TITLE-PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

Project title: product demand prediction with machine learning

Phase 3: Development Part 1 In this part you will begin building your project by loading and preprocessing the dataset. Begin conducting the product demand prediction by collecting and preprocessing the data. Collect and preprocess the product data for analysis.

Data Preprocessing:

• Data preprocessing is a crucial step within the statistics analysis and gadget gaining knowledge of pipeline.

• It includes a sequence of strategies and operations finished on uncooked statistics to clean, organize, and transform it right into a layout that is suitable for analysis or device mastering version schooling.

• Data preprocessing goals to enhance the firstclass of the records, making it greater reliable and conducive to generating accurate consequences. Here are some common tasks and techniques involved in data preprocessing:

Data Cleaning: • Handling missing values: Deciding how to deal with missing data, whether by imputing values or removing incomplete records. • Outlier detection and treatment: Identifying and handling data points that significantly deviate from the norm. Noise reduction:

• Smoothing noisy data through techniques like filtering. Data Transformation: • Data normalization: Scaling numerical features to a standard range (e.g., between 0 and 1) to ensure that they have similar influence in the analysis. • Encoding categorical variables: Converting categorical data into numerical format, such as one-hot encoding or label encoding.

• Feature engineering: Creating new features or modifying existing ones to capture more meaningful information from the data.

• Dimensionality reduction: Reducing the number of features while retaining essential information, using methods like Principal Component Analysis (PCA). Data Integration: • Merging or joining datasets: Combining data from multiple sources into a single dataset for analysis. Aggregation: Summarizing data at a higher level of granularity, such as aggregating daily sales into monthly totals. Data Reduction:

• Sampling: Reducing the size of a large dataset by randomly selecting a representative subset.

• Binning: Grouping continuous data into discrete bins to simplify analysis. • Filtering: Selecting a subset of data based on specific criteria. Data Standardization: • Ensuring that data follows a consistent format and structure. • Date and time format conversion: Converting date and time data into a uniform format. • Currency conversion: Converting monetary values into a common currency. Data Scaling: • Scaling numerical data to a common range to prevent some features from dominating the analysis. Data preprocessing is an iterative process that may involve several of these steps in various orders, depending on the specific dataset and the analysis goals. Proper data preprocessing is essential for improving the accuracy and effectiveness of machine learning models, as well as for making data more accessible for traditional statistical analysis. Here is the data preprocessing codes along with the output of the given dataset: Importing the libraries: Import three basic libraries which are very common in machine learning and will be used every time you train a model • NumPy: it is a library that allows us to work with arrays and as most machine learning models work on arrays NumPy makes it easier • Matplotlib: this library helps in plotting graphs and charts, which are very useful while showing the result of your model • Pandas: pandas allows us to import our dataset and also creates a matrix of features containing the dependent and independent variable

#Connect the google drive for reading the # Connect the google drive

from google.colab import drive drive.mount('/content/drive')

Mounted at /content/drive

# Preparing Dataset # Import the dataset import pandas as pd

dataset = pd.read\_csv('/content/drive/MyDrive/BIT/Customer-churn.csv')

# Preparing Dataset # Import the dataset import pandas as pd

dataset = pd.read\_csv('/content/drive/MyDrive/PRODUCT/Book1.csv')

print(dataset)

 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

5 9 8091 227.2875 227.2875 18

6 10 8091 327.0375 327.0375 47

7 13 8091 210.9000 210.9000 50

8 14 8091 190.2375 234.4125 82

9 17 8095 99.0375 99.0375 99

10 18 8095 97.6125 97.6125 120

11 19 8095 98.3250 98.3250 40

12 22 8095 133.2375 133.2375 68

13 23 8095 133.9500 133.9500 87

14 24 8095 139.6500 139.6500 186

15 27 8095 236.5500 280.0125 54

16 28 8095 214.4625 214.4625 74

17 29 8095 266.4750 296.4000 102

18 30 8095 173.8500 192.3750 214

19 31 8095 205.9125 205.9125 28

20 32 8095 205.9125 205.9125 7

21 33 8095 248.6625 248.6625 48

22 34 8095 200.9250 200.9250 78

23 35 8095 190.2375 240.8250 57

24 37 8095 427.5000 448.1625 50

25 38 8095 429.6375 458.1375 62

26 39 8095 177.4125 177.4125 22

27 42 8094 87.6375 87.6375 109

28 43 8094 88.3500 88.3500 133

dataset.dropna

<bound method DataFrame.dropna of 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 |
| 5 | 9 | 8091 | 227.2875 | 227.2875 | 18 |
| 6 | 10 | 8091 | 327.0375 | 327.0375 | 47 |
| 7 | 13 | 8091 | 210.9000 | 210.9000 | 50 |
| 8 | 14 | 8091 | 190.2375 | 234.4125 | 82 |
| 9 | 17 | 8095 | 99.0375 | 99.0375 | 99 |
| 10 | 18 | 8095 | 97.6125 | 97.6125 | 120 |
| 11 | 19 | 8095 | 98.3250 | 98.3250 | 40 |
| 12 | 22 | 8095 | 133.2375 | 133.2375 | 68 |
| 13 | 23 | 8095 | 133.9500 | 133.9500 | 87 |
| 14 | 24 | 8095 | 139.6500 | 139.6500 | 186 |
| 15 | 27 | 8095 | 236.5500 | 280.0125 | 54 |
| 16 | 28 | 8095 | 214.4625 | 214.4625 | 74 |
| 17 | 29 | 8095 | 266.4750 | 296.4000 | 102 |
| 18 | 30 | 8095 | 173.8500 | 192.3750 | 214 |
| 19 | 31 | 8095 | 205.9125 | 205.9125 | 28 |
| 20 | 32 | 8095 | 205.9125 | 205.9125 | 7 |
| 21 | 33 | 8095 | 248.6625 | 248.6625 | 48 |
| 22 | 34 | 8095 | 200.9250 | 200.9250 | 78 |
| 23 | 35 | 8095 | 190.2375 | 240.8250 | 57 |
| 24 | 37 | 8095 | 427.5000 | 448.1625 | 50 |
| 25 | 38 | 8095 | 429.6375 | 458.1375 | 62 |
| 26 | 39 | 8095 | 177.4125 | 177.4125 | 22 |
| 27 | 42 | 8094 | 87.6375 | 87.6375 | 109 |
| 28 | 43 | 8094 | 88.3500 | 88.3500 | 133> |

dataset.info

<bound method DataFrame.info of 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 |
| 5 | 9 | 8091 | 227.2875 | 227.2875 | 18 |
| 6 | 10 | 8091 | 327.0375 | 327.0375 | 47 |
| 7 | 13 | 8091 | 210.9000 | 210.9000 | 50 |
| 8 | 14 | 8091 | 190.2375 | 234.4125 | 82 |
| 9 | 17 | 8095 | 99.0375 | 99.0375 | 99 |
| 10 | 18 | 8095 | 97.6125 | 97.6125 | 120 |
| 11 | 19 | 8095 | 98.3250 | 98.3250 | 40 |
| 12 | 22 | 8095 | 133.2375 | 133.2375 | 68 |
| 13 | 23 | 8095 | 133.9500 | 133.9500 | 87 |
| 14 | 24 | 8095 | 139.6500 | 139.6500 | 186 |
| 15 | 27 | 8095 | 236.5500 | 280.0125 | 54 |
| 16 | 28 | 8095 | 214.4625 | 214.4625 | 74 |
| 17 | 29 | 8095 | 266.4750 | 296.4000 | 102 |
| 18 | 30 | 8095 | 173.8500 | 192.3750 | 214 |
| 19 | 31 | 8095 | 205.9125 | 205.9125 | 28 |
| 20 | 32 | 8095 | 205.9125 | 205.9125 | 7 |
| 21 | 33 | 8095 | 248.6625 | 248.6625 | 48 |
| 22 | 34 | 8095 | 200.9250 | 200.9250 | 78 |
| 23 | 35 | 8095 | 190.2375 | 240.8250 | 57 |
| 24 | 37 | 8095 | 427.5000 | 448.1625 | 50 |
| 25 | 38 | 8095 | 429.6375 | 458.1375 | 62 |
| 26 | 39 | 8095 | 177.4125 | 177.4125 | 22 |
| 27 | 42 | 8094 | 87.6375 | 87.6375 | 109 |
| 28 | 43 | 8094 | 88.3500 | 88.3500 | 133> |

dataset.isnull()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |
| 5 | 9 | 8091 | 227.2875 | 227.2875 | 18 |
| 6 | 10 | 8091 | 327.0375 | 327.0375 | 47 |
| 7 | 13 | 8091 | 210.9000 | 210.9000 | 50 |
| 8 | 14 | 8091 | 190.2375 | 234.4125 | 82 |
| 9 | 17 | 8095 | 99.0375 | 99.0375 | 99 |
| 10 | 18 | 8095 | 97.6125 | 97.6125 | 120 |
| 11 | 19 | 8095 | 98.3250 | 98.3250 | 40 |
| 12 | 22 | 8095 | 133.2375 | 133.2375 | 68 |
| 13 | 23 | 8095 | 133.9500 | 133.9500 | 87 |
| 14 | 24 | 8095 | 139.6500 | 139.6500 | 186 |
| 15 | 27 | 8095 | 236.5500 | 280.0125 | 54 |
| 16 | 28 | 8095 | 214.4625 | 214.4625 | 74 |
| 17 | 29 | 8095 | 266.4750 | 296.4000 | 102 |
| 18 | 30 | 8095 | 173.8500 | 192.3750 | 214 |
| 19 | 31 | 8095 | 205.9125 | 205.9125 | 28 |
| 20 | 32 | 8095 | 205.9125 | 205.9125 | 7 |
| 21 | 33 | 8095 | 248.6625 | 248.6625 | 48 |
| 22 | 34 | 8095 | 200.9250 | 200.9250 | 78 |
| 23 | 35 | 8095 | 190.2375 | 240.8250 | 57 |
| 24 | 37 | 8095 | 427.5000 | 448.1625 | 50 |
| 25 | 38 | 8095 | 429.6375 | 458.1375 | 62 |
| 26 | 39 | 8095 | 177.4125 | 177.4125 | 22 |
| 27 | 42 | 8094 | 87.6375 | 87.6375 | 109 |
| 28 | 43 | 8094 | 88.3500 | 88.3500 | 133> |
| **18** | False | False | False | False | False |

import seaborn as sns

**0** False False False False False

**1** False False False False False

**2** False False False False False

**3** False False False False False

**4** False False False False False

**5** False False False False False

**6** False False False False False

**7** False False False False False

**8** False False False False False

**9** False False False False False

**10** False False False False False

**11** False False False False False

**12** False False False False False

**13** False False False False False

**14** False False False False False

**15** False False False False False

**16** False False False False False

**17** False False False False False

**19** False False False False False

**ID Store ID Total Price Base Price Units Sold**

dataset.describe

<bound method NDFrame.describe of ID Store ID Total Price Base Price Units Sold

sns.pairplot(dataset)

**20** False False False False False

**21** False False False False False

**22** False False False False False

**23** False False False False False

**24** False False False False False

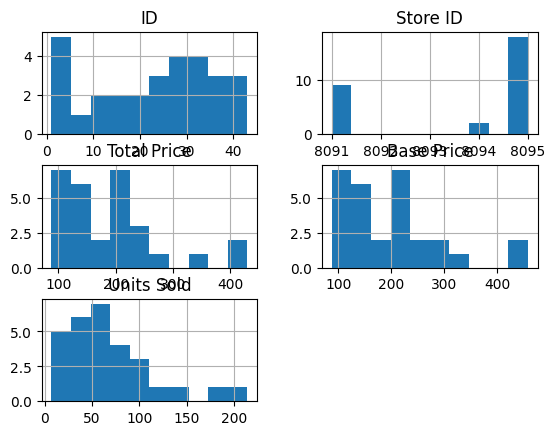
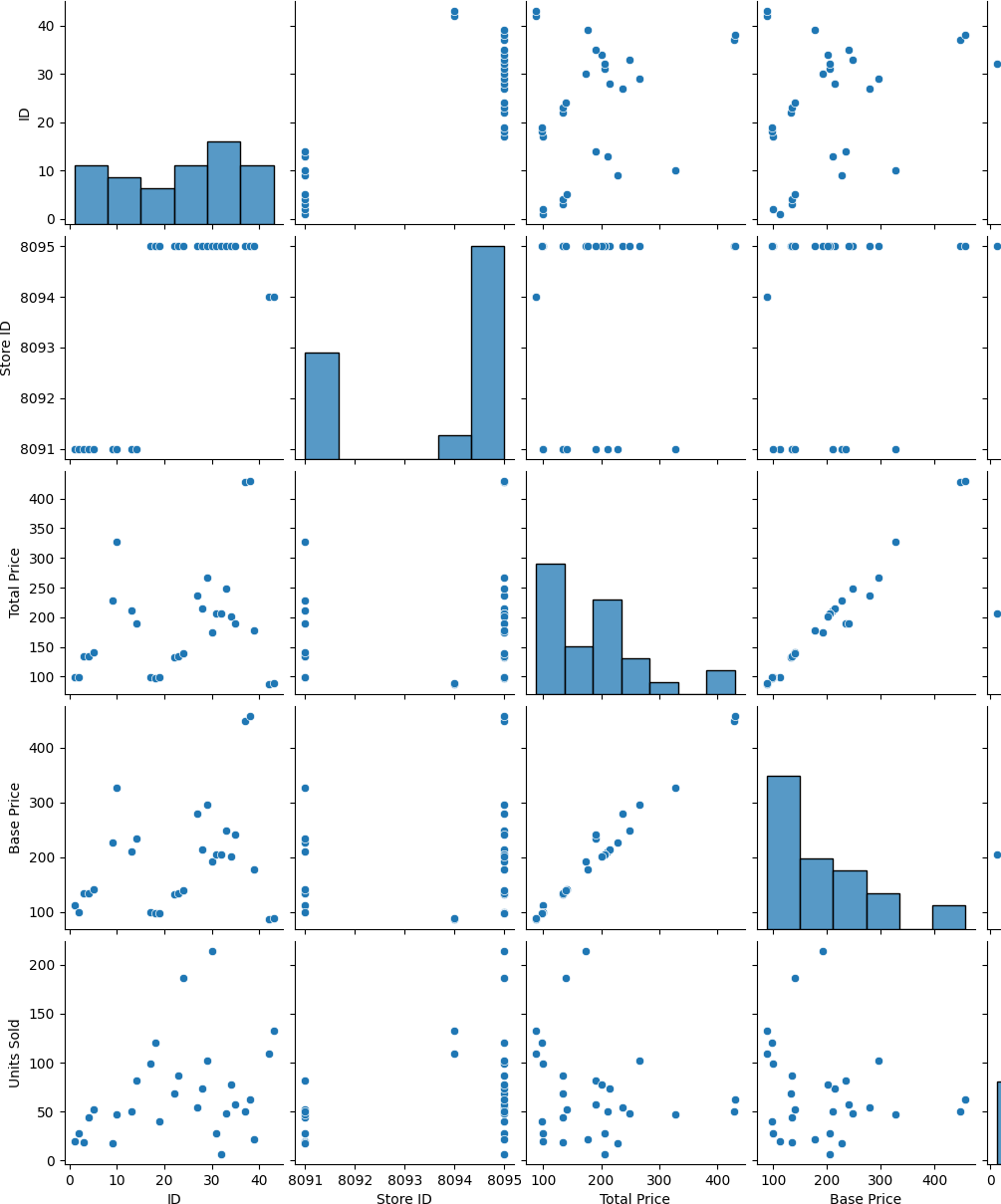
**25** False False False False False

**26** False False False False False

**27** False False False False False

**28** False False False False False

<seaborn.axisgrid.PairGrid at 0x7a950ddbbb20>



dataset.hist()

array([[<Axes: title={'center': 'ID'}>,

<Axes: title={'center': 'Store ID'}>],

[<Axes: title={'center': 'Total Price'}>,

<Axes: title={'center': 'Base Price'}>],

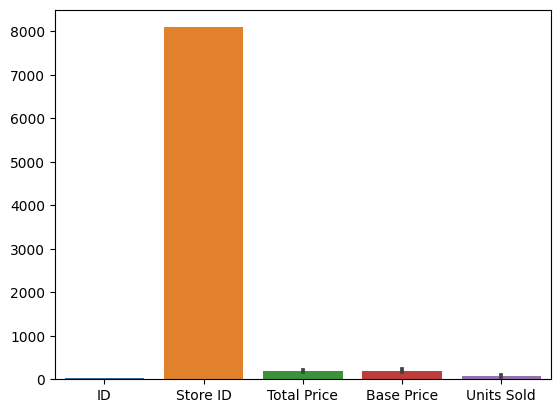
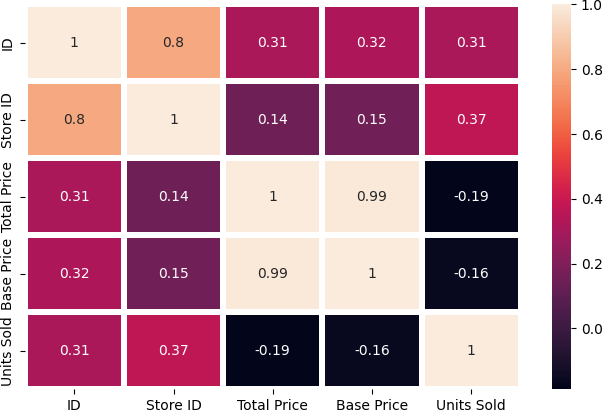
[<Axes: title={'center': 'Units Sold'}>, <Axes: >]], dtype=object)

import matplotlib.pyplot as plt

plt.figure(figsize=(8,5))

sns.heatmap(dataset.corr(),annot=True,linewidth=3) plt.show

<function matplotlib.pyplot.show(close=None, block=None)>

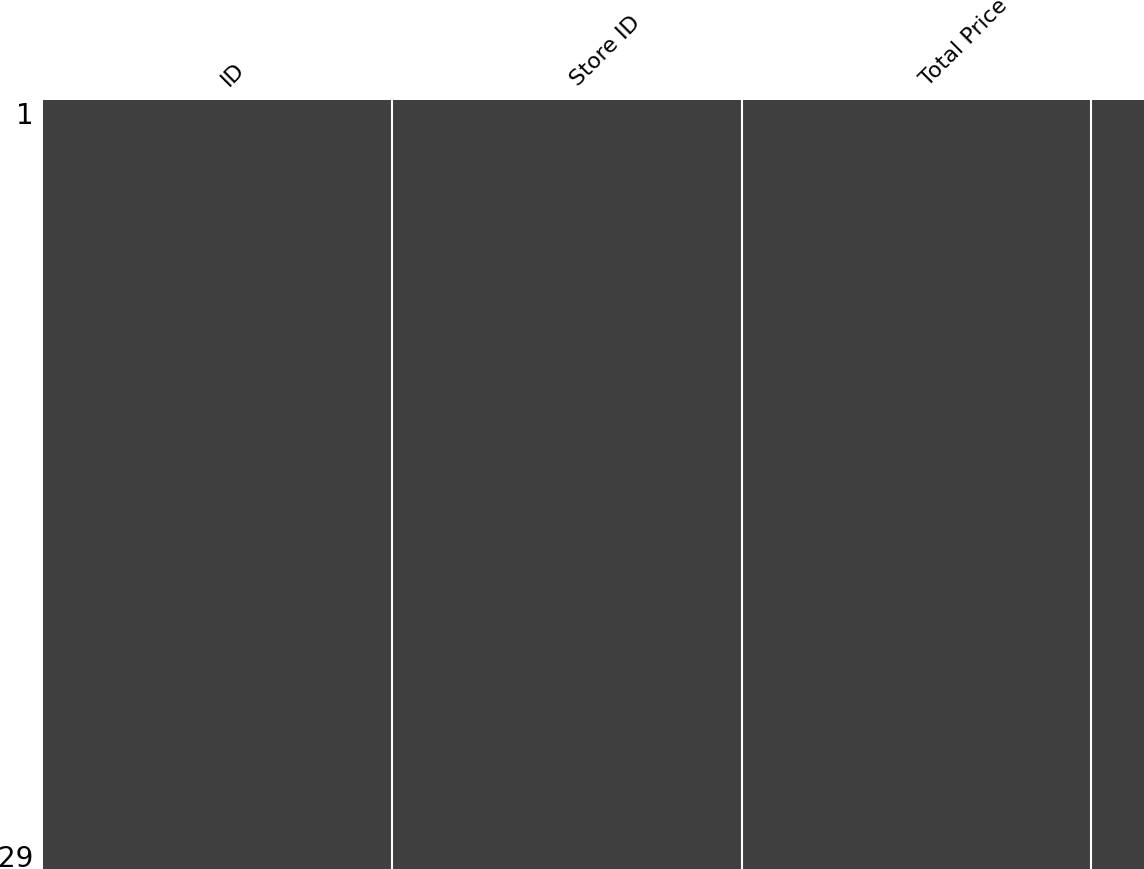


sns.barplot(dataset)

<Axes: >

import missingno as msno msno.matrix(dataset)

plt.figure(figsize=(15,9)) plt.show()



<Figure size 1500x900 with 0 Axes>