Step 1: Importing Libraries and Loading Data

In [3]:
```
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.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

Step 2: Data Preprocessing

In [7]:
```
# Load the dataset
data_path = r'C:\Users\Hello\Desktop\Sunil Customer Churn Prediction for T-Mobile.c
data = pd.read_csv(data_path)
```

In [9]:
```
# Step 1: Data Overview and Preprocessing
print("Data Overview:")
print(data.info())
```

```
Data Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 250000 entries, 0 to 249999
Data columns (total 13 columns):
 #   Column                  Non-Null Count   Dtype  
---  ------                  --------------   -----  
 0   Customer ID             250000 non-null  int64  
 1   Purchase Date           250000 non-null  object 
 2   Product Category        250000 non-null  object 
 3   Product Price           250000 non-null  int64  
 4   Quantity                250000 non-null  int64  
 5   Total Purchase Amount   250000 non-null  int64  
 6   Payment Method          250000 non-null  object 
 7   Customer Age            250000 non-null  int64  
 8   Returns                 202404 non-null  float64
 9   Customer Name           250000 non-null  object 
 10  Age                     250000 non-null  int64  
 11  Gender                  250000 non-null  object 
 12  Churn                   250000 non-null  int64  
dtypes: float64(1), int64(7), object(5)
memory usage: 24.8+ MB
None
```

In [ ]:
```
(   Customer ID     Purchase Date Product Category  Product Price  Quantity  \
 0        46251     9/8/2020 9:38      Electronics             12         3   
 1        46251    3/5/2022 12:56             Home            468         4   
 2        46251   5/23/2022 18:18             Home            288         2   
 3        46251  11/12/2020 13:13         Clothing            196         1   
 4        13593  11/27/2020 17:55             Home            449         1   

    Total Purchase Amount Payment Method  Customer Age  Returns  \
 0                    740    Credit Card            37      0.0   
 1                   2739         PayPal            37      0.0   
```

```
2                   3196         PayPal            37      0.0
3                   3509         PayPal            37      0.0
4                   3452    Credit Card            49      0.0

         Customer Name  Age  Gender  Churn
0  Christine Hernandez   37    Male      0
1  Christine Hernandez   37    Male      0
2  Christine Hernandez   37    Male      0
3  Christine Hernandez   37    Male      0
4          James Grant   49  Female      1  ,
None)
```

In [11]:
```
# Drop duplicates and unnecessary columns
data.drop_duplicates(inplace=True)
data.drop(columns=['Customer Name', 'Age'], inplace=True)  # 'Age' is duplicate of
```

In [24]:
```
# Handle missing values in 'Returns' column (replace with 0, assuming no returns)
data['Returns'].fillna(0, inplace=True)
```

```
C:\Users\Hello\AppData\Local\Temp\ipykernel_3240\1081060226.py:2: FutureWarning: A v
alue is trying to be set on a copy of a DataFrame or Series through chained assignme
nt using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  data['Returns'].fillna(0, inplace=True)
```

In [17]:
```
# Convert 'Purchase Date' to datetime format
data['Purchase Date'] = pd.to_datetime(data['Purchase Date'])
```

Step 3: Feature Engineering

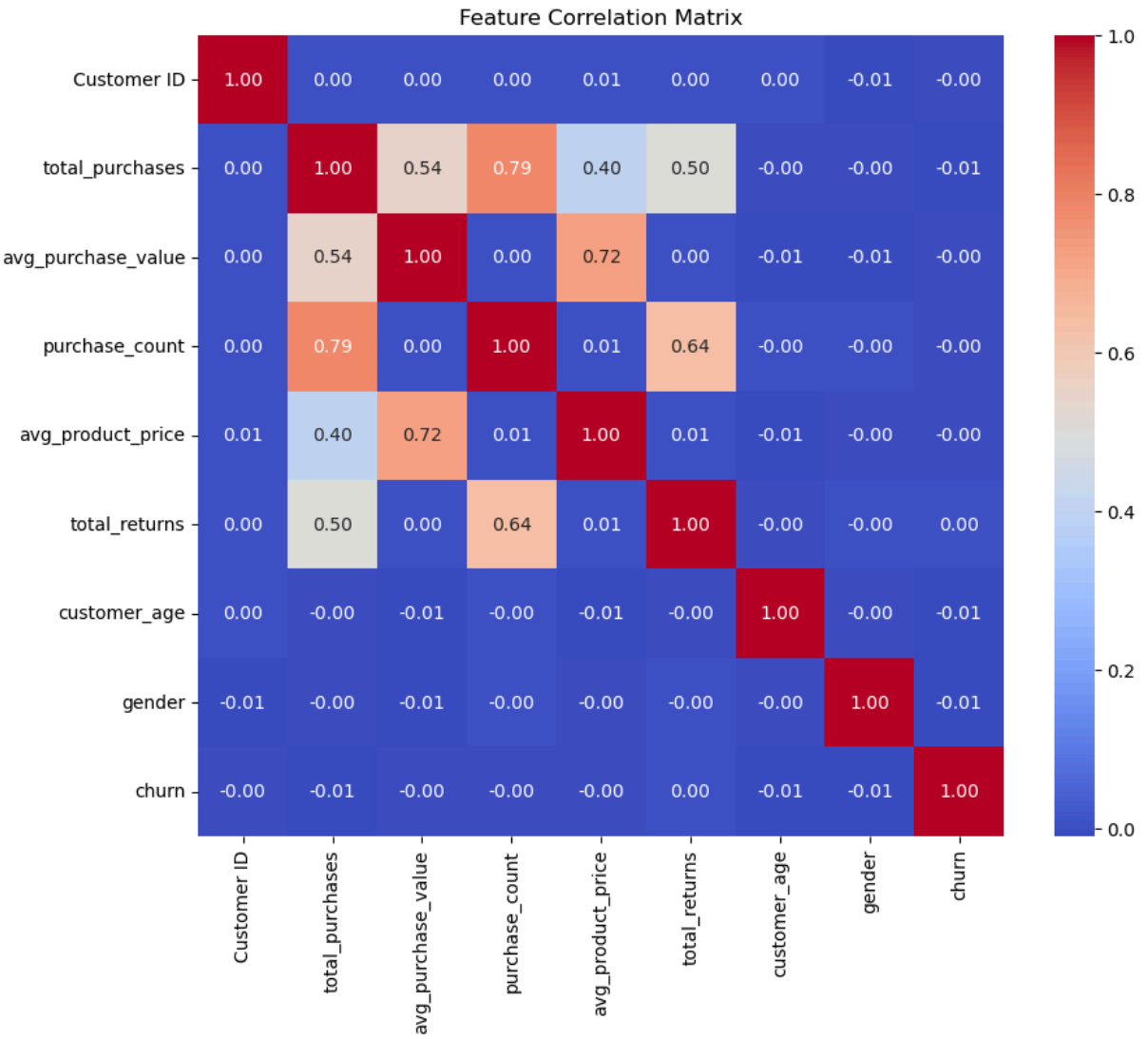
In [28]:
```
# Derive 'Total Spend' for each customer
data['Total Spend'] = data['Product Price'] * data['Quantity']
```

In [30]:
```
# Aggregate data by Customer ID for CLV features
clv_features = data.groupby('Customer ID').agg(
    total_purchases=('Total Spend', 'sum'),
    avg_purchase_value=('Total Spend', 'mean'),
    purchase_count=('Purchase Date', 'count'),
    avg_product_price=('Product Price', 'mean'),
    total_returns=('Returns', 'sum'),
    customer_age=('Customer Age', 'first'),
    gender=('Gender', 'first'),
    churn=('Churn', 'first')
).reset_index()
```

In [32]:
```
# Encode categorical variables (Gender)
clv_features['gender'] = clv_features['gender'].map({'Male': 0, 'Female': 1})
```

Step 3: Statistical Analysis

In [35]:
```
# Analyze correlation between features
plt.figure(figsize=(10, 8))
sns.heatmap(clv_features.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Feature Correlation Matrix')
plt.show()
```



Step 4: Predictive Modeling

In [38]:
```
# Define features (X) and target (y)
X = clv_features.drop(columns=['Customer ID', 'churn'])
y = clv_features['churn']
```

In [40]:
```
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
```

In [42]:
```
# Train a Random Forest Regressor
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)
```

Out[42]:
```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

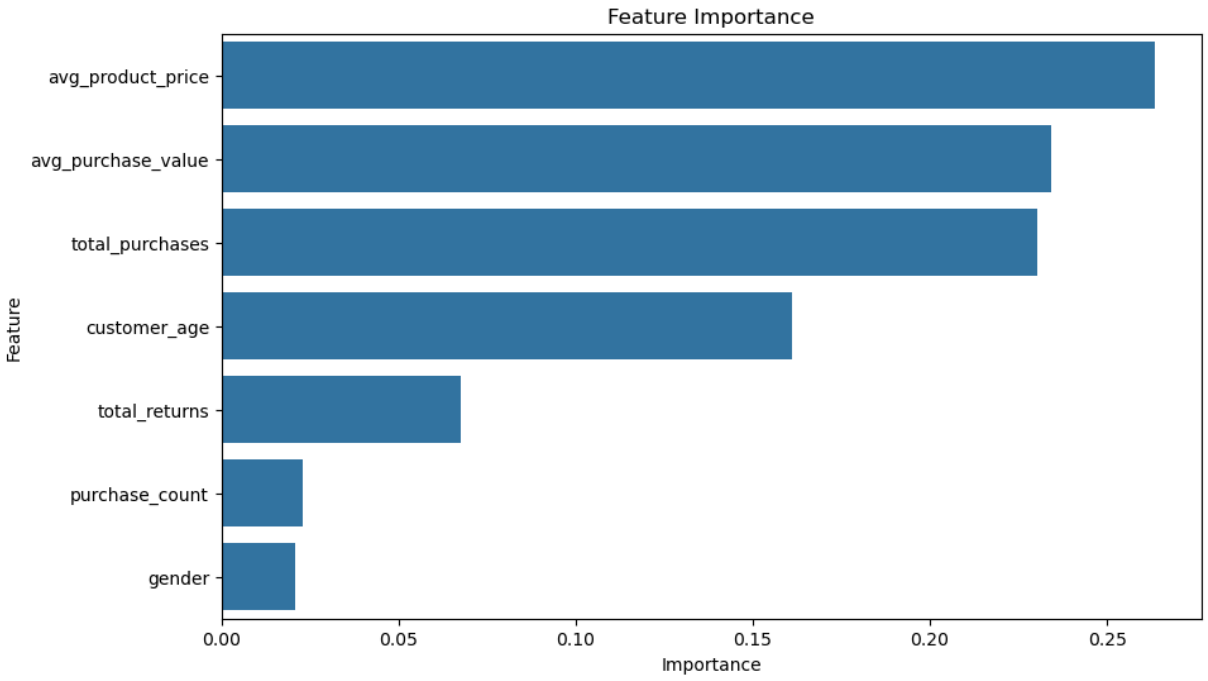
In [43]:
```
# Evaluate the model
y_pred = model.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
```

In [44]:
```
print(f"Model Performance:\nRMSE: {rmse:.2f}\nR^2: {r2:.2f}")
```

```
Model Performance:
RMSE: 0.42
R^2: -0.07
```

In [48]:
```
# Feature Importance
feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': model.feature_importances_
}).sort_values(by='Importance', ascending=False)
```

In [50]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=feature_importance)
plt.title('Feature Importance')
plt.show()
```



In [ ]: