Online Shoppers Purchase Intention Dashboard

Complete Technical Documentation

Garima Daryani | daryani.g@northeastern.edu

Dataset: Online Shoppers Purchasing Intention (UCI/Kaggle)

Link : h0ps://www.kaggle.com/datasets/imakash3011/online-shoppers-purchasing-intenBon-dataset

Percentage of Effort by Student: 100%

Submission Date: December 13, 2025

📋 Table of Contents

---

1. Project Overview

Project Name: Online Shoppers Purchase Intention Dashboard

Tool: Microsoft Power BI Desktop

Dataset Source: Kaggle - Online Shoppers Intention Dataset

Dataset Size: 12,330 rows × 18 columns

Dashboard Pages: 5 interactive pages

Total Visuals: 15+ visualizations

Executive Summary

This project analyzes online shopping behavior using a comprehensive dataset of 12,330 user sessions from an e-commerce website. The primary objective is to develop an interactive Power BI dashboard that provides actionable insights into customer purchasing pa0erns, engagement behaviors, and conversion drivers.

Through systematic feature engineering and visual analytics, this project identifies key factors influencing online purchase decisions. The dashboard reveals that deep session engagement, product-focused browsing, and multi-section exploration are strong predictors of conversion, with conversion rates varying significantly across visitor types, temporal periods, and traffic sources.

The dashboard serves as a strategic tool for understanding customer journeys, optimizing marketing efforts, and improving website performance to drive revenue growth.

Project Background

With the rapid rise of e-commerce, understanding online shopping behavior has become crucial for businesses seeking to improve customer experience and increase sales. The dataset on online shopping intentions provides valuable insights into how users interact with websites such as the time spent browsing, the type of sections visited, and the factors influencing whether they make a purchase. By analyzing and visualizing this data, we can uncover pa0erns that highlight the relationship between browsing behavior, website interactivity, and purchasing decisions.

My interest in this project stems from my own curiosity as an online shopper who often explores multiple websites, compares options, and waits for deals before buying. Beyond visualization, this dataset also offers opportunities to apply machine learning techniques, such as logistic regression, random forest to forecast purchase pa0erns. Combining personal motivation with the broader potential of ML-driven insights makes this project both engaging and impactful, while highlighting its real-world applications for online retailers.

Problem Statement

The challenge is to understand online shopping intentions by analyzing user behavior data through visualization techniques, uncovering pa0erns that can improve customer experience and increase conversion rates.

Objective

The primary objectives of this project are:

1. **Data Preparation & Feature Engineering**
   - Transform raw session data into meaningful analytical features
   - Create derived metrics that capture user engagement and behavior pa0erns
   - Develop categorical segments for intuitive analysis

2. **Dashboard Development**
   - Design an interactive, user-friendly Power BI dashboard
   - Implement multiple analytical perspectives (temporal, behavioral, technical)
   - Enable filtering and drill-down capabilities for detailed exploration

3. **Insight Genera=on**
   - Identify pa0erns and trends in customer purchasing behavior
   - Analyze the relationship between engagement metrics and conversion
   - Provide actionable recommendations for business stakeholders

4. **Visual Communication**
   - Present complex data relationships through intuitive visualizations
   - Ensure accessibility for both technical and non-technical audiences
   - Support data-driven decision-making processes

---

2. Data Source & Structure

Original Dataset Schema

Column Name	Data Type	Description	Example Values
Administrative	Integer	Number of admin pages visited	0, 1, 2, 3...
Administrative_Duration	Float	Time spent on admin pages (seconds)	0.0, 45.2, 120.5...
Informational	Integer	Number of info pages visited	0, 1, 2, 3...
Informational_Duration	Float	Time spent on info pages (seconds)	0.0, 30.1, 88.3...
ProductRelated	Integer	Number of product pages visited	0, 1, 5, 12...
ProductRelated_Duration	Float	Time spent on product pages (seconds)	0.0, 150.2, 450.8...
BounceRates	Float	Bounce rate for the session	0.0 to 0.20
ExitRates	Float	Exit rate for the session	0.0 to 0.20
PageValues	Float	Average page value	0.0 to 361.76
SpecialDay	Float	Proximity to special day	0.0 to 1.0
Month	Text	Month of session	"Jan", "Feb", "Mar"...
OperatingSystems	Integer	OS identifier	1, 2, 3, 4, 5, 6, 7, 8
Browser	Integer	Browser identifier	1, 2, 3, 4, 5... 13
Region	Integer	Geographic region	1, 2, 3... 9
TrafficType	Integer	Traffic source type	1, 2, 3... 20
VisitorType	Text	New or returning	"Returning_Visitor", "New_Visitor", "Other"
Weekend	Boolean	Weekend session flag	TRUE, FALSE
Revenue	Boolean	Conversion flag	TRUE, FALSE

Data Quality Assessment

Completeness: 100% - No missing values
Consistency: Verified data types and value ranges
Validity: All categorical values within expected ranges
Uniqueness: Session-level data (each row = one session)

---

3. Data Model

Primary Table: Fact Table (online_shoppers_intention)

```
online_shoppers_intention (Fact Table)
├── Session Attributes (18 original columns)
├── Engineered Features (15+ calculated columns)
├── Relationships: None (single table model)
└── Granularity: One row per shopping session
```

Supporting Tables Created 1. Funnel Stages (For Funnel Chart)

dax

```
FunnelStages =
{
    ("All Sessions", 1),
    ("Product Viewers", 2),
    ("Deep Sessions", 3),
    ("High Value Pages", 4),
    ("Converters", 5)
}
```

2. WaterfallStages (For Waterfall Chart)

dax

```
WaterfallStages =
DATATABLE(
    "Stage", STRING,
    "Order", INTEGER,
    "Type", STRING,
    {
        {"All Sessions", 1, "Start"},
        {"Quick Bounce", 2, "Decrease"},
        {"Multi-Explorer", 3, "Increase"},
        {"High Engagement", 4, "Increase"},
        {"Converters", 5, "Total"}
    }
)
```

3. RegionStats (For Regional Analysis)

dax

```
RegionStats =
ADDCOLUMNS(
    VALUES('online_shoppers_intention'[Region]),
    "Total_Sessions",
        CALCULATE(COUNTROWS('online_shoppers_intention')),
    "Converters",
        CALCULATE(
            COUNTROWS('online_shoppers_intention'),
            'online_shoppers_intention'[Revenue] = TRUE()
        ),
    "Conversion_Rate",
        DIVIDE(
            CALCULATE(
                COUNTROWS('online_shoppers_intention'),
                'online_shoppers_intention'[Revenue] = TRUE()
            ),
            CALCULATE(COUNTROWS('online_shoppers_intention')),
            0
        ),
    "Region_Activity",
        VAR SessionCount = CALCULATE(COUNTROWS('online_shoppers_intention'))
        RETURN
            IF(SessionCount > 1500, "High_Traffic_Region",
                IF(SessionCount > 500, "Medium_Traffic_Region", "Low_Traffic_Region")
            )
)
```

---

4. Feature Engineering {#feature-engineering}

A. Page Type Preference Features 1. Page Ratios

m

```
// Power Query - Custom Columns
Page_Admin_Ratio =
    [Administrative] / ([Administrative] + [Informational] + [ProductRelated] + 0.001)

Page_Info_Ratio =
    [Informational] / ([Administrative] + [Informational] + [ProductRelated] + 0.001)

Page_Product_Ratio =
    [ProductRelated] / ([Administrative] + [Informational] + [ProductRelated] + 0.001)
```

2. Dominant Page Type

```
m

Dominant_Page_Type =
    if [ProductRelated] > [Administrative] and [ProductRelated] > [Informational] then "Product_Focused"
    else if [Informational] > [Administrative] and [Informational] > [ProductRelated] then "Info_Focused"
    else if [Administrative] > [Informational] and [Administrative] > [ProductRelated] then "Admin_Focused"
    else "Balanced"
```

B. Duration Efficiency Features 1. Duration Per Page

```
m

Duration_Per_Page =
    ([Administrative_Duration] + [Informational_Duration] + [ProductRelated_Duration]) /
    ([Administrative] + [Informational] + [ProductRelated] + 0.001)
```

2. Product Page Efficiency

```
m

Product_Duration_Per_Page =
    if [ProductRelated] = 0 then 0
    else [ProductRelated_Duration] / [ProductRelated]
```

3. Duration Category

```
m
```

```
Duration_Category =
    if [Duration_Per_Page] < 30 then "Quick_Browser"
    else if [Duration_Per_Page] < 60 then "Normal_Browser"
    else "Deliberate_Browser"
```

C. Behavioral Flags 1. Session Depth Flag

```
m

Is_Deep_Session =
    if ([Administrative] + [Informational] + [ProductRelated]) >= 10 then "Yes"
    else "No"
```

2. Product Engagement Flag

```
m

High_Product_Engagement =
    if [ProductRelated] >= 5 then "Yes" else "No"
```

3. Multi-Page-Type Explorer

```
m

Explored_Multiple_Types =
    if [Administrative] > 0 and [Informational] > 0 and [ProductRelated] > 0 then "Yes"
    else "No"
```

4. Quick Bounce Flag

```
m

Quick_Bounce =
    if [BounceRates] > 0.05 then "Yes" else "No"
```

5. High Value Page Flag

```
m



High_Page_Value =
    if [PageValues] > 0 then "Yes" else "No"
```

D. Exit vs Bounce Comparison 1. Exit-Bounce Difference

```
m

Exit_Bounce_Diff = [ExitRates] - [BounceRates]
```

2. Abandonment Type

```
m

Abandonment_Type =
    if [BounceRates] > 0.05 and [ExitRates] > 0.05 then "High_Abandonment"
    else if [BounceRates] > 0.05 then "Immediate_Exit"
    else if [ExitRates] > 0.05 then "Late_Exit"
    else "Engaged"
```

E. Temporal Features 1. Shopping Period

```
m

Shopping_Period =
    if [Month] in {"Nov", "Dec"} then "Holiday_Season"
    else if [Month] in {"Feb", "May"} then "Valentine_Mothers"
    else "Regular_Season"
```

2. Special Day Category

```
m

Special_Day_Category =
    if [SpecialDay] = 0 then "No_Special_Day"
    else if [SpecialDay] <= 0.4 then "Near_Special_Day"
    else if [SpecialDay] <= 0.8 then "Close_To_Special_Day"
    else "On_Special_Day"
```

3. Shopper Type Time

```
m

Shopper_Type_Time =
    if [Weekend] = true then "Weekend_Shopper"
    else "Weekday_Shopper"
```

4. Month Sorting

```
m

MonthNumber =
    List.PositionOf(
        {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"},
        [Month]
    ) + 1

MonthDate = #date(2024, [MonthNumber], 1)
```

F. Visitor Behavior Patterns 1. Visitor Experience

```
m

Visitor_Experience =
    if [VisitorType] = "Returning_Visitor" then "Experienced"
    else "New"
```

2. Visitor + Weekend Combo

```
m

Visitor_Weekend_Combo =
    if [VisitorType] = "Returning_Visitor" and [Weekend] = true then "Returning_Weekend"
    else if [VisitorType] = "Returning_Visitor" then "Returning_Weekday"
    else if [Weekend] = true then "New_Weekend"
    else "New_Weekday"
```

G. Complexity Score 1. Session Complexity Score

```
dax


Session_Complexity_Score =
    IF('online_shoppers_intention'[Administrative] > 0, 1, 0) +
    IF('online_shoppers_intention'[Informational] > 0, 1, 0) +
    IF('online_shoppers_intention'[ProductRelated] > 0, 1, 0)
```

2. Complexity Category

dax

```
Complexity_Category =
  SWITCH(
    'online_shoppers_intention'[Session_Complexity_Score],
    3, "Multi_Section_Explorer",
    2, "Two_Section_Visitor",
    1, "Single_Section_Visitor",
    "Unknown"
  )
```

H. Traffic Quality Tier

dax

```
Traffic_Quality_Tier =
VAR CurrentTraffic = 'online_shoppers_intention'[TrafficType]
VAR Converters =
  CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Revenue] = TRUE(),
    ALL('online_shoppers_intention'),
    'online_shoppers_intention'[TrafficType] = CurrentTraffic
  )
VAR TotalSessions =
  CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    ALL('online_shoppers_intention'),
    'online_shoppers_intention'[TrafficType] = CurrentTraffic
  )
VAR ConvRate = DIVIDE(Converters, TotalSessions, 0)

RETURN
  IF(ConvRate > 0.20, "High_Converting_Traffic",
    IF(ConvRate > 0.10, "Medium_Converting_Traffic",
      "Low_Converting_Traffic"
    )
  )
```

---

5. DAX Measures {#dax-measures}

A. Core Conversion Metrics 1. Total Sessions

dax

```
Total_Sessions = COUNTROWS('online_shoppers_intention')
```

2. Total Converters

dax

```
Total_Converters =
CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Revenue] = TRUE()
)
```

3. Conversion Rate

dax

```
Conversion_Rate =
DIVIDE(
    [Total_Converters],
    [Total_Sessions],
    0
)
```

4. Average Duration Per Page

dax

```
Avg_Duration_Per_Page =
AVERAGE('online_shoppers_intention'[Duration_Per_Page])
```

B. Engagement Metrics 1. Deep Session Rate

dax

```
Deep_Session_Rate =
DIVIDE(
    CALCULATE(
        COUNTROWS('online_shoppers_intention'),
        'online_shoppers_intention'[Is_Deep_Session] = "Yes"
    ),
    COUNTROWS('online_shoppers_intention'),
    0
)
```

2. High Product Engagement Rate

dax

```
High_Product_Engagement_Rate =
DIVIDE(
    CALCULATE(
        COUNTROWS('online_shoppers_intention'),
        'online_shoppers_intention'[High_Product_Engagement] = "Yes"
    ),
    [Total_Sessions],
    0
)
```

3. Multi-Section Explorer Rate

dax

```
Multi_Section_Rate =
DIVIDE(
    CALCULATE(
        COUNTROWS('online_shoppers_intention'),
        'online_shoppers_intention'[Complexity_Category] = "Multi_Section_Explorer"
    ),
    [Total_Sessions],
    0
)
```

C. Abandonment Metrics 1. Average Bounce Rate

dax

```
Avg_Bounce_Rate = AVERAGE('online_shoppers_intention'[BounceRates])
```

2. Average Exit Rate

dax

```
Avg_Exit_Rate = AVERAGE('online_shoppers_intention'[ExitRates])
```

3. Quick Bounce Rate

dax

```
Quick_Bounce_Rate =
DIVIDE(
    CALCULATE(
        COUNTROWS('online_shoppers_intention'),
        'online_shoppers_intention'[Quick_Bounce] = "Yes"
    ),
    [Total_Sessions],
    0
)
```

4. High Abandonment Sessions

dax

```
High_Abandonment_Sessions =
CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Abandonment_Type] = "High_Abandonment"
)
```

D. Segment Performance Metrics 1. Returning Visitor Conversion Rate

dax

```
Returning_Conv_Rate =
CALCULATE(
    [Conversion_Rate],
    'online_shoppers_intention'[VisitorType] = "Returning_Visitor"
)
```

2. New Visitor Conversion Rate

dax

```
New_Visitor_Conv_Rate =
CALCULATE(
    [Conversion_Rate],
    'online_shoppers_intention'[VisitorType] = "New_Visitor"
)
```

3. Weekend Conversion Rate

dax

```
Weekend_Conv_Rate =
CALCULATE(
    [Conversion_Rate],
    'online_shoppers_intention'[Weekend] = TRUE()
)
```

4. Holiday Season Performance

dax

```
Holiday_Conv_Rate =
CALCULATE(
    [Conversion_Rate],
    'online_shoppers_intention'[Shopping_Period] = "Holiday_Season"
)
```

E. Funnel Measures 1. Funnel Stage Measures

dax

```
Funnel_AllSessions = COUNTROWS('online_shoppers_intention')

Funnel_ProductViewers =
CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[ProductRelated] > 0
)

Funnel_DeepSessions =
CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Is_Deep_Session] = "Yes"
)

Funnel_HighValue =
CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[High_Page_Value] = "Yes"
)

Funnel_Converters = [Total_Converters]
```

2. Dynamic Funnel Value

dax

```
Funnel_Value =
SWITCH(
    SELECTEDVALUE(FunnelStages[Stage]),
    "All Sessions", [Funnel_AllSessions],
    "Product Viewers", [Funnel_ProductViewers],
    "Deep Sessions", [Funnel_DeepSessions],
    "High Value Pages", [Funnel_HighValue],
    "Converters", [Funnel_Converters],
    BLANK()
)
```

F. Waterfall Measures 1. Waterfall Value Measure

dax

```
Waterfall_Value =
VAR CurrentStage = SELECTEDVALUE(WaterfallStages[Stage])
VAR AllSessions = COUNTROWS('online_shoppers_intention')
VAR QuickBounce = CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Quick_Bounce] = "Yes"
)
VAR MultiExplorer = CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[Complexity_Category] = "Multi_Section_Explorer"
)
VAR HighEng = CALCULATE(
    COUNTROWS('online_shoppers_intention'),
    'online_shoppers_intention'[High_Product_Engagement] = "Yes"
)
VAR Converters = [Total_Converters]

RETURN
SWITCH(
    CurrentStage,
    "All Sessions", AllSessions,
    "Quick Bounce", -QuickBounce,
    "Multi-Explorer", MultiExplorer,
    "High Engagement", HighEng,
    "Converters", Converters,
    BLANK()
)
```

G. Comparative Measures 1. Conversion vs Average

dax

```
Conv_vs_Avg =
VAR OverallAvg =
    CALCULATE(
        [Conversion_Rate],
        ALL('online_shoppers_intention')
    )
RETURN
    [Conversion_Rate] - OverallAvg
```

2. Performance Index

dax

```
Performance_Index =
VAR OverallAvg =
    CALCULATE(
        [Conversion_Rate],
        ALL('online_shoppers_intention')
    )
RETURN
    DIVIDE([Conversion_Rate], OverallAvg, 0)
```

---

6. Power Query Transformations

A. Data Import & Initial Cleaning

m

```
let
    Source = Csv.Document(
        File.Contents("C:\path\to\online_shoppers_intention.csv"),
        [Delimiter=",", Columns=18, Encoding=65001, QuoteStyle=QuoteStyle.None]
    ),
    PromotedHeaders = Table.PromoteHeaders(Source, [PromoteAllScalars=true]),
    ChangedType = Table.TransformColumnTypes(
        PromotedHeaders,
        {
            {"Administrative", Int64.Type},
            {"Administrative_Duration", type number},
            {"Informational", Int64.Type},
            {"Informational_Duration", type number},
            {"ProductRelated", Int64.Type},
            {"ProductRelated_Duration", type number},
            {"BounceRates", type number},
            {"ExitRates", type number},
            {"PageValues", type number},
            {"SpecialDay", type number},
            {"Month", type text},
            {"OperatingSystems", Int64.Type},
            {"Browser", Int64.Type},
            {"Region", Int64.Type},
            {"TrafficType", Int64.Type},
            {"VisitorType", type text},
            {"Weekend", type logical},
            {"Revenue", type logical}
        }
    )
in
    ChangedType
```

B. Feature Engineering Steps

Complete Power Query Script with All Features:

```
m
```

```
let
    Source = [Previous Steps],

    // Add Duration Per Page
    AddDurationPerPage = Table.AddColumn(
        Source,
        "Duration_Per_Page",
        each ([Administrative_Duration] + [Informational_Duration] + [ProductRelated_Duration]) /
            ([Administrative] + [Informational] + [ProductRelated] + 0.001),
type number
    ),

    // Add Duration Category
    AddDurationCategory = Table.AddColumn(
        AddDurationPerPage,        "Duration_Category",        each
if [Duration_Per_Page] < 30 then "Quick_Browser"              else
if [Duration_Per_Page] < 60 then "Normal_Browser"             else
"Deliberate_Browser",        type text
    ),

    // Add Dominant Page Type
    AddDominantPageType = Table.AddColumn(
        AddDurationCategory,          "Dominant_Page_Type",          each if [ProductRelated] > [Administrative] and
[ProductRelated] > [Informational] then "Product_Focused"             else if [Informational] > [Administrative] and
[Informational] > [ProductRelated] then "Info_Focused"               else if [Administrative] > [Informational] and
[Administrative] > [ProductRelated] then "Admin_Focused"             else "Balanced",        type text
    ),

    // Add Behavioral Flags
    AddIsDeepSession = Table.AddColumn(
        AddDominantPageType,        "Is_Deep_Session",        each if ([Administrative] +
[Informational] + [ProductRelated]) >= 10 then "Yes" else "No",        type text
    ),

    AddHighProductEngagement = Table.AddColumn(
        AddIsDeepSession,
```

```
        "High_Product_Engagement",         each if
[ProductRelated] >= 5 then "Yes" else "No",         type
text
    ),

    AddQuickBounce = Table.AddColumn(
        AddHighProductEngagement,
"Quick_Bounce",         each if [BounceRates] > 0.05
then "Yes" else "No",         type text
    ),

    AddHighPageValue = Table.AddColumn(
        AddQuickBounce,
"High_Page_Value",
        each if [PageValues] > 0 then "Yes" else "No",
type text
    ),

    // Add Abandonment Type
    AddAbandonmentType = Table.AddColumn(
        AddHighPageValue,
"Abandonment_Type",
        each if [BounceRates] > 0.05 and [ExitRates] > 0.05 then "High_Abandonment"
else if [BounceRates] > 0.05 then "Immediate_Exit"              else if [ExitRates] >
0.05 then "Late_Exit"              else "Engaged",         type text
    ),

    // Add Shopping Period
    AddShoppingPeriod = Table.AddColumn(
        AddAbandonmentType,         "Shopping_Period",         each if
List.Contains({"Nov", "Dec"}, [Month]) then "Holiday_Season"              else if
List.Contains({"Feb", "May"}, [Month]) then "Valentine_Mothers"              else
"Regular_Season",         type text
    ),

    // Add Special Day Category
    AddSpecialDayCategory = Table.AddColumn(
        AddShoppingPeriod,
        "Special_Day_Category",
```

```
        each if [SpecialDay] = 0 then "No_Special_Day"
            else if [SpecialDay] <= 0.4 then "Near_Special_Day"
            else if [SpecialDay] <= 0.8 then "Close_To_Special_Day"
            else "On_Special_Day",
        type text
    ),

    // Add Visitor Weekend Combo
    AddVisitorWeekendCombo = Table.AddColumn(
        AddSpecialDayCategory,
        "Visitor_Weekend_Combo",
        each if [VisitorType] = "Returning_Visitor" and [Weekend] = true then "Returning_Weekend"
            else if [VisitorType] = "Returning_Visitor" then "Returning_Weekday"
            else if [Weekend] = true then "New_Weekend"
            else "New_Weekday",
        type text
    ),

    // Add Month Sorting
    MonthList = {"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"},
    AddMonthNumber = Table.AddColumn(
        AddVisitorWeekendCombo,
        "MonthNumber",
        each List.PositionOf(MonthList, [Month]) + 1,
        Int64.Type
    ),

    AddMonthDate = Table.AddColumn(
        AddMonthNumber,
        "MonthDate",
        each #date(2024, [MonthNumber], 1),
        type date
    )
in
    AddMonthDate
```

7. Exploratory Data Analysis

1.1 1. Import Libraries and Load Data:

[23]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

[11]:
```
df = pd.read_csv('online_shoppers_intention.csv')
```

1.2 2. Data Inspection:

[12]:
```
df.head()
```

[12]:
```
 Administrative Administrative_Duration Informational \
0              0  0.0   0
1              0  0.0   0
2              0  0.0   0
3              0  0.0   0
4              0  0.0   0

   Informational_Duration ProductRelated ProductRelated_Duration \
0                     0.0             1                0.000000
1                     0.0             2               64.000000
2                     0.0             1                0.000000
3                     0.0             2                2.666667
4                     0.0            10              627.500000
   BounceRates ExitRates PageValues SpecialDay Month OperatingSystems \
0     0.20  0.20  0.0   0.0 Feb    1 1   0.00  0.10  0.0   0.0 Feb    2 2
                        0.20  0.20  0.0   0.0 Feb    4

  3     0.05  0.14  0.0   0.0 Feb    3 4   0.02  0.05  0.0   0.0 Feb    3

   Browser Region TrafficType     VisitorType Weekend Revenue

0       1  1     1 Returning_Visitor   False False
1       2  1     2 Returning_Visitor   False False
2       1  9     3 Returning_Visitor   False False
3       2  2     4 Returning_Visitor   False False
4       3  1     4 Returning_Visitor   True  False
```

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
 #   Column                  Non-Null Count Dtype
---  ------                  -------------- -----
 0   Administrative          12330 non-null int64
 1   Administrative_Duration 12330 non-null float64
 2   Informational           12330 non-null int64
 3   Informational_Duration  12330 non-null float64
 4   ProductRelated          12330 non-null int64
```

```
 5   ProductRelated_Duration 12330 non-null float64
 6  BounceRates             12330       non-null
                             float64
 7  ExitRates               12330       non-null
                             float64
 8  PageValues              12330       non-null
                             float64
 9  SpecialDay              12330       non-null
                             float64
 10 Month                   12330       non-null
                             object
 11 OperatingSystems        12330 non-null
                             int64
 12 Browser                 12330 non-null
                             int64
 13 Region                  12330 non-null
                             int64
 14 TrafficType             12330 non-null
                             int64
 15 VisitorType             12330       non-null
                             object
 16 Weekend                 12330 non-null bool
 17 Revenue                 12330 non-null bool
dtypes: bool(2), float64(7), int64(7), object(2)
memory usage: 1.5+ MB
```

1. Dataset size: The dataset contains 12,330 rows.
2. Data types: The data types present are integers, floating-point numbers, strings, and booleans.
3. Data integrity: There are no notable issues with the data types or columns, and they are all appropriate. Missing values: No columns contain empty or missing values.

[8]:
```
df.shape
```

[8]:
```
(12330, 18)
```

[9]:
```
df.columns
```

[9]:
```
Index(['Administrative', 'Administrative_Duration', 'Informational',
       'Informational_Duration', 'ProductRelated',
       'ProductRelated_Duration',
       'BounceRates', 'ExitRates', 'PageValues', 'SpecialDay', 'Month',
  'OperatingSystems', 'Browser', 'Region', 'TrafficType', 'VisitorType',
       'Weekend', 'Revenue'],
      dtype='object')
```

1.3 3. Data Cleaning and Preprocessing:

[12]:
```
df.isnull().sum()
```

[12]:
```
Administrative             0
Administrative_Duration    0
Informational              0
Informational_Duration     0
ProductRelated             0
ProductRelated_Duration    0
BounceRates                0
ExitRates                  0
```

```
PageValues              0
SpecialDay              0
Month                   0
OperatingSystems        0
Browser                 0
Region                  0
TrafficType             0
VisitorType             0
Weekend                 0
Revenue                 0
dtype:
int64
```

[15]:
```
df.duplicated().sum()
```

[15]:
```
np.int64(125)
```

[19]:
```
# Upon examining the 'Month' column, it was observed that most months are abbreviated (e.g., JUL, AUG, DEC), but one entry uses 'June' instead of 'Jun'.
# To maintain consistency, this will be corrected.
df.loc[df['Month'] == 'June', 'Month'] = 'Jun'

#There is an absence of data for January and April. These months will be excluded from the analysis for consistency.
month_order = ['Feb', 'Mar', 'May', 'Jun',
               'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']


df['Month'] = pd.Categorical(df['Month'], categories=month_order, ordered=True)
```

1.4 4. Statistical Summary

[112]:
```
# column classification based on data type nums =
['Administrative', 'Administrative_Duration', 'Informational', 'Informational_Duration', 'ProductRelated', 'ProductRelated_Duration', 'BounceRates', 'ExitRates', 'PageValues', 'SpecialDay','Revenue']
cats = ['Month', 'OperatingSystems', 'Browser', 'Region',
'TrafficType', 'VisitorType', 'Weekend', 'Revenue']
df[nums] = df[nums].round(1)
```

[27]:
```
df_cats = df[cats].astype('category').copy()
```

[28]:
```
df_cats.describe()
```

[28]:
```
       Month OperatingSystems Browser Region TrafficType \
count  12330           12330   12330  12330       12330
unique    10               8      13      9          20
top      May               2       2      1           2
freq    3364            6601    7961   4780        3913

              VisitorType Weekend
              Revenue
```

```
count            12330  12330   12330
unique               3      2       2
top    Returning_VisitorFalse   False
freq             10551   9462   10422
```

[29]:
```
df.describe()
```

[29]:
```
       Administrative Administrative_Duration Informational \
count    12330.000000           12330.000000  12330.000000
mean         2.315166              80.818662      0.503569
std          3.321784             176.779240      1.270156
min          0.000000               0.000000      0.000000
25%          0.000000               0.000000      0.000000
50%          1.000000               7.500000      0.000000
75%          4.000000              93.250000      0.000000
max         27.000000            3398.800000     24.000000

       Informational_Duration                       ProductRelated
       ProductRelated_Duration \
count          12330.000000  12330.000000          12330.000000
mean              34.472482     31.731468           1194.746602
std              140.749220     44.475503           1913.669359
min                0.000000      0.000000              0.000000
25%                0.000000      7.000000            184.125000
50%                0.000000     18.000000            598.950000
75%                0.000000     38.000000           1464.150000
max             2549.400000    705.000000          63973.500000
       BounceRates    ExitRates  PageValues SpecialDay \ count
12330.000000 12330.000000 12330.000000 12330.000000
mean      0.016642    0.029765    5.889416    0.061427
std       0.050807    0.058917   18.568865    0.198917
min       0.000000    0.000000    0.000000    0.000000
25%       0.000000    0.000000    0.000000    0.000000
50%       0.000000    0.000000    0.000000    0.000000
75%       0.000000    0.000000    0.000000    0.000000
max       0.200000    0.200000  361.800000    1.000000

       OperatingSystems    Browser       Region TrafficType
count              12330.000000 12330.000000 12330.000000
                                                12330.000000
mean          2.124006    2.357097    3.147364    4.069586
std           0.911325    1.717277    2.401591    4.025169
min           1.000000    1.000000    1.000000    1.000000
25%           2.000000    2.000000    1.000000    2.000000
50%           2.000000    2.000000    3.000000    2.000000
75%           3.000000    2.000000    4.000000    4.000000
max           8.000000   13.000000    9.000000   20.000000
```

1.5 5. Value Counting

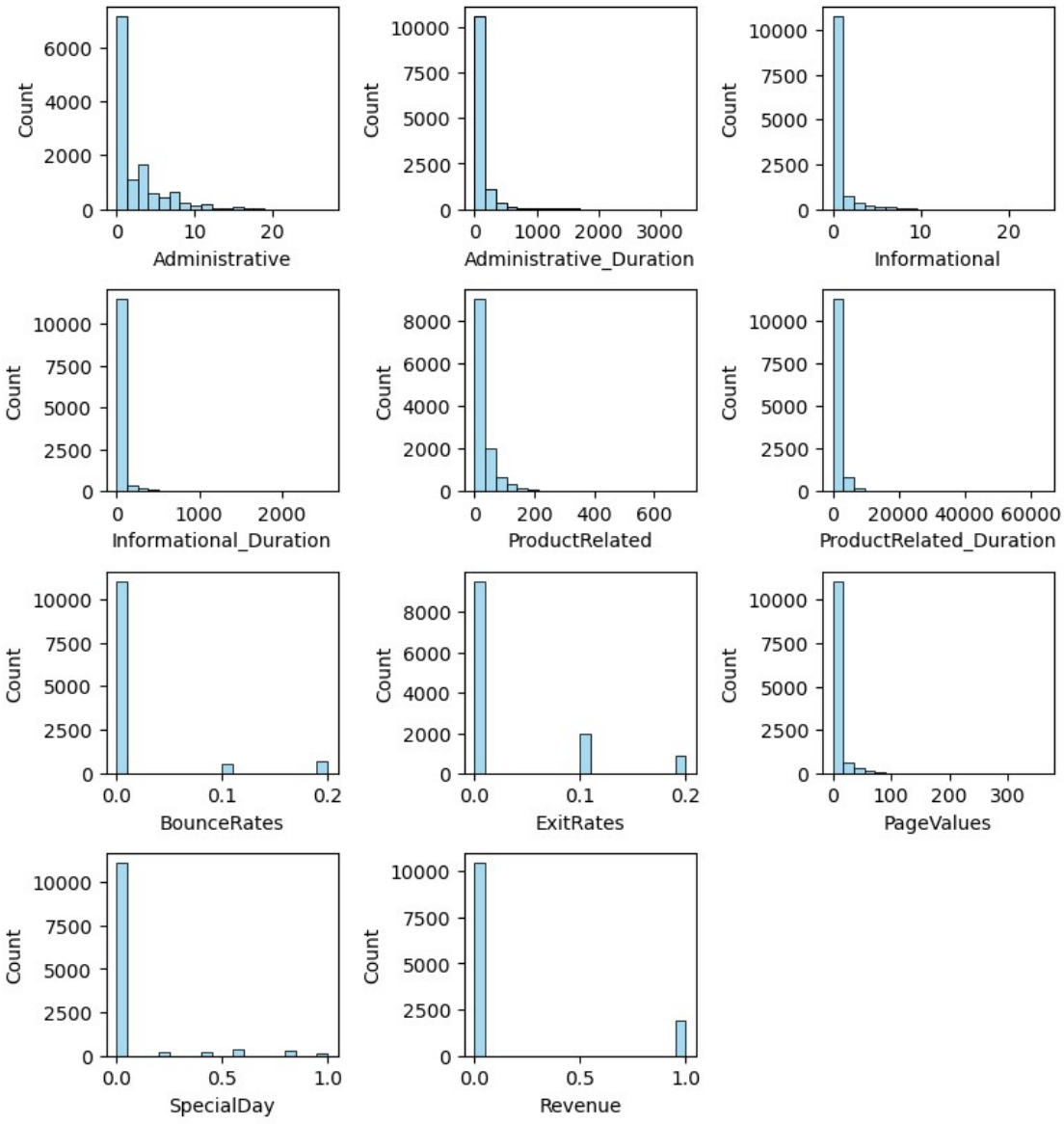
[30]:
```
for col in cats:
    print(f'''Value count in column {col}:''')
    print(df[col].value_counts())
    print()
```

```
Value count in column Month:
Month
May   3364
Nov   2998
Mar   1907
Dec   1727
Oct    549
Sep    448
Aug    433
Jul    432
Jun    288
Feb    184
Name: count, dtype: int64

Value count in column OperatingSystems:
OperatingSystems
2   6601
1   2585
3    2555
4    478
8     79
6      19
7      7
5      6
Name: count, dtype: int64

Value count in column Browser:
Browser
2    7961
1 2462 4   736
5     467
6     174
10   163 8 135
3     105
13     61
7      49
12     10
11      6
9       1
Name: count, dtype: int64

Value count in column Region:
Region
1   4780
3   2403
4   1182
```

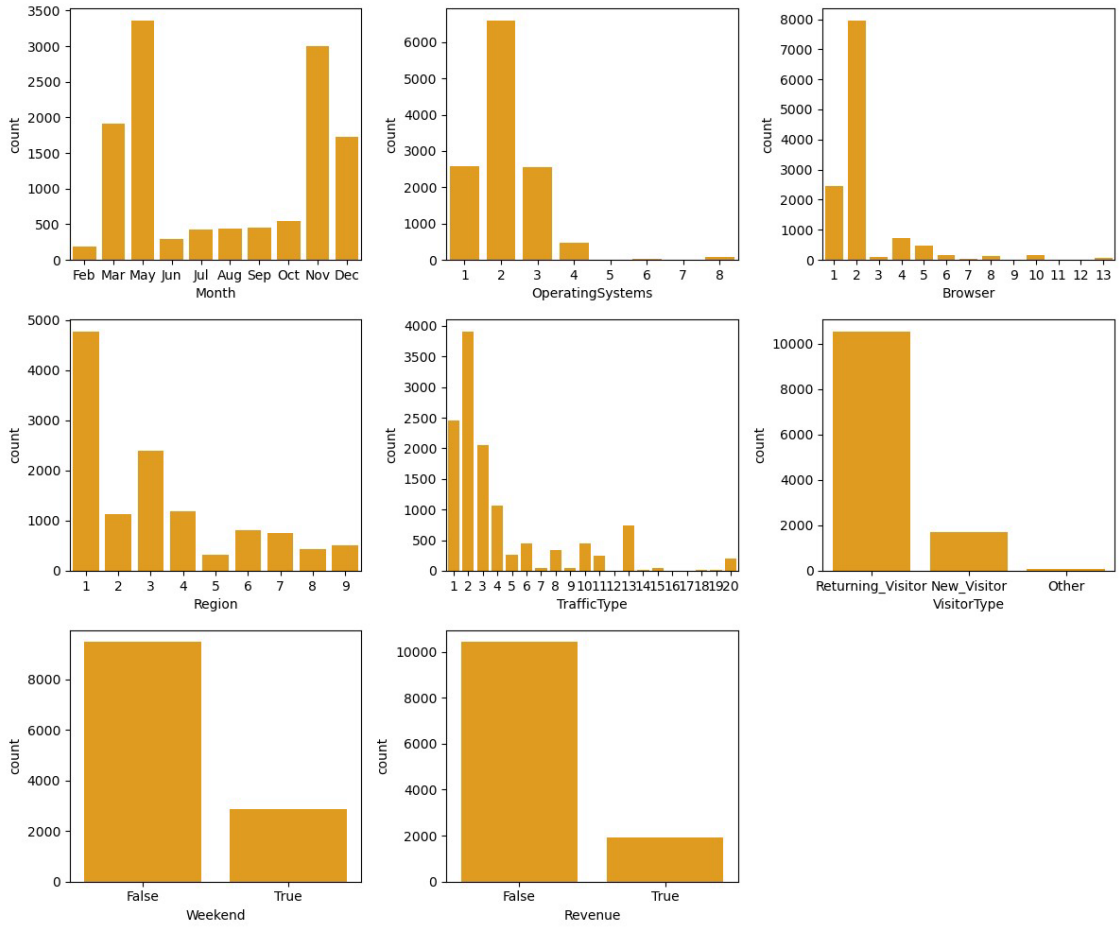
```
2   1136
6    805
7    761
9    511
8    434
5    318
Name: count, dtype: int64

Value count in column TrafficType:
TrafficType
2    3913
1    2451
3    2052
4    1069
13    738
10    450 6 444
8     343
5     260
11    247
20    198
9      42
7      40
15     38
19     17
14     13
18     10
16      3
12      1
17      1
Name: count, dtype: int64

Value count in column VisitorType:
VisitorType
Returning_Visitor  10551
New_Visitor         1694
Other                 85
Name: count, dtype: int64

Value count in column Weekend:
Weekend
False   9462
True    2868
Name: count, dtype: int64

Value count in column Revenue:
Revenue
False 10422 True
1908
Name: count, dtype: int64
```

1.6 6. Univariate Analysis

[113]:
```
# Hist plot (Numerical)
features = nums
plt.figure(figsize=(8, 10))
for i in range(0, len(features)):
    plt.subplot(5, 3,  +1)
    sns.histplot(x=df[features[i]], color='skyblue', bins = 20)
    plt.tight_layout()
```

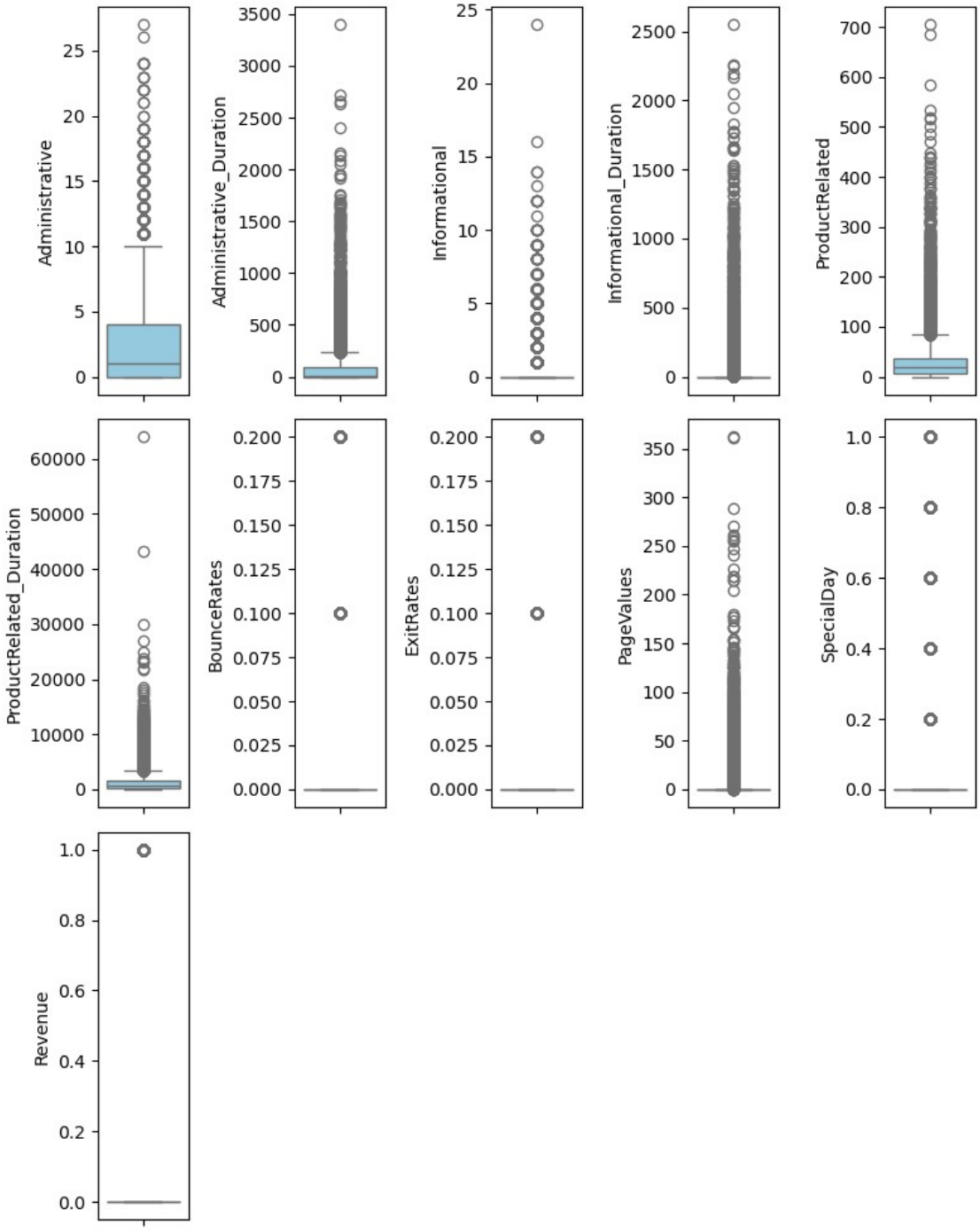


[42]:
```
# Hist plot (categorical)
plt.figure(figsize=(12, 10))
for i in range(0, len(cats)):
    plt.subplot(3, 3,  +1)
    sns.countplot(x = df[cats[i]], color='orange', orient='v')
    plt.tight_layout()
```



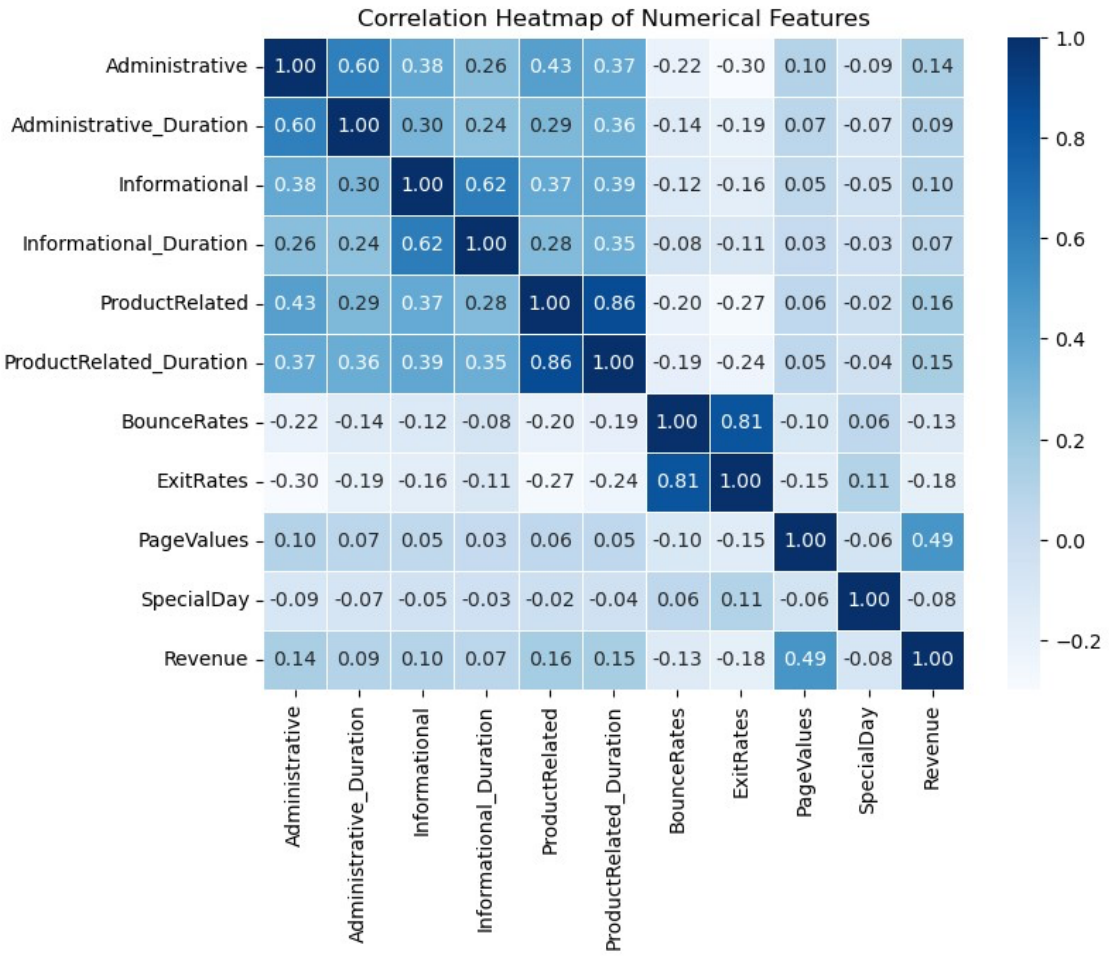
1. Customers are more active in month of may and november.
2. A significant majority of sessions use OS type 2 and Browser type 2.
3. Region 1 accounts for the majority of visitor sessions.
4. Traffic Type 2 dominates session sources.
5. Returning customers are more than new customers or other category.
6. Session activity on non-weekend days are more compared to weekends.
7. Major session activity is not converted into final purchase.

[114]:
```
# Box Plot for Features
features = nums
plt.figure(figsize=(8, 10))
for i in range(0, len(features)):
    plt.subplot(3, 5,  +1)
    sns.boxplot(y=df[features[i]], color='skyblue', orient='v')
    plt.tight_layout()
```


Administrative
Administrative_Duration
Informational
Informational_Duration
ProductRelated
ProductRelated_Duration
BounceRates
ExitRates
PageValues
SpecialDay
Revenue

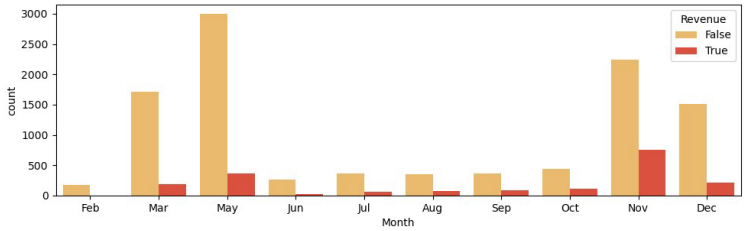
1.7 7. Bivariate Analysis

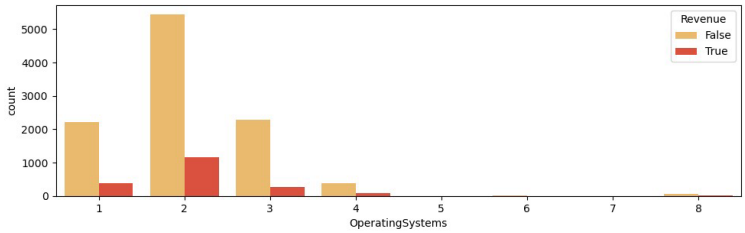
[115]:
```
# Correlation heatmap
correlation_matrix = df[nums].corr()

plt.figure(figsize=(8, 6))
sns.heatmap(
    correlation_matrix,
    annot=True,
    fmt=".2f",
    cmap='Blues',
    cbar=True,
    linewidths=.5
)

plt.title('Correlation Heatmap of Numerical Features ')
plt.show()
```

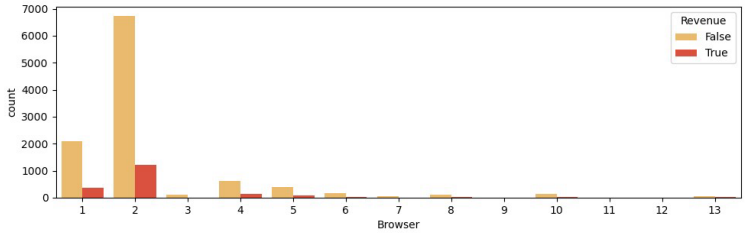


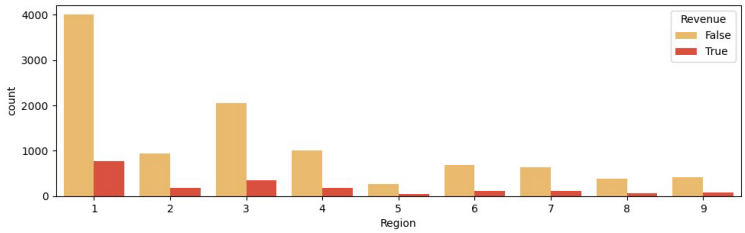
Insights: 1. There is a high correlation(0.49) with page values and revenue. 2. Bounces rates got negative influence on revenue. 3. Pages which has high bounce rates tends to have high exit rates as well (correlation of exit rates and bounce rate is high). 4. The website earns a lot of revenue from product related pages.

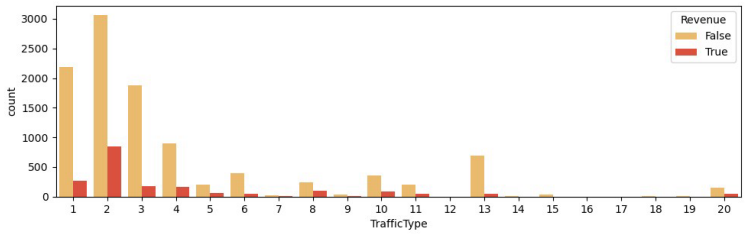
[38]:
```
features = cats
plt.figure(figsize = (10,25))
for i in range (0, len(features)):
    plt.subplot(8, 1,  +1)
    sns.countplot(x=features[i], data=df,  palette="YlOrRd", hue="Revenue")
    plt.tight_layout()
```

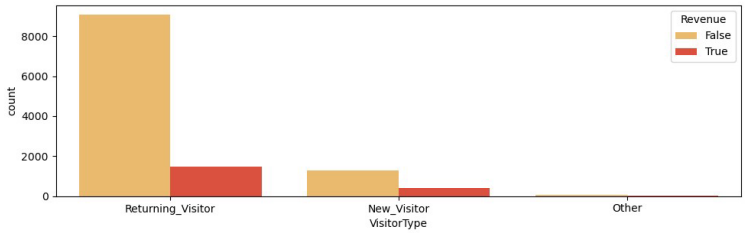

Revenue
False
True
count
Feb
Mar
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

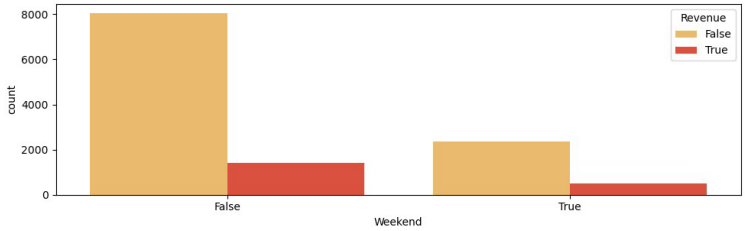

Revenue
False
True
count
OperatingSystems

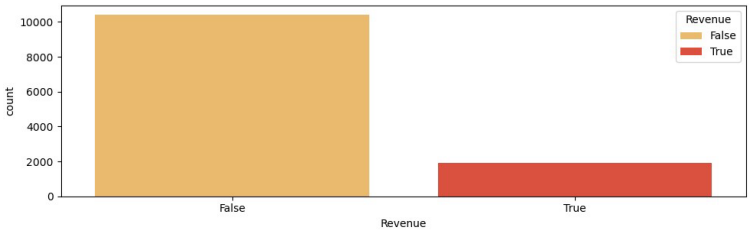

Revenue
False
True
count
Browser


Revenue
False
True
count
Region

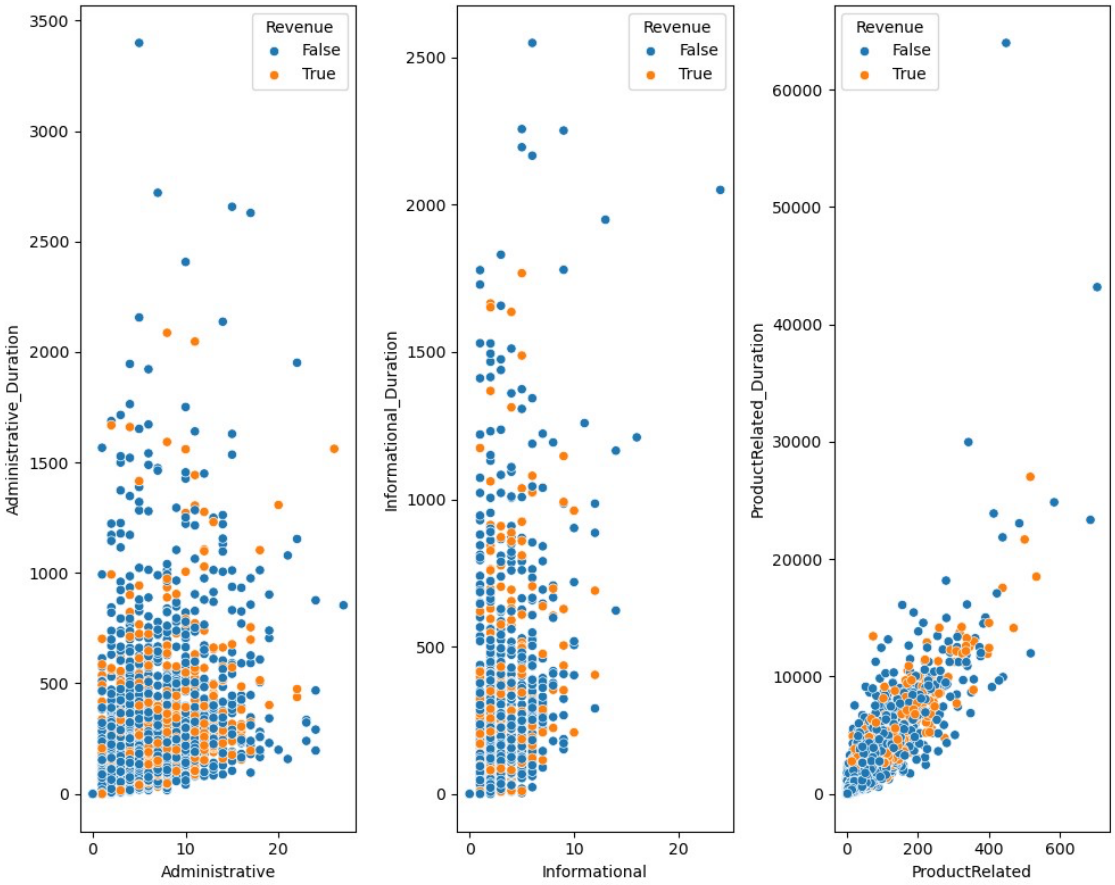

Revenue
False
True
count
TrafficType


Revenue
False
True
count
Returning_Visitor
New_Visitor
Other
VisitorType


Revenue
False
True
count
False
True
Weekend

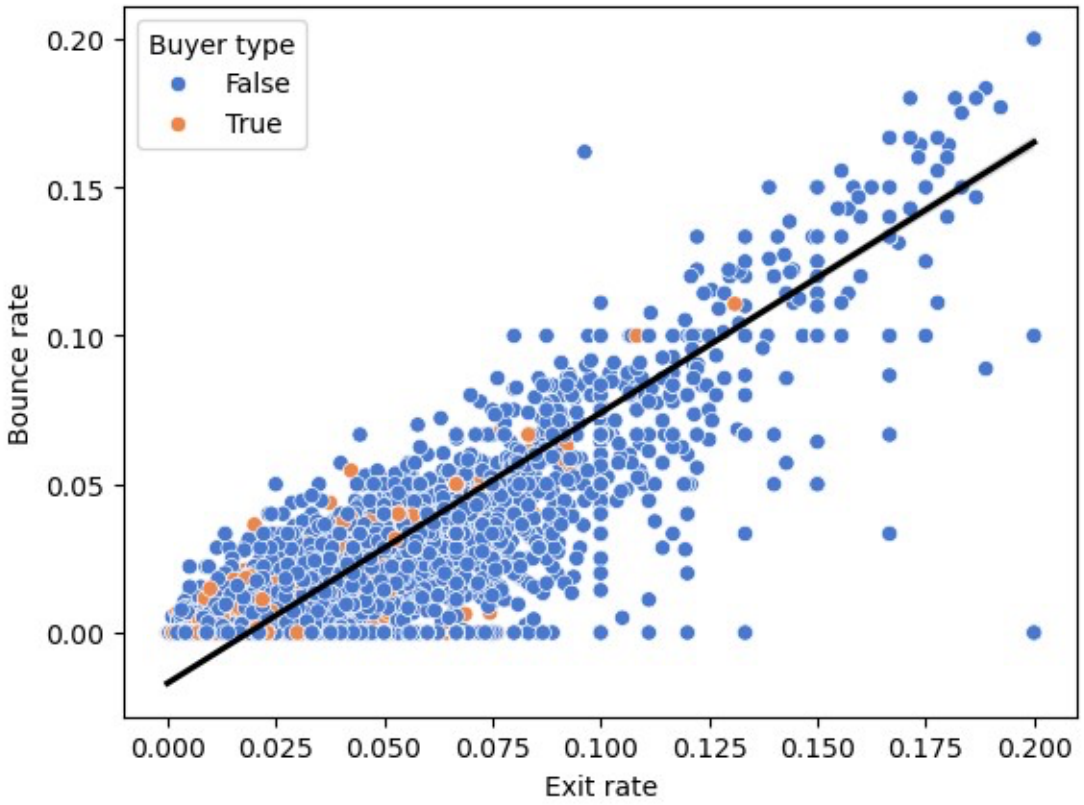

Revenue
False
True
count
False
True
Revenue

[108]:
```
# Duration of pages VS Revenue

plt.figure(figsize = (10,8))
plt.subplot(131)
sns.scatterplot(x="Administrative",  ="Administrative_Duration",hue="Revenue",
 data = df)
plt.subplot(132)
sns.scatterplot(x="Informational",  ="Informational_Duration",hue="Revenue",
 data = df)
plt.subplot(133)
sns.scatterplot(x="ProductRelated",  ="ProductRelated_Duration",hue="Revenue",
 data = df)
plt.tight_layout()
```



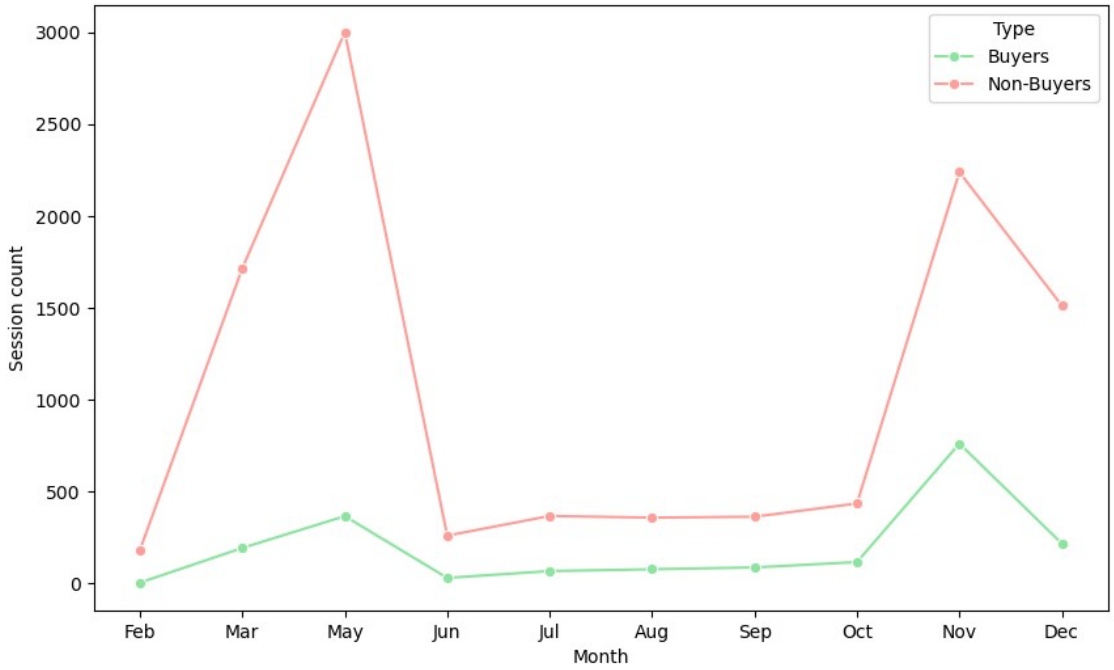
1. Administrative pages: Users spend 0-15 secs on this type of pages which is very low. A major customers spend average duration here might suggest a complex or poorly designed interface.
2. Informational pages : Users spend up to 10 seconds on this pages. This is a very short duration and may indicate that users quickly find the information they need or the content is not engaging enough to hold their attention.
3. Product Related: Users spend up to 5 minutes on product pages. It indicates a high level of interest in the products. A longer average time on product pages is often correlated with higher purchase intent. But the engagement shown on the product pages is not converting into revenue. Despite users spending ample time considering products, they are not completing a purchase.

[15]:
```
# Bounce & Exit Rates Vs Revenue

sns.scatterplot(x = 'ExitRates',    = 'BounceRates', hue='Revenue',
 palette='muted', data = df)
sns.regplot(data=df, x='ExitRates',  ='BounceRates', scatter=False,
 color='Black')
plt.xlabel('Exit rate')
plt.ylabel('Bounce rate')
plt.legend(title='Buyer type')
plt.show()
```



1. The relationship between exit rate and bounce rate reveals a positive correlation. This suggests that increasing exit rates lead to higher bounce rates.
2. The concentration of buyers(revenue = true) is higher in regions with lower bounce and exit rates. This indicates that Sessions with higher exit and bounce rates have a lower likelihood of resulting in a product purchase.
3. Lower exit and bounce rates are conducive to purchases.

[24]:
```
# Trend Analysis

month_customer = df[df['Revenue']==True].groupby('Month')['Revenue'].
  agg(Buyers='count').reset_index()
month_customer['Non-Buyers']=df[df['Revenue']==False].
  groupby('Month')['Revenue'].agg('count').values


plt.figure(figsize=(10, 6))
ax = plt.gca()


buyers_color = sns.color_palette('pastel')[2]
sns.lineplot(data=month_customer, x='Month',  ='Buyers', label='Buyers',␣
  color=buyers_color, marker='o', ax=ax)



non_buyers_color = sns.color_palette('pastel')[3]
sns.lineplot(data=month_customer, x='Month',  ='Non-Buyers',␣
  label='Non-Buyers', color=non_buyers_color, marker='o', ax=ax)


plt.ylabel('Session count')
plt.legend(title="Type")
plt.show()
```



1. The highest session counts are observed in May and November, indicating peak activity for both Buyers and Non-Buyers.
2. This indicates that May and November are significant months for product purchases.

Recommendations for business after performing EDA:

1. Enhance Product-Related Pages: Develop and optimize these pages to foster user engagement and increase sales likelihood.
2. Reduce Bounce and Exit Rates: Implement strategies to engage users effectively, thereby reducing bounce and exit rates.
3. Utilize PageValue Insights: Leverage high PageValues as indicators of user interest and purchase intent to guide content strategy and marketing efforts.
4. Capitalize on Peak Seasons: Develop targeted marketing campaigns and inventory strategies for May and November.
5. Optimize for Key OS and Browser: Focus on ensuring a seamless user experience on OS type 2 and Browser type 2.
6. Focus on Regions: Develop region-specific marketing strategies, particularly for Region 1, to cater to local preferences and trends.
7. Prioritize Effective Traffic Sources: Allocate resources and strategies towards maximizing the potential of Traffic Type 2.

8. Visualizations

Visual Inventory

Visual Type	Count	Purpose	Key Fields
Card	4	KPIs	Total Sessions, Conv Rate, Avg Duration, Converters
Funnel Chart	1	Conversion Journey	FunnelStages, Funnel_Value
Donut Chart	1	Page Type Distribution	Dominant_Page_Type, Count
Line & Column	1	Monthly Trend	Month, Sessions, Conv Rate
Waterfall	1	Session Flow	WaterfallStages, Waterfall_Value
Stacked Bar (100%)	2	Segment Comparison	Complexity, Shopping Period, Revenue
Scatter Plot	1	Duration vs Value	Duration_Per_Page, PageValues, Duration_Category
Matrix	1	Behavior Matrix	Page Type, Deep Session, Engagement, Conv Rate
Bullet Chart	1	Abandonment Metrics	Abandonment_Type, Bounce/Exit Rates, Targets
Ribbon Chart	1	Pattern Changes	Month, Abandonment_Type, Count
Column Chart	1	Exit vs Bounce	Page Type, Bounce/Exit Rates
Gauge	3	KPI Status	Quick Bounce, Exit, Engaged Rates
Treemap	1	Visitor Segments	Visitor_Weekend_Combo, Count, Conv Rate
Decomposition Tree	1	Driver Analysis	Revenue, Multiple dimensions
Sankey Diagram	1	Traffic Flow	Traffic Quality, Page Type, Revenue
Bubble Chart	1	Regional Performance	Region, Sessions, Region_Activity

Detailed Visual Configurations 1. Purchase Journey Funnel

Visual: Funnel Chart
Fields:

- Category: FunnelStages[Stage]
- Values: [Funnel_Value]

Formatting:

- Colors: Gradient from #93C5FD (light) to #1E40AF (dark)
- Data labels: Value + Percentage
- Conversion rate labels: On
- Sort: By FunnelStages[Order]

Interaction:

- Cross-filters: All visuals on page
- Drill-through: Not enabled

---

2. Waterfall Chart - Session Flow

Visual: Waterfall Chart
Fields:

- Category: WaterfallStages[Stage]
- Y-axis: [Waterfall_Value]

Formatting:

- Increase bars: #10B981 (green)
- Decrease bars: #EF4444 (red)
- Total bar: #F59E0B (gold)
- Connector: On, Gray
- Data labels: On, with +/- signs

Interaction:

- Cross-filters: All visuals
- Sort: By WaterfallStages[Order]

---

3. Scatter Plot - Engagement Quality

Visual: Scatter Chart
Fields:

- X-axis: Duration_Per_Page
- Y-axis: PageValues
- Size: [Total_Sessions]
- Legend: Duration_Category
- Details: Session details

Formatting:

- Colors: Quick=Blue, Normal=Green, Deliberate=Purple
- Transparency: 60%
- Zoom slider: Enabled
- Play axis: Month (optional)

Interaction:

- Cross-filters: Enabled
- Tooltips: Custom (add Conv Rate)

---

4. Matrix - Conversion by Behavior

Visual: Matrix
Fields:

- Rows: Dominant_Page_Type
- Columns: Is_Deep_Session, High_Product_Engagement
- Values: [Conversion_Rate] (%)

Formatting:

- Conditional formatting: Background color
  - Gradient: Red (0%) → Yellow (15%) → Green (30%)
- Font: Bold for values > 20%
- Grid: On, light gray
- Headers: Background #F3F4F6 , Bold

Interaction:

- Cross-filters: All
- Stepped layout: On

---

5. Sankey Diagram - Traffic Flow
Visual: Sankey Diagram (Custom from AppSource)
Fields:

- Source: Traffic_Quality_Tier
- Destination: Dominant_Page_Type
- Destination 2: Revenue (if supported)
- Weight: [Total_Sessions]

Formatting:

- Node colors: Categorical
- Link opacity: 40%
- Show labels: On nodes
- Link color: Source node color
- Highlight on hover: On

Interaction:

- Cross-filters: All pages
- Tooltips: Custom

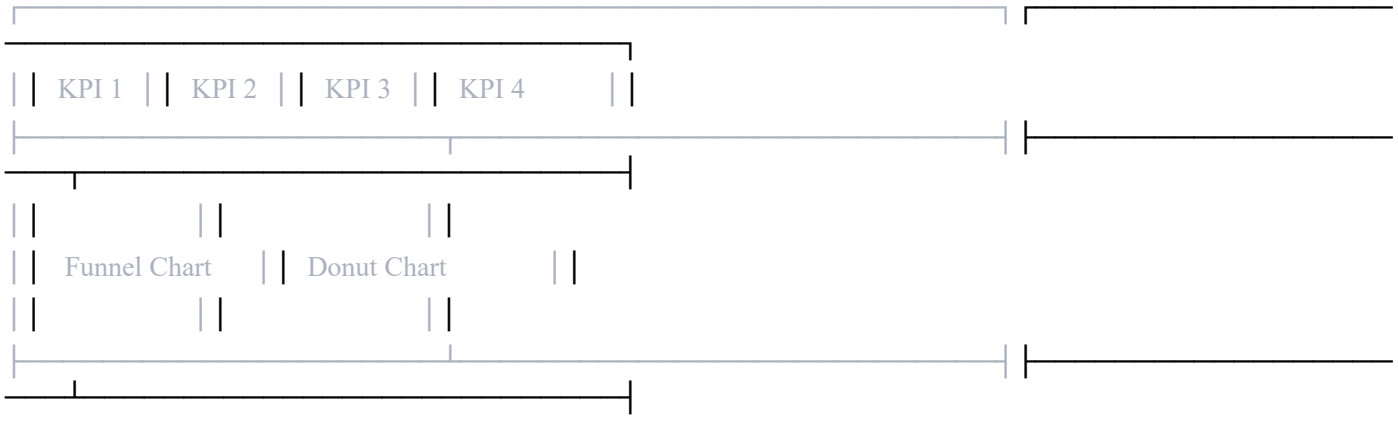
---

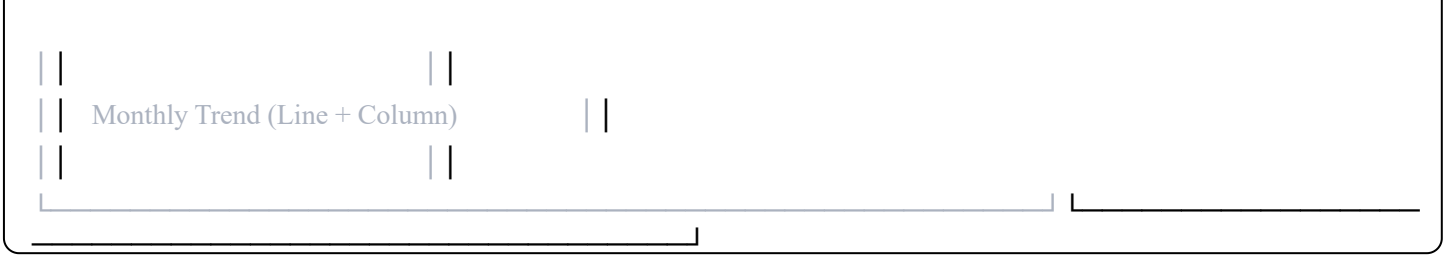
9. Dashboard Pages

Page 1: Executive Overview

Purpose: High-level KPIs and overall performance
Target Audience: C-Level executives, stakeholders
Key Visuals: 4 KPIs, Funnel, Donut, Monthly Trend
Key Insights: Overall conversion rate, funnel drop-offs, seasonality

Layout:





Filters: Month (slicer), VisitorType (slicer)

Page 2: Customer Journey & Engagement

Purpose: Deep dive into user behavior patterns
Target Audience: Marketing, UX teams
Key Visuals: Waterfall, Stacked Bar, Scatter, Matrix
Key Insights: Session flow, complexity impact, engagement sweet spot

Layout:

Filters: Synced from Page 1, plus Duration_Category

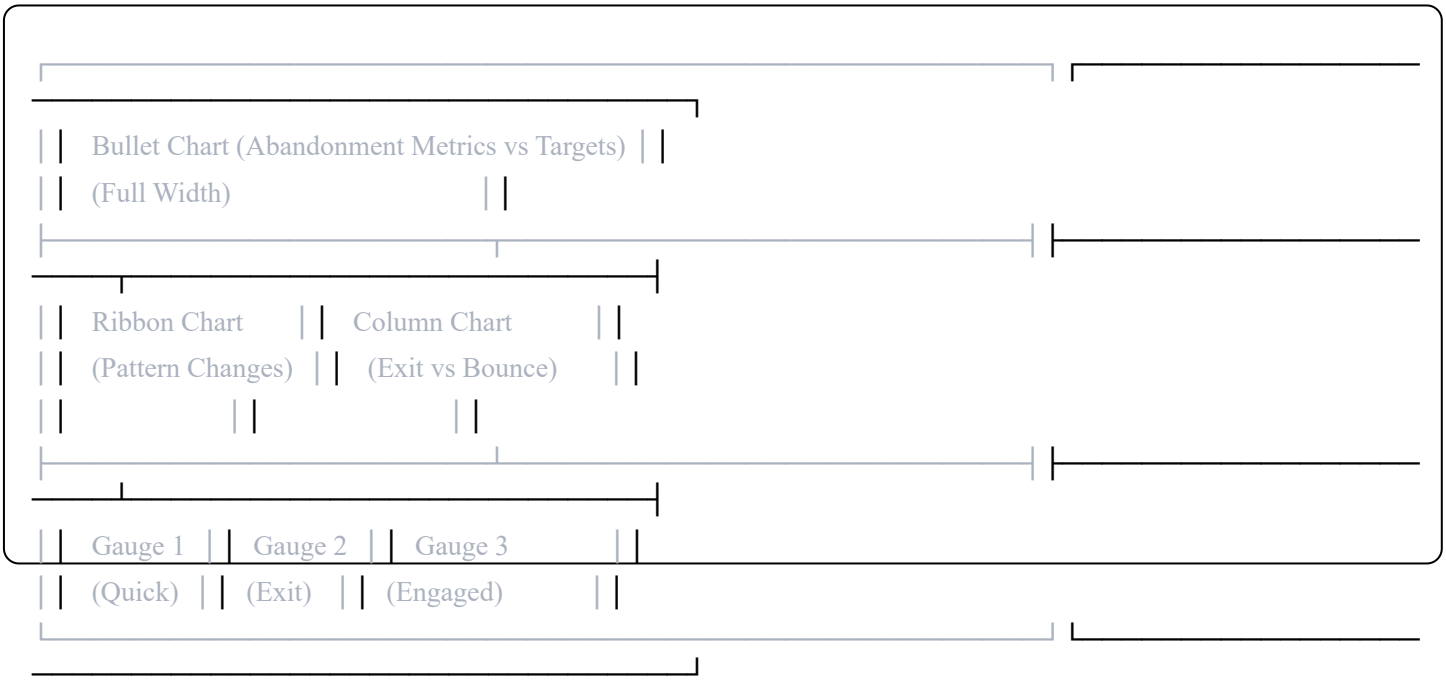
Page 3: Abandonment & Exit Analysis

Purpose: Identify friction points and abandonment patterns
Target Audience: Product managers, conversion optimization team

Key Visuals: Bullet Chart, Ribbon Chart, Column Chart, Gauges

Key Insights: Exit rates by page type, abandonment trends

Layout:



Filters: Abandonment_Type, Page Type

Page 4: Visitor Segmentation

Purpose: Understand different customer segments

Target Audience: Marketing, customer success

Key Visuals: Treemap, Stacked Bar, Decomposition Tree **Key**

Insights: High-value segments, returning vs new performance

Layout:

Filters: VisitorType, Shopping_Period, Special_Day_Category

Page 5: Traffic & Technology

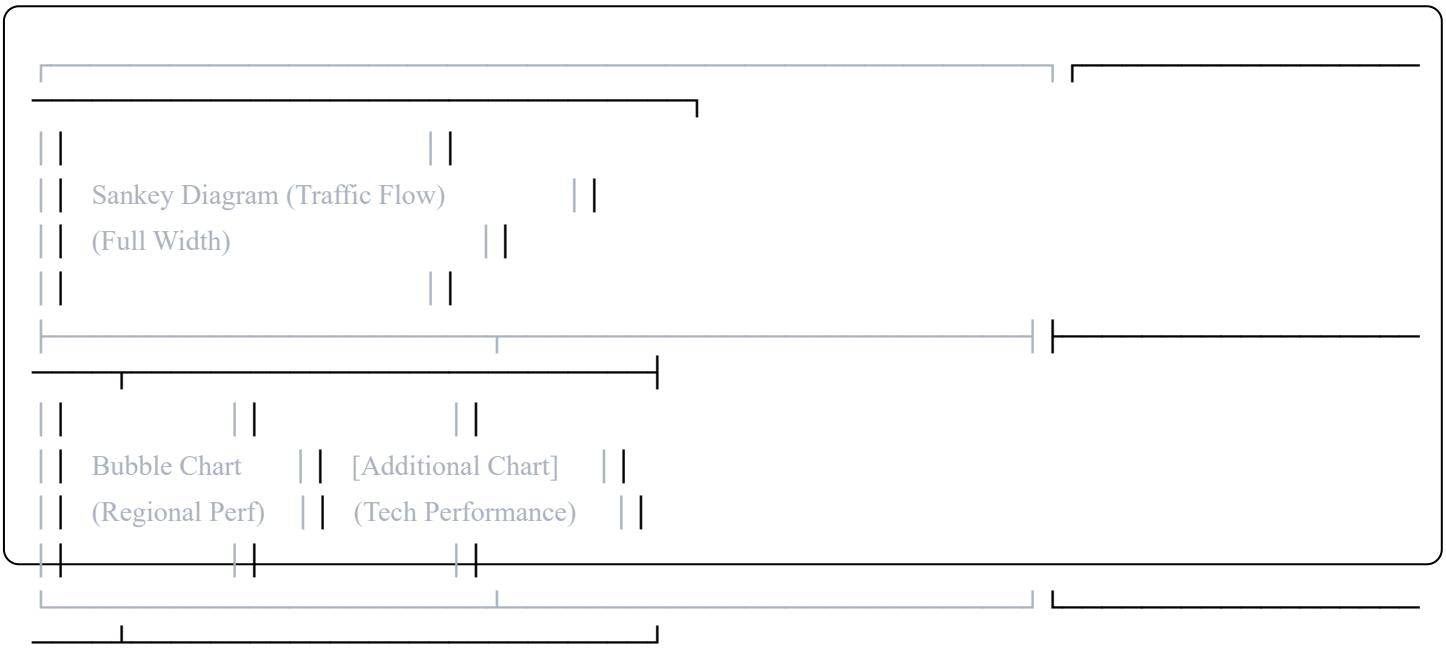
Purpose: Analyze traffic sources and regional performance

Target Audience: Digital marketing, growth team

Key Visuals: Sankey Diagram, Bubble Chart **Key**

Insights: Traffic quality, regional distribution

Layout:



Filters: Region, Traffic_Quality_Tier, Region_Activity

---

10. Performance Optimization

A. Data Model Optimization

1. Removed Unnecessary Columns:

- Kept all original 18 columns for analysis completeness
- Marked hidden: MonthNumber (used only for sorting)

2. Data Types:

- Ensured correct data types for all columns
- Text columns: Categorical data
- Numeric columns: Integer or Decimal as appropriate
- Boolean columns: True/False
- Date columns: Date format

3. Relationships:

- Single table model (no relationships needed)
- Calculated columns and measures reference same table
- No circular dependencies

B. DAX Optimization

1. Variables:

- Used VAR extensively to avoid recalculation
- Stored intermediate results in variables

2. Context Transition:

- Used CALCULATE appropriately
- Avoided unnecessary context transitions

3. Filter Context:

- Used ALL, ALLEXCEPT judiciously
- Minimized row-by-row iterations

Example Optimized Measure:

```dax
Conversion_Rate =
VAR TotalRows = COUNTROWS('online_shoppers_intention')
VAR ConvertedRows =
    CALCULATE(
        COUNTROWS('online_shoppers_intention'),
        'online_shoppers_intention'[Revenue] = TRUE()
    )
RETURN
    DIVIDE(ConvertedRows, TotalRows, 0)
```

C. Visual Performance

1. Limit Data Points: • Aggregated at appropriate level • Used sampling for scatter plots if needed

2. Reduce Visual Complexity:

- Limited visuals per page to 4-5
- Avoided overly complex custom visuals

3. Query Optimization:

- Used DirectQuery for large datasets (not needed here)
- Import mode sufficient for 12,330 rows

D. Filter Optimization

1. Slicer Optimization:

- Limited to 2-3 slicers per page
- Used dropdown for many values
- Used buttons/tiles for few values

2. Cross-filtering:

- Enabled on key visuals
- Disabled where not needed

---

11. Deployment & Maintenance

A. Publishing to Power BI Service

1. Preparation:

- Review all visuals for correctness
- Test all interactions and filters
- Verify DAX measures calculate correctly
- Check mobile layout

2. Publishing Steps:

1. File → Publish → Publish to Power BI
2. Select workspace
3. Wait for upload completion
4. Navigate to Power BI Service
5. Configure refresh schedule (if needed)

3. Sharing:

- Share with specific users
- Create app for broader distribution
- Set permissions (View/Edit)
- Enable comments for collaboration

B. Scheduled RefreshFor

if live data sources:

1. Power BI Service → Dataset settings
2. Configure data source credentials
3. Set refresh schedule:
   - Frequency: Daily/Weekly
   - Time: Off-peak hours
   - Time zone: Local
4. Email notifications: On for failures

C. Maintenance

Schedule Weekly:

- Check for data refresh errors
- Review usage analytics
- Monitor performance metrics

Monthly:

- Review and update DAX measures if needed
- Add new features based on feedback
- Optimize slow-performing visuals

Quarterly:

- Major feature additions
- Dashboard redesign if needed
- User training sessions

D. Version Control

Best Practices:

```
1. Save .pbix file with version numbers
   - dashboard_v1.0.pbix
   - dashboard_v1.1.pbix (minor changes)
   - dashboard_v2.0.pbix (major changes)

2. Use Git for source control (optional)
   - Store .pbix files
   - Track changes in README
   - Tag releases

3. Document changes
   - Maintain changelog
   - Note breaking changes
   - Update documentation
```

E. Backup Strategy

```
1. Local backups:
   - Save .pbix file daily
   - Keep last 7 versions

2. Cloud backups:
   - OneDrive/SharePoint
   - Automatic sync

3. Export data:
   - Export to CSV monthly
   - Archive historical data
```

---

12. Troubleshooting Guide

Common Issues & Solutions Issue 1: Visual Not Updating

Symptom: Visual shows old data
Solution:

- Refresh data: Home → Refresh
- Clear cache: File → Options → Clear cache
- Recreate visual if persists

Issue 2: DAX Measure Returns Error

Symptom: Error in measure calculation
Solution:

- Check column references (table name correct?)
- Verify data types
- Use variables to debug step-by-step
- Check for division by zero

Issue 3: Slow Performance

Symptom: Dashboard loads slowly
Solution:

- Reduce number of visuals per page
- Optimize DAX (use variables)
- Remove unused columns
- Check data model relationships

Issue 4: Filters Not Working

Symptom: Slicer doesn't filter visuals
Solution:

- Check "Edit interactions"
- Verify field relationships
- Ensure correct table references

---

13. Future Enhancements

Planned Features Phase 2: Advanced Analytics

- Predictive modeling (Python/R integration)
- Customer lifetime value calculation
- Churn prediction
- A/B test result visualization

Phase 3: Real-time Data

- Direct Query to live database
- Streaming datasets
- Real-time alerts
- Automated email reports

Phase 4: AI Integration

- Q&A visual
- Key influencers analysis
- Anomaly detection
- Smart narratives

14. CONCLUSION

This project successfully developed a comprehensive Power BI dashboard analyzing 12,330 e-commerce sessions to uncover actionable insights into customer purchasing behavior. Through systematic feature engineering and multi-dimensional visualization, the dashboard transforms raw browsing data into strategic intelligence.

Project Achievements:

Technical Accomplishments:

- Engineered 20+ derived features capturing engagement, navigation, temporal, and behavioral pa0erns
- Implemented robust data transformation pipeline using Power Query and DAX
- Created multi-page, interactive dashboard with intuitive visualizations
- Established KPI framework aligned with business objectives
- Enabled self-service analytics for diverse stakeholder needs

Analytical Accomplishments:

- IdenBfied engagement depth as primary conversion driver
- Quantified visitor type performance gap
- Mapped seasonal pa0erns and special day effects on purchasing behavior
- Evaluated traffic source quality revealing performance variation
- Discovered optimal browsing pa0erns (product-focused, multi-section exploration)
- Pinpointed abandonment pa0erns and friction points in customer journey

Business Value Delivered:

- Provided actionable recommendations with estimated revenue impact
- Enabled data-driven decision-making across marketing, UX, and operations
- Created framework for continuous performance monitoring
- IdenBfied "golden segments" for targeted growth strategies
- Established prioritized roadmap for conversion optimization

15. Appendix

A. Complete Measure Reference

[See Section 5 for all DAX measures]

B. Power Query Scripts

[See Section 6 for all M code]

C. Color Palette

Primary Colors:

- Blue: #2563EB
- Purple: #7C3AED
- Green: #10B981
- Red: #EF4444
- Orange: #F59E0B
- Yellow: #FCD34D

Neutral Colors:

- Dark: #1F2937
- Medium: #6B7280
- Light: #F3F4F6
- Background: #F9FAFB

D. Font Standards

- Title: Segoe UI, 16pt, Bold
- Subtitle: Segoe UI, 14pt, Semibold
- Body: Segoe UI, 11pt, Regular
- Small: Segoe UI, 10pt, Regular

E. Contact & Support Dashboard Owner: [Your Name]
Email: [Your Email]
Last Updated: December 2024
Version: 1.0 **Power BI Desktop Version:** [Your version]

Document Version History

Version	Date	Changes	Author
1.0	Dec 2024	Initial documentation	Garima Daryani

END OF TECHNICAL DOCUMENTATION