**Product Demand Prediction Using Machine Learning**

Phase-4 Project Submission

This phase of our project submission will consist of the development of feature engineering, model training and assessment, as well as various analyses as required. This file will be submitted as part of the phase-4 project.

Feature engineering:

These are a few notable instances of feature engineering.

**Features of lag:** To record prior demand patterns, such as demand from the previous week or month, provide lag features.

**Rolling statistics:** To identify patterns and seasonality, compute rolling mean, median, or other statistics over a predetermined time period.

**Seasonal and holiday indicators**: Include binary indicators for specific times of the year.

**Features of the price and promotion:** Provide details on product costs and special offers as they might have a big influence on demand.

**Weather data:** Since the weather might have an impact on the demand for particular items, if applicable, include weather data as a feature.

We can eliminate the null values from the dataset for this sub-division in the data set we have for this project.

The following code consist of the process to print and see the dataset elements which are present inside the dataset.

To begin, let's import the dataset and required Python modules for the product demand forecast task:

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

Let’s now load our data set and print the information on the data set

data = pd.read\_csv("/content/dataset.csv")

data.head()

This is the dataset output:

|  | **ID** | **Store ID** | **Total Price** | **Base Price** | **Units Sold** |
| --- | --- | --- | --- | --- | --- |
| **0** | 1 | 8091 | 99.0375 | 111.8625 | 20 |
| **1** | 2 | 8091 | 99.0375 | 99.0375 | 28 |
| **2** | 3 | 8091 | 133.9500 | 133.9500 | 19 |
| **3** | 4 | 8091 | 133.9500 | 133.9500 | 44 |
| **4** | 5 | 8091 | 141.0750 | 141.0750 | 52 |

There may be some null values in the data set that need to be removed in order to get a precised output:

Now let’s have a look at whether this dataset contains any null values or not:

1

data.isnull().sum()

**ID 0**

**Store ID 0**

**Total Price 1**

**Base Price 0**

**Units Sold 0**

**dtype: int64**

So the dataset has only one missing value in the **Total Price** column, I will remove that entire row for now:

data = data.dropna()

This concludes the Feature engineering process.

**Model Training:**

Utilizing the training data, train the chosen model. Respecting the temporal order is important when working with time series data.

Training a machine learning model to predict the product's demand at different price points is the next phase. I'll use the Total Price and Base Price columns as the model's training features, and the Units Sold column as the model's labels:

x = data[["Total Price", "Base Price"]]

y = data["Units Sold"]

Now let's use the decision tree regression approach to train our model by splitting the data into training and test sets:

xtrain, xtest, ytrain, ytest = train\_test\_split(x, y,

                                                test\_size=0.2,

                                                random\_state=42)

from sklearn.tree import DecisionTreeRegressor

model = DecisionTreeRegressor()

model.fit(xtrain, ytrain)

By the above done process we have now trained the modal for our project.

Evaluation:

Now let's input the attributes (Total Price, Base Price) into the model and anticipate the highest possible amount that might be requested using those values:

features = [["Total Price", "Base Price"]]

features = np.array([[120.00, 135.00]])

model.predict(features)

Output:

**array([49.16666667])**

Now let evaluate with some other input values for our modal:

features = [["Total Price", "Base Price"]]

features = np.array([[111.00, 120.00]])

model.predict(features)

Output:

**array([61.])**

features = [["Total Price", "Base Price"]]

features = np.array([[150.00, 97.00]])

model.predict(features)

Output:

**array([112.66666667])**

This concludes our evaluation phase for the Phase-4 project submission.