MAJOR PROJECT 1

Super Market Store Branches Sales Analysis

Name: Krati Bhat

Year: Third (2020-2024)

College: NMAMIT Institute Of Technology

Email: 4nm20is064@nmamit.in

GitHub Link: https://github.com/Krati-Bhat/Major-project-1.git

THE MI/AI MODEL

Python Code:

```
1)import pandas as pd

df=pd.read_csv('/content/Stores.csv')

df
```

Store ID	Store_Area	Items_Available	Daily_Customer_Count	Store_Sales	
0	1	1659	1961	530	66490
1	2	1461	1752	210	39820
2	3	1340	1609	720	54010
3	4	1451	1748	620	53730
4	5	1770	2111	450	46620
...	...	...	...	...	...
891	892	1582	1910	1080	66390
892	893	1387	1663	850	82080
893	894	1200	1436	1060	76440
894	895	1299	1560	770	96610
895	896	1174	1429	1110	54340

896 rows × 5 columns

```
2)df.info()

___<class 'pandas.core.frame.DataFrame'>
RangeIndex: 896 entries, 0 to 895
Data columns (total 5 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Store ID              896 non-null    int64
 1   Store_Area            896 non-null    int64
 2   Items_Available       896 non-null    int64
 3   Daily_Customer_Count  896 non-null    int64
 4   Store_Sales           896 non-null    int64
```

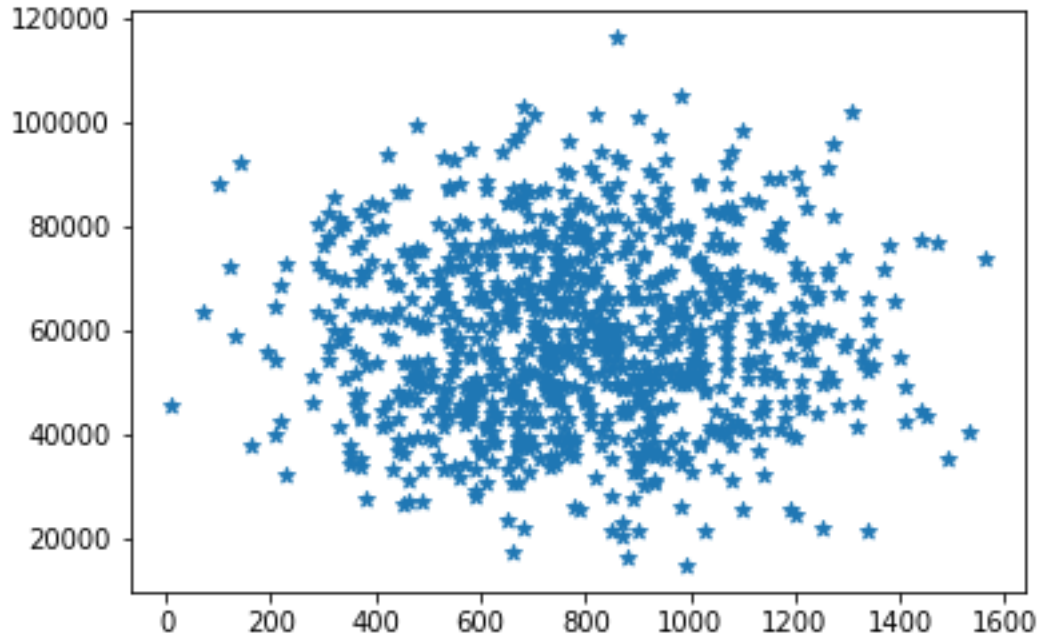
```
dtypes: int64(5)
memory usage: 35.1 K
```

3) `df.shape`

```
(896, 5)
```

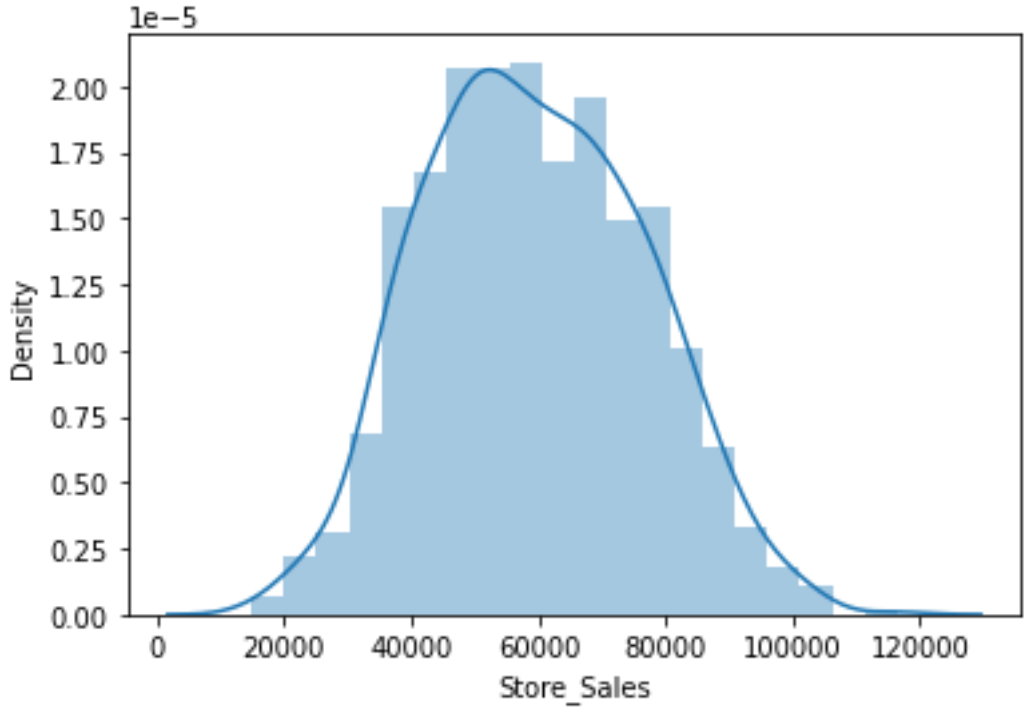
```
4)import matplotlib.pyplot as plt
plt.scatter(df['Daily_Customer_Count'],df['Store_Sales'],marker='*')
```

```
<matplotlib.collections.PathCollection at 0x7f1c057f0d50>
```




```
5) import seaborn as sns
sns.distplot(df['Store_Sales'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1c01991a50>
```




```
5)df_numeric = df_numeric.drop(['Store_Area'],axis = 1)
  df_numeric
```

```
6)df_numeric.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 896 entries, 0 to 895
Data columns (total 4 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Store ID              896 non-null    int64
 1   Items_Available       896 non-null    int64
 2   Daily_Customer_Count  896 non-null    int64
 3   Store_Sales           896 non-null    int64
dtypes: int64(4)
memory usage: 28.1 KB
```

```
7)x=df_numeric.iloc[:,1:3].values
  x
```

```
array([[1961, 530], [1752, 210], [1609, 720], ..., [1436, 1060], [1560,
770], [1429, 1110]])
```

```
8)y=df_numeric.iloc[:,3].values
  y
```

```
array([ 66490,  39820,  54010,  53730,  46620,  45260,  72240,  37720,
    46310,  44150,  71280,  57620,  60470,  59130,  66360,  78870,
    77250,  38170,  63540,  40190,  43460,  68890,  52780,  50680,
    41880,  70050,  25820,  60530,  78100,  84860,  80140,  14920,
    60460,  74560,  72430,  45460,  41570,  62870,  55170,  45480,
```

49550, 48140, 67640, 39730, 35800, 49270, 66510, 62530,
59980, 76350, 81820, 57830, 70450, 67000, 64090, 48670,
66210, 83660, 70770, 53870, 71300, 46100, 49100, 65920,
58660, 69130, 49080, 72710, 33430, 42430, 56650, 33580,
67370, 71780, 84840, 82070, 26770, 65560, 38660, 65660,
40700, 88910, 57860, 42670, 90180, 51280, 97260, 39650,
45720, 42060, 65350, 67080, 54030, 56360, 77120, 50810,
60960, 61180, 63660, 41190, 78420, 65580, 89080, 94170,
50950, 65180, 69310, 79210, 23740, 36330, 51700, 62950,
56010, 45080, 46830, 64750, 80780, 31180, 56710, 49390,
66000, 32770, 46580, 79780, 35510, 80970, 61150, 49210,
79950, 68740, 57480, 72630, 50070, 40490, 51850, 42840,
60940, 62280, 76530, 85130, 48590, 73080, 48950, 48560,
59380, 51190, 58920, 50360, 38070, 49170, 39740, 63730,
85330, 27410, 37320, 71120, 72800, 34410, 42530, 54300,
50780, 45020, 69600, 80340, 37810, 46140, 99570, 38650,
49800, 69910, 44910, 78470, 47460, 33460, 44090, 42620,
69450, 73120, 48300, 58090, 74250, 40930, 70930, 64670,
77420, 32330, 41080, 42860, 68450, 39730, 83750, 69940,
67710, 67360, 52460, 88760, 67030, 78230, 62270, 49760,
73660, 72320, 68890, 34180, 58260, 38120, 49070, 61660,
37830, 52270, 52280, 70810, 71530, 77260, 75550, 33730,
66270, 55820, 68430, 73990, 62800, 33740, 63830, 24410,
70020, 92240, 68230, 81870, 73860, 77120, 72350, 49160,
45650, 52780, 90960, 64950, 47230, 83250, 51950, 66030,
68590, 47140, 69940, 65890, 89310, 58540, 78130, 92300,
56170, 46050, 43390, 61750, 21830, 39800, 54370, 62470,
82930, 63720, 79180, 38210, 25950, 56610, 73710, 70400,
50440, 66840, 50170, 60140, 37130, 42890, 26220, 50840,
25630, 60770, 69600, 41740, 50130, 21750, 80490, 34020,
60240, 39460, 56440, 46840, 64820, 52520, 45940, 38970,
58610, 30810, 47730, 64640, 44860, 55660, 57530, 75200,
37330, 35280, 70610, 49720, 68850, 50740, 77070, 74730,
76530, 68900, 44960, 41490, 74320, 73800, 56400, 71570,
43640, 35120, 58670, 75800, 76640, 31890, 61670, 75950,
41490, 66170, 37980, 62710, 60470, 35230, 48550, 56810,
41080, 51270, 57500, 81470, 49570, 45580, 44660, 76240,
43720, 46700, 84690, 85470, 80410, 46650, 81840, 63590,
50130, 45820, 86710, 49980, 82940, 40350, 93950, 47310,
21300, 62180, 61460, 54360, 72050, 48090, 27310, 57160,
34190, 35600, 54670, 76730, 63540, 36690, 87370, 59800,
48030, 96650, 65500, 55530, 21650, 31940, 84690, 68390,
75490, 39200, 85670, 60530, 78090, 50720, 23090, 91360,
48120, 75620, 39420, 51130, 33890, 87170, 38600, 60980,
79410, 82350, 36740, 27720, 32260, 53270, 51480, 59970,
83600, 63020, 50920, 56450, 89540, 46030, 75110, 74520,
102310, 53400, 59760, 49540, 51560, 49510, 58610, 68260,
65310, 52090, 43860, 74170, 58380, 91200, 90940, 49330,
53500, 54590, 57450, 33240, 80790, 61000, 47620, 72090,
102920, 61970, 61040, 52060, 69570, 66020, 40000, 79500,
76300, 69030, 57140, 41710, 71480, 33010, 74570, 49590,
73170, 79220, 75880, 67610, 69090, 35220, 53940, 56660,
67520, 38620, 38890, 79270, 42880, 44240, 43190, 74550,
57090, 56480, 87410, 81370, 97360, 77960, 71240, 58940,
78950, 36380, 45160, 69050, 56830, 93530, 46920, 55990,
40840, 64990, 53550, 51320, 36560, 66050, 52400, 27970,
67100, 43710, 38600, 53890, 52610, 43130, 40300, 49750,
43840, 56820, 36350, 50820, 83720, 46970, 78020, 45080,
55160, 72020, 64010, 27840, 58070, 51760, 66050, 65750,

65820, 46760, 50940, 56440, 32610, 62770, 63600, 45840,
38280, 50960, 39480, 69610, 47800, 44890, 67420, 78870,
70310, 38530, 77570, 59920, 54450, 50250, 30790, 35420,
43470, 61000, 64780, 39030, 65900, 46050, 59070, 44670,
58390, 80370, 53230, 72000, 84040, 52540, 63510, 42240,
39580, 54610, 87330, 88410, 89760, 101780, 70290, 88210,
87160, 41540, 49170, 63950, 70810, 49590, 67290, 51240,
48540, 72410, 54370, 94460, 85160, 52130, 54650, 69320,
51480, 50060, 62180, 79780, 42860, 54410, 69390, 42810,
30840, 56260, 76470, 35680, 90070, 33120, 54060, 75120,
41600, 20270, 60060, 82270, 29170, 68420, 59130, 74330,
77080, 76250, 59540, 54690, 84360, 51420, 65120, 49380,
37830, 35980, 69190, 50590, 60800, 31180, 77790, 47570,
69130, 75970, 68350, 41680, 86560, 81390, 50730, 71290,
70110, 61590, 69370, 67110, 82020, 62050, 61730, 58660,
53370, 39700, 53750, 44730, 49350, 43340, 78090, 54950,
75530, 57330, 87930, 56850, 78430, 63660, 62960, 81870,
54820, 116320, 57200, 84360, 36530, 81260, 82350, 80830,
30610, 51310, 72940, 52450, 66070, 43190, 40730, 78530,
94690, 44400, 73800, 37390, 64120, 66160, 22310, 62380,
63850, 36210, 54590, 69610, 65390, 78130, 55710, 69210,
59940, 72550, 44260, 56910, 82390, 54590, 69990, 72740,
35360, 94370, 43520, 36000, 99480, 83220, 52940, 93360,
73590, 53840, 47350, 65080, 62050, 30020, 49510, 64320,
35590, 63050, 65300, 69560, 41910, 28330, 55980, 61080,
51380, 84410, 60680, 64690, 45780, 41800, 53230, 36160,
40450, 57910, 36280, 39190, 62380, 21470, 34610, 88120,
59190, 36290, 53760, 66300, 93000, 65660, 81930, 60060,
59530, 46380, 76200, 56860, 86620, 49730, 88370, 49160,
77740, 38560, 51990, 39970, 46040, 49500, 76670, 75800,
81720, 58440, 85720, 70940, 62420, 56880, 101820, 86890,
47300, 31270, 65410, 54200, 67390, 54530, 79760, 78060,
74080, 52990, 70580, 34310, 74160, 59190, 43370, 17670,
56710, 59820, 36190, 60440, 75300, 74080, 60440, 80720,
47060, 86830, 56790, 67090, 44370, 82970, 56230, 53760,
55390, 73500, 41050, 67320, 65890, 56380, 85670, 70830,
48180, 51910, 44320, 58940, 73610, 54060, 85000, 49030,
63300, 84300, 81390, 95900, 71830, 79310, 87890, 48610,
73160, 36280, 49720, 44400, 47590, 51460, 57750, 66000,
45950, 53900, 37920, 63100, 36770, 43910, 66390, 59160,
38510, 46220, 41500, 58160, 38530, 55880, 70940, 53940,
43030, 59820, 55500, 49990, 42980, 65970, 59290, 63020,
73810, 70230, 59950, 78100, 16370, 92640, 63540, 87220,
41990, 79410, 54380, 58600, 48950, 40670, 52340, 39140,
41090, 25600, 100900, 77080, 105150, 80580, 46230, 98260,
75930, 52050, 87000, 60270, 88270, 57820, 61210, 76420,
70980, 76740, 47920, 52160, 32740, 72270, 77430, 92370,
34880, 46580, 70620, 66390, 82080, 76440, 96610, 54340])

```
9)from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)

10)print(x.shape)
   print(x_train.shape)
   print(x_test.shape)
```
(896, 2)

(672, 2)
(224, 2)

0s

```
11)print(y.shape)
   print(y_train.shape)
   print(y_test.shape)
```

```
   (896,)

   (672,)

   (224,)
```

```
12)from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
x_train=scaler.fit_transform(x_train)
x_test=scaler.fit_transform(x_test)
13)from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(x_train,y_train)
```

LinearRegression()

```
14)y_pred=model.predict(x_test)
   y_pred
array([60884.63734115, 60748.8419685 , 61283.87956987, 59569.38122601,
62655.72059231, 63366.01020579, 60758.48875067, 61848.15168905,
60142.27250671, 62603.57934829, 62451.19203792, 59729.13695176,
59443.75294743, 59694.05292976, 58036.11047186, 60502.05809597,
59172.66951091, 60539.5045337 , 57745.4392997 , 58277.55214897,
60329.8148849 , 62094.57034446, 59538.95098966, 59259.5295598 ,
58508.99482992, 59876.15304536, 59993.04909214, 60411.75298309,
59216.59883501, 59481.59362363, 62862.24634532, 62850.67341036,
61067.29979311, 59700.42274584, 62648.87248882, 59746.71448567,
59665.66191649, 61832.93657088, 60424.84482924, 59356.934923 ,
60072.42765536, 57522.72885057, 60166.58507382, 59578.63376971,
59988.88659703, 59767.5819888 , 60571.94497179, 63260.08273314,
56343.71737414, 61850.68220267, 60187.24245459, 61984.35430329,
62139.58231681, 58768.29520912, 61563.56916479, 61593.92835532,
58303.22853251, 60154.32372908, 58130.09979243, 59415.64310233,
62170.70096294, 60286.84213563, 60278.82733495, 62922.36036555,
62056.44850008, 63645.54470594, 60764.29623039, 63100.73424899,
58764.77909932, 60198.89943849, 57955.41011668, 61500.74051482,
60221.08873357, 62212.22584684, 58499.22197433, 61739.66468143,
60974.70331032, 61491.57202006, 62899.17247116, 58988.20697957,
60335.81948385, 61532.73168684, 61342.91094186, 61319.16071111,
58513.24137397, 60248.35507412, 57288.37442065, 60760.80914194,
58365.10060765, 62158.8468598 , 60472.10614703, 63254.64047055,
57272.71003641, 60483.07472116, 57059.29415544, 62693.77139088,
63030.73430287, 62004.46235082, 61897.60732469, 61721.35671328,
58311.31437902, 60693.08756261, 60071.90734347, 61939.86264283,
59560.53592391, 59431.21043451, 63001.17659241, 63020.72230358,
```

```
61361.61314849, 60861.5334957 , 58251.00323955, 60960.52881592,
61712.35631641, 60810.10968281, 56904.58645381, 60622.28613643,
61382.07341002, 59759.89038077, 60558.52993298, 58129.29831236,
60076.548126 , 57773.11288185, 59192.89062873, 59557.09085993,
60817.39404924, 63325.72306491, 61253.67547411, 62331.41326353,
60192.81079059, 59007.51354704, 60651.1974616 , 59498.81894354,
63088.40185843, 60380.99955409, 60520.08489595, 58532.22474878,
62719.99710765, 60242.39249964, 58413.9228611 , 61557.56456583,
56970.50795376, 59280.59418217, 60605.34198471, 61224.20181259,
61103.1432707 , 58041.51070998, 59490.93021627, 61136.0619962 ,
60450.84440543, 60192.81079059, 58709.43193502, 58836.94434192,
58380.11860659, 61145.63773255, 61997.7693421 , 60068.50430396,
58267.75027204, 59918.04314638, 59983.12114179, 61254.0827157 ,
60589.04421828, 59249.23639233, 60753.48275103, 60791.64661988,
59844.75323105, 61832.6554027 , 60453.36191593, 59526.40847675,
60039.18573721, 59509.02806207, 59278.39986432, 60664.57047593,
61221.68430209, 61528.37207249, 59638.62171653, 64052.35246928,
60511.9150005 , 60088.80947073, 59436.3004831 , 59085.28915012,
62234.21802268, 59497.77831976, 61346.83429325, 58918.16500899,
60453.88222782, 61396.61312154, 59244.15934686, 63489.12097388,
58350.08260871, 59079.31357252, 60037.66682602, 61134.4170116 ,
61730.49618668, 60266.62101781, 59999.61602745, 58330.66297095,
59967.82197467, 60355.40721949, 58951.84319009, 63847.86593936,
56948.72590027, 60928.77678761, 60602.54330603, 59008.79331452,
59412.11398941, 61499.85498581, 59121.7660099 , 57362.86005452,
59149.39756757, 60322.0812524 , 60130.89669099, 60778.7098685 ,
58857.81184505, 57466.55118485, 63531.0110749 , 60522.08209456,
60773.18355696, 60645.755199 , 62096.04723119, 60507.71048092,
58840.19228667, 61028.77070713, 60956.32429634, 61294.28580764])
```

```
15)print(x_train[11])
```

```
[0.25227411 0.42465753]
```

```
16) model.predict([x_train[10]])
```

```
    array([58781.96090274])
```

```
17)print(x_train[1])
   [0.53911461 0.41780822]
18)model.predict([x_train[1]])
   array([59910.15646053])
```

Details:-

1)In step 1 we have to import pandas in order to read our dataset and create a dataframe

2)Then we have to perform Data Visualization

3)Divide the dataset into Input and Output

4) Use Train and Test Variables

5)Normalize the data

6)Run a regressor

7) Fit the model

8)Predict the model

9)Evaluate the output

GitHub Link:- https://github.com/Krati-Bhat/Major-project-1.git