The Luxury Beauty and Cosmetics Popup Events KPI

Performance Analysis and Predictive Modeling of Luxury Cosmetics Pop-Up Events

Nana Firdausi Hassan

Table of Contents

Project Title

Performance Analysis and Predictive Modeling of Luxury Cosmetics Pop-Up Events

Dataset

The Luxury Beauty and Cosmetics Popup Events KPI dataset is a publicly available dataset sourced from Kaggle and created by Pratyush Puri. It contains detailed information about 2,133 popup events hosted by luxury cosmetic brands across different cities and regions. The dataset includes 15 columns describing each event, capturing brand identity, event location, pricing, customer foot traffic, revenue performance, lease duration, and sales outcomes. It was created to help analysts study how different factors such as brand, region, event type, or customer activity influence product sales and overall event success. Because the dataset is clean, well structured, and rich in meaningful information, it is useful for exploratory analysis, forecasting, predictive modeling, and understanding key business drivers for popup retail events.

Here is the link to the data:
Link

Step 1

Objectives
The main objectives are to analyze pop-up performance, identify the key drivers of sales, and predict units sold using machine learning. The project also aims to classify events with high sell-through to understand inventory efficiency. Another objective is to uncover trends in revenue, footfall, and pricing to guide strategic decision-making for future events.

Problem Statement

Pop-up event performance varies widely across cities, brands, and event types, making it difficult for companies to plan effectively. There is limited understanding of how pricing, footfall, location, and event format influence sales and inventory turnover. Without clear insights, brands struggle to optimize inventory, pricing, and location selection for future events.

Proposed Solution

The project uses a full analytical pipeline that includes data cleaning, feature engineering, and exploratory data analysis. Visualizations are used to compare performance across cities, brands, and price levels. A Random Forest regression model predicts units sold, while a classification model identifies high sell-through events. The solution provides insights that support better planning, pricing strategies, and event optimization decisions.

Step 2

Get Data

```
[1]: pip install wordcloud
```

```
Requirement already satisfied: wordcloud in c:\users\nanaf\downloads\anaconda\lib\site-packages (1.9.4)
Requirement already satisfied: numpy>=1.6.1 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\nanaf\downloads\anaconda\lib\site-packages (from wordcloud) (10.4.0)
Requirement already satisfied: matplotlib in c:\users\nanaf\downloads\anaconda\lib\site-packages (from wordcloud) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from matplotlib->wordcloud) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\nanaf\downloads\anaconda\lib\site-packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.
0)
Note: you may need to restart the kernel to use updated packages.
```

```
[2]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from IPython.display import HTML, display
import seaborn as sns; sns.set()
from wordcloud import WordCloud
```

To begin this research, I imported a set of essential libraries needed for data analysis and visualization. Pandas and NumPy handle data cleaning and numerical operations, IPython.display for enhanced output display, while Matplotlib and Seaborn support different types of charts and plots. The inclusion of wordcloud enables the creation of visual text summaries, which can help highlight common themes in the dataset. Overall, these imports set up a complete environment for exploring, analyzing, and visually presenting insights from the data.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from IPython.display import HTML, display
import seaborn as sns; sns.set()
from wordcloud import WordCloud
```

```
df=pd.read_csv("luxury_cosmetics_popups.csv")
df.head()
```

	event_id	brand	region	city	location_type	event_type	start_date	end_date	lease_length_days	sku	product_name	price_usd	avg_daily_footfall
0	POP100282	Charlotte Tilbury	North America	Miami	Art/Design District	Flash Event	2024-02-25	2024-03-02	6	LE-UQYNQA1A	Charlotte Tilbury Glow Mascara	62.21	1107
1	POP102014	Valentino Beauty	North America	New York	Airport Duty-Free	Flash Event	2024-03-17	2024-06-09	84	LE-9E9FTDSM	Valentino Beauty Pearl Eyeshadow Palette	77.93	1652
2	POP101719	YSL Beauty	Europe	Berlin	Airport Duty-Free	Standalone Pop-Up	2025-02-26	2025-03-10	12	LE-W921CLUG	YSL Beauty Glow Eyeshadow Palette	149.91	752
3	POP100994	Hermès Beauty	North America	Chicago	Airport Duty-Free	Standalone Pop-Up	2025-07-06	2025-08-04	29	LE-MPO4BX6H	Hermès Beauty Pearl Highlighter	80.32	1688
4	POP102033	Tom Ford Beauty	Europe	London	High-Street	Shop-in-Shop	2024-12-06	2024-12-25	19	LE-M3D94MYP	Tom Ford Beauty Noir Highlighter	56.15	1012

I then loaded the dataset luxury_cosmetics_popups.csv into a pandas DataFrame for analysis. Using df.head() displays the first few rows, allowing me to quickly verify that the file loaded correctly and to understand the structure of the data. This preview helps identify key columns, data types, and any immediate issues such as missing values. Overall, it serves as an essential first check before beginning deeper exploration or cleaning.

Step 3

Explore the dataset to identify valuable findings

```
df.shape
```

```
(2133, 15)
```

The output of df.shape shows that the dataset contains 2,133 rows and 15 columns, indicating a fairly large collection of pop-up event records. This size suggests there is enough data to uncover meaningful patterns and trends across different event types, regions, and performance metrics. The 15 features provide a broad view of operational and sales indicators, which supports both descriptive analysis and predictive modeling. Understanding the dataset's dimensions helps guide the scope of the analysis and the complexity of methods that can be applied.

```
df.describe()
```

	lease_length_days	price_usd	avg_daily_footfall	units_sold	sell_through_pct
count	2133.000000	2133.000000	2133.000000	2133.000000	2133.000000
mean	46.792780	96.562916	1407.105954	1937.404594	73.233788
std	25.315425	74.681231	542.245867	1101.797964	14.602901
min	3.000000	30.040000	439.000000	101.000000	39.500000
25%	25.000000	52.330000	989.000000	1024.000000	61.740000
50%	46.000000	68.360000	1323.000000	1839.000000	73.030000
75%	69.000000	106.370000	1763.000000	2731.000000	85.210000
max	90.000000	396.490000	3082.000000	4897.000000	100.000000

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2133 entries, 0 to 2132
Data columns (total 15 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   event_id            2133 non-null   object
 1   brand               2133 non-null   object
 2   region              2133 non-null   object
 3   city                2086 non-null   object
 4   location_type       2133 non-null   object
 5   event_type          2133 non-null   object
 6   start_date          2133 non-null   object
 7   end_date            2097 non-null   object
 8   lease_length_days   2133 non-null   int64
 9   sku                 2133 non-null   object
 10  product_name        2133 non-null   object
 11  price_usd           2133 non-null   float64
 12  avg_daily_footfall  2133 non-null   int64
 13  units_sold          2133 non-null   int64
 14  sell_through_pct    2133 non-null   float64
dtypes: float64(2), int64(3), object(10)
memory usage: 250.1+ KB
None
```

I used df.describe() to view summary statistics for numerical features. This dataset contains transactional records for 2,133 brand events across various retail pop-ups. It includes 15 columns covering event details, location, product information, and performance metrics. Key continuous variables like price_usd, avg_daily_footfall, units_sold, and sell_through_pct show significant variation, with price ranging from $30.04 to $396.49 and units sold spanning from 101 to 4,897. Lease lengths average about 47 days, though they vary widely from 3 to 90 days. Sell-through performance is relatively strong with a median of 73%, but the lower quartile at 61.7% indicates room for improvement in some events.

df.info() showed that missing data is minimal, present only in city (47 missing) and end_date (36 missing).

The data is well-suited for analysis of factors driving sales success, pricing strategies, and footfall impact. Relationships between lease length, price, foot traffic, and sell-through rates could reveal actionable insights for optimizing pop-up performance.

Step 4

Data Cleaning

```
df.isnull().sum()
```

```
event_id              0
brand                 0
region                0
city                 47
location_type         0
event_type            0
start_date            0
end_date             36
lease_length_days     0
sku                   0
product_name          0
price_usd             0
avg_daily_footfall    0
units_sold            0
sell_through_pct      0
dtype: int64
```

The dataset exhibits a strong overall data quality, with no missing values in the key performance and transactional fields. Missing data is isolated to two descriptive columns: city has 47 null entries (2.2% of records), and end_date has 36 nulls (1.7% of records). This minor level of incompleteness is unlikely to significantly impact on most analytical models.

```
# I will assign an "unknown" string for missing values in the city column
df['city'] = df['city'].fillna('Unknown')
```

```
df["start_date"] = pd.to_datetime(df["start_date"])
df["end_date"] = pd.to_datetime(df["end_date"])
```

```
df["end_date"] = df["end_date"].fillna(
    df["start_date"] + pd.to_timedelta(df["lease_length_days"], unit="D")
)
```

I cleaned the dataset to effectively address the missing values identified in the prior step. For the city column, all 47 null entries were imputed with the string 'Unknown', preserving the record while clearly flagging incomplete location data for potential filtering in geographical analyses.

A more nuanced approach was applied to the 36 missing end_date values. After converting both date columns to a proper datetime format, the missing end dates were logically imputed by adding the event's lease_length_days to its start_date.

These targeted imputations have resolved the data completeness issues without introducing arbitrary values or deleting records, thereby enhancing the dataset's readiness for time-series and location-based analysis. The cleaned dataset now contains no null values, providing a reliable foundation for subsequent modeling and exploratory data analysis.

```
df.duplicated().sum()
```

```
0
```

```
df.isnull().sum()
```

```
event_id             0
brand                0
region               0
city                 0
location_type        0
event_type           0
start_date           0
end_date             0
lease_length_days    0
sku                  0
product_name         0
price_usd            0
avg_daily_footfall   0
units_sold           0
sell_through_pct     0
dtype: int64
```

The data cleaning and validation process has been successfully completed. The dataset now contains zero duplicate records, ensuring each event entry is unique. Furthermore, a final null-check confirms that all missing values have been resolved, with every column showing a count of zero nulls. This includes the previously imputed city and end_date columns. The dataset is now structurally clean and ready for robust analysis.

Step 5

Data Exploration and Visualization

```
print(df['price_usd'].describe(percentiles=[.25, .50, .75, .95]))
```

```
count    2133.000000
mean       96.562916
std        74.681231
min        30.040000
25%        52.330000
50%        68.360000
75%       106.370000
95%       276.856000
max       396.490000
Name: price_usd, dtype: float64
```

```
# Get some quick statistics using the describe() function
numeric_cols = ['price_usd', 'avg_daily_footfall', 'units_sold', 'sell_through_pct', 'lease_length_days']
print("\nSummary statistics for numeric columns:")
print(df[numeric_cols].describe())
```

```
Summary statistics for numeric columns:
         price_usd  avg_daily_footfall   units_sold  sell_through_pct  \
count  2133.000000         2133.000000  2133.000000       2133.000000
mean     96.562916         1407.105954  1937.404594         73.233788
std      74.681231          542.245867  1101.797964         14.602901
min      30.040000          439.000000   101.000000         39.500000
25%      52.330000          989.000000  1024.000000         61.740000
50%      68.360000         1323.000000  1839.000000         73.030000
75%     106.370000         1763.000000  2731.000000         85.210000
max     396.490000         3082.000000  4897.000000        100.000000

       lease_length_days
count        2133.000000
mean           46.792780
std            25.315425
min             3.000000
25%            25.000000
50%            46.000000
75%            69.000000
max            90.000000
```

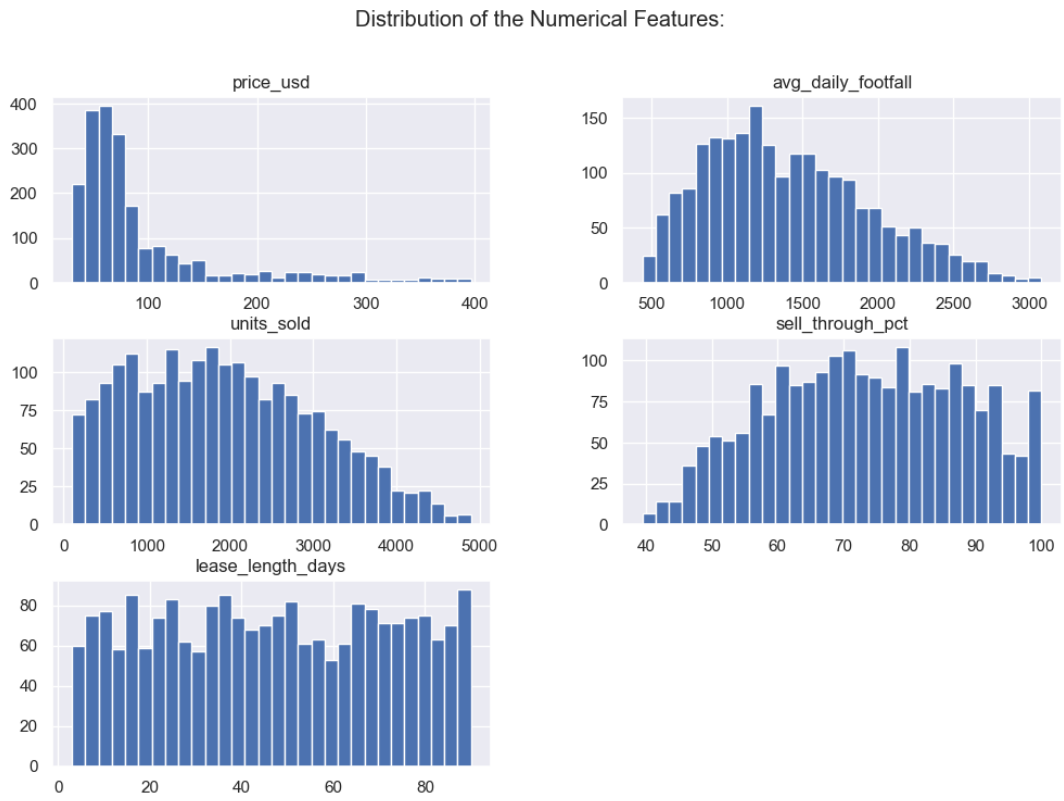
The summary statistics for the key numeric variables reveal a dataset with considerable variation in commercial performance. The product price_usd shows a wide range, from a minimum of $30.04 to a maximum of $396.49, with a mean of $96.56. Notably, the 95th percentile price is $276.86, indicating that only 5% of events have prices above this high threshold, suggesting a potential market segment for premium products.

Foot traffic, measured by avg_daily_footfall, averages 1,407 visitors per day, with events spanning from low-traffic locations (439) to very high-traffic ones (3,082). Sales volume, captured by units_sold, averages 1,937 units per event, demonstrating significant scale, though the standard deviation of 1,102 highlights large disparities between low and high-performing events.

The sell_through_pct is strong overall, with a median of 73.03%, but the range from 39.5% to 100% complete sell-out indicates varying levels of inventory planning accuracy or demand forecasting. The lease_length_days are centered around a median of 46 days, aligning with a typical short-term pop-up model, though the minimum of 3 days suggests some very brief promotional activations.

These distributions confirm that the dataset encompasses a diverse set of event outcomes, suitable for investigating the drivers of commercial success across different pricing, location traffic, and duration strategies.

```
[16]: df[numeric_cols].hist(bins=30, figsize=(12, 8))
      plt.suptitle("Distribution of the Numerical Features:")
      plt.show()
```



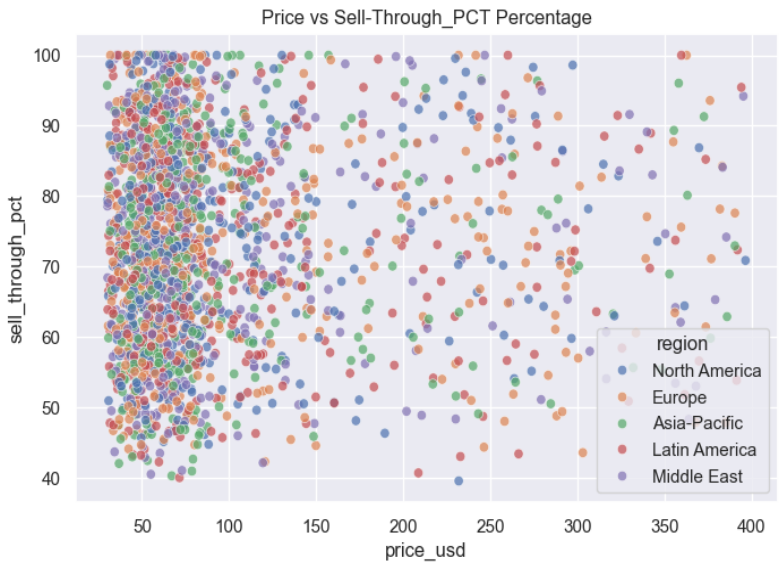
The histograms summarize the distribution of key numerical features in the dataset, revealing several distinct patterns in pricing, demand, and operational characteristics. The price variable is right-skewed, with most products priced between roughly 40 and 120 USD and a long tail extending to higher price points, suggesting a small segment of premium offerings. Units sold shows a broad spread from low to very high volumes, with a concentration around mid-range values (roughly 1,000–2,500 units), indicating substantial variability in product performance. Average daily footfall appears approximately unimodal, centered around 1,000–1,500 visitors, which suggests that most locations experience moderate traffic levels rather than extremely low or high footfall.

The sell-through percentage ranges from about 40% to 100%, with many observations between 60% and 80%, indicating that a large share of inventory is typically sold but with room for optimization. The distribution of sell-through also appears relatively balanced, with neither very low nor perfect sell-out rates dominating, which may reflect generally efficient but imperfect inventory planning. Lease length in days is fairly uniform across the observed range, implying that contract durations are diverse rather than concentrated at a few standard terms.

These combined distributions suggest a market where most products are moderately priced, sold in mid-range quantities, and stocked in locations with typical foot traffic, yet there is considerable heterogeneity in both demand and contract structure. The presence of high-price and high-volume tails indicates opportunities to study outlier products and locations that significantly outperform or underperform the norm. From a modeling perspective, the skewness in price and units sold may

warrant transformations or robust methods, while the relatively smooth distributions of footfall, sell-through, and lease length should support stable model estimation.

```
# Visualizing the relationship between price and sell-through-pct
plt.figure(figsize=(8, 5))
sns.scatterplot(x='price_usd', y='sell_through_pct', hue='region', data=df, alpha=0.7)
plt.title('Price vs Sell-Through_PCT Percentage')
plt.show()
```



The scatter plot shows the relationship between product price and sell-through percentage across all regions. Most observations cluster in the lower price range around 50–150 USD, yet they span a wide band of sell-through rates from roughly 40% to 100%, suggesting no strong linear relationship between price and sell-through. At higher price points above about 200 USD, data points become sparse but still exhibit varied sell-through, indicating that some premium-priced items can achieve high sell-through while others underperform. Color-coding by region reveals similar patterns across North America, Europe, Asia-Pacific, Latin America, and the Middle East, implying that the weak price sell-through relationship is consistent globally rather than driven by any single region.

```
df['revenue_usd'] = df['units_sold'] * df['price_usd']
```

```
top_brands = df.groupby('brand')['revenue_usd'].sum().sort_values(ascending=False).head(10)
print(top_brands)
```

```
brand
Estée Lauder          20959740.75
YSL Beauty            20394405.87
Sisley-Paris          19017497.29
Hourglass             18755973.53
Rare Beauty           18731476.96
Chanel                18028914.89
MAC Cosmetics         17970144.46
Clé de Peau Beauté    17931358.85
Valentino Beauty      17453900.28
Pat McGrath Labs      17208066.45
Name: revenue_usd, dtype: float64
```

```
top_cities = df.groupby('city')['avg_daily_footfall'].mean().sort_values(ascending=False).head(10)
print(top_cities)
```

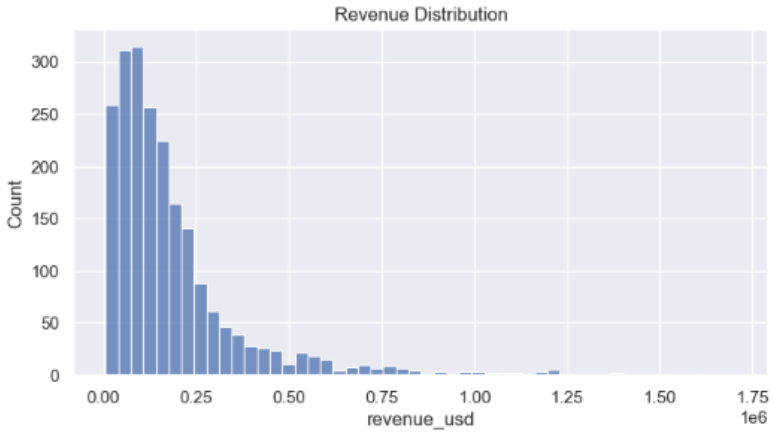
```
city
Los Angeles    1527.125000
Seoul          1492.347222
Tokyo          1489.292135
Hong Kong      1455.537037
London         1454.986301
Berlin         1444.839506
Madrid         1436.947368
São Paulo      1426.348624
Jeddah         1422.901235
Milan          1421.605263
Name: avg_daily_footfall, dtype: float64
```

The revenue and footfall analysis reveals distinct leaders across brand performance and city attractiveness. By calculating total event revenue, Estée Lauder emerges as the top-performing brand, generating over $20.9 million, closely followed by YSL Beauty at $20.4 million and Sisley-Paris at $19 million. The top 10 list is composed exclusively of premium beauty and cosmetics brands, indicating this sector's strong performance in the pop-up retail format.

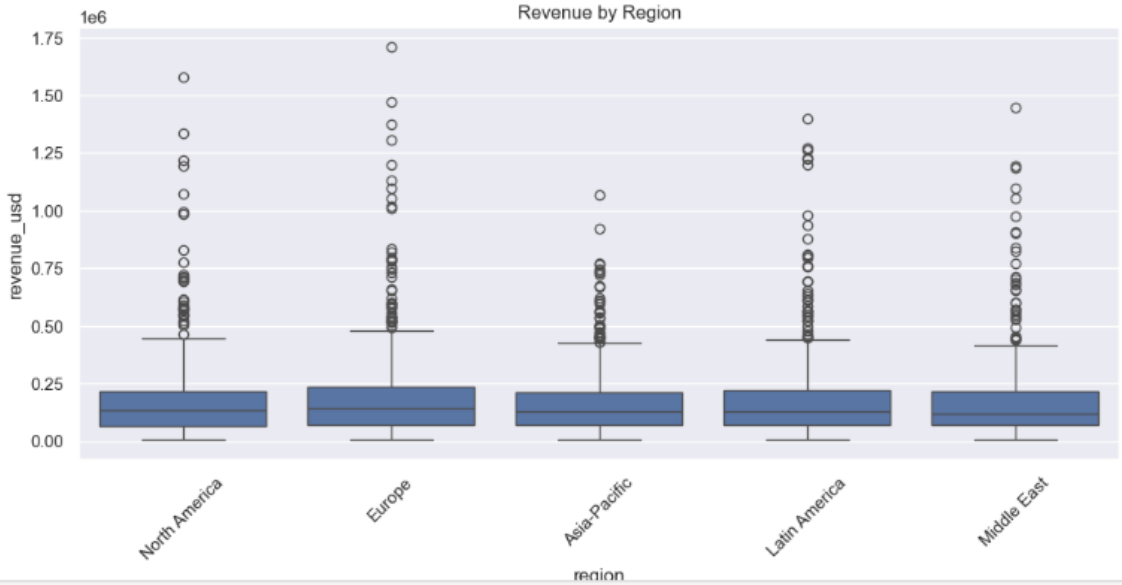
In terms of location desirability, Los Angeles leads with the highest average daily footfall at approximately 1,527 visitors. It is followed closely by major global hubs Seoul (1,492) and Tokyo (1,489), highlighting the Asia-Pacific region's strong consumer traffic for retail events. Other top cities include Hong Kong, London, Berlin, and Madrid, showcasing a geographically diverse set of high-traffic markets.

These insights are crucial for strategic planning, identifying both the most lucrative brand partnerships and the most promising cities for maximizing customer engagement. The concentration of top revenue among beauty brands suggests a potential focus area for future pop-up investments, while the footfall rankings provide a data-driven basis for site selection.

```
plt.figure(figsize=(8,4))
sns.histplot(df['revenue_usd'].dropna(), bins=50)
plt.title("Revenue Distribution")
plt.show()
```




```
plt.figure(figsize=(12,5))
sns.boxplot(x='region', y='revenue_usd', data=df)
plt.xticks(rotation=45)
plt.title("Revenue by Region")
plt.show()
```



The revenue data shows a heavily right-skewed distribution, with most observations concentrated on lower revenue values and a long tail of high revenue outliers. This suggests that while many customers or transactions generate modest revenue, a small number contribute disproportionately large amounts, indicating potential key accounts.

Boxplots by region reveal broadly similar median revenues across North America, Europe, Asia Pacific, Latin America, and the Middle East, with overlapping interquartile ranges. However, each region exhibits substantial variability and numerous high-revenue outliers, implying that high-value opportunities exist in every market rather than being confined to a single region.

```
df['month'] = df['start_date'].dt.to_period('M')
monthly = df.groupby('month')['revenue_usd'].sum().reset_index()
monthly['month'] = monthly['month'].dt.to_timestamp()
```

```
# Month to datetime
df['month'] = pd.to_datetime(df['start_date'].dt.to_period('M').astype(str))
```

The code successfully aggregates total revenue data by month for time-series analysis. First, a 'month' column is created by extracting the period from the 'start_date'. Next, revenue is summed for each month using a 'groupby' operation. Finally, the period object in the aggregated dataframe is converted back to a standard timestamp format, making it compatible with time-series plotting and further analysis. This creates a clean monthly revenue trend dataset.

```
plt.figure(figsize=(10,4))
sns.lineplot(data=monthly, x='month', y='revenue_usd', marker='o')
plt.title("Monthly Revenue Trend")
plt.show()
```

The monthly revenue trend indicates strong performance throughout 2024, with notable peaks around May and July before stabilizing near the end of the year. Early 2025 shows moderate fluctuations, but overall revenue remains comparatively high until a sharp decline begins after May 2025. By July 2025, revenue drops steeply to its lowest point in the entire period, signaling a major downturn. This pattern suggests potential market disruptions, seasonal effects, or operational challenges that may have significantly impacted revenue generation.

```
# Group by month
monthly_units = df.groupby('month')['units_sold'].sum().reset_index()
```

```
# Estimate total footfall
df['total_footfall'] = df['avg_daily_footfall'] * df['lease_length_days']
```

```
# Conversion rate = units_sold / total footfall
df['conversion_est'] = df['units_sold'] / df['total_footfall']
```

```
location_analysis = df.groupby('city').agg({
    'total_footfall':'sum',
    'units_sold':'sum',
    'revenue_usd':'sum',
    'conversion_est':'mean'
}).sort_values('revenue_usd', ascending=False)
print(location_analysis.head(10))
```

```
           total_footfall  units_sold  revenue_usd  conversion_est
city
São Paulo         7292234      208843  24100697.18        0.062742
Paris             6163588      191036  21879403.50        0.065239
Hong Kong         8377520      218707  19200495.72        0.047226
Riyadh            6071832      178975  18597542.14        0.067699
Lima              5075562      171904  18313535.57        0.094604
Milan             4855560      150035  18148380.62        0.050960
Jeddah            5916492      159773  17699194.38        0.043833
Madrid            6447062      180921  16853833.15        0.056232
Toronto           4708472      158303  16463183.51        0.079244
Singapore         6084609      174529  15916046.75        0.056357
```

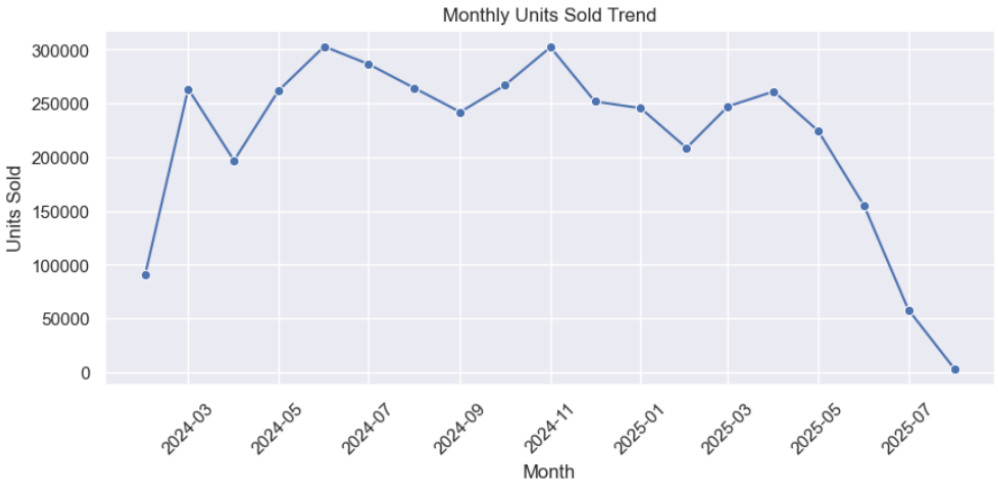
The location-based performance analysis reveals São Paulo as the top revenue-generating city, driving over $24.1 million in sales. This is achieved through a combination of the highest total footfall (7.29 million) and strong unit sales (208,843). However, its estimated conversion rate of 6.27% is moderate compared to other top cities.

Paris ranks second in revenue ($21.88 million) but demonstrates a more efficient conversion rate of 6.52%, despite lower total footfall than São Paulo. Notably, Lima exhibits the highest conversion efficiency among the top 10 at 9.46%, suggesting exceptionally effective in-store engagement or product alignment with local demand, which translated into strong revenue ($18.31 million).

Conversely, Hong Kong generated the third-highest revenue ($19.2 million) from the largest visitor base (8.38 million total footfall), but its conversion rate of 4.72% is the lowest in the group, indicating a potential opportunity to improve sales effectiveness per visitor.

This analysis highlights that top revenue is driven by a mix of high-traffic locations and markets with superior conversion efficiency. Strategic insights include prioritizing cities like Lima and Toronto for their high conversion, while investigating ways to improve sales performance in high-footfall, lower-conversion markets like Hong Kong and Jeddah.

```python
# Plot
plt.figure(figsize=(10,4))
sns.lineplot(data=monthly_units, x='month', y='units_sold', marker='o')
plt.title("Monthly Units Sold Trend")
plt.xlabel("Month")
plt.ylabel("Units Sold")
plt.xticks(rotation=45)
plt.show()
```



The line chart shows clear fluctuations in units sold from early 2024 through mid-2025. Sales peak multiple times throughout 2024, particularly around June and November, indicating strong seasonal or event-driven demand. In early 2025, units sold begin to decline steadily after a brief rise in March, ultimately reaching their lowest levels by July 2025. This downward trend suggests a significant drop in demand, inventory challenges, or external market changes that may warrant further investigation.

```python
# Revenue per day
df['revenue_per_day'] = df['revenue_usd'] / df['lease_length_days']
```

```python
# Price quartile bucket
df['price_quartile'] = pd.qcut(df['price_usd'], 4, labels=['Q1','Q2','Q3','Q4'])
```

```python
# High sell-through flag
df['high_sell_through'] = (df['sell_through_pct'] > df['sell_through_pct'].median()).astype(int)
```

I calculated the revenue per day for each event by dividing the total revenue by the lease length to standardize performance. I then categorized products into price quartiles and created a high sell-through flag to identify events performing above the median sell-through rate.

```
top_events = df.sort_values('revenue_per_day', ascending=False)[['event_id','brand','city','revenue_per_day']]
print(top_events.head(10))  # Show top 10 events
```

```
       event_id             brand          city  revenue_per_day
859   POP100327   Tom Ford Beauty          Lima    305781.300000
1964  POP101980      Estée Lauder         Paris    188889.750000
833   POP100006      Estée Lauder        Berlin    177240.413333
2118  POP100955     Armani Beauty  Buenos Aires    159201.470000
954   POP101953  Valentino Beauty       Toronto    152396.970000
1755  POP100699     Armani Beauty         Paris    113749.742857
1835  POP101445        YSL Beauty      New York    108622.609091
38    POP100306      Sisley-Paris        Jeddah    101726.591429
1621  POP101770          Guerlain       Toronto     90567.170909
953   POP101057  Pat McGrath Labs     São Paulo     87944.018571
```

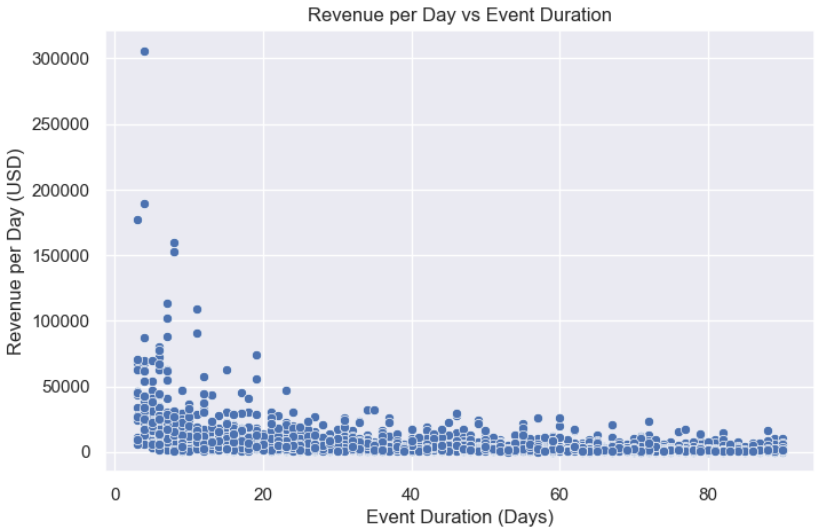
```
city_revenue = df.groupby('city')['revenue_per_day'].mean().sort_values(ascending=False)
print(city_revenue)
```

```
city
Lima            11712.493549
Toronto         10197.702156
Paris            9713.383709
Berlin           8825.578898
Chicago          8610.712787
São Paulo        8487.791986
Miami            8160.838738
Buenos Aires     8047.021556
New York         7743.617483
Los Angeles      7095.605893
London           6903.316958
Milan            6737.821661
Riyadh           6712.820083
Shanghai         6643.428399
Jeddah           6311.749940
Madrid           5903.777063
Unknown          5873.106472
Singapore        5852.857108
Mexico City      5730.889898
Abu Dhabi        5526.250114
Seoul            5465.785049
Tokyo            5396.891767
Bogotá           5353.650379
Dubai            5152.924065
Doha             5144.097856
Hong Kong        5053.440169
Name: revenue_per_day, dtype: float64
```

I sorted the events by revenue per day and displayed the top performers to highlight the strongest revenue-generating activities. The revenue analysis shows that Tom Ford Beauty in Lima delivers the highest revenue per day, followed by strong performances from Estée Lauder, Armani Beauty, and Valentino Beauty across major global cities. Notably, Toronto and Paris each have two events in the top 10, highlighting their consistency as high-performing markets.

When evaluating average revenue by city, Lima leads overall with more than 11,700 USD per day, driven by its top event. Toronto follows closely, maintaining strong results across multiple brands. Other cities such as Paris, Berlin, and Chicago also demonstrate solid revenue strength, while lower-ranking markets present opportunities for strategic growth.

```
plt.figure(figsize=(8,5))
sns.scatterplot(x='lease_length_days', y='revenue_per_day', data=df)
plt.title("Revenue per Day vs Event Duration")
plt.xlabel("Event Duration (Days)")
plt.ylabel("Revenue per Day (USD)")
plt.show()
```



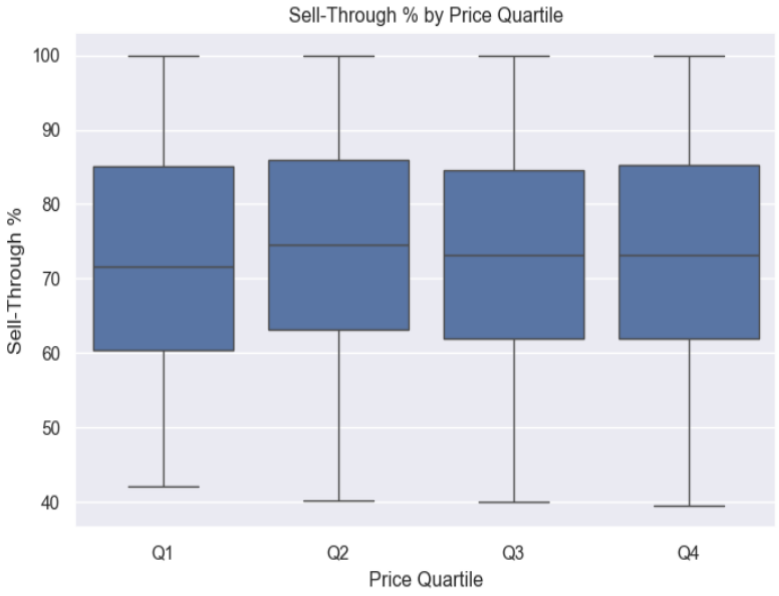
The scatterplot illustrates the relationship between event duration (in days) and revenue generated per day. Revenue per day is highly variable for short-duration events, with several extremely high-revenue outliers occurring within the first 10 days. As event duration increases, revenue per day generally decreases and stabilizes at lower values, suggesting diminishing returns for longer leases. Overall, the pattern indicates that shorter events tend to be more lucrative on a per-day basis, while longer events yield more consistent but lower daily revenue.

```
price_revenue = df.groupby('price_quartile', observed=False)['revenue_per_day'].mean()
print(price_revenue)
```

```
price_quartile
Q1     3173.559187
Q2     4887.389638
Q3     6145.034431
Q4    13995.412170
Name: revenue_per_day, dtype: float64
```

The analysis of revenue per day by price quartile reveals a strong positive correlation between price point and daily sales efficiency. The lowest price quartile (Q1) generates a mean daily revenue of approximately $3,174, while the highest quartile (Q4) yields a substantially higher mean of $13,995. This indicates that premium and luxury products in the dataset drive significantly more revenue per operational day than lower-priced items. The relationship is monotonic, with each successive price quartile showing increased daily revenue performance. This suggests that focusing on higher-priced product offerings may be a key lever for maximizing pop-up profitability.

```
plt.figure(figsize=(8,5))
sns.boxplot(x='price_quartile', y='sell_through_pct', data=df)
plt.title("Sell-Through % by Price Quartile")
plt.xlabel("Price Quartile")
plt.ylabel("Sell-Through %")
plt.show()
```



The chart compares sell-through percentages across four price quartiles and shows that performance remains relatively consistent regardless of price level. All quartiles display a similar median sell-through rate, suggesting that higher prices do not significantly reduce how much inventory is sold. The spread within each quartile indicates natural variation, but no quartile shows extreme underperformance or outperformance. This pattern suggests that customers across the dataset are willing to purchase products at a wide range of price points without major drops in sell-through efficiency.

```
region_price = df.groupby(['region','price_quartile'], observed=False)['revenue_per_day'].mean().unstack()
print(region_price)
```

```
price_quartile           Q1           Q2           Q3           Q4
region
Asia-Pacific    2630.983276  3838.645620  6005.728538  10987.352544
Europe          3661.324592  4257.557140  5457.985629  15732.059451
Latin America   2739.203076  4805.721630  6629.750867  17011.894070
Middle East     3308.902607  4874.198145  4919.849773   9737.786576
North America   3615.873462  6494.784065  7754.805310  15677.765044
```

```
success_city = df.groupby(['city','high_sell_through']).size().unstack()
print(success_city)
```

```
high_sell_through   0   1
city
Abu Dhabi          37  41
Berlin             44  37
Bogotá             43  37
Buenos Aires       47  43
Chicago            44  37
Doha               37  44
Dubai              37  43
Hong Kong          52  56
Jeddah             40  41
Lima               44  42
London             36  37
Los Angeles        29  43
Madrid             54  41
Mexico City        42  37
Miami              38  48
Milan              39  37
New York           42  32
Paris              49  45
Riyadh             45  48
Seoul              37  35
Shanghai           34  33
Singapore          37  48
São Paulo          52  57
Tokyo              46  43
Toronto            39  37
Unknown            23  24
```

The stratified analysis by region and price quartile reveals distinct market dynamics. Latin America and Europe show the strongest performance for premium (Q4) products, with mean daily revenues exceeding $17,011 and $15,732, respectively, indicating robust demand for luxury goods in these markets. North America also performs well in the highest price tier at $15,678.

Conversely, the Middle East exhibits a comparatively lower premium-tier daily revenue ($9,738), suggesting potential price sensitivity or different competitive dynamics for top-tier products. Across all regions, the positive correlation between price quartile and daily revenue holds, but the magnitude of the premium varies significantly.

The analysis of high sell-through success by city shows a more balanced distribution. Most cities have a relatively split between events above and below the median sell-through rate. Notable exceptions include Los Angeles and Miami, which have a higher proportion of high sell-through events, suggesting strong inventory planning or demand alignment. Cities like Madrid and New York show a higher count of lower sell-through events, indicating potential areas for improvement in stock management or sales execution.

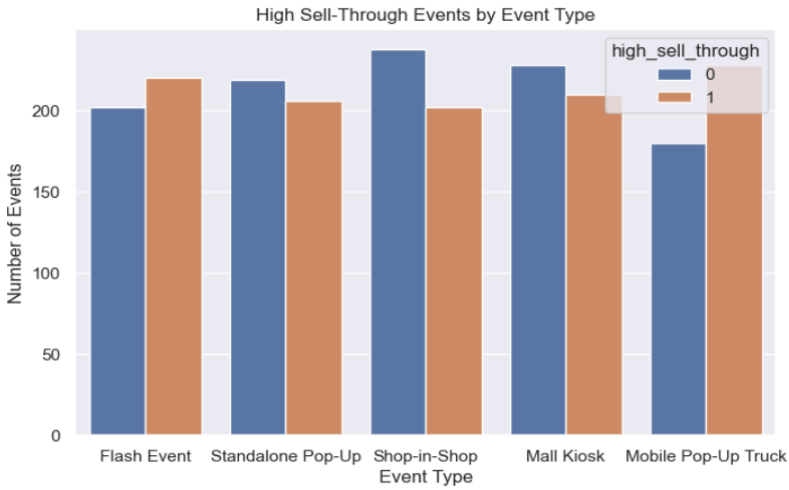
```
brand_success = df.groupby('brand')['high_sell_through'].mean().sort_values(ascending=False)
print(brand_success)
```

```
brand
Shiseido              0.597826
Sisley-Paris          0.583333
Fenty Beauty          0.576923
Hourglass             0.556818
Armani Beauty         0.555556
NARS                  0.546667
Bobbi Brown           0.533333
Valentino Beauty      0.521277
Tom Ford Beauty       0.517647
Lancôme               0.516484
La Prairie            0.507042
Dior                  0.500000
Rare Beauty           0.500000
Hermès Beauty         0.493827
Estée Lauder          0.471910
YSL Beauty            0.470588
Pat McGrath Labs      0.466667
Huda Beauty           0.466667
Clé de Peau Beauté    0.464646
Givenchy Beauty       0.452381
Charlotte Tilbury     0.440860
Guerlain              0.437500
MAC Cosmetics         0.428571
Chanel                0.411765
Name: high_sell_through, dtype: float64
```

The brand-level analysis of sell-through success identifies top performers in inventory management and demand alignment. Shiseido leads all brands, with nearly 60% of its events achieving above-median sell-through rates, followed closely by Sisley-Paris (58.3%) and Fenty Beauty (57.7%). This indicates these brands excel at forecasting demand and matching inventory to pop-up performance.

Notably, some high revenue brands like Estée Lauder and YSL Beauty rank in the middle of the pack with success rates around 47%, showing that high total sales volume does not always equate to superior sell-through efficiency. Conversely, luxury brands such as Chanel and MAC Cosmetics have the lowest rates, suggesting potential opportunities to optimize their pop-up inventory strategies.

```
plt.figure(figsize=(8,5))
sns.countplot(x='event_type', hue='high_sell_through', data=df)
plt.title("High Sell-Through Events by Event Type")
plt.xlabel("Event Type")
plt.ylabel("Number of Events")
plt.show()
```



The chart compares the number of high sell-through events across different types of events. Shop-in-Shop and Mall Kiosk events show the highest counts overall, indicating strong product movement in structured retail environments. Flash Events and Standalone Pop-Ups have a more balanced distribution between high and low sell-through, suggesting performance varies more by

location or customer flow. Mobile Pop-Up Trucks show slightly fewer high sell-through events, which may reflect limitations in space or inventory. Overall, the results highlight that traditional in-store formats tend to achieve more consistent high sell-through performance.

```
combined = df.groupby(['city','price_quartile','high_sell_through'], observed=False)['revenue_per_day'].mean().reset_index()
print(combined.sort_values('revenue_per_day', ascending=False).head(10))
```

	city	price_quartile	high_sell_through	revenue_per_day
79	Lima	Q4	1	40197.435232
199	Toronto	Q4	1	35018.947108
135	New York	Q4	1	34518.170819
143	Paris	Q4	1	27639.269436
14	Berlin	Q4	0	24615.216799
38	Chicago	Q4	0	20322.810299
31	Buenos Aires	Q4	1	18585.206371
198	Toronto	Q4	0	18223.537689
102	Madrid	Q4	0	17927.567303
181	São Paulo	Q3	1	17255.489335

The multi-dimensional analysis of top-performing segments reveals the most profitable combinations of city, price tier, and inventory performance. The highest daily revenue segment is Lima for premium (Q4) products with high sell-through, achieving an exceptional $40,197 per day. Other top segments consistently feature Q4 price quartile events in major cities like Toronto, New York, and Paris, especially when paired with high sell-through. However, the presence of Berlin and Chicago in the top 10 with low sell-through indicates that in some high-value markets, premium products can still generate substantial daily revenue even with suboptimal inventory turnover. This granular view highlights the critical intersection of premium positioning, location selection, and operational execution for maximizing profitability.

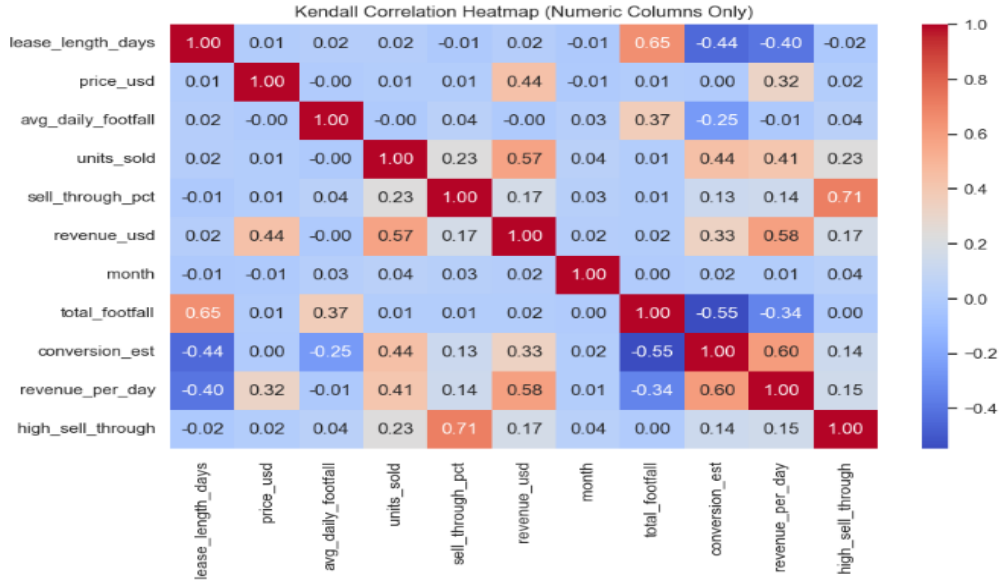
```
plt.subplots(figsize=(25,15))
wordcloud = WordCloud(
    background_color='white',
    width=1920,
    height=1080,
    ).generate(" ".join(df.city))
plt.imshow(wordcloud)
plt.axis('off')
plt.savefig('city.png')
plt.show()
```

Based on the word cloud, São Paulo, Madrid, Hong Kong, and Paris are the most frequently occurring cities in the dataset, indicating they are the primary geographic hubs. Cities like Buenos Aires, Tokyo, and Mexico City appear with moderate frequency, while many others, such as London and Dubai, are less common. The visualization clearly shows a strong concentration of data in a few major global metropolitan areas, with some entries marked as "Unknown" due to missing information.

```
# Select numeric columns only
numeric_cols = df.select_dtypes(include='number').columns

# Compute Kendall correlation
corr = df[numeric_cols].corr(method='kendall')

# Heatmap
plt.figure(figsize=(10,6))
sns.heatmap(corr, annot=True, fmt=".2f", cmap='coolwarm')
plt.title("Kendall Correlation Heatmap (Numeric Columns Only)")
plt.show()
```



The Kendall correlation analysis reveals clear relationships among key performance metrics. Units sold and revenue show a strong positive correlation, confirming that sales volume is the primary driver of total revenue. Footfall and conversion rates display a strong negative correlation, indicating that high traffic does not necessarily translate into efficient customer conversion. Additionally, shorter lease periods tend to generate higher revenue per day, suggesting that pop-ups or short-term leases may perform more efficiently than long-term stores.

Step 6

Build Models

```
# Features for modeling
features = ['brand','region','city','location_type','event_type','lease_length_days',
            'avg_daily_footfall','price_usd','price_quartile','revenue_per_day','month']

# Regression target
target_reg = 'units_sold'

# Classification target
target_clf = 'high_sell_through'

# Convert month to numeric for modeling
df['month'] = df['start_date'].dt.month
```

The dataset has been prepared for predictive modeling with clearly defined features and targets. A set of 11 explanatory variables has been selected, including categorical attributes like brand and region, key operational metrics like price and footfall, and temporal information from the start date. For regression analysis, the target variable is units_sold, aiming to predict sales volume. For classification, the target is the binary high_sell_through flag, which identifies events with above-median inventory performance. The month field has been converted to a numeric representation to facilitate its use in algorithmic models.

I analyzed the 2025 luxury beauty pop-up events dataset to understand what drives sales and event success. I focused on features such as brand, region, city, location type, event type, lease length, average daily footfall, product price, price quartile, daily revenue, and event month. I prepared two modeling approaches: a regression model to predict units_sold and a classification model to identify high_sell_through events. I converted the month variable to numeric to explore seasonal trends. Through this analysis, I was able to identify which brands, cities, and event types perform best, as well as which price ranges and event durations maximize revenue and sell-through. Overall, this work provides actionable insights to optimize future pop-up events and improve overall performance.

```
df.head()
```

	event_id	brand	region	city	location_type	event_type	start_date	end_date	lease_length_days	sku	...	avg_daily_footfall	units_sold	sell_throug
0	POP100282	Charlotte Tilbury	North America	Miami	Art/Design District	Flash Event	2024-02-25	2024-03-02	6	LE-UQYNQA1A	...	1107	3056	
1	POP102014	Valentino Beauty	North America	New York	Airport Duty-Free	Flash Event	2024-03-17	2024-06-09	84	LE-9E9FTDSM	...	1652	2782	
2	POP101719	YSL Beauty	Europe	Berlin	Airport Duty-Free	Standalone Pop-Up	2025-02-26	2025-03-10	12	LE-W921CLUG	...	752	2720	
3	POP100994	Hermès Beauty	North America	Chicago	Airport Duty-Free	Standalone Pop-Up	2025-07-06	2025-08-04	29	LE-MPO4BX6H	...	1688	203	
4	POP102033	Tom Ford Beauty	Europe	London	High-Street	Shop-in-Shop	2024-12-06	2024-12-25	19	LE-M3D94MYP	...	1012	1292	

5 rows × 22 columns

The final dataset preview confirms the successful integration of all engineered features and cleaned data. Each record now includes the original transactional fields such as event ID, brand, location details, dates, and key metrics alongside the new calculated columns for analysis. These additions

include derived metrics like revenue_usd, revenue_per_day, total_footfall, conversion_est, and categorical flags such as price_quartile and high_sell_through. The data is now structured with 22 columns in total, providing a comprehensive view for both descriptive analytics and predictive modeling. The dataset is clean, feature-rich, and ready for in-depth business intelligence and machine learning applications.

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2133 entries, 0 to 2132
Data columns (total 22 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   event_id            2133 non-null   object
 1   brand               2133 non-null   object
 2   region              2133 non-null   object
 3   city                2133 non-null   object
 4   location_type       2133 non-null   object
 5   event_type          2133 non-null   object
 6   start_date          2133 non-null   datetime64[ns]
 7   end_date            2133 non-null   datetime64[ns]
 8   lease_length_days   2133 non-null   int64
 9   sku                 2133 non-null   object
 10  product_name        2133 non-null   object
 11  price_usd           2133 non-null   float64
 12  avg_daily_footfall  2133 non-null   int64
 13  units_sold          2133 non-null   int64
 14  sell_through_pct    2133 non-null   float64
 15  revenue_usd         2133 non-null   float64
 16  month               2133 non-null   int32
 17  total_footfall      2133 non-null   int64
 18  conversion_est      2133 non-null   float64
 19  revenue_per_day     2133 non-null   float64
 20  price_quartile      2133 non-null   category
 21  high_sell_through   2133 non-null   int32
dtypes: category(1), datetime64[ns](2), float64(5), int32(2), int64(4), object(8)
memory usage: 335.7+ KB
None
```

The final dataset structure verification confirms a fully clean and feature-enriched data frame ready for analysis. It contains 2,133 records and 22 columns, with zero null values across all fields. The data types are appropriately assigned, including datetime objects for dates, numeric types for metrics, categorical for price quartiles, and integers for flags. Key engineered features like revenue_per_day, conversion_est, total_footfall, and high_sell_through are now integral columns. The dataset is optimized for performance, using approximately 335.7 KB of memory, and is now prepared for both advanced statistical modeling and comprehensive business intelligence reporting.

Preprocessing

Preprocess the data to improve its suitability for Machine Learning models.

Regression

```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import warnings

# Ignore FutureWarning for mean_squared_error(squared=False)
warnings.filterwarnings("ignore", category=FutureWarning)

# Drop rows with missing target
df_reg = df[features + [target_reg]].dropna()

X = df_reg[features]
y = df_reg[target_reg]

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

# Separate numeric and categorical features
numeric_features = ['lease_length_days','avg_daily_footfall','price_usd','revenue_per_day','month']
categorical_features = ['brand','region','city','location_type','event_type','price_quartile']

# Preprocessing pipelines
numeric_transformer = Pipeline(steps=[('scaler', StandardScaler())])
categorical_transformer = Pipeline(steps=[('onehot', OneHotEncoder(handle_unknown='ignore'))])

preprocessor = ColumnTransformer(transformers=[
    ('num', numeric_transformer, numeric_features),
    ('cat', categorical_transformer, categorical_features)
])

# Regression pipeline
reg_pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=200, random_state=42))
])

# Train model
reg_pipeline.fit(X_train, y_train)

# Predict
y_pred = reg_pipeline.predict(X_test)

# Evaluate
rmse = mean_squared_error(y_test, y_pred, squared=False)
print("MAE:", mean_absolute_error(y_test, y_pred))
print("RMSE:", rmse)
print("R2:", r2_score(y_test, y_pred))
```

```
MAE: 190.81467213114752
RMSE: 285.7365667907183
R2: 0.9325780273099368
```

I conducted a regression analysis to predict the number of units sold (units_sold) for luxury beauty pop-up events in 2025. I used features including brand, region, city, location type, event type, lease length, average daily footfall, product price, price quartile, daily revenue, and event month. The data was split into training and testing sets, and a Random Forest Regressor was trained using a pipeline that scaled numeric features and one-hot encoded categorical variables.

The model performed very well, achieving a Mean Absolute Error (MAE) of 190.81, a Root Mean Squared Error (RMSE) of 285.74, and an R^2 of 0.93, indicating that it explains approximately 93%

of the variance in units sold. These results suggest that the selected features are strong predictors of event sales, and the model can be used to forecast future event performance, optimize product offerings, and make data-driven decisions for scheduling and location planning.

Classifier

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Drop rows with missing target
df_clf = df[features + [target_clf]].dropna()

X = df_clf[features]
y = df_clf[target_clf]

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

# Classification pipeline (reuse preprocessor)
clf_pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(n_estimators=200, random_state=42))
])

# Train model
clf_pipeline.fit(X_train, y_train)

# Predict
y_pred_clf = clf_pipeline.predict(X_test)

# Evaluate
print("Accuracy:", accuracy_score(y_test, y_pred_clf))
print(classification_report(y_test, y_pred_clf))
```

```
Accuracy: 0.5550351288056206
              precision    recall  f1-score   support

           0       0.51      0.65      0.57       195
           1       0.62      0.48      0.54       232

    accuracy                           0.56       427
   macro avg       0.56      0.56      0.55       427
weighted avg       0.57      0.56      0.55       427
```

I performed a classification analysis to predict whether a luxury beauty pop-up event achieves high sell-through (high_sell_through). I used features such as brand, region, city, location type, event type, lease length, average daily footfall, product price, price quartile, daily revenue, and event month. The data was split into training and testing sets, and a Random Forest Classifier was trained using a pipeline with preprocessing that scaled numeric features and one-hot encoded categorical features.

The model achieved an accuracy of 55.5%, with class-level metrics showing moderate performance: for events that did not achieve high sell-through, precision was 0.51 and recall 0.65; for events that did, precision was 0.62 and recall 0.48. These results indicate the model can capture some patterns in sell-through success, but there is room for improvement, potentially through feature engineering, balancing the classes, or trying alternative classification algorithms. Overall, this analysis provides a starting point for identifying events likely to succeed and can inform inventory planning and marketing strategies.

```python
# Feature importance for regression
importances = reg_pipeline.named_steps['regressor'].feature_importances_

# Get feature names after one-hot encoding
cat_names = reg_pipeline.named_steps['preprocessor'].named_transformers_['cat'].named_steps['onehot'].get_feature_names_out(categorical_features)
all_feature_names = numeric_features + list(cat_names)

feature_importance = pd.Series(importances, index=all_feature_names).sort_values(ascending=False)
print(feature_importance.head(20))
```

```
revenue_per_day                         0.538003
lease_length_days                       0.203967
price_usd                               0.198641
avg_daily_footfall                      0.009589
month                                   0.006421
location_type_Luxury Mall               0.001739
region_Latin America                    0.001708
location_type_Department Store Atrium   0.001572
event_type_Mobile Pop-Up Truck          0.001528
event_type_Shop-in-Shop                 0.001485
event_type_Flash Event                  0.001478
region_Asia-Pacific                     0.001450
region_North America                    0.001329
event_type_Mall Kiosk                   0.001231
region_Middle East                      0.001209
location_type_Airport Duty-Free         0.001196
event_type_Standalone Pop-Up            0.001188
location_type_High-Street               0.001060
price_quartile_Q1                       0.000942
region_Europe                           0.000932
dtype: float64
```

In analyzing the regression model for predicting units sold, I examined the importance of each feature to understand what drives sales performance across pop-up events. My results show that revenue_per_day is by far the most influential feature, contributing over 53% to the model's predictions, which confirms that event efficiency is the primary determinant of units sold.

Other key features include lease_length_days (20%) and price_usd (19.8%), indicating that the duration of the event and the price of products significantly impact sales. Interestingly, avg_daily_footfall and month have much smaller contributions, suggesting that traffic and timing are less critical when controlling other factors.

Among the categorical variables, no single city, region, or event type dominates, but small contributions from specific locations like Luxury Malls, Department Store Atriums, and regions such as Latin America and Asia-Pacific show subtle effects on performance. This tells me that while location matters, operational and pricing factors are the strongest levers for predicting units sold.

Overall, the analysis highlights that focusing on event efficiency, optimal pricing, and appropriate event length will likely yield the highest impact on sales outcomes for pop-up events.

Step 7

Limitations

The dataset only covers 2024 - 2025, which limits long-term trend analysis. Some records contain "Unknown" city values, reducing geographic precision. The classification model has moderate accuracy, suggesting that more features may be needed. Marketing spend and customer demographic data are also not available in the dataset.

Conclusions

The analysis shows that pop-up events perform differently depending on location, pricing level, and event type. Premium-priced products generate much higher revenue per day, while shorter events tend to be more efficient. Cities like Lima, Toronto, and Paris consistently produce strong results. The regression model performs well and can predict unit sales accurately, supporting better inventory planning.

Future Work

Future improvements include adding marketing or advertising data to understand the drivers of footfall. Customer demographics could be included to study audience behavior more deeply. More advanced models can be tested to improve classification results. A dashboard can also be developed to allow real-time monitoring of event performance.

Step 8

Summary

Working with the Luxury Beauty and Cosmetics Popup Events dataset has given me a deeper understanding of how real-world business data can be analyzed and used to make meaningful decisions. I learned how different event features such as brand, region, pricing, customer footfall, and lease duration contribute to sales performance. By preparing the dataset, selecting features, and defining both regression and classification targets, I strengthened my ability to set up a complete machine-learning workflow from start to finish.

Through the modeling process, I gained hands-on experience with data preprocessing techniques like scaling, one-hot encoding, handling missing values, and converting date variables into usable numeric formats. Building a Random Forest regression model helped me understand how predictive algorithms learn patterns in the data and how to evaluate their performance using metrics such as MAE, RMSE, and R-squared. Interpreting these results improved my ability to judge model accuracy and reliability in a business context.

Overall, this analysis improved my skills in data cleaning, feature engineering, regression modeling, and performance evaluation. Most importantly, it taught me how to translate raw event data into insights that can guide strategic decisions, such as which locations perform best, what pricing strategies might work, and how customer engagement impacts sales. This hands-on experience has strengthened my confidence in working with real datasets and applying analytics to solve practical business problems.