

Assignment 2

Machine Learning as a Service

[Item level sales prediction and Forecast as a service for a Retail Store](#)

Github Username	shrestha1
Github Repos	Experiment Repo: shrestha1/AT2-ADML-24996124 (github.com) API Repo: shrestha1/AT2-ADML-24996124-API: Machine Learning As a Service (github.com) Streamlit Repo: shrestha1/AT2-ADML-24996124-API: Machine Learning As a Service (github.com)
URLs	Backend: https://at2-adml-24996124-backend-latest.onrender.com/ Frontend: https://at2-adml-24996124-frontend-latest.onrender.com/

Dipesh Shrestha
Student ID: 24996124

36106 - Machine Learning Algorithms and Applications
Master of Data Science and Innovation
University of Technology of Sydney

Table of Contents

Executive Summary

1. Business Understanding	2
a. Business Use Cases	3
b. Key Objectives	3
2. Data Understanding	4
3. Data Preparation	5
4. Modeling	11
5. Evaluation	14
a. Results and Analysis	14
b. Business Impact and Benefits	18
c. Data Privacy and Ethical Concerns	20
6. Deployment	21
7. Conclusion	22
8. References	23

Executive Summary

The retail business in the U.S has experienced remarkable growth, largely due to its ability to adapt to evolving consumer demands. By introducing in-demand items, these businesses would successfully generate substantial revenue. However, underlying risks persist due to periodic events, such as festivals or seasonal fluctuations, along with several factors that might lead to lower sales of certain items while simultaneously driving higher demand for others.

To mitigate these risks, businesses must be proactive and prepare for these trends, which can often be anticipated through data analysis. Embracing a data driven approach will enhance awareness of sales patterns and revenue generation. Consequently, leveraging machine learning techniques to predict which items should be sold during specific periods and to forecast their sales can significantly support business growth and resilience. Furthermore, utilizing machine learning models as a service offers businesses the advantage of on-the-go predictions, enabling them to make timely and informed decisions, optimize inventory management and drive sales performance.

To perform item level sales prediction and forecasting different models a series of experiments were performed with evaluation metrics as R2 score provided a priority with less emphasis on the RMSE and MAE scores. The item level sales prediction model could evaluate 61% of sales variability whereas the total forecast could answer for 81% with error of around 7K. The project was carried out using CRISP-DM methodology to incorporate iterative experimentation and testing.

1. Business Understanding

a. Business Use Cases

Machine learning as a service is increasingly being adopted by retailers to enhance operational efficiency and improve decision-making processes. This project specifically focuses on item-level sales prediction for individual stores and total sales forecasting across all stores. This prediction provides retailers with insights for inventory management, marketing strategies, and long-term planning.

1. Item Level Sales Prediction

- **Business use cases:** Predicting the sales of specific items in a particular store allows retailers to optimize inventory, avoid stockouts, and reduce overstocking. It helps retailers ensure they have the right products available at the right time, enhancing customer satisfaction and increasing sales.
- **Challenges:** Variations in sales due to factors like seasonality, promotions, economic conditions, and customer preferences can make sales predictions complex. Item level sales are often influenced by unpredictable local events, requiring highly granular data and models.
- **Opportunity:** By accurately predicting item-level sales retailers can improve their replenishment processes, minimize waste, and increase profitability. Tailored marketing campaigns can be developed to target specific product demand.

2. Total Sales Forecast

Business use cases: Forecasting overall sales across all stores is essential for strategic planning, budgeting, and resource allocation. Retailers use total sales forecasts to set corporate goals, plan marketing strategies, and manage supply chain operations.

Challenges: Total sales must account for broader market trends, economic factors, competitive actions, and regional preference. Fluctuation in consumer behavior due to external factors (e.g. holidays, pandemics or economic downturns) make accurate forecasts more difficult.

Opportunity: Accurate sales forecasting enables better financial planning, optimized staffing, and more efficient supply chain management, leading to reduced costs and improved customer services.

b. Key Objectives

The primary objective of this project is to develop and deploy machine learning models capable of predicting item-level sales for specific products in a store on a given date, as well as forecasting total sales across all stores. These predictions aim to drive business growth by enabling better decision-making and operational efficiency. The trained models will eventually be integrated into a production environment as a service.

Key Goal of the Project

- **Item-Level sales prediction:** Accurately predict the sales of individual items for a specific store on a particular date to optimize inventory, reduce
- **Total Sales Forecasting:** Provide forecast of total sales across the entire store network.
- **Model deployment:** Successfully deploy the trained models as a service to facilitate real time predictions and continuous model updates based on incoming data, enhancing the retailer's ability to respond quickly to market changes

Stakeholders	Use case
Retail Managers	To optimize inventory management, resource allocation, and staffing based on anticipated demand.
Marketing and Sales Teams	To create targeted marketing campaigns and promotions aligning efforts with periods of high potential demand to maximize sales and customer reach.
Financial and Executive Leadership	To monitor financial performance, adjust budgets, plan investments, and make strategic business decisions based on future sales projection and seasonal trends.

Machine Learning approach can model complex interactions between numerical variables such as time, and the sales fluctuation due to different factors. Also, machine learning models are excellent in identifying patterns in large datasets which are critical for prediction and forecasting over time. ML can learn from historical data and provide prediction and forecast as new data comes in, making it particularly relevant for this use case.



2. Data Understanding

The provided data came in CSV files, separating training and evaluation datasets, as well as additional files for the calendar, events, and weekly item prices. The data contains sales records from 10 stores of an American retailer, spanning three states: California, Texas and Wisconsin. Despite being structured into various CSV files, there was no information confirming the legitimacy of the data source, raising concerns about its reliability and quality. However, the information about the data is listed in the table below.

Data	Shape	Data type	Description
Training Data	Rows: 30490 Column: 1547	Categorical: Id, item_id, dept_id, cat_id, store_id and state_id Numerical: d1-d1541 each day sales quantity up to 1541 days	It includes the record sales quantity of each item in a store for 1541 days in a long format. Memory usage: 359.9+ MB
Evaluation Data	Entries: 30490 Column: 400	Numerical: d_1542- d_1941	Includes the sales quantity of the items from day count 1542 to 1941.
Calendar	Entries: 1969 Column: 3	Categorical: date and d Numerical: wm_yr_wk	It includes information especially about the date. The wm_yr_wk is encoded form to represent particular year and its week. Similar d represents the information amount day past. Like d_1, d_2 etc. This has information of 1969 days from the date 2011-01-29 as first day. Memory usage: 46.3+ KB
Calendar events	Entries: 167 Column: 3	Categorical: date, event_name and event_type	It provides information about any events occurred in the particular date.
Weekly Sales	Entries: 6841121 Columns: 4	Categorical: store_id, item_id, Numerical: wm_yr_wk and sell_price	Includes information about the unit sales price of an item in a stored at given year week.



3. Data Preparation

Exploratory data analysis was not conducted on the raw data; instead, the data underwent a preparation phase tailored to the specific use cases before any further analysis was performed.

I) Item Level Sales Prediction

Initially the training data was transformed using a melting process to obtain the daily sales quantity at item level. As a result, the size of the data frame increased significantly to 2.8GB, with the number of records reaching 46.98 million. The columns in the training data were reduced to include only the necessary ones. Since the **dept_id**, **cat_id** and **state_id** could be inferred from the **store_id** and **item_id** by analyzing the patterns, these were not included in the dataset.

The training data was then merged with calendar data, followed by the sales data. This merge was aimed to obtain information about the unit selling price, denoted as **sell_price**, for specific dates. However, this process introduced **NaN** values into the dataframe.

From the merged dataframe, the target value “total sales” was calculated by multiplying the **sales_quantity** with the **sell_price**. Subsequently the final table was created with the columns **store_id**, **item_id**, **date**, and **total sales**.

Data	Shape	Data type	Description
Merged Data	Rows: 46985090 Column: 4	Categorical: item_id, store_id and date Numerical: total sales	This data provided compact information item wise total sales in a particular store at a given time.

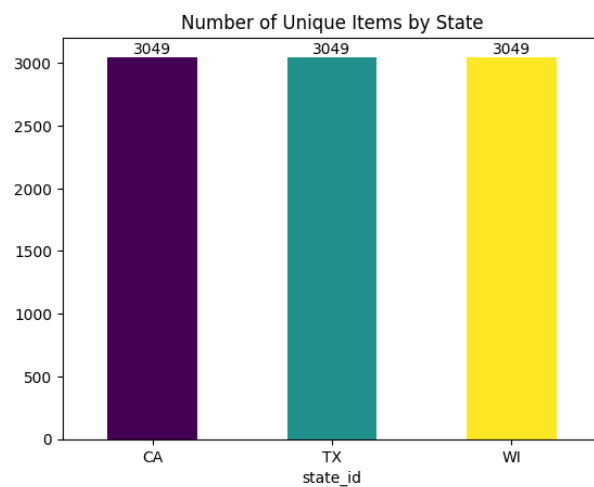
	store_id	item_id	date	total_sales
0	CA_1	HOBBIES_1_001	2011-01-29	0.0
1	CA_1	HOBBIES_1_002	2011-01-29	0.0
2	CA_1	HOBBIES_1_003	2011-01-29	0.0
3	CA_1	HOBBIES_1_004	2011-01-29	0.0
4	CA_1	HOBBIES_1_005	2011-01-29	0.0

Figure 1 Final table

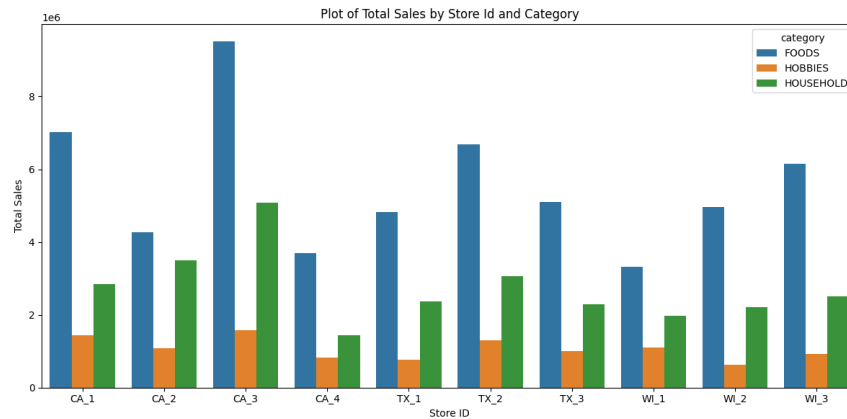
As previously mentioned, the **store_id** is a combination of the state and store number, while the **item_id** is a combination of category, department, and item number. These features were extracted and further explored to analyze the data, uncovering the patterns and relationships between different store and item characteristics.



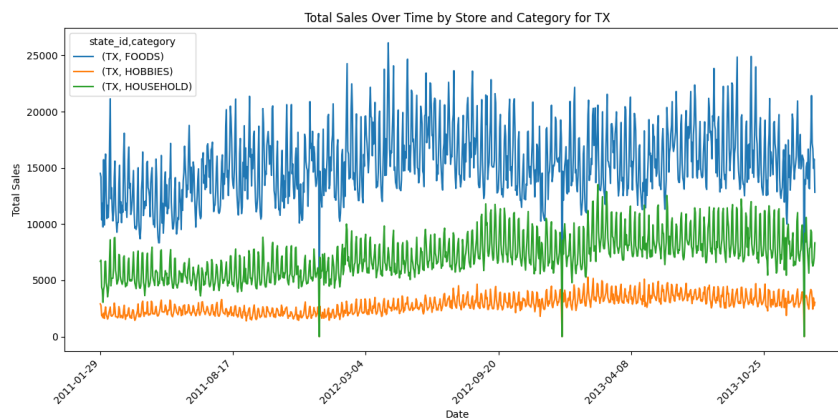
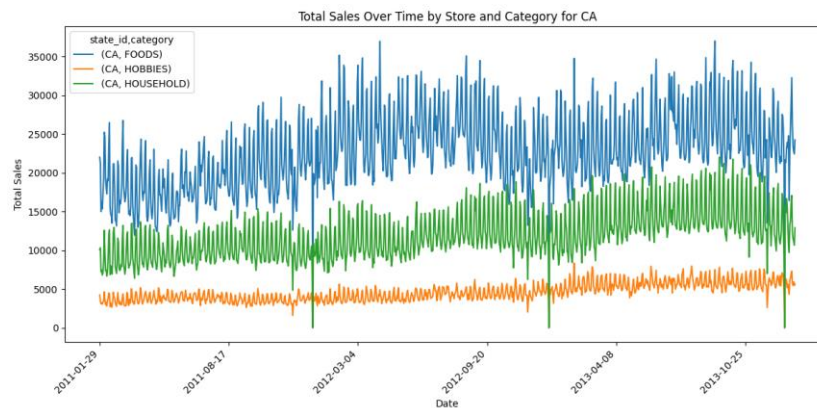
As from the bar graph, it can be observed that CA state has the highest number of stores with 4 whereas TX and WI have 3 each.

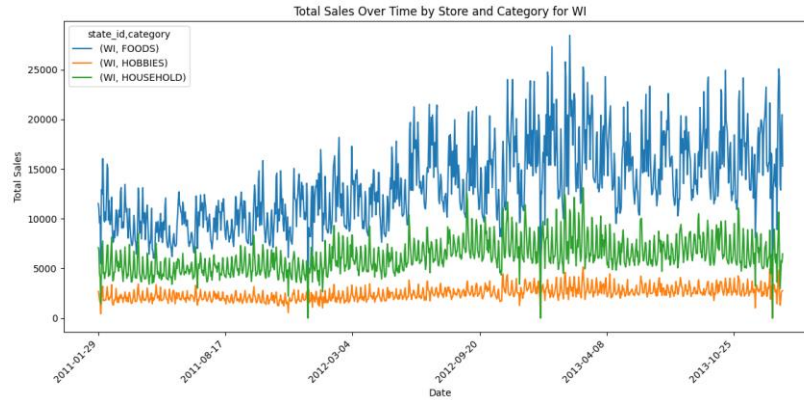


Also, it was evident that the number of items in all the stores was the same.



Similarly, an analysis of the different item categories revealed that food items were the most sold, followed by household products and then hobbies. Across all categories, the highest sales were observed in the CA_3 store.





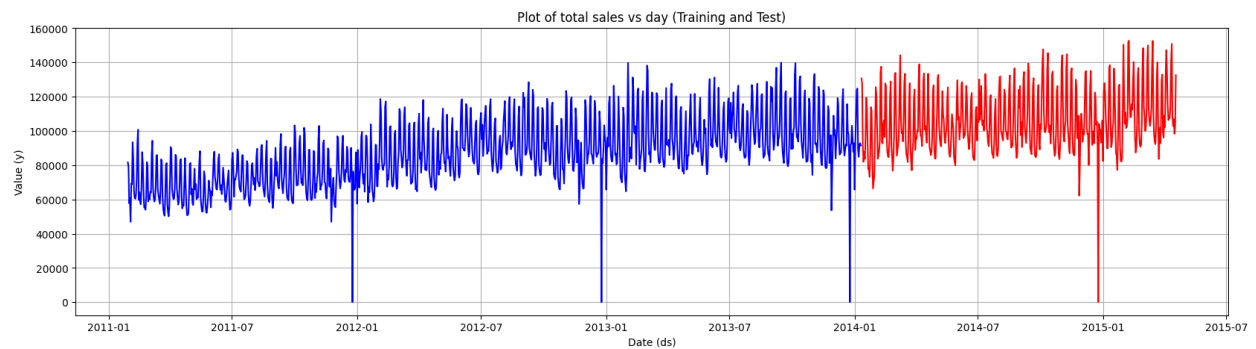
Observing the total sales across categories over time, the food category made a significant contribution to total revenue generation. A notable pattern is the sharp drop in sales toward the end of each year, specifically in December. This decline aligns with the Christmas season, a major holiday celebrated in late December.

Phase	Action	Details
Dropping Features	Dropped Unnecessary Columns: dept_id, cat_id, state_id, wm_yr_wk, d, sales_quantity, and sell_price	<ul style="list-style-type: none"> dept_id, cat_id and state_id were removed to memory usagem as they could be derived from item_id and store_id. This step helped retain only essential information. wm_yr_wk, d, sales_quantity, and sell_price were dropped as they acted as identifiers, which could lead to overfitting by hindering the model's learning capacity.
Feature Extraction	Selected key columns and derived new features	<ul style="list-style-type: none"> Selected essential columns: store_id, item_id, date, and total_sales. Extracted additional information: store_id contained state and store number. item_id contained category, department, and item number. Derived year, month, day, and weekday from the date column.
Feature Encoding	One hot encoding for state and category feature.	<ul style="list-style-type: none"> Applied one-hot encoding to state and category, which each had three distinct values, using a custom transformer pipeline. This enabled the model to handle categorical data efficiently without introducing bias form identifier like features.
Target Generation	Multiplying the unit selling price with the quantity sold.	<ul style="list-style-type: none"> The sales quantity and sell price were multiplied to obtain the item level sales for each store on a given date which is the target for prediction task.

II) Total Sales Forecast

For this task, the final table obtained from the prediction was used. The total revenue for each date was summed to get the day wise sales of the store.

Phase	Action	Details
Dropping Features	Dropped Columns: store, state, category, department, item	<ul style="list-style-type: none">Since data preparation is for total sales forecast for upcoming days, these features would be irrelevant for chosen model.
Feature Merging	Year, Month, and Day were merged.	<ul style="list-style-type: none">The year, month and day were merged to get the date for each entry.This was later converted to datetime format.
Target Generation	Total Sales of each date is accumulated.	<ul style="list-style-type: none">The total sales for each day are calculated by summing the sales for all items on the specific date.



4. Modeling

The modeling phase was divided into two parts for two specific use cases.

I) Item Level sales Prediction

To predict the sales of an item in a specific store on a given date, the problem was framed as a regression task. A series of experiments were conducted, beginning with Linear Regression as the baseline model, followed by LightGBM and then a Decision tree model. Finally, the XGBoost model was employed to create a lightweight model. The parameters of both the Decision Tree and XGBoost Regressor were fine-tuned to optimize performance and identify the best model.

Experiment 1

Model	Hyperparameters	Rationale
Linear Regression (Baseline Model)	Default	<ul style="list-style-type: none">Linear Regression was chosen for baseline model development due to its speed and efficiency, making it an ideal choice for initial testing and evaluation.Training a linear regression model is straightforward, allowing for quick iteration and analysis during the early stages of model development.This model allowed for a benchmark against which more complex models could be evaluated.

Experiment 2

Model	Hyperparameters	Rationale
Light GBM	Default	<ul style="list-style-type: none">Given the large size of the dataset, Light GBM was chosen because it efficiently handles large-scale data using a histogram-based algorithm, which reduces memory usage and accelerates computation compared to other gradient boosting methods.With an aim to improve the prediction of the item level sales, this model was trained and evaluated.

Experiment 3

Model	Hyperparameters	Rationale
Decision Tree	i) Default ii) Params: min_sample_split: range (3,70), min_sample_leaf:[1, 2, 4, 6], max_depth:range(15,100)	<ul style="list-style-type: none"> The previous model performed poorly, exhibiting significant underfitting on the dataset. Given the hierarchical nature of the data, a decision tree model was chosen as it is well-suited to handle datasets with tree-like structures. Decision trees can efficiently split the data into distinct segments based on feature values, capturing important interactions and patterns. It is also relatively easy to train and allows for straightforward hyperparameter tuning, making them a practical choice for this problem. Initially, the model was trained with the default parameters to intentionally overfit the data and establish a baseline. Afterwards, the model was fine-tuned to achieve the best possible predictions. Fine tuning was conducted using RandomSearchCV with 5-fold cross validation, allowing for an efficient exploration of the hyperparameter space. The best performing parameters were then selected to make predictions on the test data, leading to improved model performance. Best Params: {'min_samples_split': 42, 'min_samples_leaf':6, 'max_depth':41}

Experiment 4

Model	Hyperparameters	Rationale
XGBoost Regressor	Default	<ul style="list-style-type: none"> This model was chosen for its lightweight model structure that is efficient and scalable with its optimized distributed gradient boosting techniques. It is particularly effective for structured data, which is typical in retail sales prediction. The main objective was to overfit the model and then fine tune it later.
	Params: Max Depth: range (3,15,1) Min child weight: range (3, 15,1), gamma: [0,0.5,0.05]	<ul style="list-style-type: none"> The model was fine-tuned using the Hyperopt Library, with the search space focusing on key hyperparameters such as max_depth, min_child_weight and gamma An objective function was defined to minimize the RMSE score, and 5-fold cross-validation was applied to ensure robust evaluation and selection of the optimal hyperparameter set. Best Params: {'gamma':0.05, 'max_depth':13, 'min_child_weight':9}



II) Total Sales Forecast

To forecast total sales, a null regressor model was first developed to predict mean sales. Subsequently, the Prophet model was selected, initially utilizing default parameters. To enhance its performance and better capture trends, holiday effects and seasonal patterns were incorporated into the model. This iterative approach aimed to improve the model performance and ensure that the forecasts reflect both regular and exceptional sales fluctuations.

Experiment 1

Model	Hyperparameters	Rationale
Null Regressor	Default	<ul style="list-style-type: none">Linear Regression was chosen for baseline model development as it is fast and efficient model for baseline model development and testingIt is easier to train linear regression model.

Experiment 2

Model	Hyperparameters	Rationale
Prophet (train data < 20115-04-12, test data >= 20115-04-12)	Default	Cross validation: initial='730days', period='7days', horizon='7days' <ul style="list-style-type: none">The default model was trained to overfit the model and later finetune the model.
	Holidays	<ul style="list-style-type: none">With the same procedure, the prophet model was trained again with the explicit information of the data.The performance of the model improved and was able to capture the sharp drop in the total sales, however model had performed poorly on the 5th day of the test data when evaluated using Cross validation initial='730days', period='7days', horizon='7days'.
	Seasonality	<ul style="list-style-type: none">The main objective was to include the fluctuation during the seasons. This seasonality was added with the provided hyperparameters.

	Yearly, and weekly seasonality with additional seasonality of name='monthly', period=30.5, fourier_order=5	<ul style="list-style-type: none"> However, the model could not perform better.
	Seasonality and Holidays Holidays, Yearly, and weekly seasonality with additional seasonality of name='quarterly', period=30.5, fourier_order=5	<ul style="list-style-type: none"> Same as previous, but this time holiday was also included to train on the total sales data. The model performance was outstanding on test and cross validation data. This model was then chosen for forecasting.

5. Evaluation

a. Results and Analysis

l) Item level sales prediction

The models were primarily evaluated using the R2 score, as it better reflects the model's ability to capture the variance in the data and assess its learning capacity. While the RMSE and MAE scores were lower, these metrics alone could not provide complete understanding of the model's effectiveness in learning and prediction of the item level sales variations. Additionally, the model selection process took into consideration the resources constraints, ensuring that the chosen model could be efficiently deployed within the available computational limits.

Model	Performance	Insights
Linear Regression	Train Set RMSE: 8.9 MAE: 4.12 R2 Score: 0.01 Test Set: RMSE: 9.55	<ul style="list-style-type: none"> The model failed to learn effectively, performing poorly on the test data with an R2 score near zero. This indicates that the model was unable to accurately predict item sales for the specified time and store location.

	MAE: 4.77 R2 Score: 0.01	
Light GBM	Train Set RMSE: 8.59 MAE: 3.97 R2 Score: 0.08 Test Set: RMSE: 9.2 MAE: 4.31 R2 Score: 0.065	<ul style="list-style-type: none"> Similarly, this model was evaluated, and while its performance showed a slight improvement, the R2 score remained below 0, indicating that the model failed to explain any meaningful variance in the data. Despite the lower RMSE and MAE scores, it suggested the pseudo reduction of prediction error, it was not able to learn from the data.
Decision Tree { min_sample_split:42, min_sample_leaf:6, max_depth:41 }	Train Set RMSE: 5.03 MAE: 2.006 R2 Score: 0.68 Test Set: RMSE: 5.61 MAE: 2.26 R2 Score: 0.61	<ul style="list-style-type: none"> The R2 score, which measures the proportion of variance explained by the model, improved significantly after tuning, exceeding 0.6 for both the training and test datasets. This indicates that the model was able to capture a reasonable amount of variability in the data that indicates 61% of sales variation could be predicted. As a result, the model demonstrated strong performance in predicting item-level sales, providing reliable prediction. Although this model performed well, the size of the pickle file exceeded 1.6+ GB, creating challenge during deployment due to resource limitations. As a result, this model was set aside as a backup option to be deployed once sufficient computational resource become available.
XGBoost {'gamma':0.05, 'max_depth':13, 'min_child_weight':9}	Train Set RMSE: 7.54 MAE: 3.14 R2 Score: 0.29 Test Set: RMSE: 7.83 MAE: 3.49 R2 Score: 0.24	<ul style="list-style-type: none"> The R2 score of this model was higher than that of the baseline model but lower than the previous model, indicating moderate performance in explaining the variance in the data. Although the model underfitted the data, it was chosen for item-level sales prediction due to its lower risk of significant loss, even though it may limit sales prediction by being overly conservative. The primary reason for selecting this model was the resource limitations during deployment, as it was more lightweight and feasible to implement given the available computational resources.

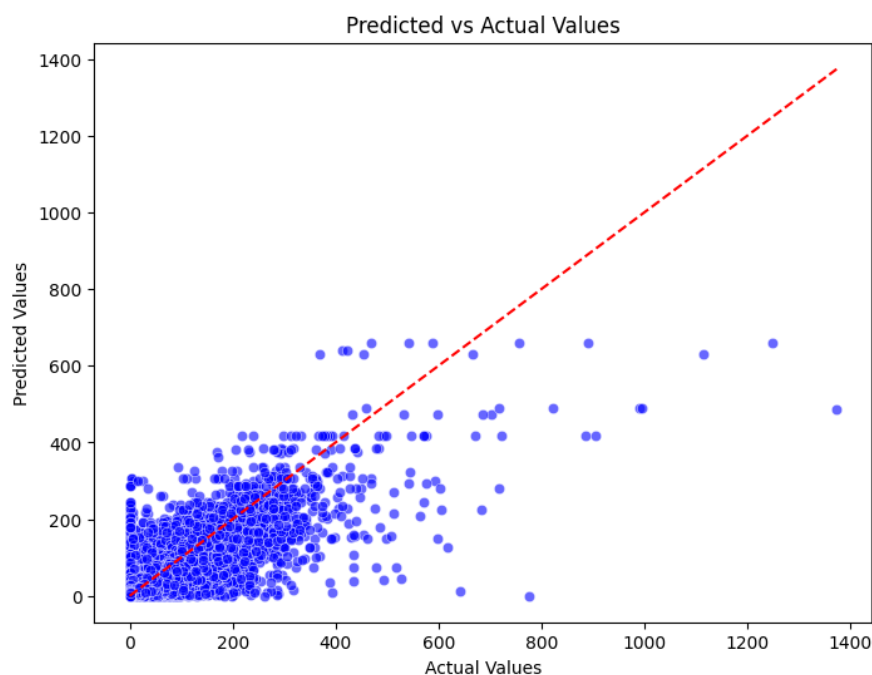


Figure 2 Predicted vs Actual Graph of Best Performed model Decision Tree

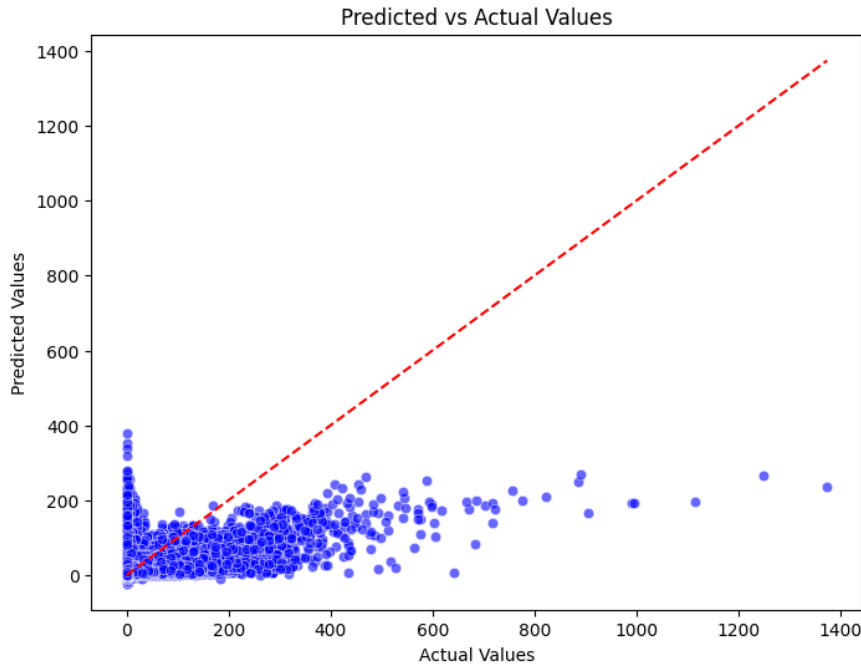


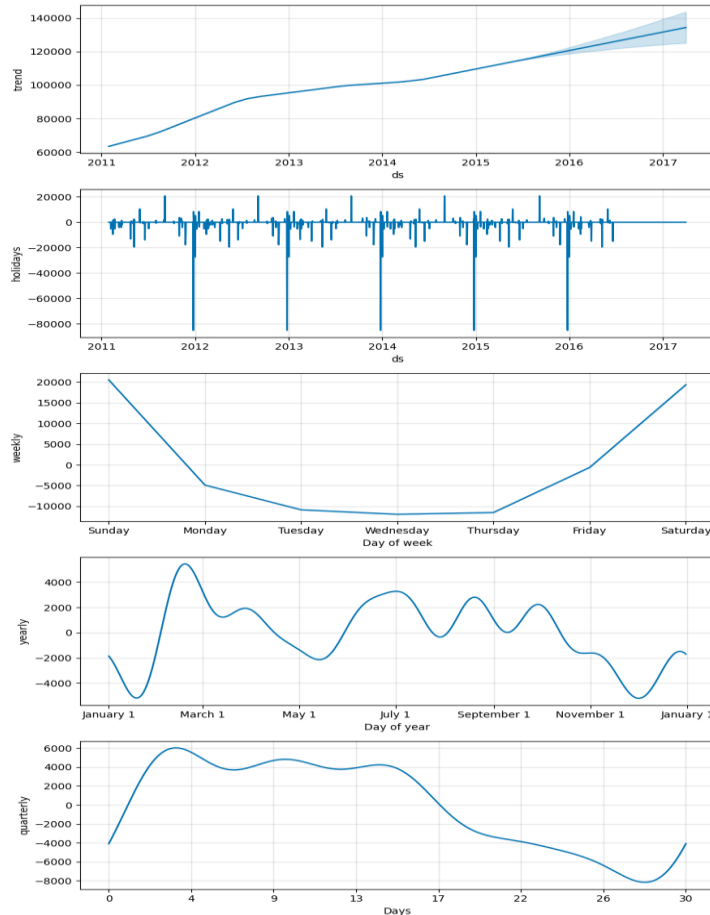
Figure 3 Prediction vs Actual Graph of Moderate Performance XGBoost Deployed.

II) Total Sales Forecast

Model selection and evaluation were based on the performance metrics R2 score and MAPE (Mean Absolute Percentage Error). Although RMSE and MAE were also considered, they were not prioritized. The MAPE score was calculated after performing cross-validation, using an initial period of 730 days, with a rolling window of 7 days and a forecasting horizon of 7 days.

Model	Performance	Cross Validation	insights
Null Regressor (train data < 20115-04-12, test data >= 20115-04-12)	Train Set RMSE: 19157.7 MAE: 15062.69 R2 Score: 0.0 Test Set: RMSE: 26544.85 MAE: 20590.56 R2 Score: -1.078		The R2 score obtained was negative in the test data indicating the model is unable to learn with RMSE score greater than the train set.
Prophet Default (train data < 20115-04-12, test data >= 20115-04-12)	Train Set RMSE: 10319.43 MAE: 7748.2 R2 Score: 0.68 Test Set: RMSE: 10319.43 MAE: 7748.26 R2 Score: 0.68	Cross validation: initial='730days', period='7days', horizon='7days' MAPE score: < 0.09 for all days expect 5 th day > 52.	<ul style="list-style-type: none"> The model was able to learn and performed well on test data with R2 score far better than Baseline model. The RMSE and MAE score also dropped has comparatively similar performance for train and test set. However, the MAPE score after cross-validation was observed very high for 5th day.
Prophet with Holidays	Test Set: RMSE:8974.66 MAE: 7201.46	Cross validation:	<ul style="list-style-type: none"> The model performed better with the inclusion of holidays information.

<p>(train data < 20115-04-12, test data >= 20115-04-12)</p>	<p>R2 Score: 0.76</p>	<p>initial='730days', period='7days', horizon='7days'</p> <p>MAPE score: < 0.09 for all days except 5th day > 52.</p>	<ul style="list-style-type: none"> The R2 score increased to 0.76 and the RMSE and MAE score dropped slightly as compared to previous model. However, after the cross validation the MAPE score was same as before.
<p>Prophet with Seasonality</p> <p>(train data < 20115-04-12, test data >= 20115-04-12)</p> <p>Seasonality:</p> <p>Yearly, and weekly seasonality with additional seasonality of name='monthly', period=30.5, fourier_order=5</p>	<p>Test Set: RMSE:17581.59 MAE: 15075.47 R2 Score: 0.088</p>	<p>Cross validation:</p> <p>initial='760days', period='7days', horizon='7days'</p> <p>MAPE score: < 0.17 for all days except 5th day > 59.</p>	<ul style="list-style-type: none"> The model was provided information to track the seasonality, but it dropped the performance resulting in very low R2 score and conversely very high RMSE and MAE score. MAPE score also increased for everyday forecast. Although the mode performed poor it provided information about the inclusion of seasonality could capture seasonal fluctuations.
<p>Prophet with Seasonality and Holidays</p> <p>(train data < 20115-04-12, test data >= 20115-04-12)</p> <p>Holidays, Yearly, and weekly seasonality with additional seasonality of name='quarterly', period=30.5, fourier_order=5</p>	<p>Test Set: RMSE:7958.24 MAE: 6155.95 R2 Score: 0.81</p>	<p>Cross validation:</p> <p>initial='730days', period='7days', horizon='7days'</p> <p>MAPE score: < 0.05 for all days except 5th day < 0.26.</p>	<ul style="list-style-type: none"> The model provided information of seasonality and holiday and the R2 drastically increased to 0.81 improving the model's capability to capture seasonal and holiday fluctuation in sales. MAPE score dropped below 0.05 for other days and for 5th day the score was below 0.26 which is an improvement.
<p>Prophet with Seasonality and Holidays</p> <p>(all data except upcoming 7 days)</p> <p>Holidays, Yearly, and weekly seasonality with additional seasonality of name='quarterly', period=30.5, fourier_order=5</p>	<p>Test Set: RMSE:6918.26 MAE: 5564.10 R2 Score: 0.85</p>	<p>Cross validation:</p> <p>initial='730days', period='7days', horizon='7days'</p> <p>MAPE score: < 0.0752 for all days along with 5th day < 0.05.</p>	<ul style="list-style-type: none"> The model was trained on whole data and the forecast was performed for the upcoming 7 days. R2 score improved with the inclusion of more data where 85% of the sales variation could be forecasted with error of approx. 7k as indicated by RMSE. MAPE score after cross validation, was below 0.05 for 5th day and 0.07 for rest of the days. This model was chosen as the final model for the application development purpose.



Testing the final model on the forecast for the next two years revealed that the confidence intervals of the trend forecast began to widen starting in 2016. While the forecasts were reliable up until mid-2016, the model struggled to capture the periodic trends thereafter. This is evident in late 2016, where the expected sharp drop in total sales due to the Christmas season was missed by the model.

b. Business Impact and Benefits and Risks

I) Item level Sales Prediction

The final model selected for deployment was XGBoost, despite its lower performance compared to the Decision Tree model. This decision was made due to resource constraints in the deployment environment. The available system offered only 512MB of memory, while the Decision Tree model exceeded 1.5GB, surpassing the memory limit and forcing the choice of a more lightweight model like XGBoost to fit within the resource limitations.



The model has several business implications and benefits across different stake holders

1. Retail Stores

While the model can predict item-level sales for individual stores, it tends to underestimate sales compared to actual figures. Although this conservative prediction minimizes the risk of overstocking or losses, it may also limit the retail store's ability to fully capitalize on sales opportunities, potentially underestimating the demand and leading to missed sales.

2. Marketing and Sales Team

The model's lower than expected sales predictions could give the marketing and sales teams an inaccurate view of demand, potentially causing them to avoid or delay promotional activities. This conservative forecast might lead to missed opportunities for driving higher sales through marketing campaigns or pricing strategies.

3. Financial and Executive Teams

For financial planning and executive decision-making, the model's prediction provides a cautious outlook, which can be useful for conservative budgeting and risk mitigation. However, the underestimation of sales may result in overly conservative financial projections, potentially affecting investment decisions, inventory planning, and broader business strategies

Despite its limitations, the XGBoost model offers a balance between performance and resource efficiency, making it suitable for environments with constrained computational resources while still delivering useful insights into item-level sales trends.


II) Total Forecast

The Prophet model, incorporating seasonality and holiday effects, was chosen for forecasting total sales due to its ability to handle complex trends and periodic fluctuations.

1. Retail Stores

The Prophet model provides valuable insights into long-term sales trends and seasonal patterns, helping retail stores plan inventory levels more effectively. By forecasting expected demand during peak seasons or holidays, stores can optimize stock levels to avoid shortages or overstocking, ensuring a smoother operational flow.

2. Marketing and Sales Teams



The sales forecasts generated by the Prophet model enable marketing and sales teams to time promotions and campaigns more strategically. With accurate insights into when demand is likely to rise or fall, they can align promotional efforts with key sales periods, such as holidays, boosting overall revenue by capitalizing on high-demand times.

3. Financial and Executive Teams

For financial planning and executive decision-making, the Prophet model offers a reliable forecast of total sales, considering both seasonality and holiday effects. This helps the executive team make informed decisions around budgeting, resource allocation, and growth strategies. By identifying predictable sales fluctuations, financial teams can better plan for cash flow, ensuring that the business remains resilient during low-demand periods while maximizing opportunities during high-demand times.

c. Data Privacy and Ethical Concerns

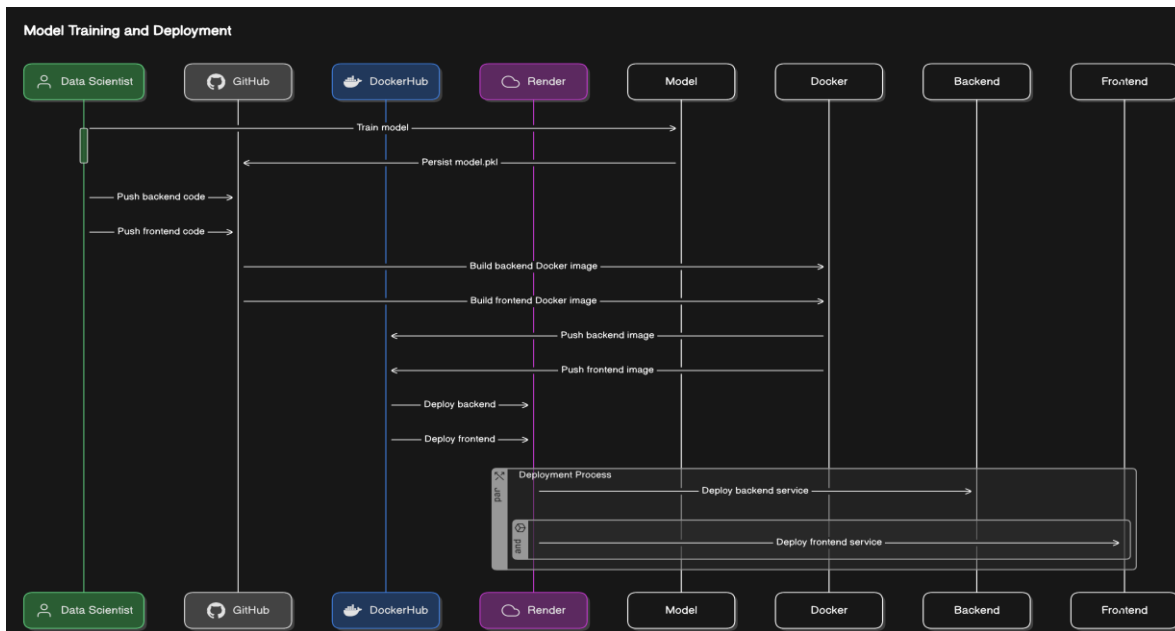
The project was conducted using a dataset provided by an undisclosed American retail store. To maintain data privacy, the retailer's identity was anonymized, and store information was presented using coded labels such as "CA_1". Similarly, items were categorized by general categories, with item names encoded. No specific details about retailer, stores, or individual items were disclosed. This encoding ensured the protection of sensitive information, such as most daily purchased items or the store's daily sales of the specific item. Additionally, using these encoded features helped to prevent bias or unfairness in the decision-making process.

However, the data collection method is unclear, raising ethical concerns about whether the data was collected legally and with proper consent. For data collection to be ethical, it must be fair, unbiased, and involve the explicit consent of the retail store and all relevant stakeholders, including indigenous peoples. Ensuring transparency and inclusion in the data collection process helps prevent bias or discrimination against any specific groups, promoting fairness in the analysis and decision making.



6. Deployment

After achieving satisfactory results, an application was developed to serve the machine learning model as a service, enabling its use in real-time applications. The application was deployed on the Render using the docker image from the docker hub. The whole Training and deployment process is explained in the graph below.



Project	Github URL	Render URL	Details
AT2-ADML-24996124	shrestha1/AT2-ADML-24996124 (github.com)		It contains all the experiments for developing the model using
AT2-ADML-24996124-API	shrestha1/AT2-ADML-24996124-API: Machine Learning As a Service (github.com)	Application: Frontend-App EndPoint: Endpoint-URL	<p>This repository includes the application code for frontend and backend.</p> <p>The developed models are used in the backend providing service as an endpoint.</p> <p>In the frontend the application can be used by the retail store to get the item level prediction and</p>

Challenges:

- Render had limited resources, which constrained the selection of the optimal model. This restriction impacted the ability to utilize more complex and potential high-performing models.

7. Conclusion

The project offered a valuable opportunity to engage with real world data through time series analysis, prediction and forecasting. It highlighted the importance of efficiently managing large datasets while employing the CRISP-DM methodology, guiding the process from data preparation to modeling. The careful training and fine-tuning of models was crucial, especially given the substantial volume of data involved. The models were primarily evaluated using R2 score, with less emphasis on RMSE or MAE.

Overall, the project successfully navigated the challenges posed by large datasets, delivering satisfactory performance in item-level sales prediction and forecasting, which has been deployed. It has met its primary objectives, providing stakeholders with actionable insights.

Future work

1. Model update: The Decision Tree model demonstrated optimal performance, should be considered for deployment in the future as it is not deployed due to resource limitations.
2. Resource Upgrade: Enhancing computational resources for model development and deployment would enable exploration and testing of more complex models.



8. References

1. 3.2.4.3.6. *sklearn.ensemble.GradientBoostingRegressor* — *scikit-learn 0.21.2 documentation*. (2009). Scikit-Learn.org. <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html>
2. *Quick Start*. (2017). Prophet. https://facebook.github.io/prophet/docs/quick_start.html

■ ■ ■