```
import pandas as pd
df=pd.read_csv('/content/Dataset .csv')
df.fillna(df.mean(numeric_only=True), inplace=True)
df.fillna('unknown', inplace=True)
from sklearn.preprocessing import LabelEncoder
categorical_cols = df.select_dtypes(include=['object']).columns
label_encoders = {}
for col in categorical_cols:
    label_encoders[col] = LabelEncoder()
    df[col] = label_encoders[col].fit_transform(df[col])


from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

target_col = 'Aggregate rating'
X = df.drop(columns=[target_col])
y = df[target_col]

numeric_columns = X.select_dtypes(include=['float64', 'int64']).columns

numeric_transformer = StandardScaler()


preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_columns),
    ])

pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(random_state=42))
])

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

pipeline.fit(X_train, y_train)

y_pred = pipeline.predict(X_test)
print("MSE:", mean_squared_error(y_test, y_pred))
print("R-squared:", r2_score(y_test, y_pred))
```

```
MSE: 0.027315152799581344
R-squared: 0.9879991894430911
```

```
from sklearn.metrics.pairwise import cosine_similarity
recommendation_features = ['Cuisines', 'Price range', 'Aggregate rating']

feature_matrix = df[recommendation_features].values
similarity_matrix = cosine_similarity(feature_matrix)
restaurant_index = 0
similar_indices = similarity_matrix[restaurant_index].argsort()[::-1][1:6]
print("Recommended restaurants:")
print(df.iloc[similar_indices][['Restaurant Name', 'Cuisines', 'Aggregate rating']])
```

```
Recommended restaurants:
      Restaurant Name  Cuisines  Aggregate rating
361              2942       924               4.8
8821             6613       626               3.3
1962             1878       625               3.3
3784             1524       604               3.1
4966              788       588               3.1
```

```
import folium

map_center = [df['Latitude'].mean(), df['Longitude'].mean()]

restaurant_map = folium.Map(location=map_center, zoom_start=12)

for _, row in df.iterrows():
    folium.Marker(
        location=[row['Latitude'], row['Longitude']],
        popup=f"{row['Restaurant Name']}, Rating: {row['Aggregate rating']}"
    ).add_to(restaurant_map)

restaurant_map.save("restaurant_map.html")

grouped_data = df.groupby('City').agg({
    'Aggregate rating': 'mean',
    'Price range': 'mean',
    'Restaurant Name': 'count'
}).rename(columns={'Restaurant Name': 'Number of Restaurants'})

print(grouped_data)
```

```
      Aggregate rating  Price range  Number of Restaurants
City
0             4.300000     3.300000                     20
1             3.965000     2.650000                     20
2             4.161905     2.571429                     21
3             3.555000     1.700000                     20
4             3.395000     2.650000                     20
...                ...          ...                    ...
136           3.900000     2.000000                      1
137           4.250000     3.250000                     20
138           3.200000     2.000000                      1
```

```
139           3.300000     2.000000                   1
140           4.292857     2.857143                  14

[141 rows x 3 columns]
```