Number of Orders Prediction

In [ ]:
```
!pip install lightgbm
```

```
Requirement already satisfied: lightgbm in c:\users\armut\anaconda3\lib\site-packages (4.1.0)
Requirement already satisfied: numpy in c:\users\armut\anaconda3\lib\site-packages (from lightgbm) (1.24.3)
Requirement already satisfied: scipy in c:\users\armut\anaconda3\lib\site-packages (from lightgbm) (1.10.1)
```

STEP 1- Import Dataset

In [ ]:
```
#import libraries
import pandas as pd
import numpy as np
```

In [ ]:
```
dataset = pd.read_csv("OS_Regional_Sales_Data.csv")
dataset
```

Out[ ]:

	OrderNumber	Sales Channel	WarehouseCode	ProcuredDate	OrderDate	ShipDate	DeliveryDate	CurrencyCode	_SalesTeamID	_CustomerID	_StoreID	_ProductID	Order Quantity	Discount Applied	
0	SO - 000101	In-Store	WARE-UHY1004	31/12/17	31/5/18	14/6/18	19/6/18	USD	6	15	259	12	5	0.075	1
1	SO - 000102	Online	WARE-NMK1003	31/12/17	31/5/18	22/6/18	27/7/18	USD	14	20	196	27	3	0.075	1
2	SO - 000103	Distributor	WARE-UHY1004	31/12/17	31/5/18	21/6/18	1/7/18	USD	21	16	213	16	1	0.050	
3	SO - 000104	Wholesale	WARE-NMK1003	31/12/17	31/5/18	2/6/18	7/6/18	USD	28	48	107	23	8	0.075	1
4	SO - 000105	Distributor	WARE-NMK1003	10/4/18	31/5/18	16/6/18	26/6/18	USD	22	49	111	26	8	0.100	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7986	SO - 0008087	In-Store	WARE-MKL1006	26/9/20	30/12/20	7/1/21	14/1/21	USD	9	41	339	29	1	0.075	
7987	SO - 0008088	Online	WARE-NMK1003	26/9/20	30/12/20	2/1/21	4/1/21	USD	14	29	202	3	6	0.050	1
7988	SO - 0008089	Online	WARE-UHY1004	26/9/20	30/12/20	23/1/21	26/1/21	USD	14	32	241	35	5	0.200	2
7989	SO - 0008090	Online	WARE-NMK1003	26/9/20	30/12/20	20/1/21	25/1/21	USD	20	42	112	36	8	0.100	
7990	SO - 0008091	In-Store	WARE-UHY1004	26/9/20	30/12/20	13/1/21	19/1/21	USD	6	41	237	43	5	0.075	1

7991 rows × 16 columns

STEP 2:- Data Inspection and Data Handling

In [ ]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7991 entries, 0 to 7990
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   OrderNumber      7991 non-null   object 
 1   Sales Channel    7991 non-null   object 
 2   WarehouseCode    7991 non-null   object 
 3   ProcuredDate     7991 non-null   object 
 4   OrderDate        7991 non-null   object 
 5   ShipDate         7991 non-null   object 
 6   DeliveryDate     7991 non-null   object 
 7   CurrencyCode     7991 non-null   object 
 8   _SalesTeamID     7991 non-null   int64  
 9   _CustomerID      7991 non-null   int64  
 10  _StoreID         7991 non-null   int64  
 11  _ProductID       7991 non-null   int64  
 12  Order Quantity   7991 non-null   int64  
 13  Discount Applied 7991 non-null   float64
 14  Unit Cost        7991 non-null   object 
 15  Unit Price       7991 non-null   object 
dtypes: float64(1), int64(5), object(10)
memory usage: 999.0+ KB
```

In [ ]:
```
dataset.isnull().sum()
```

Out[ ]:
```
OrderNumber        0
Sales Channel      0
WarehouseCode      0
ProcuredDate       0
OrderDate          0
ShipDate           0
DeliveryDate       0
CurrencyCode       0
_SalesTeamID       0
_CustomerID        0
_StoreID           0
_ProductID         0
Order Quantity     0
Discount Applied   0
Unit Cost          0
Unit Price         0
dtype: int64
```

STEP 3:- Data Visualization

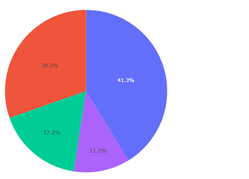
In [ ]:
```
!pip install plotly
```

```
Requirement already satisfied: plotly in c:\users\armut\anaconda3\lib\site-packages (5.9.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\armut\anaconda3\lib\site-packages (from plotly) (8.2.2)
```

In [ ]:
```
import plotly.express as px
```

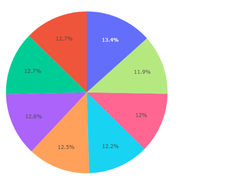
In [ ]:
```
pie1 = dataset["Sales Channel"].value_counts()
sales = pie1.index
orders = pie1.values

fig = px.pie(dataset, values=orders, names=sales)
fig.show()
```



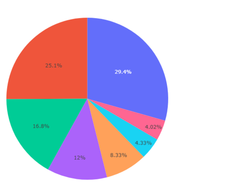
In [ ]:
```
pie2 = dataset["Order Quantity"].value_counts()
order = pie2.index
orders = pie2.values

fig = px.pie(dataset, values=orders, names=order)
fig.show()
```



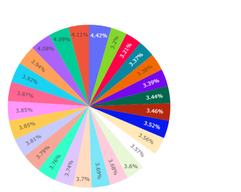
In [ ]:
```
pie3 = dataset["Discount Applied"].value_counts()
discount = pie3.index
orders = pie3.values

fig = px.pie(dataset, values=orders, names=discount)
fig.show()
```



In [ ]:
```
pie4 = dataset["_SalesTeamID"].value_counts()
sales = pie4.index
orders = pie4.values

fig = px.pie(dataset, values=orders, names=sales)
fig.show()
```



STEP 4:- Data Preprocessing

In [ ]:
```
dataset["Sales Channel"]=dataset["Sales Channel"].map({"In-Store":1, "Online":2, "Distributor":3, "Wholesale":4})
```

In [ ]:
```
dataset
```

Out[ ]:

	OrderNumber	Sales Channel	WarehouseCode	ProcuredDate	OrderDate	ShipDate	DeliveryDate	CurrencyCode	_SalesTeamID	_CustomerID	_StoreID	_ProductID	Order Quantity	Discount Applied	
0	SO - 000101	1	WARE-UHY1004	31/12/17	31/5/18	14/6/18	19/6/18	USD	6	15	259	12	5	0.075	1
1	SO - 000102	2	WARE-NMK1003	31/12/17	31/5/18	22/6/18	27/7/18	USD	14	20	196	27	3	0.075	1
2	SO - 000103	3	WARE-UHY1004	31/12/17	31/5/18	21/6/18	1/7/18	USD	21	16	213	16	1	0.050	
3	SO - 000104	4	WARE-NMK1003	31/12/17	31/5/18	2/6/18	7/6/18	USD	28	48	107	23	8	0.075	1
4	SO - 000105	3	WARE-NMK1003	10/4/18	31/5/18	16/6/18	26/6/18	USD	22	49	111	26	8	0.100	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7986	SO - 0008087	1	WARE-MKL1006	26/9/20	30/12/20	7/1/21	14/1/21	USD	9	41	339	29	1	0.075	
7987	SO - 0008088	2	WARE-NMK1003	26/9/20	30/12/20	2/1/21	4/1/21	USD	14	29	202	3	6	0.050	1
7988	SO - 0008089	2	WARE-UHY1004	26/9/20	30/12/20	23/1/21	26/1/21	USD	14	32	241	35	5	0.200	2
7989	SO - 0008090	2	WARE-NMK1003	26/9/20	30/12/20	20/1/21	25/1/21	USD	20	42	112	36	8	0.100	
7990	SO - 0008091	1	WARE-UHY1004	26/9/20	30/12/20	13/1/21	19/1/21	USD	6	41	237	43	5	0.075	1

7991 rows × 16 columns

In [ ]:
```
X = np.array(dataset[["Sales Channel", "_SalesTeamID", "_ProductID", "Order Quantity"]])
y = np.array(dataset["_StoreID"])
```

In [ ]:
```
X
```

Out[ ]:
```
array([[ 1,  6, 12,  5],
       [ 2, 14, 27,  3],
       [ 3, 21, 16,  1],
       ...,
       [ 2, 14, 35,  5],
       [ 2, 20, 36,  8],
       [ 1,  6, 43,  5]], dtype=int64)
```

In [ ]:
```
y
```

Out[ ]:
```
array([259, 196, 213, ..., 241, 112, 237], dtype=int64)
```

In [ ]:
```
from sklearn.model_selection import train_test_split
```

In [ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=10)
```

In [ ]:
```
len(X_train)
```

Out[ ]:
```
5593
```

STEP 5:- Building the ML Model

In [ ]:
```
import lightgbm as ltb
model = ltb.LGBMRegressor()
```

In [ ]:
```
model.fit(X_train, y_train)
```

```
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.000469 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 91
[LightGBM] [Info] Number of data points in the train set: 5593, number of used features: 4
[LightGBM] [Info] Start training from score 182.455563
```

Out[ ]:
```
LGBMRegressor()
```

In [ ]:
```
y_pred = model.predict(X_test)
```

In [ ]:
```
y_pred
```

Out[ ]:
```
array([218.45676241, 170.33840693, 177.4299488 , ..., 179.2994573 ,
       206.52673302, 181.20513479])
```

In [ ]:
```
y_test
```

Out[ ]:
```
array([ 48, 284, 200, ..., 337, 352, 247], dtype=int64)
```

In [ ]:
```
data = pd.DataFrame(data={"Predicted Orders": y_pred.flatten()})
```

In [ ]:
```
data
```

Out[ ]:

	Predicted Orders
0	218.456762
1	170.338407
2	177.429949
3	221.668841
4	173.667053
...	...
2393	176.952095
2394	199.416563
2395	179.299457
2396	206.526733
2397	181.205135

2398 rows × 1 columns