GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY, Tiruttani

1103-GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PHASE 4

PROJECT TITLE

PRODUCT DEMAND PREDICTION WITH MACHINE LEARNIN

Annapoorna VN

3rd yr, 5th sem

Reg no.:110321104001

vasaviannapoorna26@gmail.com

PHASE 3 : PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

GIVEN DATASET:

LINK:

https://www.kaggle.com/datasets/chakradharmattapalli/product-demand-prediction-with-machine-learning

STEP TO LOADING THE DATASET:

1. Import Libraries:

Start by importing the necessary libraries for data manipulation and analysis.

Python:

```
import pandas as pd
```

2. Load the Dataset:

Assuming you have a CSV file named "product_demand_data.csv" with the dataset in the same directory as your Python script, you can load it like this:

Python:

```
# Load the dataset

data = pd.read_csv("product_demand_data.csv")
```

If your dataset is in a different format (e.g., Excel, SQL database), Pandas provides functions to read those as well (e.g., pd.read_excel(), pd.read_sql()).

3. Explore the Dataset:

It's a good practice to take a quick look at the dataset to understand its structure. You can do this by examining the first few rows and some basic statistics:

Python:

```
# Display the first few rows of the dataset

print(data.head())
```

```
# Get summary statistics

print(data.describe())
```

This will give you an initial understanding of the data and help you identify any missing values or outliers.

4. Data Preprocessing:

Depending on the dataset, you might need to perform data preprocessing tasks such as handling missing values, encoding categorical variables, and creating new features. You can use Pandas for these tasks, along with libraries like NumPy and Scikit-learn.

Python:

```
# Check for missing values

print(data.isnull().sum())

# Handle missing values (e.g., fill with the mean)
data['Price'].fillna(data['Price'].mean(), inplace=True)
```

5.Feature engineering:

Feature engineering is a crucial step in building effective machine learning models for product demand prediction. The goal of feature engineering is to create informative and relevant input features from the raw data that can help the model better understand the underlying patterns and relationships that drive product demand.

1. Understanding the Problem:

- Before diving into feature engineering, it's essential to have a deep understanding of the problem domain. This includes understanding the product, the market, and the factors that can influence demand.

2. Data Preprocessing:

- Start by cleaning and preprocessing your raw data. This may involve handling missing values, dealing with outliers, and normalizing or scaling numerical features.

3. Time-based Features:

- For product demand prediction, time is often a critical factor. Create time-based features, such as day of the week, month, quarter, year, holidays, and seasonality patterns. These features can capture trends and cyclic behavior.

4. Lagged Features:

- Create lag features by shifting your target variable (demand) backward in time. For instance, you can create features like demand for the previous day, week, or month. This can help the model capture dependencies and autocorrelation in the data.

5. Rolling Window Statistics:

- Calculate rolling window statistics like moving averages, standard deviations, or sums over a certain time window. These features can help the model capture short-term trends and fluctuations.

6. Model training:

Training a machine learning model for product demand prediction is a valuable application in various industries, as it can help businesses optimize inventory management, production planning, and overall resource allocation.

1. Data Collection:

- Gather historical data on product demand, including sales, inventory levels, and any relevant external factors (e.g., promotions, seasonality, economic indicators).

2. Data Preprocessing:

- Handle missing data and outliers appropriately.

- Convert timestamps to date features and extract relevant information like day of the week, month, and year.

- Encode categorical variables, such as product categories and locations, using techniques like one-hot encoding or label encoding.

3. Feature Engineering:

- Create additional features that could be valuable for prediction, such as lag features (past demand), rolling statistics (moving averages), and other relevant metrics.

4. Train-Validation-Test Split:

- Split your dataset into three parts: training data, validation data, and test data. The training data is used to train the model, the validation data to fine-tune hyperparameters, and the test data to evaluate model performance.

5. Model Selection:

- Choose an appropriate machine learning model. Common choices include:

- Time series models (e.g., ARIMA, Exponential Smoothing, Prophet)

- Regression models (e.g., Linear Regression, Random Forest, Gradient Boosting)

- Deep learning models (e.g., Recurrent Neural Networks, Long Short-Term Memory networks)

6. Model Training:

- Train the selected model using the training data.

- Hyperparameter tuning: Optimize model parameters using the validation set, such as learning rate, number of trees (for ensemble methods), and network architecture (for deep

7. Model Evaluation:

- Use evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) to assess the model's accuracy.

8. Model Interpretability:

- Depending on the model used, consider techniques like feature importance analysis to understand which factors most influence product demand.

9. Model Deployment:

- Once you're satisfied with the model's performance, deploy it in a production environment. Ensure that it can make real-time predictions or batch predictions as needed.

10. Monitoring and Maintenance:

- Continuously monitor the model's performance and retrain it periodically with new data to keep it up to date.

7. Evaluation:

Evaluating a product demand prediction model is crucial to ensure that it performs well and meets the desired accuracy and reliability standards. Here are some common evaluation metrics and techniques used for product demand prediction with machine learning:

1. Mean Absolute Error (MAE):

MAE measures the average absolute difference between the predicted and actual demand. It gives an idea of the magnitude of errors in the predictions.

2. Mean Squared Error (MSE):

MSE calculates the average of the squared differences between predicted and actual demand. It penalizes larger errors more heavily, making it more sensitive to outliers.

3. Root Mean Squared Error (RMSE):

RMSE is the square root of the MSE. It provides a more interpretable measure of error since it's in the same unit as the target variable.

4. R-squared (R^2) Score:

R^2 measures the proportion of the variance in the target variable that is explained by the model. A higher R^2 indicates a better fit, but it's important to interpret it in the context of your specific problem.

5. Mean Absolute Percentage Error (MAPE):

MAPE calculates the percentage difference between predicted and actual demand. It's useful when you want to understand the relative error in predictions.

PROGRAM:

Python:

```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split

# Load the dataset
data = pd.read_csv("product_demand_data.csv")

# Handle missing values
data['Price'].fillna(data['Price'].mean(), inplace=True)

# Encode categorical variables (if applicable)
# For example, if 'Promotion' is a categorical variable with values 'Yes' and 'No':
```

```
data = pd.get_dummies(data, columns=['Promotion'], drop_first=True)


# Split the data into features (X) and the target (y)
X = data.drop('Demand', axis=1)  # Features (exclude the 'Demand' column)
y = data['Demand']  # Target variable


# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

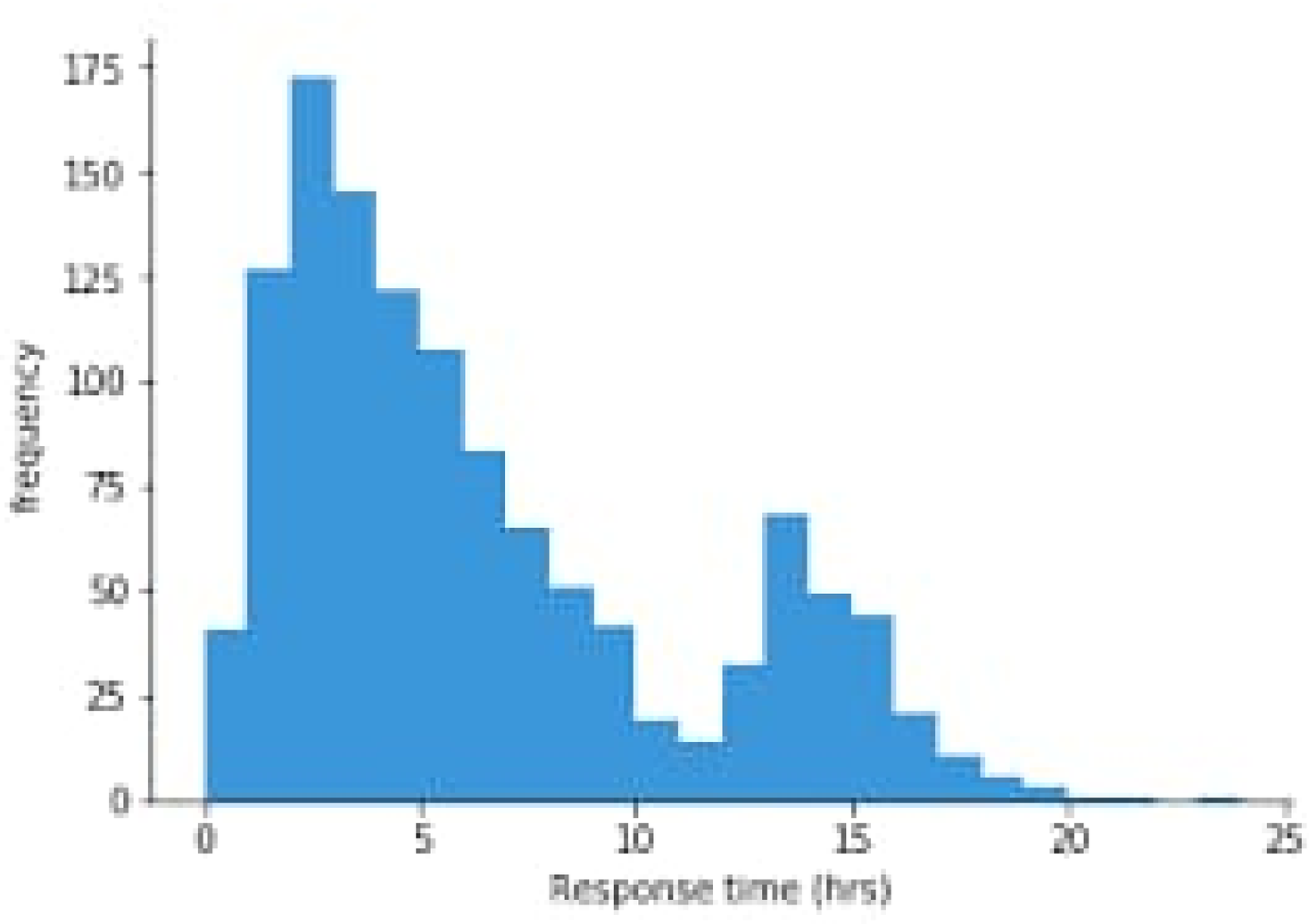
OUTPUT:1

```
Date           0
ProductID      0
Price         10
Promotion      0
Demand         0
dtype: int64
```

OUTPUT:2

```
Date           0
ProductID      0
Price          0
Promotion      0
Demand         0
dtype: int64
```

DEMAND PRODUCT ANALYSIS : Graph



CONCLUSION:

In conclusion, product demand prediction with machine learning offers a powerful and data-driven approach to help businesses optimize their operations and inventory management. By leveraging historical data, sophisticated algorithms, and predictive modeling techniques, companies can gain valuable insights into consumer behavior, market trends, and seasonal variations. This enables them to make informed decisions, reduce carrying costs, minimize stockouts, and improve customer satisfaction.

Machine learning models, such as regression, time series forecasting, and neural networks, provide accurate predictions that adapt to changing circumstances, ensuring better resource allocation and improved profitability. Moreover, these models can help businesses stay

competitive in dynamic markets by enhancing their ability to plan and respond effectively to changing demand patterns.