```
import pandas as pd
from datetime import datetime
from sklearn.model_selection import train_test_split
from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Step 2: Load Dataset
df = pd.read_excel("/content/Online Retail.xlsx")
df.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
[illegible]	[illegible]	[illegible]	KNITTED UNION FLAG HOT WATER	[illegible]	2010-12-01 [illegible]	[illegible]	[illegible]	United [illegible]

```
df.dropna(subset=['CustomerID'], inplace=True)
df = df[~df['InvoiceNo'].astype(str).str.startswith('C')]
df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
```

```
<ipython-input-5-4b292717a725>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  df['TotalPrice'] = df['Quantity'] * df['UnitPrice']
```

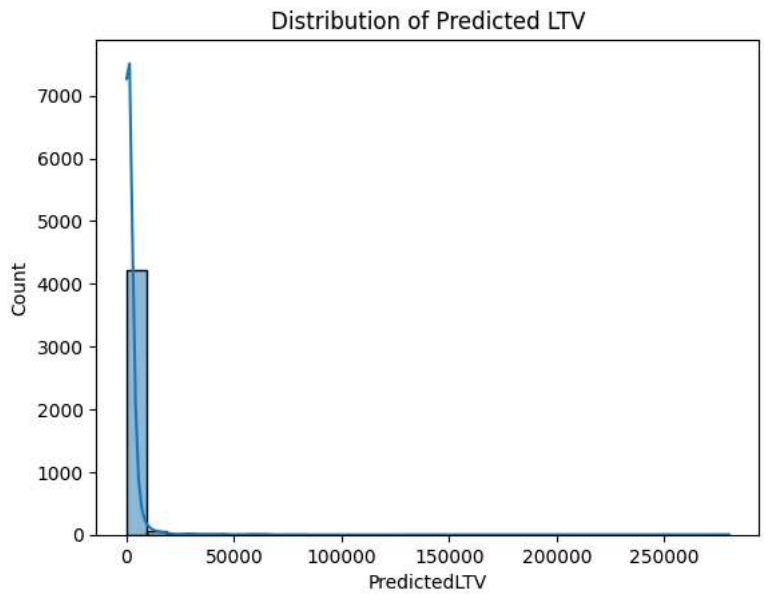
```
snapshot = df['InvoiceDate'].max() + pd.Timedelta(days=1)
rfm = df.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (snapshot - x.max()).days,
    'InvoiceNo': 'nunique',
    'TotalPrice': 'sum'
}).rename(columns={'InvoiceDate': 'Recency', 'InvoiceNo': 'Frequency', 'TotalPrice': 'Monetary'})

rfm['AOV'] = rfm['Monetary'] / rfm['Frequency']

```

```
X = rfm[['Recency', 'Frequency', 'AOV']]
y = rfm['Monetary']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = XGBRegressor()
model.fit(X_train, y_train)

# Step 6: Evaluation
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print("MAE:", mae)
print("RMSE:", rmse)
```

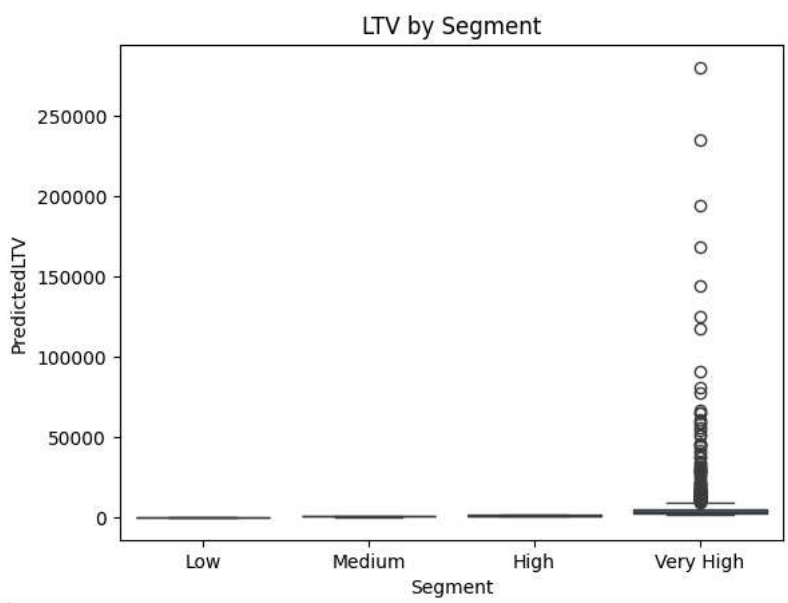
```
MAE: 208.95508759955652
RMSE: 1863.0781089577395
```

```
rfm['PredictedLTV'] = model.predict(X)
rfm['Segment'] = pd.qcut(rfm['PredictedLTV'], 4, labels=['Low', 'Medium', 'High', 'Very High'])

```

```
sns.histplot(rfm['PredictedLTV'], bins=30, kde=True)
plt.title("Distribution of Predicted LTV")
plt.show()
```




```
sns.boxplot(x='Segment', y='PredictedLTV', data=rfm)
plt.title("LTV by Segment")
plt.show()

```




```
rfm.reset_index()[['CustomerID', 'PredictedLTV', 'Segment']].to_csv("ltv_predictions.csv", index=False)
```