GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY, Tiruttani

1103-GRT INSTITUTE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PHASE 3

PROJECT TITLE

Product Demand Prediction With Machine Learning

Name: Annapoorna VN

3rd yr, 5th sem

Reg no.:110321104001

Vasaviannapoorna26@gmail.com

PHASE 3: PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Predic ng product demand with machine learning involves using historical data and various algorithms to forecast future demand for a par cular product. Here's a simplified process:

1. Data Collec on: Gather historical data on the product's sales, including factors that could affect demand like pricing, promo ons, seasonality, and external events.

2. Data Preprocessing: Clean and prepare the data by handling missing values, outliers, and encoding categorical variables.

3. Feature Engineering: Create relevant features from the data, such as lag variables (past sales), seasonality indicators, and any external data that could impact demand.

4. Split Data: Divide the dataset into training and tes ng sets for model evalua on.

5. Model Selec on: Choose an appropriate machine learning model, such as linear regression, decision trees, random forests, or more advanced techniques like me series forecas ng with ARIMA or deep learning with LSTM.

6. Model Training: Train the selected model on the training data.

7. Model Evalua on: Assess the model's performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) on the tes ng data.

8. Hyperparameter Tuning: Fine-tune the model's hyperparameters to improve its performance.

9. Deployment: Once you have a sa sfactory model, deploy it to predict future demand.

10. Monitoring and Upda ng: Con nuously monitor the model's performance and update it as more data becomes available.

Remember that the effec veness of your demand predic on model depends on the quality of your data, the choice of the right algorithm, and the con nuous refinement of the model over me.

3.1 DATASET AND ITS DETAIL EXPLANATION AND IMPLEMENTATION OF PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Certainly, I can provide a simplified example of product demand predic on with machine learning using a sample dataset and Python. For a real-world applica on, you would need a more comprehensive dataset, but this example will illustrate the process. I'll use a synthe c dataset for demonstra on purposes.

Sample Dataset Explana on:

Let's assume we have a dataset with the following columns:

- Date: The date of each sales record.
- ProductID: Unique iden fier for each product.
- Price: The price of the product.
- Promo on: Indicates whether a product was on promo on (1 for yes, 0 for no).

- Demand: The number of units sold on that date.

Here's how you can implement product demand predic on:

python

Import necessary libraries import

pandas as pd

from sklearn.model_selec on import train_test_split from

sklearn.ensemble import RandomForestRegressor from

sklearn.metrics import mean_squared_error, r2_score import

matplotlib.pyplot as plt

Load the dataset

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

Data preprocessing and feature engineering

data['Date'] = pd.to_date me(data['Date'])

data['Month'] = data['Date'].dt.month data['Day']

= data['Date'].dt.day

data['Day_of_week'] = data['Date'].dt.dayofweek

Split the data into training and tes ng sets

X = data[['Price', 'Promo on', 'Month', 'Day', 'Day_of_week']] y

= data['Demand']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

```
# Create and train a machine learning model (Random Forest) model =
RandomForestRegressor(n_es mators=100, random_state=42)
model.fit(X_train, y_train)

# Make predic ons on the test set y_pred =
model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred) print(f"Mean
Squared Error: {mse}") print(f"R-squared:
{r2}")

# Visualiza on (op onal) plt.sca er(y_test,
y_pred) plt.xlabel("Actual Demand")
plt.ylabel("Predicted Demand") plt.
tle("Actual vs. Predicted Demand")
plt.show()
```

In this example, we:

1. Load and preprocess the dataset.
2. Split the data into training and tes ng sets.
3. Choose a machine learning model (Random Forest) and train it.
4. Make predic ons on the test set and evaluate the model's performance.
5. Op onally, visualize the actual vs. predicted demand.

In a real-world scenario, you would work with more complex datasets, perform hyperparameter tuning, and consider me series analysis if your data involves temporal pa erns. Addi onally, you may deploy the model for ongoing demand predic on.

3.2 BEGIN BUILDING THE PROJECT BY LOAD THE DATASET

To begin building the project for product demand predic on with machine learning, you should start by loading the dataset. Here's a step-by-step guide using Python and the Pandas library:

1. Import Libraries:

Start by impor ng the necessary libraries for data manipula on 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 func ons to read those as well (e.g., `pd.read_excel()`, `pd.read_sql()`).

3. Explore the Dataset:

It's a good prac ce 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 sta s cs:

python

Display the first few rows of the dataset print(data.head())

Get summary sta s cs print(data.describe())

This will give you an ini al understanding of

the data and help you iden fy 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 crea ng new features. You can use Pandas for these tasks, along with libraries like NumPy and Scikit-learn.

For example, to handle missing values, you can use:

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)
```

These are the ini al steps for loading and exploring your dataset.

3.3 PREPROCESS DATASET

Preprocessing the dataset is a crucial step in preparing the data for machine learning. It involves tasks like handling missing values, encoding categorical variables, and feature engineering. Here's a simplified example of how to preprocess the dataset for product demand predic on:

python

```
# Import necessary libraries import
pandas as pd
from sklearn.model_selec on 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 'Promo on' is a categorical variable with values 'Yes' and 'No':
data = pd.get_dummies(data, columns=['Promo on'], 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 tes ng sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

In this code:

1. We handle missing values in the 'Price' column by filling them with the mean value.
2. If you have categorical variables (e.g., 'Promo on'), we use one-hot encoding to convert them into numerical format.
3. We split the dataset into features (X) and the target variable (y).
4. Finally, we split the data into training and tes ng sets to be used for building and evalua ng the machine learning model.

Depending on your dataset, you may need to perform addi onal preprocessing steps, such as scaling numerical features, dealing with outliers, and performing more advanced feature engineering. Addi onally, for me series data, you might need to consider me-based features and special handling.

3.4 PERFORMING DIFFERENT ANALYSIS NEEDED

1. Exploratory Data Analysis (EDA): EDA involves visualizing and summarizing your data to understand its characteris cs. You can create various plots and sta s cs to iden fy trends, pa erns, and outliers in your dataset. Common EDA tools include histograms, sca er plots, and box plots.

2. Correla on Analysis: Calculate correla ons between different features and the target variable ('Demand'). This can help you iden fy which features are most strongly associated with demand. You can use techniques like Pearson correla on or create correla on matrices.

3. Time Series Analysis (if applicable): If your data involves me-based informa on (e.g., sales over me), consider me series analysis to iden fy seasonality and trends. Methods like decomposi on and autocorrela on can be useful.

4. Feature Importance Analysis: If you're using a machine learning model, you can analyze feature importance scores to understand which features have the most impact on your predic ons. Techniques like feature importance plots for tree-based models can be used.

5. Residual Analysis: If you're using regression models, analyze the residuals (the difference between predicted and actual values) to check if there's any pa ern in the errors. Ideally, residuals should be normally distributed and randomly sca ered around zero.

6. Model Valida on: This involves assessing the performance of your machine learning model. Use metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared to evaluate how well your model predicts demand.

7. Hyperparameter Tuning: Op mize the hyperparameters of your machine learning model to improve its predic ve performance. Techniques like grid search or random search can be employed.

8. Cross-Valida on: Perform cross-valida on to assess how well your model generalizes to new data. This helps you understand whether your model might be overfi ng the training data.

9. Comparison of Mul ple Models: Try different machine learning algorithms and compare their performance to select the one that best fits your dataset.

10. Feature Engineering: If necessary, experiment with crea ng new features that might improve the predic ve power of your model.