PREDICTIVE MODELING FOR REVENUE GROWTH

USING ONLINE RETAIL DATASET

Adwaid R

adwaid.dhwani@gmail.com

02/05/2024

PROJECT OBJECTIVE

Develop a predictive model to forecast monthly revenue growth for a UK-based online fashion retailer using historical sales data.

In today's tough retail world, knowing how much money a store will make in the future is super important. That's why we're working on a project to predict how much money a UK online fashion store will make each month. We're using fancy math and data analysis to create a smart system that looks at past sales to guess how much money the store will make in the future. Our goal is to give the store owners helpful information so they can make smart choices about how to run their business and where to spend their money.

Revenue forecasting helps businesses in many ways:

- Guides budgeting, inventory management, and marketing strategies.
- Helps businesses adapt strategies, find new opportunities, and avoid risks.
- Using data for forecasting improves customer experiences, product choices, and profits.

DATASET OVERVIEW

Online Retail II dataset from UCI Machine Learning Repository.

This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail.The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers.

	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
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

There are 541909 rows and 8 columns in the dataset

DATA CLEANING

Data cleaning is a crucial step in the machine learning , as it involves identifying and removing any missing, duplicate, or irrelevant data. The goal of data cleaning is to ensure that the data is accurate, consistent, and free of errors, as incorrect or inconsistent data can negatively impact the performance of the ML model.

```
[6]:  df.isnull().sum()

[6]:  InvoiceNo           0
      StockCode           0
      Description      1454
      Quantity            0
      InvoiceDate         0
      UnitPrice           0
      CustomerID     135080
      Country             0
      dtype: int64
```

We're missing more than 130,000 values in the CustomerID, which is about 25% of all the data & about 1500 values in description . If we just get rid of these and carry on, our analysis and predictions might not be very good. Each customer uses the same InvoiceNo every time they buy something in a certain period. So, I'm giving new IDs to customers who don't have one yet. I'm checking the InvoiceNo to see if it belongs to the same customer or a different one. If it matches, they'll get the same ID, and if not, they'll get a new one. And the most frequent values will be added to the description.

DATA CLEANING

Inserting values to missing columns

```
unique_customer_ids = df['CustomerID'].dropna().unique()  # Droping NaN values

new_customer_id = int(unique_customer_ids.max()) + 1
prev_invoice_no = None  # Storing the previous InvoiceNo
for index, row in df.iterrows():
    if pd.isnull(row['CustomerID']):  # Check if CustomerID is missing
        if row['InvoiceNo'] == prev_invoice_no:
            df.at[index, 'CustomerID'] = df.loc[index - 1, 'CustomerID']  # Assign CustomerID of same invoiceno
        else:
            new_customer_id += 1
            df.at[index, 'CustomerID'] = new_customer_id  # Create a new CustomerID

    prev_invoice_no = row['InvoiceNo']  # Update the previous InvoiceNo
```

```
df['Description'] = df['Description'].fillna(df['Description'].mode()[0])
```

Removing duplicate & negative values

```
df.drop_duplicates(inplace=True)
```

```
negative_qty = df[df['Quantity'] < 0]
negative_unitprice= df[df['UnitPrice']<0]
print(f"\nNumber of records with negative Quantity: {negative_qty.shape[0]}")
print(f"\nNumber of records with negative Unit Price: {negative_unitprice.shape[0]}")

# Remove rows with values less than zero
df = df[df['Quantity'] > 0]
df = df[df['UnitPrice']> 0]
```

DATA CLEANING

Added a new column which stores the total amount for that transaction

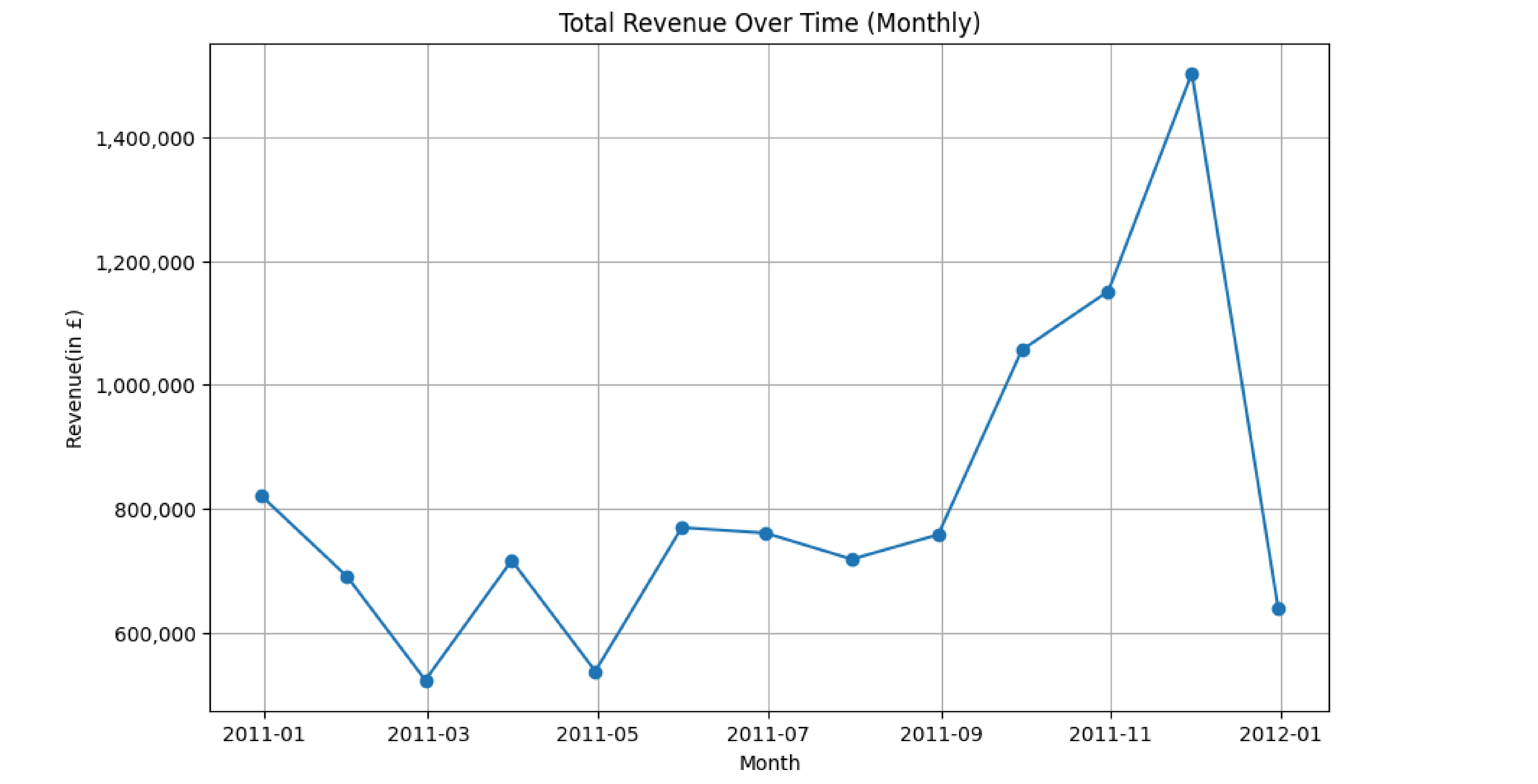
```
df['TotalAmount'] = df['Quantity'] * df['UnitPrice']
```

```
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])

df['Date'] = df['InvoiceDate'].dt.date
df['Year'] = df['InvoiceDate'].dt.year
df['Month'] = df['InvoiceDate'].dt.month
df['Day'] = df['InvoiceDate'].dt.day
df['Weekday'] = df['InvoiceDate'].dt.weekday
df['Date'] = pd.to_datetime(df['Date'])
```

Extracted the date,year,month,day and weekday from Invoice Date

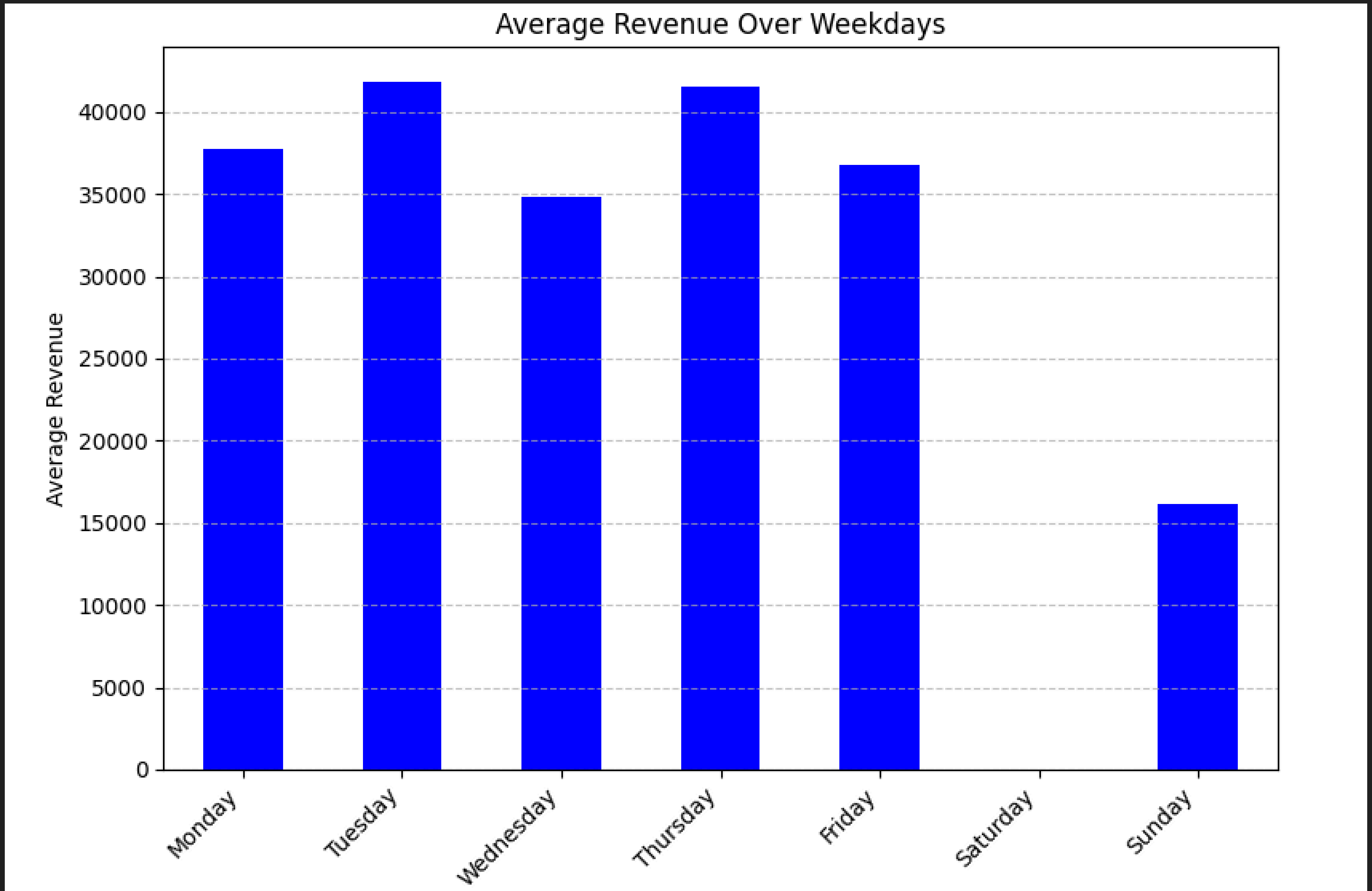
This is useful for analyzing the sales in different time periods

EXPLORATORY DATA ANALYSIS



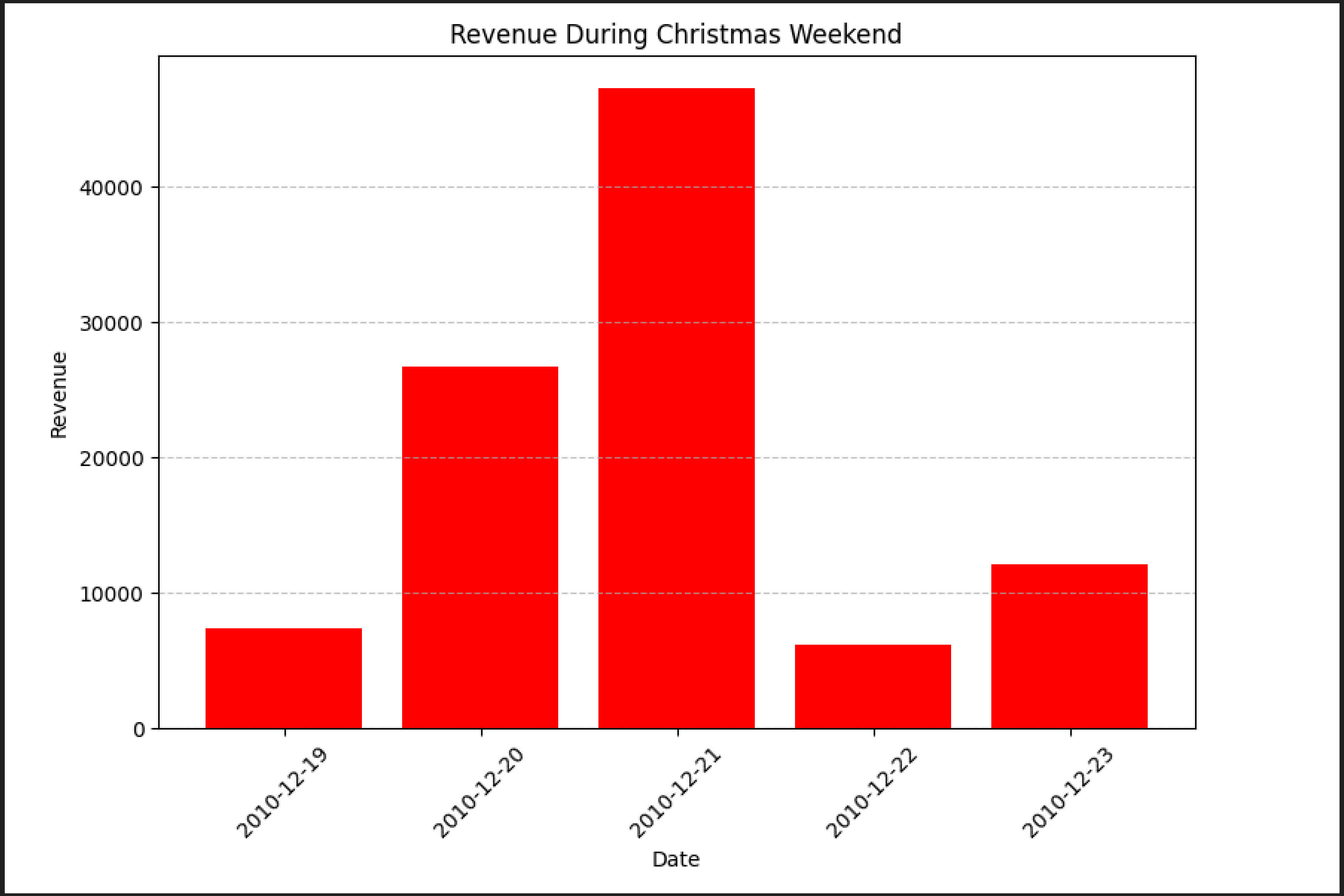
Based on the graph, there are clear seasonal patterns in the business's revenue. The most profitable period begins in September 2011, reaching its peak after November 2011. However, from our available data, we can anticipate a decline in revenue following December.

EXPLORATORY DATA ANALYSIS



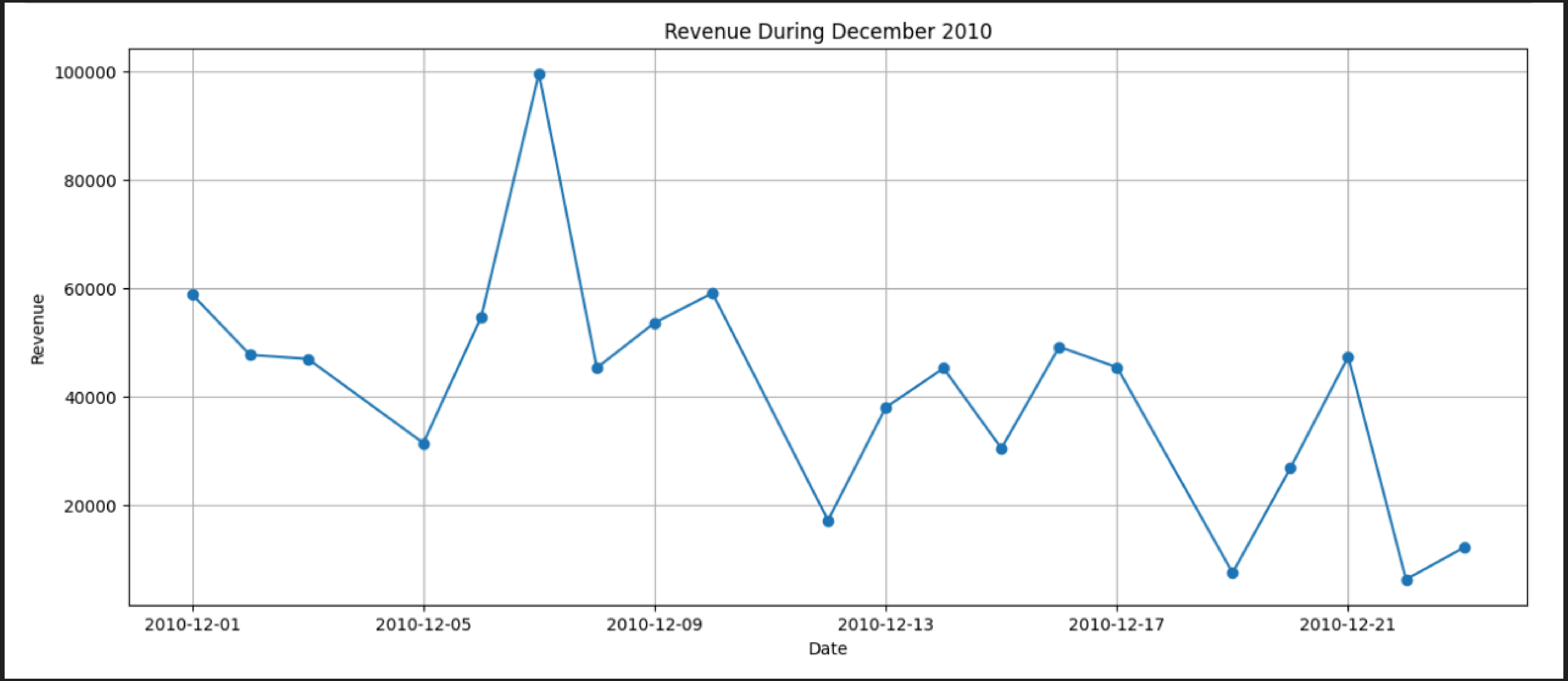
The graph clearly indicates that the retail store is closed on Saturdays. Sundays consistently show the lowest revenue. On weekdays, excluding Saturdays and Sundays, the average revenue exceeds £30,000. Specifically, Tuesdays and Thursdays demonstrate the highest average revenue, surpassing £40,000.

EXPLORATORY DATA ANALYSIS



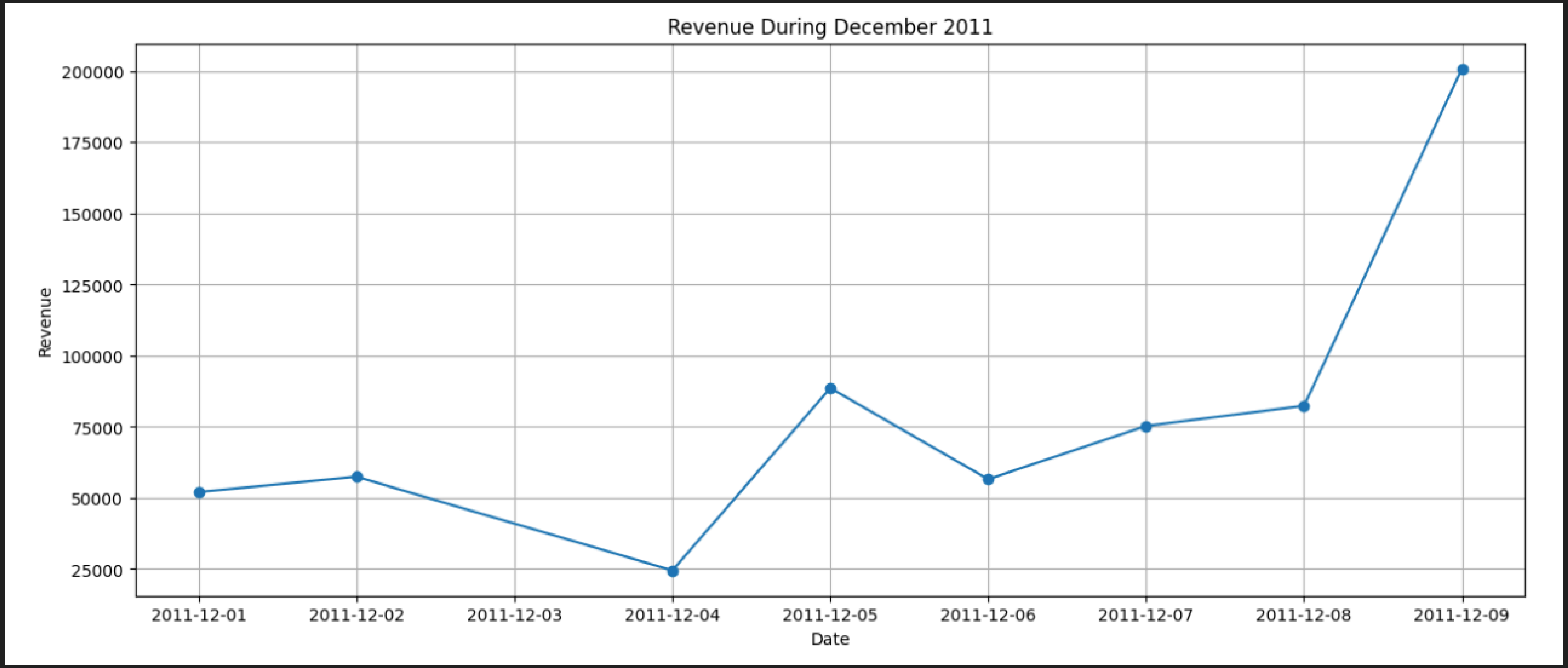
The graph shows that the shop was closed from December 24th, 2010, to January 1st, 2011. During this period, total sales were below the average weekly sales. With the exception of December 21st, each day experienced sales below the average.

EXPLORATORY DATA ANALYSIS



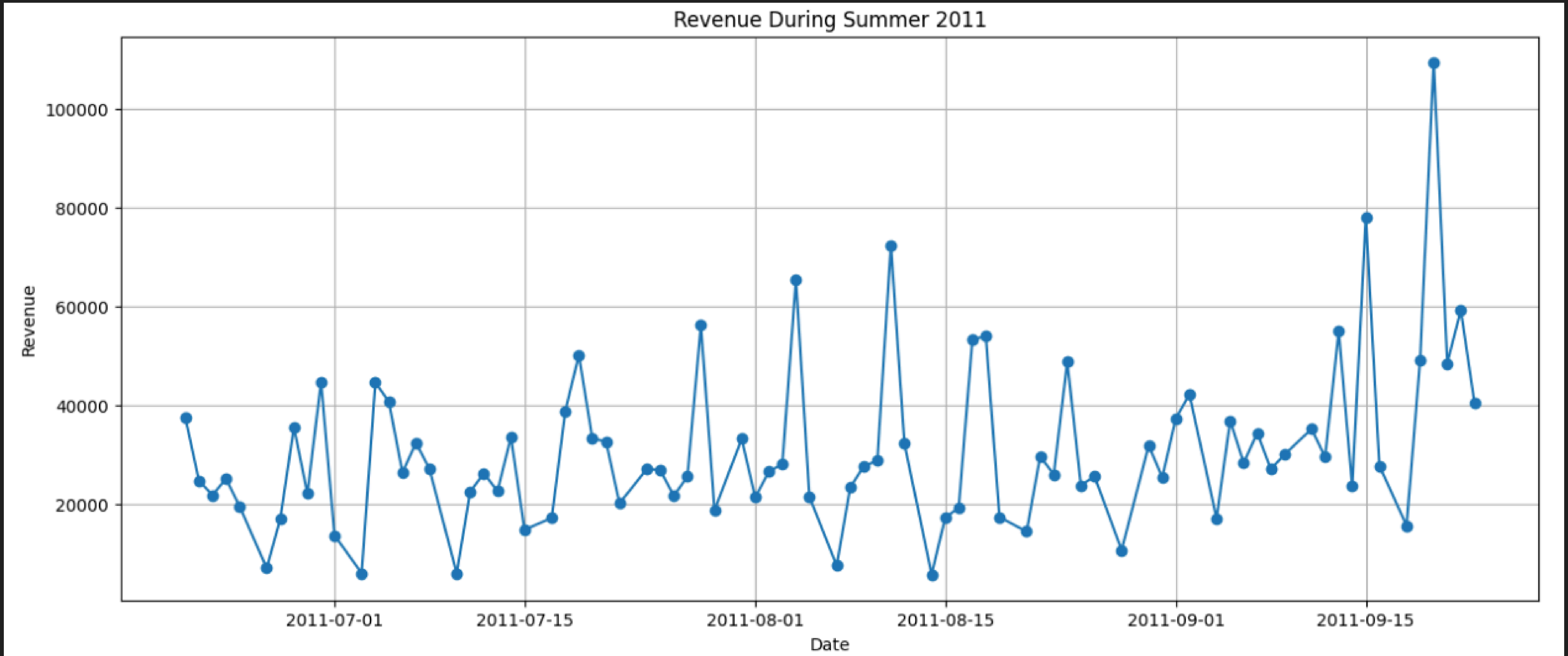
When analyzing December 2010 sales, we observe an initial spike in sales after the first week, followed by a decline to average levels. The sales during Christmas, however, were notably below average. Despite this, there is a distinct peak in sales performance following the first week. This suggests that customers may have been purchasing items well in advance, possibly due to an average delivery time of about two weeks. This behavior shows that customers were being careful to avoid possible delays in getting their products.

EXPLORATORY DATA ANALYSIS

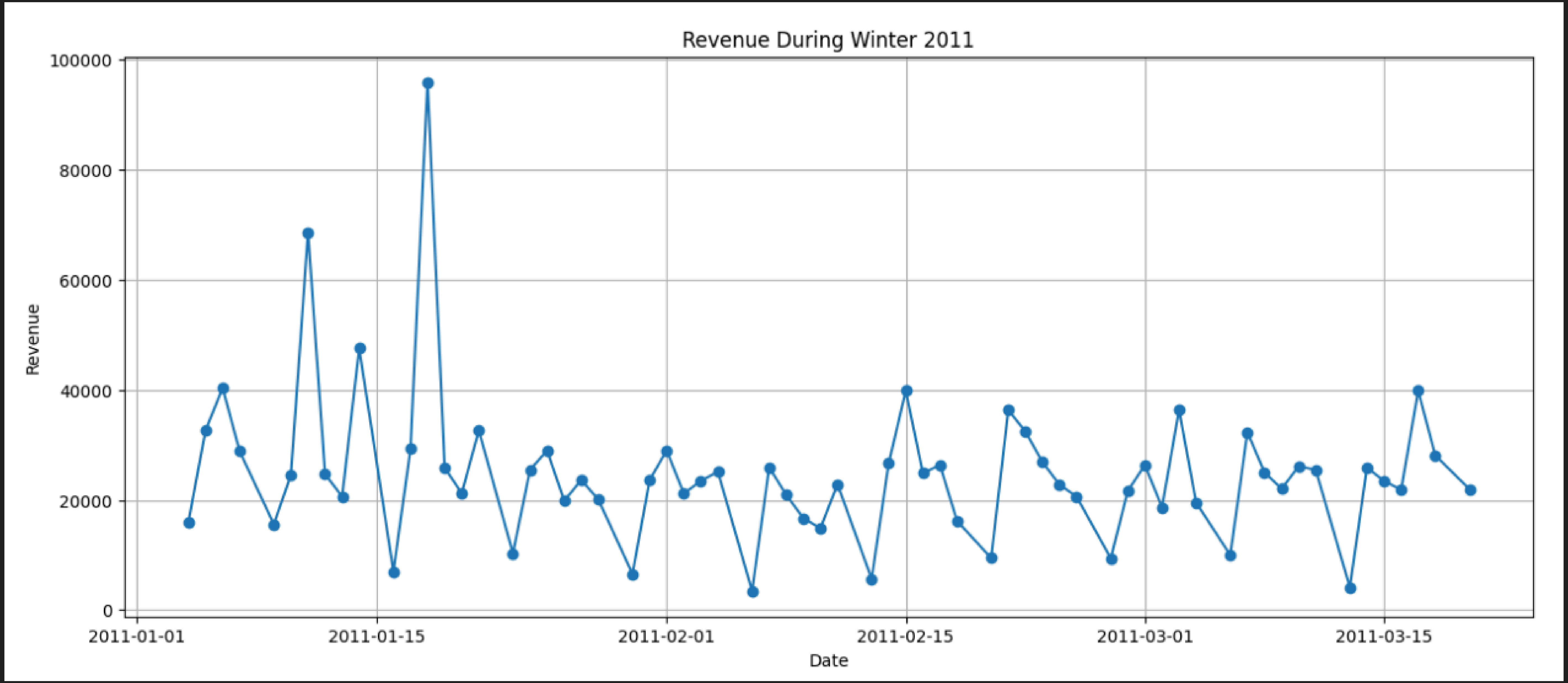


We observe similar patterns in December 2011. However, our data only covers until December 9th. Sales peak above average midweek, reaching a peak of £200,000 on December 9th.

EXPLORATORY DATA ANALYSIS



EXPLORATORY DATA ANALYSIS



During the winter season, sales remain relatively flat, with few spikes observed except at the beginning of the year. They typically remain below average during this time. In contrast, summer sales are more consistent, often surpassing average levels, and tend to peak towards the end of the season.

EXPLORATORY DATA ANALYSIS

Top Selling Products

Product Name	Quantity Sold
PAPER CRAFT , LITTLE BIRDIE	80,995
MEDIUM CERAMIC TOP STORAGE JAR	78,033
WORLD WAR 2 GLIDERS ASSTD DESIGNS	54,951
JUMBO BAG RED RETROSPOT	48,371
WHITE HANGING HEART T-LIGHT HOLDER	37,872
POPCORN HOLDER	36,749
PACK OF 72 RETROSPOT CAKE CASES	36,396
ASSORTED COLOUR BIRD ORNAMENT	36,362
RABBIT NIGHT LIGHT	30,739
MINI PAINT SET VINTAGE	26,633

Most Profitable Categories

Item Name	Revenue (£)
DOTCOM POSTAGE	206,248.77
Manual	74,101.28
POSTAGE	34,992.23
REGENCY CAKESTAND 3 TIER	28,065.76
AMAZON FEE	13,761.09
Adjust bad debt	11,062.06
PARTY BUNTING	9,850.68
SET OF 3 CAKE TINS PANTRY DESIGN	8,120.53
CREAM SWEETHEART MINI CHEST	7,497.46
WHITE HANGING HEART T-LIGHT HOLDER	7,437.57

EXPLORATORY DATA ANALYSIS

Product Metrics

	Description	StockTurnRate	AveragePrice	SalesVolume
0	4 PURPLE FLOCK DINNER CANDLES	3.641026	2.450513	142
1	50'S CHRISTMAS GIFT BAG LARGE	14.844961	1.426589	1915
2	DOLLY GIRL BEAKER	13.926136	1.506420	2451
3	I LOVE LONDON MINI BACKPACK	4.459770	4.616667	388
4	I LOVE LONDON MINI RUCKSACK	1.000000	4.150000	1
5	NINE DRAWER OFFICE TIDY	1.757576	16.090606	58
6	OVAL WALL MIRROR DIAMANTE	1.512821	10.774936	236
7	RED SPOT GIFT BAG LARGE	16.631068	1.377184	1713
8	SET 2 TEA TOWELS I LOVE LONDON	10.156934	3.557847	2783
9	SPACEBOY BABY GIFT SET	2.706522	15.913424	498

Customer Metrics

	CustomerID	AveragePurchaseFrequency	AverageSpendPerVisit	TotalSpend
0	12346.0	NaT	1.040000	1.04
1	12347.0	2 days 00:24:10.276243093	2.644011	481.21
2	12348.0	9 days 10:12:08	5.764839	178.71
3	12349.0	0 days 00:00:00	8.289041	605.10
4	12350.0	0 days 00:00:00	3.841176	65.30

FORECASTING

Since we only have data for 12 months, typical seasonal patterns may not apply. Because of our small dataset , I'm using machine learning to predict future outcomes.

```
data_sorted = df.sort_values(by='InvoiceDate')
country_encoder = OneHotEncoder()
country_encoded = country_encoder.fit_transform(data_sorted[['Country']]).toarray()

country_encoded_df = pd.DataFrame(country_encoded, columns=country_encoder.get_feature_names_out(['Country']))

data_sorted = pd.concat([data_sorted, country_encoded_df], axis=1)
```

I'm using one-hot encoding to handle the "Country" column, which contains categories like different countries. Since our data involves transactions across countries, it's important to keep this information for accurate forecasting.

```
X = data_sorted.drop(['TotalAmount', 'InvoiceNo', 'Description', 'InvoiceDate','Date','StockCode', 'Country'], axis=1)
y = data_sorted['TotalAmount']

train_size = int(0.8 * len(data_sorted))  # 80% for training
X_train, X_test = X[:train_size], X[train_size:]
y_train, y_test = y[:train_size], y[train_size:]
```

Using time-based splitting for training and testing sets for the chronological order of transactions.

FORECASTING

```
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 5, 10]
}

rf_regressor = RandomForestRegressor(random_state=42)
grid_search = GridSearchCV(estimator=rf_regressor, param_grid=param_grid, cv=5, scoring='r2', n_jo
bs=-1)
grid_search.fit(X_train, y_train)


best_params = grid_search.best_params_
print("Best parameters:", best_params)

best_rf_model = grid_search.best_estimator_

train_score = best_rf_model.score(X_train, y_train)
test_score = best_rf_model.score(X_test, y_test)

print("Train R^2 Score:", train_score)
print("Test R^2 Score:", test_score)
```

Using Gridsearch method to find the best parameters for the machine learning model

```
Best parameters: {'max_depth': None, 'min_samples_split': 2, 'n_estimators': 300}
Train R^2 Score: 0.9117258639762632
Test R^2 Score: 0.9914038847528889
```

FORECASTING

Random Forest

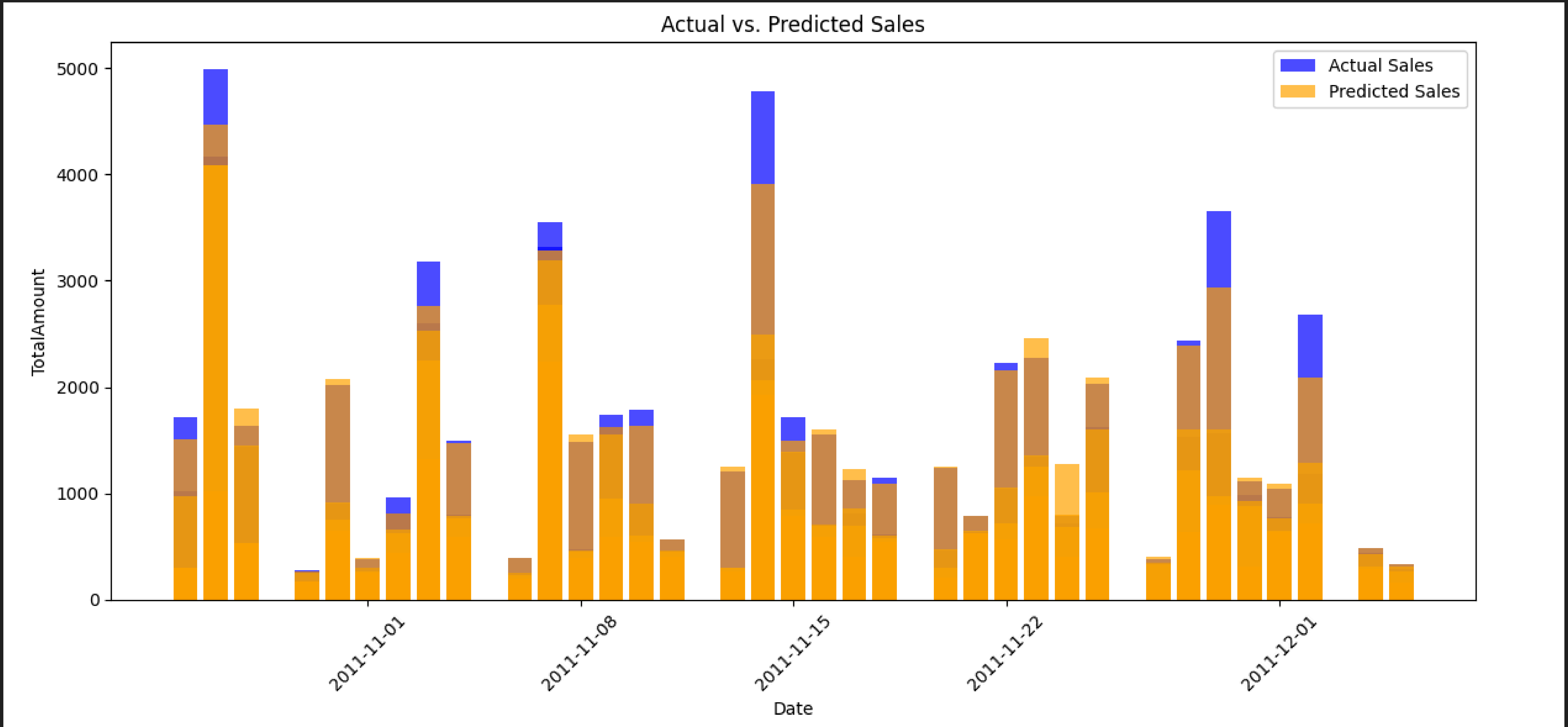
```
rf_model = RandomForestRegressor(**best_params, random_state=42)
rf_model.fit(X_train, y_train)

train_score = rf_model.score(X_train, y_train)
test_score = rf_model.score(X_test, y_test)

print("Train R^2 Score:", train_score)
print("Test R^2 Score:", test_score)

y_pred = rf_model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f'Root Mean Squared Error (RMSE): {rmse}')
```

```
Train R^2 Score: 0.9117258639762632
Test R^2 Score: 0.9914038847528889
Root Mean Squared Error (RMSE): 6.209527891944454
```

The Random Forest model achieved strong performance with a Train R^2 score of 0.912 and Test R^2 score of 0.991, indicating robust predictive capability. Root Mean Squared Error (RMSE) of 6.21 suggests that the model's predictions closely align with the actual sales data, showcasing its effectiveness in capturing underlying patterns and trends.

The numbers we got show that it's almost always right, with just a tiny bit of error. So, we can trust its forecasts to guide our decisions confidently.

FORECASTING



FORECASTING

- The accuracy of our sales predictions underscores the reliability and strength of our machine learning model.
- This precision in forecasting highlights the effectiveness of the chosen machine learning approach.
- Furthermore, the model consistently delivers dependable forecasts, enabling enhanced inventory management, more effective marketing strategies, and improved financial planning.
- Anticipating market trends and customer demand has become significantly easier, providing a competitive advantage for the company.

THANKS

Adwaid R