**Product Demand Prediction with machine learning**

**Project Title:** product demand prediction

Phase3: **Development Part 1**

**TOPIC:** Start building the product demand prediction model

By loading and pre-processing the dataset.

***INTRODUCTION :***

Building a product demand prediction model is a data-driven process

involves harnessing the power of machine learning to analyze

historical product data and make informed price predictions.

This journey begins with the fundamental steps of data loading and

preprocessing.

This introduction will guide you through the initial steps of the

process. We'll explore how to import essential libraries, load the product demand dataset, and perform preprocessing steps.

Data preprocessing is crucial as it helps clean, format, and prepare the data for further analysis.

THE BELOW ARE GIVEN DATASET

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | STORE ID | TOTAL  PRICE | BASE  PRICE | UNITS  SOLD |
| 1 | 8091 | 99.0375 | 111.8625 | 20 |
| 2 | 8091 | 99.0375 | 99.0375 | 28 |
| 3 | 8091 | 133.95 | 133.95 | 19 |
| 4 | 8091 | 133.95 | 133.95 | 44 |
| 5 | 8091 | 141.075 | 141.075 | 52 |
| 9 | 8091 | 227.2875 | 227.2875 | 18 |
| 10 | 8091 | 327.0375 | 327.0375 | 47 |
| 13 | 8091 | 210.0 | 210.9 | 50 |
| 14 | 8091 | 234.4125 | 234.4125 | 82 |
| 17 | 8095 | 99.0375 | 327.00375 | 99 |
| 19 | 8095 | 97.6125 | 210.9 | 120 |
| 22 | 8095 | 98.325 | 234.4125 | 40 |
| 23 | 8095 | 133.2375 | 99.0375 | 60 |

:

The given above dataset contain “5000”Rows and 5 columns.

**Necessary step to follow:**

**1.Import Libraries:**

* Start by importing the necessary libraries

**PROGRAM:**

**Import pandas as pd**

**Import numpy as np**

**From Sklearn.model\_selection import train\_train**

**\_split**

**from Sklearn.preprocessing import StandardScaler**

**2. Load the Dataset:**

Load your dataset into a Pandas Data Frame. You can typically find

Product demand datasets in CSV format, but you can adapt this code to other formats as needed.

**3. Exploratory Data Analysis (EDA):**

Perform EDA to understand your data better. This includes

checking for missing values, exploring the data's statistics, and

visualizing it to identify patterns.

Program:

# Check for missing values

print(df.isnull().sum())

# Explore statistics

print(df.describe())

# Visualize the data (e.g., histograms, scatter plots, etc.)

**LOAD FORECASTING:**

1. With an increase in population and an overall expansion of the energy infrastructure, electricity demand is increasing rapidly.
2. To manage this increasing demand efficiently, so-called smart grids are used.
3. The cardinal feature of demand-side management in smart grids is load forecasting, as it allows the operators of the smart grid to make efficient and effective decisions, which is the topic of interest of this blog post.

Feature Engineering:

* Feature engineering is the pre-processing step of machine learning, which is used to transform raw data into features that can be used for creating a predictive model using Machine learning or statistical Modelling.
* Feature engineering in machine learning aims to improve the performance of models.

PROGRAM :

# Example: One-hot encoding for categorical variables

df = pd. get \_ dummies ( df, columns= [' Avg. ID ', ' Avg. Store ID’]);

1. Split the Data:

Split your dataset into training and testing sets. This helps you evaluate

your model's performance later.

Program:

X = df. drop('price', axis=1) # Features

y = df ['price'] # Target variable

X \_train, X \_test, y \_train, y \_test = train \_test \_split (X, y, test \_size=0.2,

Random \_state=42)

1. Feature Scaling:

Apply feature scaling to normalize your data, ensuring that all

features have similar scales. Standardization (scaling to mean=0 and

std=1) is a common choice.

Program:

scaler =Standard Scaler ()

X \_train = scaler. fit \_transform (X \_train)

X \_test = scaler. transform (X \_test)

Importance of loading and processing dataset:

Loading and preprocessing the dataset is an important first step in

building any machine learning model.

However, it is especially important for **product demand prediction model** as product demand dataset are often complex and noisy.

By loading and preprocessing the dataset, we can ensure that the

machine learning algorithm is able to learn from the data effectively and accurately.

**Challenges involved in loading and preprocessing a product demand dataset:**

There are a number of challenges involved in loading and preprocessing

a product demand dataset including:

* **Inadequate or nonexistent data profiling:**

Data analysts and business users should never be surprised by state of the data when doing analytics-or worse, have their decisions be affected by faulty data that they are unaware of DATAPROFILING.

One of the core steps in data preparation process, should prevent that from happening. but they are different reasons:

* The people who gather and prepare the data assume it’s valid because it already being used in report or spreadsheets.
* Someone who collects a large volume of data only profiles a sample data set because of the time it would take to do the full.
* Custom-coded SQL queries or spreadsheet function used to profile data aren’t comprehensive enough to find all of the anomalies or other problems in the data

**MISSING OR INCOMPLETE DATA:**

A common data quality issue is fields or attributes with missing value, such as nulls or blanks Zero that represent a missing value rather than the number 0, or an entire fields missing in a delimited file. The data preparation question raised by these missing values are a whether they indicate that there is an error in the data.

**INVALID DATA VALUES:**

**Invalid values are another common data quality issue. They include misspellings other typos, duplicate entries and outliers, such as**

* **Wrong dates or number that aren’t reasonable given the data’s context.**
* **These errors can be created even in modern enterprise application with data validation features and then end up in curated data sets.**

**Data Enrichment:**

**One of the key steps in creating the business context needed for analytics is enriching data.**

**1.Examples of data enrichment measures include the following:**

* **Calculating business metrics and KPIs;**
* **Filtering data based on business rules application to the planned analytics;**
* **Augmenting data with additional internal or external sources.**

**Enriching data aren’t easy task. Deciding what needs to be done in a dataset is often complicated and the required data required data enrichment work can be a time-consuming procedure.**

**How to overcome the challenges of loading and preprocessing a Product demand dataset:**

***There are a number of things that can be done to overcome the challenges of loading and preprocessing a product demand dataset, including:***

 **Use a data preprocessing library:**

There are a number of libraries available that can help with data

preprocessing tasks, such as inadequate , missing or incomplete data and invalid data values, data enrichment.

**Carefully consider the specific needs of your model**:

**The best way to preprocess the data will depend on the specific**

**machine learning algorithm that you are using.**

It is important to ***in a way that is compatible with the algorithm.*** carefully ***consider the requirements of the algorithm and to preprocess the data***

**Validate the preprocessed data:**

* **It is important to validate the preprocessed data to ensure that it is in a format that can be used by the machine learning algorithm and that it is of high quality.**

**This can be done by inspecting the data visually or by using statistical methods.**

1.Loading the dataset:

* Loading the dataset using machine learning is the process of bringing the data into the machine learning environment so that it can be used to train and evaluate a model.
* The specific steps involved in loading the dataset will vary depending on the machine learning library or framework that is being used. However, there are some general steps that are common to most machine learning frameworks:

1.**Identify the dataset:**

The first step is to identify the dataset that you want to load. This dataset may be stored in a local file, in a database, or in a cloud storage service.

**2.Load the dataset:**

Once you have identified the dataset, you need to load it into the

machine learning environment. **This may involve using a built-in**

**function in the machine learning library, or it may involve writing your own code.**

**3.Preprocess the dataset:**

**Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training and**

**test sets.**

Program:

import pandas as pd

import numpy as np

import seaborn a sns

import matplotlib .pyplot as plt

from Sklearn . model \_ selection import train \_ test \_split

from sklearn. preprocessing import Standard Scaler

from sklearn. metrics import r2\_score,

mean \_ absolute \_ error , mean \_ squared \_error

from sklearn. linear \_model import LinearRegression

from sklearn. linear \_model import Lasso

from sklearn. ensemble import RandomForestRegressor

from sklearn.svm import SVR

import xgboost as xg

%matplotlib inline

import warnings

warning . filter warning ("ignore")

/opt/conda/lib/python3.10/site-packages/scipy/\_\_init\_\_.py:146:

User Warning: A NumPy version >=1.16.5 and <1.23.0 is required for

this version of SciPy (detected version 1.23.5

warning. warn (f"A NumPy version > = {np \_ min version} and

< {np \_ max version}"

**LOADING DATASET:**

dataset = pd. read \_ csv('E:/product\_demand.csv')

**Data Exploration**

**DATA SET:**

**OUTPUT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | STORE ID | TOTAL  PRICE | BASE  PRICE | UNITS  SOLD |
| 1 | 8091 | 99.0375 | 111.8625 | 20 |
| 2 | 8091 | 99.0375 | 99.0375 | 28 |
| 3 | 8091 | 133.95 | 133.95 | 19 |
| 4 | 8091 | 133.95 | 133.95 | 44 |
| 5 | 8091 | 141.075 | 141.075 | 52 |
| 9 | 8091 | 227.2875 | 227.2875 | 18 |
| 10 | 8091 | 327.0375 | 327.0375 | 47 |
| 13 | 8091 | 210.0 | 210.9 | 50 |
| 14 | 8091 | 234.4125 | 234.4125 | 82 |
| 17 | 8095 | 99.0375 | 327.00375 | 99 |
| 19 | 8095 | 97.6125 | 210.9 | 120 |
| 22 | 8095 | 98.325 | 234.4125 | 40 |
| 23 | 8095 | 133.2375 | 99.0375 | 60 |

:

**2.Preprocessing the dataset**:

* **Data preprocessing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.**
* **This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.**

Some common data preprocessing tasks include:

 Data cleaning: This involves identifying and correcting errors and

inconsistencies in the data. For example, this may involve

removing duplicate records, correcting typos, and filling in missing

values.

 Data transformation: This involves converting the data into a

format that is suitable for the analysis task. For example, this may

involve converting categorical data to numerical data, or scaling

the data to a suitable range.

 Feature engineering: This involves creating new features from

the existing data. For example, this may involve creating features

that represent interactions between variables, or features that

represent summary statistics of the data.

 Data integration: This involves combining data from multiple

sources into a single dataset. This may involve resolving

inconsistencies in the data, such as different data formats or

different variable names.

Data preprocessing is an essential step in many data

science projects. By carefully preprocessing the data, data scientists can

improve the accuracy and reliability of their results.

PROGRAM :

**Import the libraries**:

**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(“E:/product\_demand.csv”)**

**data.head()**

**OUTPUT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **STORE ID** | **TOTAL PRICE** | **BASE PRICE** | **UNITS SOLD** |
| **1** | **8091** | **99.0375** | **111.8625** | **20** |
| **2** | **8091** | **99.0375** | **99.0375** | **28** |
| **3** | **8091** | **133.95** | **133.95** | **19** |
| **4** | **8091** | **133.95** | **133.95** | **44** |

**# Whether the dataset contain null (or) not**

**IN[2]:**

**data.isnull().sum()**

**OUT[2]:**

|  |  |
| --- | --- |
| **ID** | **0** |
| **Store ID** | **0** |
| **Total price** | **1** |
| **Base price** | **0** |
| **Units sold** | **0** |
| **dtypes** | **Int64** |

**IN [3]**

**data = data.dropna()**

**uses of dropna():**

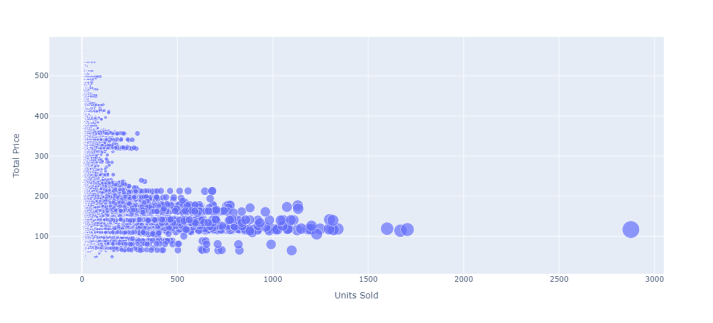
**#The dataset has only one missing value in the TOTAL PRICE COLUMN, it will remove entire row**

**IN[4]:**

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

**fig.show()**

**OUT[4]:**



**IN[5]:**

**print(data.corr())**

**OUT[5]:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **ID** | **Store ID** | **Total price** | **Base price** | **Units Sold** |
| **ID** | **1.000000** | **0.007464** | **0.008473** | **0.01892** | **0.1061** |
| **Store ID** | **0.007464** | **1.000000** | **-0.03831** | **0.03884** | **0.0043** |
| **Total price** | **0.008473** | **-0.03831** | **1.000000** | **0.95888** | **0.2356** |
| **Base price** | **0.018932** | **-0.03884** | **0.958885** | **1.00000** | **0.1400** |
| **Units Sold** | **-0.01061** | **-0.00437** | **-0.23562** | **0.140032** | **1.0000** |

**IN [6]:**

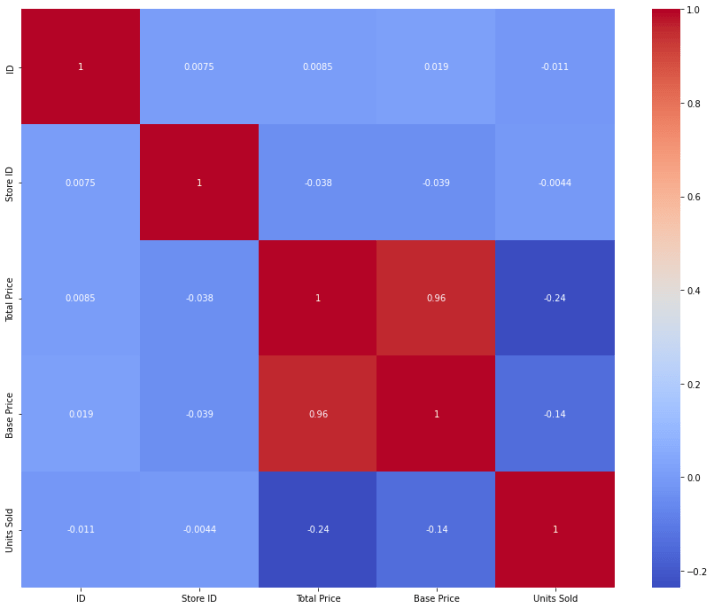
**correlations = data.corr(method='pearson')**

**plt.figure(figsize=(15, 12))**

**sns.heatmap(correlations, cmap="coolwarm", annot=True)**

**plt.show()**

**OUT [6]:**



**Program:**

# Importing necessary libraries

**import pandas as pd**

**import numpy as np**

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.preprocessing import StandardScaler**

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

**y = data["Units Sold"]**

**xtrain, xtest, ytrain, ytest = train\_test\_split test\_size=0.2, random\_state=42)**

**from sklearn.tree import DecisionTreeRegressor**

**model = DecisionTreeRegressor()**

**model.fit(xtrain, ytrain)**

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

**features = np.array([[133.00, 140.00]])**

**model.predict(features)**

**OUTPUT:**

array([27.])

**Conclusion:**

1.**In the quest to build a product demand prediction model, we have embarked on a critical journey that begins with loading and preprocessing the dataset.**

* **We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.**

**2. Data preprocessing emerged as a pivotal aspect of this process.**

* **It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.**

**3. With these foundational steps completed, our dataset is now primed for the subsequent stages of building and training a house price prediction.**