In [11]:
```
# Libraries
#nbconvert:hide_input
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dense, Dropout
from keras.optimizers import Adam

import pandas as pd
from matplotlib import pyplot as plt
from sklearn.preprocessing import StandardScaler
import seaborn as sns

import plotly.express as px
from sklearn.ensemble import IsolationForest

import plotly.graph_objects as go
from plotly.subplots import make_subplots

from statsmodels.nonparametric.smoothers_lowess import lowess
from statsmodels.tsa.seasonal import DecomposeResult, seasonal_decompose

from sklearn.metrics import mean_squared_error

from datetime import datetime, timedelta
import requests
import math

import warnings
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
warnings.filterwarnings("ignore")
plt.style.use("ggplot")
plt.rcParams.update({"font.size": 14, "axes.labelweight": "bold", "lines.linewidth": 2})
```

In [12]:
```
#nbconvert:hide_input
df = pd.read_csv("../../data/processed/ail_price.csv", parse_dates=["date"], index_col="
df = df.sort_values(by="date")

features = pd.read_csv("../../data/processed/complete_data/features.csv", parse_dates=["

features = features.sort_values(by="date")
price = pd.DataFrame(df['price'])
feature_df = pd.merge(price, features, left_index=True, right_index=True)
```

Proposal Report : Power Price Prediction

Contributors

- Arjun Radhakrishnan
- Sneha Sunil
- Gaoxiang Wang
- Mehdi Naji

Executive Summary

Our proposed business solution aims to build an interpretable and explainable data science product ready to be deployed on the cloud for power price prediction in Alberta Energy Market. Our solution can enable

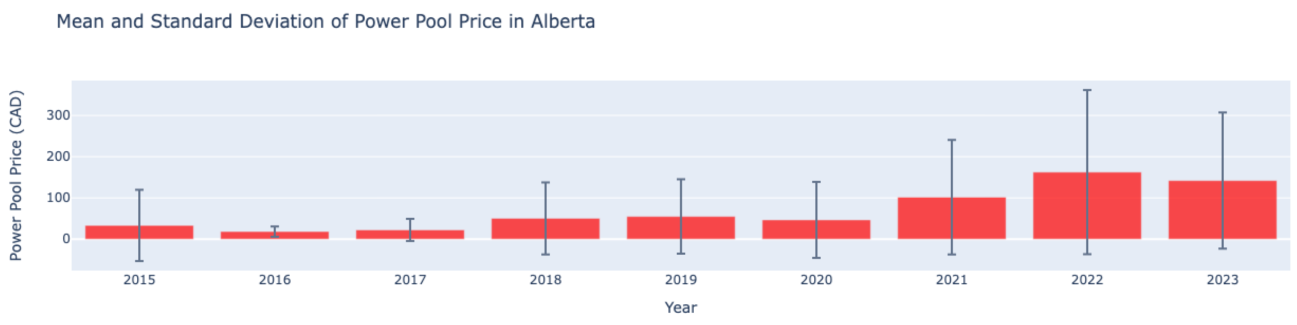
organizations to make informed decisions about their energy purchases by forecasting hourly energy prices in advance, along with confidence intervals. The solution will also address the lack of interpretability and explainability in the current system[1]. This product will be accompanied by an intuitive tableau dashboard showcasing relevant visualizations to enable stakeholders to monitor real-time hourly predictions with a margin of error.

Introduction

Over the past years, the electricity price evolution in Alberta has exhibited a mix of trends and volatility. One notable trend has been the overall increase in electricity prices, driven by a combination of factors. Alberta's transition towards cleaner energy sources, such as wind and solar, has led to additional costs associated with infrastructure development and integration.

As the below plot shows, at least since 2016, the average and standard deviation of electricity prices in Alberta have been evidently rising. This upward trend and increasing volatility can be attributed to multiple factors. The growing demand for electricity, driven by population growth and expanding industries, has strained the power infrastructure, resulting in tighter supply-demand dynamics and higher prices. Additionally, government policies favoring cleaner energy sources have necessitated significant investments in infrastructure and new power generation facilities, passing on the costs to consumers. The integration of intermittent renewable energy sources has introduced price volatility, while the maintenance and upgrades of aging transmission and distribution infrastructure have further contributed to the rising prices. Furthermore, fluctuations in fuel costs, such as natural gas, impact the overall cost of electricity generation in Alberta. Collectively, these factors have contributed to the evident increase in power prices in the province.

In [13]:
```
price = pd.DataFrame(df['price'])
ppx = pd.DataFrame()
ppx['std'] = price.groupby(price.index.year).std()
ppx['mean'] = price.groupby(price.index.year).mean()

fig = go.Figure()
fig.add_trace(go.Bar(
    x=ppx.index,
    y=ppx['mean'],
    error_y=dict(
        type='data',
        array=ppx['std'],
        visible=True
    ),
    marker=dict(color='red'),
    opacity=0.7
))

fig.update_layout(
    title='Mean and Standard Deviation of Power Pool Price in Alberta',
    xaxis=dict(title='Year'),
    yaxis=dict(title='Power Pool Price (CAD)'),
    showlegend=False
)

fig.show()
```



Another essential factor contributing to the rising power prices volatility in Alberta is the deregulated nature of the electricity market. In Alberta, the electricity market operates under a deregulated framework, where prices are determined by supply and demand dynamics in a competitive market. The government does not directly intervene in setting the prices. Instead, prices are influenced by various market participants, including power generators, transmission companies, and retailers. This market structure allows for more flexibility but also exposes prices to market forces. Therefore, the deregulated nature of the electricity market in Alberta plays a crucial role in the volatility of the power price.

The prediction of electricity prices in Alberta is of great importance to a wide range of stakeholders. Consumers rely on price forecasts to manage their energy costs and make informed decisions about their electricity usage. Businesses, particularly energy-intensive industries, need accurate price predictions to effectively plan their operations and minimize the impact of price fluctuations. Investors and financial institutions depend on price forecasts to assess investment opportunities and manage financial risks. Government bodies and regulatory authorities require reliable predictions to develop effective energy policies and ensure grid stability. Market participants, including power generators, traders, and retailers, rely on price forecasts to optimize their operations and mitigate risks in the dynamic electricity market. With the increasing uncertainty caused by rising price volatility, accurate electricity price predictions play a crucial role in enabling stakeholders to navigate the market, make informed choices, and contribute to a sustainable energy landscape in Alberta.

Alt Text

Power share generated by fuel sources at 2022.

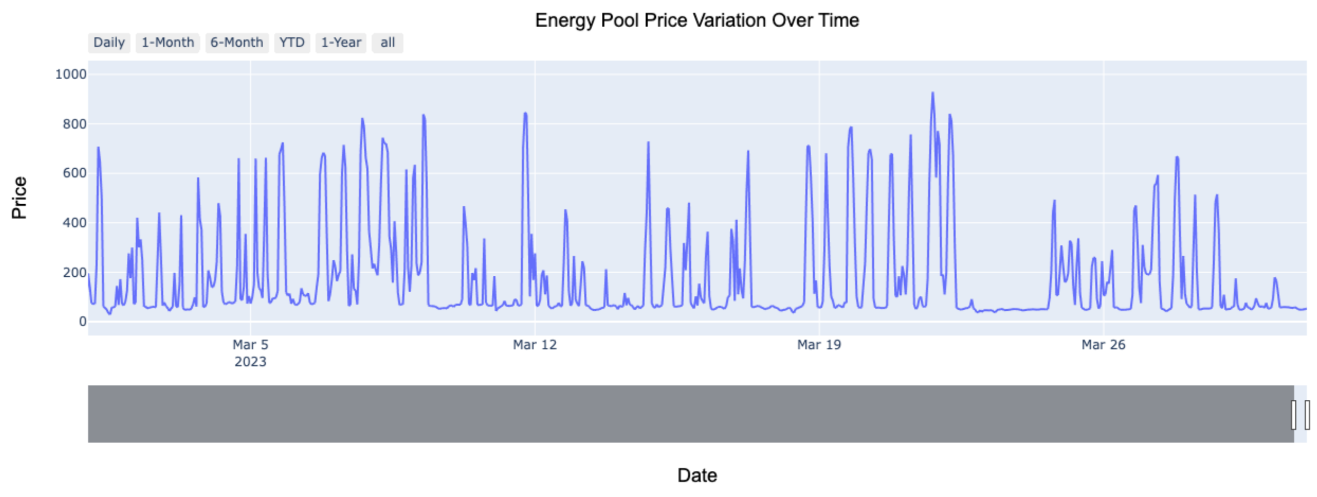
The major purpose of this project is to enhance the horizon and accuracy of the existing 6-hour-ahead price prediction offered by the Alberta Electric System Operator (AESO). AESO is the independent system operator responsible for operating the electrical grid and facilitating the competitive electricity market in Alberta. This report begins with a problem statement, defining the objective and identifying the metrics used to evaluate the prediction performance. The analysis section includes data exploration, exploratory data analysis, examination of different algorithms and techniques, and establishing a benchmark for comparison. The methodology section outlines the steps taken, including data cleaning, wrangling, and preprocessing, as well as the implementation and refinement of the predictive models. The results section focuses on model evaluation and validation, assessing the accuracy and performance of the developed models. Finally, the report addresses deployment and deliverables, discussing the practical implementation and the final outcomes of the project.

Introduction to Data

High Volatility of Power Pool Price

One of the primary characteristics that stood out in this project was the remarkable volatility of the pool price. It was immediately evident that the price exhibited significant fluctuations, capturing the attention of all involved. The interactive plot below vividly depicts the considerable variation in price specifically for the month of March 2023. As you explore the plot, you will witness the dramatic ups and downs, revealing the dynamic nature of the pool price during that period. It serves as a striking reminder of the inherent unpredictability and excitement that accompanies this particular aspect of the project.

In [14]:
```
fig = px.line(df, y="price")
fig.update_layout(
    title={
        "text": "Energy Pool Price Variation Over Time",
        "font": {"size": 18, "family": "Arial", "color": "black"},
        "y": 0.97,
        "x": 0.5,
        "xanchor": "center",
        "yanchor": "top",
    },
    xaxis_title={
        "text": "Date",
        "font": {"size": 18, "family": "Arial", "color": "black"},
    },
    yaxis_title={
        "text": "Price",
        "font": {"size": 18, "family": "Arial", "color": "black"},
    },
    xaxis_range=["2023-03-01", "2023-03-31"],
    height=500,
)
fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list(
            [
                dict(count=1, label="Daily", step="day", stepmode="backward"),
                dict(count=1, label="1-Month", step="month", stepmode="backward"),
                dict(count=6, label="6-Month", step="month", stepmode="backward"),
                dict(count=1, label="YTD", step="year", stepmode="todate"),
                dict(count=1, label="1-Year", step="year", stepmode="backward"),
                dict(step="all")
            ]
        )
    ),
)
fig.show()
```

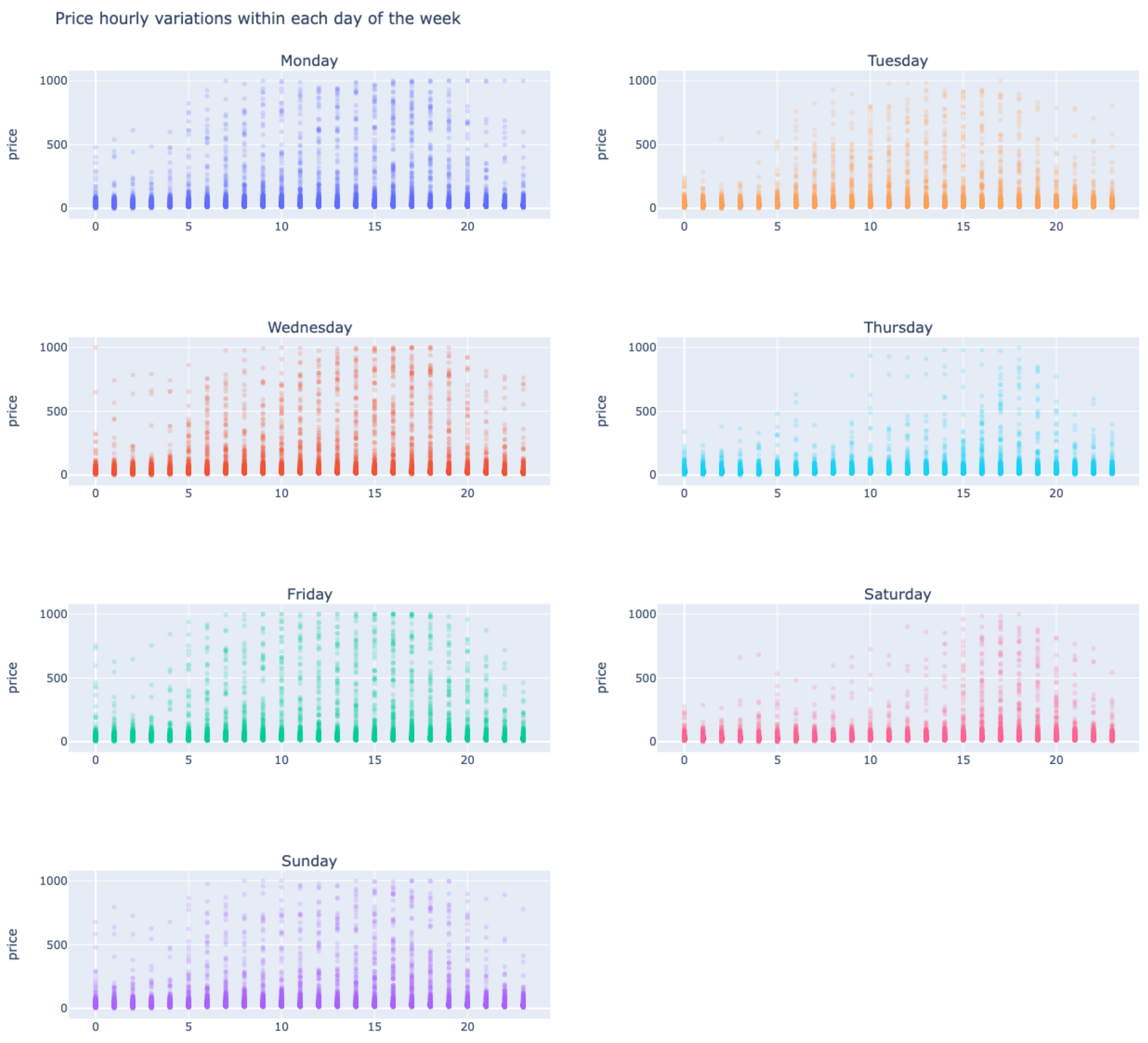


Daily Seasonality of Price Variation

Below plot demonstrates that the variation in prices is not evenly distributed across the 24-hour cycle or throughout the days of the week. Upon careful observation of the graphs, it becomes apparent that the distribution of prices significantly differs between weekdays and weekends. During weekdays, the price experiences higher values primarily during working hours, which typically span from around 7 am to 6 pm. In some cases, this trend even extends to 7 pm. Conversely, during off-working hours, the price tends to be lower.

However, the price distribution undergoes a noticeable change during weekends, with higher values occurring more frequently during the evenings. This deviation from the weekday pattern suggests a shift in market dynamics or consumer behavior during the weekend, resulting in increased prices during the evening hours.

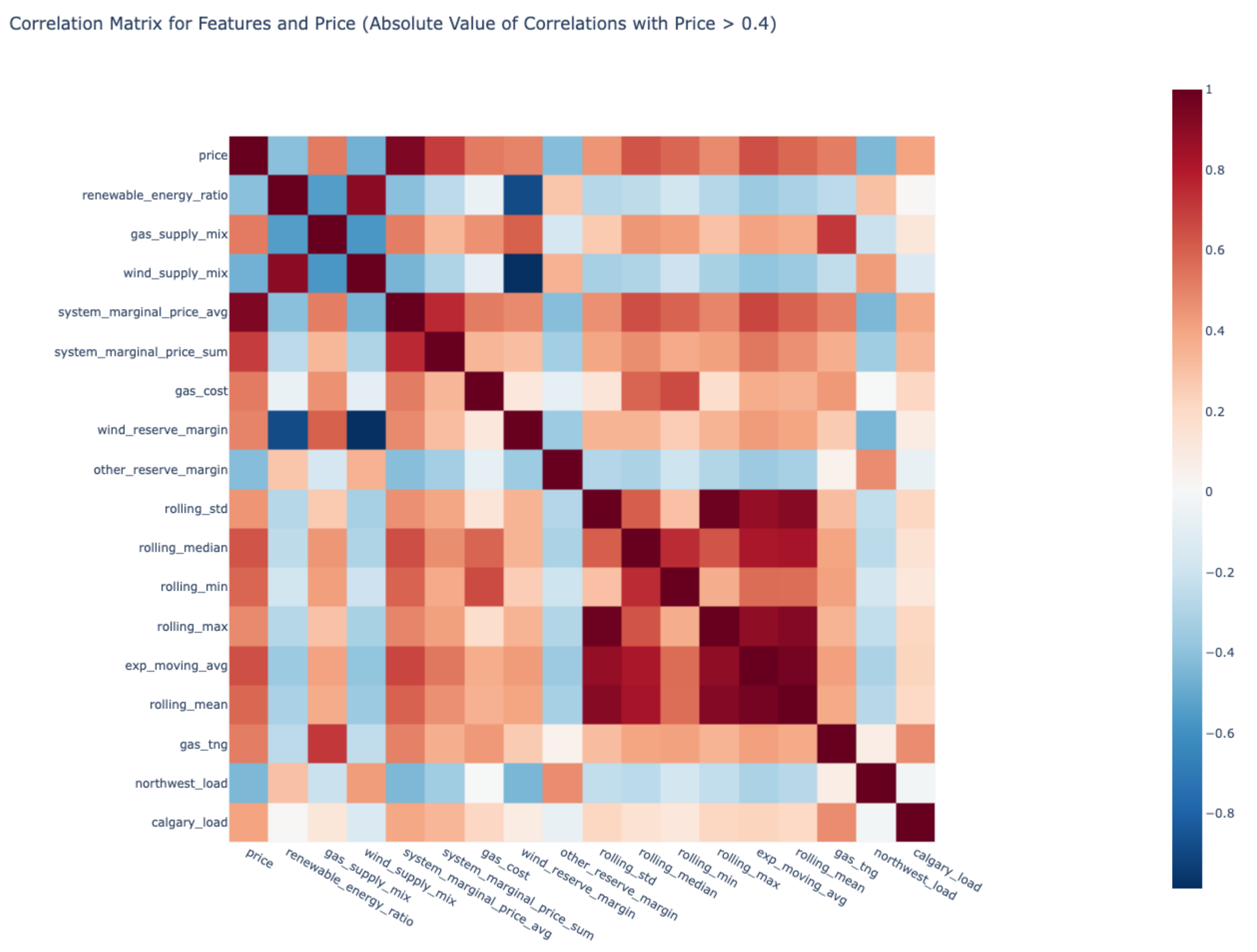
In [15]:
```
days_of_week = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sun

df['day'] = df.index.day_name()
df['hour'] = df.index.hour

Monday = df[df['day'] == 'Monday']
Tuesday = df[df['day'] == 'Tuesday']
Wednesday = df[df['day'] == 'Wednesday']
Thursday = df[df['day'] == 'Thursday']
Friday = df[df['day'] == 'Friday']
Saturday = df[df['day'] == 'Saturday']
Sunday = df[df['day'] == 'Sunday']

data = [Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday]

# Create subplots with a grid layout of 2 rows and 4 columns
fig = make_subplots(rows=4, cols=2, subplot_titles=days_of_week)

fig.update_layout(height=1200)

# Iterate over each day and add a scatter plot to the corresponding subplot
for i, day in enumerate(days_of_week):
    subplot_row = (i % 4) + 1
    subplot_col = (i // 4) + 1

    fig.add_trace(go.Scatter(x=data[i]['hour'], y=data[i]['price'], mode='markers',
                            marker=dict(size=5, opacity=0.2)),
                row=subplot_row, col=subplot_col)

    # Add x-axis label and y-axis label for each subplot
    #fig.update_xaxes(title_text='Hour', row=subplot_row, col=subplot_col)
    fig.update_yaxes(title_text='price', row=subplot_row, col=subplot_col)

# Drop the legend
fig.update_layout(showlegend=False)

# Add a title for the whole plot
fig.update_layout(title_text="Price hourly variations within each day of the week")

# Show the figure
fig.show()
```



Features cross correlation

In below matrix, the correlation matrix of features included in this project and the price is illustrated. This plot provides a deeper understanding of how each feature relates to our target variable, that is price. Using this plot, we can identify patterns and dependencies between the variables. The color scale ranges from red to blue, with red indicating positive correlation and blue indicating negative correlation. This plot also has sorted the features based on correlation with price and only kept the features that thier correlation absolute value with price is higher than 0.4.

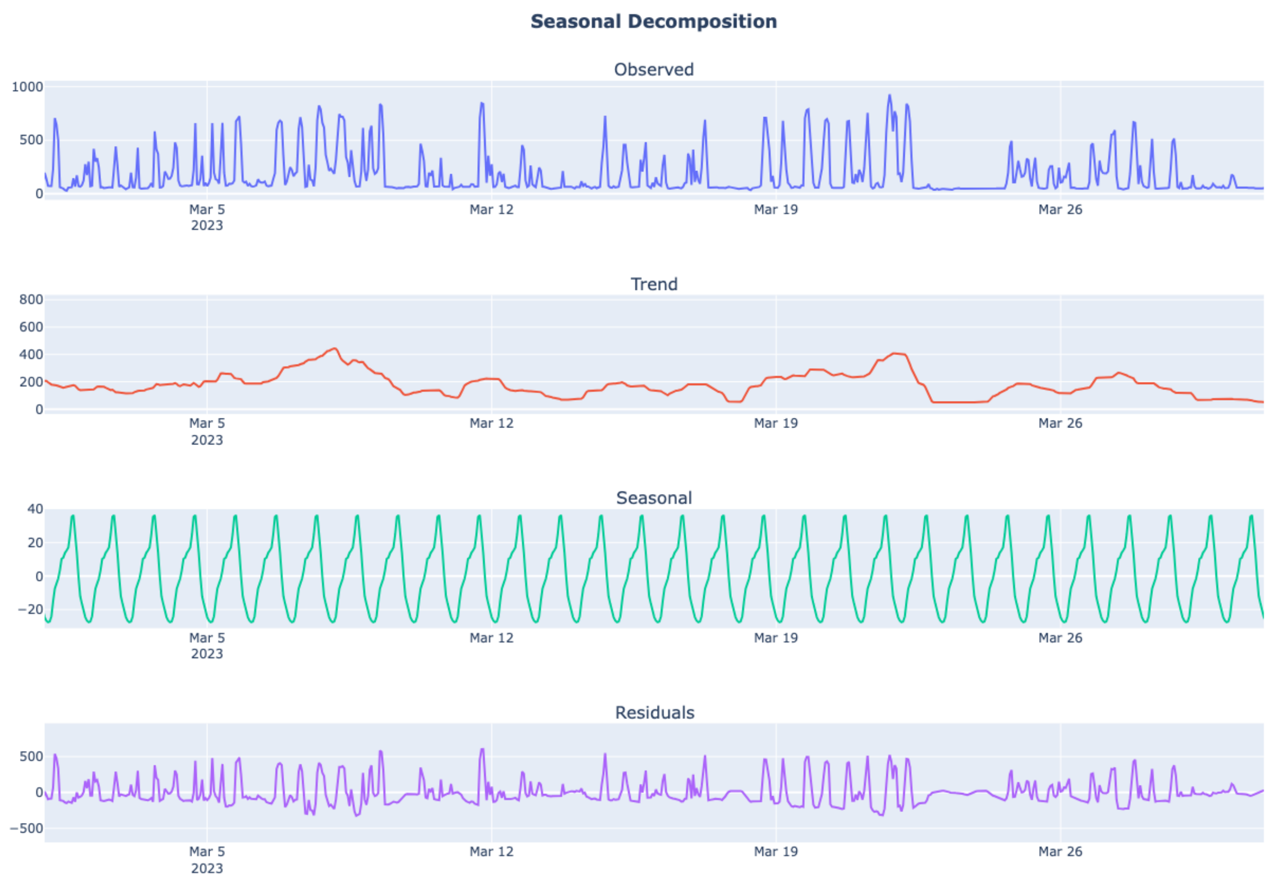
In [18]:
```
corr = feature_df.corr(method="spearman")
corr_with_price = corr['price']
selected_features = corr_with_price[abs(corr_with_price) > 0.3].index.tolist()

sorted_corr = corr[selected_features].loc[selected_features]

fig = px.imshow(sorted_corr,
            color_continuous_scale='RdBu_r',
            title="Correlation Matrix for Features and Price (Absolute Value of Correlat
fig.update_layout(
    width=1000,
    height=1000,
)
fig.show()
```



Seasonality Decomposition of the Price

In [17]:
```
def plot_seasonal_decompose(result:DecomposeResult, dates:pd.Series=None, title:str="Sea
    x_values = dates if dates is not None else np.arange(len(result.observed))
    fig = (make_subplots(rows=4, cols=1, subplot_titles=["Observed", "Trend", "Seasonal"
        .add_trace(go.Scatter(x=x_values, y=result.observed, mode="lines", name='Observe
        .add_trace(go.Scatter(x=x_values, y=result.trend, mode="lines", name='Trend'), r
        .add_trace(go.Scatter(x=x_values, y=result.seasonal, mode="lines", name='Seasona
        .add_trace(go.Scatter(x=x_values, y=result.resid, mode="lines", name='Residual')
        .update_layout(height=900, title=f'<b>{title}</b>', margin={'t':100}, title_x=0.
        .update_xaxes(range=["2023-03-01", "2023-03-31"], row=1, col=1)
        .update_xaxes(range=["2023-03-01", "2023-03-31"], row=2, col=1)
        .update_xaxes(range=["2023-03-01", "2023-03-31"], row=3, col=1)
        .update_xaxes(range=["2023-03-01", "2023-03-31"], row=4, col=1)
        )

    return fig

price = pd.DataFrame(df['price'])
price['hour'] = price.index.hour
price['day'] = price.index.strftime('%A')
price['week'] = price.index.week
price['month'] = price.index.strftime('%B')
day_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday
price['day'] = pd.Categorical(price['day'], categories=day_order, ordered=True)

month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August',
price['month'] = pd.Categorical(price['month'], categories=month_order, ordered=True)
```

```
decomposition = seasonal_decompose(df['price'], model='additive', period=24)
fig = plot_seasonal_decompose(decomposition, dates=df.index)
fig.show()
```



Data Science Techniques

To address the challenges faced by organizations in Alberta, our cutting-edge product offers a comprehensive solution that empowers them to effectively analyze costs, optimize energy purchases, and ultimately maximize their profits. The scientific objectives of our product are

- Forecasting energy prices for the next 12 hours
- Interpretability of predictions
- Scalable and Realtime prediction pipeline
- Reactive Tableau Dashboard for visualizing the real-time results

Our project utilized two primary data sources:

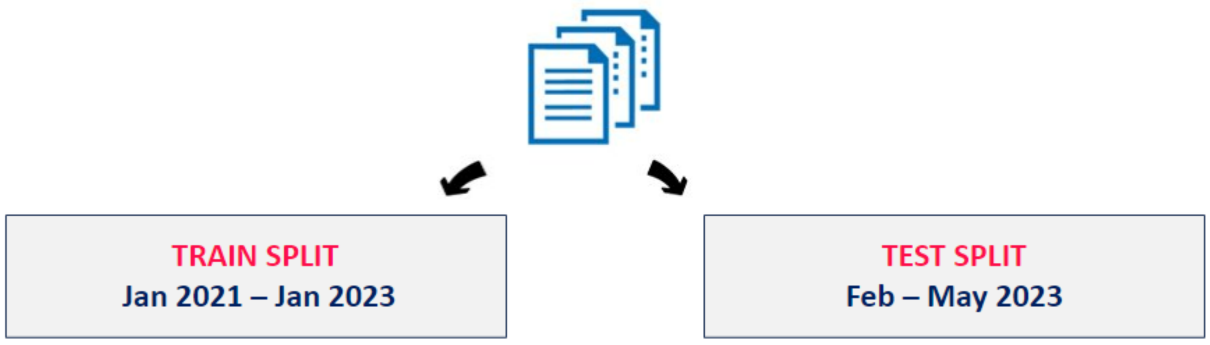
Historical Time-series Data: This dataset, publicly accessible through Tableau, spans from 2015 to 2023. With approximately 72,000 observations, it encapsulates 110 distinct features.

Open-source API Data : Through a public API service which is hosted by AESO, we can avail the real-time and historical data for certain selective features.

Our initial dataset incorporated approximately 110 features. These features consisted of information about power generation/supply by fuel type, energy load distribution across regions, import and export data, and system marginal price data. The main target that we are forecasting 12 hours in advance is the power pool price (CAD) which is the balanced average power price per hour for the Alberta province and is finalized by

AESO based on supply and demand. It is capped between 0 and 1000 to ensure that the Alberta electricity market is stable and fair. Our feature set predominantly comprises numerical data, with one exception of an ordinal feature.

For the modelling purpose, we partitioned our data into two subsets: training and testing. The training data encompasses the period from January 2021 to January 2023, while the test set refers to the data from February - May 2023. Given the absence of real-time data of all features through the API, we leveraged historical data to simulate a real-time system. The simulated system provides real time predictions starting from February 1, 2023. When real-time data becomes accessible for the clients, they can seamlessly swap the data sources, thereby enabling real-time data flow into the model which will make real time predictions.



Data preprocessing

In our pursuit to accurately predict future prices based on historical values and other influential factors like supply and demand, we transformed the time-series data into a tabular format. This involved creating lagged versions of the various features, including price, paired with the corresponding target price for each specific hour. To address the inherent challenges of our target variable - price like extreme volatility we attempted several transformation techniques:

`Log Transformation` : Given the skewed nature of the price data, a logarithmic transformation was used with an intent to normalize its distribution. This transformation could potentially make patterns in the data more interpretable and better meet the assumptions of downstream modeling techniques.

`Standard Scaler` : The standard scaler transformation was applied to standardize the price values. This technique could help minimize the effect of outliers and varying scales across different features by scaling the values to have a mean of 0 and a standard deviation of 1.

`Discretization` : The discretization transformation was used as it often complements tree-based models such as LightGBM, improving their performance by turning continuous features into several bins, each representing a range of price values.

Despite these efforts, none of these transformations led to significant improvements in our model's predictive performance. Consequently, we decided to proceed with the original scale of the price variable. This approach might also simplify the interpretation and communication of the model results, as the predictions will be in the same scale as the original data.

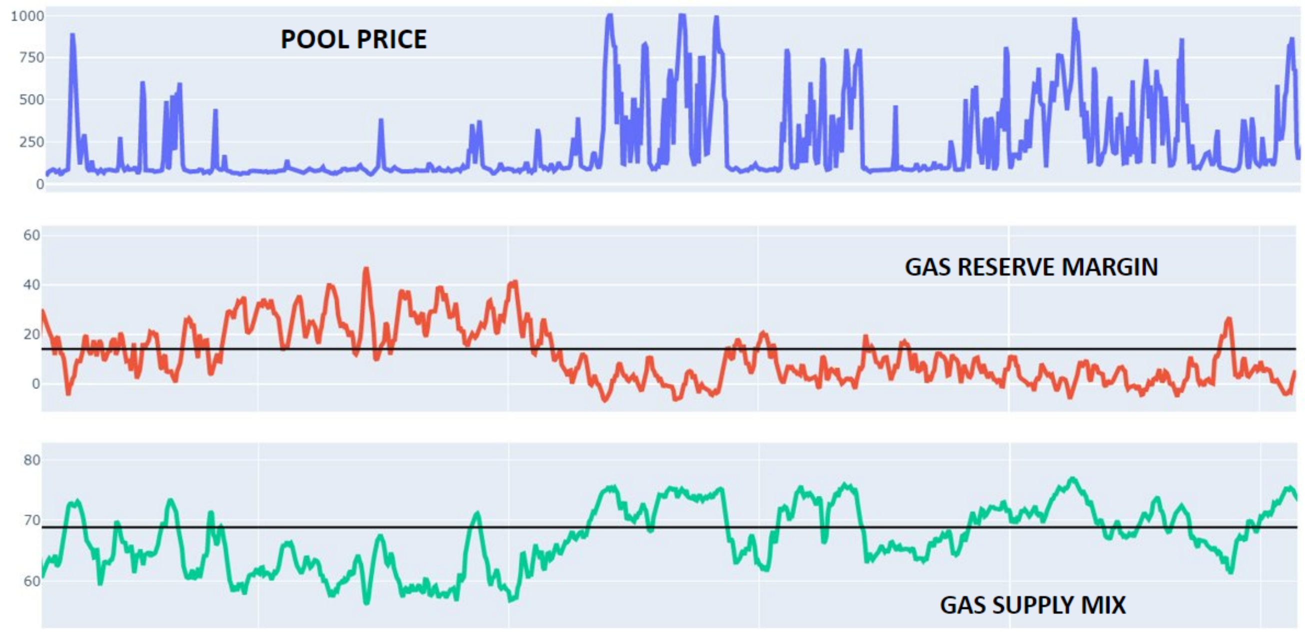
Feature Selection and Engineering

In the process of feature selection, our primary strategy involved examining the correlations between various features and the price in the preliminary round. Subsequently, we further refined our selection by leveraging the coefficients from an Elastic Net model, and the feature importances deduced from training a Random Forest Regressor model.

Pursuing a second strategy, we investigated the correlation between lagged features and future prices projected for periods ranging from 1 to 12 hours. We identified features exhibiting correlations of absolute value greater than 0.3 and incorporated them into our feature set. Interestingly, both strategies resulted in almost identical sets of features.

Upon finalizing our feature set, considering the importance of interpretability in our model, we conducted comprehensive market research and engineered several key features showing significant correlation with the price. One such feature is the load on gas reserve, a buffer/reserve of energy, readily available to meet sudden load demands and peak usage hours. As evidenced in our data visualizations, a dwindling gas reserve tends to correspond with an increase in price. In case of gas supply mix which is the proportion of energy generation using gas by the total energy generation, when the supply is mostly using gas, the price increases as gas is costly compared to the rest of the sources.

For more information about the key engineered features, please check out the glossary



It's worth noting that we found some features to be inter-correlated, suggesting that Principal Component Analysis (PCA) could potentially streamline our dataset by reducing dimensionality. However, PCA tends to complicate feature interpretability, presenting a challenge for end-users. While PCA offers benefits such as reducing computational burden and enhancing model performance, these gains must be weighed against the increased complexity and reduced interpretability. As such, we elected not to employ PCA in our model development.

Modelling Strategy

Since the price is extremely volatile, tackling our first scientific objective which is forecasting energy for the next 12 hours seemed like a complex task. Hence, we needed models that is apt for time series forecasting which is able to pick up the temporal patterns. Computational efficiency for initial training and repeated updates was also needed. As a baseline model for our problem, we chose the **SARIMA** model which is a classical model as it fulfilled the above requirements and also supported probabilistic forecasting. It was also possible to obtain confidence intervals using this model. Using SARIMA, we performed univariate forecasting, predicting the future values of the price based on the past values of the price alone. We got an average error of 83.85 CAD with a standard deviation in errors as approx. 73.11 CAD.

Since ours is a multi step forecasting problem, we then decided to switch to more sophisticated machine learning models. Here instead of just relying on the past values of the price alone, we incorporated the key engineered features that we had developed.

In terms of prediction strategy for the multistep forecasting problem, we chose the **direct strategy** over the recursive strategy. In recursive strategy, we iteratively forecast the future time step values using the predicted values. This means that the model is first trained with historical data uptil the cut off point. The cut off hour refers to the specific point in time up to which you use the data to train your model. Once the model is trained, an initial prediction is made for the next time step after the cut off point which is then again used to train the model and used as an input for the next prediction. We did not use this strategy for our modelling pipeline because the prices are extremely volatile and the recursively predicting price would propagate errors making the pipeline unstable and predictions unreliable.

Since we have a forecasting horizon of 12 steps, we went ahead with the **direct strategy** where we train 12 models to predict the power price for each timestep in the forecasting horizon which is 1,2, 3... 12. In this approach, all the 12 models are trained using the same data and each of this model will directly forecast the target for the required time step. For eg, at each hour, model 1 will always predict for power price one time step into the future and model 12 will always predict the price for 12 timesteps into the future. This will prevent the accumulation of errors in the initial predictions which can affect the subsequent future predictions as in the recursive strategy . Hence, the propagation of error is controlled because we dont rely on the previous predictions but on the real pool prices for each of the model. Each prediction is made independently based on each of the 12 models, leading to accurate results. When a data point becomes available, we update and train the model based on the real time data so that the future predictions are reliable. Hence, when we get a new data point, we update the cut off time by 1 hour and refit all the 12 models with the new data. One key factor that we considered while choosing our final model is its ability to update its parameters with minimum compute time.

One disadvantage with this approach is that we will have 12 models in our pipeline which make it bulky and computationally intensive. Moreover, once a data point is avaialable, we will have to refit the data along with the new data point to all the 12 models which can be time consuming.

We used the sktime package for building our base pipeline as it is one of the most popular packages that supports time series forecasting which enables us to use machine learning models on the time series data. Once the features were engineered, data was preprocessed and pipeline was set up, we extensively researched on the models that were a good fit for our problem. We especially checked for models that were scalable, provides accurate predictions and handles multicollinearity within the data.

Experimented models

Our first choice was Elastic Net CV which is a linear regression model that handles multicollinearity well and supports feature selection. Since interpretibility was also a main focus for us, linear model would be a good choice as it is easy to interpret. Results -

Next we tried random forest regressor model which is an ensemble model which constructs multiple decision trees at the training time and outputs the mean predictions of the individual trees. This model reduces overfitting. We also tried XGBoost which is a gradient boosting algorithm which is efficient and checked off our requirements. Finally, we integrated the light GBM model in our pipeline as it checked off all our requirements. It can handle large datasets with relatively shorter fit times, especially when loaded on a GPU. Additionally, light GBM supports warm initialization, refitting the model on new data extremely fast.

Cross Validation

Once we selected a set of our models, we then performed cross validation for all these models. Our training data was from Jan 1st 2021 - Jan 31st 2023. From this split, we took one year worth of data as the initial training window to capture any seasonality in the price variation over time. The data for the entire month of Jan 2023 was used to create 63 folds to validate our model. Since ours is time series data, we had to ensure that temporal order of data was maintained in all the folds. Our first fold comprises of an initial training window of 1 year and validation set of 12 hours of data. We make the predictions for these 12 hours and compare it with the actual prices in the validation set to get the errors. Next, we expand the training window by including these 12 hours of data and start predicting for the next 12 hours. This process goes on till 63 folds.

We have chosen the metric as **RMSE - Root mean square error**. Since interpretibility was an important focus for our project, we choose RMSE as it is easily understandable and is in the same scale as the values. The unit is CAD/MW.

Cross Validation Results

In the comparison of various machine learning models for forecasting purposes, the Light GBM model had an RMSE of 89.72 and a Std Dev of 71.60 (Cad/MW). The other models demonstrated varying performance levels. XGBoost, with a slightly higher RMSE of 89.12 and almost identical Std Dev of 77.67, performed nearly as well as Light GBM. The Random Forest Regressor offered decent performance with an RMSE of 144.41 and Std Dev of 72.57, while the ElasticNetCV had an RMSE of 82.85 and the lowest Std Dev of 74.04. The ARIMA model being the baseline had an RMSE of 83.85 and a Std Dev of 73.117. During cross validation, Light GBM, XGBoost and ElasticNetCV performed well, but Light GBM out of all were computationally more efficient and fast and had lower fit times when compared to the rest.

After cross validation, we used the test set for the validation of the model where light GBM generalized well and performed the best with an RMSE of 133.972 and std deviation of 114.48. Elastic Net CV model, being a linear model did not do a good job in the test set and could not generalize well. To conclude, we selected light GBM as our final model

Interpretibility of predictions

For our second scientific objective, which is obtaining the features importances, we relied on the **SHAP (SHapley Additive exPlanations)** which helps us in interpreting and explaining the predictions of our model. For each prediction the model make, we can easily obtain the SHAP values of each of the features for this prediction, which will quantify the impact and contribution of each feature for the prediction. This enable local interpretibility and explain ability of the predictions. We have a base value which is the average value of the predictions made by the model, also called as the model's expected output. This will act as a reference point for comparing the impact of each feature. The SHAP values can be positive or negative. If a SHAP value is positive it means that that feature positively influenced the prediction and pushed the prediction value to be higher than the base score and vice versa for the negative values. So in our dashboard, we have specifically depicted how much percentage increase/decrease each feature contributed in the current prediction when compared to the base score.

Quantifying Uncertainity

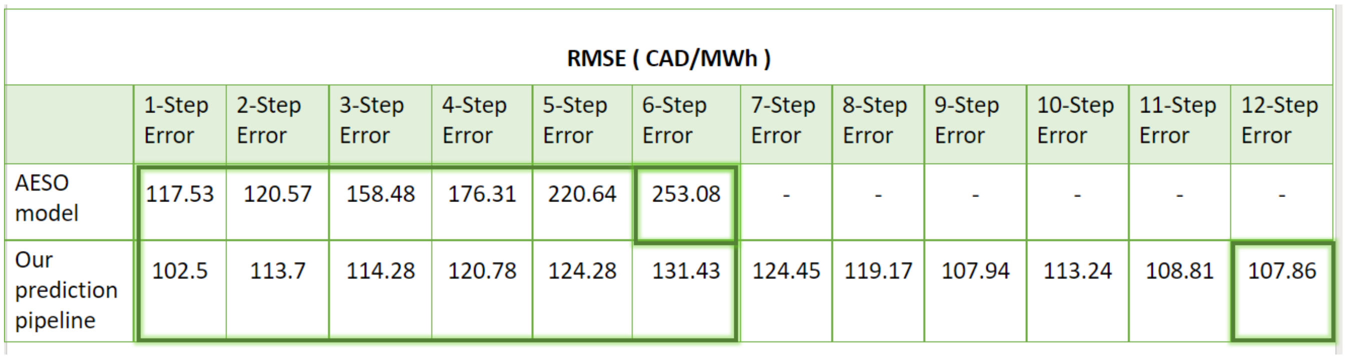
For quantifying the uncertainity of our predictions, we are obtaining the 95% confidence intervals by using Quantile regression. As our final model Light GBM supports quantile regression, we trained two separate lightGBM models which objective as quantile regression. One model was trained to give the upper limit of the confidence interval by configuring the desired quantile as 0.975 (97.5th percentile) and the second model was trained to output the lower limit of the interval by configuring the desired quantile as 0.025 (2.5th percentile). We obtain these limits/predicted quantiles for the 95% confidence interval after training the models by using predict() method for each prediction.

Test Results

Coming to the most awaited test results, our modelling pipeline outperformed AESO's model across all timesteps as shown in the table. Our prediction pipeline demonstrates superiority not just in short-term 1-step and 2-step forecasts but continues to maintain lower RMSE values across multiple steps, right up to 12-step forecasts. Importantly, our model also has the capacity to make forecasts beyond the 6-step limit of the AESO model. This extended range and increased accuracy throughout the forecasting horizon underscore the enhanced performance and superior predictive power of our model compared to the AESO model.

It is noteworthy that the RMSE of our model consistently stays in a relatively lower range, even at higher step predictions. This implies that our model's accuracy doesn't degrade significantly with the extension of the forecast horizon, which is a highly desirable feature in multi-step forecasting models. Overall, this analysis establishes the robustness and efficacy of our prediction pipeline in multi-step forecasting.

We can see below the stepwise errors for the time range of May 25 - May 31st.



RMSE (CAD/MWh)												
	1-Step Error	2-Step Error	3-Step Error	4-Step Error	5-Step Error	6-Step Error	7-Step Error	8-Step Error	9-Step Error	10-Step Error	11-Step Error	12-Step Error
AESO model	117.53	120.57	158.48	176.31	220.64	253.08	-	-	-	-	-	-
Our prediction pipeline	102.5	113.7	114.28	120.78	124.28	131.43	124.45	119.17	107.94	113.24	108.81	107.86

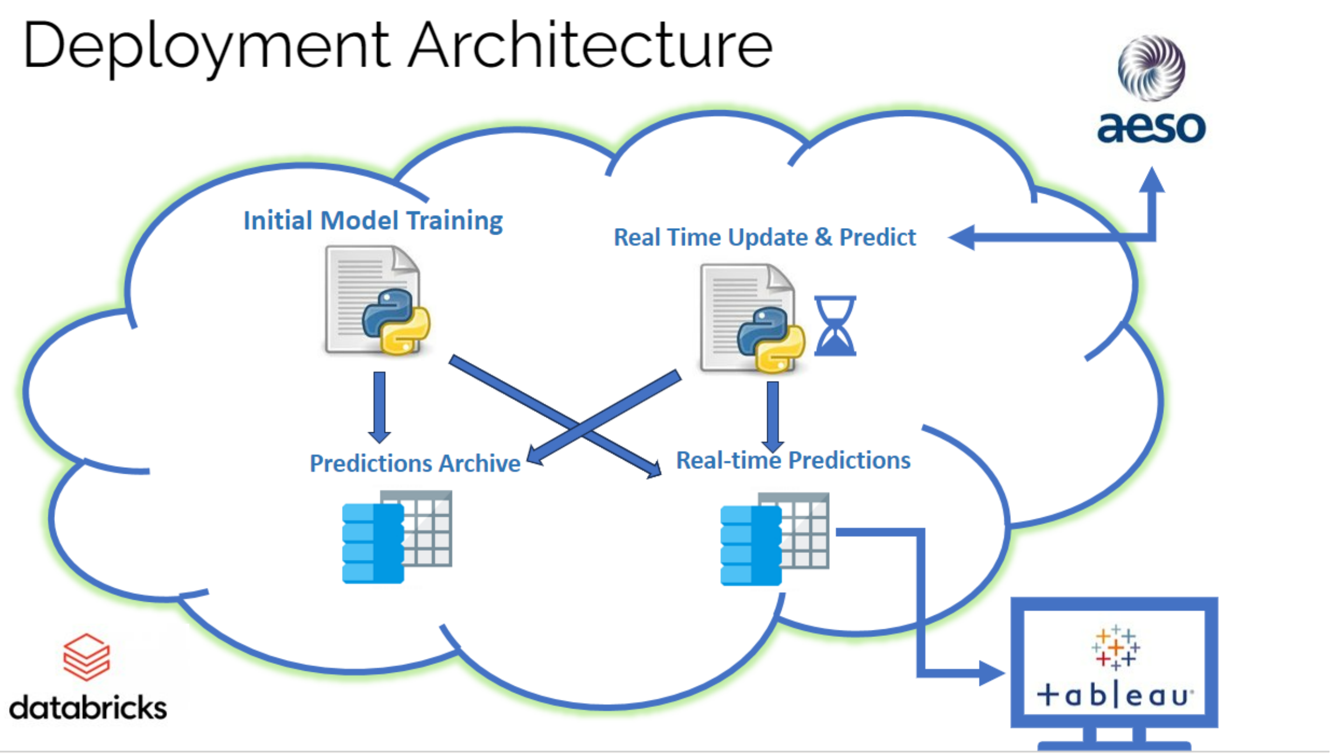
Deployment Architecture

Inorder to make our machine learning pipeline scalable, we have deployed our product in Databricks. Our architecture involves two main jobs running within Databricks. We also have two storage units - one for storing the predictions for dashboard and the second one for archiving all the predictions made by the model. The first job serves as our initial training pipeline, responsible for training the model using the training dataset and subsequently saving it. We also store the model predictions, upper and lower limits, as well as the prediction explanations. These details are stored in both the real-time predictions table and the archive table.

To ensure the continuous availability of updated insights, our tableau dashboard regularly retrieves new data from the real-time predictions table and refreshes its charts accordingly. This mechanism enables us to monitor the latest trends and patterns in a visual and intuitive manner.

In addition to the initial training pipeline, we have implemented an update job that runs on an hourly basis. This job retrieves the new actual power price for the past hour, typically published by the AESO API, every hour. However, due to current limitations in data availability, we simulate this process using historical data. The update job leverages these new values to refit the data in all 12 models, thereby generating the next set of predictions. These updated predictions are then seamlessly integrated with Tableau, allowing us to promptly update our visualizations and plots.

To ensure timely and accurate generation of real-time hourly predictions, the update job operates on a timer trigger, guaranteeing that our predictions and other artifacts remain up to date and aligned with the latest data. This regular and automated update cycle empowers us to gain real-time insights and make informed decisions based on the most recent information available.



Data Product and Result

Our data solution is an integration of a Machine Learning (ML) pipeline and a Tableau dashboard, designed to empower our partners by providing precise forecasts of power prices.

The dashboard is structured into three primary sections. The first section features a 24-hour energy price timeline chart at the top. This graph succinctly represents the recorded energy prices for the previous 12 hours and anticipated prices for the forthcoming 12 hours, complete with a 95% confidence interval for each hour's prediction to uphold the reliability of our forecasts.

The second section unveils a dynamic bar chart located at the lower left, with accompanying explanatory text. This combination efficiently showcases the top four influences on our hourly predictions, while other contributing factors are encapsulated into a single metric for the sake of simplicity.

The third section presents a time series plot of four significant global influences that exhibit a strong correlation with price. These factors include the ratio of total energy produced by gas and wind, as well as the available reserves of each to meet potential demand spikes. The ML pipeline, which refreshes every hour, ensures the values in the Tableau dashboard are continuously updated for real-time accuracy. Our solution is offered in two formats: one that can be configured within Databricks, and another designed for local use, which supports further development.

The Databricks version of our product leverages data from AESO’s Tableau data source. This data is processed and modeled in Databricks, following which the relevant data frames for each 12-hour power price prediction are exported to Databricks' Hive warehouse. This pipeline is seamlessly integrated with Tableau and refreshes every hour. For the local production version, the prediction process is initiated by user command and Tableau workbook is connected to the local CSV files.

We have developed this product with a focus on providing a replicable and manageable solution to our clients, and we trust that the combination of Databricks and Tableau perfectly matches their needs.

On the advantages side, the computing engine in Databricks is scalable and expandable, which is vital for such data-intensive processes. Meanwhile, Tableau, known for its ease of maintenance and multitude of built-in features, serves as an efficient tool for visualizing the predicted results. However, these platforms do present challenges. The computation speed in Databricks is currently slower compared to personal laptops, and Tableau can experience latency when handling large data sources or performing complex calculations. Additionally, there are certain features that are either absent or difficult to implement in Tableau compared to Python's Plotly.

While Plotly might provide a more feature-rich visualization experience, it demands deployment on a separate platform and faces difficulties in maintaining a smooth connection with Databricks data. By utilizing Tableau, we circumvent these issues, making it a more user-friendly solution that can be easily maintained by our partners post-handover, without requiring substantial additional training.

Our solution's prediction results are promising, showcasing an RMSE for 12-step forecasting at 124.56, a significant improvement over AESO’s 6-step ahead prediction of 323.15. We also manage to generate a 95% prediction interval, and interpret our results for better understanding. For instance, if we're predicting the price at 7 AM from 8 hours before, and the predicted price is around USD 107 with a 95% confidence interval ranging from USD 47 to USD 380, we can provide a detailed breakdown of how various features influence this prediction relative to the base value of $105 (the average power price over 2022). This offers a comprehensive understanding of how factors such as gas supply mix, total reserve margin, and weekly profile are affecting the prediction (pushing the prediction upwards), enabling better decision making. Without our product, users would only have access to less accurate prices with greater variations and no explanatory context. We aim to make this process more accurate and comprehensible, thereby enhancing their decision-making capabilities.

Limitations

While this project succeeded in meeting its scientific objectives, it is important to acknowledge some limitations that influenced both the process and outcomes:

Price is complex

Price is determined when markets clear, i.e. when demand and supply match. Any factor affecting either the supply or demand side can potentially impact the price dynamics. Therefore, price is by no means a simple identity. Power prices in a liberalized market like Alberta are also influenced by numerous interconnected factors. While this project's machine learning pipeline accounted for several key variables, accurately predicting power prices requires considering a wide range of factors, including unexpected events and unusual circumstances. For example, factors such as unforeseen failures, interruptions, or disruptions in power generation can significantly impact supply and subsequently affect prices. Furthermore, economic downturns or booms, and even fluctuations in the stock market can all contribute to power price fluctuations through various mechanisms. Apparently, in the short duration of this project, It was not possible to include all of the considerable factors into the model, or even identify them all.

High volatility means high uncertainty

The power price data in Alberta exhibited a lack of clear patterns or trends, making it challenging to develop accurate predictions. Also, both average and variance of the power price has been consistantly increasing in past few years. This upward trend in both the average price and the volatility of prices adds another layer of complexity to the forecasting task. The seasonal influences were intertwined with extreme price volatility, further increasing the uncertainty. As a result, accurately forecasting power prices beyond a short-term window remains difficult. Also, as we extended our forecast window to 12 hours, the inherent uncertainty in predicting power prices became more pronounced.

Data availability, a real binding constraint

The availability of reliable and comprehensive data posed significant challenges for our project, particularly due to the requirement of predicting power prices on an hourly basis in real time. The availability of timely and accurate data is crucial for the success of our forecasting model.

Even when we identified valuable features with significant associations to price, obtaining them in an hourly and real-time format proved to be a formidable task. The continuous and up-to-date data required to feed into our model was not readily accessible. As a result, we had to rely on a combination of historical data obtained from Tableau and real-time data obtained through APIs for some specific features. This reliance on multiple data sources along with limited availability of data for many features, resulted in a patchwork of data, which introduced complexities and potential limitations to the accuracy and robustness of our model.

Striking the Balance: the Accuracy-Explainability Tradeoff

Another intriguing aspect we encountered was the classic tug-of-war between accuracy and interpretability. We all know they often pull in opposite directions. Striking a balance between these two was an ongoing challenge in this project. Simpler models like ARIMA provided better interpretability but lacked the sophistication to handle the complexity of this project's problem. On the other hand, more intricate models offered higher accuracy but were often considered black-box models, which compromised interpretability. We opted for the LightGBM model, which provided a middle ground, but achieving the ideal balance remains an area of exploration. Navigating the accuracy-explainability tradeoff does not have a straightforward recipe, as there are numerous potential models that can still be explored.

Features with known future values are not in yet

In this project, the selected model paradigm relied on predicting power prices without reliable knowledge of future features for the next 12 hours. This limitation arose due to the unavailability of data specifically pertaining to those features. However, it is worth noting that the future values, either forecasted or actual, for certain related features can be accessible.

By incorporating a model paradigm that utilizes features with known future values, there is a potential to enhance the accuracy of the forecasting performance of this project. These related features can provide valuable insights into the dynamics of the power market, contributing to a more comprehensive and precise prediction of power prices. By leveraging available information on forecasted or actual values in the coming hours, we can potentially improve the model's forecasting capabilities and capture short-term variations in power prices more accurately.