ijepes.2020.106542`. URL: `https://www.sciencedirect.com/science/article/pii/S0142061520322626`.

[6] Luyao Zou et al. "Edge-assisted Attention-based Federated Learning for Multi-Step EVSE-enabled Prosumer Energy Demand Prediction". In: *2023 International Conference on Information Networking (ICOIN)*. 2023, pp. 116–121. DOI: `10.1109/ICOIN56518.2023.10048987`.

Appendix

Model	Consumption	Production
Ridge	455.94	302.73
Elastic Net	494.08	306.22
Lasso	457.50	305.47
Huber	455.99	302.83
XGBoost	480.79	327.36
Mean	1364.39	126.93
Median	1435.54	104.35

Table 4: Cross validation RMSE of best models

	Ridge	Elastic Net	Lasso	Huber	XGBoost	Mean	Median
MAE	245.28	243.93	237.94	245.27	242.40	550.51	500.17
RMSE	485.10	546.27	484.97	485.13	744.04	1364.39	1435.54
MSE	235324.53	298411.56	235194.84	235351.20	553600.42	1861548.43	2060776.64

Table 5: Test error for energy consumption models

	Ridge	E. Net	Lasso	Huber	XGB	Mean	Median
MAE	109.76	106.18	106.26	109.76	43.56	90.21	14.62
RMSE	203.60	194.93	199.31	203.60	151.64	126.93	104.35
MSE	41453.89	37997.30	39724.47	41454.00	22994.30	16110.58	10889.90

Table 6: Test error for energy production models