Prediction of Electricity Consumption

Denis Mitana and Miroslav Sumega

Slovak University of Technology in Bratislava, Ilkovičova 2, 842 16 Bratislava, Slovakia

Abstract. Predicting electricity consumption is not a simple task. To do that, you have to train artificial intelligence models on lots of data to predict accurately. However these data are specific in terms of containing building types and climatic conditions and might not be available. Therefore it is important to build general models which are able to solve these issues. In order to tackle this problem, we use Support Vector Regression (SVR), Extremely Randomized Trees (EXRT) and Extreme Gradient Boosting (XGBoost) models to predict electricity consumption of multiple buildings from ASHRAE - Great Energy Predictor III data. We use model trained on one climatic zone to predict electricity consumption of buildings from different climatic zones to prove that it is possible to achieve acceptable results when predicting for buildings which model has never seen. We also found out that most important features are values of previous consumption. With our best model, which is XGBoost, we achieved results of 18.67 RMSE, 9.38 % NRMSE and 1.16 RMSLE.

Keywords: Electricity consumption prediction · Extremely Randomized Trees · XGBoost.

1 Introduction

A lot of investments are being made to improve building efficiencies, but are the improvements working? Our work is focused on answering this question. Nowadays, buildings are built with these improvements so real energy consumption is available. But we are not able to easily obtain energy consumption without any retrofits to deduce how much was building efficiency improved. In our work, we predict these values in order to show the difference between real energy consumption and energy consumption without any retrofits to let building owners realize how much they saved. With better estimates of these energy-saving investments, large scale investors and financial institutions will be more inclined to invest in this area to enable progress in building efficiencies. Current methods do not scale very well. Some assume a specific meter type or do not work with different building types. We try to overcome these issues.

Our task is prediction of the energy consumption for multiple buildings for one hour ahead. It is a regression task. To fulfill this task we analyzed our data in order to find any potential problems. Then we focused on the data preprocessing to remove these problems. After that we tried to train the best model using machine learning algorithms to predict the energy consumption. It was followed by analysis of results.

2 Related Work

In order to decide which algorithms we will use for prediction of consumption, we analyzed existing solutions. In [6] they used bagging ensembles of various algorithms including SVR, XGBoost and EXRT which were most accurate. Weights of algorithms in ensemble were optimized with biologically inspired algorithms. They achieved 3.677 % MAPE. Predictions were done for one day ahead. RBF-like neural network was used in [8] to predict one day ahead in hourly intervals. Achieved error was 3.87 % MAPE. Authors in [9] used Fuzzy Cognitive Maps with Structure Optimization Genetic Algorithm to make predictions for next three steps. They achieved 0.077 MAE. Bayesian Networks were used in [1] to predict for 15 minutes and one hour ahead. For hour ahead they achieved 7.20 % NRMSE. In [7] Improved Elman Neural Network was used for predictions. They achieved 3.03 % NRMSE when predicting for two days ahead with hourly resolution. Self-recurrent Wavelet Neural Network trained by Levenberg-Marquardt algorithm was used in [3]. They achieved 2.29 % NRMSE for day ahead predictions with hourly resolution.

3 Data Analysis

The dataset for this problem is publicly available on Kaggle and is associated with the competition[1]. It includes three years of hourly meter readings from over one thousand buildings at several different sites around the world. By problem definition only one year of these three years is intended for training and the remaining two years are intended for testing.

Except hourly meter readings the dataset contains also hourly weather information and buildings information. Weather data contains information about air and dew temperature, wind speed and direction, precipitation, cloud coverage and site of measurement. Building data contains information about the gross floor area of the building, number of its floors, year it was built and also for what purpose the building is used. It also contains information about the site it is in. This allows for mapping of weather data for one site to multiple buildings in that same site. During data analysis we mostly focused on missing values, outliers and correlations between attributes.

We found out that values of some attributes are missing. From buildings data it is information about the number of its floors and the year the building was built, where more than 50 % of values are missing. However, we can not fill in these missing values because it is not correct to use information about other buildings to do it. From weather data it is mainly information about cloud coverage where almost half of the data is missing. We are not considering using attributes where more than 50 % of values are missing. Precipitation data was missing approximately 36 % of all data therefore we will decide whether to use it or not.

[1] `https://www.kaggle.com/c/ashrae-energy-prediction/overview`

As outliers we considered values outside of $< 0.25\ quantile - 1.5 * IQR$, $0.75\ quantile + 1.5 * IQR >$, where IQR is *Inter Quartile Range*. Most outliers were present in the weather data. These are values in air temperature and sea level pressure attributes. Outliers were also present in building data, specifically in square feet attribute. There were buildings, which were more than 4 times the median of square feet. They represent around 10 % of all buildings.

We also looked at how attributes correlate with each other and with the target attribute. We found out that there are not many correlations except air and dew temperature. However it might be caused by the fact that there are various buildings from various areas together in the dataset and therefore the correlation might not have been calculated correctly.

The target value, which we will try to predict, is the measured energy consumption of the building. However there are four types of meters in the dataset. We therefore analyzed these meters and decided to use only the electricity meter, because it represents slightly more than half of all measured values. Most records have meter reading values smaller than 200. Therefore, we may not use records which have the meter reading value above 200.

4 Data Preprocessing

To be able to use some machine learning algorithm, we preprocessed data and chose relevant attributes. In preprocessing we mainly filled in missing values, replaced outliers, encoded categorical attributes using One Hot Encoding and scaled values to zero mean and unit variance. Missing values are filled in using rolling average with 5 hour window. Upper and lower outliers are replaced by 95th percentile and 5th percentile, respectively. After preprocessing of buildings, weather and meter data, it is all merged together[2]. In the following sections we describe selected attributes and their preprocessing in more detail.

Buildings data From buildings data we chose only *primary use* and *square feet* attributes. We merge less numerous categories of *primary use* attribute to category *Other* and encode values. *Square feet* attribute is scaled.

Weather data From weather data we chose *air temperature*, *dew temperature*, *sea level pressure*, *wind speed* and *wind direction* attributes. Each of them is scaled and has filled in missing values except *sea level pressure*. For this attribute we were not able to fill in all missing values, because for the 5th site there are no values of it. Therefore we dropped this attribute. We also created a variant where outliers of *air temperature* and *dew temperature* are replaced.

Meter data From meter data we chose only the electricity meter readings. To avoid outliers we take only records where *meter reading* is smaller than 200. We also append 5 previous readings to each record, in order to simulate a time series. At first we scaled *meter reading* attribute, but as experiments revealed, that was not a good idea, because predicted values become negative after scaling back. Therefore, we dropped scaling of *meter reading* attribute.

[2] It is important to note that timestamp of weather data has to be shifted one hour ahead to be able to predict the energy consumption for one hour ahead.

5 Methods

We experimented with SVR, EXRT and XGBoost algorithms. We chose these algorithms because they had a good prediction performance in existing works focused on a task of predicting electricity consumption.

SVR [4] is based on Support Vector Machine (SVM) algorithm and it is used for regression instead of classification. SVR uses the same principles as the SVM with only a few differences. SVR keeps in mind that part of the error is tolerated when minimizing it utilizing the hyperplane which maximizes the margin. The disadvantage is that training time increases more than quadratically[3] with respect to the number of training data and we have quite a lot of training data.

EXRT [5] is an ensemble method based on combining many weak trees to one strong predictor. Combination of weak trees is based on training each tree on random features and random decision boundaries of trees.

XGBoost algorithm [2] is highly optimized gradient boosting with some improvements to algorithm as well. Gradient boosting is a method of combining weak predictors into one strong predictor. It is based on training model, evaluating it and training new model on data on which previous model performed poorly. Then it repeats the same process for combination of trained models. That way overall performance is improved by adding new models. Gradient boosting uses gradient descent to evaluate how good models fit data. XGBoost adds regularization, sparsity awareness, weighted quantile sketch and cross validation to gradient boosting to improve prediction accuracy.

6 Experiments

By task definition the dataset is already split into train and test sets. The test set is unlabelled and is used to evaluate the competition. Since there are no predefined labelled validation and test sets, we used 5-fold cross validation on training data to train and evaluate our solution. Shown results are therefore means of 5 runs.

We evaluate our methods on known regression metrics *Root Mean Square Error* (RMSE) and *Normalized RMSE* (NRMSE). We also use *Root Mean Square Logarithmic Error* (RMSLE) because it is used as scoring metric at Kaggle competition. The following equations describe these metrics:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}, \tag{1}$$

$$NRMSE = \frac{RMSE}{y_{max} - y_{min}} * 100, \tag{2}$$

[3] https://www.thekerneltrip.com/machine/learning/computational-complexity-learning-algorithms/

Table 1. Performance of algorithms using default hyperparameters. * denotes outliers presence in the data.

Algorithm	RMSE	NRMSE	RMSLE
EXTR	34.84	17.59	1.84
EXTR*	34.78	17.56	1.86
XGboost	**22.34**	**11.25**	**1.26**
XGboost*	24.06	12.13	1.33

$$RMSLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(log(\hat{y}_i + 1) - log(y_i + 1))^2}, \tag{3}$$

where N is the size of training data, $\hat{y}_i$ is the i-th predicted value and y_i is the i-th target value. We do not use *Mean Average Percentage Error* because the target value can be zero and therefore we would have to replace it with another value or remove given sample, which in turn would affect resulting error.

We conducted multiple experiments to choose the best machine learning algorithm and to optimize its hyperparameters. In each experiment we used preprocessed data as described in section 4. It is important to note that we experimented only with the electricity meter readings lower than 200 and buildings from site 0. Size of our training set is approx. 605K records.

6.1 Basic Experiments

Sanity check was performed using SVR algorithm. We used *scikit-learn*[4] implementation with default hyperparameters. To reduce the number of training data and thus training time we used only buildings with *building id* lower than 10. Resulting number of training data was approx. 62K. The goal was to prove correctness of our data preprocessing and it was proven. On the other hand, training took approx. 20 minutes on a tenth of our training data which would cause enormously long training time on all training data due to SVR's time complexity. Therefore, we did not use SVR in any other experiment.

Next, we compared EXRT[5] and XGBoost[6] algorithms. Both algorithms were used with default hyperparameters and trained with and without outliers. XGBoost performed significantly better than EXTR and with replaced outliers even better (tab. 1). We consider default hyperparameter settings to be reasonable for comparing performance of algorithms. So in further experiments we used only XGBoost algorithm and whereas it performed better with replaced outliers, we continued to use only data preprocessing which replaces outliers.

[4] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR`

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor`

[6] `https://xgboost.readthedocs.io/en/latest/python/python_api.html?highlight=xgbregressor#xgboost.XGBRegressor`

Table 2. Performance of XGBoost algorithm when optimizing *booster* hyperparameter.

booster	RMSE	NRMSE	RMSLE
gbtree	22.34	11.25	1.26
gblinear	**21.34**	**10.71**	**1.26**
gbdart	22.34	11.25	1.26

Table 3. Performance of XGBoost algorithm using *gblinear* booster and optimizing its hyperparameters, namely *feature selector* (FS), *learning rate* (α) and *number of estimators* (N).

FS	α	N	RMSE	NRMSE	RMSLE
shuffle	0.1	100	21.49	10.77	1.28
cyclic	0.1	100	21.46	10.77	1.28
cyclic	0.1	200	21.36	10.72	1.26
cyclic	0.3	200	**21.34**	**10.71**	**1.26**
cyclic	0.6	200	21.34	10.71	1.26
cyclic	0.3	300	21.34	10.71	1.26

6.2 Hyperparameters Optimization

We optimized only XGBoost's hyperparameters, because it outperforms other algorithms with default settings. As optimization method we chose Grid Search. XGBoost has quite a lot of hyperparameters so we performed multiple runs and designed settings according to results of previous run. Due to space limitation, tables with results show only the most important settings and not all of them.

We started with *booster* hyperparameter, because set of other hyperparamers is based on used booster. Since *gblinear* booster performed best (tab. 2), we optimized its hyperparameters, namely *feature selector*, *learning rate* and *number of estimators*. Using *cyclic* feature selector, more estimators and larger learning rate results in better performance (tab. 3). However further increase of learning rate and number of estimators no longer worked and performance did not change. In addition, achieved results (tab. 3) are equal to results achieved with default hyperparameters (tab. 2). Since this optimization did not improve anything, we decided to optimize *gbtree* booster. We experimented with *learning rate*, *number of estimators*, *L2 regularization* and *max depth*. This optimization (tab. 4) was successful and we outperformed baselines with default hyperparameters (tab. 2). Using shallower trees, more estimators and lower learning rate improved performance. It was also useful to turn off L2 regularization. We expected that using more estimators would result in better performance, but we also expected that this improvement would not be so large, hence we ended up with hyperparameters optimization and consider this as the best achieved results.

Table 4. Performance of XGBoost algorithm using *gbtree* booster and optimizing its hyperparameters, namely *learning rate* (α), *number of estimators* (N), *L2 regularization* (λ) and *max depth* (MD).

α	N	λ	MD	RMSE	NRMSE	RMSLE
0.01	100	1	6	26.65	13.36	1.04
0.1	100	1	6	19.66	9.88	1.16
0.1	100	1	3	18.84	9.46	1.16
0.1	100	0	3	18.81	9.45	1.16
0.1	500	0	3	**18.67**	**9.38**	**1.16**
0.3	500	0	3	19.94	10.02	1.21
0.1	100	0	2	19.41	9.75	1.20

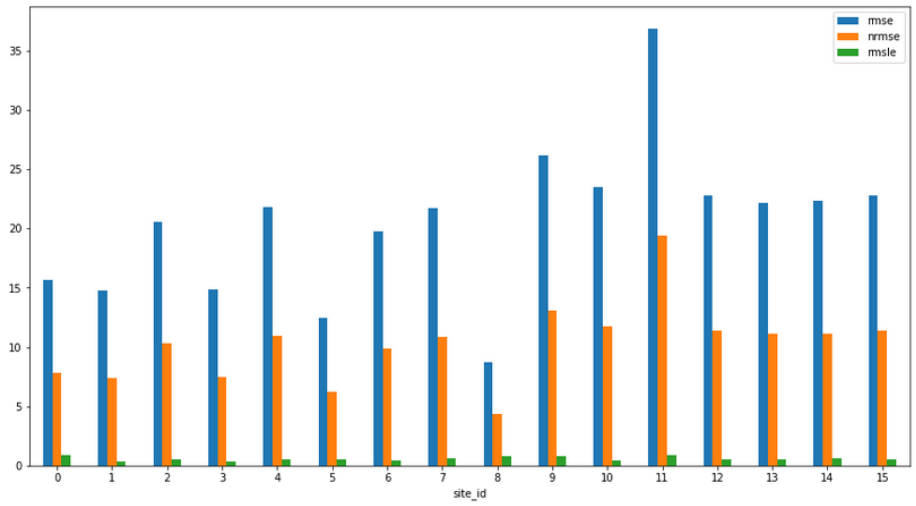

Fig. 1. Performance of model trained on site 0 and evaluated on all sites.

6.3 Evaluation

In order to evaluate our best model, we performed experiments to see how good is our model on data from different sites with different climatic conditions and which features were most important to better explain our model.

Model Generalization To perform this experiment, we used our best model trained on site with ID 0 and then evaluated its performance on other sites.

Model trained on site 0 has better performance on sites 1, 3, 5 and 8 on all metrics when compared to performance on site 0 (fig. 1). This is probably caused by the fact that weather patterns and buildings in those sites are similar to site 0 with site 8 being almost identical. Stable weather or consumption patterns might also be the cause that predictions are more accurate that on site 0.

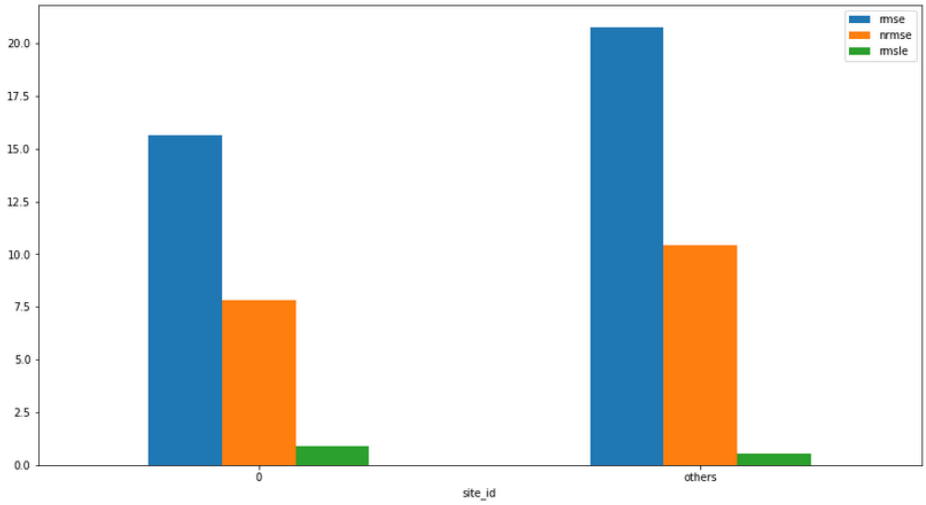

Fig. 2. Model performance comparison on site 0 and aggregated other sites.

However on sites 9 - 15 there is large decrease in accuracy of predictions. This is probably caused by different weather conditions. Site 0 is somewhat warm, with no measurements under $-5\ °C$ however sites 9 - 15 are shifted towards lower temperatures, where minimum is approximately $-20\ °C$. It might also be caused by unstable consumption patterns.

Overall performance (fig. 2) of our model is better on site 0 than on other sites. This is expected result since our model was trained on site 0. However decrease in performance is small, therefore our model can be used in different climatic conditions.

Feature Importance In this experiment we looked at which features were most important for our model.

Feature importance experiment (fig. 3) shows that most important features were previous meter readings. Of these, meter reading from previous hour was the most important by far. This is probably caused by the fact that it should be quite similar to predicted value since we predict only for one hour ahead. We can see that importance of previous meter reading decreases when going further into the past except for increase for 5 hours before predicted value.

Dew temperature is the most important from all weather features. Air is more humid when dew temperature is higher, therefore it should transfer heat better than dry air. As a result, buildings may need to use air conditioning or heaters to balance temperature inside. That is probably the cause why our model considers dew temperature as a most important weather feature.

At last from buildings features the most important is the feature describing whether building is used for offices. However there should be approximately the same number of each type of building, therefore we do not think the that number

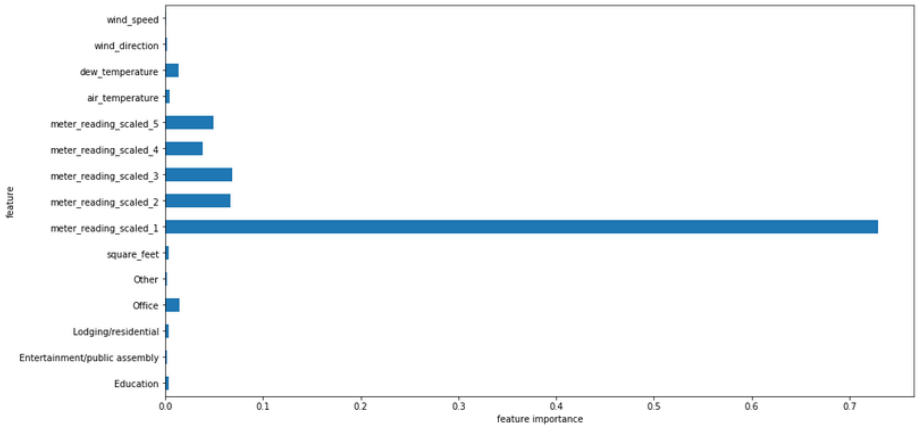

Fig. 3. Feature importance of model trained on site 0.

of buildings per type is the cause of increased importance. We guess that Office feature is most important of building features because office buildings are usually used for work and when combined with meter reading, model might be able to tell if work is going to end or already did and therefore consumption will decrease or whether work starts and consumption will increase.

Comparison with Existing Solutions Since our model is trained only using electricity meter readings and one site, we can not directly compare achieved results with the leaderboard at Kaggle. Also we evaluated our solution using 5-fold cross validation and we did not use provided test dataset. Apart from differences, our best achieved RMSLE is 1.16 and it is comparable result considering Kaggle's best result, which is 1.231 RMSLE.

To our best knowledge there is no paper using the same dataset for direct comparison. But considering related work in electricity consumption prediction, our best achieved NRMSE, which is 9.38 %, is on the same scale as their.

7 Conclusion

In this work, we presented models for predicting electricity consumption in buildings for one hour ahead based on SVR, EXRT and XGBoost algorithms using buildings and weather information along with previous consumption values. However we used XGBoost in most of experiments, because it was most accurate. In experiments, we addressed issues of predicting electricity consumption for different types of buildings in different climatic conditions. Although we are not able to compare directly with existing solutions, based on performed evaluation of results, we suppose that our solution is correct and it would achieve competitive results also in large scale. Some of challenges left for the future is to

handle multiple meter types and utilize other sites to make model more general and robust.

References

1. Bassamzadeh, N., Ghanem, R.: Multiscale stochastic prediction of electricity demand in smart grids using bayesian networks. Applied energy **193**, 369–380 (2017)
2. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. pp. 785–794 (2016)
3. Chitsaz, H., Shaker, H., Zareipour, H., Wood, D., Amjady, N.: Short-term electricity load forecasting of buildings in microgrids. Energy and Buildings **99**, 50–60 (2015)
4. Drucker, H., Burges, C.J., Kaufman, L., Smola, A.J., Vapnik, V.: Support vector regression machines. In: Advances in neural information processing systems. pp. 155–161 (1997)
5. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1), 3–42 (2006)
6. Halaš, P., Lóderer, M., Rozinajová, V.: Prediction of electricity consumption using biologically inspired algorithms. In: 2017 IEEE 14th International Scientific Conference on Informatics. pp. 98–103 (2017)
7. Liu, Y., Wang, W., Ghadimi, N.: Electricity load forecasting by an improved forecast engine for building level consumers. Energy **139**, 18–30 (2017)
8. Nguyen, D.H., Nguyen, A.T.: An approach for the electricity consumption prediction based on artificial neural network. In: 2019 SICE International Symposium on Control Systems (SICE ISCS). pp. 78–83 (2019)
9. Poczeta, K., Papageorgiou, E.I., Yastrebov, A.: Application of fuzzy cognitive maps to multi-step ahead prediction of electricity consumption. In: 2018 Conference on Electrotechnology: Processes, Models, Control and Computer Science (EPMCCS). pp. 1–5 (2018)