**Project 10: Product Demand Analysis**

**PHASE 3: DEVELOPMENT PART 1**

**410721243039 : JANASTHUTHI M S**

**DEVELOPMENT PART 1**

**1. Collect Historical Sales Data:**

-Access your company's internal systems (e.g., ERP, CRM) to retrieve historical sales data. Make sure the data includes relevant information such as Product ID, Product Name, Sales Date, Sales Quantity, and Sales Price.

**2. Collect External Factors Data:**

- Gather external factors data from various sources. This may include economic indicators, market trends, and seasonality factors that could affect product demand. You can collect data from government websites, industry publications, social media, or any other relevant sources.

**3. Data Cleaning:**

Start by cleaning the historical sales data:

Remove duplicates: Check for and eliminate duplicate records, if any.

Handle outliers: Identify and decide how to deal with any outliers in the data.

Correct errors: Look for and correct any data entry errors or inconsistencies.

Validate data integrity: Ensure data consistency and accuracy.

**4. Handling Missing Values:**

Assess the historical sales data and external factors data for missing values.

Decide on appropriate strategies to handle missing values, such as imputation (filling in missing values) or removing rows with missing values. The choice will depend on the nature and extent of missing data.

**5. Feature Engineering:**

Create new features that can enhance the predictive power of your model. Some common feature engineering techniques for demand prediction include:

Lag features: Include past sales values for the same or related products.

Rolling statistics: Calculate moving averages or other statistical measures over time.

Seasonal indicators: Add binary variables indicating seasons, holidays, or other important events.

Interaction terms: Create features representing interactions between different variables.

**6. Data Integration:**

Merge the cleaned historical sales data and external factors data, linking them by a common identifier like the Product ID.

**7. Scaling and Normalization:**

- Depending on the machine learning algorithm you plan to use, you may need to scale or normalize your data to bring all features to the same scale. Common techniques include Min-Max scaling (scaling to a specific range) and standardization (z-score normalization).

**8. Dataset Splitting:**

- Split your integrated dataset into training and test sets. The typical split is 70-30 or 80-20, where the larger portion is used for training and the smaller for testing. This helps evaluate the model's performance.

**9. Data Saving:**

Save the preprocessed dataset in a format that can be easily accessed by your chosen machine learning framework. Common formats include CSV or databases.

**10. Documentation:**

Keep detailed records of the preprocessing steps, any assumptions made, and any transformations applied to the data. Proper documentation is crucial for reproducibility and collaboration.

**PYTHON CODE**

import pandas as pd

# Load the historical sales data

df\_sales = pd.read\_csv('sales\_data.csv')

# Load the external factors data

df\_external\_factors = pd.read\_csv('external\_factors.csv')

# Preprocess the data

# Combine the historical sales data with external factors data using a common key, such as 'product\_id'

df = df\_sales.merge(df\_external\_factors, on='product\_id')

# Data cleaning (if needed)

# Example: Removing rows with missing values

df.dropna(inplace=True)

# Feature engineering (if needed)

# Example: Creating a new feature 'year' from 'sales date'

df['year'] = pd.to\_datetime(df['sales date']).dt.year

# Scaling (if needed)

# Example: Min-Max scaling 'sales quantity' and 'sales price'

df['sales quantity'] = (df['sales quantity'] - df['sales quantity'].min()) / (df['sales quantity'].max() - df['sales quantity'].min())

df['sales price'] = (df['sales price'] - df['sales price'].min()) / (df['sales price'].max() - df['sales price'].min())

# Save the preprocessed data to a new CSV file (optional)

df.to\_csv('preprocessed\_data.csv', index=False)

**CONCLUSION**

In this project's Development Part 1, the focus was on loading and preprocessing the dataset for a product demand prediction model. The data collection involved historical sales data from internal systems and external factors data from various sources. Preprocessing steps included data cleaning, handling missing values, feature engineering, and scaling for data normalization. The provided Python code example showcased these steps. Moving forward, the project will proceed to model selection, training, and evaluation, with the ultimate goal of enhancing demand forecasting and optimizing inventory management for better decision-making.