رواد مصر الرقمية

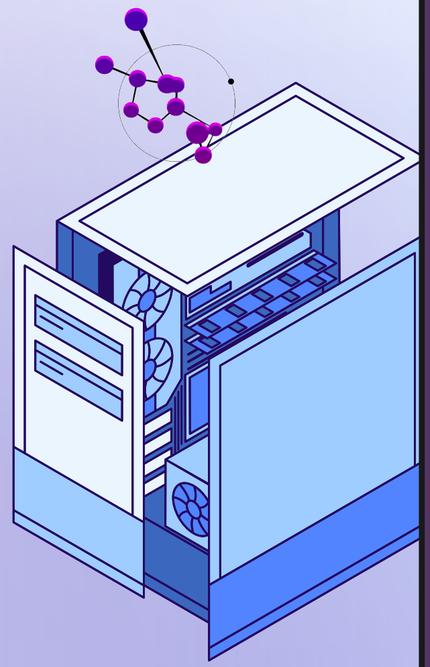
وزارة الاتصالات وتكنولوجيا المعلومات
Ministry of Communications and Information Technology

MOBILE USERS BEHAVIOR DATA MANAGEMENT AND ANALYSIS

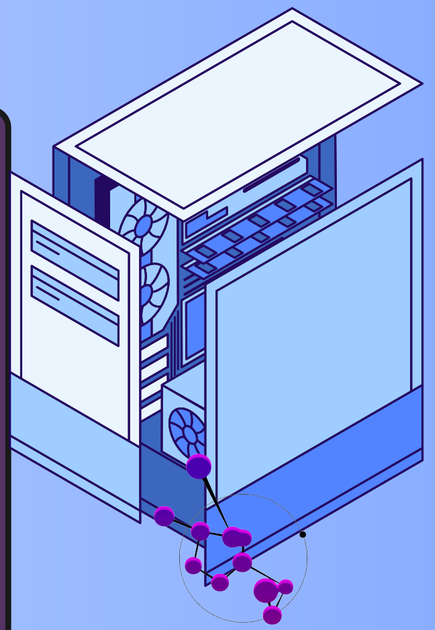
BY SOAD ATEF AND HAGER AHMED

www.reallygreatsite.com

INTRODUCTION



- **What is the project about?**
- The project revolves around managing, cleaning, and analyzing mobile users' behavioral data. It combines SQL, SSIS (SQL Server Integration Services), and Python to handle data efficiently.
- **What tools were used?**
- SQL for database setup and management.
- SSIS for data extraction, transformation, and loading (ETL).
- Python for data analysis and model building.



GOAL AND IMPORTANCE

- **Goal:** The goal of this project is to collect, clean, analyze, and store mobile user behavior data to gain insights that help improve services, marketing strategies, and user satisfaction.
- **Importance:**
  - Informed decision-making based on data.
  - Insights into user trends, preferences, and patterns.
  - Support for product improvement and customer satisfaction strategies.
  - Data-driven decisions are critical for competitive business strategies.
  - The ability to handle large datasets is essential for mobile user behavior analysis.
  - Combining SQL, SSIS, and Python provides an efficient solution for comprehensive data management and analysis

رواد مصر الرقمية

KEY CONCEPTS

- SQL Database to manage user data efficiently.
- SSIS for loading and transforming large datasets.
- Python for further data cleaning and advanced analysis.

DATA OVERVIEW

- **Dataset:** Mobile Users Behavior Dataset
- S**ource:** The dataset was sourced from user interactions across various mobile platforms.
- **Number of Records:** Over 700 user behavior records.
- **Main Features:**
- User ID
- Device Model
- App Usage
- Operating_System
- Device ID
- App_Usage_Time
- Screen_On_Time
- Battery_Drain
- Number_of_Apps_Installed
- Data_Usage
- Age
- Gender
- User_Behavior_Class
- OS_ID

رواد مصر الرقمية

DATA CLEANING

Handled Missing Data:

- Identified and managed missing values in columns such as app usage and interaction type. Methods like mean imputation and removal of records were used.

Removed Duplicates:

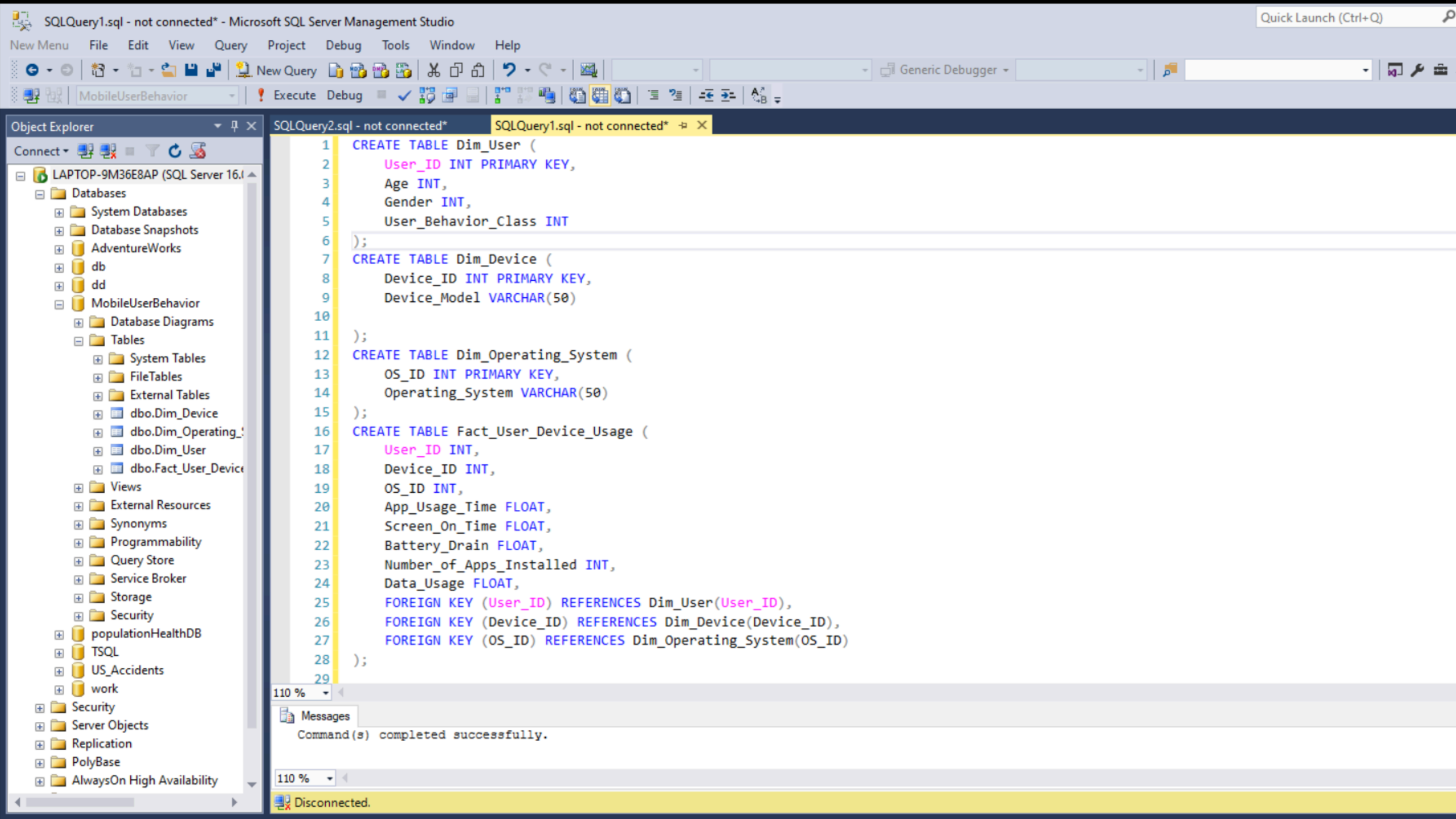
- Ensured data integrity by identifying and removing duplicate records.

```python
import pandas as pd

data = pd.read_csv("user_behavior.csv")
data.columns = data.columns.str.replace(' ', '_')

DeviceModel = data["Device_Model"]
OS = data["Operating_System"]
Gender = data["Gender"]
DeviceModelNumeric = {
    "Google Pixel 5": 1,
    "OnePlus 9": 2,
    "iPhone 12": 3,
    "Xiaomi Mi 11": 4,
    "Samsung Galaxy S21": 5,
}
OSNumeric = {
    "Android": 0,
    "iOS": 1,
}
GenderNumeric = {
    "Male": 0,
    "Female": 1,
}



data["Device_ID"] = data["Device_Model"].replace(DeviceModelNumeric)
data["OS_ID"] = data["Operating_System"].replace(OSNumeric)
data["Gender"] = data["Gender"].replace(GenderNumeric)

output_path = "analuser_behavior_dataset.csv"
data.to_csv(output_path, index=False)
```

```python
import pandas as pd

# Assuming you have a DataFrame df with 'User_ID' and 'OS_ID'
df = pd.read_csv("analuser_behavior_dataset.csv")

# Create a new composite key column by concatenating 'User_ID' and 'OS_ID'
df['Composite_Key1'] = (df['User_ID'].astype(str) + df['OS_ID'].astype(str)).astype(int)
df['Composite_Key2'] = df['User_ID'] * 1000

df = df.drop(columns=['Device_ID', 'OS_ID'])

df = df.rename(columns={'Composite_Key2': 'Device_ID'})
df = df.rename(columns={'Composite_Key1': 'OS_ID'})
# Now 'Composite_Key' is a combination of both columns
print(df)
df.to_excel('analuser_behavior_dataset.xls', index=False, engine='openpyxl')
```

Console 1/A

```
In [1]:
```


SQLQuery1.sql - not connected* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
New Menu File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
MobileUserBehavior
Execute Debug
Object Explorer
Connect
LAPTOP-9M36E8AP (SQL Server 16.
Databases
System Databases
Database Snapshots
AdventureWorks
db
dd
MobileUserBehavior
Database Diagrams
Tables
System Tables
FileTables
External Tables
dbo.Dim_Device
dbo.Dim_Operating_
dbo.Dim_User
dbo.Fact_User_Device
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
populationHealthDB
TSQL
US_Accidents
work
Security
Server Objects
Replication
PolyBase
AlwaysOn High Availability
SQLQuery2.sql - not connected*
SQLQuery1.sql - not connected*
1 CREATE TABLE Dim_User (
2 User_ID INT PRIMARY KEY,
3 Age INT,
4 Gender INT,
5 User_Behavior_Class INT
6);
7 CREATE TABLE Dim_Device (
8 Device_ID INT PRIMARY KEY,
9 Device_Model VARCHAR(50)
10
11);
12 CREATE TABLE Dim_Operating_System (
13 OS_ID INT PRIMARY KEY,
14 Operating_System VARCHAR(50)
15);
16 CREATE TABLE Fact_User_Device_Usage (
17 User_ID INT,
18 Device_ID INT,
19 OS_ID INT,
20 App_Usage_Time FLOAT,
21 Screen_On_Time FLOAT,
22 Battery_Drain FLOAT,
23 Number_of_Apps_Installed INT,
24 Data_Usage FLOAT,
25 FOREIGN KEY (User_ID) REFERENCES Dim_User(User_ID),
26 FOREIGN KEY (Device_ID) REFERENCES Dim_Device(Device_ID),
27 FOREIGN KEY (OS_ID) REFERENCES Dim_Operating_System(OS_ID)
28);
29
110 %
Messages
Command(s) completed successfully.
110 %
Disconnected.

SQLQuery11.sql - not connected* - Microsoft SQL Server Management Studio

Quick Launch (Ctrl+Q)

New Menu File Edit View Query Project Debug Tools Window Help

New Query Generic Debugger

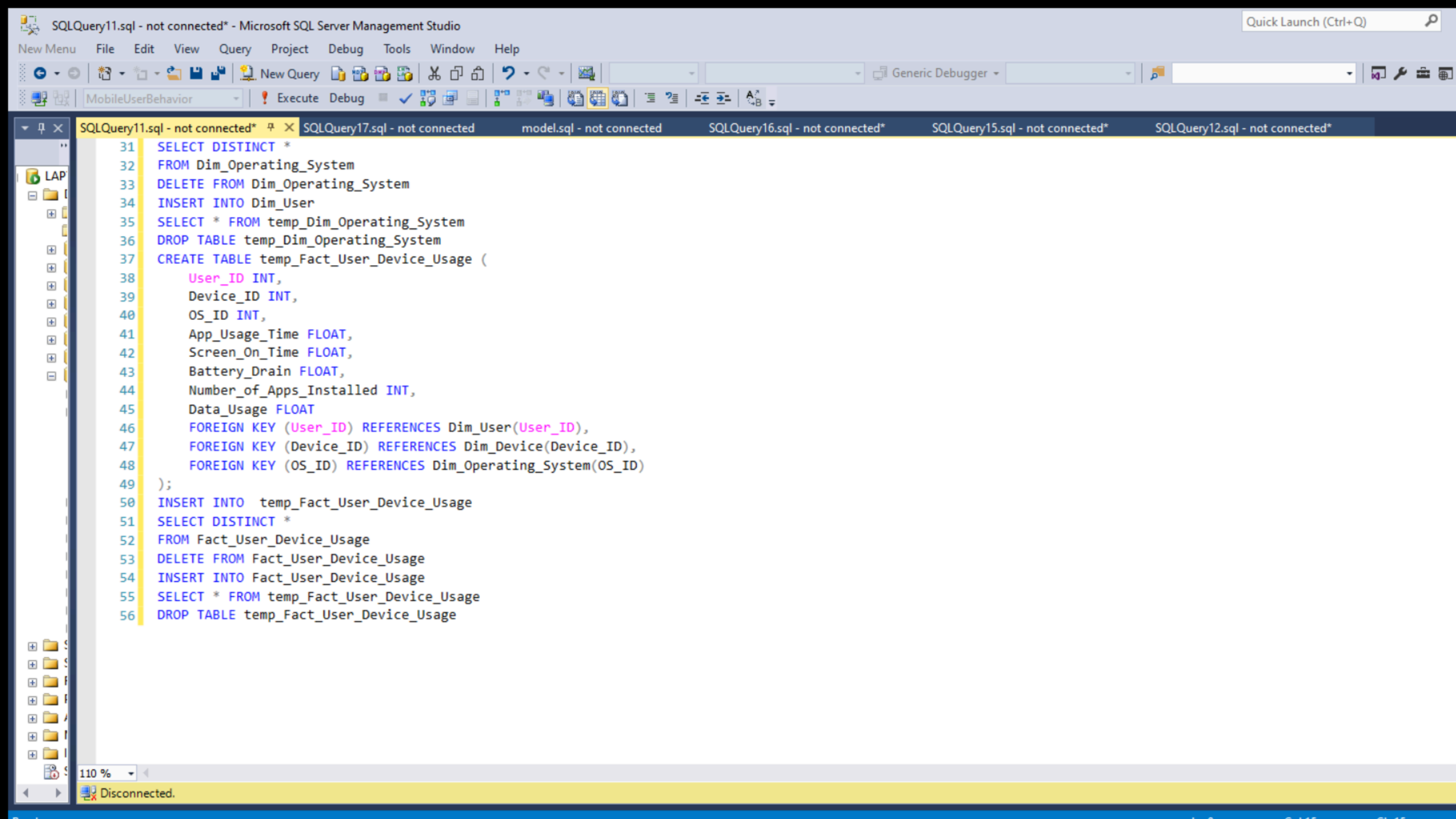
MobileUserBehavior Execute Debug

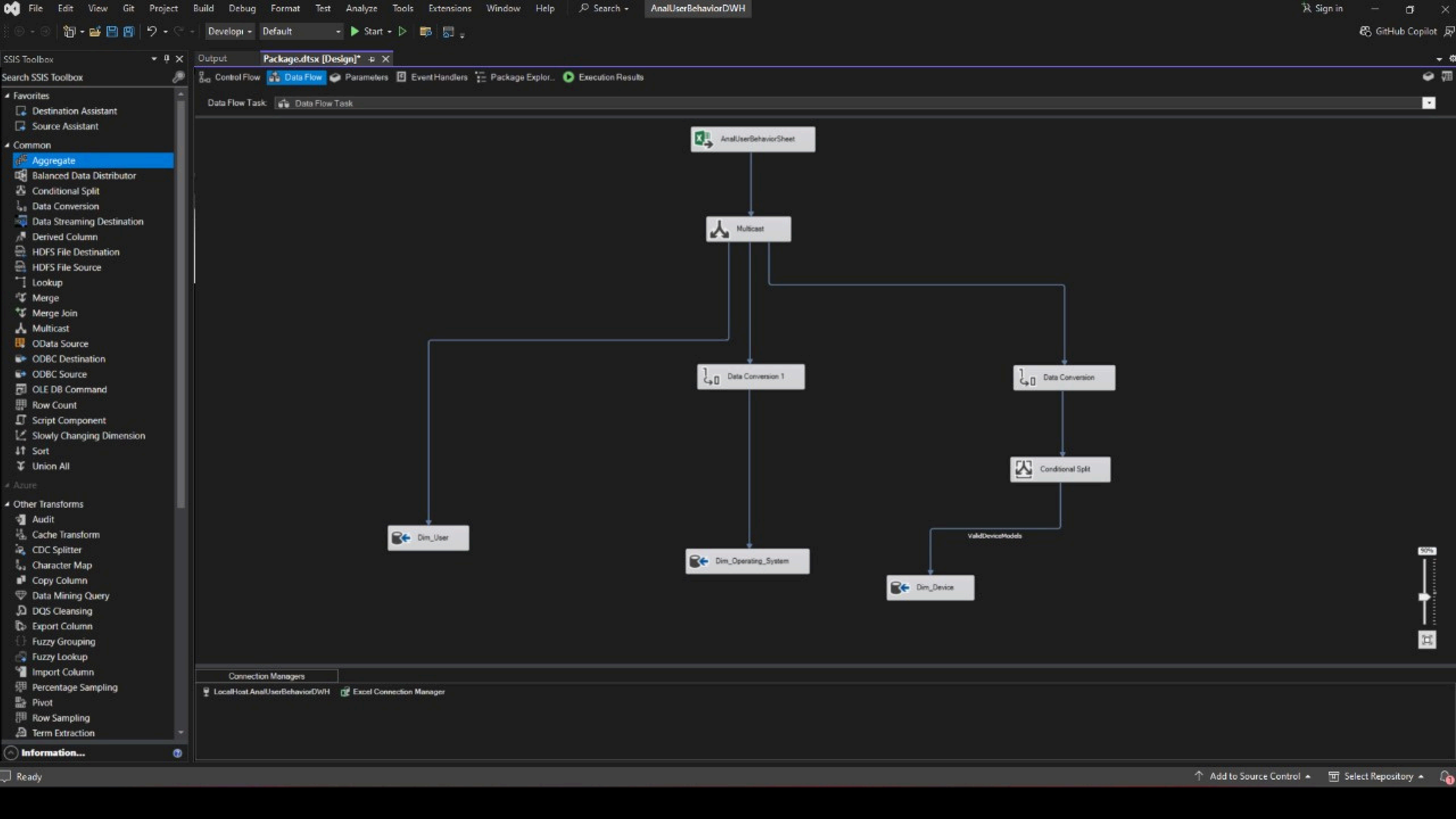
SQLQuery11.sql - not connected* | SQLQuery17.sql - not connected | model.sql - not connected | SQLQuery16.sql - not connected* | SQLQuery15.sql - not connected* | SQLQuery12.sql - not connected*

```
CREATE TABLE temp_Dim_User (
    User_ID INT PRIMARY KEY,
    Age INT,
    Gender INT,
    User_Behavior_Class INT
);
INSERT INTO  temp_Dim_User
SELECT DISTINCT *
FROM Dim_User
DELETE FROM Dim_User
INSERT INTO Dim_User
SELECT * FROM temp_Dim_User
DROP TABLE temp_Dim_User
CREATE TABLE temp_Dim_Device (
    Device_ID INT PRIMARY KEY,
    Device_Model VARCHAR(50),
    Brand VARCHAR(50)
);
INSERT INTO  temp_Dim_Device
SELECT DISTINCT *
FROM Dim_Device
DELETE FROM Dim_Device
INSERT INTO Dim_Device
SELECT * FROM temp_Dim_Device
DROP TABLE temp_Dim_Device
CREATE TABLE temp_Dim_Operating_System (
    OS_ID INT PRIMARY KEY,
    Operating_System VARCHAR(50)
);
INSERT INTO  temp_Dim_Operating_System
SELECT DISTINCT *
FROM Dim_Operating_System
DELETE FROM Dim_Operating_System
```

110 %

Disconnected.


SQLQuery11.sql - not connected* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
New Menu File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
MobileUserBehavior
Execute Debug
SQLQuery11.sql - not connected*
SQLQuery17.sql - not connected
model.sql - not connected
SQLQuery16.sql - not connected*
SQLQuery15.sql - not connected*
SQLQuery12.sql - not connected*
31 SELECT DISTINCT *
32 FROM Dim_Operating_System
33 DELETE FROM Dim_Operating_System
34 INSERT INTO Dim_User
35 SELECT * FROM temp_Dim_Operating_System
36 DROP TABLE temp_Dim_Operating_System
37 CREATE TABLE temp_Fact_User_Device_Usage (
38 User_ID INT,
39 Device_ID INT,
40 OS_ID INT,
41 App_Usage_Time FLOAT,
42 Screen_On_Time FLOAT,
43 Battery_Drain FLOAT,
44 Number_of_Apps_Installed INT,
45 Data_Usage FLOAT
46 FOREIGN KEY (User_ID) REFERENCES Dim_User(User_ID),
47 FOREIGN KEY (Device_ID) REFERENCES Dim_Device(Device_ID),
48 FOREIGN KEY (OS_ID) REFERENCES Dim_Operating_System(OS_ID)
49);
50 INSERT INTO temp_Fact_User_Device_Usage
51 SELECT DISTINCT *
52 FROM Fact_User_Device_Usage
53 DELETE FROM Fact_User_Device_Usage
54 INSERT INTO Fact_User_Device_Usage
55 SELECT * FROM temp_Fact_User_Device_Usage
56 DROP TABLE temp_Fact_User_Device_Usage
110 %
Disconnected.


File Edit View Git Project Build Debug Format Test Analyze Tools Extensions Window Help
Search
AnalUserBehaviorDWH
Sign in
GitHub Copilot
Start
SSIS Toolbox
Search SSIS Toolbox
Favorites
Destination Assistant
Source Assistant
Common
Aggregate
Balanced Data Distributor
Conditional Split
Data Conversion
Data Streaming Destination
Derived Column
HDFS File Destination
HDFS File Source
Lookup
Merge
Merge Join
Multicast
OData Source
ODBC Destination
ODBC Source
OLE DB Command
Row Count
Script Component
Slowly Changing Dimension
Sort
Union All
Azure
Other Transforms
Audit
Cache Transform
CDC Splitter
Character Map
Copy Column
Data Mining Query
DQS Cleansing
Export Column
Fuzzy Grouping
Fuzzy Lookup
Import Column
Percentage Sampling
Pivot
Row Sampling
Term Extraction
Information...
Output
Package.dtsx [Design]*
Control Flow
Data Flow
Parameters
Event Handlers
Package Explor...
Execution Results
Data Flow Task:
Data Flow Task
AnalUserBehaviorSheet
Multicast
Data Conversion 1
Data Conversion
Conditional Split
ValidDeviceModels
Dim_User
Dim_Operating_System
Dim_Device
50%
Connection Managers
LocalHost.AnalUserBehaviorDWH
Excel Connection Manager
Ready
Add to Source Control
Select Repository

FEATURE SELECTION

- **Why Feature Selection?** To remove irrelevant data and focus on features that would significantly impact the analysis.
- **Key Features Selected:**
  - Number of apps installed
  - Battery Drain
  - Data Usage

```python
from sklearn.feature_selection import SelectKBest ,mutual_info_regression,chi2
from sklearn.preprocessing import LabelEncoder
import pandas as pd

file_path = 'cleaned_user_behavior_dataset.csv'
df = pd.read_csv(file_path)

X= df.drop('User Behavior Class' , axis=1)
Y= df['User Behavior Class']

le = LabelEncoder()
for columns in X.columns:
    X[columns]= le.fit_transform(X[columns])

Y = le.fit_transform(Y)

selector = SelectKBest(score_func= mutual_info_regression,  k=3)#score_fun -> method of selection , k -> number of feature
selector.fit_transform(X,Y)
selected_feature = X.columns[selector.get_support()].values#-> select columns with True values
print(X[selected_feature])
```

History Help Variable Explorer Plots Files

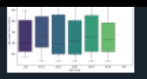
Console 1/A

```
In [7]: runfile('C:/Users/soada/Downloads/untitled23.py', wdir='C:/
Users/soada/Downloads')
     Battery Drain (mAh/day)  Number of Apps Installed  Data Usage (MB/
day)
0                        391                        54
376
1                        275                        30
316
2                        158                        21
108
3                        350                        44
292
4                        280                        46
332
..                       ...                       ...
...
695                      220                        15
131
696                      407                        55
397
697                      190                        11
162
698                       47                         3
58
699                      266                        37
279

[700 rows x 3 columns]
```

MODEL SELECTION AND TRAINING

- **What kind of models?** Machine learning models were developed using Python for predictive analytics**. Models tested included:**
- Logistic Regression for user retention prediction.
- Linear Regression
- Lasso
- Ridge
- **Training Process:** The dataset was split into training and test sets. The models were trained using the training data and validated against the test data.

```python
import pandas as pd
from sklearn.linear_model import LogisticRegression, LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline

data = pd.read_csv("analuser_behavior_dataset.csv")

x = data.drop(['Device_Model', 'Operating_System','User_Behavior_Class'], axis=1)
y = data["User_Behavior_Class"]
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size=0.2, random_state=42)
lr = LinearRegression()
lr.fit(xTrain, yTrain)
yPredict = lr.predict(xTest)
mse = mean_squared_error(yTest, yPredict)
print("Mean Squared Error : ", mse)
print("linear regression accuracy : ", lr.score(xTest, yTest))
# ------------------------------------------------------------------------------ Lasso -----------------
lasso = make_pipeline( PolynomialFeatures(degree=2), Lasso(alpha=0.3))
lasso.fit(xTrain, yTrain)
print("lasso accuracy : ", lasso.score(xTest, yTest))
# ------------------------------------------------------------------------------ Rigid -----------------
rg = Ridge(alpha=0.3)
rg.fit(xTrain, yTrain)
yPredict = rg.predict(xTest)
print("Ridge accuracy : ", rg.score(xTest, yTest))
# ------------------------------------------------------------------------------ Logistic Regression  ---
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

lr = LogisticRegression()
lr.fit(xTrain,yTrain)
Y_predict = lr.predict(xTest)
print('Logistic Regression accuracy : ',accuracy_score(yTest, Y_predict))

```



History | Help | Variable Explorer | Plots | Files

Console 1/A

```
In [6]: runfile('C:/Users/soada/Downloads/untitled19.py', wdir='C:/Users/soada/
Downloads')
Mean Squared Error :  0.0318563757754207
linear regression accuracy :  0.9825297995187956
lasso accuracy :  0.9902959564068873
Ridge accuracy :  0.982529917472532
Logistic Regression accuracy :  0.5785714285714286
C:\Users\soada\anaconda3\Lib\site-packages\sklearn\linear_model\_coordinate_descent.py:
678: ConvergenceWarning: Objective did not converge. You might want to increase the
number of iterations, check the scale of the features or consider increasing
regularisation. Duality gap: 5.801e+00, tolerance: 1.117e-01
  model = cd_fast.enet_coordinate_descent(
C:\Users\soada\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:469:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(

In [7]:
```

رواد مصر الرقمية

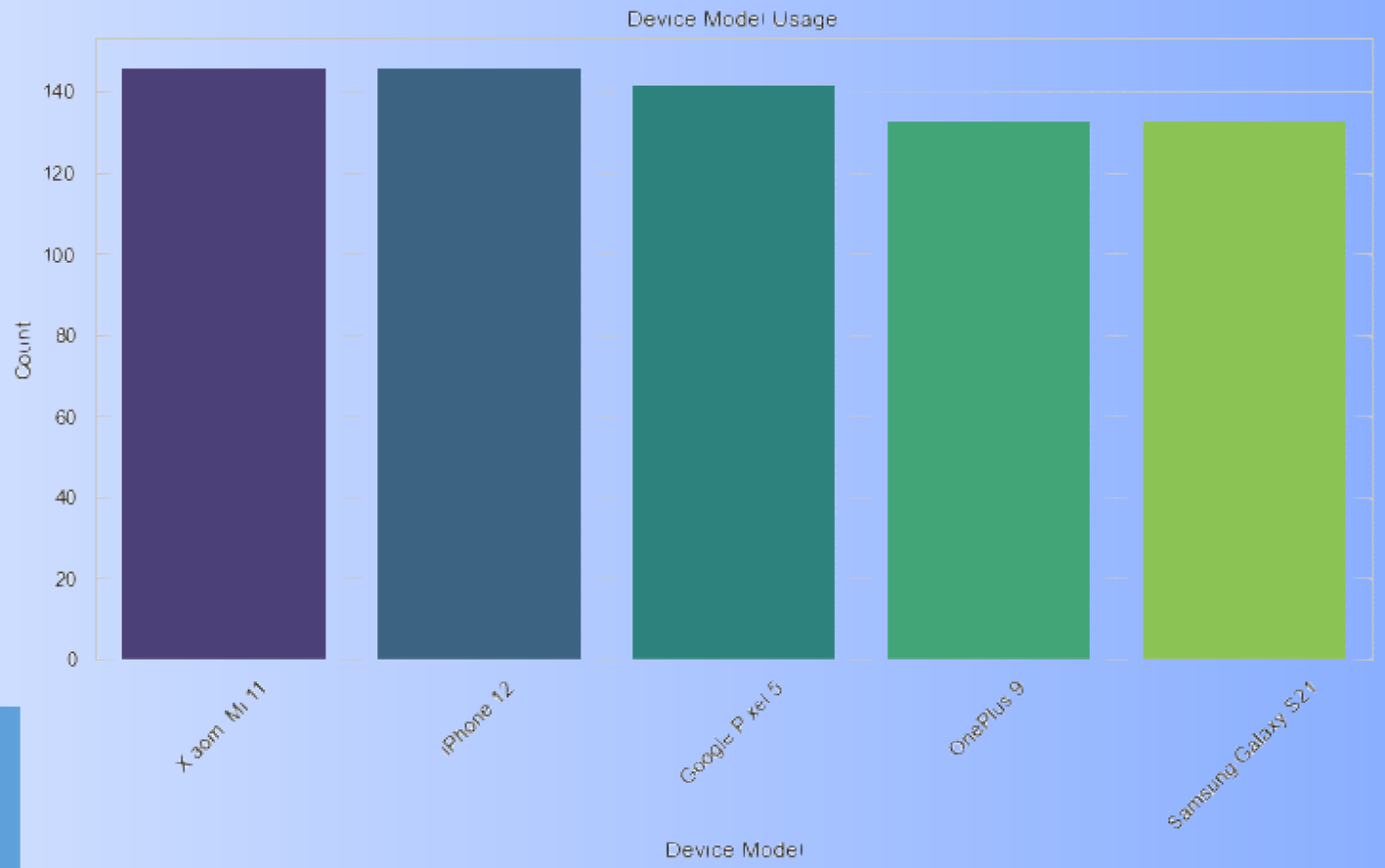
DATA VISUALIZATION

- **Why Visualization?** To gain better insights from the data through graphical representation.

- **Visualizations Created:**
- **Bar Charts**
- **HeatMap**
- **Box Plot**
- **Histogram**
- **Line Graph**

رواد مصر الرقمية

DATA VISUALIZATION



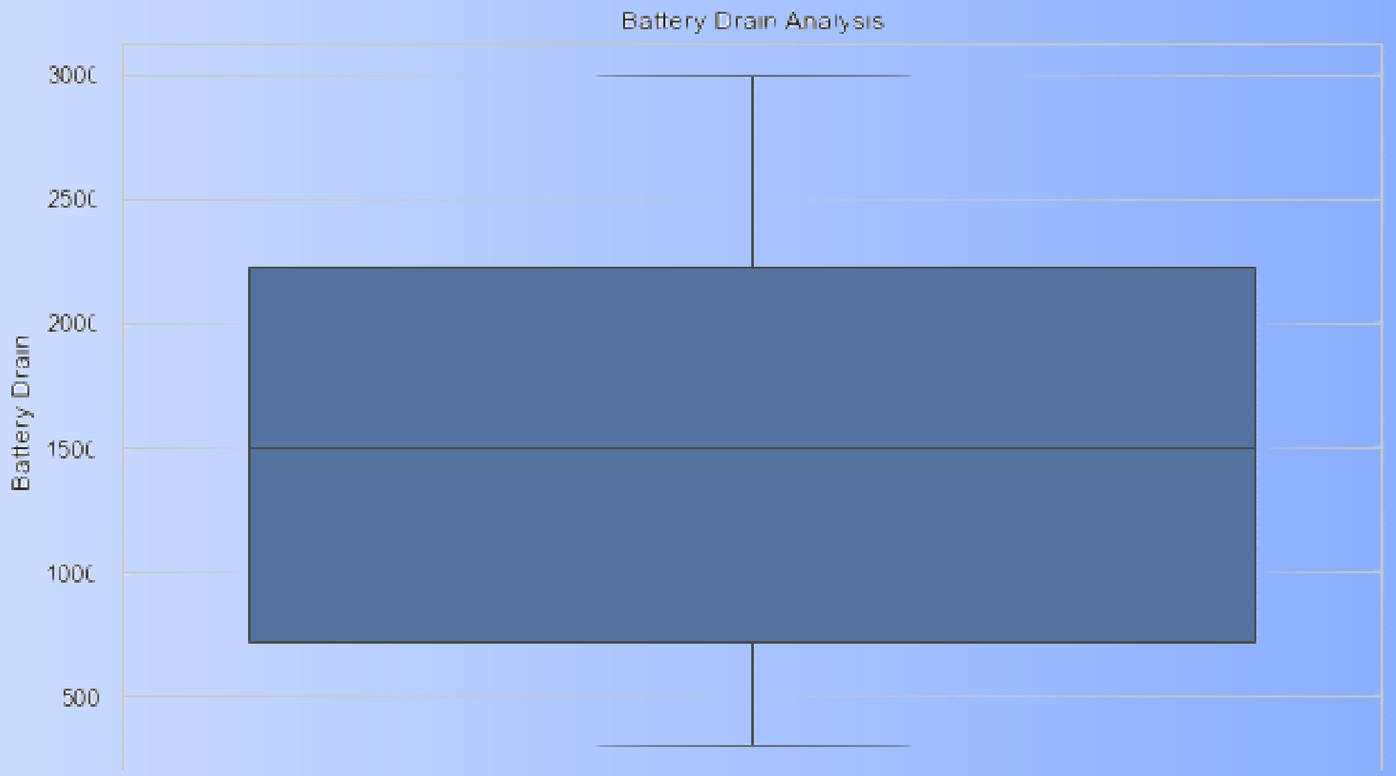
Device Model Usage

Type: Bar Chart

Purpose: Display the number of users for each device model to identify popular devices among users

رواد مصر الرقمية

DATA VISUALIZATION



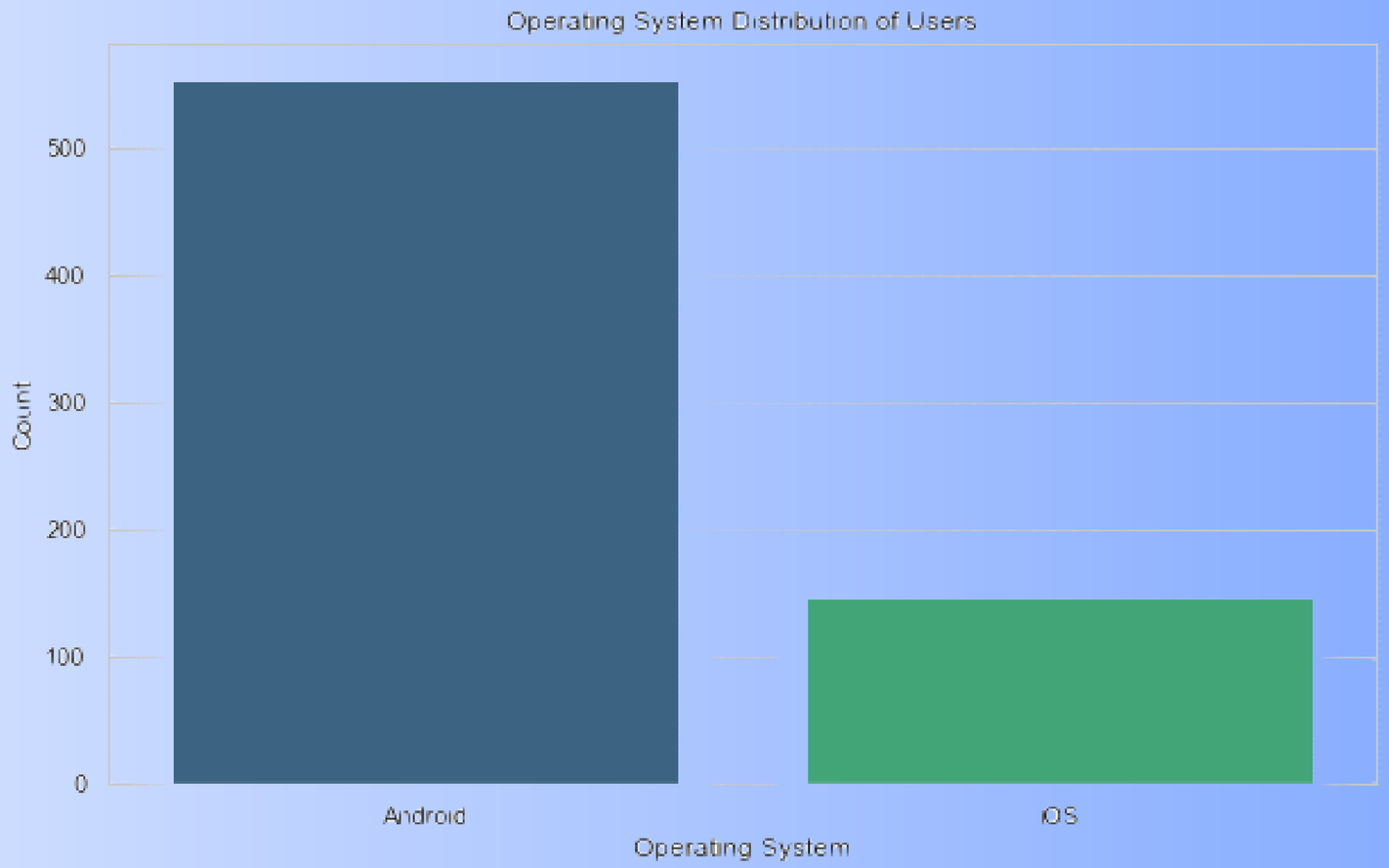
Battery Drain Analysis

Type: Box Plot

Purpose: Show the distribution of battery drain values to identify outliers and the general trend of battery usage among users.

رواد مصر الرقمية

DATA VISUALIZATION



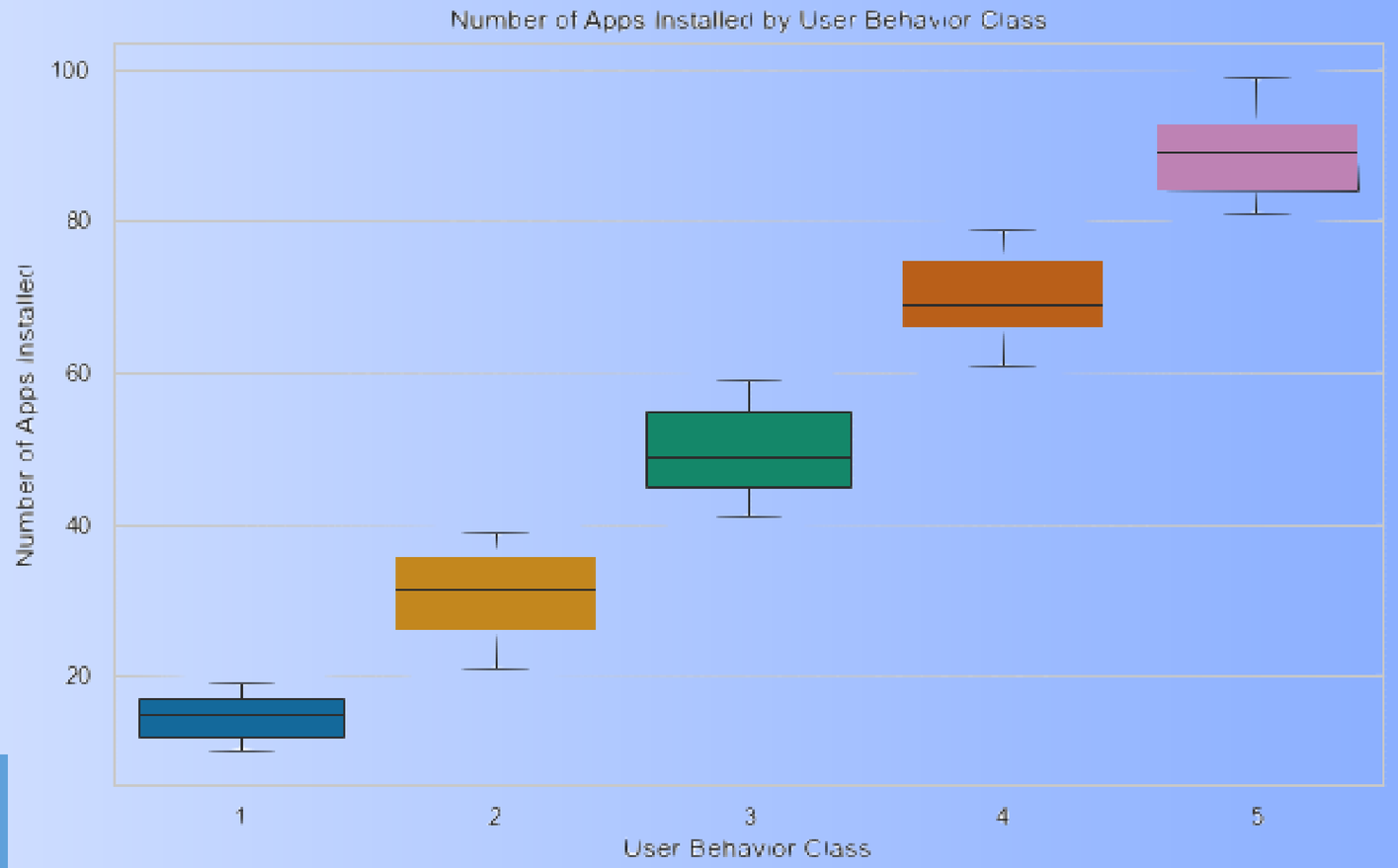
Operating System Distribution

Type: Bar Chart

Purpose: Illustrate the distribution of users across different operating systems (e.g., iOS, Android).

رواد مصر الرقمية

DATA VISUALIZATION



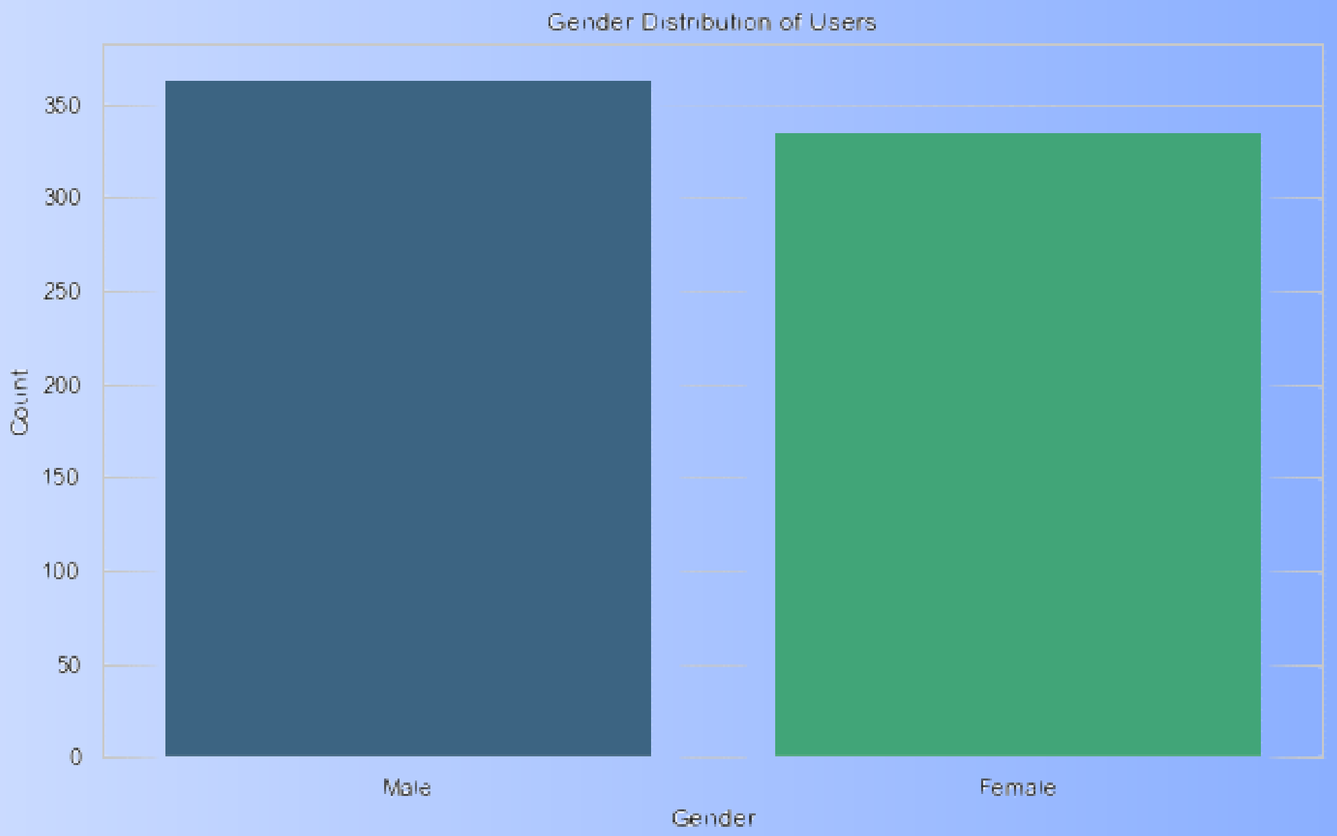
Number of Apps Installed vs. User Behavior Class

Type: Box Plot

Purpose: Compare the number of apps installed across different user behavior classes to identify any trends.

رواد مصر الرقمية

DATA VISUALIZATION



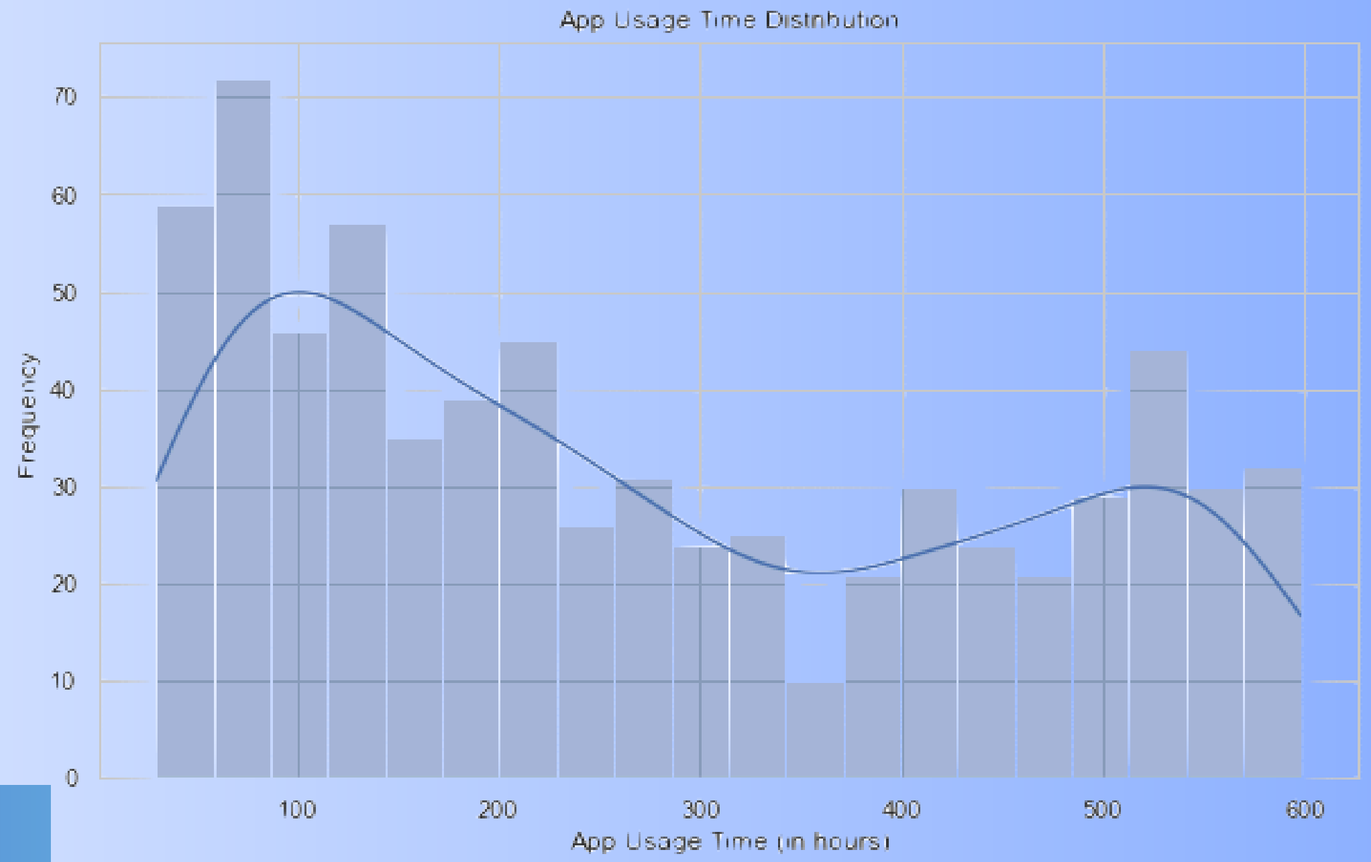
Gender Distribution

Type: Bar Chart

Purpose: Visualize the proportion of male and female users to assess gender representation in the dataset.

رواد مصر الرقمية

DATA VISUALIZATION



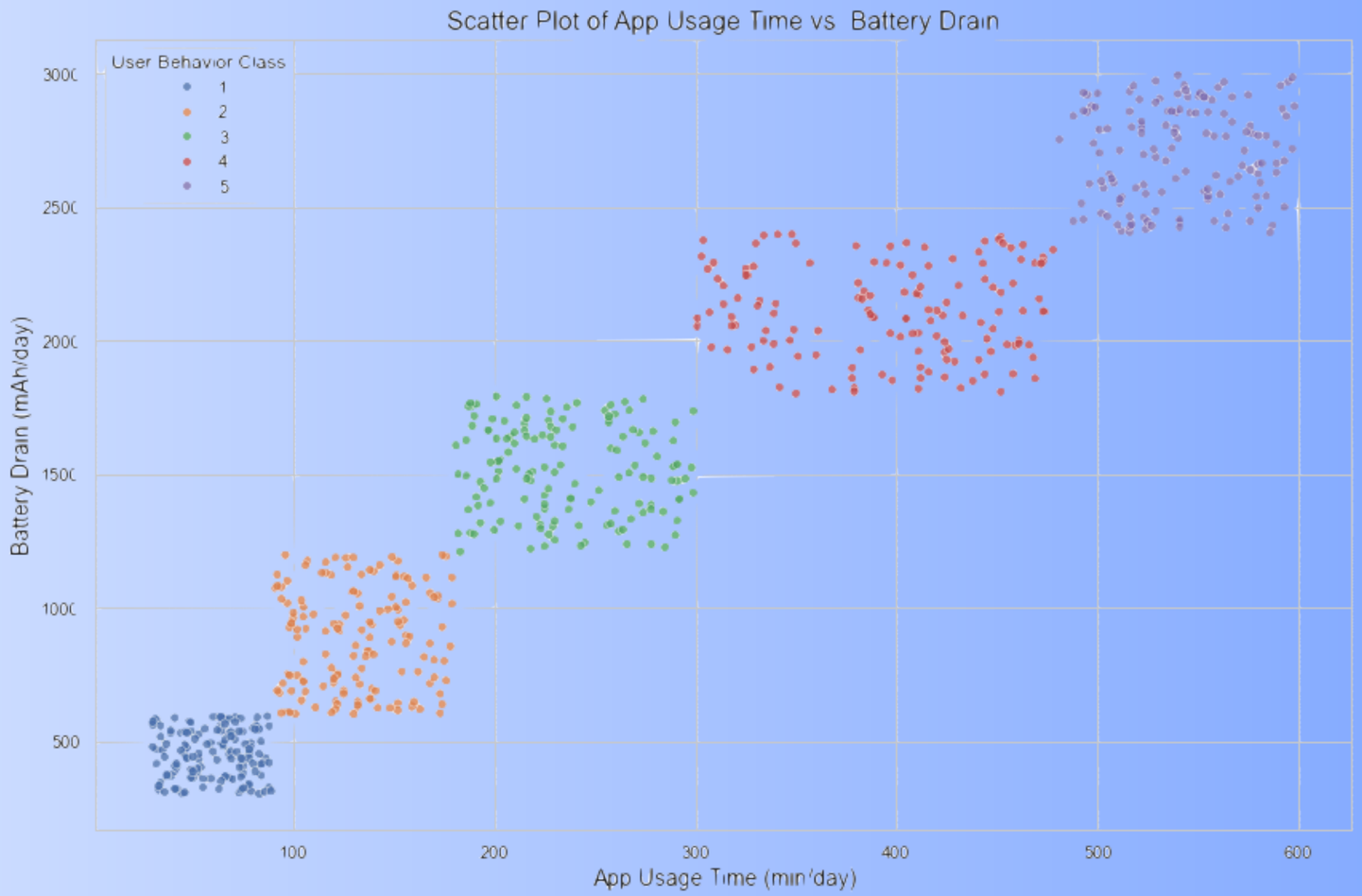
App Usage Time distribution

Type: Histogram

Purpose: Analyze the distribution of app usage time to see how much time users spend on their devices.

رواد مصر الرقمية

DATA VISUALIZATION



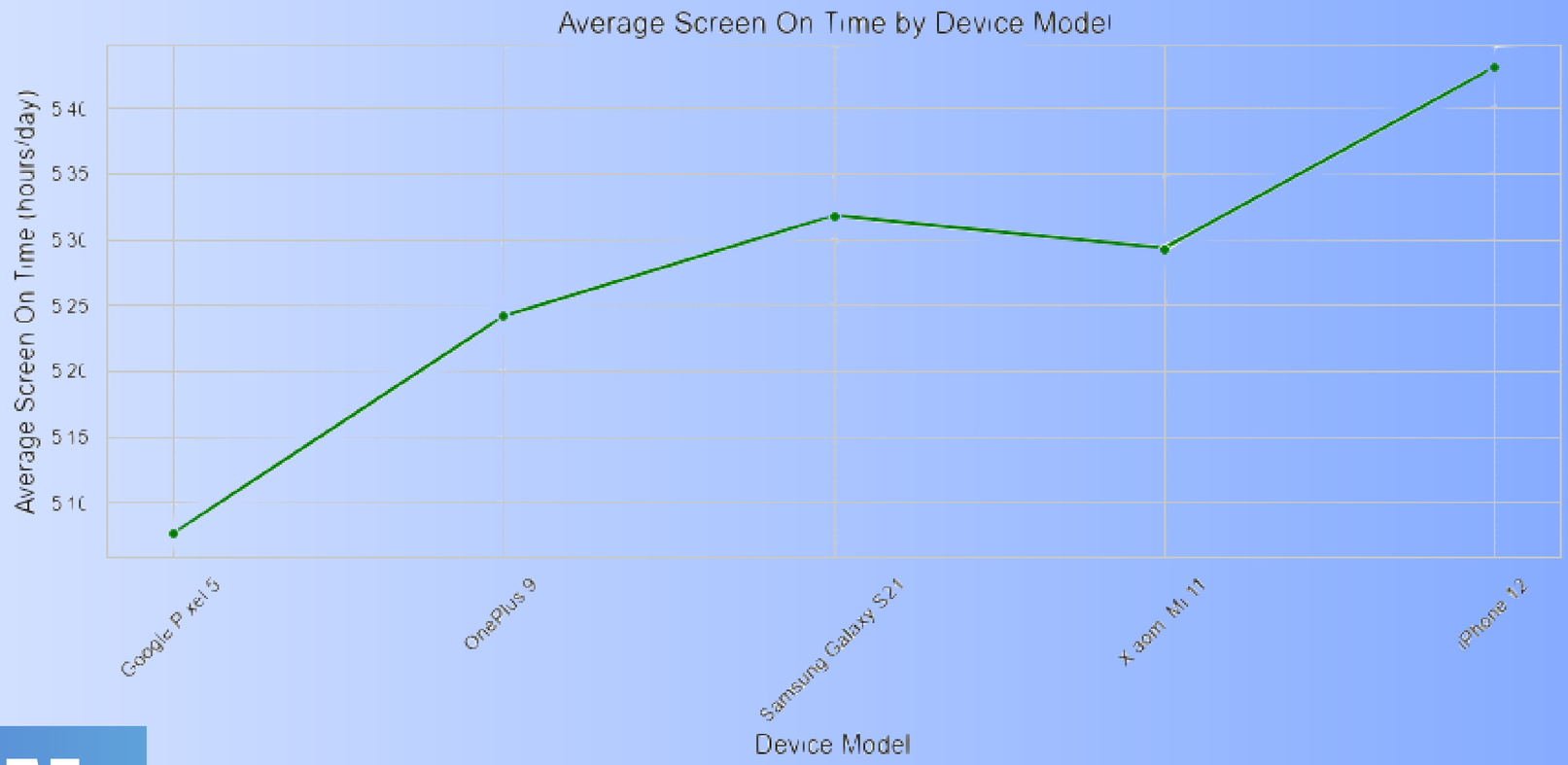
App Usage Time VS Battery Drain

Type: Scatter Plot

Purpose: Analyze the distribution of app usage time to see how much battery is drained

رواد مصر الرقمية

DATA VISUALIZATION



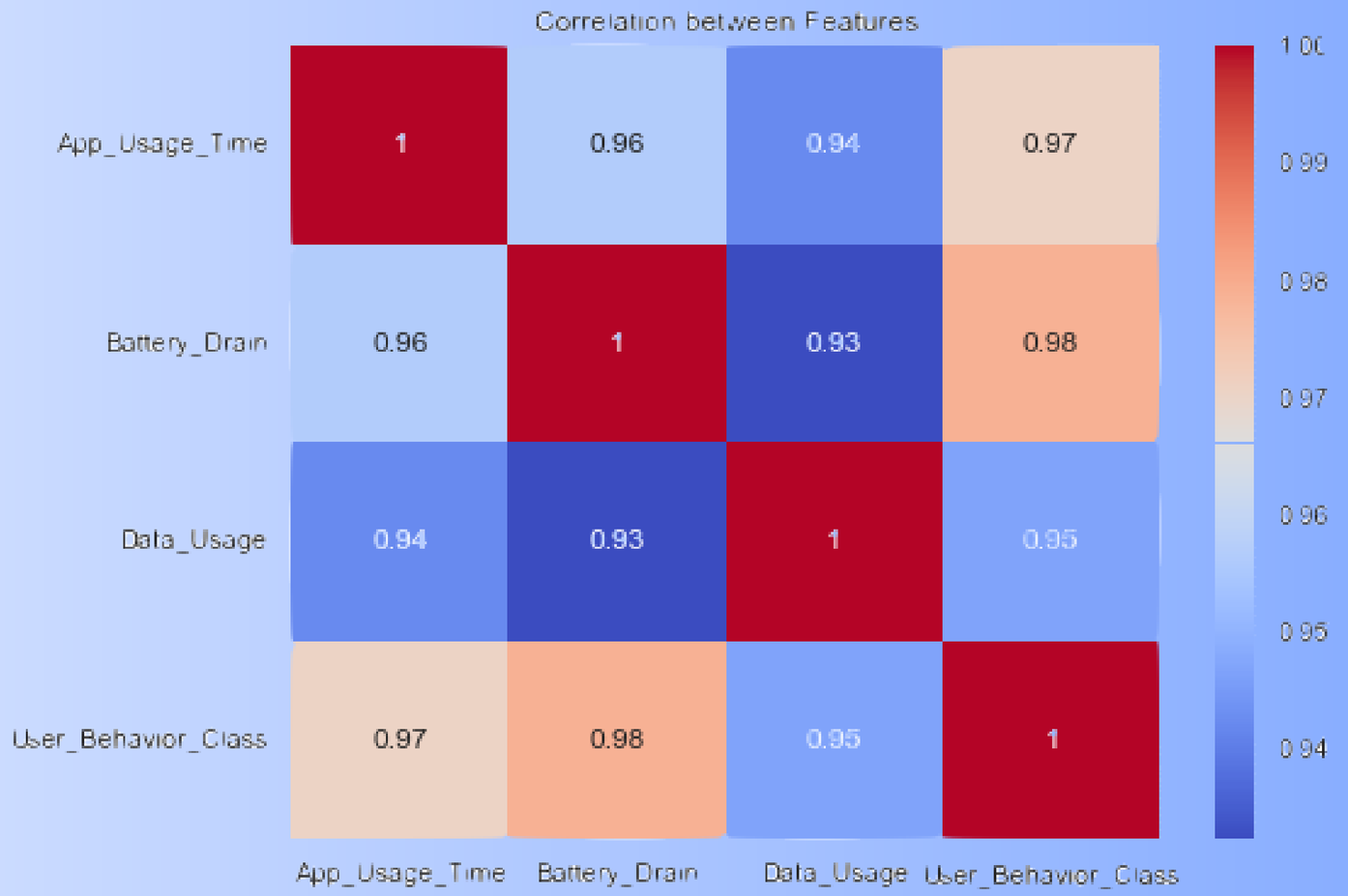
Average Screen On Time by Device Model

Type: Line Graph

Purpose: Analyze the average screen time

رواد مصر الرقمية

DATA VISUALIZATION

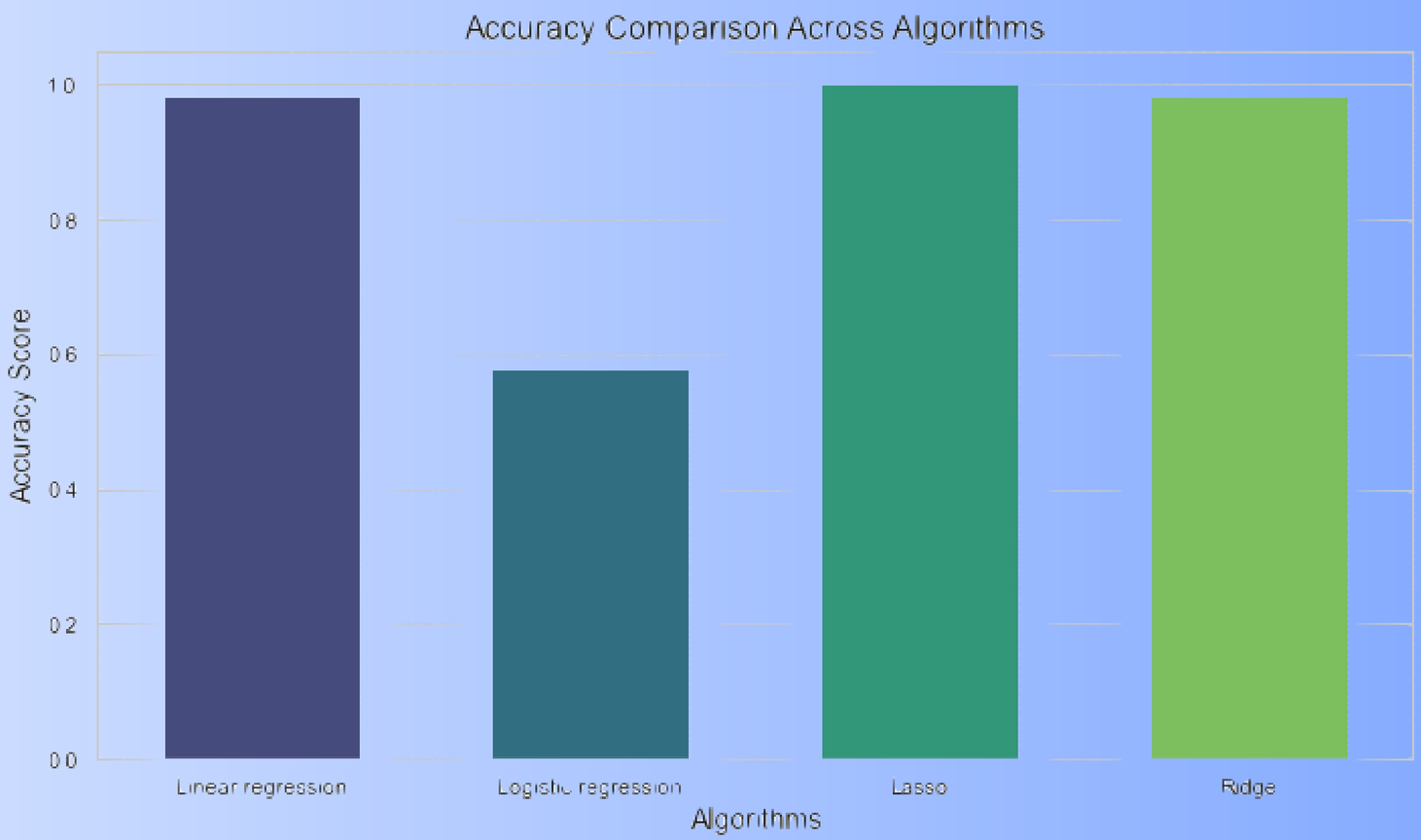


Correlation Between Features

Type: Table

رواد مصر الرقمية

DATA VISUALIZATION

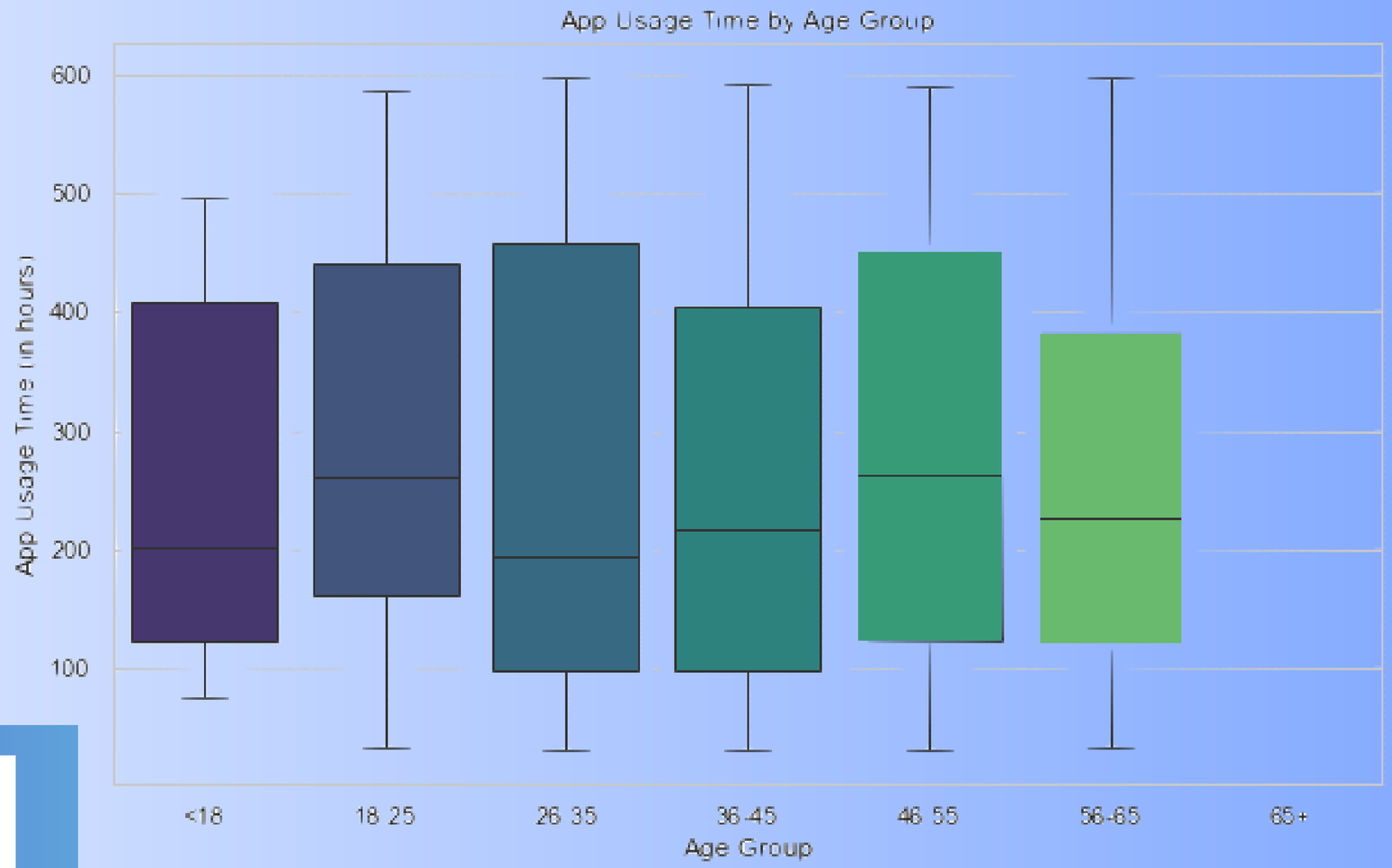


Accuracy Comparison Across Algorithms

Type: Bar Graph

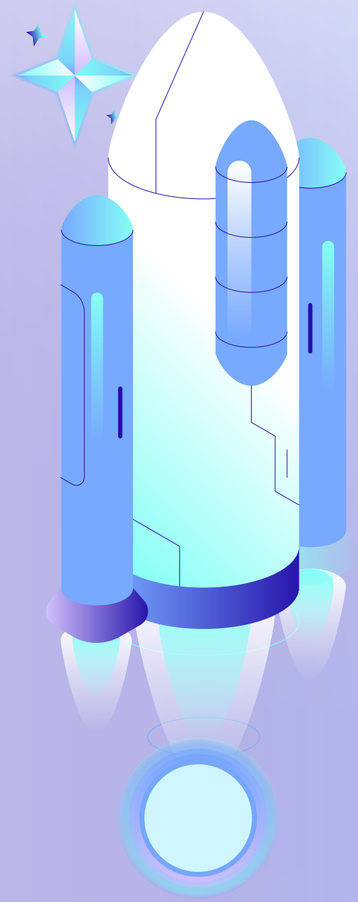
رواد مصر الرقمية

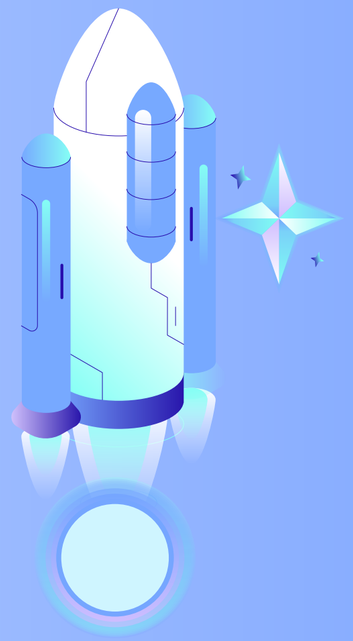
DATA VISUALIZATION



Type: Box plot

رواد مصر الرقمية





RESULTS

Insights Gained:

The most used apps are entertainment-based and are used by men with the age between 26-35

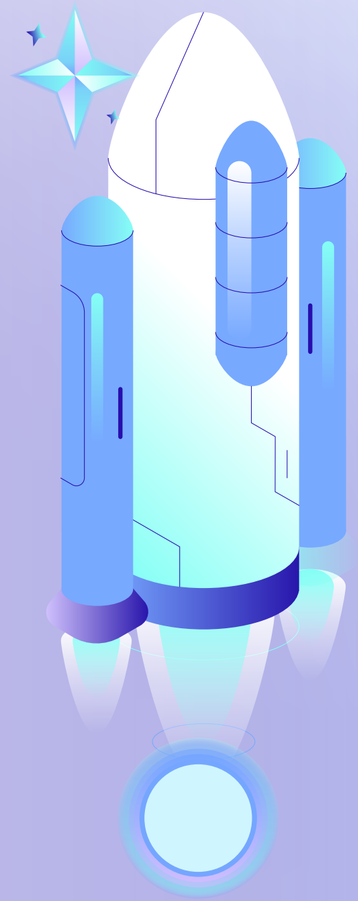
Device type significantly influences app usage patterns.

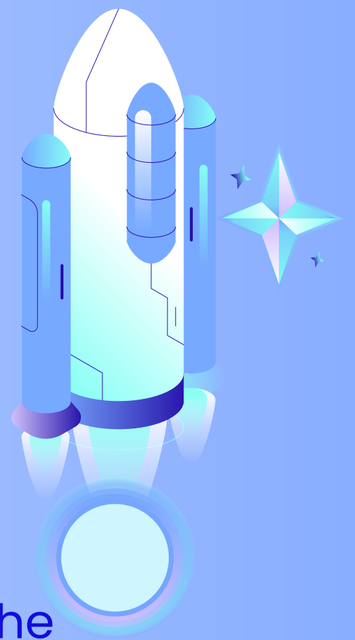
Model Performance:

Lasso achieved 99% accuracy in predicting user retention.

Logistic Regression had an accuracy of 57% in classifying app behavior

رواد مصر الرقمية





CONCLUSION

Key Takeaways:

- Data management and cleaning are crucial steps that determine the quality of analysis.
- Machine learning models provide actionable insights into user behavior.
- Visualizations are an effective tool to communicate findings.

Future Work: Implement additional models to predict user behavior more accurately and optimize the database for faster processing.

رواد مصر الرقمية

THANK YOU!

FROM: HAGER AHMED AND SOAD ATEF