

MAHATMA EDUCATION SOCIETY'S  
PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE  
(Autonomous)  
NEW PANVEL  
PROJECT REPORT ON  
**“Analyzing Pizza Sales Trends for Profitable Insights”**  
IN PARTIAL FULFILLMENT OF  
MASTERS OF DATA ANALYTICS  
SEMESTER I – 2023-24  
PROJECT GUIDE  
Name: Prof. Sanjana Bhangale  
SUBMITTED BY: Athulkrishna Pramod  
ROLL NO: 3105

## Evaluation sheet for continuous assessment with rubrics

Class: MSC-DA(PART 1)

Subject: PREDICTIVE ANALYTICS

Details about the continuous Assessment 2/Project work


Name of the Student: Athulkrishna Pramod

Roll Number: 3105

Class / Division: MSC-DA

Name of Evaluator: Prof. Sanjana Bhangale

Please circle appropriate score

Grading Criteria	Fair	Good	Excellent	Total
Introduction/ Description of the Case 	1	2	3	/3
SWOT Analysis of the company used for case analysis pertaining to the case ( <b>strength</b> of CA2 topic: e.g main important feature of CA1, <b>Weakness</b> : limitations of the project, <b>Opportunities</b> : in carrier in the future, <b>Threat</b> : obstacles that can cause failure to project CA2	3	4	5	/5
Learnings from the case	2	3	4	/4
Delivery/presentation skills	1	2	3	/3
Total				/15

- Inform the class the rubric format and the method of evaluation.

Co-ordinator,

# Pizza Sales Dataset

Pizza Sales.csv is a dataset sourced from Kaggle, offering insights into pizza sales within a restaurant or chain setting. This tabular dataset, stored in CSV format, likely originates from a pizza business and includes key fields such as date, pizza type, quantity sold, price, total sales and potentially order details.

The dataset's primary purpose is to support diverse analyses and research endeavours, from understanding pizza sales trends over time to uncovering the most favoured pizza varieties. It serves as a valuable resource for assessing revenue patterns, profit margins, and the impact of promotional activities. Researchers and data analysts can utilize this dataset for market research purposes, gaining insights into consumer preferences and pizza consumption habits.

Overall, "Pizza Sales.csv" presents an opportunity to delve into the delectable world of pizza economics and consumer behaviour, providing a rich dataset for a variety of data-driven investigations.

## Dataset Link:

<https://www.kaggle.com/datasets/shilongzhuang/pizza-sales/data>

**The below table provides a summary of the columns present in the dataset:**

<b>Column Names</b>	<b>Data Types</b>	<b>Description</b>
order_details_id	Integer	Unique identifier for each order placed by a table
order_id	Integer	Unique identifier for each pizza placed within each order, pizzas of the same type and size are kept in the same row, and the quantity increases
pizza_id	Object	Unique key identifier that ties the pizza ordered to its details, like size and price
quantity	Integer	Quantity ordered for each pizza of the same type and size
order_date	Object	Date the order was placed
order_time	Object	Time the order was placed
unit_price	Float64	Price of the pizza in USD
total_price	Float64	unit_price * quantity
pizza_size	Object	Size of the pizza (Small, Medium, Large, X Large, or XX Large)
pizza_category	Object	Category of the pizza (Classic, Supreme, Veggie, or Chicken)
pizza_ingredients	Object	Ingredients used in the pizza
pizza_name	Object	Name of the pizza as shown in the menu

# CODE AND OUTPUT

## ▼ Importing Important Libraries

✓  
3s

```
from google.colab import drive
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors
%matplotlib inline
drive.mount('/content/drive')
```

🔗 Drive already mounted at /content/drive; to attempt to forcil

## ▼ Import Dataset

✓  
1s

```
file = '/content/drive/My Drive/PA Project/Pizza Sales.csv'
df = pd.read_csv(file)
```

## ▼ Explore the Dataset

[ ] df.head()

	order_details_id	order_id	pizza_id	quantity	order_date	order_time	unit_price	total_price	pizza_size	pizza_category	pizza_ingredients	pizza_name	
0		1	1	hawaiian_m	1	01-01-15	11:38:36	13.25	13.25	M	Classic	Sliced Ham, Pineapple, Mozzarella Cheese	The Hawaiian Pizza
1		2	2	classic_dbx_m	1	01-01-15	11:57:40	16.00	16.00	M	Classic	Pepperoni, Mushrooms, Red Onions, Red Peppers,...	The Classic Deluxe Pizza
2		3	2	five_cheese_l	1	01-01-15	11:57:40	18.50	18.50	L	Veggie	Mozzarella Cheese, Provolone Cheese, Smoked Go...	The Five Cheese Pizza
3		4	2	ital_supr_l	1	01-01-15	11:57:40	20.75	20.75	L	Supreme	Calabrese Salami, Capocollo, Tomatoes, Red Oni...	The Italian Supreme Pizza
4		5	2	mexicana_m	1	01-01-15	11:57:40	16.00	16.00	M	Veggie	Tomatoes, Red Peppers, Jalapeno Peppers, Red O...	The Mexicana Pizza

[4] df.tail()

	order_details_id	order_id	pizza_id	quantity	order_date	order_time	unit_price	total_price	pizza_size	pizza_category	pizza_ingredients	pizza_name	
48615		48616	21348	ckn_alfredo_m	1	31-12-15	21:23:10	16.75	16.75	M	Chicken	Chicken, Red Onions, Red Peppers, Mushrooms, A...	The Chicken Alfredo Pizza
48616		48617	21348	four_cheese_l	1	31-12-15	21:23:10	17.95	17.95	L	Veggie	Ricotta Cheese, Gorgonzola Piccante Cheese, Mo...	The Four Cheese Pizza
48617		48618	21348	napolitana_s	1	31-12-15	21:23:10	12.00	12.00	S	Classic	Tomatoes, Anchovies, Green Olives, Red Onions,....	The Napolitana Pizza
48618		48619	21349	mexicana_l	1	31-12-15	22:09:54	20.25	20.25	L	Veggie	Tomatoes, Red Peppers, Jalapeno Peppers, Red O...	The Mexicana Pizza
48619		48620	21350	bbq_ckn_s	1	31-12-15	23:02:05	12.75	12.75	S	Chicken	Barbecued Chicken, Red Peppers, Green Peppers,....	The Barbecue Chicken Pizza

```
[ ] df.shape
```

```
(48620, 12)
```

```
[ ] df.columns
```

```
Index(['order_details_id', 'order_id', 'pizza_id', 'quantity', 'order_date',  
      'order_time', 'unit_price', 'total_price', 'pizza_size',  
      'pizza_category', 'pizza_ingredients', 'pizza_name'],  
      dtype='object')
```

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 48620 entries, 0 to 48619  
Data columns (total 12 columns):  
#   Column                Non-Null Count  Dtype  
---  ---  
0   order_details_id      48620 non-null  int64  
1   order_id              48620 non-null  int64  
2   pizza_id              48620 non-null  object  
3   quantity              48620 non-null  int64  
4   order_date            48620 non-null  object  
5   order_time            48620 non-null  object  
6   unit_price            48620 non-null  float64  
7   total_price           48620 non-null  float64  
8   pizza_size            48620 non-null  object  
9   pizza_category        48620 non-null  object  
10  pizza_ingredients      48620 non-null  object  
11  pizza_name             48620 non-null  object  
dtypes: float64(2), int64(3), object(7)  
memory usage: 4.5+ MB
```

Categorical Data: pizza\_size, pizza\_category, pizza\_ingredients, pizza\_name

Numerical Data: quantity, unit\_price, total\_price

```
[ ] df.describe()
```

	order_details_id	order_id	quantity	unit_price	total_price
count	48620.000000	48620.000000	48620.000000	48620.000000	48620.000000
mean	24310.500000	10701.479761	1.019622	16.494132	16.821474
std	14035.529381	6180.119770	0.143077	3.621789	4.437398
min	1.000000	1.000000	1.000000	9.750000	9.750000
25%	12155.750000	5337.000000	1.000000	12.750000	12.750000
50%	24310.500000	10682.500000	1.000000	16.500000	16.500000
75%	36465.250000	16100.000000	1.000000	20.250000	20.500000
max	48620.000000	21350.000000	4.000000	35.950000	83.000000

## ▼ Data Cleaning

```
[ ] df.isnull().sum()
```

```
order_details_id    0
order_id            0
pizza_id            0
quantity            0
order_date          0
order_time          0
unit_price          0
total_price         0
pizza_size          0
pizza_category      0
pizza_ingredients   0
pizza_name          0
dtype: int64
```

```
[ ] df.duplicated().sum()
```

```
0
```

```
[ ] number_of_pizza = df.pizza_name.nunique()
print(f"The restaurant sells {number_of_pizza} types of pizzas!")
```

```
The restaurant sells 32 types of pizzas!
```

There is Nothing to clean, lets move on to the next step that is Exploratory Data Analysis (EDA).

## ▼ Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a crucial phase in the data analysis process where data professionals delve into a dataset to uncover its underlying patterns, characteristics, and potential issues. Through summary statistics, data visualizations, and data cleaning, EDA offers insights into the data's distribution, relationships, and outliers.

In this step I would like to explore these questions:

1. Which category of pizza is ordered the most?
2. Which size of pizza is ordered the most?
3. What are our best and worst-selling pizzas?
4. What's our average order value?
5. What is the total revenue up to the latest order date?
6. What is the total number of orders up to the latest order date?
7. Which month was revenue earned the highest?
8. What is the average unit price and revenue of most sold 5 pizzas?



1. Which category of pizza is ordered the most?

```
[ ] # Show the number of orders for each category of pizza
categories = df.pizza_category.value_counts()

# Find the index of the highest value (highest number of orders)
highest_index = categories.idxmax()

# Set the color of the highest bar to green, and others to blue color
colors = ['#EE6A50' if i == highest_index else '#76EEC6' for i in categories.index]

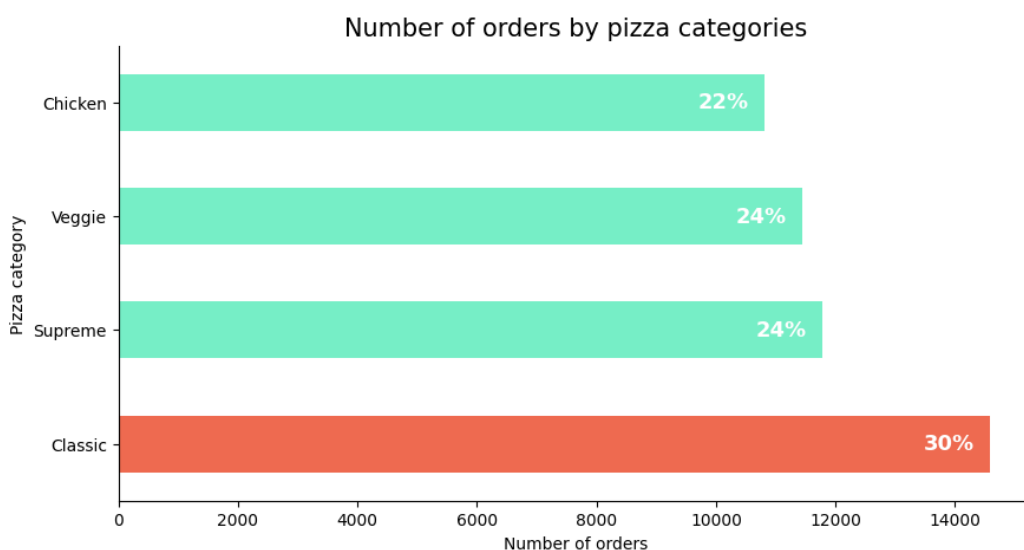
# Plot the bar chart
plots = categories.plot.barh(figsize = (10,5), color = colors)
plt.ylabel('Pizza category')
plt.xlabel('Number of orders')
plt.title('Number of orders by pizza categories', fontsize = 15)

# Total number of orders
total = len(df)

# Iterating over the bars one-by-one
for bar in plots.patches:

    # Using Matplotlib's annotate function and
    # passing the coordinates where the annotation shall be done
    # y-coordinate: bar.get_y() + bar.get_width() / 2
    # x-coordinate: bar.get_width()
    # ha and va stand for the horizontal and vertical alignment
    plots.annotate(format(bar.get_width() / total, '.0%'),
                   (bar.get_width(), bar.get_y() + bar.get_height() / 2),
                   ha='center', va='center',
                   size=13, xytext=(-25, 0),
                   textcoords='offset points', color = 'white', fontweight = 600)

# Hide the right and top spines
plots.spines[['right', 'top']].set_visible(False);
```



By analysing the above graph we get to see that most of the people orders classic pizza.

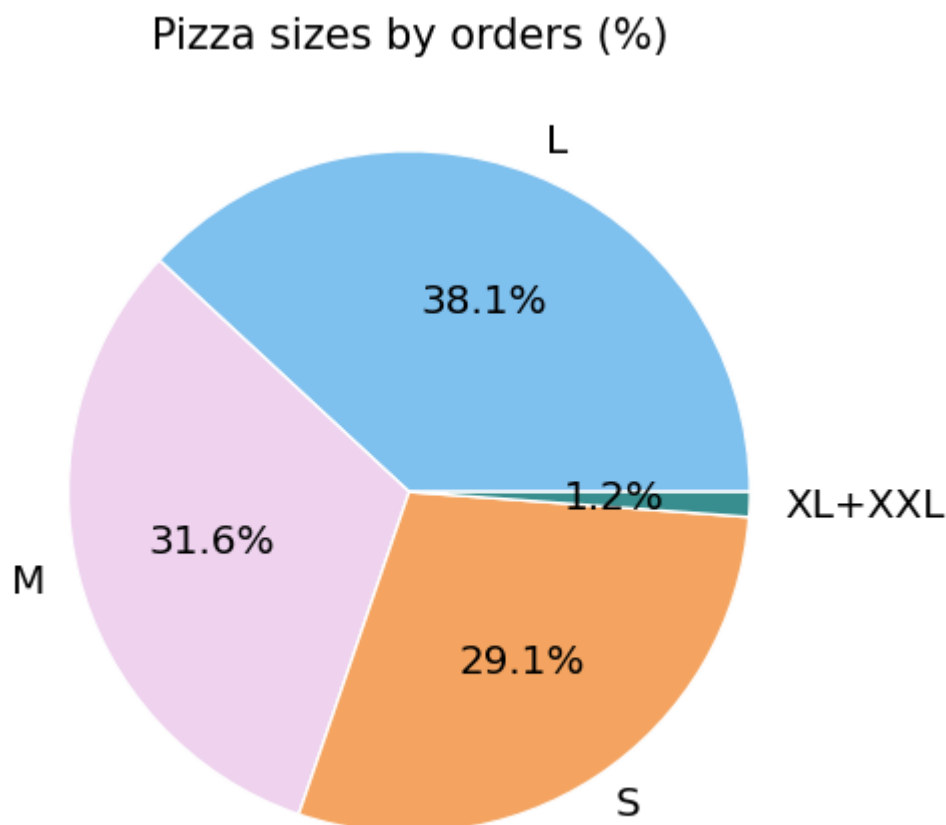
2. Which size of pizza is ordered the most?

```
[ ] total = len(df) # total number of orders
L = len(df[df.pizza_size == 'L']) / total # % of Large pizzas to total no. of orders
M = len(df[df.pizza_size == 'M']) / total # % of Medium pizzas to total no. of orders
S = len(df[df.pizza_size == 'S']) / total # % of Small pizzas to total no. of orders
XL = len(df[df.pizza_size == 'XL']) / total # % of Extra Large pizzas to total no. of orders
XXL = len(df[df.pizza_size == 'XXL']) / total # % of Extra Extra Large pizzas to total no. of orders

# Plot the pie chart
figure, ax = plt.subplots()
patches, texts, pcts = ax.pie([L,M,S,XL+XXL],
                              labels = ["L","M","S","XL+XXL"],
                              autopct = '%.1f%%',
                              wedgeprops={'linewidth':1.0, 'edgecolor': 'white'},
                              textprops = {'size': 'x-large'},
                              shadow = False,
                              colors= ['#7EC0EE', '#EED2EE', '#F4A460', '#388E8E'])

# Title of pie chart
ax.set_title('Pizza sizes by orders (%)', fontsize = 15)

# Show the pie chart
plt.tight_layout();
```



This Pie chart represents that the Large Sized Pizza is ordered most and the XL / XXL sized pizzas are ordered the least.

3. What are our best and worst-selling pizzas?

```
[ ] print(f'The best selling pizza is: {df.pizza_name.value_counts().nlargest(1)}' '\n')
    print(f'The worst selling pizza is : {df.pizza_name.value_counts().nsmallest(1)}')
```

```
The best selling pizza is: The Classic Deluxe Pizza    2416
Name: pizza_name, dtype: int64
```

```
The worst selling pizza is : The Brie Carre Pizza    480
Name: pizza_name, dtype: int64
```

4. What's our average order value?

```
[ ] print(f'Mean: USD {df.total_price.mean():.2f}')
    print(f'Median: USD {df.total_price.median():.2f}')
```

```
Mean: USD 16.82
```

```
Median: USD 16.50
```

5. What is the total revenue up to the latest order date?

```
[ ] df.total_price.sum()
```

```
817860.05
```

6. What is the total number of orders up to the latest order date?

```
[ ] len(df)
```

```
48620
```

7. Which month was revenue earned the highest?

```
[ ] # Extract the month and month_name from order_date
month = []
month_name = []
df['date_column'] = pd.to_datetime(df['order_date'])
for i in df.date_column:
    # 1 = January, 2 = February, ..., 12 = December
    month = i.month
    # January, February, ..., December
    month_name.append(i.month_name())

df['month'] = month
df['month_name'] = month_name
total_revenue = df.total_price.sum() # total revenue = sum of prices of all orders

# Find the total revenue by each month using groupby
revenue_per_month_df = df.groupby(['month_name']).agg({'total_price': 'sum'})

# Rank the monthly total revenue from highest to lowest
revenue_per_month_df['rank'] = revenue_per_month_df['total_price'].rank(method="dense", ascending=False)

# Sort from the lowest revenue to the highest revenue
revenue_per_month_df = revenue_per_month_df.sort_values('rank', ascending=True).reset_index()
revenue_per_month_df
```

	month_name	total_price	rank
0	January	71620.15	1.0
1	March	71301.40	2.0
2	November	71004.85	3.0
3	July	70880.65	4.0
4	April	70312.00	5.0
5	August	69497.30	6.0
6	June	68161.45	7.0
7	October	68152.20	8.0
8	May	67648.80	9.0
9	February	64419.45	10.0
10	September	63803.70	11.0
11	December	61058.10	12.0

In this statistical report, the month of January ranks first in earning the most revenue of 71k dollars and December month ranks last with the earning 61k dollars.

## Visualizing the above code

```
[ ] # Create a bar plot to visualize monthly total revenue
# Define a custom color palette for the bars
custom_colors = ['#8B475D', '#ff7f0e', '#2ca02c', '#d62728',
                 '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22',
                 '#17becf', '#B0C4DE', '#FFC0CB']

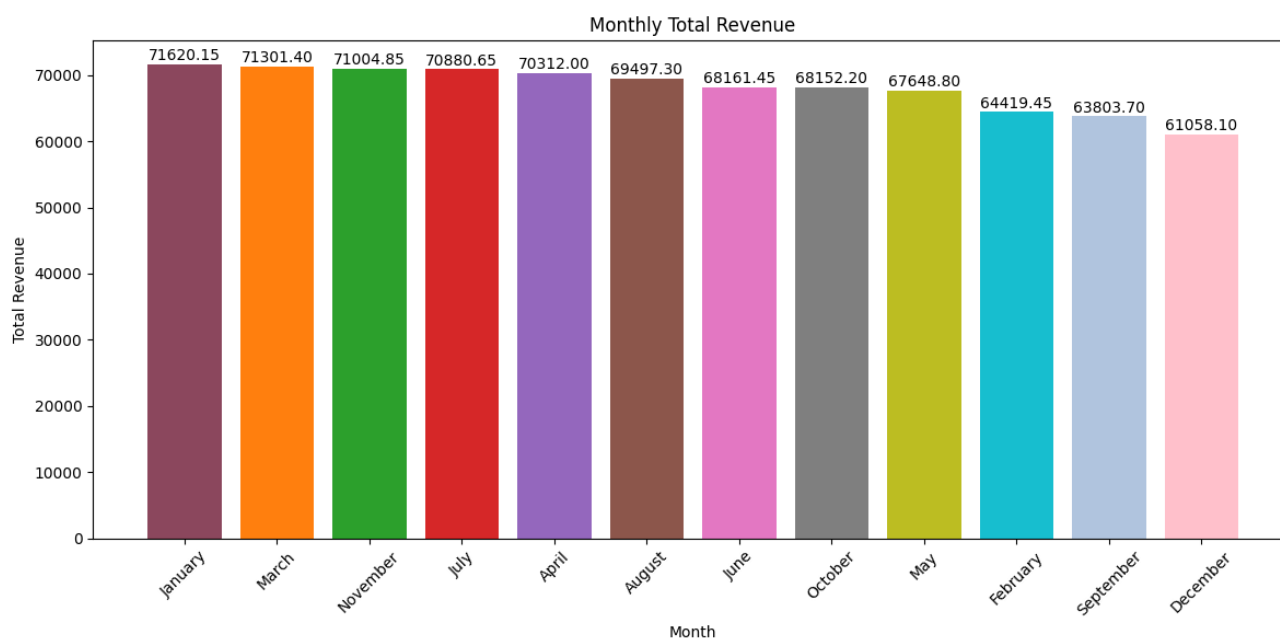
# Create a bar plot with the custom color palette
plt.figure(figsize=(12, 6)) # Set the figure size
bars = plt.bar(revenue_per_month_df['month_name'],
               revenue_per_month_df['total_price'], color=custom_colors)

plt.xlabel('Month')
plt.ylabel('Total Revenue')
plt.title('Monthly Total Revenue')
plt.xticks(rotation=45) # Rotate x-axis labels for better readability

# Annotate the bars with revenue values
for bar, revenue in zip(bars, revenue_per_month_df['total_price']):
    plt.text(bar.get_x() + bar.get_width() / 2, revenue + 100,
             f'{revenue:.2f}', ha='center', va='bottom')

plt.tight_layout() # Ensure the labels fit within the figure area

# Display the plot
plt.show()
```



It is a visual representation of the above code showing that the Jan month brought the most revenue in the following year.

8. What is the average unit price and revenue of most sold 5 pizzas?

```
[ ] top_pizza_analysis = df.groupby('pizza_name').agg(
    average_unit_price=('unit_price', 'mean'),
    revenue_per_pizza=('unit_price', lambda x: (x * df['quantity']).sum())
).nlargest(5, 'revenue_per_pizza')
print("Average Unit Price and Revenue of Top 3 Pizzas:\n", top_pizza_analysis)
```

```
Average Unit Price and Revenue of Top 3 Pizzas:
              average_unit_price  revenue_per_pizza
pizza_name
The Thai Chicken Pizza          18.286069          43434.25
The Barbecue Chicken Pizza       17.572934          42768.00
The California Chicken Pizza     17.448523          41409.50
The Classic Deluxe Pizza         15.575952          38180.50
The Spicy Italian Pizza          18.104663          34831.25
```

This analysis shows that the ‘Thai Chicken Pizza’ is the top pizza sold in the restaurant with total revenue of 43k dollars

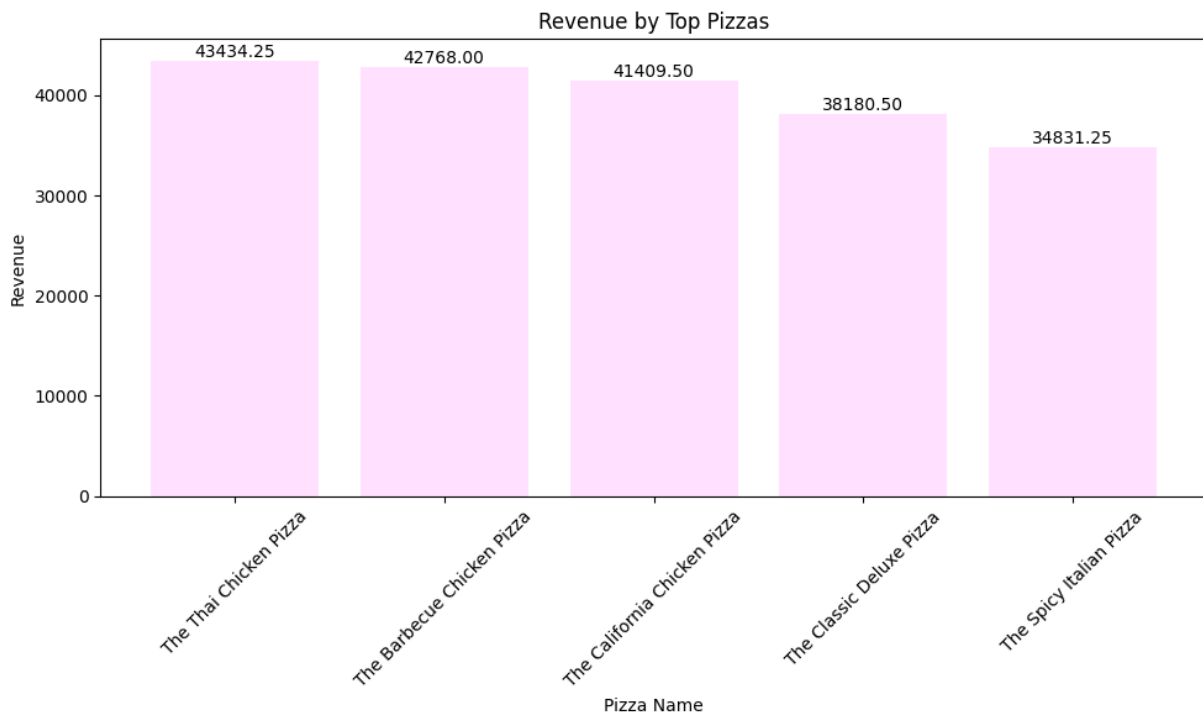
Visualizing Revenue By Top Pizzas

```
[ ] # Create a bar plot to visualize revenue by top pizzas
plt.figure(figsize=(10, 6)) # Set the figure size
bars = plt.bar(top_pizza_analysis.index, top_pizza_analysis
    ['revenue_per_pizza'], color='#FFE1FF')
plt.xlabel('Pizza Name')
plt.ylabel('Revenue')
plt.title('Revenue by Top Pizzas')
plt.xticks(rotation=45) # Rotate x-axis labels for better readability

# Annotate the bars with revenue values
for bar, revenue in zip(bars, top_pizza_analysis['revenue_per_pizza']):
    plt.text(bar.get_x() + bar.get_width() / 2, revenue, f'{revenue:.2f}',
        ha='center', va='bottom')

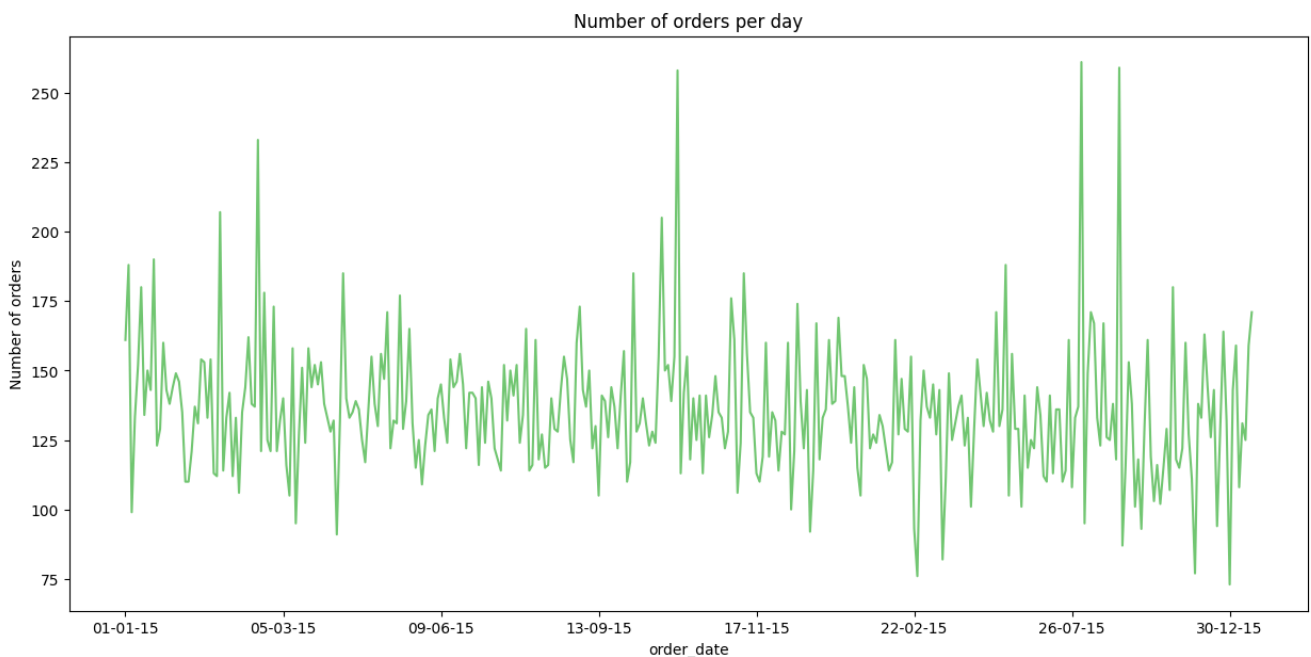
plt.tight_layout() # Ensure the labels fit within the figure area

# Display the plot
plt.show()
```



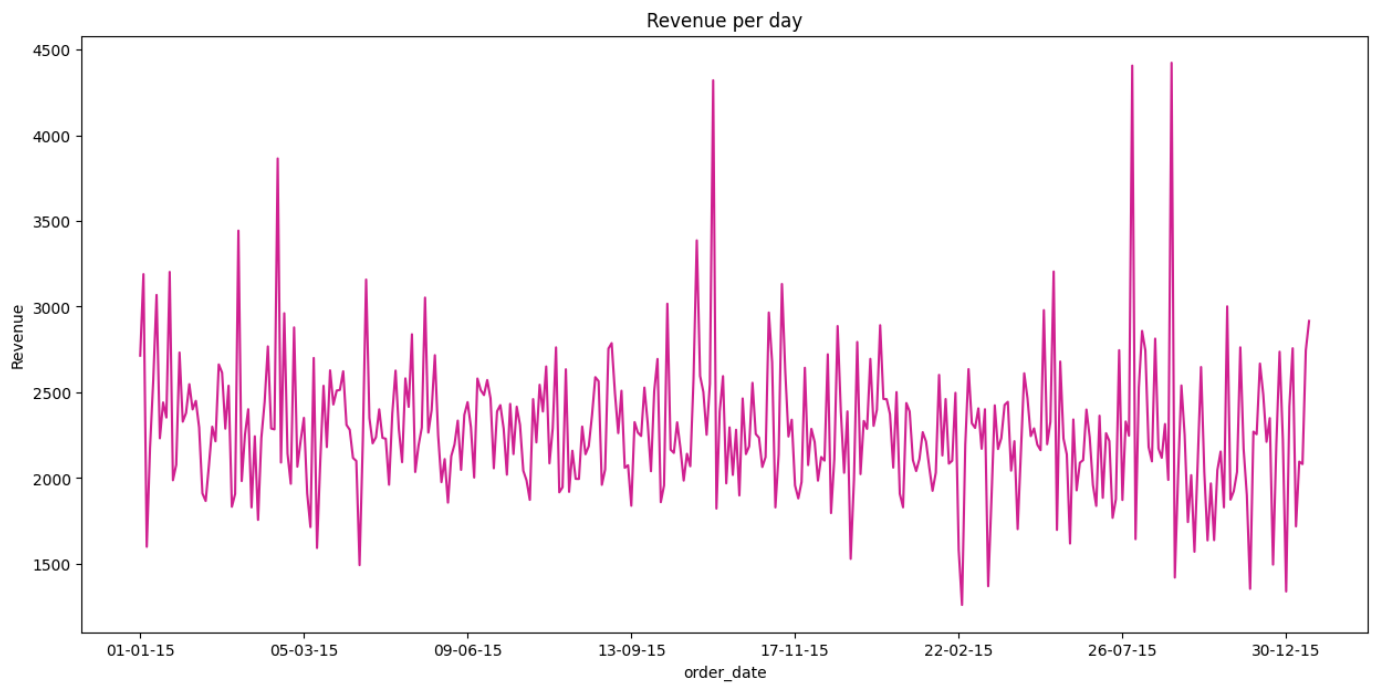
It is a visual representation of the above code showing that these are the top 5 pizzas sold in the restaurant and the Thai Chicken Pizza ranks first among them.

## Visualizing Orders Per Day



This graph shows that the No of orders per day is around between 100 – 175 orders daily.

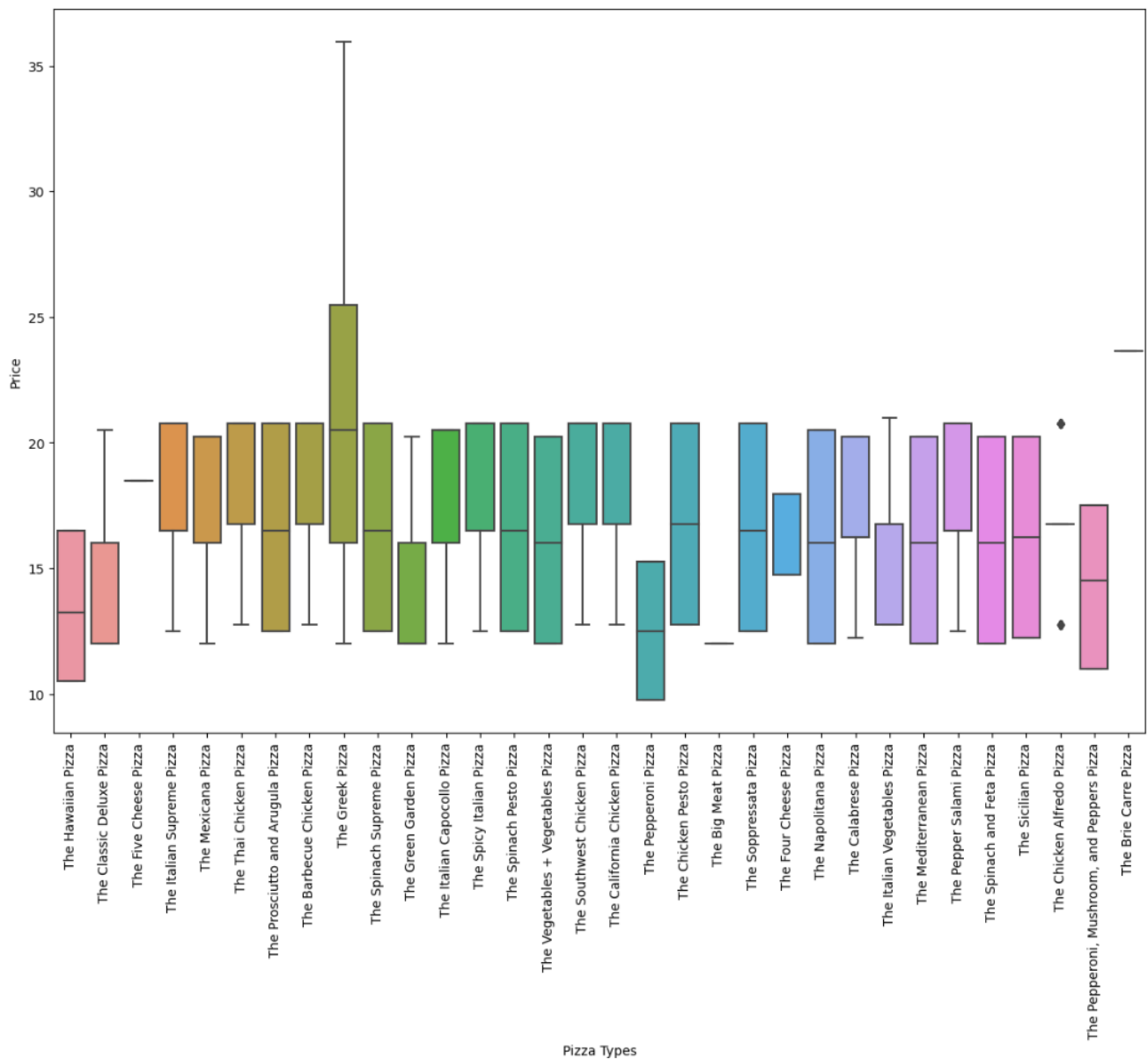
## Visualizing Revenue Per Day



This graph gives an insight of revenue per day and the average revenue per day is around between 1800 – 2800 USD per day.



## Outlier Detection



This boxplot gives an insight of outliers present in the features, here the features are Pizza types & Price, In this only the Chicken Alfredo Pizza type is having some outliers rest of the types are mostly equally spread.

## ▼ Data Preparation

```
[ ] df.corr()
```

```
<ipython-input-26-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in I
df.corr()
```

	order_details_id	order_id	quantity	unit_price	total_price	month
order_details_id	1.000000	0.999990	0.003639	-0.003286	-0.000847	NaN
order_id	0.999990	1.000000	0.003621	-0.003323	-0.000889	NaN
quantity	0.003639	0.003621	1.000000	0.007142	0.541926	NaN
unit_price	-0.003286	-0.003323	0.007142	1.000000	0.836087	NaN
total_price	-0.000847	-0.000889	0.541926	0.836087	1.000000	NaN
month	NaN	NaN	NaN	NaN	NaN	NaN

Converting Categorical Data to numerical using Label Encoder

```
[ ] from sklearn.preprocessing import LabelEncoder
le = LabelEncoder();
df['pizza_id'] = le.fit_transform(df['pizza_id'])
df['pizza_category'] = le.fit_transform(df['pizza_category'])
df['pizza_ingredients'] = le.fit_transform(df['pizza_ingredients'])
df['pizza_name'] = le.fit_transform(df['pizza_name'])
df['month_name'] = le.fit_transform(df['month_name'])
df['order_date'] = le.fit_transform(df['order_date'])
df['order_time'] = le.fit_transform(df['order_time'])
df['date_column'] = le.fit_transform(df['date_column'])
df['pizza_size'] = le.fit_transform(df['pizza_size'])
```

## ▼ Model Building

### ▼ Decision Tree

```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score

# Define features and target
X = df[['unit_price', 'total_price', 'pizza_category', 'pizza_name', 'month_name', 'pizza_size', 'order_time']]
y = df['quantity']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 2: Train a Decision Tree model (you can customize hyperparameters as needed)
dct_model = DecisionTreeRegressor(random_state=42)
dct_model.fit(X_train, y_train)

# Make predictions
y_pred = dct_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = (np.sqrt(mse))
r2 = r2_score(y_test, y_pred)

print(f'The RMSE for the Decision Tree model is : {rmse}')
print(f'The R2 Score for the Decision Tree model is : {r2}')
```

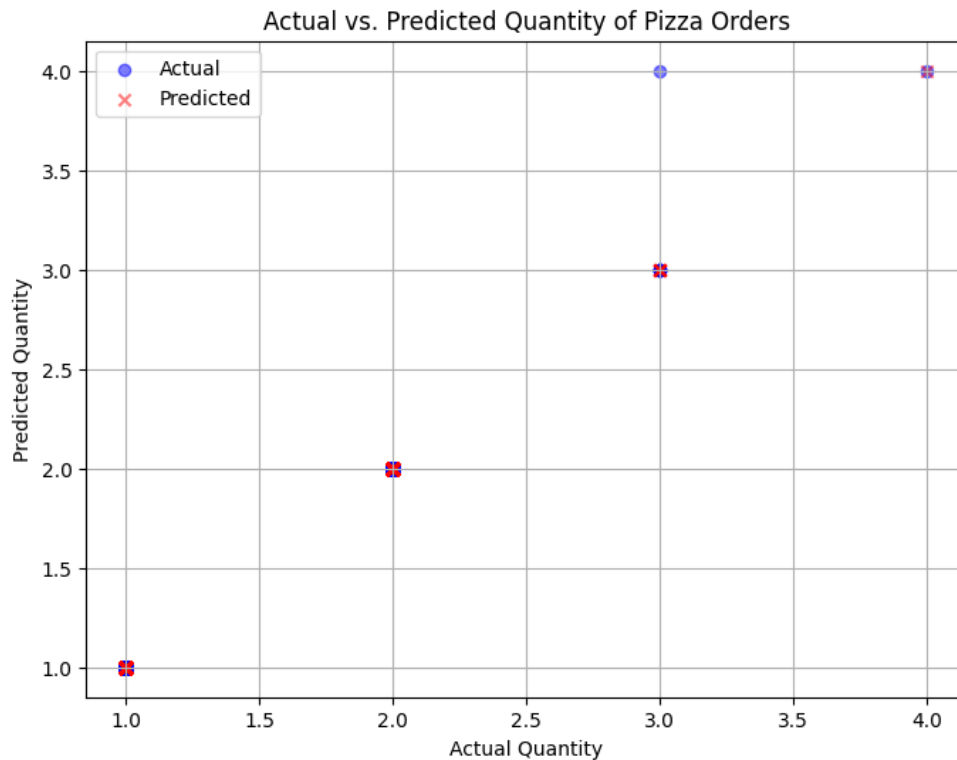
The RMSE for the Decision Tree model is : 0.01014092392893546

The R2 Score for the Decision Tree model is : 0.9952410316644643

The Decision Tree model performs exceptionally well with a very low RMSE (0.0101) and an R2 Score close to 1 (0.9952).

```
# Graph for the above code
import matplotlib.pyplot as plt

# Create a scatter plot of actual vs. predicted values with different colors for actual and predicted points
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, alpha=0.5, c='blue', label='Actual', marker='o') # Actual points in blue circles
plt.scatter(y_test, y_test, alpha=0.5, c='red', label='Predicted', marker='x') # Predicted points in red X's
plt.title('Actual vs. Predicted Quantity of Pizza Orders')
plt.xlabel('Actual Quantity')
plt.ylabel('Predicted Quantity')
plt.legend()
plt.grid(True)
plt.show()
```



This graph represents the actual VS predicted values of the quantity that will be ordered by the customers. In this the actual and predicted values are almost same.

### ▼ To Know the Feature Importance

```
[40] import pandas as pd
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor

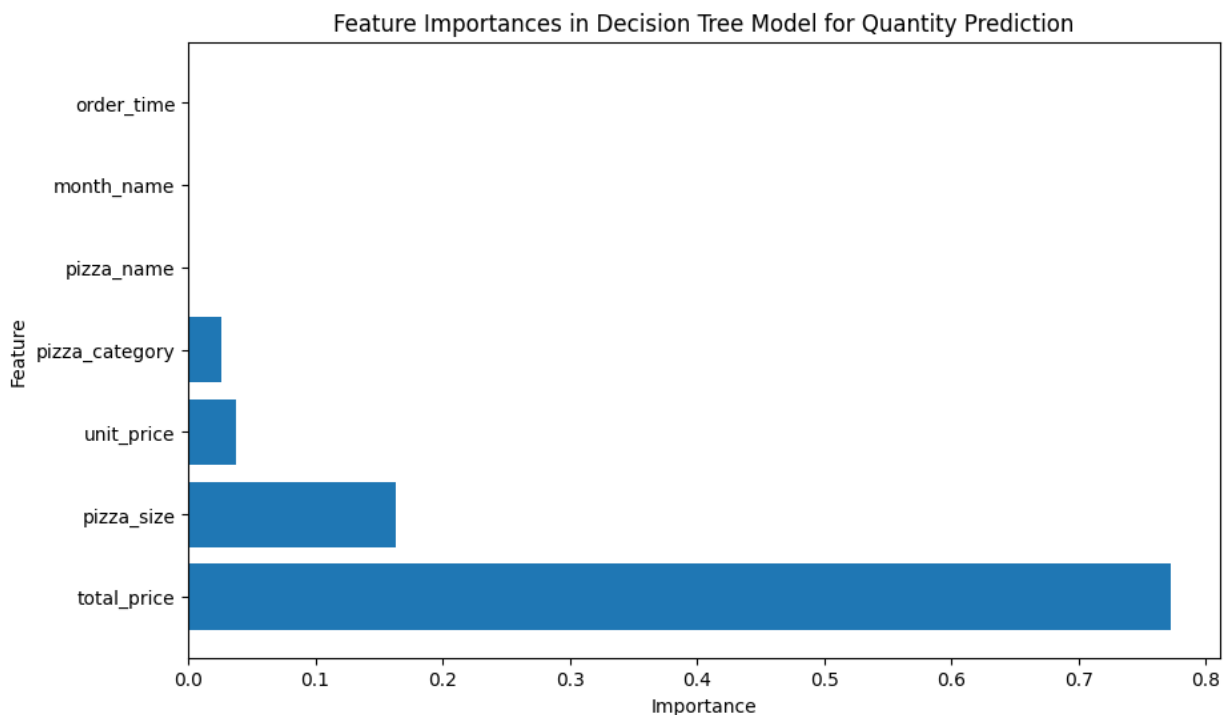
# Define features and target
X = df[['unit_price', 'total_price', 'pizza_category', 'pizza_name', 'month_name', 'pizza_size', 'order_time']]
y = df['quantity']

# Train a Decision Tree model
dct_model = DecisionTreeRegressor(random_state=42)
dct_model.fit(X, y) # Train on the entire dataset

# Get feature importances
feature_importances = dct_model.feature_importances_

# Create a DataFrame to display feature names and their importances
feature_importance_df = pd.DataFrame({'Feature': X.columns, 'Importance': feature_importances})
feature_importance_df = feature_importance_df.sort_values(by='Importance', ascending=False)

# Create a bar plot to visualize feature importances
plt.figure(figsize=(10, 6))
plt.barh(feature_importance_df['Feature'], feature_importance_df['Importance'])
plt.title('Feature Importances in Decision Tree Model for Quantity Prediction')
plt.xlabel('Importance')
plt.ylabel('Feature')
plt.show()
```



The above graph gives us a view that how important are the other features to predict the quantity of Pizzas that will be sold. And `total_price` and `pizza_size` plays an important role.

## Conclusion:

In this context, the decision tree model is the best fit for this specific dataset with very low RMSE and very high  $r^2$  score, and if we look on the EDA part, the customer mostly orders Large sized Classic Thai Chicken Pizzas because these pizzas have bought most revenue to the restaurant so the restaurant must focus on improving the quality of the mentioned and the most sales of the pizzas are on Friday in every month. With these insights the restaurant can improve its sales.