In [ ]:

```
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, LabelEncoder
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusior
```

In [ ]:

```
df = pd.read_csv("pizza_sales.csv")
```

In [ ]:

```
features = ['quantity', 'unit_price', 'total_price']
target = 'pizza_category'  # Assuming this column exists
df = df.dropna(subset=[target])  # Drop rows with missing target values
```

In [ ]:

```
label_encoder = LabelEncoder()
df[target] = label_encoder.fit_transform(df[target])
```

In [ ]:

```
X = df[features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar
```

In [ ]:

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

In [ ]:

```
svm_model = SVC(kernel='rbf', C=1.0, gamma='scale')  # Using RBF kernel
svm_model.fit(X_train, y_train)
```

Out[7]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

In [ ]:

```
y_pred = svm_model.predict(X_test)
```

In [ ]:

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test, y_pred))
```

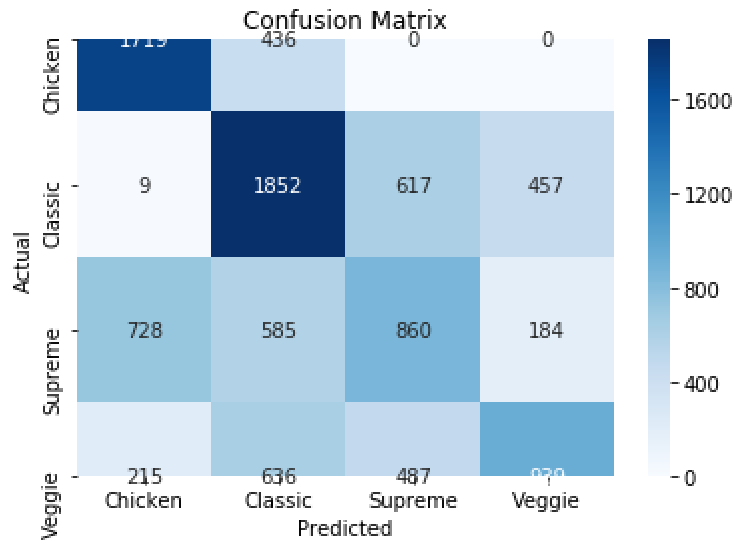
```
Accuracy: 0.5522418757712876
Classification Report:
               precision    recall  f1-score   support

           0       0.64      0.80      0.71      2155
           1       0.53      0.63      0.57      2935
           2       0.44      0.36      0.40      2357
           3       0.59      0.41      0.49      2277

    accuracy                           0.55      9724
   macro avg       0.55      0.55      0.54      9724
weighted avg       0.55      0.55      0.54      9724
```

In [ ]:

```
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6,4))
sns.heatmap(conf_matrix, annot=True, cmap='Blues', fmt='d', xticklabels=labe
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```



In [ ]: