PRODUCT DEMAND PREDICTION USING MACHINE LEARNING

PROJECT TITLE: Product demand prediction using machine learning

PHASE 4: Devlopment Part 2

SUBMITTED BY

G.janifar

TOPIC: Continue building the product demand prediction using machine learning model by features engineering, model training and evaluation.

DEFINITION:

In demand forecasting, machine learning algorithms can analyze historical sales patterns and predict future trends. The first step is collecting data about past sales, such as product type, quantity sold,

purchase frequency, seasonality, discounts, and more.

FEATURES ENGINEERING:

Predicting product demand using a machine learning model through feature engineering involves creating relevant input features that capture the factors influencing demand. Here's a high-level approach:

1. Data Collection: Gather historical data on product demand. This dataset should include information on products, time periods, and any relevant influencing factors (e.g., price, promotions, weather, seasonality).

2. Data Preprocessing:
   - Handle missing data.
   - Encode categorical features (e.g., product categories, location).
   - Create time-related features (e.g., day of the week, month).
   - Normalize or scale numerical features (e.g., price).

3. Feature Engineering:
   - Create lag features: Include past demand as features to

capture trends.

- Generate rolling statistics (e.g., rolling averages or moving sums) to capture seasonality.

- Create interaction features (e.g., product-category interaction).

- Include external factors (e.g., holidays, economic indicators) that might affect demand.

4. Data Splitting: Split the dataset into training and testing sets for model evaluation.

5. Model Selection::

- Choose an appropriate machine learning algorithm for regression or time series forecasting (e.g., linear regression, decision trees, Random Forest, ARIMA, LSTM).

- Train multiple models and evaluate their performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE).

6. Model Training:

- Train the selected model on the training data using the

engineered features.

7. Model Evaluation:

- Evaluate the model on the testing data to assess its predictive accuracy.

- Consider using cross-validation for more robust model evaluation.

8. Hyperparameter Tuning:

- Optimize model hyperparameters to improve performance.

9. Deployment:

- Deploy the trained model into a production environment for making real-time predictions.

- Implement a monitoring system to detect model performance degradation.

10. Continuous Improvement::

- Continuously update the model with new data to adapt to changing demand patterns.

- Refine feature engineering based on feedback and changing

business conditions.

The success of your demand prediction model depends on the quality of your data, the relevance of your engineered features, and the choice of an appropriate machine learning algorithm. It's an iterative process that may require fine-tuning and constant monitoring for accuracy.

MODEL TRAINING:

Feature Engineering:

Create relevant features that may affect demand, such as seasonality, trends, or lag variables (e.g., sales from previous months).

Data Split:

Divide your dataset into training and testing sets. Typically, you would use a large portion for training and a smaller portion for testing (e.g., 80/20 split).

Model Selection:

Choose a machine learning model suitable for time-series forecasting. Popular choices include Linear Regression, Decision Trees, Random Forests, XGBoost, and ARIMA for time-series data.

Model Training:

Train the selected model on the training data, using the historical data to learn patterns and relationships that influence demand.Model Evaluation:

Evaluate the model's performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) on the test data.

Hyperparameter Tuning:

Fine-tune the model's hyperparameters to improve its performance.Model Validation: Validate the model's performance on a holdout dataset or through cross-validation to ensure it's robust.

Prediction:

Once the model performs well, you can use it to make demand predictions for future periods.

Deployment:

Integrate the trained model into your business processes, so it can provide real-time or periodic demand forecasts.Monitoring and Maintenance:

Continuously monitor the model's performance and retrain it periodically to adapt to changing demand patterns.

EVALUATION:

```
import pandas as pd
import numpy as np
import plotly.express as px
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor

data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/
Website-data/master/demand.csv")
data.head()
```

DATA SET:

	ID	Store ID	Total Price	Base Price	Units Sold
ID	1.000000	0.007464	0.008473	0.018932	-0.010616
Store ID	0.007464	1.000000	-0.038315	-0.038848	-0.004372
Total Price	0.008473	-0.038315	1.000000	0.958885	-0.235625
Base Price	0.018932	-0.038848	0.958885	1.000000	

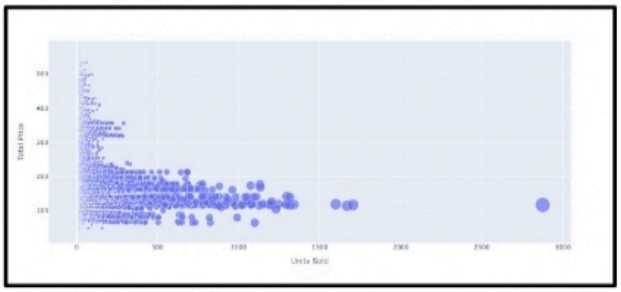
-0.140032

Units Sold -0.010616 -0.004372 -0.235625 -0.140032 1.000000

SCATTER PLOT:

Here I will use a scatter plot to see how the demand for the product varies with the price change:

```
fig = px.scatter(data, x="Units Sold", y="Total Price", size='Units Sold')
fig.show()
```



The correlation between the features of the dataset:

```
print(data.corr())
```

ID Store ID Total Price Base Price Units Sold ID

1.000000 0.007464 0.008473 0.018932 -0.010616

Store ID 0.007464 1.000000 -0.038315 -0.038848 -0.004372

Total Price 0.008473 -0.038315 1.000000 0.958885 -0.235625

Base Price	0.018932	-0.038848	0.958885	1.000000	-0.140032
Units Sold	-0.010616	-0.004372	-0.235625	-0.140032	1.000000

CONCLUSION:

The conclusion of a product demand prediction using machine learning would depend on the specific analysis and results of the project.