In [ ]:

```
'''LEVEL 03 - TASK 03'''

'''Task: Data Visualization

-->Create visualizations to represent the distribution
of ratings using different charts (histogram, bar
plot, etc.).

-->Compare the average ratings of different cuisines
or cities using appropriate visualizations.

-->Visualize the relationship between various
features and the target variable to gain insights.'''
```

In [1]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import warnings

warnings.filterwarnings("ignore")

data = pd.read_csv('Dataset.csv')
```

In [2]:

```
non_numeric_cols = data.select_dtypes(exclude=[float, int]).columns.to
print("Columns with non-numeric values:")
print(non_numeric_cols)
```

```
Columns with non-numeric values:
['Restaurant Name', 'City', 'Address', 'Locality', 'Locality Verbose', 'C
uisines', 'Currency', 'Has Table booking', 'Has Online delivery', 'Is del
ivering now', 'Switch to order menu', 'Rating color', 'Rating text']
```

In [3]:

```
data.drop(columns=non_numeric_cols, inplace=True)
```

In [4]:

```
X = data.drop(columns=['Aggregate rating'])
y = data['Aggregate rating']
```

In [5]:

```
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.
```

In [6]:

```
# Function to train and evaluate a regression model
def train_and_evaluate_model(model):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    return mse, r2
```

In [7]:

```
# Training and evaluating Linear Regression model
linear_regression = LinearRegression()
mse_lr, r2_lr = train_and_evaluate_model(linear_regression)
```

In [8]:

```
# Training and evaluating Decision Tree Regression model
decision_tree = DecisionTreeRegressor()
mse_dt, r2_dt = train_and_evaluate_model(decision_tree)
```

In [9]:

```
# Training and evaluating Random Forest Regression model
```

In [8]:
```
# Training and evaluating Decision Tree Regression model
decision_tree = DecisionTreeRegressor()
mse_dt, r2_dt = train_and_evaluate_model(decision_tree)
```

In [9]:
```
# Training and evaluating Random Forest Regression model
random_forest = RandomForestRegressor()
mse_rf, r2_rf = train_and_evaluate_model(random_forest)
```

In [10]:
```
# Print the performance metrics
print(f"Linear Regression ---> MSE: {mse_lr:.2f}, R2 Score: {r2_lr:.2
print(f"Decision Tree Regression ---> MSE: {mse_dt:.2f}, R2 Score: {r
print(f"Random Forest Regression ---> MSE: {mse_rf:.2f}, R2 Score: {r
```

```
Linear Regression ---> MSE: 1.58, R2 Score: 0.31

Decision Tree Regression ---> MSE: 0.15, R2 Score: 0.94

Random Forest Regression ---> MSE: 0.08, R2 Score: 0.97
```

In [ ]:
```

```

In [ ]:
```

```