Assignment 5 Project - Predicting Energy Demand

Karan Kapur (1008384874) and Keagan Rankin (1008578692)

1 Part B: problem statement

Anticipating energy demand is increasingly essential for both private and public entities. For private companies, predicting demand can help mitigate risk and optimize power usage/sell back to the grid for profit. For the government, knowing future demand helps them predict buyer response and the best time to import/export power from neighboring provinces. As renewable energy and battery deployment start to dominate the energy sector, it will also become increasingly important to predict demand in order to facilitate grid support services (e.g. energy buy-back from batteries, two-way power flow from solar PV). In response, we propose two machine learning models for predicting future energy demand using open-source data provided by provincial and federal agencies. We train our models on hourly data from 2018-2023 (~50,000 data points), engineer features using time-series properties and dimensional reduction of climatic data, and deploy the model using Azure's cloud framework and AutoML.

2 Data set and exploratory analysis

2.1 Power data

All power data was taken from Ontario's Independent Electricity System Operator (IESO). They have well-documented, hourly data for energy demand (in MWh), power price (in $/MW), as well as net imports and exports to the province between 2018-2023 (Figure 1).

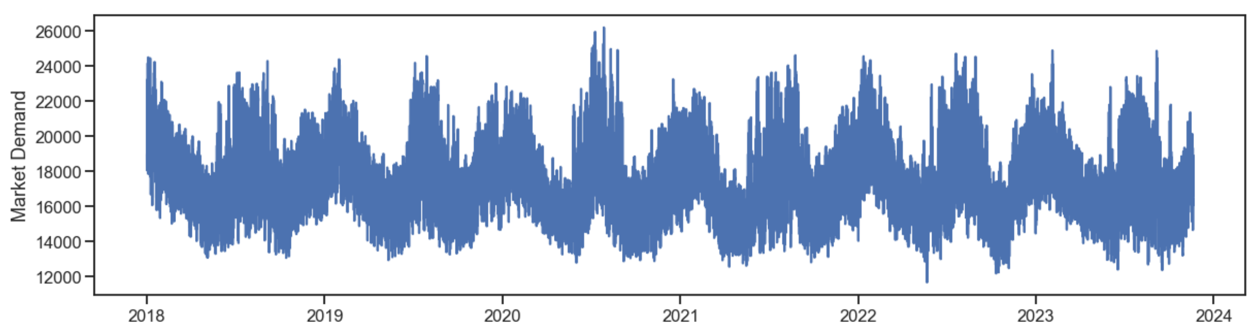


Figure 1 - Ontario hourly demand in Megawatts from 2018-2023

Figure 2 shows that demand for power is relatively stationary across years, with some outlying dates in 2020 (likely due to the pandemic). Demand for power is lowest overnight and peaks around 6-8pm when people arrive home from work. This daily demand trend curve is consistent across many years.

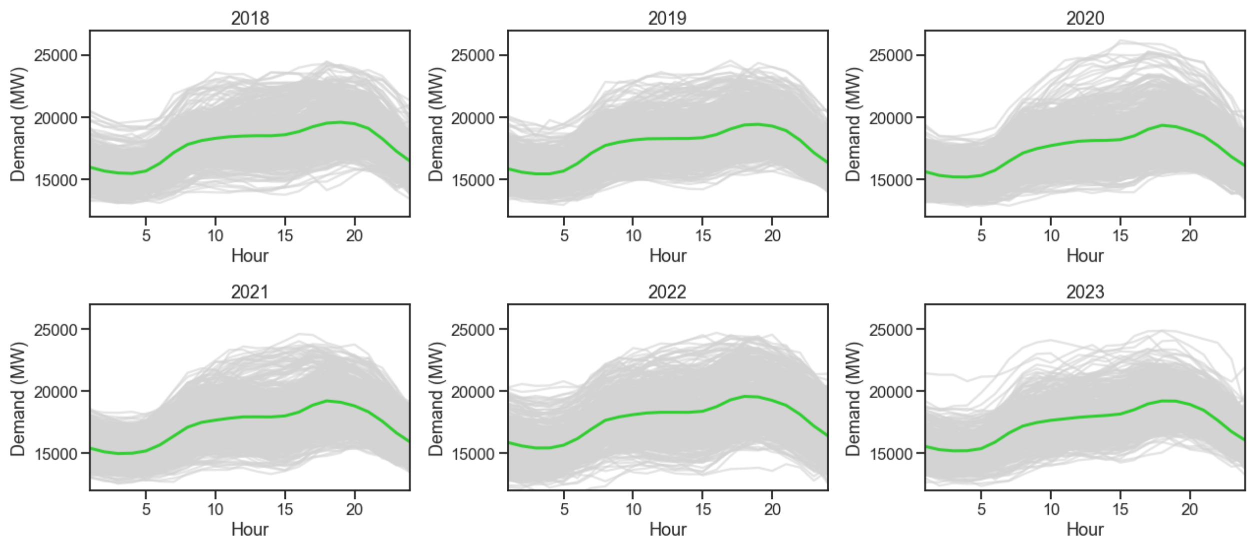


Figure 2 - Daily Ontario power demand. The individual time-series for each day are plotting in grey, and the mean is plotting in green.

By contrast, a variable like energy price is much more inconsistent. Prices see spikes which tend to happen around large changes in demand and energy import/exports to the province (dubbed “ramp” by the IESO and other energy operators) (Figure 3).

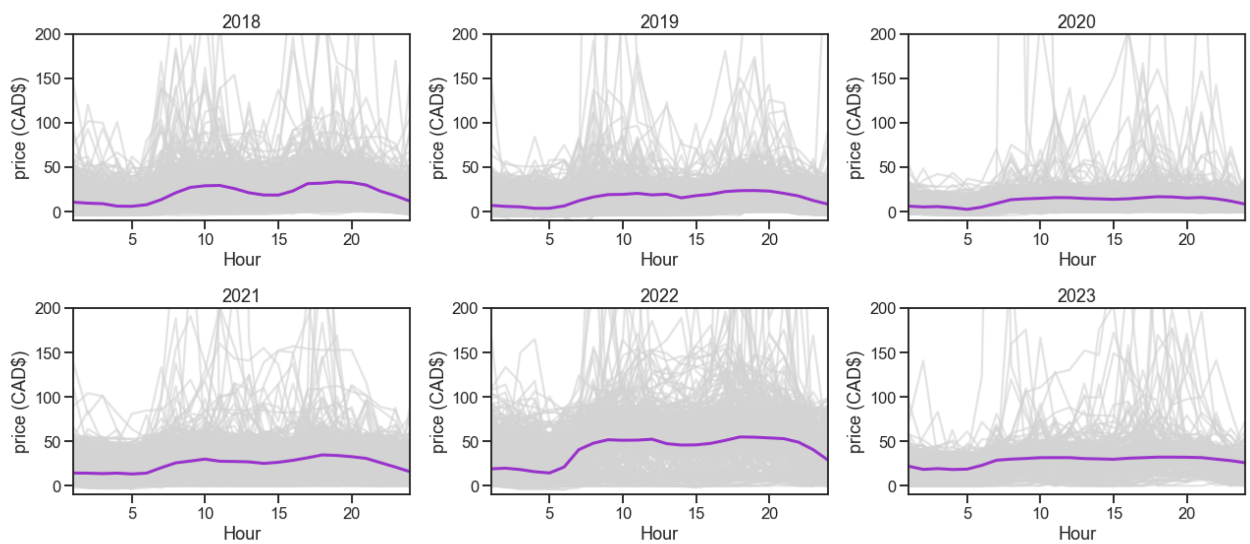


Figure 3 - Daily Ontario energy price. The individual time-series for each day are plotting in grey, and the median is plotting in purple.

2.2 Weather data

Climatic data comes from Environment and climate change Canada. We take the daily temperature readings from Pearson International Airport and join them to the hourly power data using datetime features in Pandas. Figure 4 shows the first two components of a singular value decomposition on climatic data. Seasonal variations are the largest drivers of variance in the data, with temperature and Month (both correlated) changing with component 1.

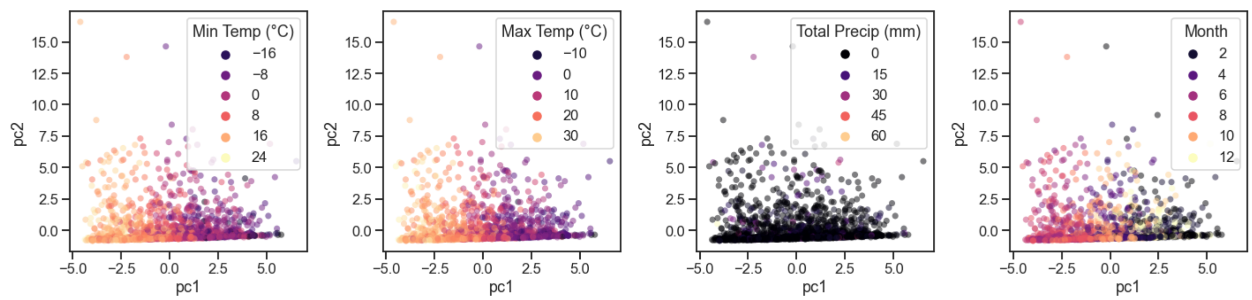


Figure 4 - PCA of Pearson Airport daily climatic data from 2018-2023. Specific variable values are overlaid on each subplot and indicated in the legends.

2.3 Data cleaning and split

After the initial exploratory analysis above, we combined all of the data based on a datetime (as it came from multiple sources). We split the data into training and testing sets according to best practices with time series. The training set contained data from 2018 up to Sept 2022, and the test set contained data from Sept 2022 to November 2023. We then created a function to clean and impute missing data (which was concentrated in the climatic features). Table 1 summarizes our cleaning/imputation method.

Table 1 - Missing values and imputation method

Feature	**Imputation**	**Reasoning**
Temperature and heating/cooling degree days	Monthly median	The monthly average temperature is a good guess at the temperature of any given day. We saw no correlation between extreme weather events (e.g. heavy precipitation) and missing data.
Precipitation (rain and snow)	Zero	We saw no correlation between extreme weather and missing precipitation data. We assume that these missing values are days when sensors were taken down for repair, meaning there was probably no precipitation (there were only around 14 missing days)

Maximum daily gust speed and gust direction	K-nearest neighbourhood	There were more missing values for the maximum wind gust and direction. We did not have good priors for this data, so we filled it using K-NN on other weather data (e.g. assuming that variables like min temp may be associated with the wind direction)

3 Feature engineering and Results

3.1 Choosing features and time-series dependence

We used the training set to explore several features. Most of the time series had some autocorrelation as shown in Figure 5. The periodicity ended up being useful as we used 24-hour lagged values to predict the values for the next day.

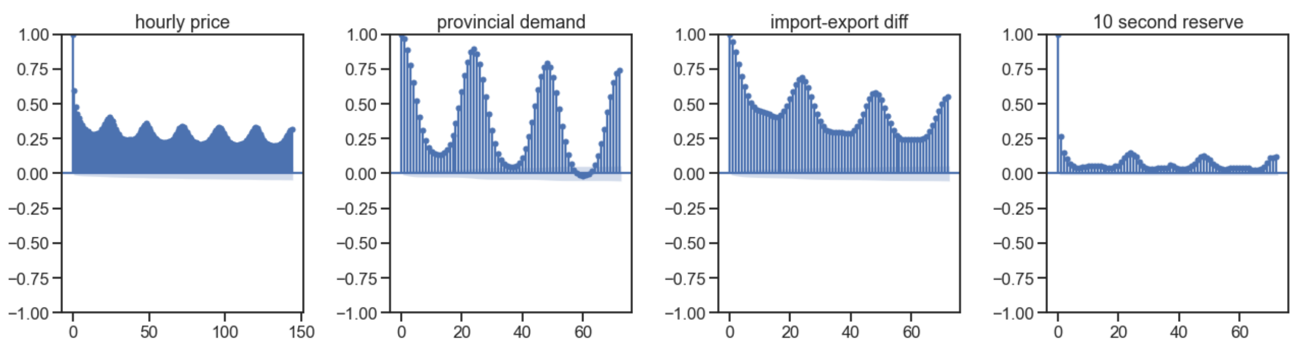


Figure 5 - Autocorrelation of IESO time-series data. Price data demonstrates no more autocorrelation if differenced, while other variables do.

From the IESO time-series data, we chose the following lagged features to predict demand:

- The demand from 24 hours before the prediction.
- The 3-hour moving average from 23, 24, and 25 hours for 2, 3, and 4 days before the prediction.

We chose to not use features from the day we were trying to predict because 1) we wanted to avoid information leakage (specifically from weather data) 2) we wanted our model to forecast an entire day ahead to improve its flexibility for real-world use cases, and 3) lagged features closer to the time of prediction (e.g. using the demand from the previous hour as a variable) actually resulted in a worse model that was overpowered by this predictor and resembled a random-walk prediction. We found that lagged price was not a very significant feature for demand, nor was other data from

IESO (demand could be quite inelastic to price as big consumers like industrial operations need energy to function through the day regardless of price).

We also analyzed the climatic data in relation to demand. We included the previous day's cooling degree days, gust direction and the month of prediction as features in our model as they were strongly correlated with Demand over a year period (Figure 6)

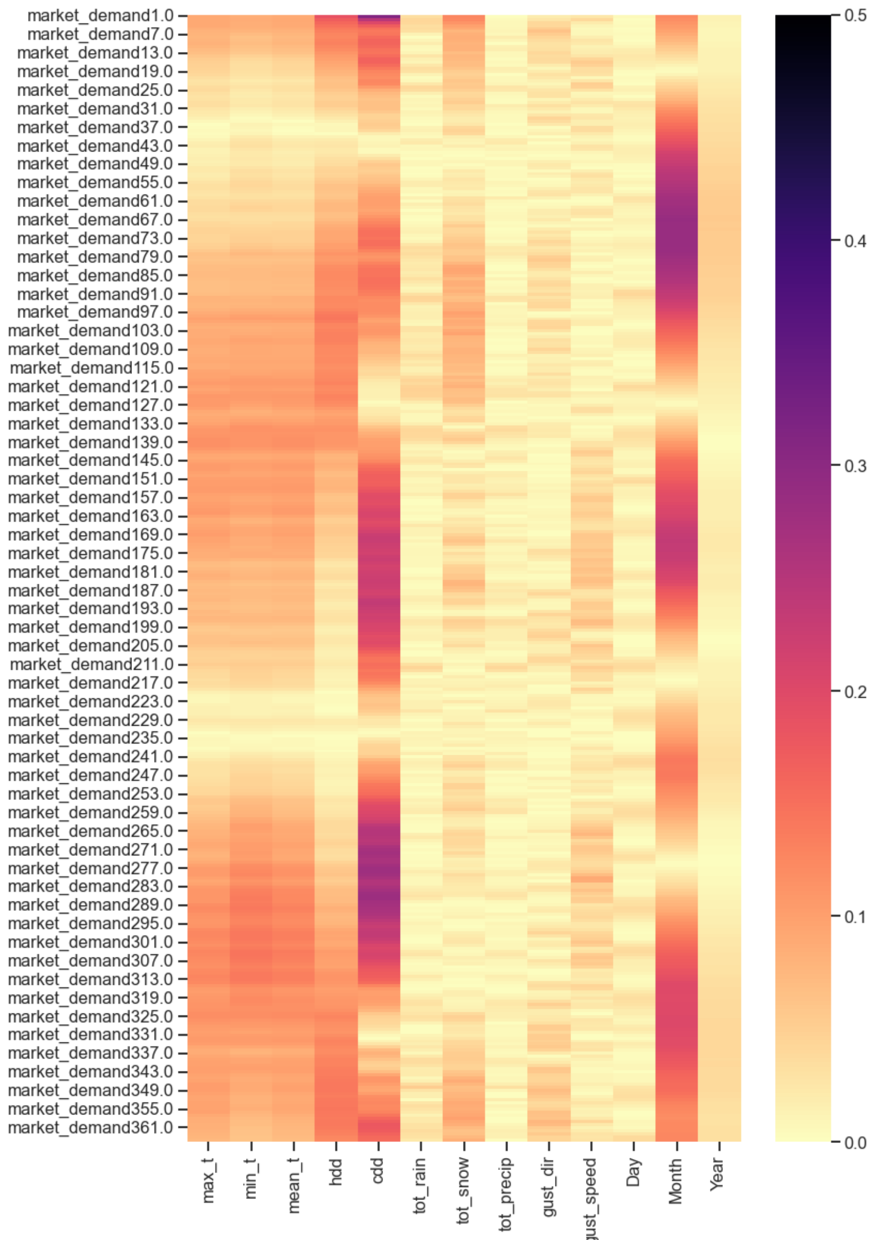


Figure 6 - Correlation of demand N-hours back with different climatic variables. The legend shows the Pearson R coefficient.

3.2 Two models and results

We chose to apply a linear regression and a gradient boosting regression model to our data. These models were easy to implement in scikit-learn on Azure and were of appropriate complexity for our 7 feature model (XGBoost could be overkill). We fit our baseline linear model using L^1 (lasso) regularization with α = 1.0. Our model achieved a mean squared error (MSE) of 953086 (≡ 976 MW off on average) and an R^2 of 0.78 on the test set. These errors were a bit lower than how the model performed in validation on just the training set. Figure 7 shows results of the one-day ahead linear model predictions.

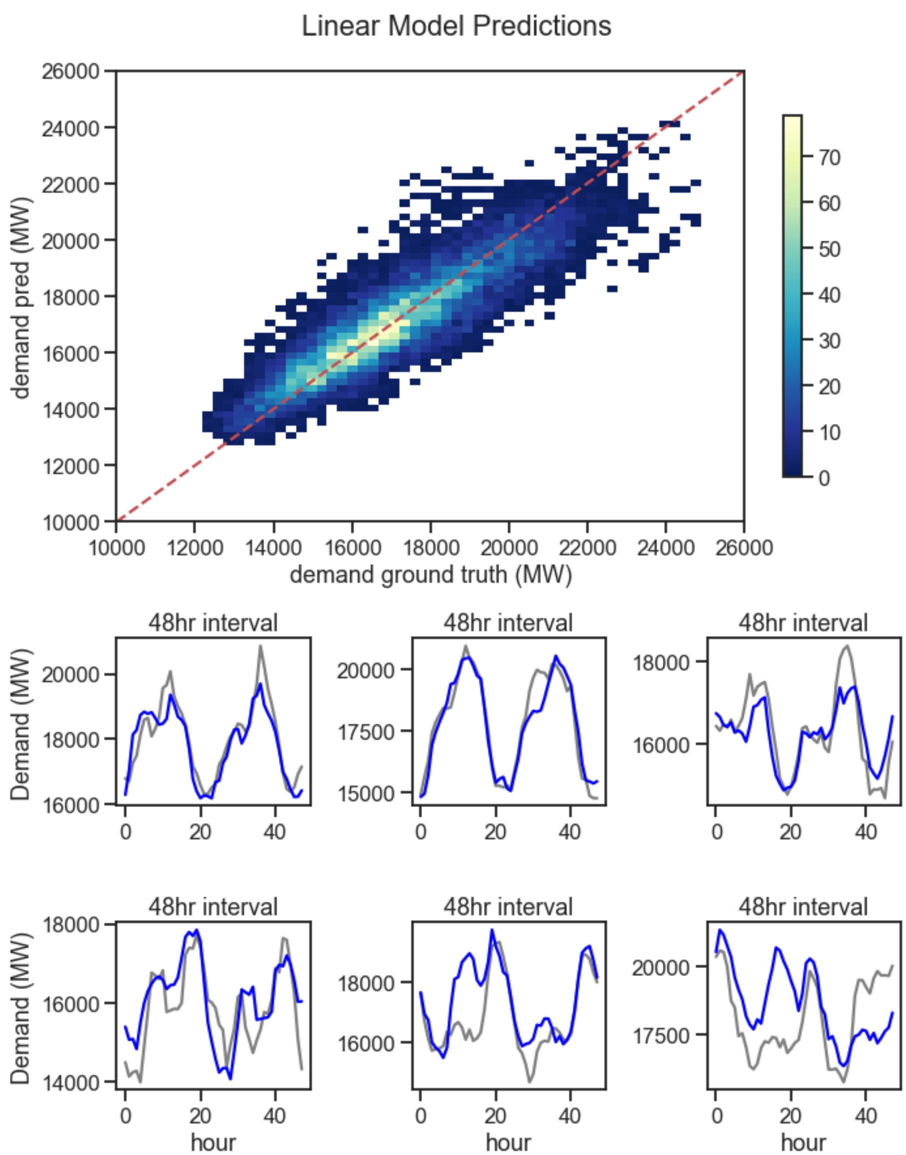


Figure 7 - Results of testing the linear model. The six plots below are random draws of 48-hour intervals from the test set, with predictions in blue and ground-truth in grey.

Our gradient boosting model was initialized with a minimum sample split+leaf of 5 and limited to using 5 features per ensemble unit to avoid overfitting. It did only slightly better than the linear model (without hyperparameter tuning), achieving a MSE of 909269 (average error of 950 MW) and an R^2 of 0.79 on the test set. Figure 8 shows results for the gradient boosting model.

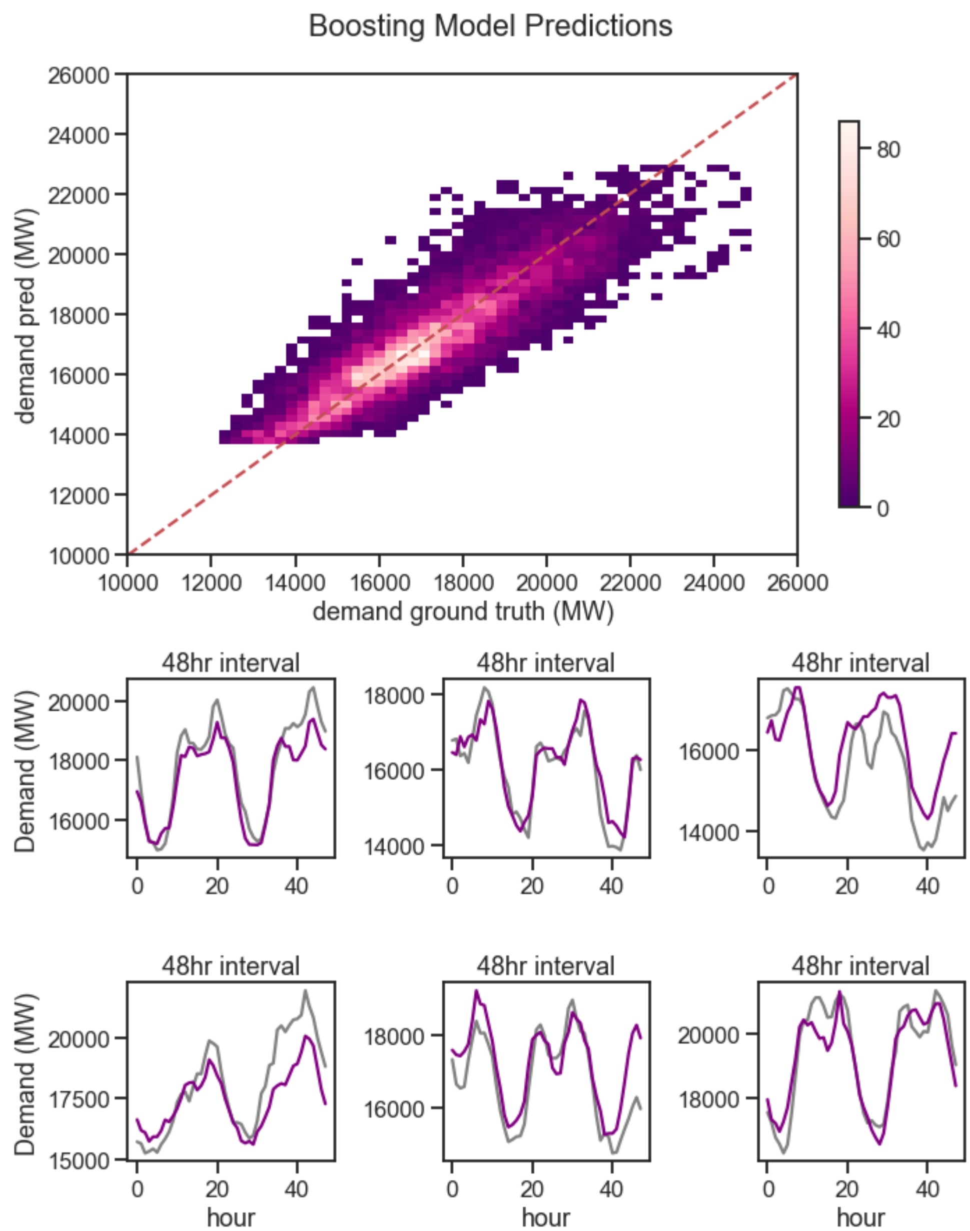


Figure 8 - Results of testing the gradient boosting model. The six plots below are random draws of 48-hour intervals from the test set, with predictions in purple and ground-truth in grey.

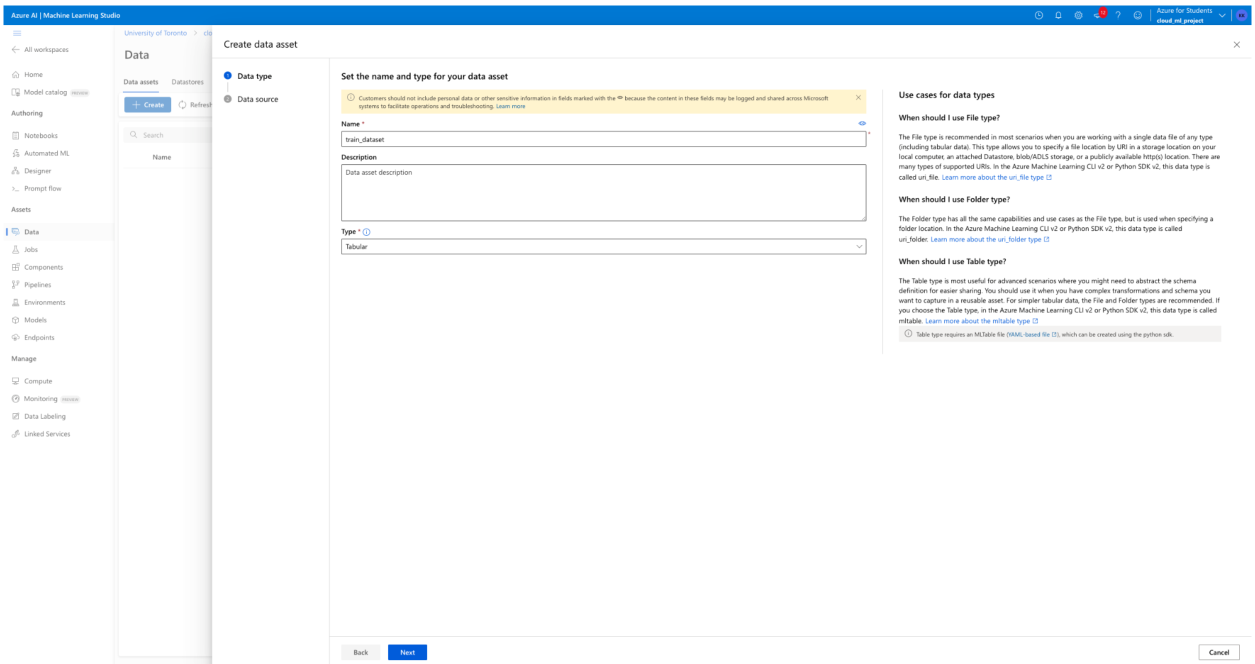
3.3 Improving predictions

We theorized methods of improving our demand prediction. Improved features could include one-hot encoded dates like holidays and weekends as well as more diverse climatic data from around Ontario. More lagged features farther back in time could also help improve predictions (though would cut off more training data at the beginning of the feature set). Supply data was poorly-organized on the IESO website but could be cleaned and used to improve demand estimates.

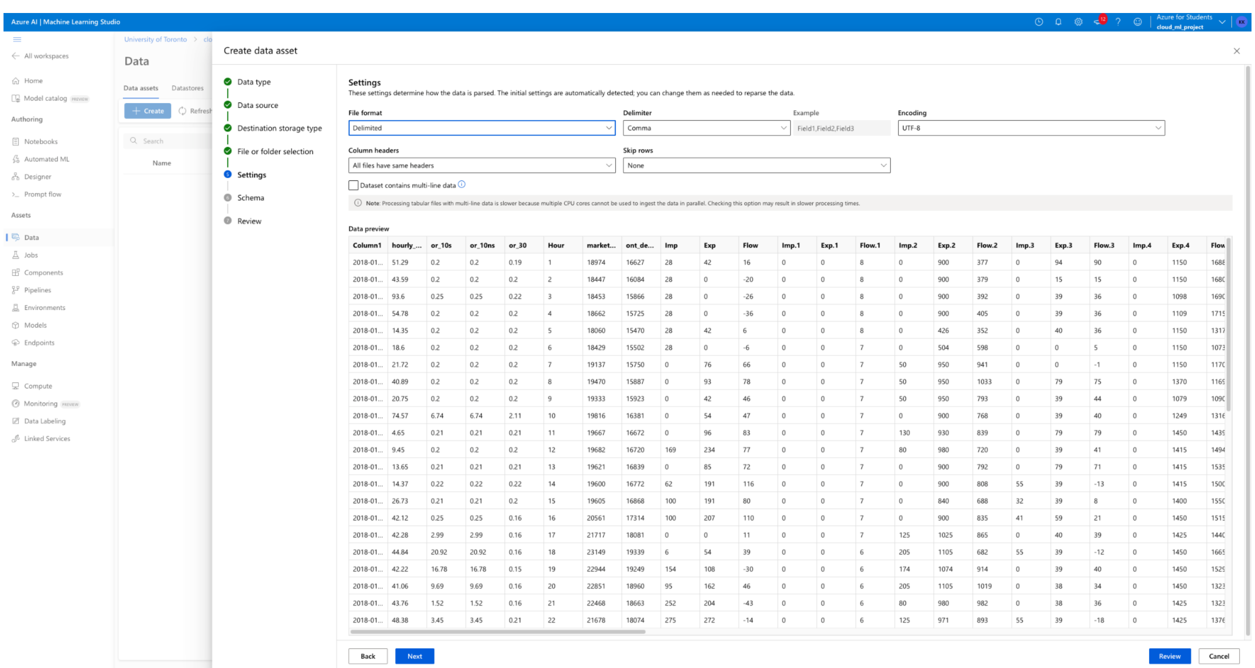
Different models could probably also improve predictions. XGBoost could improve over our current gradient boosting model (see AutoML implementation below). A linear regression with ARIMA errors (ARIMAX) might capture the autocorrelation of the time series better. The time series could also be fed into a deep-learning architecture (e.g. LSTM) which might achieve marginal gains.

4 AutoML implementation

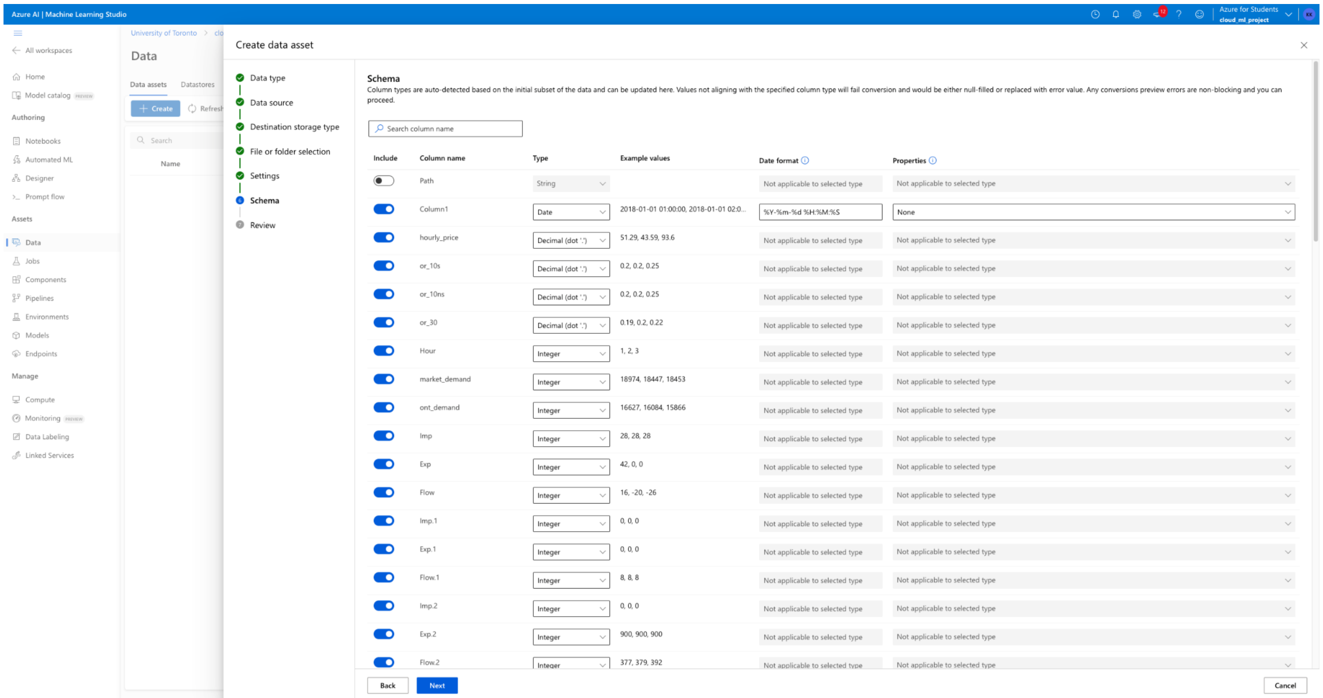
Create data asset:



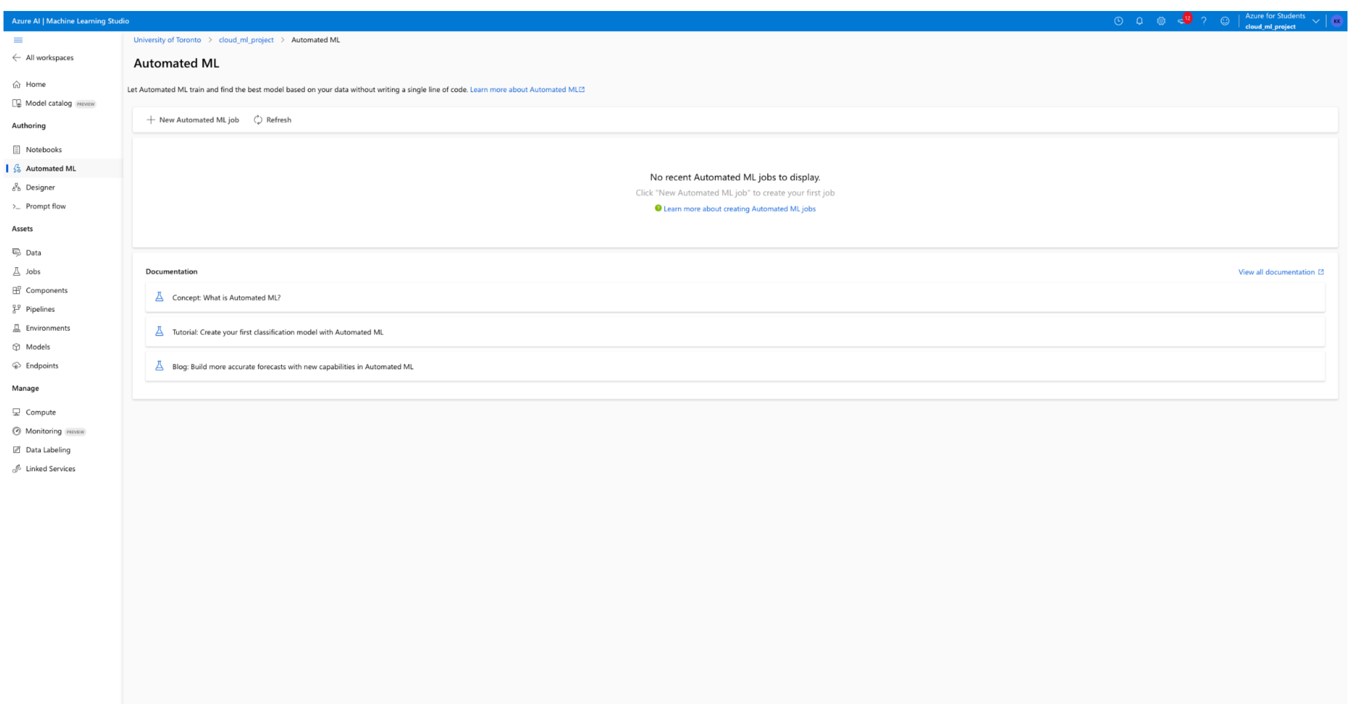
Reviewing dataset:



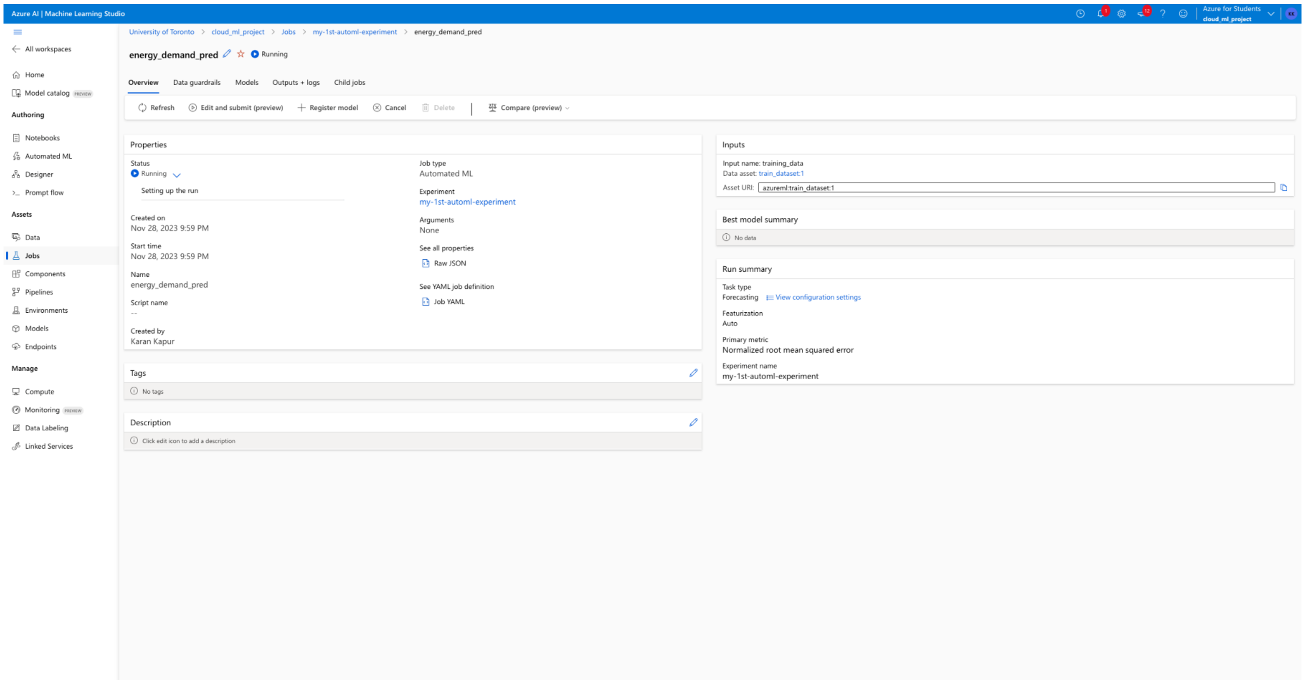
Reviewing schema:



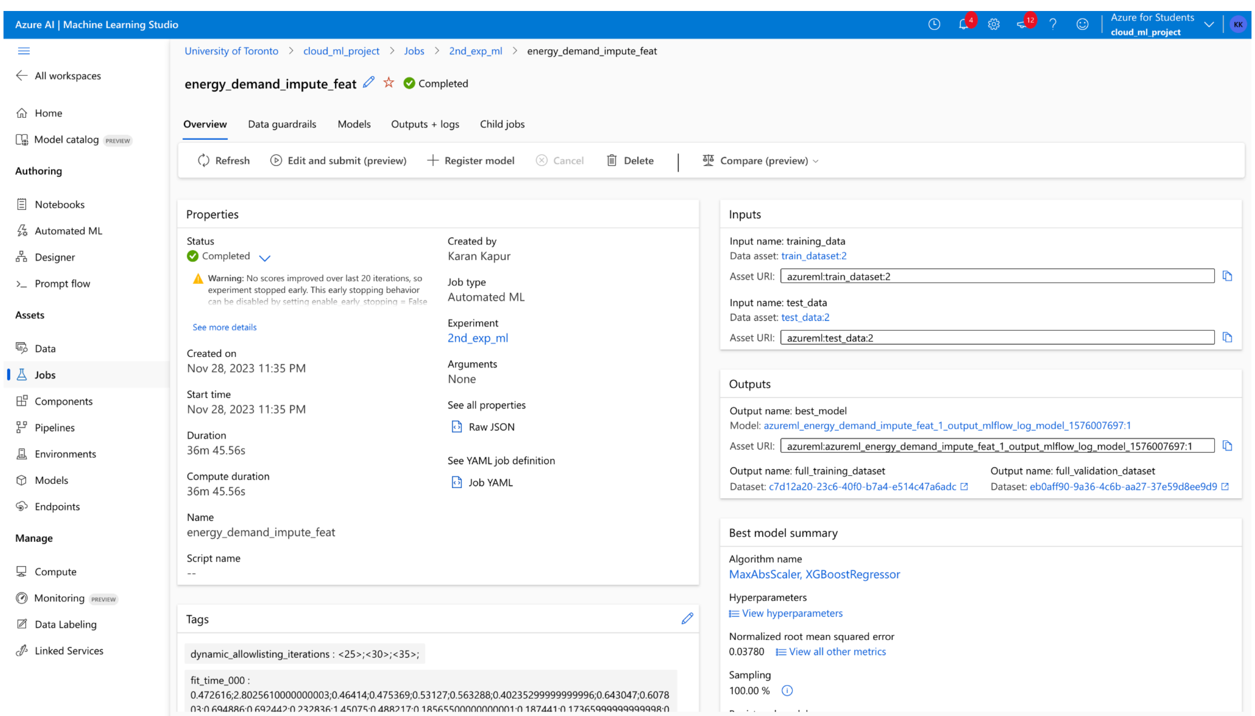
Created workspace and entered the AutomatedML job centre:



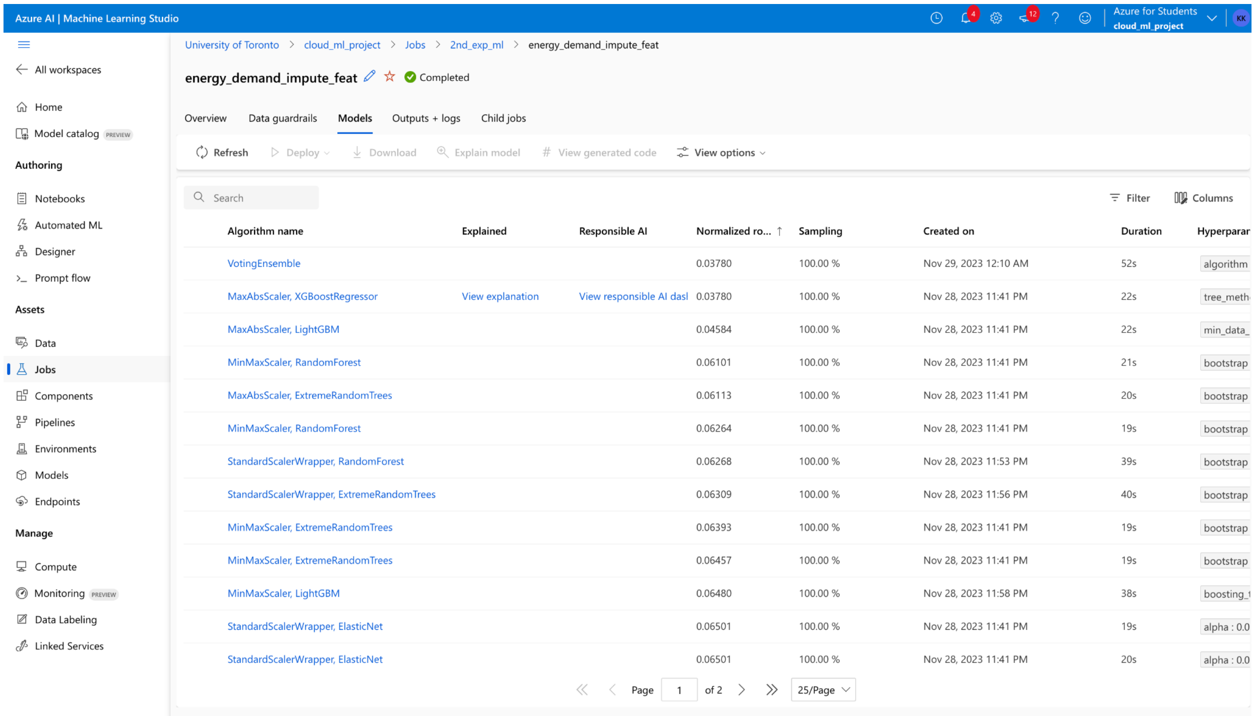
AutomatedML started:



Model training completed:



Models page:


Azure AI | Machine Learning Studio
Azure for Students
cloud_ml_project
All workspaces
Home
Model catalog PREVIEW
Authoring
Notebooks
Automated ML
Designer
Prompt flow
Assets
Data
Jobs
Components
Pipelines
Environments
Models
Endpoints
Manage
Compute
Monitoring PREVIEW
Data Labeling
Linked Services
University of Toronto > cloud_ml_project > Jobs > 2nd_exp_ml > energy_demand_impute_feat
energy_demand_impute_feat
Completed
Overview
Data guardrails
Models
Outputs + logs
Child jobs
Refresh
Deploy
Download
Explain model
View generated code
View options
Search
Filter
Columns
Algorithm name | Explained | Responsible AI | Normalized ro... | Sampling | Created on | Duration | Hyperparar
VotingEnsemble | 0.03780 | 100.00 % | Nov 29, 2023 12:10 AM | 52s | algorithm
MaxAbsScaler, XGBoostRegressor | View explanation | View responsible AI dasl | 0.03780 | 100.00 % | Nov 28, 2023 11:41 PM | 22s | tree_meth
MaxAbsScaler, LightGBM | 0.04584 | 100.00 % | Nov 28, 2023 11:41 PM | 22s | min_data_
MinMaxScaler, RandomForest | 0.06101 | 100.00 % | Nov 28, 2023 11:41 PM | 21s | bootstrap
MaxAbsScaler, ExtremeRandomTrees | 0.06113 | 100.00 % | Nov 28, 2023 11:41 PM | 20s | bootstrap
MinMaxScaler, RandomForest | 0.06264 | 100.00 % | Nov 28, 2023 11:41 PM | 19s | bootstrap
StandardScalerWrapper, RandomForest | 0.06268 | 100.00 % | Nov 28, 2023 11:53 PM | 39s | bootstrap
StandardScalerWrapper, ExtremeRandomTrees | 0.06309 | 100.00 % | Nov 28, 2023 11:56 PM | 40s | bootstrap
MinMaxScaler, ExtremeRandomTrees | 0.06393 | 100.00 % | Nov 28, 2023 11:41 PM | 19s | bootstrap
MinMaxScaler, ExtremeRandomTrees | 0.06457 | 100.00 % | Nov 28, 2023 11:41 PM | 19s | bootstrap
MinMaxScaler, LightGBM | 0.06480 | 100.00 % | Nov 28, 2023 11:58 PM | 38s | boosting_
StandardScalerWrapper, ElasticNet | 0.06501 | 100.00 % | Nov 28, 2023 11:41 PM | 19s | alpha : 0.0
StandardScalerWrapper, ElasticNet | 0.06501 | 100.00 % | Nov 28, 2023 11:41 PM | 20s | alpha : 0.0
Page 1 of 2
25/Page

Best model is XGBoostRegressor and following are the model details:

Hyperparameters:

Data Transformation:

Data transformation:

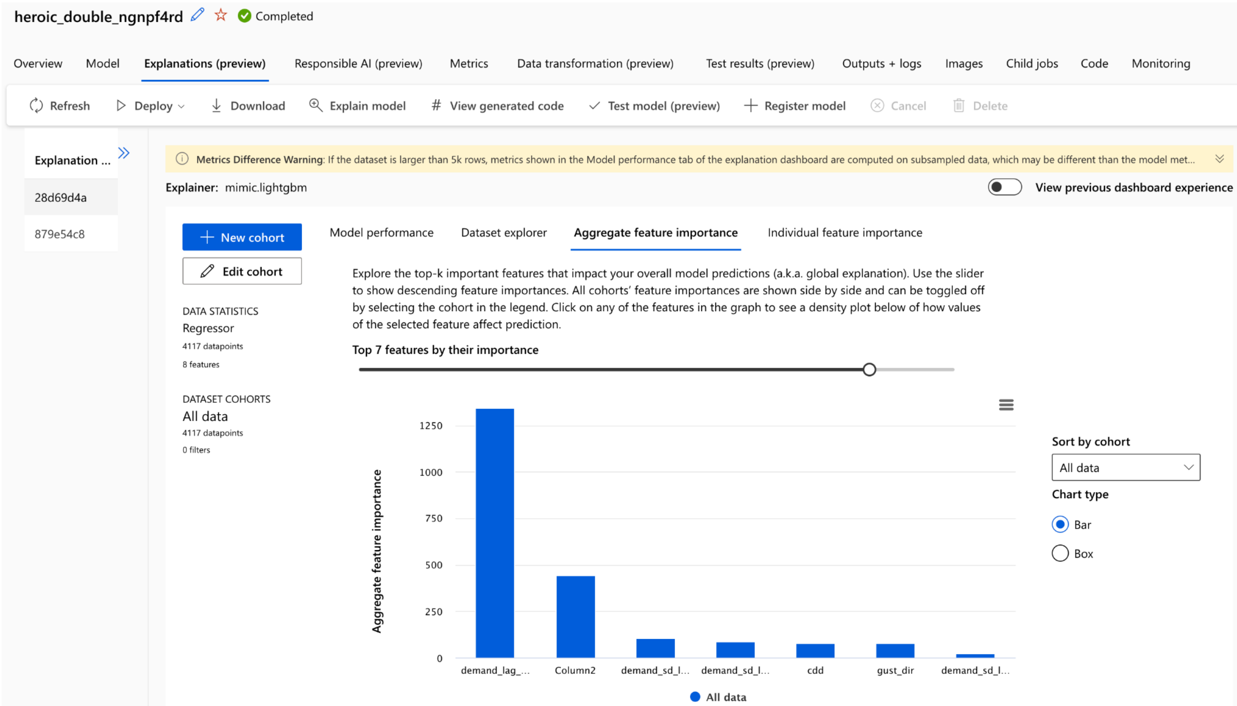

```
{
    "spec_class": "preproc",
    "class_name": "MaxAbsScaler",
    "module": "sklearn.preprocessing",
    "param_args": [],
    "param_kwargs": {},
    "prepared_kwargs": {}
}
```


Training algorithm:

Training algorithm:


```
{
    "spec_class": "sklearn",
    "class_name": "XGBoostRegressor",
    "module": "automl.client.core.common.model_wrappers",
    "param_args": [],
    "param_kwargs": {
        "tree_method": "auto"
    },
    "prepared_kwargs": {}
}
```

Model explanations:



Other metrics

Run Metrics

Explained variance
0.93916

Mean absolute error
423.08

Mean absolute percentage error
2.4016

Median absolute error
340.15

Normalized mean absolute error
0.029144

Normalized median absolute error
0.023431

Normalized root mean squared error
0.037798

Normalized root mean squared log error
0.038180

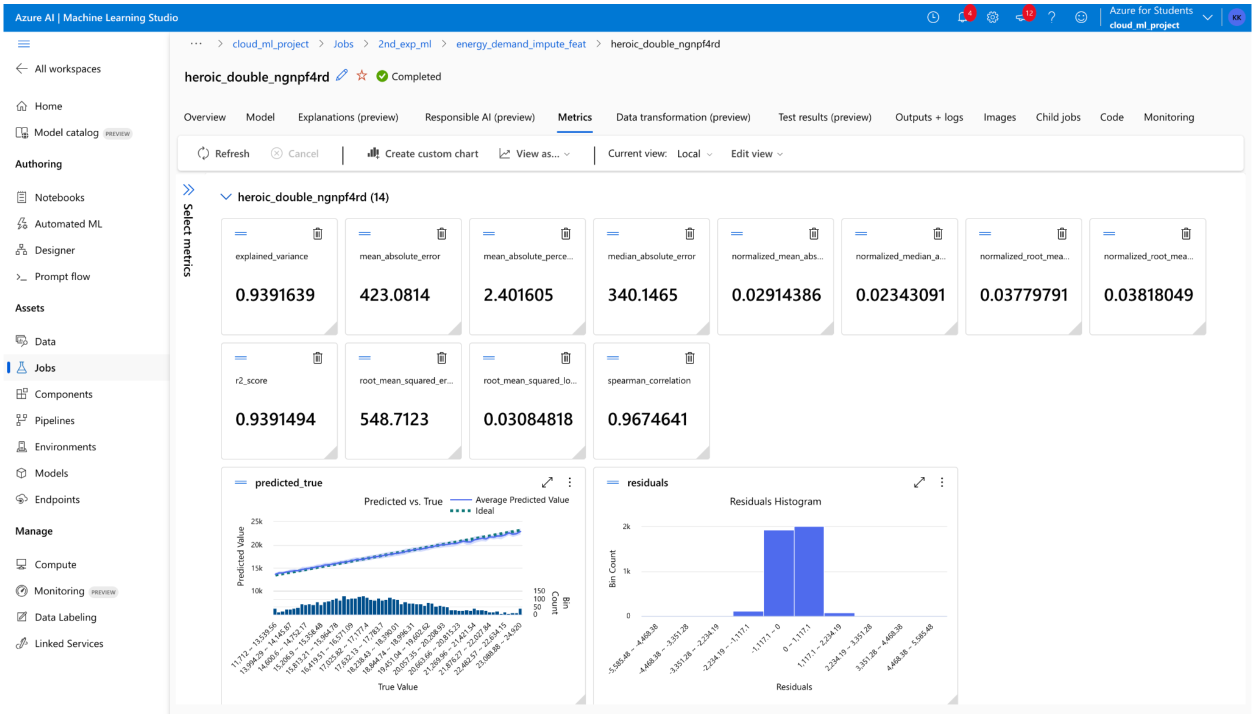
R2 score
0.93915

Root mean squared error
548.71

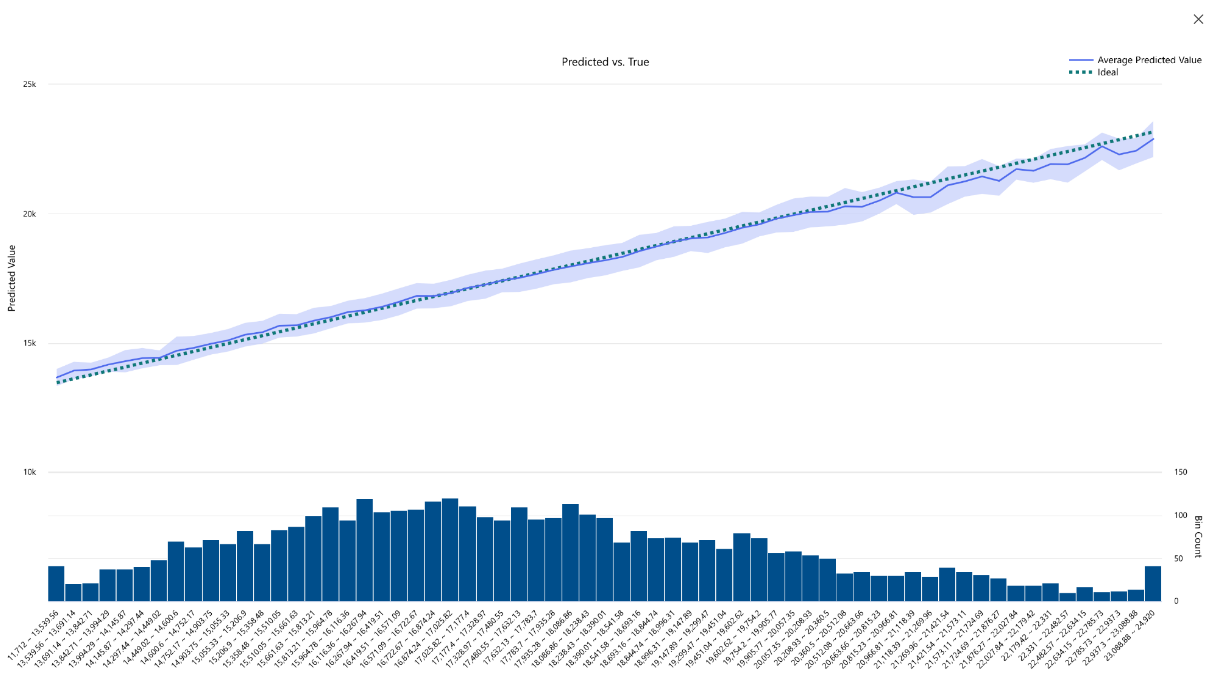
Root mean squared log error
0.030848

Spearman correlation
0.96746

Close

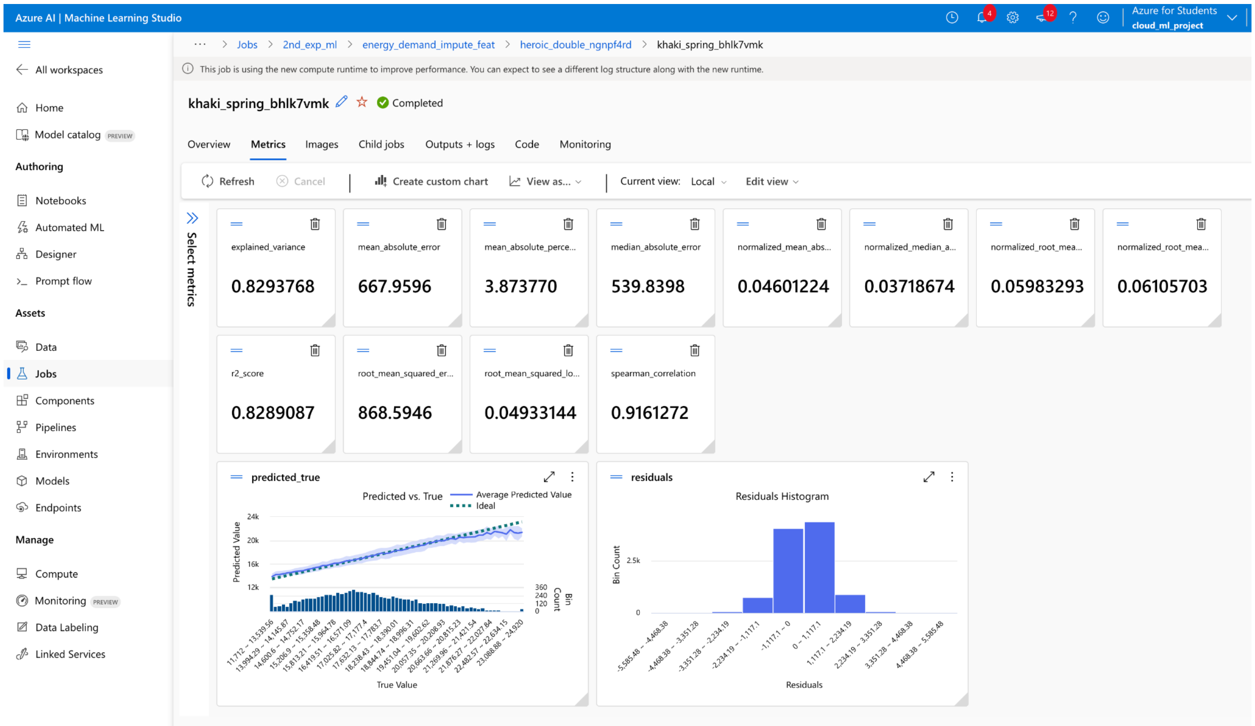


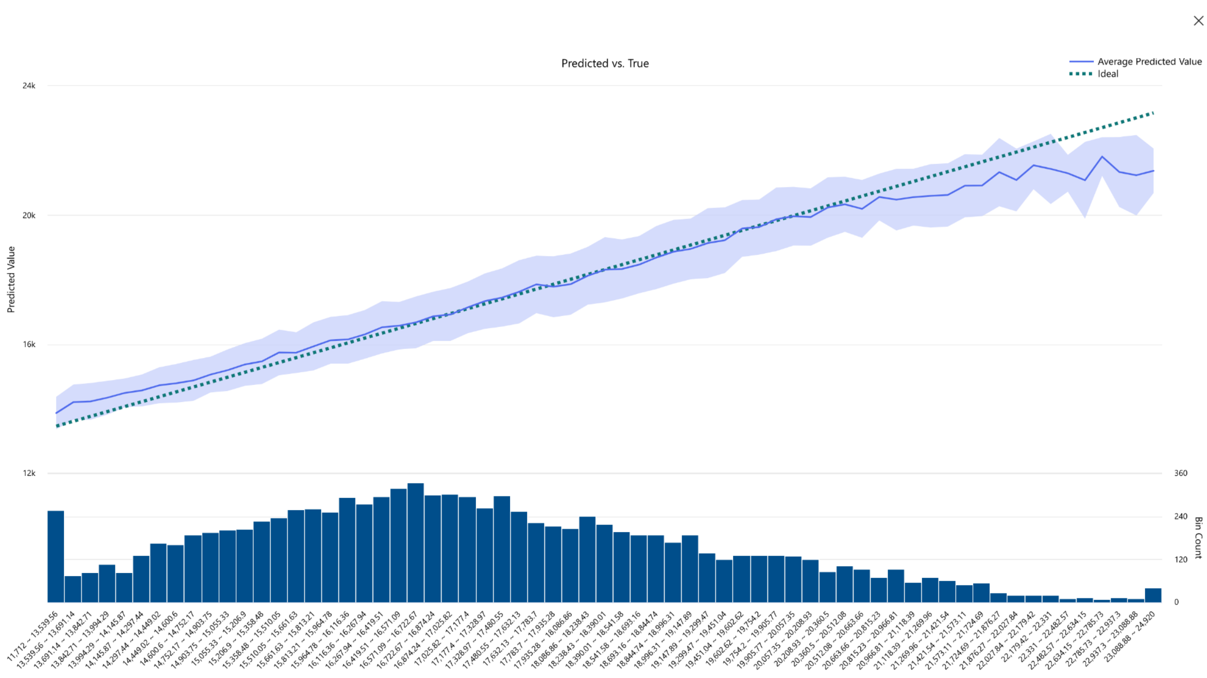
Azure AI | Machine Learning Studio
heroic_double_ngnpf4rd
Completed
Metrics
explained_variance
0.9391639
mean_absolute_error
423.0814
mean_absolute_perce...
2.401605
median_absolute_error
340.1465
normalized_mean_abs...
0.02914386
normalized_median_a...
0.02343091
normalized_root_mea...
0.03779791
normalized_root_mea...
0.03818049
r2_score
0.9391494
root_mean_squared_er...
548.7123
root_mean_squared_lo...
0.03084818
spearman_correlation
0.9674641
predicted_true
Predicted vs. True
residuals
Residuals Histogram



Predicted vs. True
Average Predicted Value
Ideal
Predicted Value
Bin Count
True Value

Results on test set:





5 Concluding Remarks

Energy price forecasting is a difficult but important task in the energy sector, and will become increasingly important with the sustainable energy transition. Models like the one presented here provide an opportunity to both support the grid and gain a competitive advantage by predicting future market changes.

The code and data for our project is stored at this Github link. We have attached screenshots of our model running in Azure and AutoML.