وزارة الاتصـــــالات
وتكنولوجيا المعلومات

Sales Forecasting and Demand Prediction

رواد مصر الرقمية

AI & DATA SCIENCE TRACK

PROJECT OVERVIEW

Objective Build a predictive model to forecast future sales and demand using historical and external data.

Business Value Supports better inventory, staffing, and marketing decisions.

رواد مصر الرقمية

INTRODUCTION

The aim of this project is to develop a robust machine learning model that accurately forecasts future sales and product demand using historical data and external factors. Doing so, we help businesses make smarter, data-driven decisions to optimize inventory, improve demand planning, and enhance overall operational efficiency.

PRACTICAL BENEFITS OF THE PROJECT

- Supports data-driven decision making
- Reduces costs caused by poor estimation
- Enhances the competitive advantage of companies
- Assists in both short- and long-term strategic planning, helping to reduce waste, save time and money, and improve customer service


PROBLEM
SOLUTION

01 Operational Inefficiency

- Overstock, stockouts, poor shift planning, and supply chain delays lead to wasted resources and lost revenue.

✅ Our model forecasts product-level demand, enabling just-in-time inventory, optimized staffing, and coordinated vendor planning.

02 Uninformed Decision-Making

- Marketing budgets are misallocated; financial forecasts are based on outdated assumptions.

✅ We use data-driven demand predictions to guide marketing timing, revenue planning, and strategic decisions.

03 Reliance on Manual Planning

- Traditional methods lack scalability and adaptability to dynamic market changes.

✅ By integrating AI/ML, we automate forecasting and planning, helping businesses become more agile and future-ready.

Dataset Summary

رواد مصر الرقمية

Sources: Kaggle

Features:

Product, Customer, Region, Order Date, Sales, Profit, etc.

data.head()

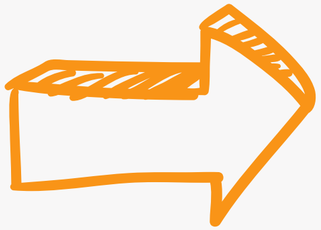
Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Postal Code	City	...	Product ID	Category	Sub-Category	Product Name	Sales	Quantity	Discount
CA-2014-AB10015140-41954	2014-11-11	2014-11-13	First Class	AB-100151402	Aaron Bergman	Consumer	73120.0	Oklahoma City	...	TEC-PH-5816	Technology	Phones	Samsung Convoy 3	221.980	2	0.0
IN-2014-JR162107-41675	2014-02-05	2014-02-07	Second Class	JR-162107	Justin Ritter	Corporate	NaN	Wollongong	...	FUR-CH-5379	Furniture	Chairs	Novimex Executive Leather Armchair, Black	3709.395	9	0.1
IN-2014-CR127307-41929	2014-10-17	2014-10-18	First Class	CR-127307	Craig Reiter	Consumer	NaN	Brisbane	...	TEC-PH-5356	Technology	Phones	Nokia Smart Phone, with Caller ID	5175.171	9	0.1
ES-2014-KM1637548-41667	2014-01-28	2014-01-30	First Class	KM-1637548	Katherine Murray	Home Office	NaN	Berlin	...	TEC-PH-5267	Technology	Phones	Motorola Smart Phone, Cordless	2892.510	5	0.1

PREPROCESSING & FEATURE ENGINEERING

```python
for col in numerical_columns:
    q1 = data[col].quantile(0.25)
    q3 = data[col].quantile(0.75)
    iqr = q3 - q1
    low = q1 - 1.5 * iqr
    up = q3 + 1.5 * iqr
    data[col] = data[col].clip(lower=low, upper=up)
```

Removing outliers from training data

```
Outliers in Row ID: 0
Outliers in Sales: 5655
Outliers in Quantity: 877
Outliers in Discount: 4172
Outliers in Profit: 9755
Outliers in Shipping Cost: 5909
```



```
Outliers in Row ID: 0
Outliers in Sales: 0
Outliers in Quantity: 0
Outliers in Discount: 0
Outliers in Profit: 0
Outliers in Shipping Cost: 0
```

رواد مصر الرقمية

EXTRACTING YEAR, MONTH, DAY, ECONOMIC INDICATORS

```
if 'Order Date' in data.columns:
    data['Order Date'] = pd.to_datetime(data['Order Date'])

    data['Year'] = data['Order Date'].dt.year
    data['Month'] = data['Order Date'].dt.month
    data['Weekday'] = data['Order Date'].dt.day_name()
    data['IsWeekend'] = data['Weekday'].isin(['Saturday', 'Sunday'])
```

```
data['Season'] = data['Month'].apply(lambda x: 'Winter' if x in [12, 1, 2] else
                                     'Spring' if x in [3, 4, 5] else
                                     'Summer' if x in [6, 7, 8] else
                                     'Fall')
```

```
data[['Order Date', 'Year', 'Month', 'Weekday', 'IsWeekend', 'Season']].head()
```

	Order Date	Year	Month	Weekday	IsWeekend	Season
0	2014-11-11	2014	11	Tuesday	False	Fall
1	2014-02-05	2014	2	Wednesday	False	Winter
2	2014-10-17	2014	10	Friday	False	Fall
3	2014-01-28	2014	1	Tuesday	False	Winter
4	2014-11-05	2014	11	Wednesday	False	Fall

EXTRACTING PROMOTION FLAG, DISCOUNTCATEGORY

```
data['PromotionFlag'] = data['Discount'].apply(lambda x: 1 if x > 0 else 0)
data['DiscountCategory'] = pd.cut(
    data['Discount'],
    bins=[-1, 0, 5, 15, 100],
    labels=['No Discount', 'Low', 'Medium', 'High']
)
```

```
data[['Discount', 'PromotionFlag', 'DiscountCategory']].head()
```

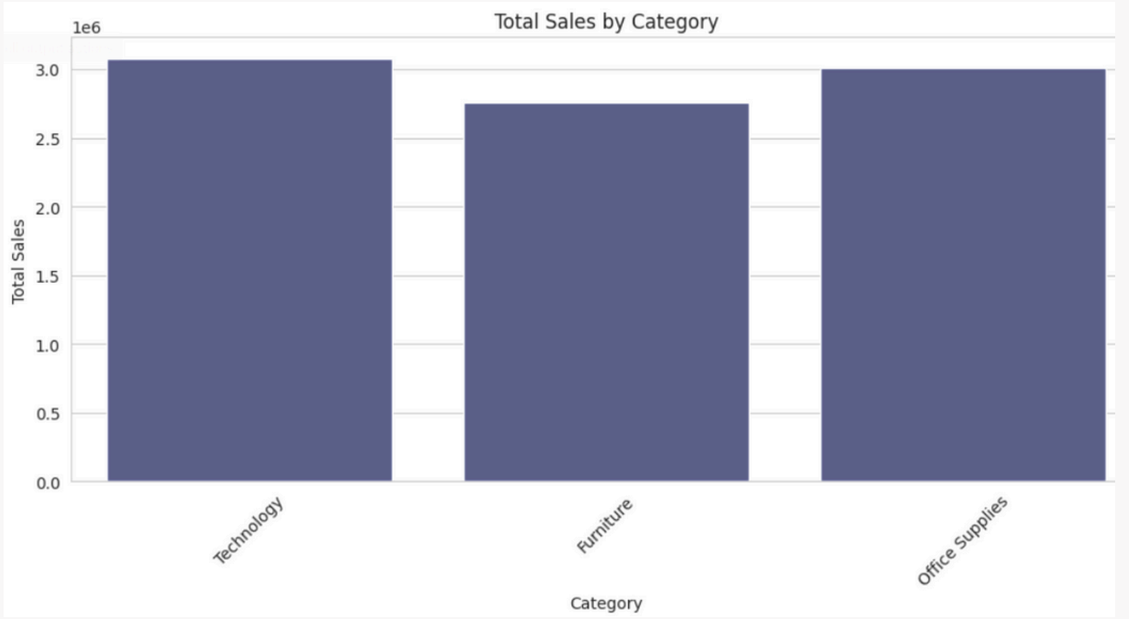
	Discount	PromotionFlag	DiscountCategory
0	0.0	0	No Discount
1	0.1	1	Low
2	0.1	1	Low
3	0.1	1	Low
4	0.0	0	No Discount

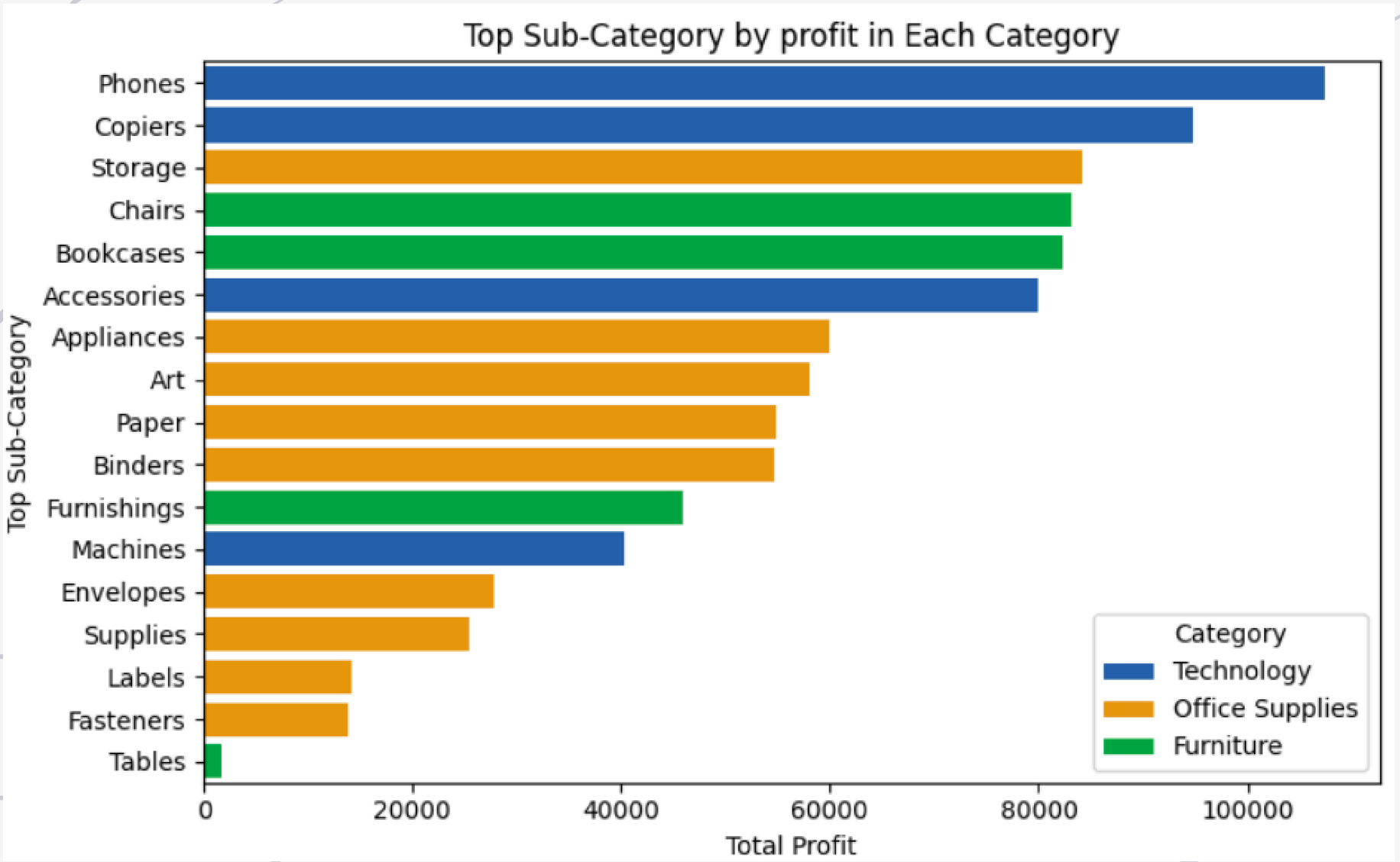
DATA EXPLORATION

EDA

SALES VS PROFIT

The analysis indicates that the Office Supplies category is the most successful in the store in terms of both sales and profit, making it an ideal candidate for business expansion and enhanced marketing campaigns

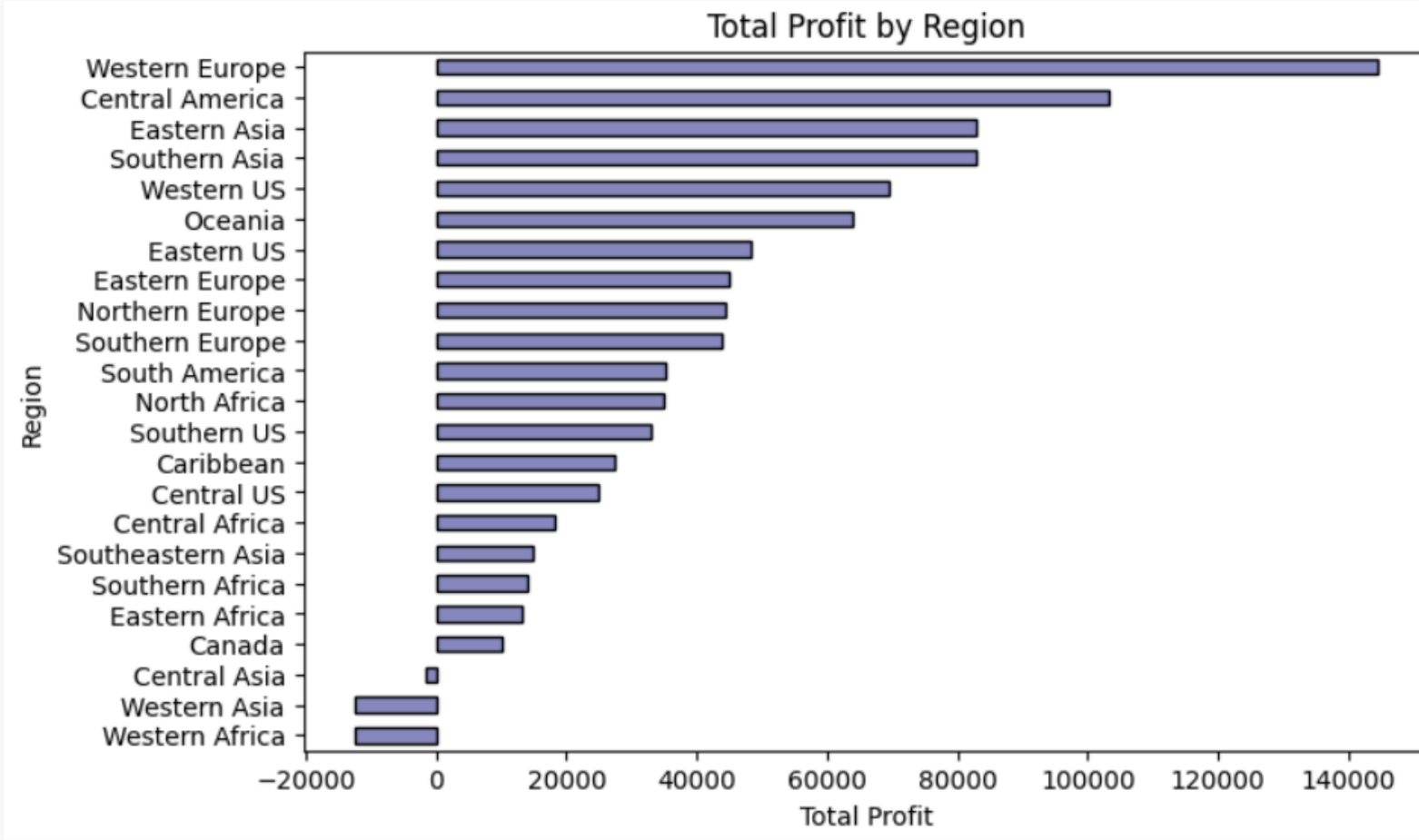




رواد مصر الرقمية

NEGATIVE PROFIT

The analysis reveals that some regions, such as Western Asia and Western Africa, are operating at a loss. This suggests the need for a closer look at operational efficiency, pricing, or customer behavior in those areas



TIME SERIES

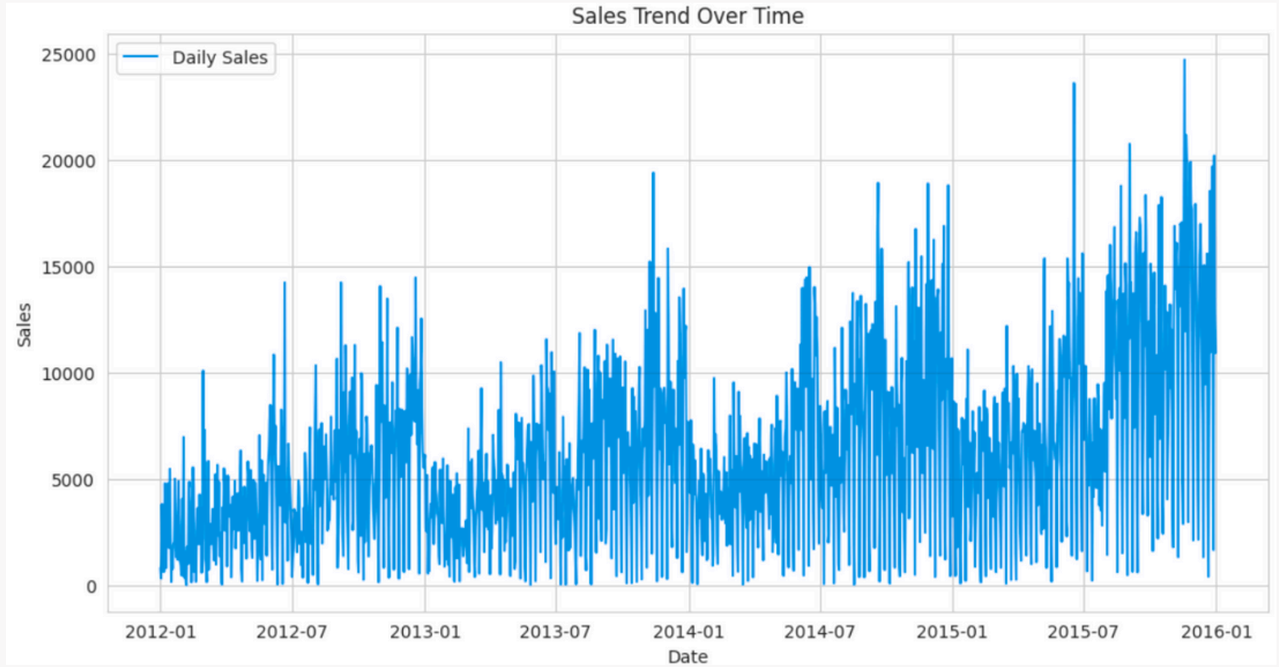
We wanted to predict future sales and demand based on past performance.

```
data['Order Date'] = pd.to_datetime(data['Order Date'])
data.groupby("Order Date")["Sales"].sum().reset_index()
data["ds"] = data['Order Date']
data["y"] = data["Sales"].astype(float)
data.head()
```

	Order Date	Sales	ds	y
0	2014-11-11	221.980000	2014-11-11	221.980000
1	2014-02-05	581.495063	2014-02-05	581.495063
2	2014-10-17	581.495063	2014-10-17	581.495063
3	2014-01-28	581.495063	2014-01-28	581.495063
4	2014-11-05	581.495063	2014-11-05	581.495063

We needed a model that can:

- Understand chronological order.
- Detect and learn from seasonal trends.
- Make accurate future forecasts.



FEATURE SELECTION

the process of choosing the most relevant variables (features) in a dataset that significantly influence the target outcome

```
prod_info = data[['Product ID', 'Sub-Category', 'Category']].drop_duplicates()
```

```
merged = filtered.merge(prod_info , on='Product ID',how='left')

merged
```

	Order Date	Segment	Country	Product ID	PromotionFlag	Sales	Sub-Category	Category
0	2012-01-01	Consumer	Algeria	OFF-ST-6261	0	408.300	Storage	Office Supplies
1	2012-01-01	Consumer	Australia	FUR-FU-4075	1	113.670	Furnishings	Furniture
2	2012-01-01	Consumer	Australia	OFF-PA-3990	1	55.242	Paper	Office Supplies
3	2012-01-01	Consumer	Australia	OFF-SU-3002	1	120.366	Supplies	Office Supplies
4	2012-01-01	Consumer	Hungary	OFF-ST-6230	0	66.120	Storage	Office Supplies

ENCODING CATEGORICAL DATA

- ML models require numeric input.
- Categorical variables like "Segment" or "Season" can't be used directly.
- Encoding transforms categories into machine-readable format.

OneHotEncoder

Creates binary columns for each category

LabelEncoder

Assigns a unique number to each category

OneHotEncoder

```
encoder = OneHotEncoder(sparse_output=False) # Use sparse_output instead of sparse

encoded = encoder.fit_transform(merged[['Segment', 'Sub-Category', 'Category', 'Season']])
encoded_df = pd.DataFrame(encoded, columns=encoder.get_feature_names_out(['Segment', 'Sub-Category', 'Category', 'Season']))
data_encoded = pd.concat([merged[['Order Date', 'Year', 'Month', 'Weekday', 'IsWeekend', 'Sales','Country', 'PromotionFlag', 'Product ID']], encoded_df], axis=1)
```

LabelEncoder

```
le = LabelEncoder()

data_encoded['Country'] = le.fit_transform(data_encoded['Country'])
data_encoded['Product ID'] = le.fit_transform(data_encoded['Product ID'])
```

The result data

	Order Date	Year	Month	Weekday	IsWeekend	Sales	Country	PromotionFlag	Product ID	Segment_Consumer	...	Sub-Category_Storage	Sub-Category_Supplies	Sub-Category_Tables	Category_Furniture
0	2012-01-01	2012	1	6	True	408.300	2	0	2732	1.0	...	1.0	0.0	0.0	0.0
1	2012-01-01	2012	1	6	True	113.670	6	1	513	1.0	...	0.0	0.0	0.0	1.0
2	2012-01-01	2012	1	6	True	55.242	6	1	2151	1.0	...	0.0	0.0	0.0	0.0
3	2012-01-01	2012	1	6	True	120.366	6	1	2801	1.0	...	0.0	1.0	0.0	0.0
4	2012-01-01	2012	1	6	True	66.120	65	0	2722	1.0	...	1.0	0.0	0.0	0.0

choose model

Gradient Boosting Regressor

- Using all features
- RMSE: 133.52
- MAE: 97.37
- R2: 51.3%

Random Forest Regressor

- Using all features
- RMSE: 123.56
- MAE: 88.94
- R2: 58%

Arima

- Using date only
- RMSE: 2862.68
- MAE: 2231.45
- R2: 55.3%

Prophet

- Using date only
- RMSE: 36139.98
- MAE: 28828.36
- R2: 81%

رواد مصر الرقمية

RANDOMIZED SEARCH

Gradient Boosting Regressor

```
gb = GradientBoostingRegressor(random_state=42)
gb_param_grid = {
    "n_estimators": [100, 200],
    "learning_rate": [0.01, 0.05, 0.1],
    "max_depth": [3, 5, 7],
    "min_samples_split": [2, 5],
    "min_samples_leaf": [1, 2],
    "subsample": [0.8, 1.0]
}
rs_gb = RandomizedSearchCV(
    gb, gb_param_grid, n_iter=20, cv=tscv, n_jobs=-1,
    scoring='neg_mean_squared_error', random_state=42
)
```

Random Forest Regressor

```
rf = RandomForestRegressor(random_state=42)
rf_param_grid = {
    "n_estimators": [100, 200, 300],
    "max_depth": [10, 20, None],
    "min_samples_split": [2, 5, 10],
    "min_samples_leaf": [1, 2, 4],
    "max_features": ["sqrt", "log2"]
}
rs_rf = RandomizedSearchCV(
    rf, rf_param_grid, n_iter=20, cv=tscv, n_jobs=-1,
    scoring='neg_mean_squared_error', random_state=42
)
```

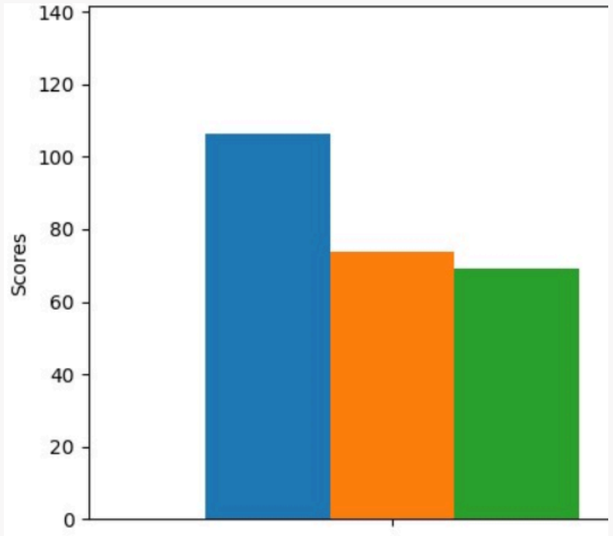
THE BEST MODEL

Gradient Boosting Regressor

RMSE
106.49

MAE
73.92

R2
68.9%

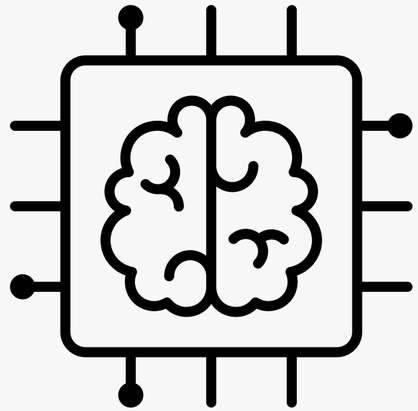



```python
gb = GradientBoostingRegressor(random_state=42)
gb_param_grid = {
    "n_estimators": [100, 200,300],
    "learning_rate": [0.01, 0.05, 0.1],
    "max_depth": [3, 5, 7],
    "min_samples_split": [2, 4],
    "min_samples_leaf": [1, 2],
    "subsample": [0.8, 1.0],
    "max_features": ["sqrt", "log2", None]
}

rs_gb = RandomizedSearchCV(
    gb,
    gb_param_grid,
    n_iter=25,
    cv=5,
    n_jobs=-1,
    scoring='neg_mean_squared_error',
    random_state=42,
    verbose=1
)
```

DEPLOYMENT

MODEL DRIFT



Since the data is static and no new inputs are expected, drift monitoring has not been implemented

MODEL RETRAINING STRATEGY

- **Current Status:**

  As of deployment, no new data has been added. Therefore, retraining is not scheduled.
- **Planned Strategy (If Needed):**
  - Periodic data checks (monthly/quarterly)
  - Trigger retraining based on:
    - Drop in forecast accuracy
    - Introduction of new seasonal trends or products

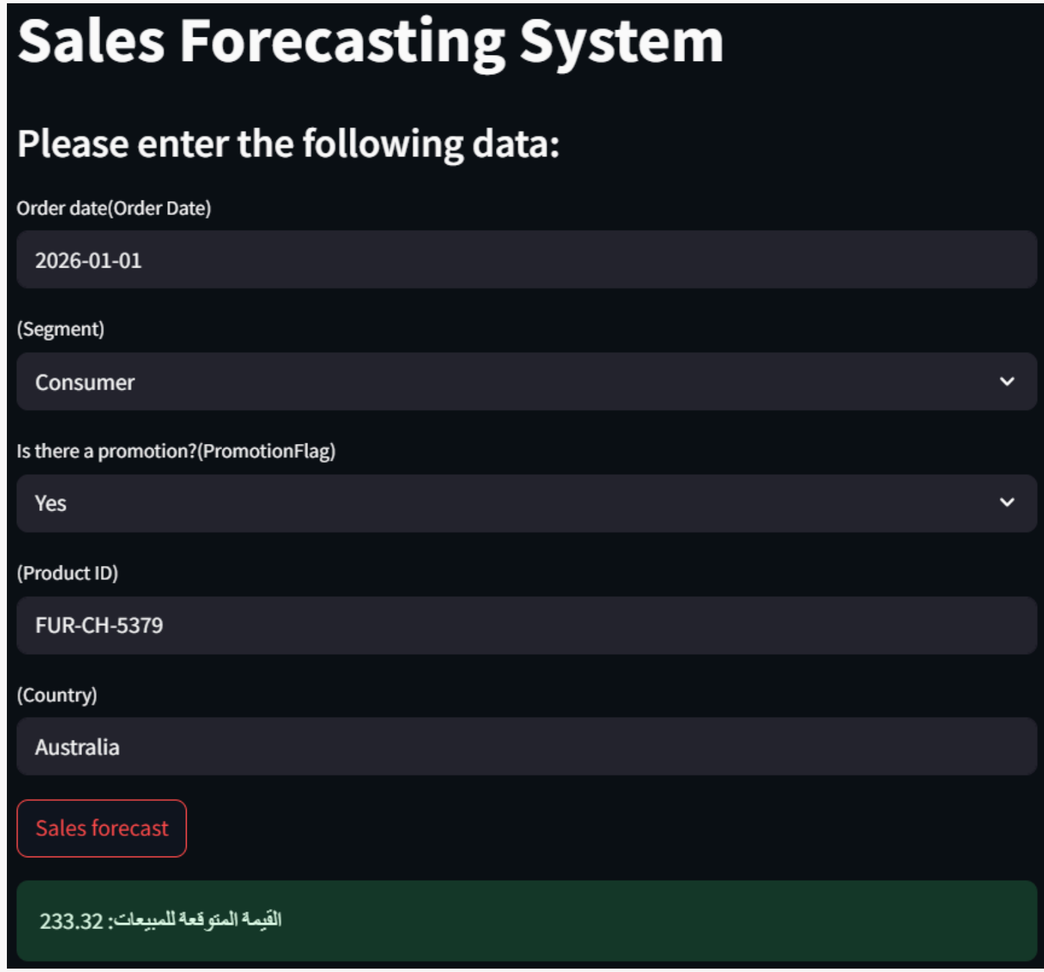
DEPLOYING WITH STREAMLIT

Streamlit is an open-source Python library that makes it easy to create and share interactive web apps for data science

WHY STREAMLIT?

- Lets users select a date range.
- Optionally adjust features (e.g., segment, category, promotions).
- Displays sales forecast results in real time.

MODEL **DEPLOYMENT**



THANK YOU