MULTIVARIATE TIME-SERIES PREDICTION USING DEEP LEARNING

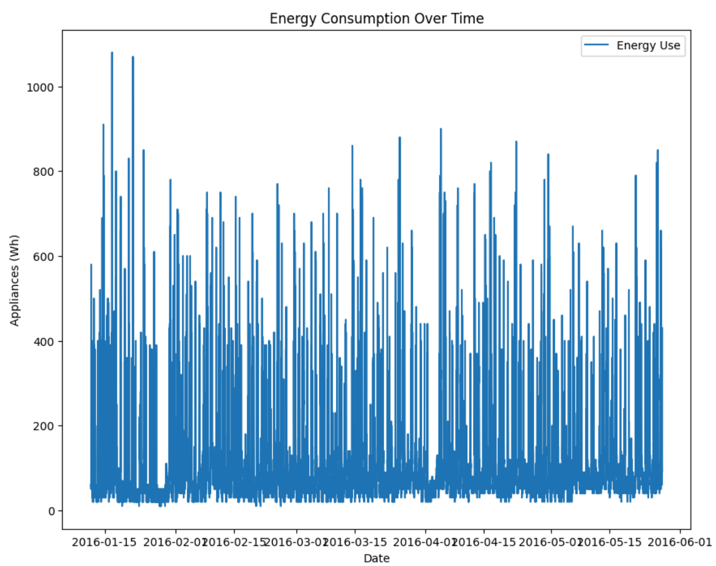
AI /ML Internship Assignment

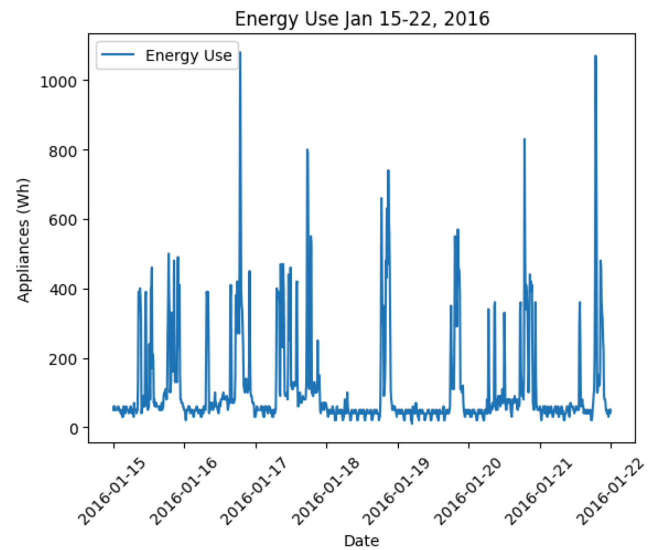
Chamudi Siriwardhane

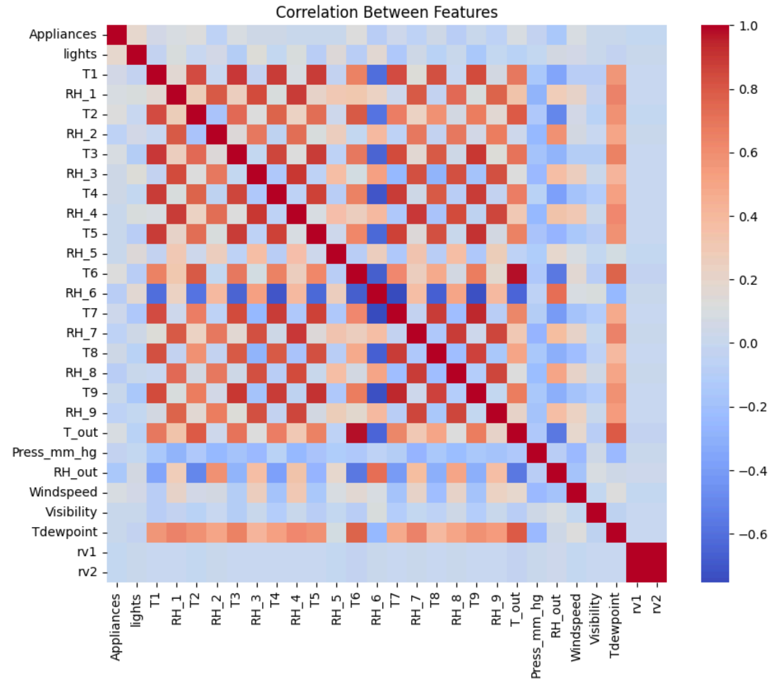
2025/07/02

Introduction

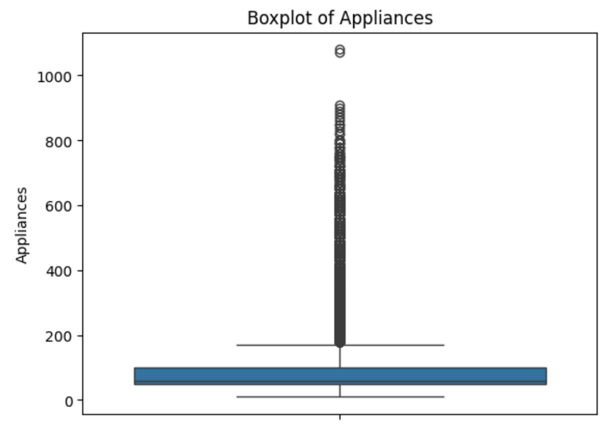
- A line plot showed energy use (Appliances) goes up in the evening, probably when people are home.
- Zoomed up line chart shows the trends in each day more clearly.
- Energy use has been decreased over the months indicating changes in energy use in different seasons. (Eg,, More energy consumption in Winter)
- A heatmap showed Appliances weak correlation related to indoor temperature (T1, ~0.15. correlation) and humidity (RH_1, ~0.093 correlation).
    - Note: Date feature is excluded as seaborn.heatmap only calculates correlations for numeric columns.







- A box plot showed some very high energy values (outliers), likely from unusual appliance use.



Preprocessing

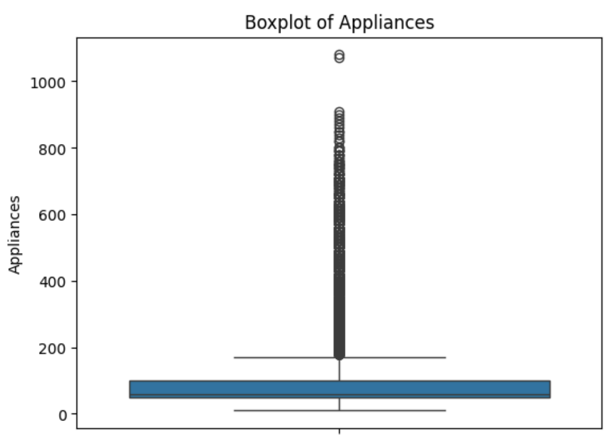
I cleaned and prepared the data so the model so the modal can perform better.

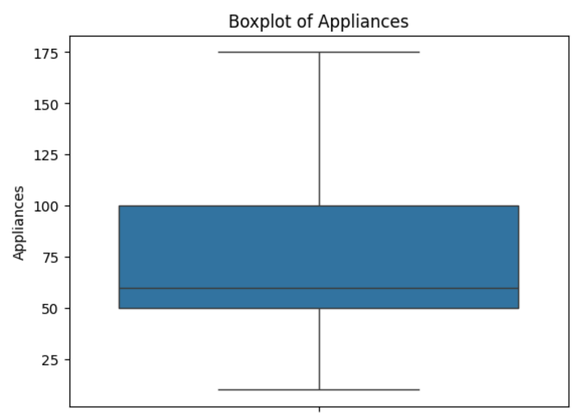
Missing Values:

- No missing values were found.

Outliers:

- Found outliers in Appliances using a box plot and capped them to avoid extreme values messing up the model. Because Capping keeps the data but reduces the impact of weird spikes.





Scaling: Used StandardScaler to make all numbers excluding the target variable (Appliances) similar in size. So the model treat all features fairly.

Feature Engineering

Since correlations with Appliances were weak (e.g., 0.145 with T1), I added time features like Hour to capture daily patterns. I used all available features in the LSTM model because it can find hidden patterns by combining them, even if individual correlations are low.

Time Features:

Added Hour, DayOfWeek, and Month from the Date column. Made IsWeekend (1 for weekends, 0 for weekdays) from WeekStatus. Why? Energy use changes by time of day and weekday/weekend.

Rolling Average:

Added a 1-hour average for Appliances to smooth out quick changes. Because this helps the model see overall trends.

Domain-Specific Features

Added holidays because energy use can change in holidays.

Lagged Features:

Added Appliances_Lag1 (energy use 10 minutes ago) and Appliances_Lag3 (30 minutes ago). Because past energy use helps predict future use.

Interaction Feature:

Made T1_RH1 by multiplying T1 (temperature) and RH_1 (humidity). Because temperature and humidity together affect energy use (e.g., air conditioning).

Feature Selection:

Picked the top 10 features with the strongest correlation to Appliances. These features are the most useful for predictions.

Top 10 features

- Appliances_Rolling
- Appliances_Lag1
- Appliances_Lag3

- Hour
- Lights
- RH_out
- T2
- T6
- T_out
- RH_8
- RH_6

No dominant predictors.

Splitting Data:
Split the data into 80% for training and 20% for testing.

Final subset used for training and testing = data[Top_features] + data[Appliances]

Model Design

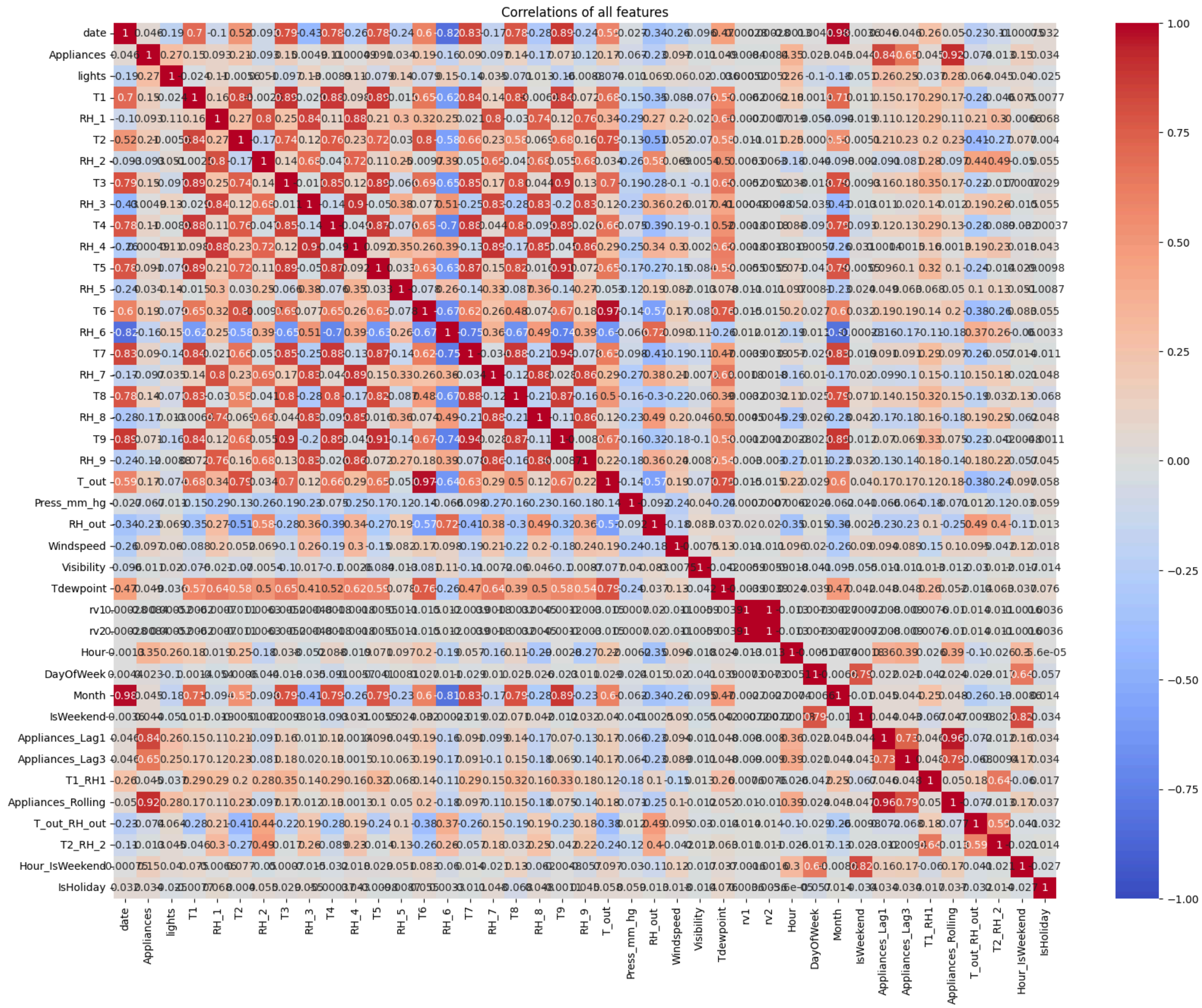
I built two models:

Baseline Model (Linear Regression):

Used a simple Linear Regression model to set a starting point.

LSTM Model:

- Built a deep learning model with Two LSTM layers in the initial model (32 and 16 units) to learn patterns over time.
- A Dropout layer to prevent overfitting.
- A Dense layer to predict Appliances.



- Used 6 time steps (1 hour) in the initial model for input.

- Used tanh activation (default for LSTM) and Adam optimizer (good for learning quickly).
- Used Mean Squared Error (MSE) as the loss function because its good for predicting numbers.
- Chose LSTM because it is great for time-series data like energy use, as it remembers past patterns.
- Ran LSTM model with different hyperparameters.

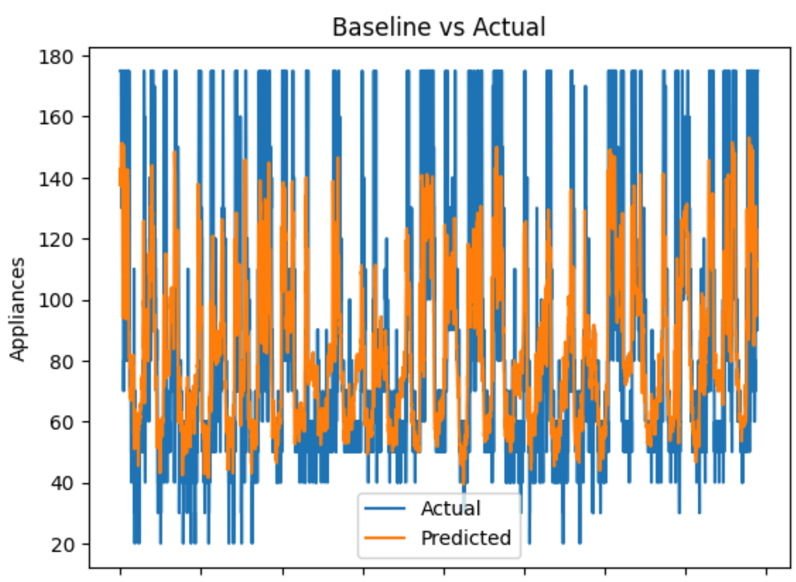
Results

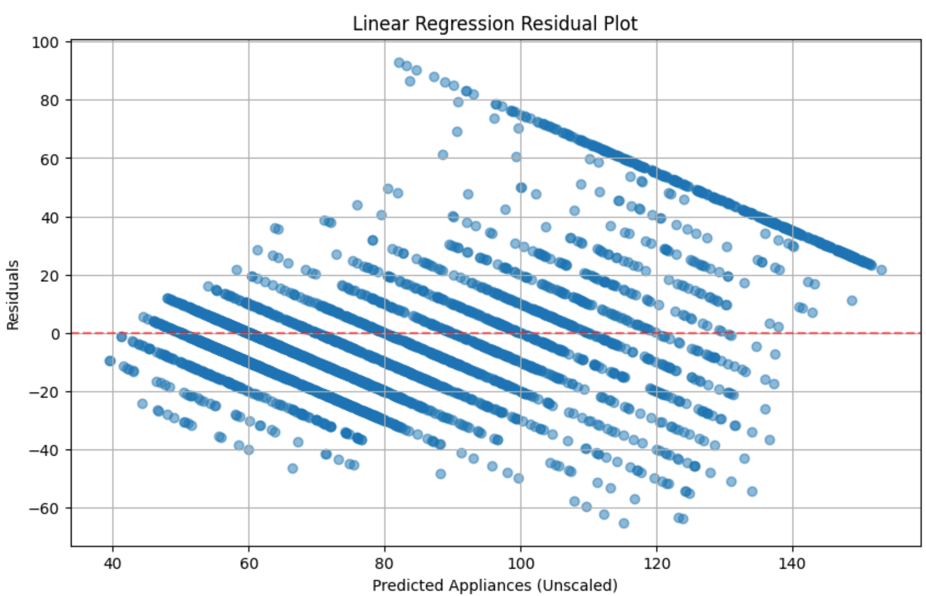
Metrics:

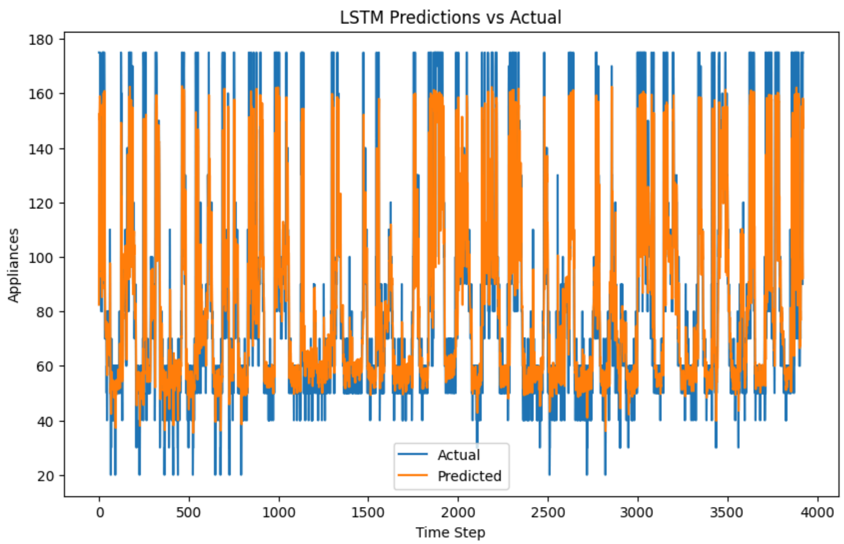
1) Linear Regression - MAE: 15.52 , RMSE: 21.51
2) LSTM - MAE: 13.28, RMSE: 21.31

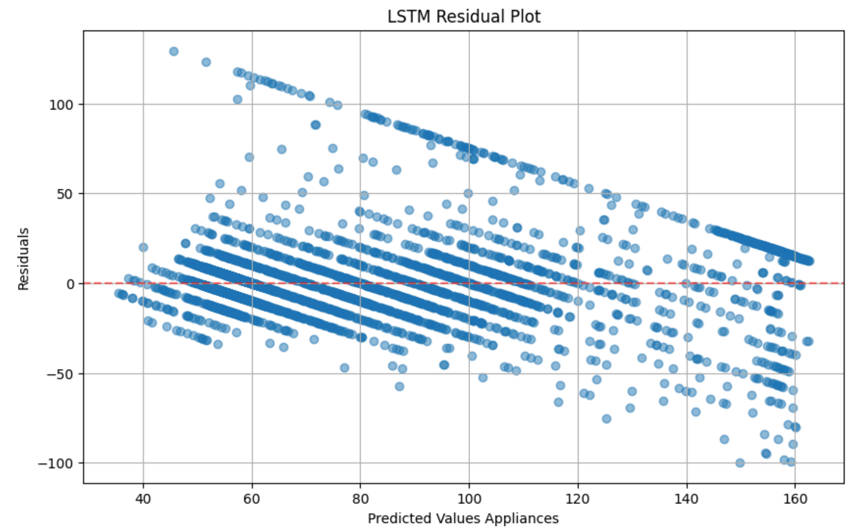
The LSTM model significantly outperformed linear regression in terms of mean absolute error (MAE), reducing it from **15.52** to **13.28**, which indicates a **~14.4% improvement in average prediction accuracy. Root mean squared error (RMSE) saw a slight improvement 0.93%**

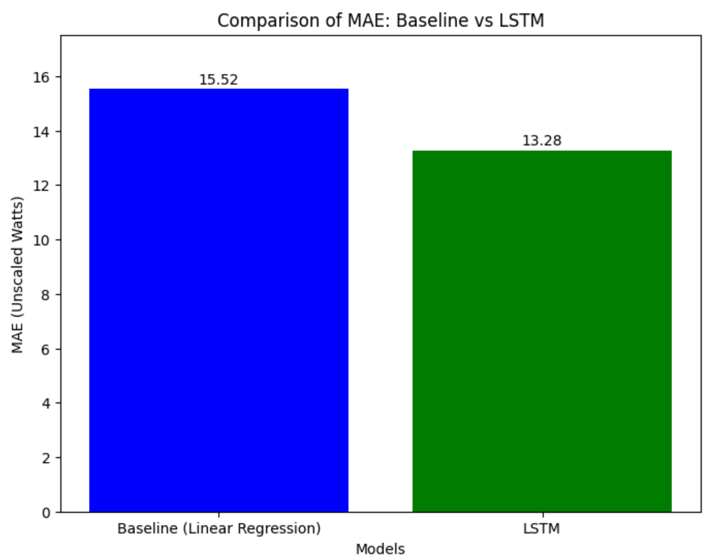
Plots:











- A plot of predicted vs. actual Appliances showed the LSTM follows the real data closely, especially for daily patterns.

- A residual plot showed most errors are small, meaning the model is fairly accurate.
- Bar plot depicts the difference of Mean Absolute Errors (MAE) between the baseline and LSTM. LSTM outperforms the baseline, reducing the average error by roughly 1.89 watts.

Observation:

The LSTM handles peaks in energy use better than the simple model.

Model Optimization

I improved the LSTM model by adding new features.

Early Stopping:

Stopped training if the model didn't improve after 5 epochs. This prevents the model from overfitting.

Changing configurations:

I experimented with different hyperparameters.

- ❖ Initial configuration:
  - 50 epochs, LSTM layers with 32 and 16 units
  - Dropout: 0.2
  - Timesteps: 6
  - Results: MAE = 13.9149, RMSE = 21.2934
- ❖ Second configuration:
  - Early stopping enabled (stopped at 20/100 epochs)
  - LSTM layers with 128 and 64 units

- Dropout: 0.05
- Timesteps: 12
- Results: MAE = 13.6939, RMSE = 21.2720

❖ Third configuration (best MAE):

- Early stopping enabled (stopped at 15/100 epochs)
- LSTM layers with 256 and 128 units
- Dropout: 0.1
- Timesteps: 24
- Results: MAE = 13.2757, RMSE = 21.3130

Dropout:

Tried different dropout rates (eg. 0.2, 0.05, 0.1).

After tweaks, MAE improved by 4.59% and RMSE by 0.09%.

Challenges and Solutions

- Challenge: I didn’t understand how to prepare data for LSTM (it needs 3D shapes).
- Solution: Watched a TensorFlow tutorial and used a function to create sequences.

- Challenge: The model was overfitting (training error was much lower than test error).
- Solution: Added dropout and early stopping to keep the model balanced.
- Challenge: Choosing the best features was hard.
- Solution: Used correlation to pick the top 10 features, which was simple and effective.

Conclusion

I built an LSTM model that predicts appliance energy use better than a simple Linear Regression model. The model learned from time-based and past energy features, reducing errors significantly. In the future, I could add holiday data or try a different model (like a mix of CNN and LSTM) to improve predictions further. This project taught me how to clean data, add features, and use deep learning for time-series.

References

TensorFlow. (n.d.). Time Series Tutorial. https://www.tensorflow.org/tutorials

Kaggle. (n.d.). Time Series Basics. https://www.kaggle.com/learn/time-series