Phase-3 Submission Template

Student Name: DURGADEVI.S
Register Number:31223U09020
Institution: Islamiah Women's Arts And Science College,Vaniyambadi
Department: BCA
Date of Submission: 19/01/2026
Github Repository Link: https://github.com/durgadevishanmugam05-code/Restaurant-Sales-Data-Analysis-and-Insights.git

1. Problem Statement

The restaurant industry generates large volumes of sales data daily, but most businesses fail to utilize this data effectively for decision-making. The problem addressed in this project is the lack of data-driven insights for understanding sales performance, customer demand, and pricing strategies. This project aims to analyze restaurant sales data to identify trends, patterns, and key influencing factors such as promotions, weather, events, and ingredient costs.
Machine Learning techniques are applied to **predict selling prices (regression)** and **classify sales performance (classification)**. The solution helps restaurants optimize pricing, improve promotions, and enhance operational efficiency.

Problem Type:

- Regression (Predicting actual selling price)
- Classification (High vs Low sales category)

2. Abstract

This project focuses on analyzing restaurant sales data to extract meaningful insights and build predictive models. The objective is to understand how factors such as promotions, weather conditions, events, and ingredient costs impact restaurant sales. The dataset was cleaned and preprocessed to handle missing values, duplicates, and outliers. Exploratory Data Analysis (EDA) and

visualizations were used to uncover trends and correlations. Feature engineering enhanced model performance by extracting time-based and business-related features. Machine learning models were implemented for regression and classification tasks. The results provide actionable insights for better pricing, sales forecasting, and business planning.

3. System Requirements

Hardware Requirements

- Minimum RAM: 8 GB
- Processor: Intel i5 / AMD Ryzen 5 or higher

Software Requirements

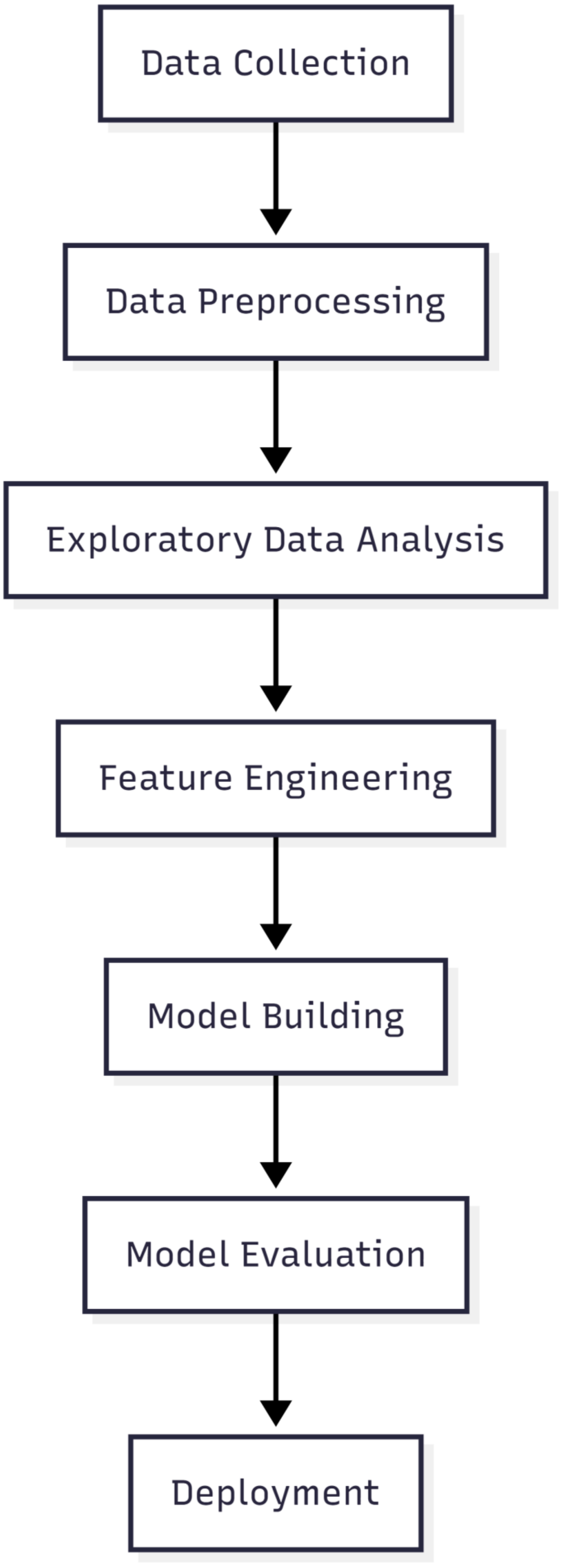
- Python 3.9 or above
- IDE: Jupyter Notebook / Google Colab
- Libraries:
  - pandas
  - numpy
  - matplotlib
  - seaborn
  - scikit-learn
  - xgboost

4. Objectives

- Perform data cleaning and preprocessing on restaurant sales data
- Analyze sales trends using statistical and visual techniques
- Identify key factors affecting selling price and quantity sold
- Build regression models to predict actual selling price
- Build classification models to categorize high and low sales
- Provide insights that support better business and marketing decisions

5. Flowchart of Project Workflow Steps:

Data Collection → Data Preprocessing → Exploratory Data Analysis → Feature Engineering → Model Building → Model Evaluation → Deployment

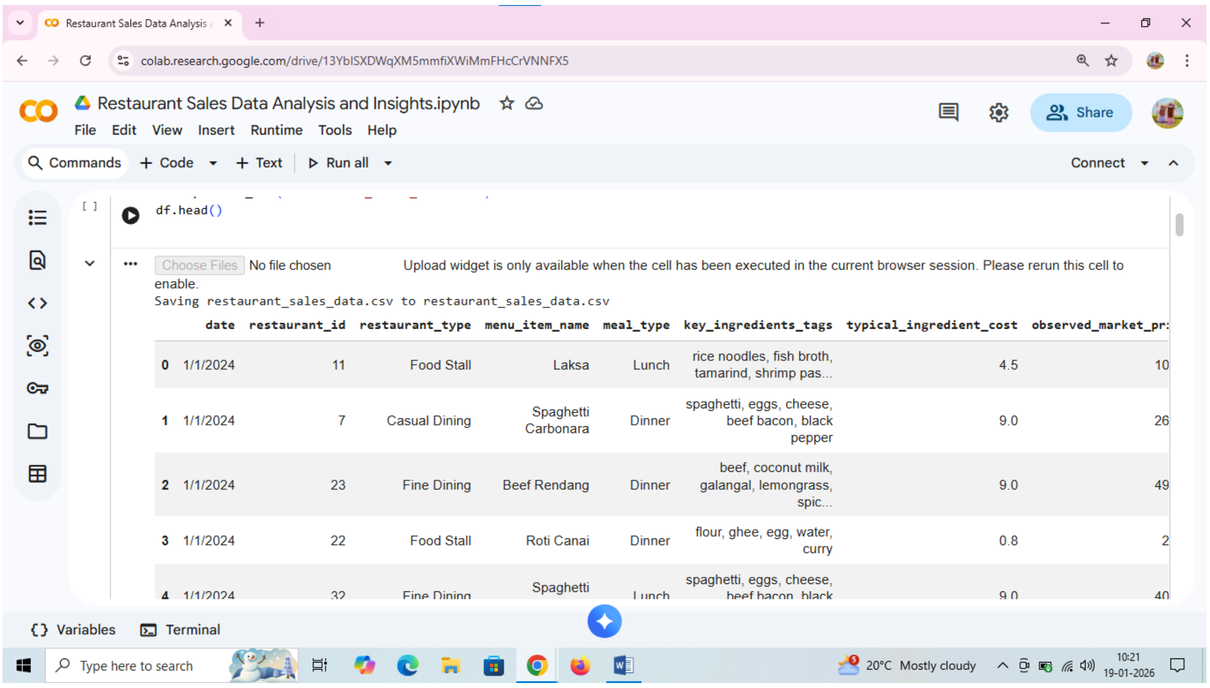

Data Collection
Data Preprocessing
Exploratory Data Analysis
Feature Engineering
Model Building
Model Evaluation
Deployment

6. Dataset Description

- **Source:** Kaggle (Public Dataset)
- **Type:** Public
- **Structure:**
  - Rows:5
  - Columns: 13
- **Link:** https://www.kaggle.com/datasets/alexandchen/restaurant-sales-report-2024-2025

7.Dataset Columns:

- date
- restaurant_id
- restaurant_type
- menu_item_name
- meal_type
- key_ingredients_tags
- typical_ingredient_cost
- observed_market_price
- actual_selling_price
- quantity_sold
- has_promotion
- special_event
- weather_condition

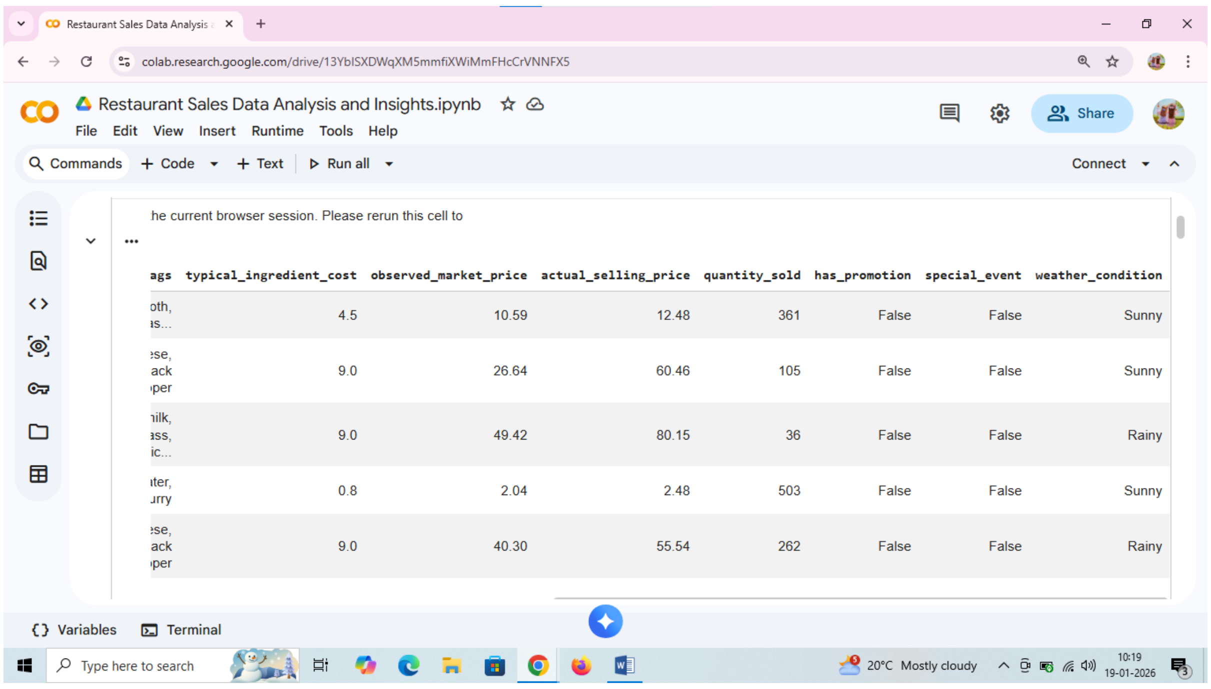

Restaurant Sales Data Analysis
colab.research.google.com/drive/13YbISXDWqXM5mmfiXWiMmFHcCrVNNFX5
Restaurant Sales Data Analysis and Insights.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Connect
Variables Terminal
Type here to search
20°C Mostly cloudy
10:21
19-01-2026

```
df.head()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving restaurant_sales_data.csv to restaurant_sales_data.csv
```

	date	restaurant_id	restaurant_type	menu_item_name	meal_type	key_ingredients_tags	typical_ingredient_cost	observed_market_pr
0	1/1/2024	11	Food Stall	Laksa	Lunch	rice noodles, fish broth, tamarind, shrimp pas...	4.5	10
1	1/1/2024	7	Casual Dining	Spaghetti Carbonara	Dinner	spaghetti, eggs, cheese, beef bacon, black pepper	9.0	26
2	1/1/2024	23	Fine Dining	Beef Rendang	Dinner	beef, coconut milk, galangal, lemongrass, spic...	9.0	49
3	1/1/2024	22	Food Stall	Roti Canai	Dinner	flour, ghee, egg, water, curry	0.8	2
4	1/1/2024	32	Fine Dining	Spaghetti	Lunch	spaghetti, eggs, cheese, beef bacon, black	9.0	40



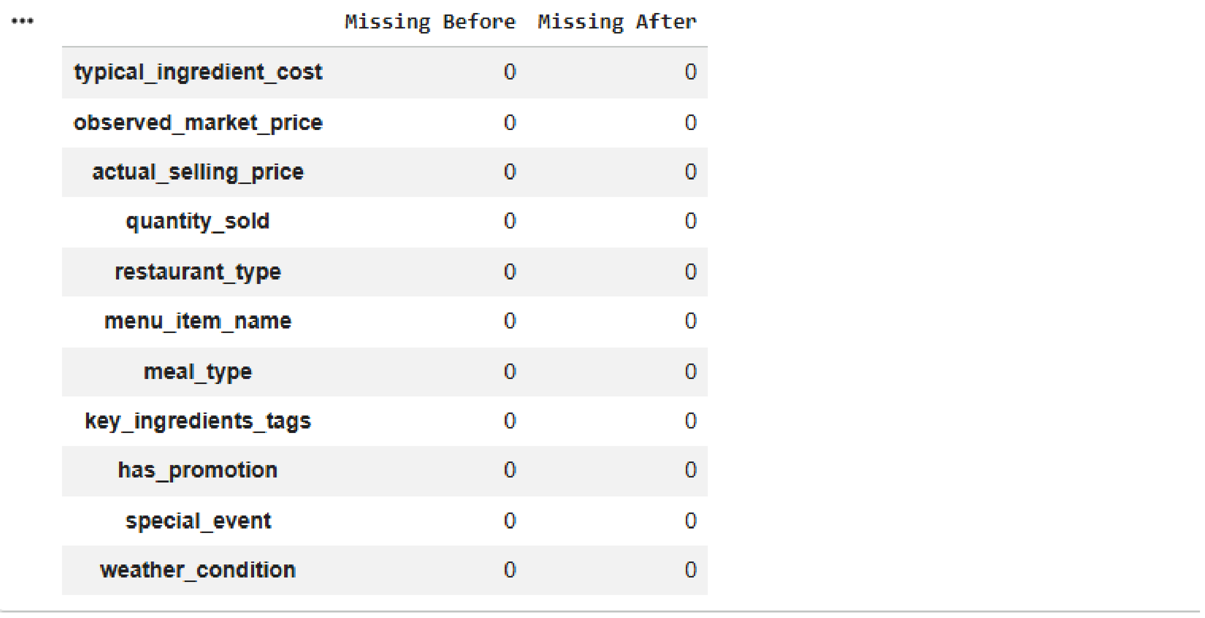
he current browser session. Please rerun this cell to

ags	typical_ingredient_cost	observed_market_price	actual_selling_price	quantity_sold	has_promotion	special_event	weather_condition
oth, ıs...	4.5	10.59	12.48	361	False	False	Sunny
ese, ack per	9.0	26.64	60.46	105	False	False	Sunny
nilk, ass, ic...	9.0	49.42	80.15	36	False	False	Rainy
ater, ırry	0.8	2.04	2.48	503	False	False	Sunny
ese, ack per	9.0	40.30	55.54	262	False	False	Rainy

7. Data Preprocessing

Steps Performed:

- Converted date column to datetime format
- Handled missing values using median (numerical) and mode (categorical)
- Removed duplicate records
- Created new features: day, month, year, day_of_week, is_weekend
- Treated outliers using IQR method
- Encoded categorical variables using One-Hot Encoding
- Scaled numerical features using StandardScale



	Missing Before	Missing After
typical_ingredient_cost	0	0
observed_market_price	0	0
actual_selling_price	0	0
quantity_sold	0	0
restaurant_type	0	0
menu_item_name	0	0
meal_type	0	0
key_ingredients_tags	0	0
has_promotion	0	0
special_event	0	0
weather_condition	0	0

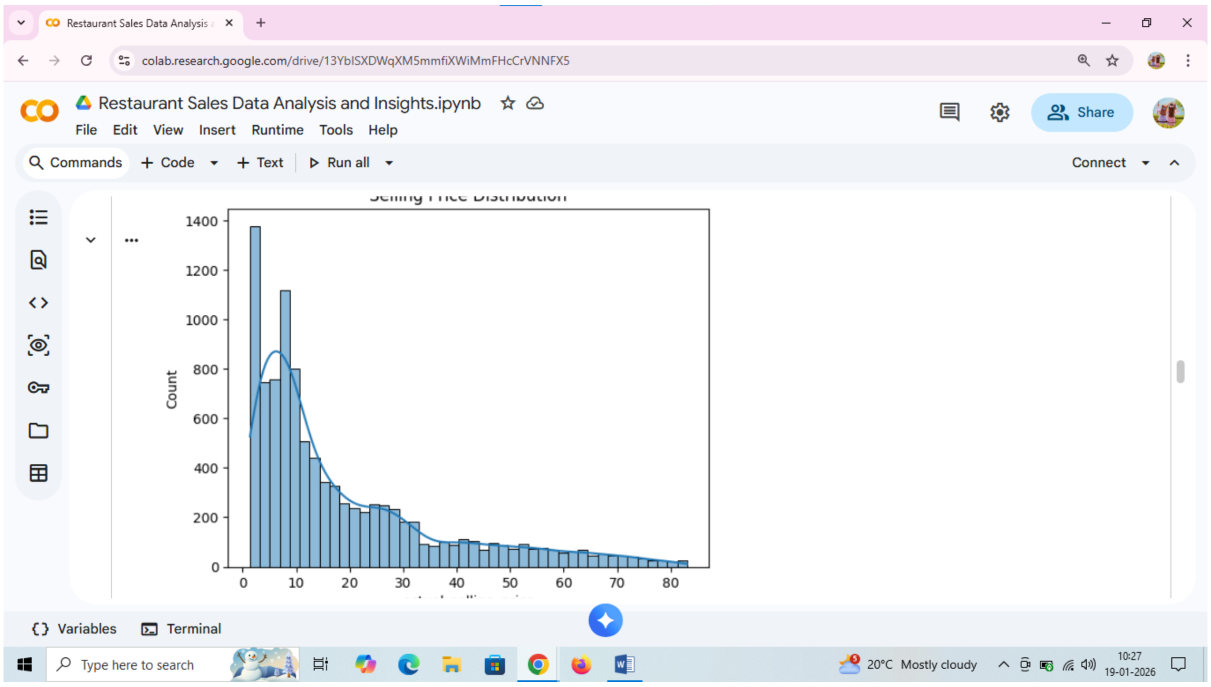
8. Exploratory Data Analysis (EDA)

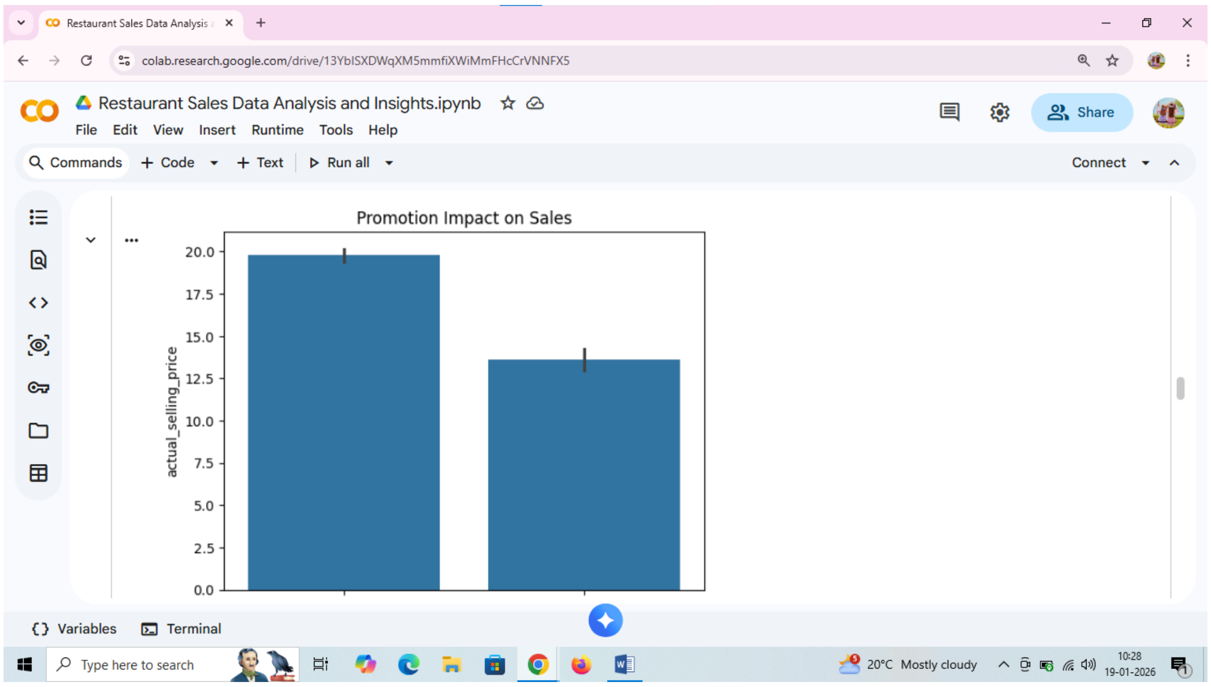
Techniques Used:

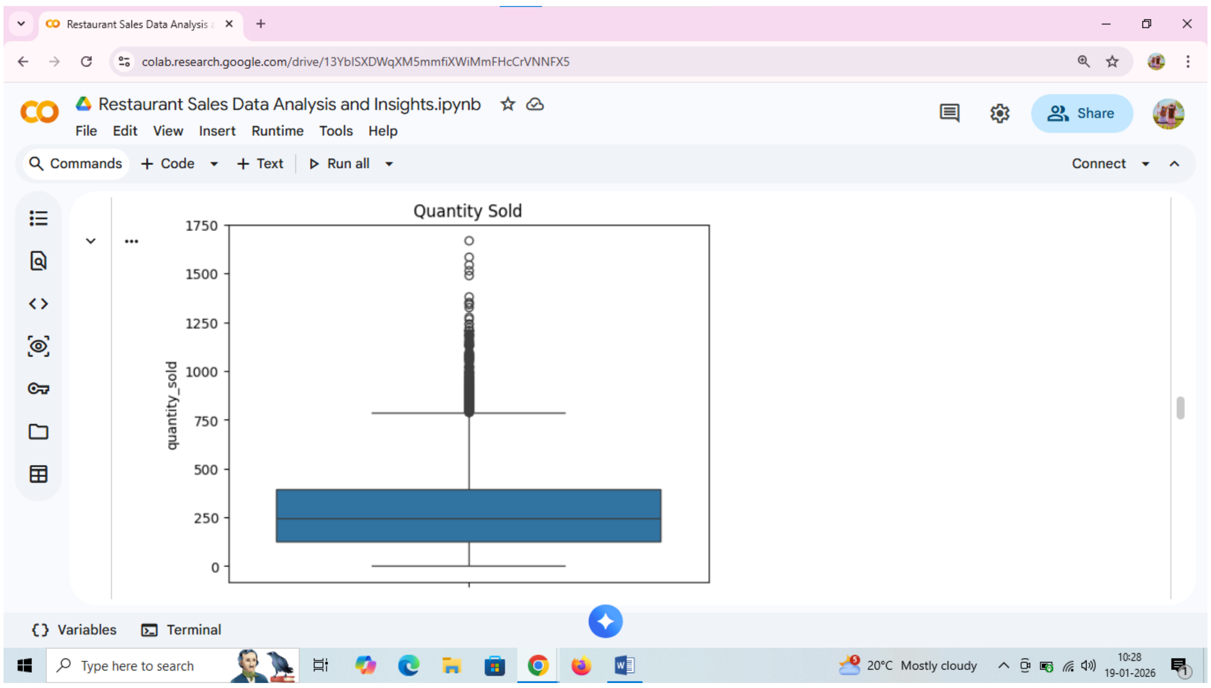
- Histograms for distribution analysis
- Boxplots for outlier detection
- Bar charts for categorical comparison
- Line plots for time-based trends
- Heatmaps for correlation analysis

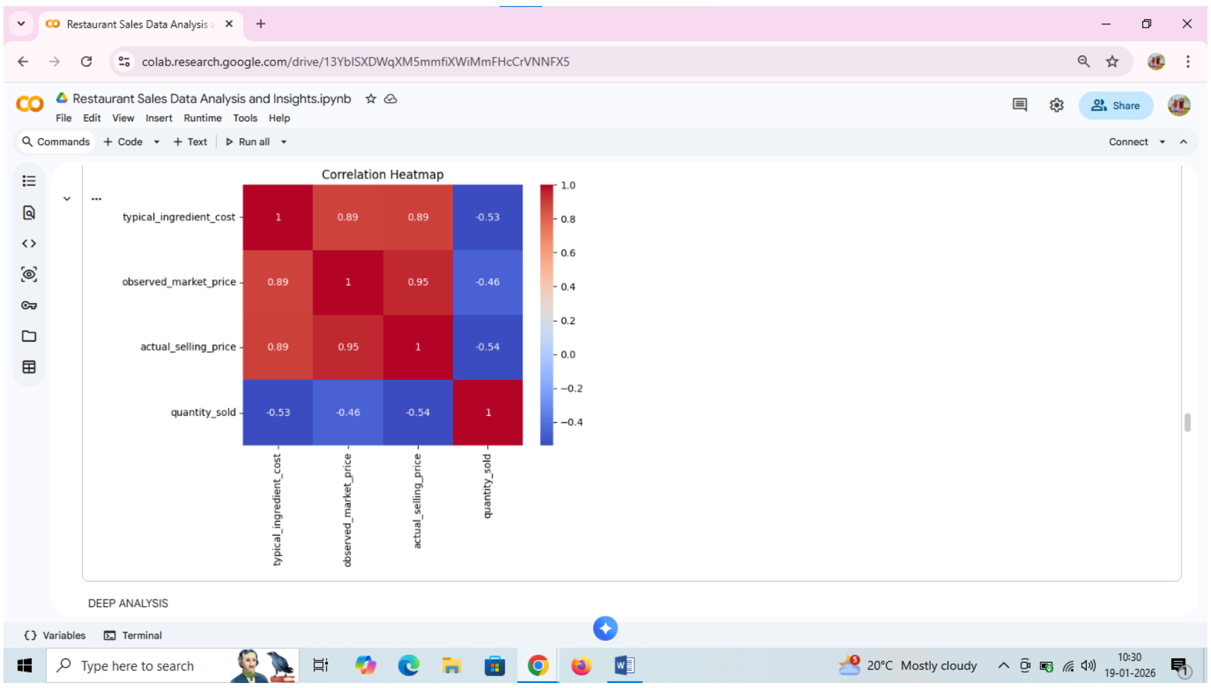
Key Insights:

- Promotions significantly increase sales
- Weekend sales are higher than weekdays
- Weather conditions impact customer demand
- Ingredient cost strongly influences selling price
- Certain menu items consistently generate higher revenue


Restaurant Sales Data Analysis
colab.research.google.com/drive/13YbISXDWqXM5mmfiXWiMmFHcCrVNNFX5
Restaurant Sales Data Analysis and Insights.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all Connect
Selling Price Distribution
Count
0 200 400 600 800 1000 1200 1400
0 10 20 30 40 50 60 70 80
Variables Terminal
Type here to search
20°C Mostly cloudy 10:27 19-01-2026


Restaurant Sales Data Analysis
colab.research.google.com/drive/13YbISXDWqXM5mmfiXWiMmFHcCrVNNFX5
Restaurant Sales Data Analysis and Insights.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all Connect
Promotion Impact on Sales
actual_selling_price
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0
Variables Terminal
Type here to search
20°C Mostly cloudy 10:28 19-01-2026





	typical_ingredient_cost	observed_market_price	actual_selling_price	quantity_sold
typical_ingredient_cost	1	0.89	0.89	-0.53
observed_market_price	0.89	1	0.95	-0.46
actual_selling_price	0.89	0.95	1	-0.54
quantity_sold	-0.53	-0.46	-0.54	1

9. Feature Engineering

New Features Created:

- Day, Month, Year
- Day of Week
- Is Weekend

Feature Selection:

- Selected features based on correlation and importance scores

Impact:

Feature engineering improved model accuracy by capturing temporal and business-specific patterns affecting sales.

10. Model Building

Regression Models:

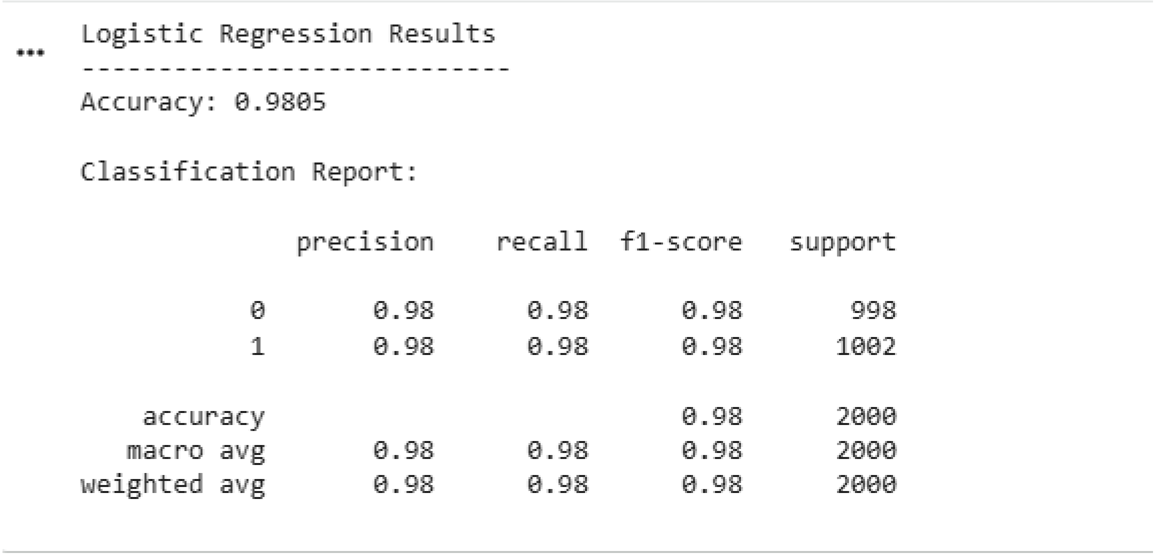
- Linear Regression (Baseline)
- Random Forest Regressor
- XGBoost Regressor

Classification Models:

- Logistic Regression
- Random Forest Classifier

Reason for Selection:

- Linear models provide baseline comparison
- Tree-based models capture non-linear relationships
- XGBoost improves predictive accuracy



```
Logistic Regression Results
----------------------------
Accuracy: 0.9805

Classification Report:

              precision    recall  f1-score   support

           0       0.98      0.98      0.98       998
           1       0.98      0.98      0.98      1002

    accuracy                           0.98      2000
   macro avg       0.98      0.98      0.98      2000
weighted avg       0.98      0.98      0.98      2000
```

```
...
    Random Forest Classification Results
    ------------------------------------
    Accuracy: 0.9745
```

```
confusion_matrix(y_test_c, y_pred_rf_c)
```

```
array([[966,  32],
       [ 19, 983]])
```

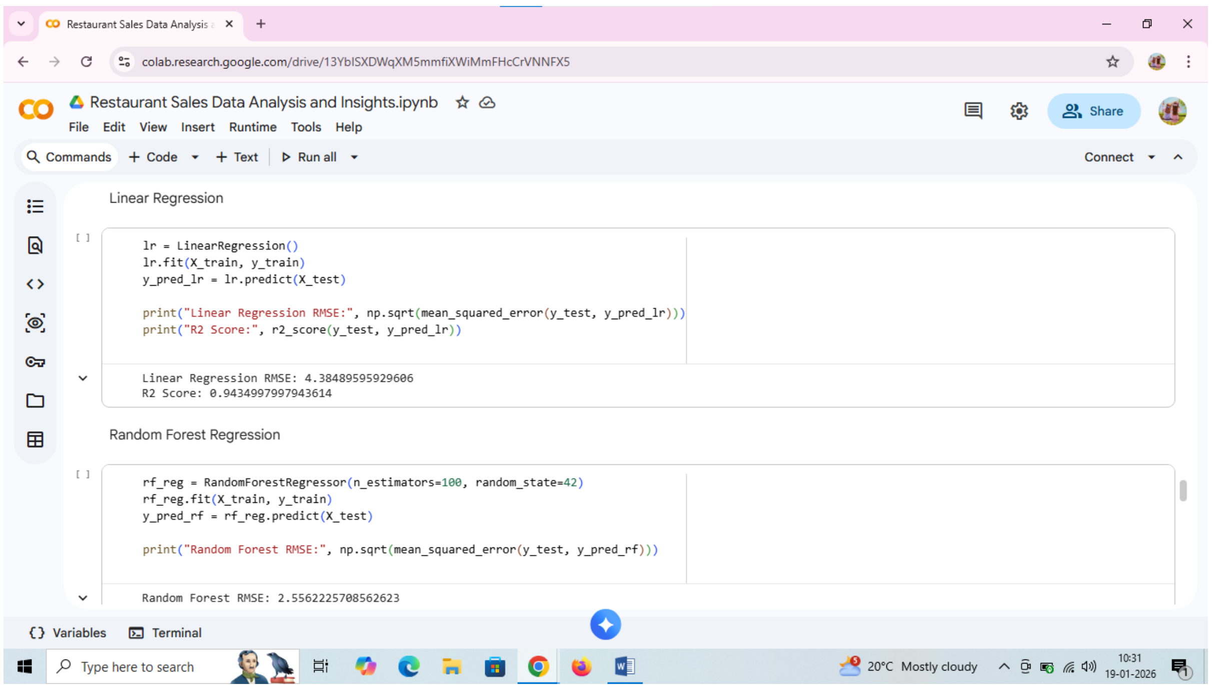
11. Model Evaluation

Regression Metrics:

- RMSE
- R² Score

Classification Metrics:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix


Restaurant Sales Data Analysis
colab.research.google.com/drive/13YbISXDWqXM5mmfiXWiMmFHcCrVNNFX5
Restaurant Sales Data Analysis and Insights.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Connect
Variables Terminal
Type here to search
20°C Mostly cloudy
10:31
19-01-2026

Linear Regression

```
lr = LinearRegression()
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)

print("Linear Regression RMSE:", np.sqrt(mean_squared_error(y_test, y_pred_lr)))
print("R2 Score:", r2_score(y_test, y_pred_lr))
```

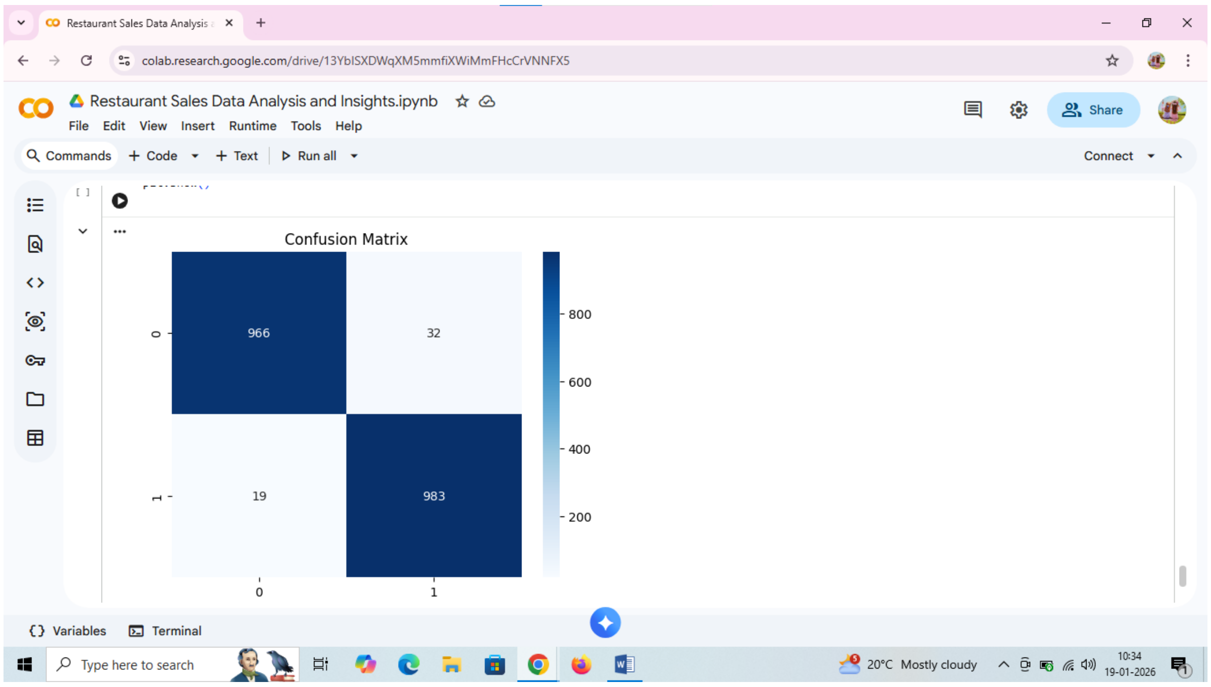
```
Linear Regression RMSE: 4.38489595929606
R2 Score: 0.9434997997943614
```

Random Forest Regression

```
rf_reg = RandomForestRegressor(n_estimators=100, random_state=42)
rf_reg.fit(X_train, y_train)
y_pred_rf = rf_reg.predict(X_test)

print("Random Forest RMSE:", np.sqrt(mean_squared_error(y_test, y_pred_rf)))
```

```
Random Forest RMSE: 2.5562225708562623
```



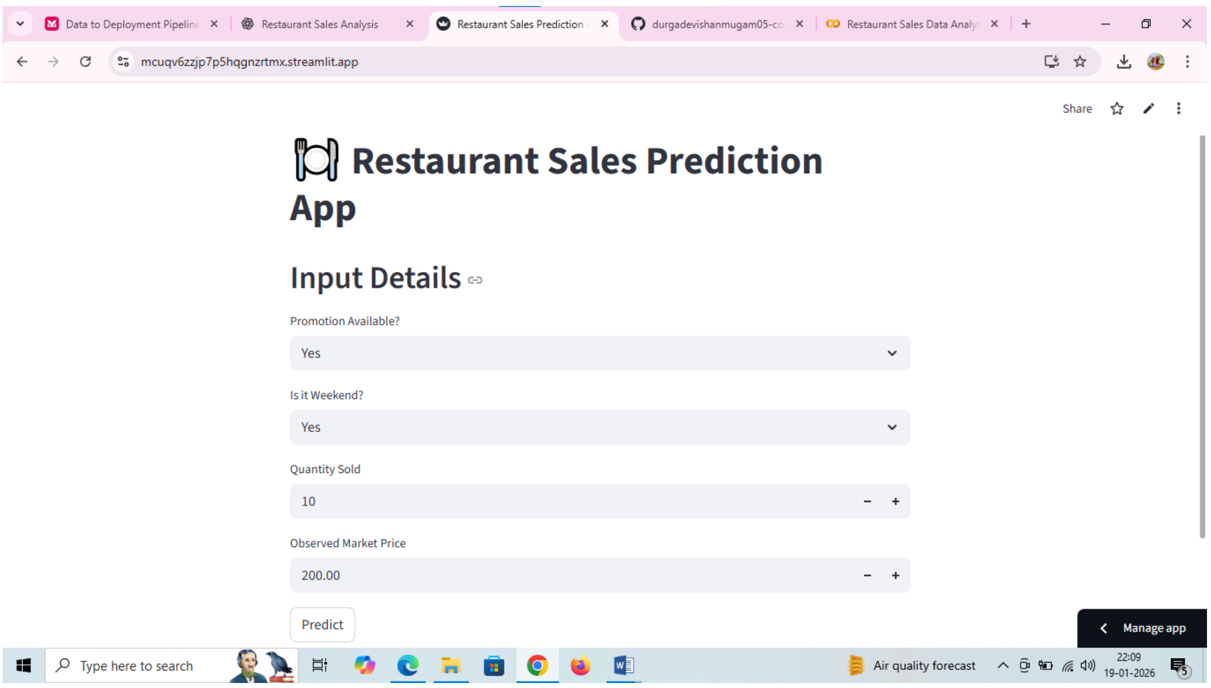
12. Deployment

Deployment Method:

- Streamlit Cloud

Details:

- Public Link: https://mcuqv6zzjp7p5hqgnzrtmx.streamlit.app/
- UI Screenshot:


Restaurant Sales Prediction App

Input Details

Promotion Available?
Yes

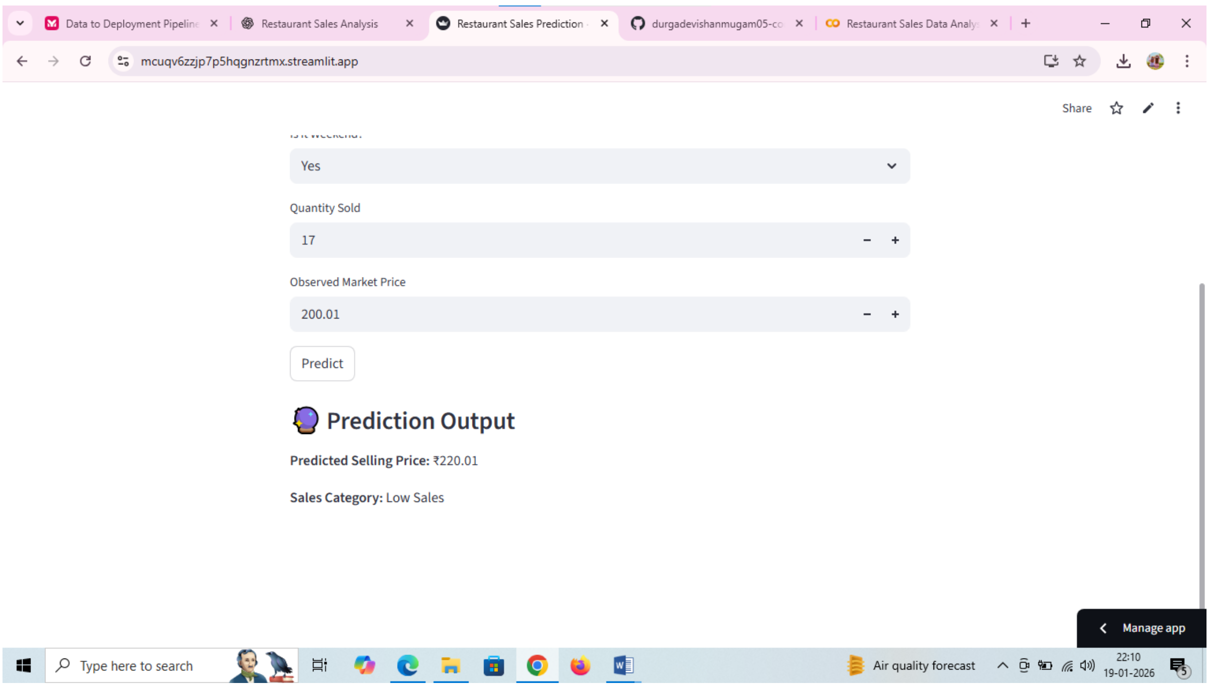
Is it Weekend?
Yes

Quantity Sold
10

Observed Market Price
200.00

Predict

- Sample Output:
  - Input: Promotion = Yes, Weekend = Yes
  - Output: Predicted Selling Price / High Sales Category


Yes

Quantity Sold
17

Observed Market Price
200.01

Predict

Prediction Output

Predicted Selling Price: ₹220.01

Sales Category: Low Sales

13. Source Code

The complete source code includes:

- Data Cleaning & EDA Notebook
- Feature Engineering Script

- Regression & Classification Model Code
- Streamlit Deployment App

IMPORT LIBRARIES

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.ensemble import RandomForestRegressor,
RandomForestClassifier
```

LOAD DATASET

```
 from google.colab import files
import pandas as pd

# Upload file from your computer
uploaded = files.upload()

# Suppose your file is 'restaurant_sales_data.csv'
df = pd.read_csv("restaurant_sales_data.csv")
df.head()
```

DATA UNDERSTANDING

```
df.info()
df.describe()
df.isnull().sum()
```

DATA CLEANING

1.Convert Date

```
df['date'] = pd.to_datetime(df['date'])
```

2.Handle Missing Values

```
import pandas as pd

before = df[num_cols + cat_cols].isnull().sum()

# Cleaning
df[num_cols] = df[num_cols].fillna(df[num_cols].median())
df[cat_cols] = df[cat_cols].fillna(df[cat_cols].mode().iloc[0])

after = df[num_cols + cat_cols].isnull().sum()

comparison = pd.DataFrame({
    'Missing Before': before,
    'Missing After': after
})
```

Comparison

3.To visually confirm cleaning

```
df[num_cols + cat_cols].info()
```

DATA ANALYSIS

1.Promotion Impact

```
df.groupby('has_promotion')['actual_selling_price'].mean()
```

2.Weather Impact

```
df.groupby('weather_condition')['quantity_sold'].mean()
```

DATA VISUALIZATION

1.Selling Price Distribution

```
sns.histplot(df['actual_selling_price'], kde=True)
plt.title("Selling Price Distribution")
plt.show()
```

2.Promotions vs Sales

```
sns.barplot(x='has_promotion', y='actual_selling_price', data=df)
plt.title("Promotion Impact on Sales")
plt.show()
```

3.Quantity Sold Boxplot

```
sns.boxplot(y=df['quantity_sold'])
plt.title("Quantity Sold")
plt.show()
```

4.Correlation Heatmap

```
sns.heatmap(df[num_cols].corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```

DEEP ANALYSIS

1.Weekend vs Weekday

```
df.groupby('is_weekend')['quantity_sold'].mean()
```

2.Top Menu Items

```
df['menu_item_name'].value_counts().head(10)
```

REGRESSION MODEL

1. Import REQUIRED Libraries

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
```

2.Convert date column properly

Colab may fail silently if date is object type.

```
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df['date'] = df['date'].astype(int) // 10**9
```

3. One-Hot Encoding

```
df_reg = pd.get_dummies(df, drop_first=True)
```

4. Split features and target

```
X = df_reg.drop('actual_selling_price', axis=1)
y = df_reg['actual_selling_price']
```

5. Train-Test Split

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

6. Feature Scaling

```
scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Linear Regression

```
lr = LinearRegression()
lr.fit(X_train, y_train)

y_pred_lr = lr.predict(X_test)

rmse_lr = np.sqrt(mean_squared_error(y_test, y_pred_lr))
r2_lr = r2_score(y_test, y_pred_lr)

print("Linear Regression Results")
print("-------------------------")
print("RMSE:", rmse_lr)
print("R2 Score:", r2_lr)
```

Random Forest Regression

```
rf_reg = RandomForestRegressor(
    n_estimators=100,
    random_state=42
)

rf_reg.fit(X_train, y_train)

y_pred_rf = rf_reg.predict(X_test)

rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_rf))

print("\nRandom Forest Regression Results")
print("--------------------------------")
print("RMSE:", rmse_rf)
```

CLASSIFICATION MODEL

🎯 High vs Low Sales Category

1. Import REQUIRED Libraries

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
```

2. Create Target Variable (High vs Low Sales)

```
df['high_sales'] = (df['actual_selling_price'] >
                    df['actual_selling_price'].median()).astype(int)

df[['actual_selling_price', 'high_sales']].head()
```

3. Handle Date Column

```
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df['date'] = df['date'].astype(int) // 10**9
```

4. One-Hot Encoding

```
df_clf = pd.get_dummies(df, drop_first=True)
```

5. Split Features and Target

```
Xc = df_clf.drop(['actual_selling_price', 'high_sales'], axis=1)
yc = df_clf['high_sales']
```

6. Train-Test Split

```
X_train_c, X_test_c, y_train_c, y_test_c = train_test_split(
    Xc, yc, test_size=0.2, random_state=42
)
```

7. Feature Scaling

```
scaler_c = StandardScaler()

X_train_c = scaler_c.fit_transform(X_train_c)
X_test_c = scaler_c.transform(X_test_c)
```

LOGISTIC REGRESSION

```
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train_c, y_train_c)

y_pred_log = log_reg.predict(X_test_c)

print("Logistic Regression Results")
print("----------------------------")
print("Accuracy:", accuracy_score(y_test_c, y_pred_log))
print("\nClassification Report:\n")
print(classification_report(y_test_c, y_pred_log))
```

RANDOM FOREST CLASSIFICATION

```
rf_clf = RandomForestClassifier(
    n_estimators=100,
    random_state=42
)

rf_clf.fit(X_train_c, y_train_c)

y_pred_rf_c = rf_clf.predict(X_test_c)

print("\nRandom Forest Classification Results")
print("-----------------------------------")
print("Accuracy:", accuracy_score(y_test_c, y_pred_rf_c))
```

Confusion Matrix

```
confusion_matrix(y_test_c, y_pred_rf_c)
```

CONFUSION MATRIX

```
cm = confusion_matrix(y_test_c, y_pred_rf_c)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```

14. Future Scope

- Integrate real-time data and APIs
- Include customer demographics and feedback analysis
- Apply deep learning models for time-series forecasting
- Build dashboards for restaurant managers

Project Summary: Restaurant Sales Data Analysis and Insights

This project focuses on analyzing restaurant sales data to extract meaningful insights and build predictive machine learning models that support business decision-making. The dataset contains information related to restaurant type, menu items, pricing, promotions, special events, weather conditions, and sales performance. Data cleaning techniques such as handling missing values, correcting data types, and encoding categorical variables were applied to ensure data quality and consistency.

Exploratory Data Analysis (EDA) was performed using visualizations like histograms, boxplots, and correlation heatmaps to understand sales trends, pricing behavior, and the impact of promotions and events. Feature engineering techniques were applied to enhance model performance, including target variable creation and numerical feature scaling.

Two machine learning tasks were implemented: **Regression** to predict the *actual selling price* and **Classification** to categorize sales into *High* and *Low* sales segments. Linear Regression and Random Forest Regression models were used for price prediction, while Logistic Regression and Random Forest Classification models were used for sales categorization. Among the models tested, Random Forest models delivered superior performance by effectively capturing non-linear patterns in the data.

Finally, the best-performing models were deployed using **Streamlit Cloud**, enabling real-time predictions through an interactive web interface. The deployed application allows users to input sales-related parameters and receive instant predictions, making the solution practical and scalable for real-world restaurant analytics.