Thursday 4/24/22

Presentation

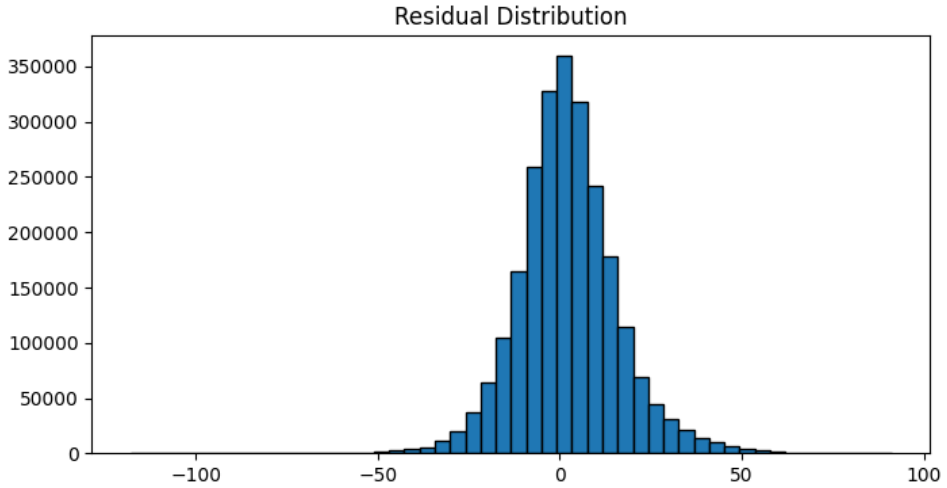
- **Final training dataset -** preprocessed and transformed ready to be split, cross validated, and fit.
- Here are the versions just before the final transformations:
    -
    - Important transformation steps included:
        - Raw data sources
          – Read in SCADA load/control dumps and year-by-year historical weather feeds
          – Cleaned timestamps, merged on local_time → one giant DataFrame
        - Missing-value handling
          – Imputed or forward-filled gaps in weather (temperature, humidity, precipitation)
          – Dropped any rows still missing critical covariates
        - Feature engineering
          – Extracted temporal features (hour-of-day, day-of-week, holiday flags)
          – Built lag/rolling-window stats on weather and load to capture short-term trends
          – Encoded categorical/location info as needed
        - Target construction
          – Created multi-horizon targets via
            - y_plus_24h = OnLine_Load_MW.shift(-24),
              y_plus_48h = …,
              y_plus_72h = …,

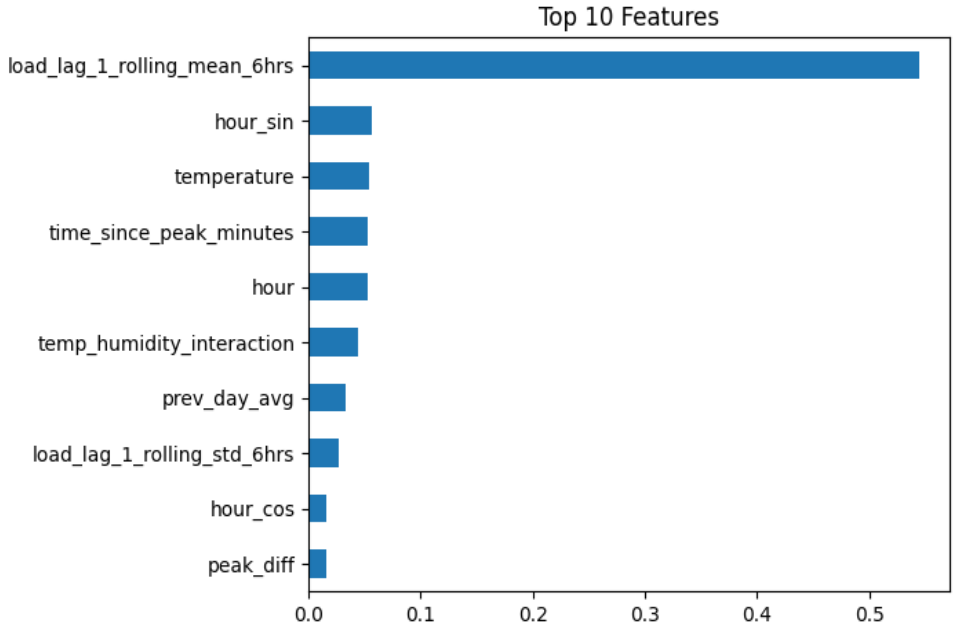
          – Filtered to only rows with all three horizons present

        - Leakage-column drop
          – Removed OnLine_Load_MW, Load_Control_MW, Control_Threshold_MW, timing fields, etc.
        - Serialization
          – Cast feature matrix to float32, packed into HDF5 under key "df", wrote out as data/training/east_river_training-v2.h5
- First modeling step was always:
    - Read, sort on local_time, splilt off X vs multi-output y
    - StandardScaler > persist
    - Build sequences via TimeseriesGenerator or raw matrix slices
    - Chrono 60/20/20 train/val/test split
- Baseline XGBoost experimentation
    - Basic loop over horizons, fit and evalutate

-

horizon_h	MAE	RMSE
24	9.544760	13.032488
48	10.007958	13.590096
72	10.359988	13.970080

- 

- 

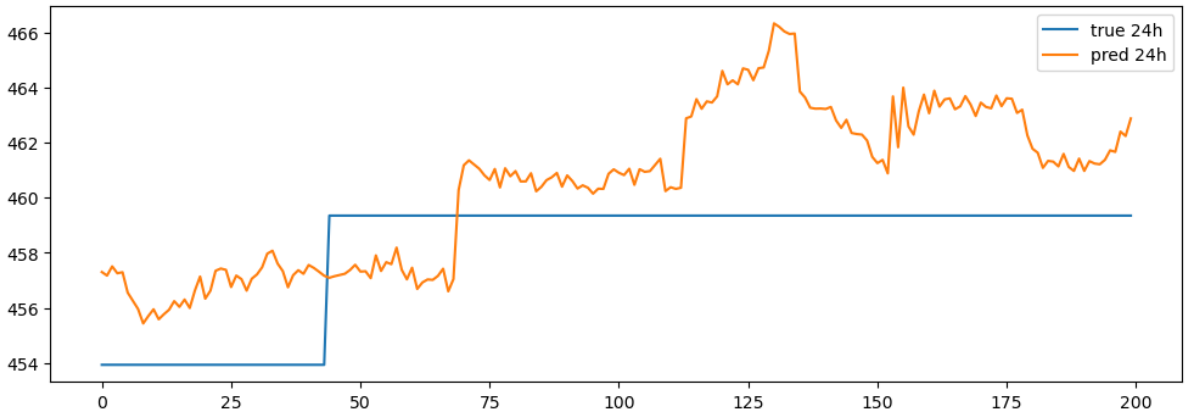
- Add in TimeSeriesSplit and cross-validation using
  - Used smaller tuning slices and eventually used the whole dataset

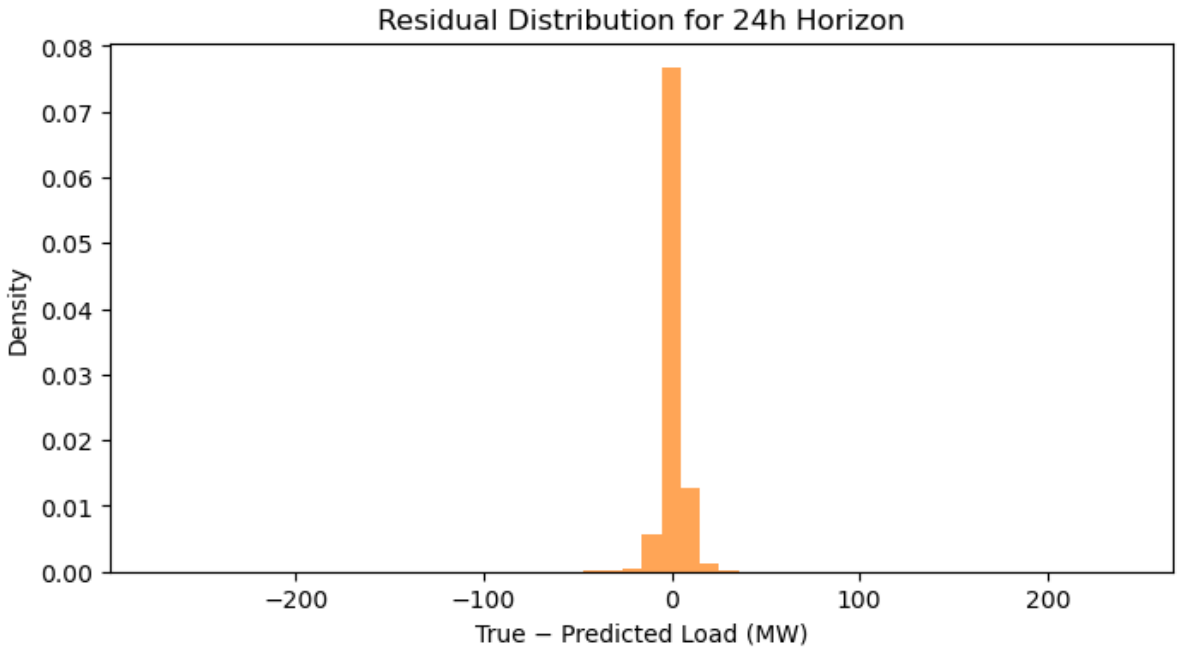
horizon_h	best_params	cv_MAE	MAE	RMSE
24	{'subsample': 1.0, 'n_estimators': 200, 'max_d...	26.194804	12.062893	15.863531
48	{'subsample': 1.0, 'n_estimators': 200, 'max_d...	26.248424	12.418220	16.270211
72	{'subsample': 1.0, 'n_estimators': 200, 'max_d...	27.548616	12.587905	16.470813

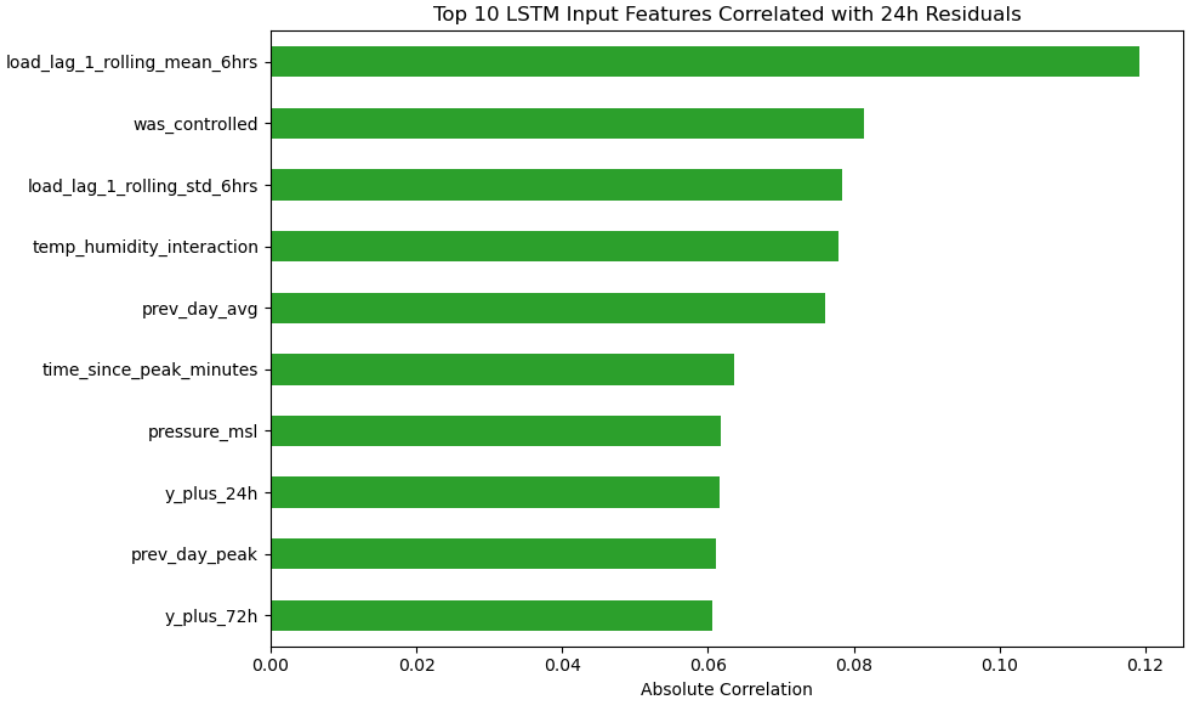
- Improved on the pipeline a bit by :;
  - Train/val/test_split added
  - XGBRegressor per horizon
  - Increase CV splits

```
⇒ 24h HO MAE: 10.72, RMSE: 14.77
⇒ 48h HO MAE: 11.25, RMSE: 15.27
⇒ 72h HO MAE: 11.72, RMSE: 16.05
```

- LSTM trial : small-capacity, regularized LSTM that's auto-stopped and lr-adjusted based on validation performance. (tons of help from GitHub co-pilot)
  - Sequence generation & splits
    – Uses TimeseriesGenerator to build sliding windows of length SEQ_LEN
    – Chronological 60/20/20 split into train/val/test generators
  - Model topology
    – Single LSTM layer (32 units) with L2 weight decay on both input & recurrent kernels
    – Dropout(0.5) after the LSTM
    – Dense(16, relu) with L2 regularization
    – Final Dense layer with one output per horizon (24/48/72 h)
  - Optimization & regularization
    – Adam optimizer (lr = 5e-4)
    – ReduceLROnPlateau callback to cut lr by 0.2 when val_loss stalls (patience=3)
    – EarlyStopping (patience=10, restore_best_weights) to avoid over-fitting
    – ModelCheckpoint to persist the best weights to multi_lstm.h5
  - Training & evaluation
    – Trains on train_gen, monitors val_gen loss each epoch
    – After fitting, loads best weights and predicts on test_gen
    – Computes per-horizon MAE/RMSE on the hold-out window
    – Plots a sample 24 h forecast vs. true values
  - Set epochs at 50
  - Needs and improvement opportunities:
    - PCA and better feature engineering/selection
    - Real time weather data
    - Better historical weather data points
    - Better local event data
    - `drop y_plus columns from inputs and re-scale`
    - Drop
  - Results (probably highly overfit due to feature selection)
    -
```
24h HO MAE: 3.62, RMSE: 6.69
48h HO MAE: 4.11, RMSE: 7.33
72h HO MAE: 4.68, RMSE: 8.01
```

- 

- 

- 

- Key takeaways:

        - Most errors like within 5MW, with very few extreme outliers. Under 1% average error.
        - A few of these features seem curious.
        - Rapid loss drops in epochs indicated strong model grasp
- Project organization and deliverables
    - Repo
        -
```
East_River/
├─ data/
│  └─ sample/
├─ schema/
│  └─ generated_schema.json
├─ notebooks/
│  ├─ EDA/
│  │  └─ feature_engineering/
│  └─ modeling/
│     ├─ baseline_xgboost/
│     ├─ experimentation/
│     ├─ lstm/
│     └─ wrangling/
├─ scripts/
│  ├─ train_lstm.py
│  ├─ predict_lstm.py
│  ├─ predict.py
│  └─ sample_data.py
├─ models/
│  ├─ multi_lstm/
│  └─ xgb_multi_horizon_models/
├─ environment.yml
└─ README.md
```
        - Dataset schema with dtypes and descriptions included.
        - Training scripts provide the model training pipeline
        - Predict scripts provide a way to ingest new data, match it to schema and feed it to the model pkl. However, my highly messy ETL process would make any data engineer cringe. Dataset reproducibility was an afterthought here, unfortunately.
- **Next steps / Suggestions**
    - **Improve data source and ETL pipelines**
    - **Conduct a more thorough EDA**
    - **Conduct more educated feature engineering**
    - **Conduct feature selection iterations**
    - **Integrate a real-time weather forecast**

Tuesday 4/1/25

Progress

- Static dataset was augmented the dataset with additional weather and calendar features.
- Engineered temporal, seasonality, and peak indicator features.
- Validated weather data and zero value data to the best of our domain knowledge

- Quickly processed data for modeling
    - Remove irrelevant features
    - Imputed null values with time interpolation on weather data when creating 30min intervals.
    - Imputed null load values from EOY 2024 with KNN
    - Imputed null lag and previous features with fillna median.
    - Validate target column
    - Scale numerical
    - Encode categorical
    - Remove features with multicollinearity
    - Time-ordered train-test split (70/30) - stopped here

Questions

- **Target variable –** would forecasting **Online_Load_MW**, East River General more accurately determine *when* and *how* to apply load control to remain under critical thresholds? This approach seems to support a more adaptive strategy, aligning with the overarching goal of reducing peak demand charges and optimizing grid stability.
- GPUs available? – stuck at preprocessing and haven't ran baseline model.
  - Jess confirmed we can try the Online_Load_MW target variable approach.

Monday 2/24/25

Progress

- Agreed on approach and basic model architectures
    - XGBoost with data enhancements – baseline
    - LSTM
    - TFT – might be a bit too ambitious for us at this point
- Drafting a thorough README to start the documentation process
- Completed cleaning and timestamp transformation on SCADA dataframe to match historical weather data timestamps
- Working on historical weather data cleaning and transformations:
    - Handling missing time stamp observations for consecutive data for all locations.
        - Forward fill vs KNN as methods to
        - Prepare to match with SCADA data into one df