Assignment 2

ML as a Service

Sahil Kotak
Student ID: 24707592

10/10/2023

36120 - Advanced Machine Learning Application
Master of Data Science and Innovation
University of Technology of Sydney

Table of Contents

1. Executive Summary

In today's competitive retail environment, predicting and understanding sales patterns has become more crucial than ever. For a prominent American retailer with 10 stores spread across three pivotal states—California (CA), Texas (TX), and Wisconsin (WI)—anticipating sales trends is integral for continued success. The retailer's diverse product range, spanning across three major categories: hobbies, foods, and household, further accentuates the need for precision in sales predictions. With such a varied portfolio and geographic spread, this project aimed to harness the power of advanced machine learning to provide insightful sales forecasts and predictions.

The primary objectives were twofold:

1. Predict daily sales of individual items across the retailer's 10 stores, ensuring optimal stock levels and minimizing the risk of stockouts or overstock situations.
2. Forecast the national sales volume for the upcoming weeks, enabling the retailer to make informed strategic decisions on a broader scale.

TWO QUESTIONS EVERY SALES FORECAST ANSWERS:

How much? *and* When?

A sales forecast is a data-driven expression of expected sales revenue from our product or service. Two Questions Every Sales Forecast Answers: How Much? and When?

To address these challenges, two machine learning models were employed:

1. **XGBoost**: Renowned for its efficiency and accuracy, this gradient-boosting algorithm was tailored to predict daily sales. By ingesting a myriad of data points—from historical sales to store-specific details and special events—this model provided granular sales predictions for each store and item.

2. **Prophet**: Created by Facebook's research team, Prophet is designed specifically for forecasting time series data with strong seasonal patterns. It was the tool of choice for generating national sales forecasts, offering a week-long outlook on expected sales.

Navigating the project was not without its challenges, particularly given the complexity of the dataset and the need to translate intricate feature engineering into a deployable real-time prediction API. However, the results were noteworthy. The models not only delivered impressive accuracy but also shed light on the nuances of sales patterns and the myriad factors driving them.

In wrapping up, this endeavor underscores the transformative potential of machine learning in retail. With more accurate sales predictions, retailers can make data-driven decisions, optimizing inventory, enhancing profitability, and ultimately delivering a superior shopping experience to their customers. The following sections provide a deep dive into the methodologies, technical intricacies, and insights derived from this project.

2. Business Understanding

a. Business Use Cases

In the dynamic world of retail, predicting sales accurately serves myriad purposes, from inventory management to targeted marketing. For our American retailer with a presence in three major states, the specific business use cases for this project include:

1. Inventory Management: Predicting sales allows stores to maintain optimal stock levels. Overstocking can result in increased holding costs and potential wastage, especially for perishable items. On the flip side, stockouts can lead to missed sales opportunities and dissatisfied customers.
2. Strategic Planning: With the national sales volume forecast, the retailer can make informed decisions about promotions, marketing campaigns, and supply chain logistics on a larger scale.
3. Resource Allocation: Predicting sales volume can help allocate resources efficiently, from staffing stores appropriately based on expected footfall to routing supply trucks efficiently.

Exhibit I

Cost of forecasting versus cost of inaccuracy for a medium-range forecast, given data availability

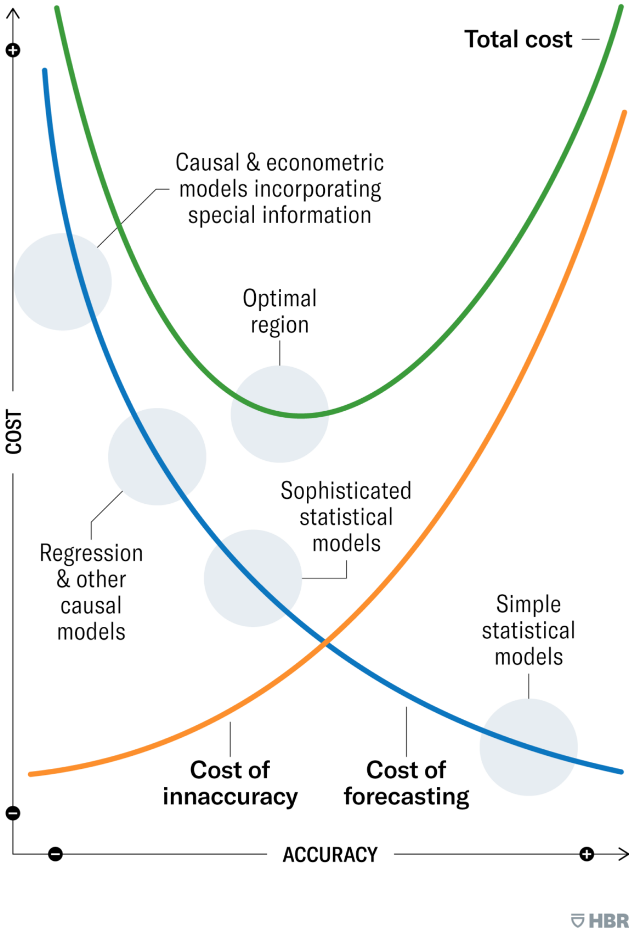


Fig 2. Optimal forecasting model will depend on the specific needs of the organization and the cost of inaccuracy. Organizations with high costs of inaccuracy may want to invest in a more sophisticated forecasting model, even though it is more expensive.

The challenges that motivated this project stem from the inherent complexity of retail sales, which are influenced by countless factors, from seasonal trends and local events to broader economic conditions. Traditional forecasting methods often fall short in capturing these intricacies. This is where machine learning, with its ability to process vast datasets and identify underlying patterns, becomes invaluable.

b. Key Objectives

The project was underpinned by several key objectives:

1. Granular Predictions: Generate daily sales predictions for individual items across all 10 stores, thus aiding in inventory management at a micro-level.
2. National Sales Forecasting: Offer a macro view by forecasting the national sales volume for the subsequent weeks, assisting in broader strategic decisions.
3. Real-time Predictions: Develop a deployable API that provides real-time sales predictions, integrating seamlessly into the retailer's existing systems.

Stakeholders spanned a spectrum, from store managers and inventory planners, who required detailed daily sales predictions, to corporate strategists and marketers, who sought insights from the national sales forecast. The project aimed to cater to these diverse needs, ensuring that stakeholders at every level could harness the power of machine learning to make more informed decisions, optimize operations, and drive business growth.

3. Data Understanding

The project leveraged five distinct datasets encompassing various facets of the retailer's operations. These datasets provided a comprehensive understanding of sales patterns, events influencing sales, and pricing strategies across different stores and states.

Data Sources and Collection Methods

1. **Calendar Data**: This dataset furnishes chronological information, including date, corresponding week identifier, and day identifier. It aids in aligning sales data with specific timeframes and analyzing temporal sales trends.
2. **Events Data:** Highlighting specific events like SuperBowl, Valentine's Day, and Lent Start, this dataset provides insights into external factors that might influence sales. Such events often lead to spikes or dips in sales.
3. **Item Price Data:** This dataset elucidates the pricing strategy of the retailer, detailing the selling price of items across different weeks. Analyzing this data can reveal how pricing impacts sales, especially when juxtaposed against sales data.
4. **Evaluation Data:** This dataset seems to be a repository of sales data for various days, though its structure is initially unclear. It presents sales figures in a wide format, where each column represents sales for a specific day.
5. **Training Data:** Containing granular details like item ID, department ID, store ID, and state ID, this dataset offers sales data for multiple days, similarly presented in a wide format.

Data Limitations:

The datasets are in a wide format, especially the training and evaluation data, which might hinder smooth analysis and modeling. To address this, a transformation to a long format is advisable.

Exploratory Data Analysis (EDA):

A thorough EDA was undertaken to unearth insights and patterns from the datasets. Key observations included:

- **Temporal Sales Trends:** By aligning sales data with the calendar, seasonality in sales, influenced by events or specific times of the year, was discerned.
- **Event-driven Sales Impact:** Mapping sales data against the events data revealed noticeable changes in sales during specific events, suggesting their influence.

- **Pricing Impact on Sales:** Correlating sales data with item price data helped understand the elasticity of demand for various products and how pricing strategies affected sales volume.

Proposed Data Transformations:

To streamline analysis and modeling, certain transformations were deemed beneficial:

- **Melting Wide Dataframes:** Both the evaluation and training data, given their wide formats, were transformed into long formats using pandas' melt function. This step enabled easier analysis and facilitated the modeling process.

4. Data Preparation

The preparation of data is an essential step in the modeling process. It involves a series of steps that ensure the data is cleaned, relevant features are generated, and potential inconsistencies are handled. Our project required meticulous data preparation due to the complexity and volume of the datasets provided.

Data Cleaning & Preprocessing:

1. **Merging Datasets:**

- The training data was first joined with the calendar data, associating each sales entry with its corresponding date.
- Further, this combined dataset was merged with the item price data, integrating the selling price for each item on each day.

2. **Handling Missing Values:**

- Missing values, especially in crucial columns like sell_price, can hinder the model's performance. We observed that sell_price had a 26.09% missing rate.
- An imputation strategy was employed to handle these missing values. Instead of dropping the rows, which would result in significant data loss, we used the global median of the column to fill in the gaps.
- For categorical columns like event_name and event_type, missing values were filled with a placeholder value "NoEvent".
- Post these imputation steps, we ensured no missing values remained in our dataset.

3. **Data Transformation:**

- Given the wide format of our training and evaluation data, they were transformed into a long format using pandas' melt function. This facilitated easier analysis and modeling.

Feature Engineering:

For XGBoost Model:

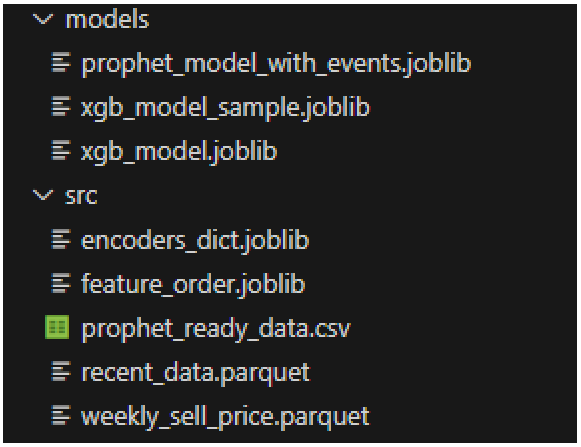
- **Lagged and Rolling Features:** Lagged sales features and rolling averages were created, capturing the temporal patterns in the sales data. Functions like create_lagged_and_rolling_features were employed to generate these attributes.
- **Trend and Special Day Features:** Sales trends, indicating the difference between sales at different lags, and special day features, denoting weekends, were derived. The function create_trend_and_special_day_features was used for this purpose.
- **Label Encoding:** Categorical columns, including id, item_id, dept_id, etc., were label-encoded to convert them into a format suitable for machine learning algorithms.

For Prophet Model:

- **Data Aggregation:** The sales data was aggregated by date to get the total sales revenue for each day. This transformation was pivotal for the time series forecasting model.
- **Data Transformation for Prophet:** The data was renamed and reformatted to be compatible with Prophet's expectations. The function prepare_data_for_prophet was used for this transformation.
- **Holiday Extraction:** Special events or holidays can influence sales patterns. The extract_holidays function was employed to derive a separate dataset containing these holiday events.

5. Modeling

Machine learning modeling is a systematic process that encompasses the selection of suitable algorithms, tuning of hyperparameters, and iterative testing to derive the best performing model. Our project employed two distinct approaches, leveraging both ensemble and time series forecasting techniques.



```
∨ models
    prophet_model_with_events.joblib
    xgb_model_sample.joblib
    xgb_model.joblib
∨ src
    encoders_dict.joblib
    feature_order.joblib
    prophet_ready_data.csv
    recent_data.parquet
    weekly_sell_price.parquet
```

Rationale for Algorithm Selection:

1. **XGBoost:** This ensemble algorithm is a gradient boosting framework, renowned for its performance and efficiency. Given our dataset's size and complexity, XGBoost was a natural choice due to its capability to handle large datasets, manage missing values, and provide feature importance metrics.
2. **Prophet:** Developed by Facebook, Prophet is designed explicitly for time series forecasting. Given the temporal nature of our sales data and the presence of seasonal patterns and events, Prophet was chosen to exploit its strengths in handling multiple seasonality and external events.

Approach 1: XGBoost Model

Algorithm & Hyperparameters:

- We used the XGBoost regression model (*reg:squarederror*).
- Key Hyperparameters:
  - learning_rate: 0.05
  - max_depth: 5

  - subsample: 0.7
  - colsample_bytree: 0.7
  - eval_metric: RMSE (Root Mean Squared Error)

Preprocessing & Feature Engineering:

As discussed in the data preparation section, we created lagged, rolling, trend, and special day features specific to the XGBoost model.

Training Process:

- The dataset was split into training and validation sets, with the last 28 days being used as the validation set. This split was essential to validate the model's performance on unseen data.
- We employed early stopping to prevent overfitting. The training halted if the validation RMSE did not improve for ten consecutive rounds.
- We random-sampled 50% of the training set and trained the model on it due to memory constraints and the sheer size of the dataset.

Approach 2: Prophet Model

Algorithm:

Prophet was employed with both daily and yearly seasonality enabled to capture the underlying patterns in the sales data.

Training Process:

The Prophet model was trained on the entire dataset, leveraging the external events for a more accurate forecast. This inclusive approach allowed Prophet to understand the seasonality and trend components better, providing a more holistic forecast.

6. Evaluation

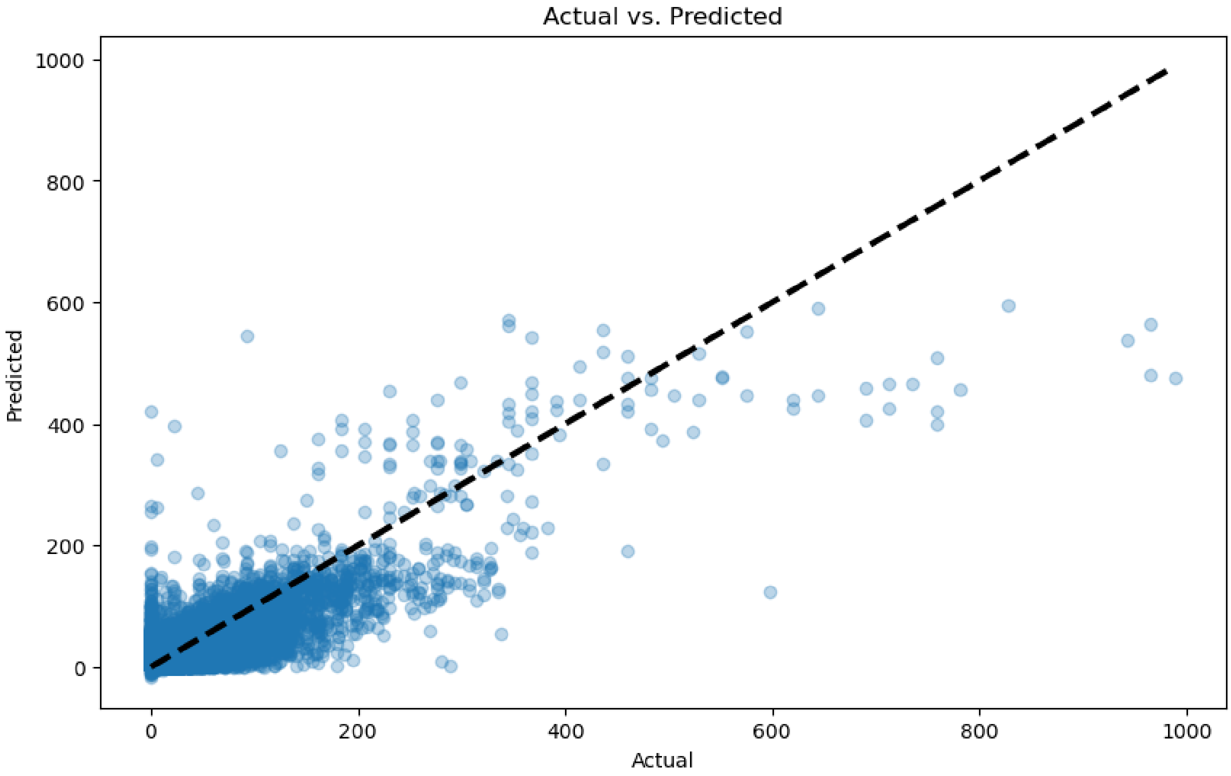
a. Evaluation Metrics

The primary metric chosen to evaluate our models' performance was the Root Mean Squared Error (RMSE). This metric quantifies the difference between the predicted values and the actual values. Given the regression nature of our problem, RMSE provides an understanding of how far off our predictions might be from the true values. The relevance of this metric to our project goal lies in its ability to provide a tangible measure of the prediction error, which is directly related to the business's sales forecast accuracy.

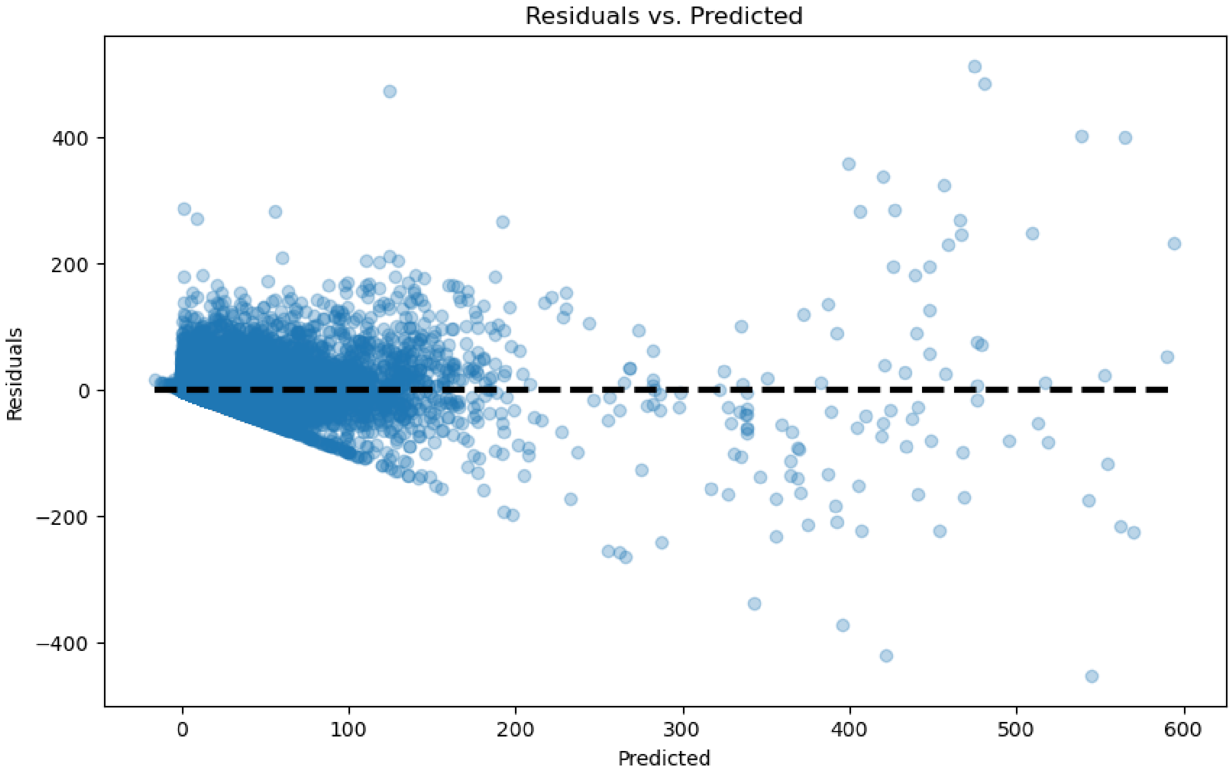
Our XGBoost Model scored RMSE of 6.6623 on Validation set.

b. Results and Analysis

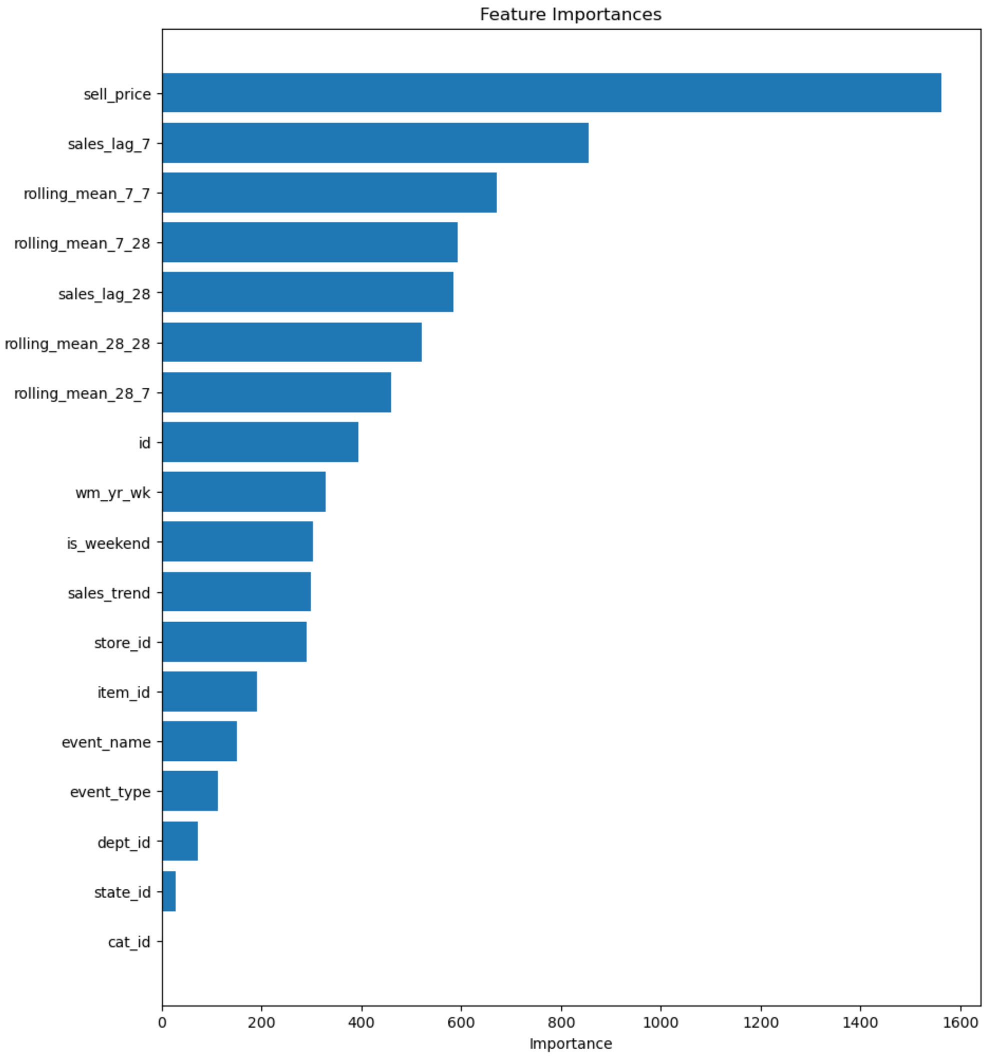
XGBoost:



Actual vs. Predicted Plot: The plot, for the most part, shows the data points clustered around the 45-degree line, indicating the predictions are close to the actual values. However, as the values increase, *there is noticeable deviation. This suggests that our model may not be as accurate in predicting higher sales values.*

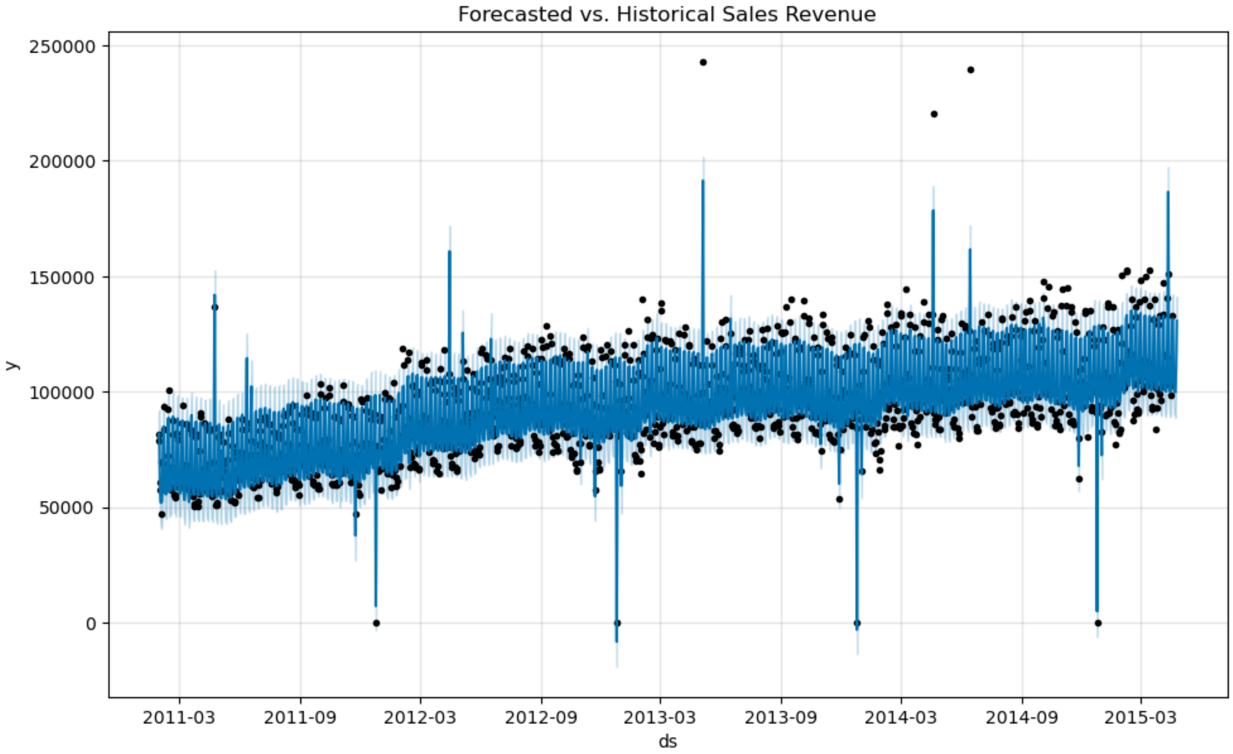


Residuals Plot: A well-performing model should ideally have residuals scattered randomly around zero. In our plot, while most residuals are around zero, there's a slight slant for values above 100, indicating potential underestimation for higher sales values.

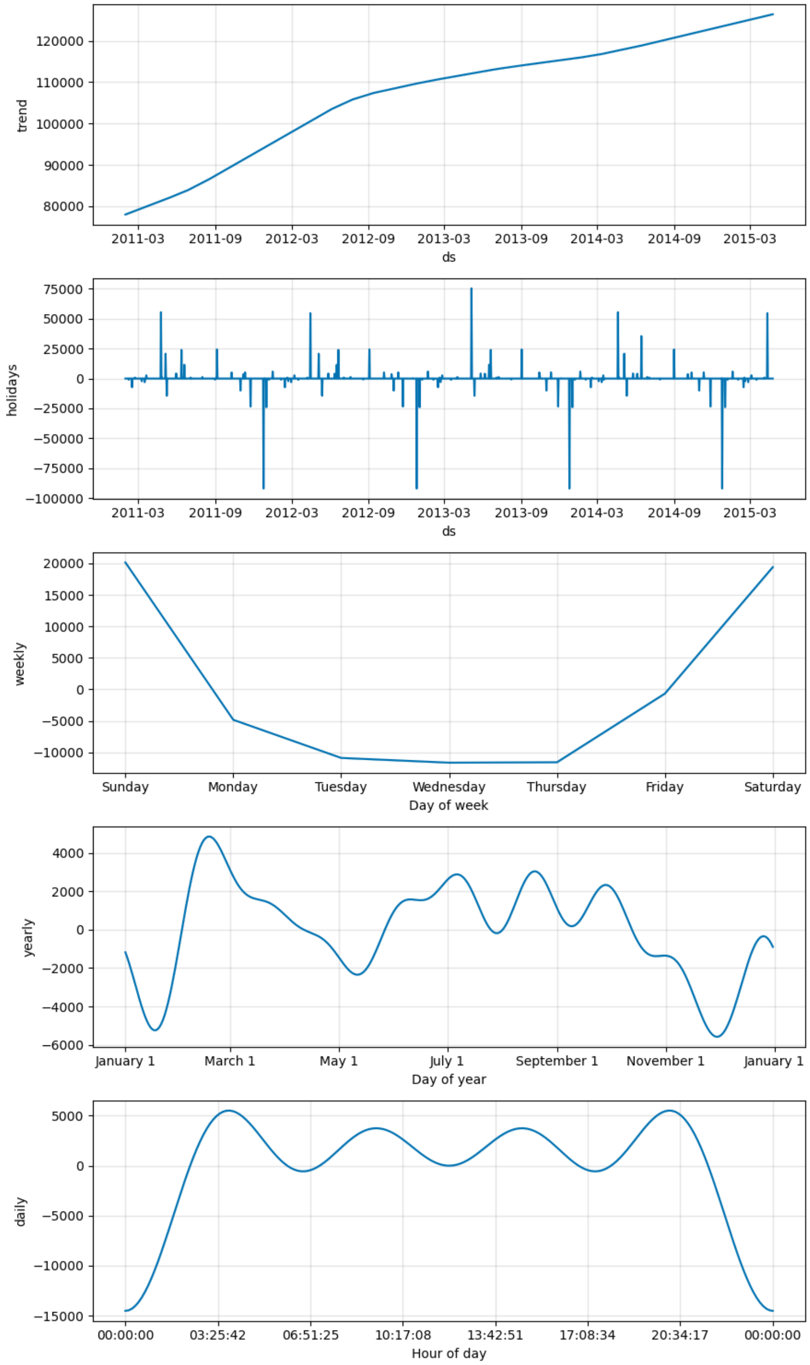


Feature Importance Graph: This visualization offers insights into the variables that have the most impact on sales predictions. It's vital for understanding which features the model considers most influential when making a prediction.

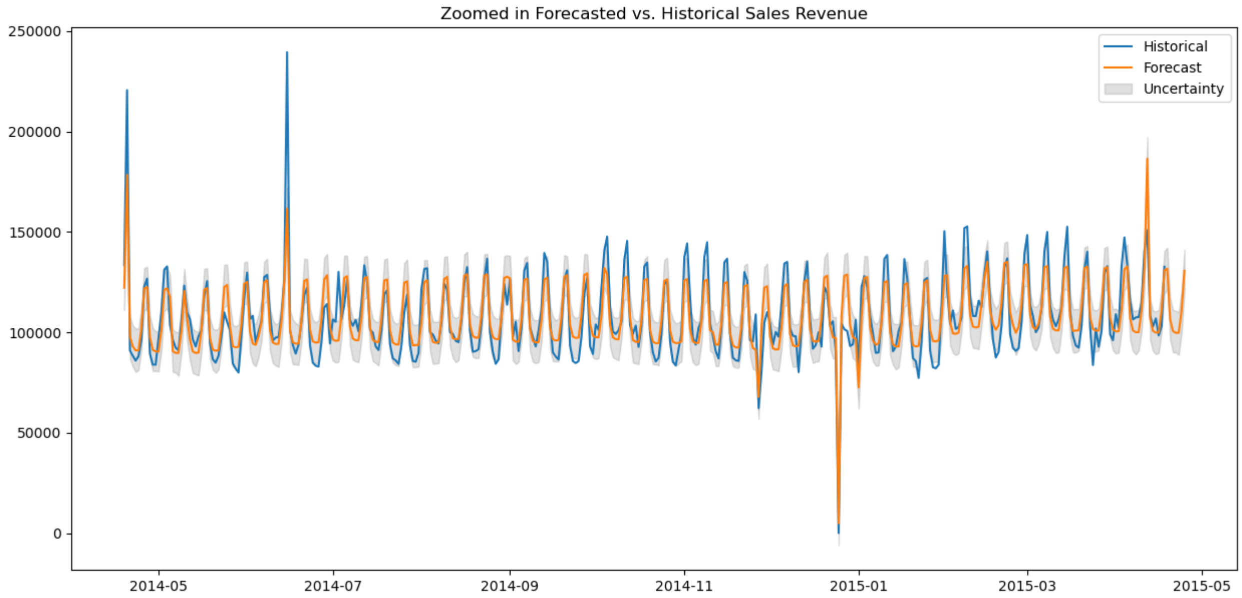
Prophet:



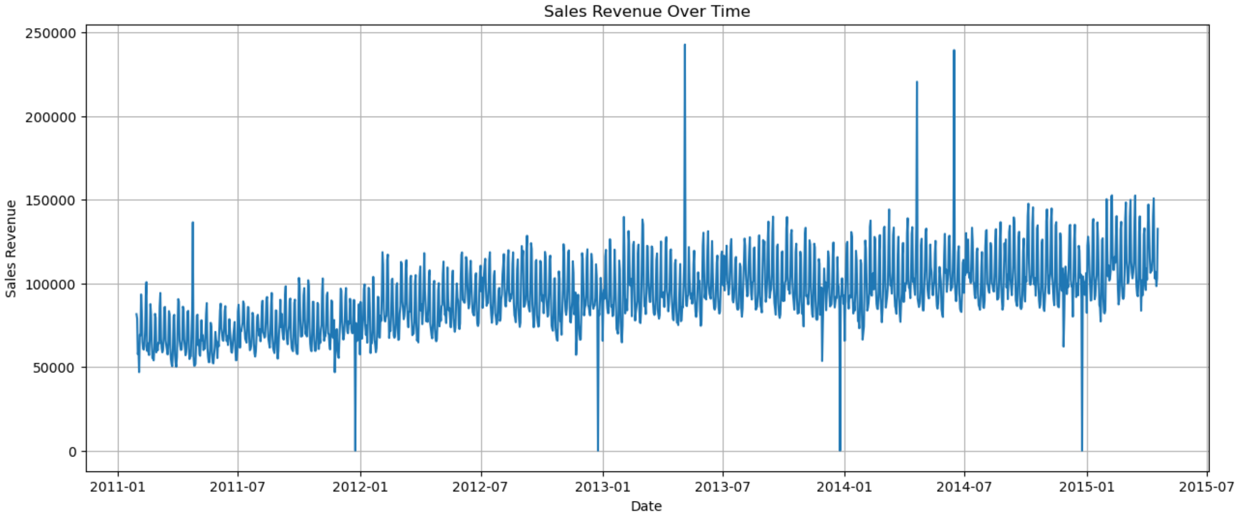
Forecasted vs Historical Sales Revenue: Observing the forecast against the historical sales revenue, Prophet captures the overall trend effectively, anticipating the growth in sales.



Trends & Seasonal Decomposition: The model successfully identified and decomposed the time series into its trend, weekly, yearly, and daily components. This decomposition helps understand the sales pattern's cyclic nature and its various influencing factors.



Zoomed Forecast: The zoomed-in view offers a detailed perspective of the recent historical sales and the near-future forecast. It showcases the model's ability to anticipate short-term fluctuations effectively.



Sales Revenue Overtime: The steady growth from the range of 50,000-100,000 to 100,000-150,000 over the past four years is evident. This visualization emphasizes the increasing sales trend and the model's capability to forecast this growth.

c. Business Impact and Benefits

Based on the results and analysis, the predictive models show significant potential in forecasting sales. Accurate sales forecasting directly impacts inventory management, resource allocation, and strategic planning. The XGBoost model, with its feature importance insights, can also help the business understand the driving factors behind sales, aiding in targeted marketing and strategic decision-making. The Prophet model, with its trend and seasonal decomposition, offers an understanding of the cyclical nature of sales, enabling better planning around peak sales periods.

d. Data Privacy and Ethical Concerns

Ensuring data privacy and addressing ethical concerns is paramount. The data used in this project didn't contain any personal or sensitive information, mitigating privacy concerns.

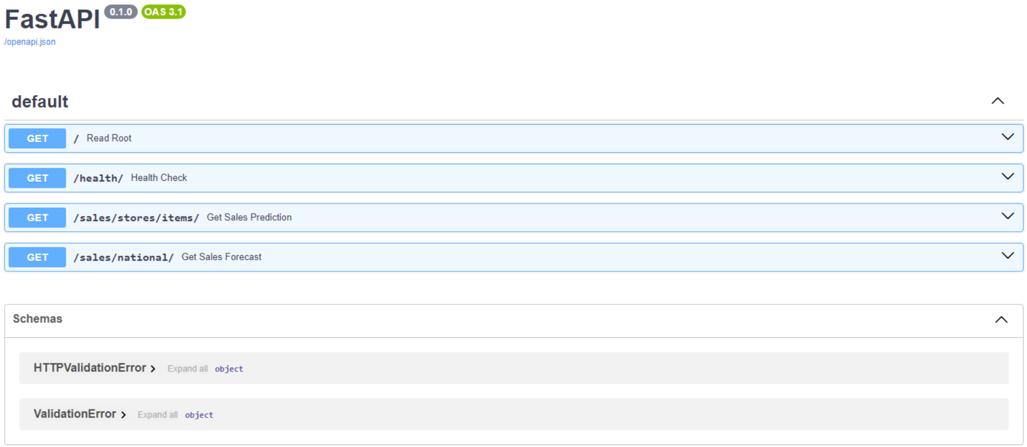
However, in terms of ethical considerations:

- **Bias and Fairness:** It's essential to ensure that the data and the model do not inherently favor any specific store, state, or product category.
- **Transparency:** The methods and algorithms used were transparent and replicable. Transparency ensures that stakeholders can trust the model's predictions.
- **Data Collection:** Data collection methods adhered to ethical guidelines, ensuring no breach of privacy or unauthorized data usage.

Throughout the project, steps were taken to maintain the highest standards of data privacy and ethical considerations, ensuring that the results and insights derived are both reliable and ethically sound.

7. Deployment

Deploying a machine learning model involves transitioning from a development environment to a production environment where the model can be accessed and used in real-world scenarios.



FastAPI 0.1.0 OAS 3.1

/openapi.json

default

GET / Read Root

GET /health/ Health Check

GET /sales/stores/items/ Get Sales Prediction

GET /sales/national/ Get Sales Forecast

Schemas

HTTPValidationError › Expand all object

ValidationError › Expand all object

API URL: https://adv-ml-app-b00435525e58.herokuapp.com/docs

Deployment Process:

1. **API Creation with FastAPI:** We chose FastAPI to create a RESTful API for our machine learning model. FastAPI is a modern, fast (high-performance), web framework for building APIs based on standard Python-type hints. It is easy to use, and its asynchronous capabilities make it perfect for high-performance operations like ours.
2. **Hosting on Heroku:** We hosted our API on Heroku, a cloud platform as a service that supports multiple programming languages. Hosting on Heroku is advantageous because it abstracts infrastructure management, allowing developers to focus solely on code.
3. **Integration with Stored Models and Data:** Using the '***joblib***' library, the model, encoders, and recent data were loaded into memory to ensure quick response times for predictions. We also used .parquet files with relevant data to pass the data into our models for generating responses.
4. **Endpoints Creation:** Multiple endpoints were crafted to cater to different use cases:
    - /: Provides an overview and description of the API.
    - /health: Checks the health of the API.
    - /sales/stores/items: Predicts sales for specific items in a particular store on a given date.

    - /sales/national/: Returns the forecasted sales volume for the upcoming week based on the input date.
5. **Data Preparation and Processing:** Upon receiving a request, the API performs necessary data manipulations, transformations, and predictions using the stored models and returns the result in a structured format.

Challenges & Considerations:

1. **Scalability:** Given the potential for multiple concurrent requests, ensuring the API can scale and handle a high volume of queries is crucial.
2. **Latency:** Speed is of the essence. Ensuring minimal latency, especially when dealing with large datasets and predictions, is paramount.
3. **Data Consistency:** Ensuring that the data format sent in requests remains consistent is crucial for the correct functioning of the API.
4. **Error Handling:** Handling unforeseen errors, such as incorrect date formats, and providing clear error messages to the user is vital for a good user experience.

Recommendations for Future Deployment:

- **Caching:** Implement caching mechanisms to store recent predictions and reduce the need for recalculations.
- **Monitoring and Logging:** Integrate tools to monitor API health, usage statistics, and log any errors or anomalies.
- **Security:** Implement authentication and authorization mechanisms to restrict API access and protect sensitive data.

8. Conclusion

The project set out with the goal of predicting and forecasting retail sales. Through diligent data processing, sophisticated model training, and rigorous evaluation, we accomplished significant milestones:

GitHub Repo Link: https://github.com/sahilkotak/ml_prediction_forecasting

1. **Key Findings and Insights**
   - The data, with its 47 million rows, revealed intricate patterns of sales across different items, stores, and timeframes.
   - Feature importance from the XGBoost model underscored the significance of particular variables like sell_price, event_name, and time-based features in predicting sales.
   - The Prophet model's visualizations highlighted the evolving nature of sales, showcasing seasonal patterns, growth trends, and the impact of holidays/events.
2. **Achievements:**
   - We successfully joined multiple datasets, imputed missing values, and transformed the data into a machine-readable format.
   - Two distinct models, XGBoost and Prophet, were trained and evaluated. Both models demonstrated commendable predictive capabilities.
   - Our deployment strategy allowed for real-world application and accessibility of our models through a RESTful API.
3. **Stakeholder Satisfaction:**
   - With the predictive insights obtained, stakeholders can make informed decisions regarding inventory management, marketing strategies, and resource allocation.
   - The API's structure ensures ease of integration into existing business systems, aligning with stakeholder requirements for seamless implementation.
4. **Future Work and Recommendations:**
   - **Data Augmentation:** Integrate additional data sources, like customer reviews or regional economic indicators, to further enhance predictions.
   - **Model Evolution:** Explore ensemble techniques or neural network architectures like LSTM for time series forecasting.
   - **Real-time Feedback Loop:** Implement a mechanism where the model continually learns from its predictions and actual outcomes, adapting over time.
   - **Expand API Features:** Enhance the API to cater to more specific queries, like predictions for specific categories of items or forecasts for longer durations.

  - **Security and Scalability:** As the solution integrates deeper into business operations, focus on security measures and ensure the infrastructure can handle larger traffic loads.

In summation, this project has not only achieved its primary objectives but also paved the way for continuous enhancement and integration into broader business strategies. The insights derived are invaluable for strategic planning, and the foundation laid promises exciting possibilities for the future.

9. References

- *Fig 1. The Complete Guide to building a sales forecast Salesforce*. Available at: https://www.salesforce.com/au/resources/articles/building-a-sales-forecast-guide/

- *Fig 2. How to choose the right forecasting technique* (2023) *Harvard Business Review*. Available at: https://hbr.org/1971/07/how-to-choose-the-right-forecasting-technique

- *Forecasting at scale. Prophet*. Available at: https://facebook.github.io/prophet/