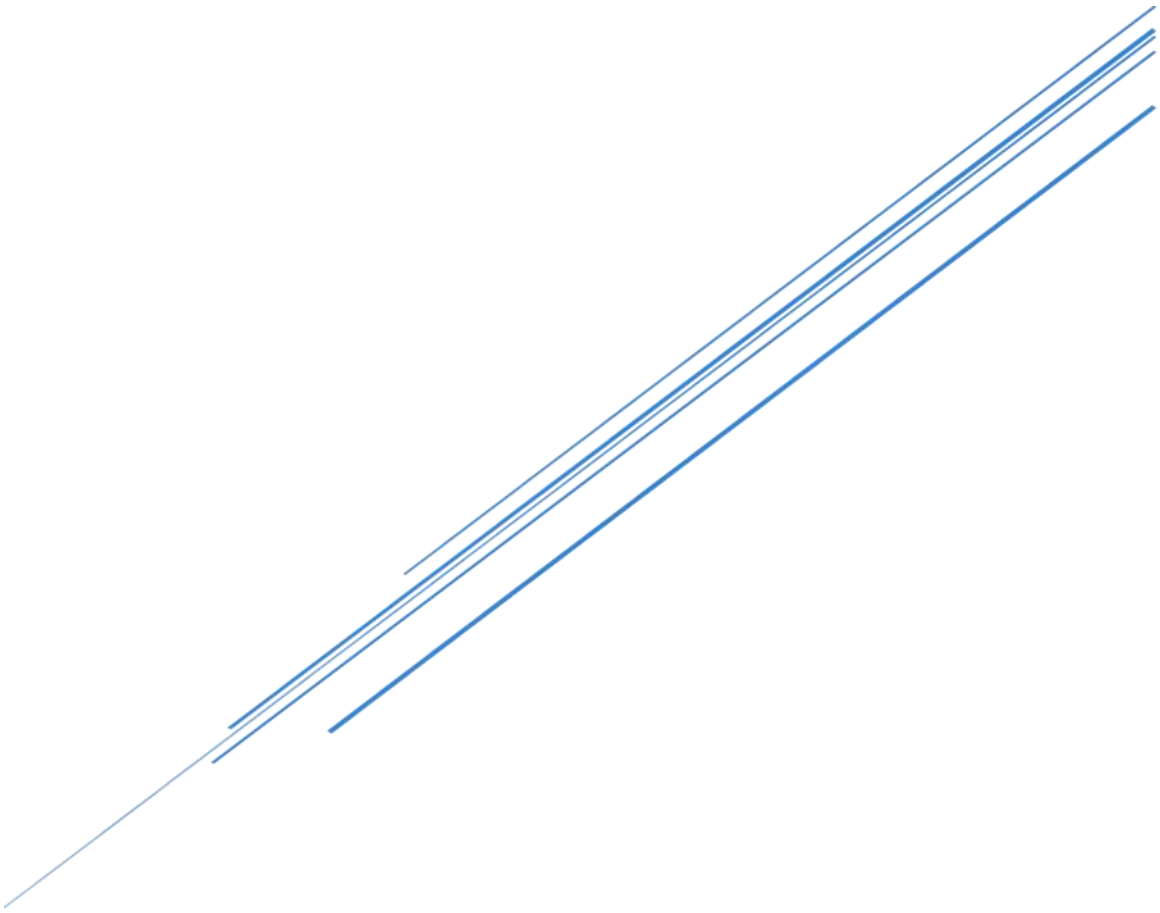
Luna Wear Sales Forecasting Report



BRAND PULSE
-ENGAGEMENT THAT CONVERTS-

BRAND PULSE

Table of Content

Abstract

This project elaborates [procedures on predicting sales data to manage inventory stock for Luna Wear, a clothing brand which is a client of Brand Pulse. To do so we procured historical sales data of over 50 stores and 50 items that were from 2013 to 2017. Given the absence of a definitive test set, the team decided to partition the data in such a way that data from 2013 till 2016 will be used for training the models and 2017 will serve as our test set (ground truth). We conducted extensive analysis on the dataset including exploratory data analysis, data cleaning etc. Some key features like day of the week, weekend or not, holiday, were introduced to enhance model performance. As this would be a prototype serving as a proof of concept, we trained the models on a smaller set of 1 store and 10 items. This also saved us computational cost and time. Our best model turned out to be Gradient Based Regressor which achieved the lost R2 score.

Introduction

Sales Prediction and Inventory Management plays a crucial role in cutting down substantial costs for a business operations to run profoundly. This also aids in Inventory and Stock management for variating sales trends throughout the period (year). The objective of this project is to develop a model capable enough to forecast the sales of an item based on the patterns learned from historical data, going under the preface that the trends did not variate throughout the years. The dataset comprises sales records from 50 stores across 50 products for the years 2013 to 2017. Due to the absence of a designated test set, we structured our approach by training the model on data from 2013 to 2016 and validating it against actual sales data from 2017.

The initial phase of the project involved understanding the data we were dealing with which is explained under Exploratory Data Analysis. To increase model efficiency, we altered the data in such a way that new features were created, this provided the model opportunity to learn from more crucial data and its trends. This step introduced key features which defined temporal and categorical trends helping the model to understand the variance better. We conducted Data Cleaning procedures to handle any missing values.

To save computational burden, we created a subset of the train data and trained a bunch of regressors that the team decided could be strong enough the understand the patterns well, and out of the scores they generated we assessed and chose the best regressor. The best

performing model was “Gradient Boosting Regressor” which was hyperparameter tuned and generated predictions.

This study provides insights into effective sales forecasting methodologies and highlights the importance of feature engineering and model selection in predictive analytics. The findings demonstrate the potential of machine learning techniques in enhancing business decision-making processes in the retail sector.

Methodology

Data exploration

In this section, we basically perform an exploratory analysis initially and preprocessing of the data. It also starts by verifying the dimensions of the dataset using shape attribute, then validates its structure and data types with, A descriptive summary is analyzed too. Next up, we process the "date" column by splitting it up at the hyphen ("-") into individual parts (year, month, day), extracting the year as a new column ("year") right after it gets converted into an integer. The set of the data is then filtered into subsets of two: train, which includes almost all the records where the "year" is not 2017 and belongs to store 1 and item 1, and test, which basically has the records from 2017 for the same store and item. And finally, both datasets of train and test are printed to ensure the process of filtering.

Feature Engineering

Splitting the Date Column into Year, Month, and Day

The initial step basically involves splitting the "date" column, which is sort off assumed to be in the format "YYYY-MM-DD", into three individual columns: "year", "month", and "day". Since raw string operations might lead to a few errors, the values extracted are converted explicitly into numeric format, also making sure that any values which are non-numeric are converted to NaN. To basically maintain quality of the data, any row containing values which

are missing in these new columns are also removed using , forbid issues in downstream computations.

Identifying Weekends and Weekdays

To enhance up the set of data with insights which are temporal, we created a new feature to determine whether a specific date given falls on a weekend (Saturday or Sunday) or a weekday (Monday to Friday).

Identifying Public Holidays in India

Holidays specifically which are public can significantly impact the operations of business, sales and customer footfall. So, the holidays library is integrated into the code to determine whether a specific date is a public holiday in India. Since there is a possibility that the "date" column might contain errors and inconsistencies, then firstly it is converted into a standardised format of datetime.

Identifying the Day of the Week

Talking about another feature which is helpful is added to the set of data is the "weekday" column, which explicitly labels each date specifically with a number from 0 (Monday) to 6 (Sunday). All This information is pretty much useful in a lot of applications, such as prediction of the patterns of sales, as different days of the week consist varying trends.

Dropping the Original Date Column

After fetching all the features which are relevant under the category of time-based, the original "date" column is no longer useful and needed. Also, It is taken out from the set of data.

Exploratory Data Analysis

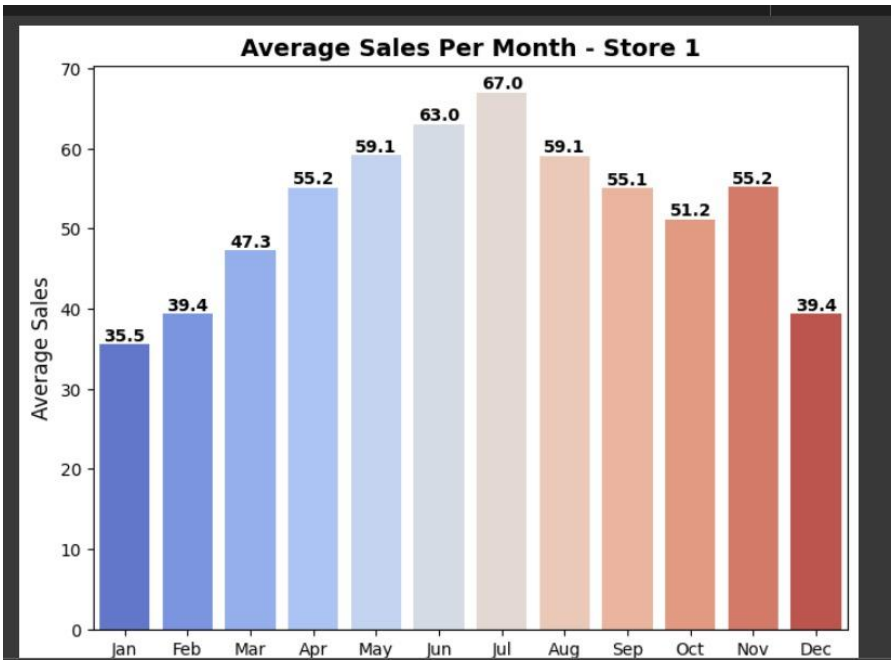


Fig 01: Bar Plot for Average of sales across months.

- This bar chart shows average sales for each month of the year for Store 1, with the highest sales in July (67.0) and the lowest in January (35.5).
- Sales increase from January to July and decrease towards December, indicating seasonal influences on sales.

Monthly Sales Trends (Actual vs Predicted)

Fig 02: Sales in a month (actual vs predicted).

- Shows actual sales consistently outperforming predictions throughout the year
- Peak sales occur in month 7 (July) with a sharp decline afterwards, while the lowest point is in month 12 (December)

Average of sales per day of a week

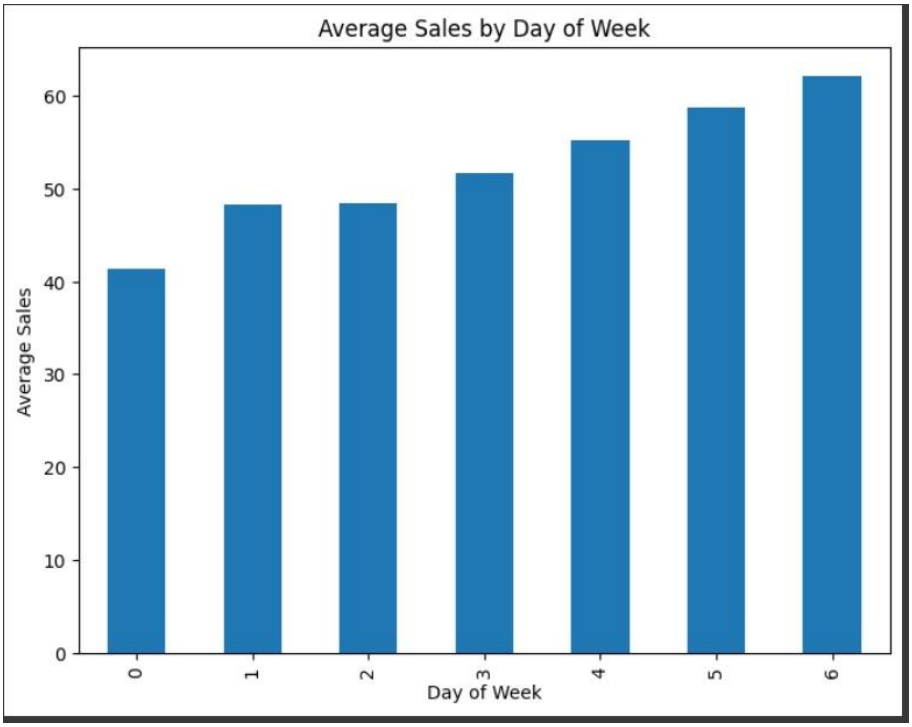


Fig 03: Sales trends per day (week).

- Shows a clear upward trend from day 0 to day 6 of the week
- Weekend sales (days 5-6, likely Saturday-Sunday) are about 50% higher than early week sales (days 0-1)

Actual Vs Predicted Sales (2017) - Scatter Plot

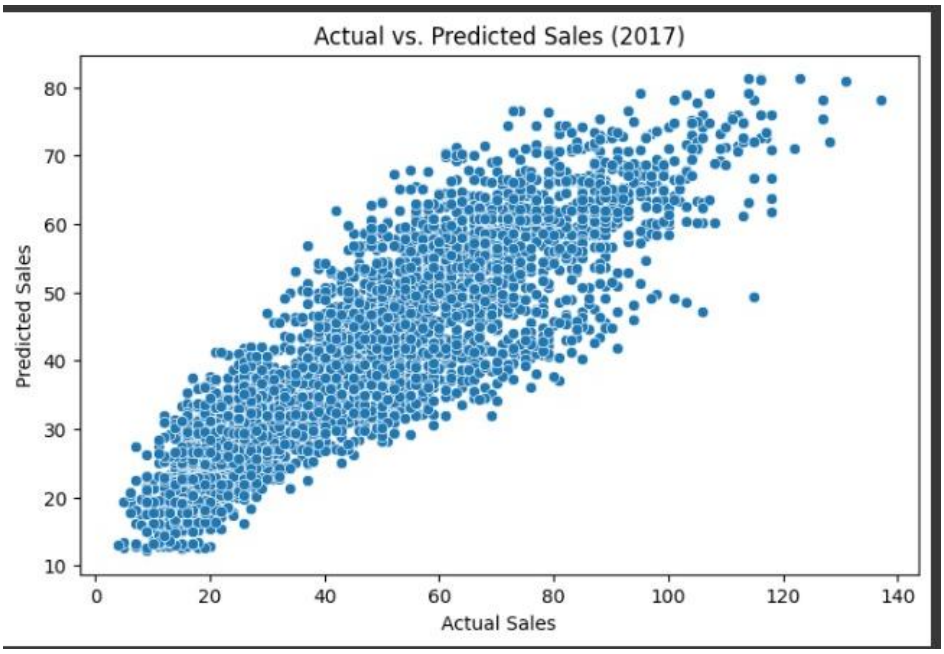


Fig 04: Actual Vs Predicted (Scatter).

- Shows a strong positive correlation between actual and predicted sales
- The spread of points widens at higher sales volumes, indicating less accurate predictions for higher sales numbers

Sales Trend Over Days in January 2017

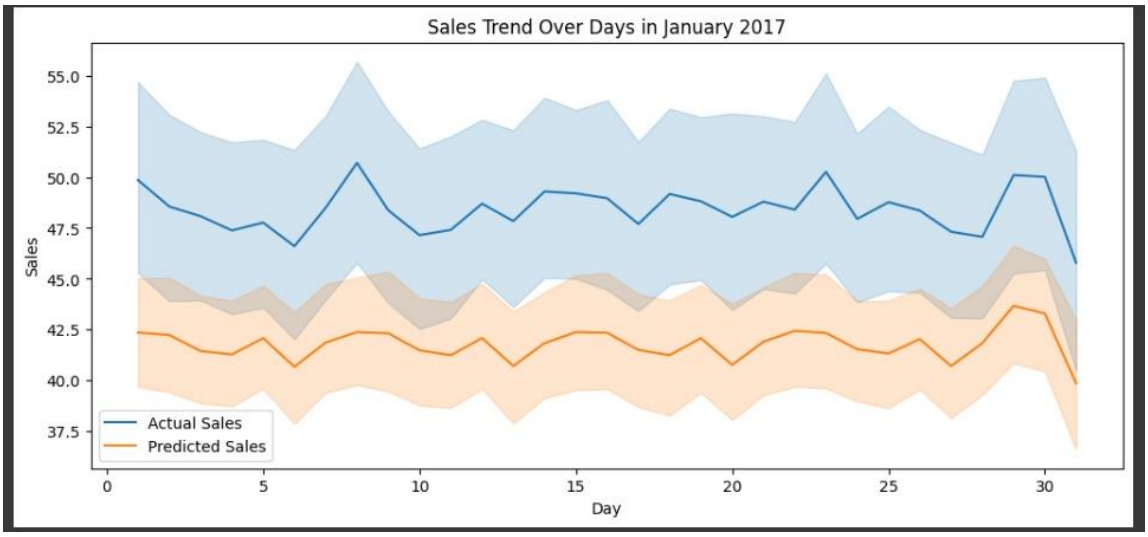


Fig 05: Sales Trends in January 2017.

- Actual vs Predicted Sales: The blue line represents actual sales, while the orange line shows predicted sales over January 2017. The shaded regions around each line indicate variability or confidence intervals in the data.
- Trend Observation: There is a consistent gap between actual and predicted sales, suggesting underprediction in the forecast. Both exhibit daily fluctuations, with peaks and troughs at similar intervals.

Outlier Detection

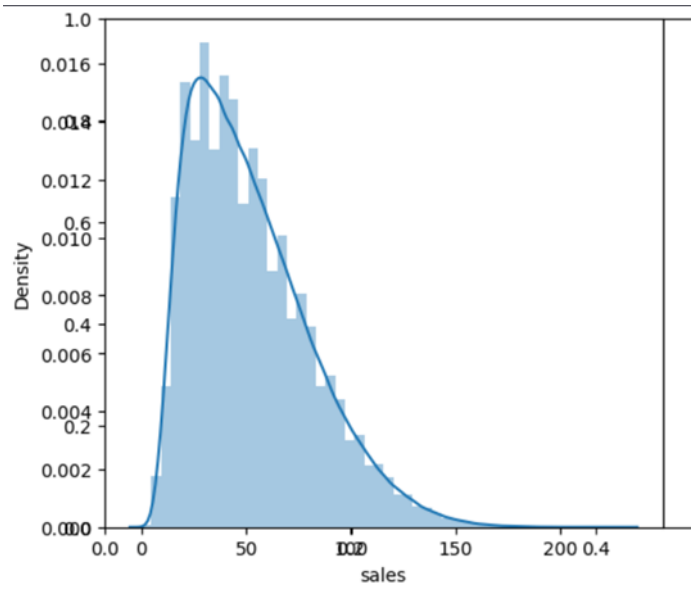


Fig 06: Outlier Analysis using Dist. Plot.

- The distribution plot (a histogram with a density curve) shows that the data of sales is right-skewed, which typically means tah a lot of the values are concentrated into the direction of the lower end, while a few of the values which are higher extend the tail towards the right direction.
- This basically suggests us that there are some values of extreme sales that occur inconsistently, which could be the potential outliers.

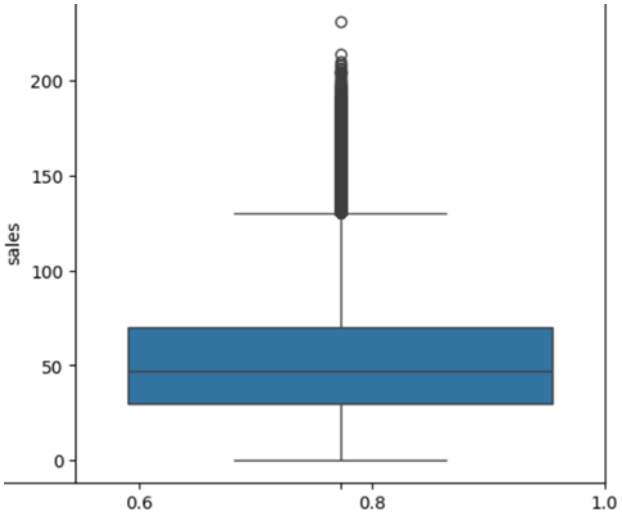


Fig 07: Box plot to explain outliers.

- The box plot shows the distribution of "sales" graphically. The whiskers extend up to 1.5 times the interquartile range (IQR) beyond the first and third quartiles, while the box itself displays the IQR (middle 50% of the data).
- Outliers are numbers that are noticeably greater above the typical range of data, as indicated by the little circles above the upper whisker.

Models Considered

The team extensively researched and concluded on a curated list of models that seemed fit for the task at hand.

Here are the models the team trained with a sample of data:

1. KNeighbors Regressor:

   One non-parametric technique for forecasting continuous values is KNN regression. The main concept is to average the target values of the K nearest neighbors in the feature space to estimate the target value for a new data point. Although different distance metrics can be employed, Euclidean distance is commonly utilized to estimate the distance between data points. (Geeks for Geeks, 2014, para. 04)

2. Gradient Boosting Regressor:

   One kind of ensemble approaches is gradient boosting, which involves building several weak models and combining them to improve overall performance. (Masui, T., 2025, para. 03)

3. Extra Tree Regressor:

Like Random Forest but with more randomization, Extra Trees (Extremely Randomized Trees) is an ensemble machine learning model that mixes many decision trees. Although both make use of several trees, Extra Trees creates splits at random and doesn't look for the best thresholds. It usually utilizes the entire dataset for each tree rather than bootstrap sampling, but it keeps things unpredictable by choosing random split points. (Baladram, S., 2024, para. 04)

4. Random Forest Regressor:

An ensemble learning approach is used for regression in machine learning via the supervised learning algorithm and bagging technique known as random forest regression. In random forests, there is no contact between the trees throughout the tree-building process since they operate in parallel.
(Chakure, A., 2023, para. 02)

5. Decision Tree Regressor:

Decision Tree Regression predicts continuous values. It does this by splitting the data into smaller subsets based on decision rules derived from the input features. Each split is made to minimize the error in predicting the target variable. At the leaf node of the tree the model predicts a continuous value which is typically the average of the target values in that node. (GeeksforGeeks., 2025, para. 03)

6. Linear Regression:

A model that explains the connection between a dependent variable and one or more independent variables is created using linear regression analysis. Simple and multiple linear regression analysis are distinguished based on the presence or absence of independent variables. (Test. T, n.d., para. 02)

7. XGB Regressor:

A family of ensemble machine learning methods known as gradient boosting can be applied to issues involving regression predictive modeling or classification. Decision tree models are used to build ensembles. To fix the forecast errors caused by earlier models, trees are added to the ensemble one at a time and fitted. Boosting is the name given to this kind of ensemble machine learning model. The gradient descent optimization technique and any arbitrary differentiable loss function are used to fit the models. The approach is called "gradient boosting" because, like a neural network, it minimizes the loss gradient when the model is fitted. (Machine Learning, n.d., para. 06)

8. Lasso Regressor:

A regularization method called Lasso regression uses a penalty to stop overfitting and improve statistical model accuracy. One type of regularization for

linear regression models is Lasso regression, sometimes referred to as L1 regularization. (IBM. ,2024, para. 01)

9. Ridge Regression:

    There are several forms of regularization for linear regression models, including ridge regression, commonly referred to as L2 regularization. A statistical technique for lowering mistakes brought on by overfitting on training data is regularization. (IBM. ,2024, para. 01)

Data Splitting

The strategy used to split the data was simple yet effective considering the absence of a test set to evaluate the model on. We split the data in such a way that train set would have all the data from years 2013 till 2016 and test set will be data on year 2017.

The train and test split in this was to separate the ‘sales’ - our target feature from the dataset. This would be our y_train and y_test respectively.

Data Transformation

Judging by the data we were dealing with, mostly categorical in nature, the team decided to only scale the data for simplicity as no other transformations seemed necessary.

Workflow

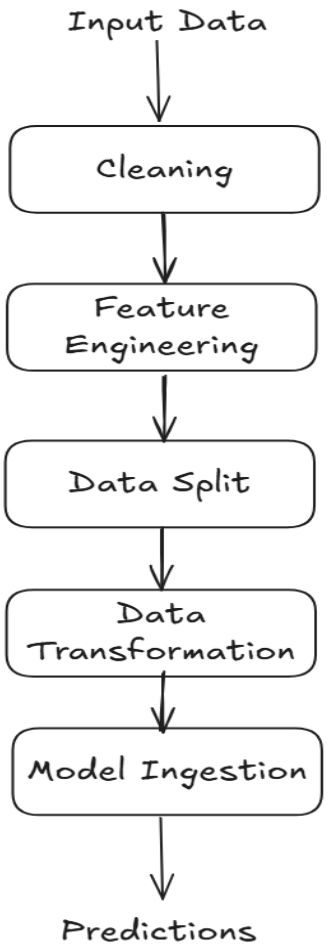


Fig 08: Workflow of Data

The picture above visualizes a detailed explanation of how the data goes through various stages before generating predictions.

At first, the input is handled by adequate checks to understand if the data is complete and accurate. Cleaning procedures are acted upon where missing values are eradicated to ensure accurate model predictions. Then in the feature engineering phase, the data goes through an extensive pipeline where the initial inputs are used to generate new features that can aid to the model in explaining the trends better. Some of the features that are generated are Holidays, Weekend, Day of the Week, Year, Day, Month etc. These features can help the model uncover hidden patterns.

After that the data is split internally to run a train and test cycle to compare its predictions to some truth value. In case of evaluation of model, this stage is skipped as we do no need to have data split internally when in inference mode. After this phase, the newly established data goes through some transformations, particularly scaled to represent the spread of the data but in a limited range for the model to generate computations faster and efficiently.

The model then utilizes the transformed data and generates predictions. The flagship model used in this project is Random Forest Regressor which is Hyperparameter Tuned using GridSearchCV.

Dashboard Visuals

As a product to the client the team worked on developing an extensive dashboard covering various crucial visuals that can explain the data well across the time period.

Find the dashboard at the link here:

https://app.powerbi.com/view?r=eyJrIjoiZmQ0OTE1NjEtMjcwZS00N2FiLTg2NGYtMzBiNWIxYWVlNzA2IiwidCI6ImI2NDE3Y2QwLTFmNzMtNDQ3MS05YTM5LTIwOTUzODIyYTM0YSIsImMiOjN9

Here are some of the visualizations included in the dashboard:

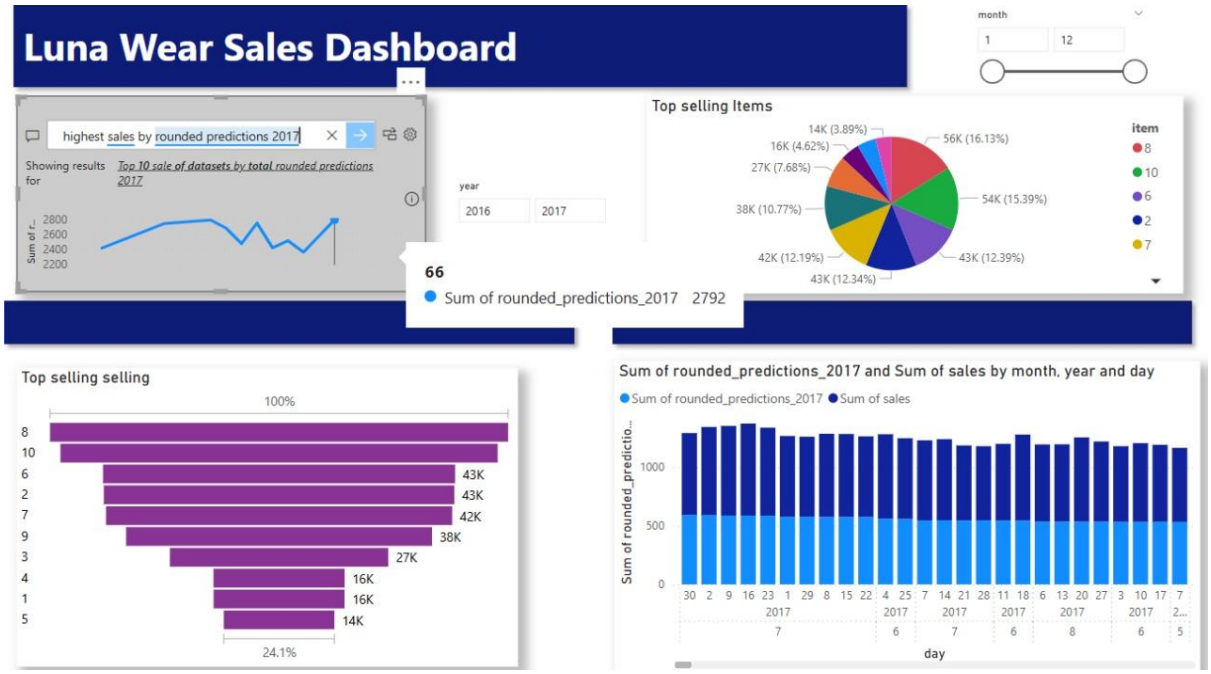


Fig 09: Dashboard visual explaining sales in 2017

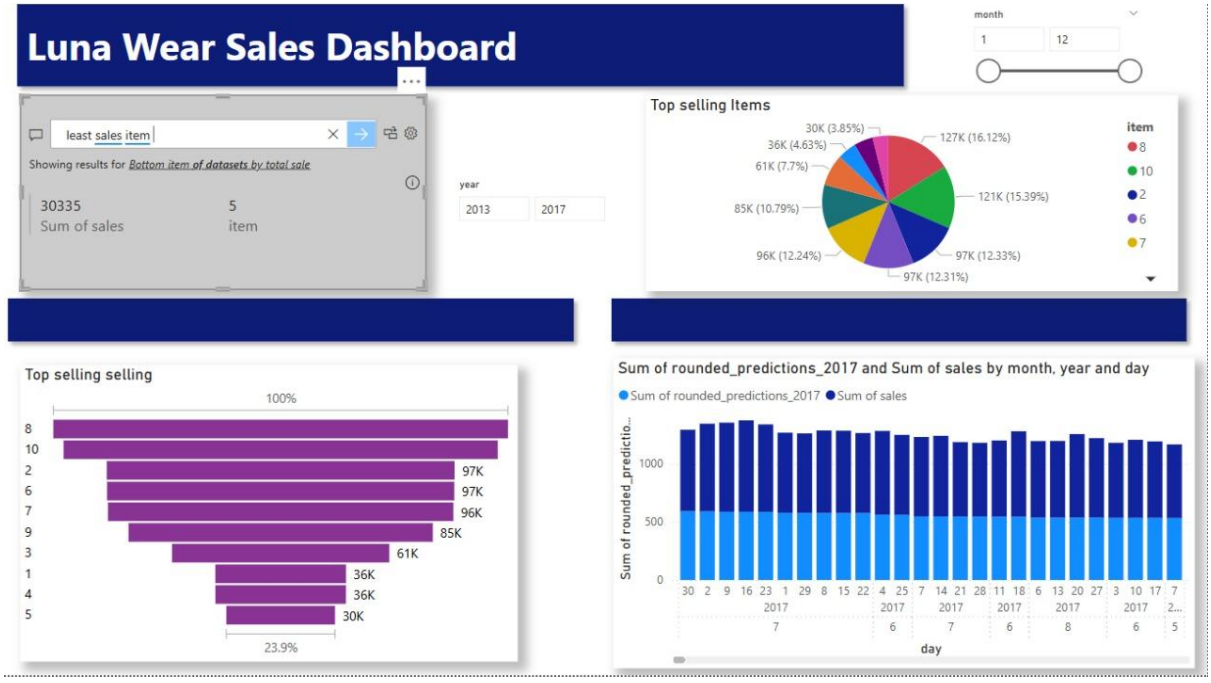


Fig 10: Dashboard Visuals Explaining sales in 2017 and least sold item in the year.

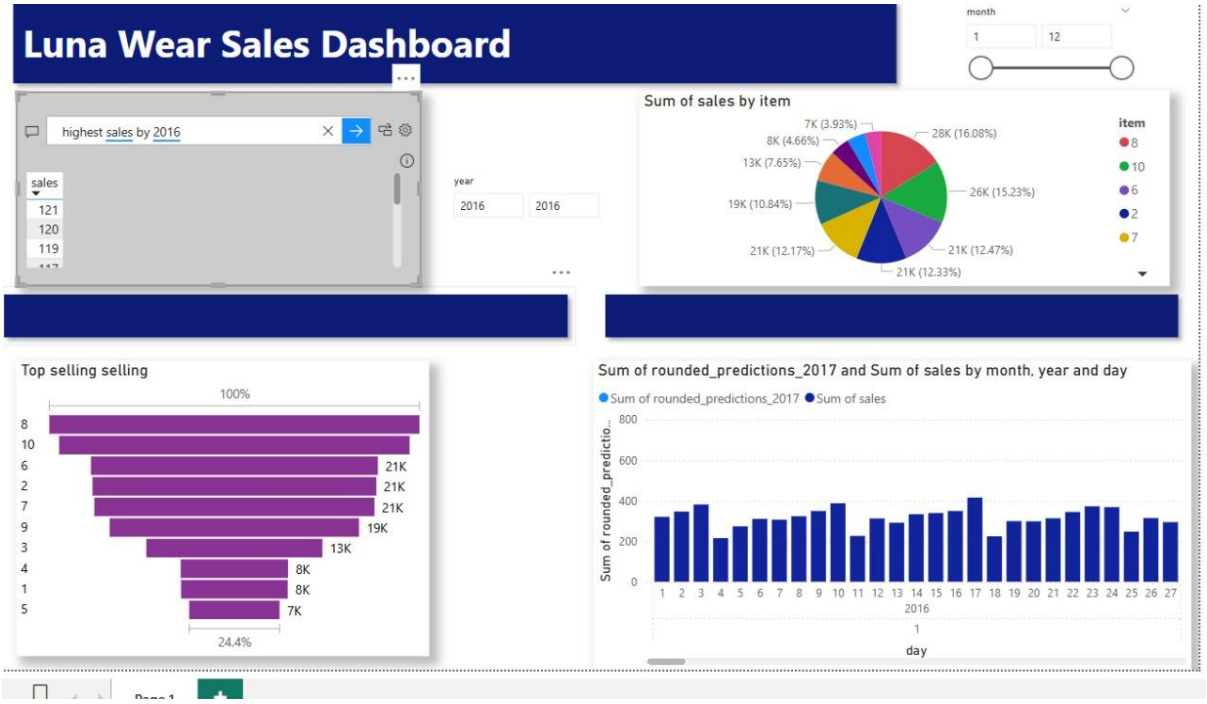


Fig 11: Dashboard Visuals Explaining sales in 2016

Results

After our extensive approach on finding the best model and hyperparameter tuning it. We got the best model that we delivered as the flagship to the customers which is:

Model Name: Random Forest Regressor

Hyper Parameters: learning_rate=0.05, max_features='sqrt'

Performance

The model performed exceptionally well based on the naive environment it was trained on and the strategies implemented to explain the trends. The model achieved an explained variance score of 0.72 which meant the model was confidently and correctly was able to explain 72% of the variance in the data it was trained on. The credit to this goes to the expansion of features in the feature engineering phase.

Although there is always opportunity for improvement, the model's ability to fit the data effectively is highlighted by its R2 score of 0.649. It is fair for predicting sales data that the model's forecasts depart from the actual values by an average of 24.33%, as indicated by the Mean Absolute Percentage Error (MAPE) of 24.33%.

Total training time for this model was around 67 Minutes and hyperparameter tuned for up to 2 hours.

Future Work

As part of the iterative progression of the product the team discussed in introducing an “alerts feature” where in after the generation of the predictions if the inventory data on available stock for each item is provided. The alert system takes in these values and alerts the owner and any other responsible stakeholder of the dip if the stock goes below a threshold set by the organisation.

This can further enhance the stocking strategies of the company and help plan for well ahead in the future.

References

GeeksforGeeks. (2024, June 17). *KNearest Neighbors (KNN) Regression with ScikitLearn*. GeeksforGeeks. https://www.geeksforgeeks.org/k-nearest-neighbors-knn-regression-with-scikit-learn/

Masui, T. (2025, January 31). *All You Need to Know about Gradient Boosting Algorithm − Part 1. Regression*. Towards Data Science. https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502/

Baladram, S. (2024, November 30). Extra Trees | Medium. *Medium*. https://medium.com/@samybaladram/extra-trees-explained-a-visual-guide-with-code-examples-4c2967cedc75

Chakure, A. (2023, April 27). *Random forest regression in Python explained*. Built In. https://builtin.com/data-science/random-forest-python#:~:text=What%20Is%20Random%20Forest%20Regression,trees%20while%20building%20the%20trees.

GeeksforGeeks. (2025, January 29). *Python | Decision Tree Regression using sklearn*. GeeksforGeeks. https://www.geeksforgeeks.org/python-decision-tree-regression-using-sklearn/

T-Test, Chi-Square, ANOVA, Regression, Correlation. . . (n.d.). https://datatab.net/tutorial/linear-regression

Machine Learning Mastery.. (n.d.). https://machinelearningmastery.com/xgboost-for-regression/

IBM. (2024, January 16). *Lasso Regression*. Ibm.com. https://www.ibm.com/think/topics/lasso-regression

IBM. (2023, September 21). *Ridge Regression*. Ibm.com. https://www.ibm.com/think/topics/ridge-regression