

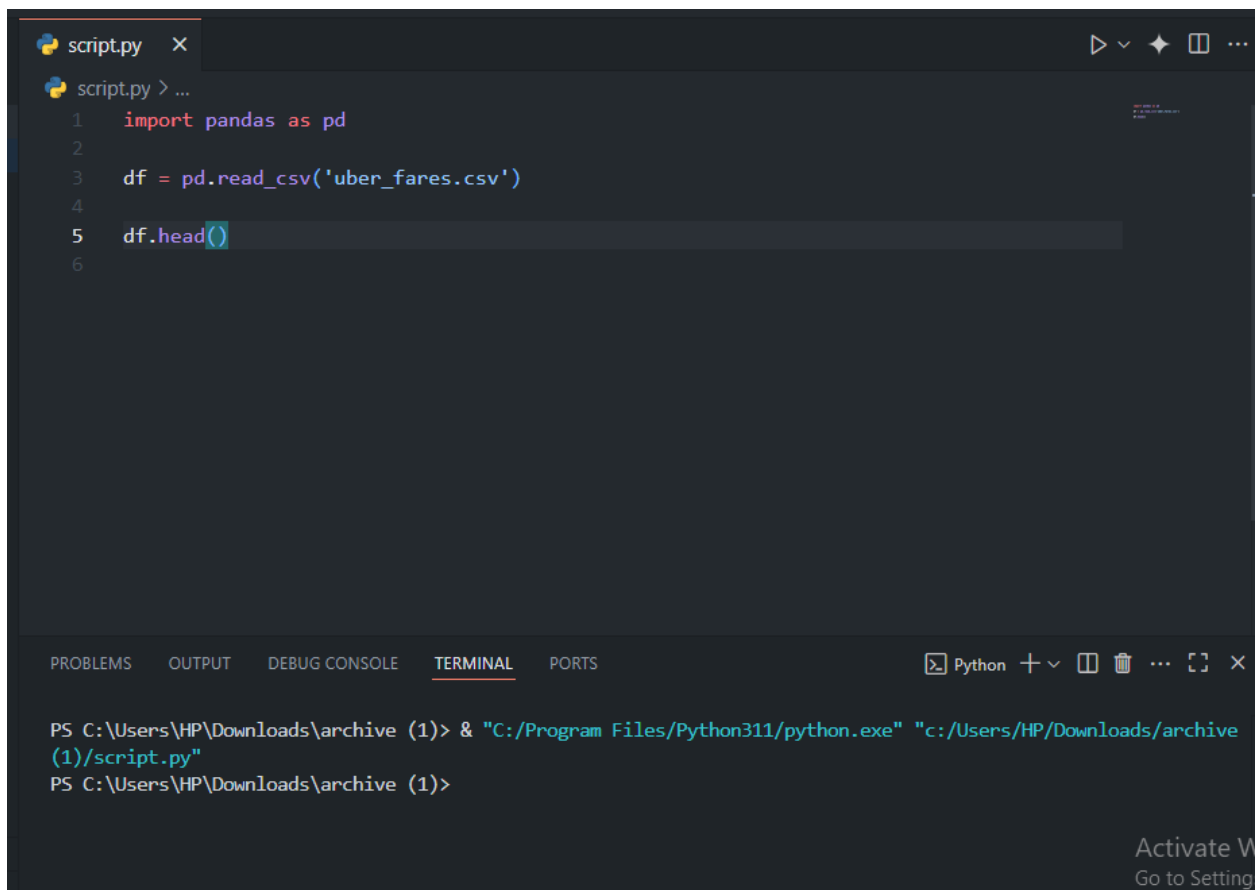
ASSIGNMENT OF BIGDATA

ID: 26092

NAME: MUHIRE Samuel

1. Data Understanding and Preparation

a. Load the dataset into a Pandas DataFrame using Python



The screenshot shows a code editor window with a file named 'script.py'. The code in the editor is as follows:

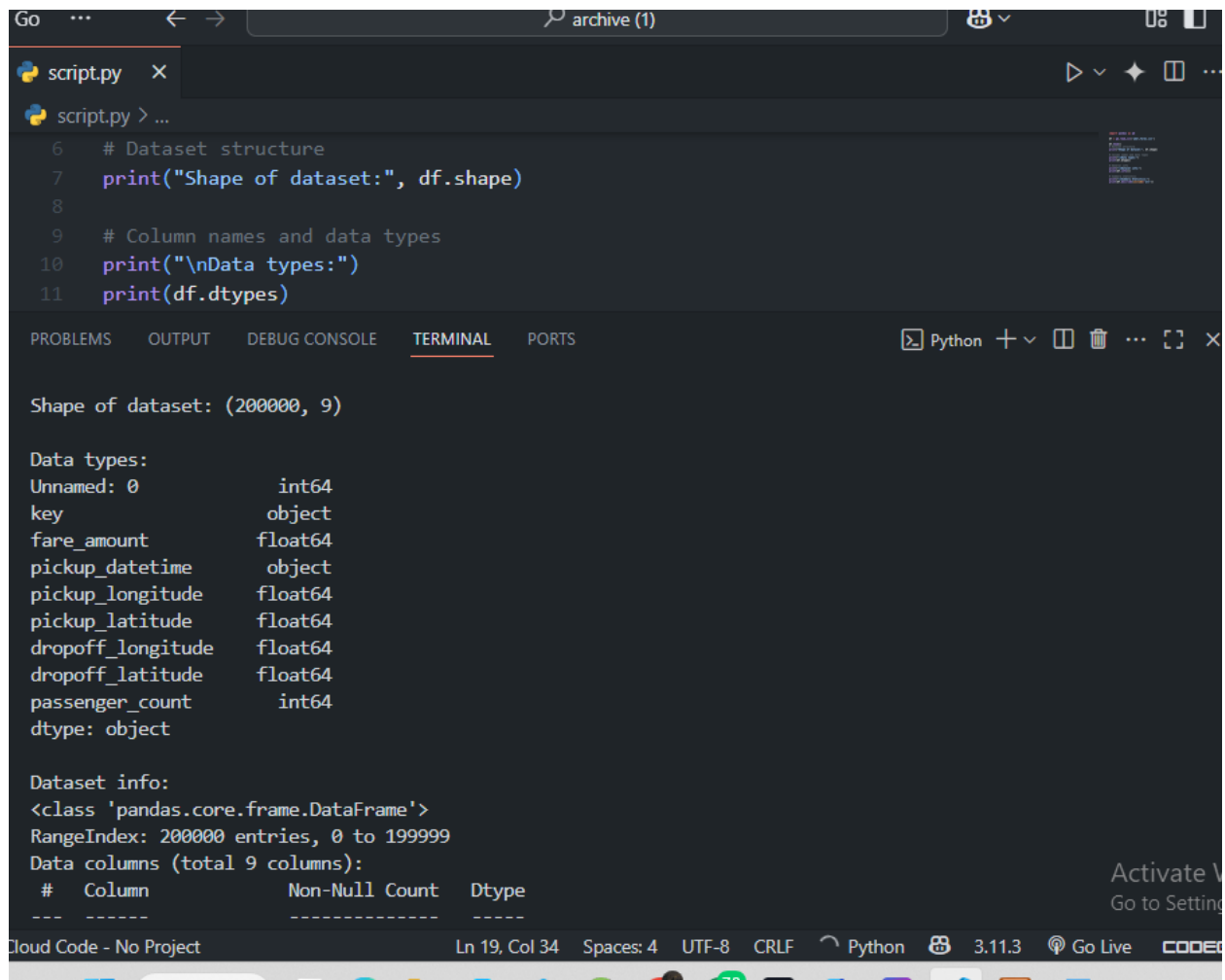
```
1 import pandas as pd
2
3 df = pd.read_csv('uber_fares.csv')
4
5 df.head()
6
```

Below the code editor is a terminal window. The terminal shows the command to run the script and the resulting prompt:

```
PS C:\Users\HP\Downloads\archive (1)> & "C:/Program Files/Python311/python.exe" "c:/Users/HP/Downloads/archive (1)/script.py"
PS C:\Users\HP\Downloads\archive (1)>
```

At the bottom right of the terminal, there is a message: "Activate W Go to Setting".

b. Perform comprehensive exploratory data analysis (EDA) to understand:



The image shows a Visual Studio Code editor window with a Python script named `script.py` and its terminal output. The script contains the following code:

```
6 # Dataset structure
7 print("Shape of dataset:", df.shape)
8
9 # Column names and data types
10 print("\nData types:")
11 print(df.dtypes)
```

The terminal output shows the following results:

```
Shape of dataset: (200000, 9)

Data types:
Unnamed: 0      int64
key            object
fare_amount    float64
pickup_datetime object
pickup_longitude float64
pickup_latitude float64
dropoff_longitude float64
dropoff_latitude float64
passenger_count int64
dtype: object

Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
#   Column      Non-Null Count  Dtype
---  -

```

The status bar at the bottom indicates the file is in the `archive (1)` folder, and the editor is set to Python 3.11.3. The terminal output is truncated on the right side.

b.Handle missing values and clean the data for analysis

Go ... ← → archive (1) 🔍

script.py ×

script.py > ...

```
21 # Drop rows with missing values
22 df_cleaned = df.dropna()
23
24 # Remove invalid passenger counts (Uber only allows 1-6)
25 df_cleaned = df_cleaned[(df_cleaned['passenger_count'] > 0) & (df_cleaned['passenger_
26
27 # Remove invalid fare_amount (no negative fares)
28 df_cleaned = df_cleaned[df_cleaned['fare_amount'] > 0]
29
30 # Remove invalid coordinates (basic range check for NYC region)
31 df_cleaned = df_cleaned[
32     (df_cleaned['pickup_latitude'].between(40, 42)) &
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python + - 🗑️ ... [] ×

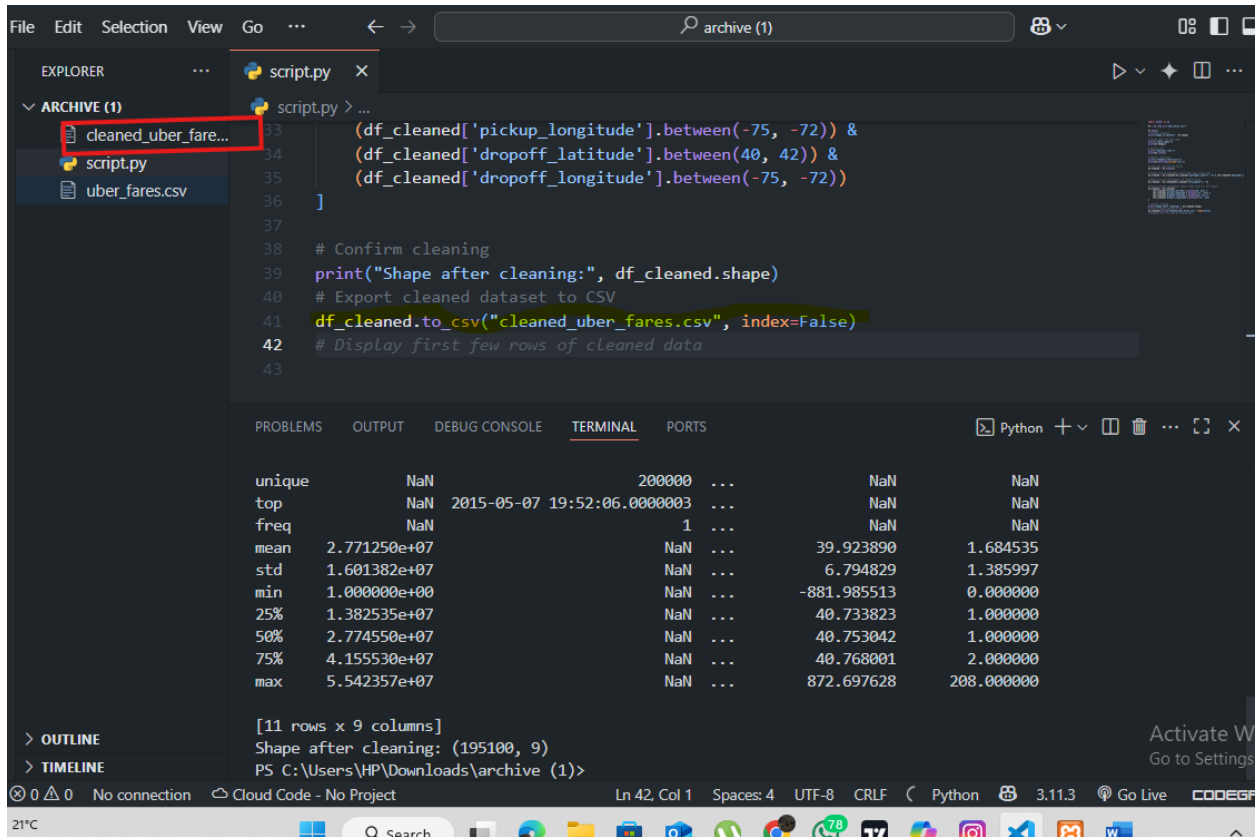
unique	NaN	200000	...	NaN	NaN
top	NaN	2015-05-07 19:52:06.0000003	...	NaN	NaN
freq	NaN	1	...	NaN	NaN
mean	2.771250e+07	NaN	...	39.923890	1.684535
std	1.601382e+07	NaN	...	6.794829	1.385997
min	1.000000e+00	NaN	...	-881.985513	0.000000
25%	1.382535e+07	NaN	...	40.733823	1.000000
50%	2.774550e+07	NaN	...	40.753042	1.000000
75%	4.155530e+07	NaN	...	40.768001	2.000000
max	5.542357e+07	NaN	...	872.697628	208.000000

[11 rows x 9 columns]
Shape after cleaning: (195100, 9)
PS C:\Users\HP\Downloads\archive (1)>

Activate Wind
Go to Settings to a

Cloud Code - No Project Ln 40, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.3 Go Live CODEGPT

c. Export the cleaned dataset as a CSV file for Power BI import



The screenshot shows a VS Code editor with a Python script in the main editor and its output in the terminal. The Explorer sidebar on the left shows a file named 'cleaned_uber_fare...' which is highlighted with a red box. The script in the main editor is as follows:

```
script.py
33 (df_cleaned['pickup_longitude'].between(-75, -72)) &
34 (df_cleaned['dropoff_latitude'].between(40, 42)) &
35 (df_cleaned['dropoff_longitude'].between(-75, -72))
36 ]
37
38 # Confirm cleaning
39 print("Shape after cleaning:", df_cleaned.shape)
40 # Export cleaned dataset to CSV
41 df_cleaned.to_csv("cleaned_uber_fares.csv", index=False)
42 # Display first few rows of cleaned data
43
```

The terminal output shows the following statistics:

Statistic	Value
unique	NaN
top	NaN
freq	NaN
mean	2.771250e+07
std	1.601382e+07
min	1.000000e+00
25%	1.382535e+07
50%	2.774550e+07
75%	4.155530e+07
max	5.542357e+07

The terminal also shows the shape of the cleaned dataset: [11 rows x 9 columns] and the path to the CSV file: PS C:\Users\HP\Downloads\archive (1)>

2. Exploratory Data Analysis (EDA)

```
archive (1)
TERMINAL
Python
Shape after cleaning: (195100, 9)
=== Descriptive Statistics ===
   Unnamed: 0   fare_amount   pickup_longitude   pickup_latitude   dropoff_longitude   dropoff_latitude   passenger_count
count  1.951000e+05  195100.000000   195100.000000   195100.000000   195100.000000   195100.000000   1
mean    2.771531e+07    11.350275    -73.975113     40.750976    -73.974176     40.751272
std     1.600896e+07     9.802850     0.039402     0.030144     0.039242     0.033545
min     1.000000e+00     0.010000    -74.719883     40.002405    -74.728123     40.005487
25%     1.383361e+07     6.000000    -73.992273     40.736453    -73.991597     40.735332
50%     2.776238e+07     8.500000    -73.982110     40.753304    -73.980541     40.753747
75%     4.154671e+07    12.500000    -73.968361     40.767549    -73.965380     40.768330
max     5.542357e+07    499.000000    -72.750302     41.366138    -72.759090     41.500000

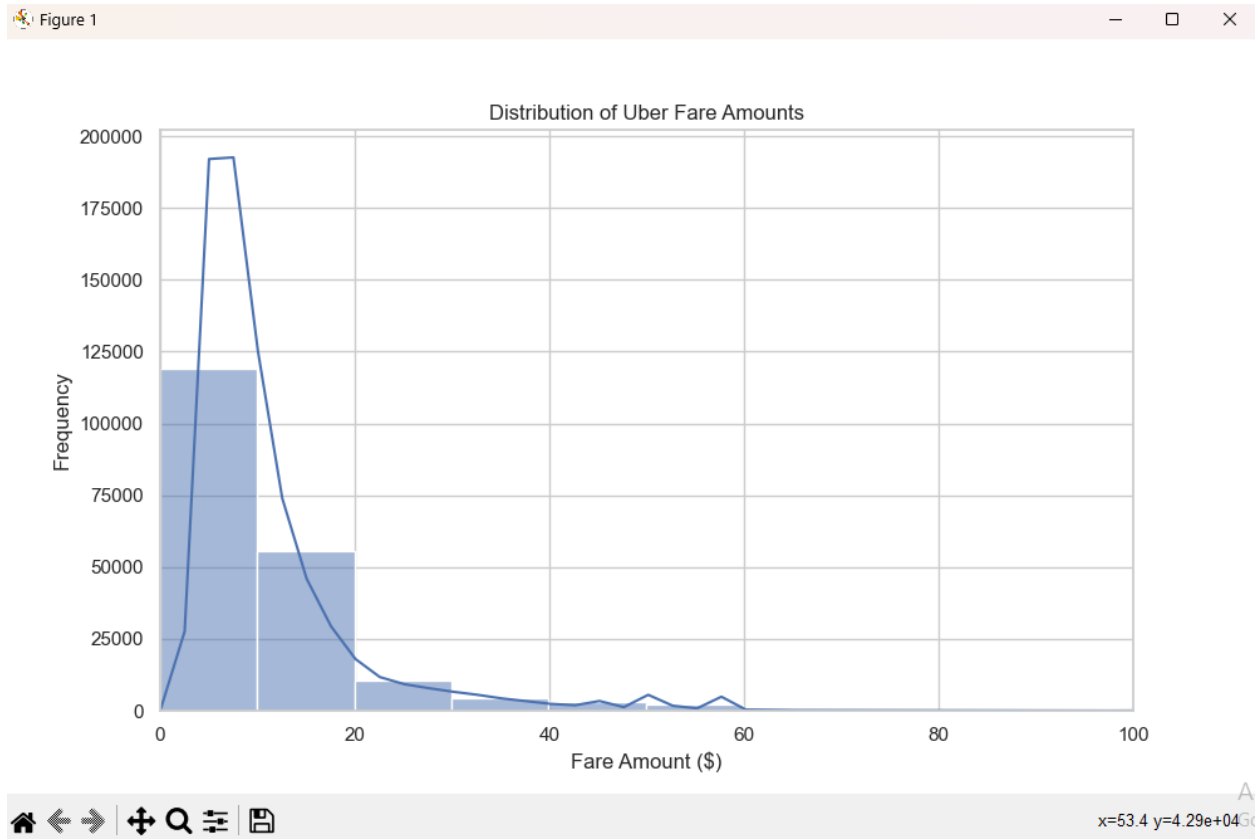
Median Fare Amount: 8.5

Mode Passenger Count: 1

Fare Amount Range: 498.99

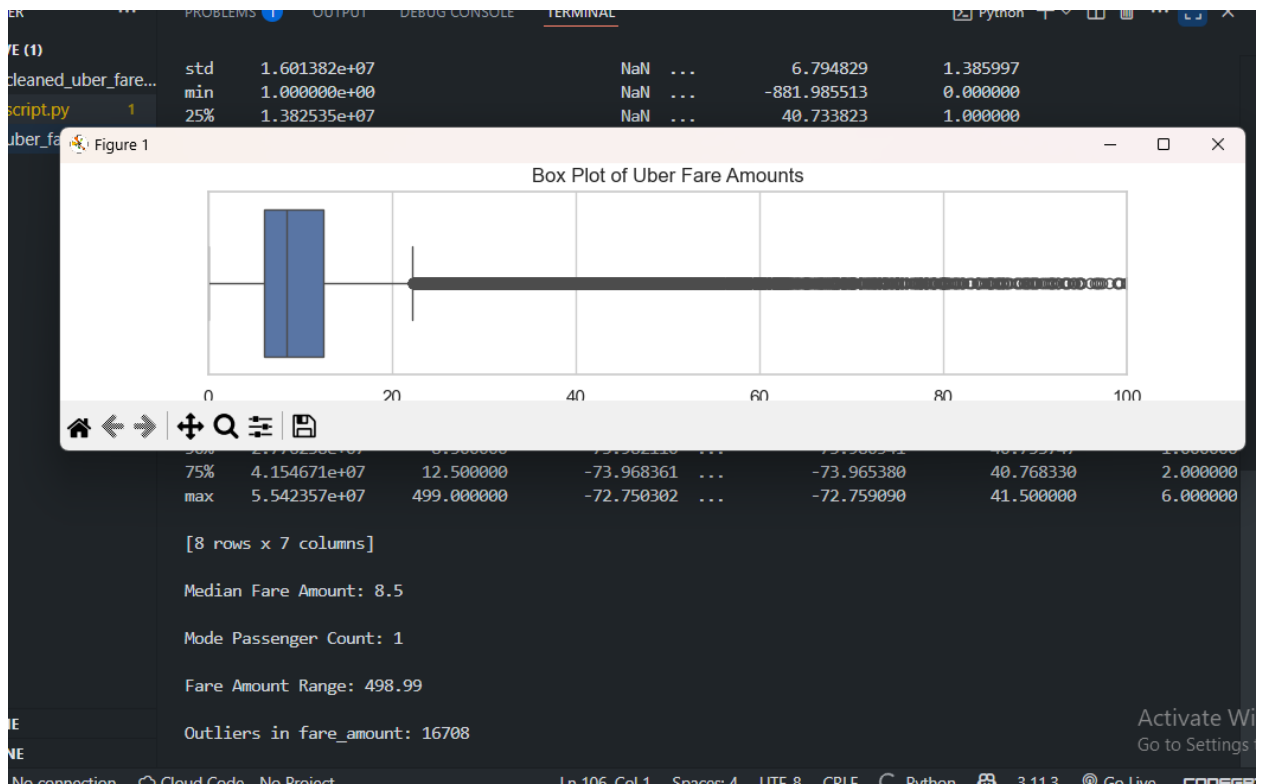
Outliers in fare_amount: 16708
PS C:\Users\HP\Downloads\archive (1)>
```

b. Create visualizations showing fare distribution patterns

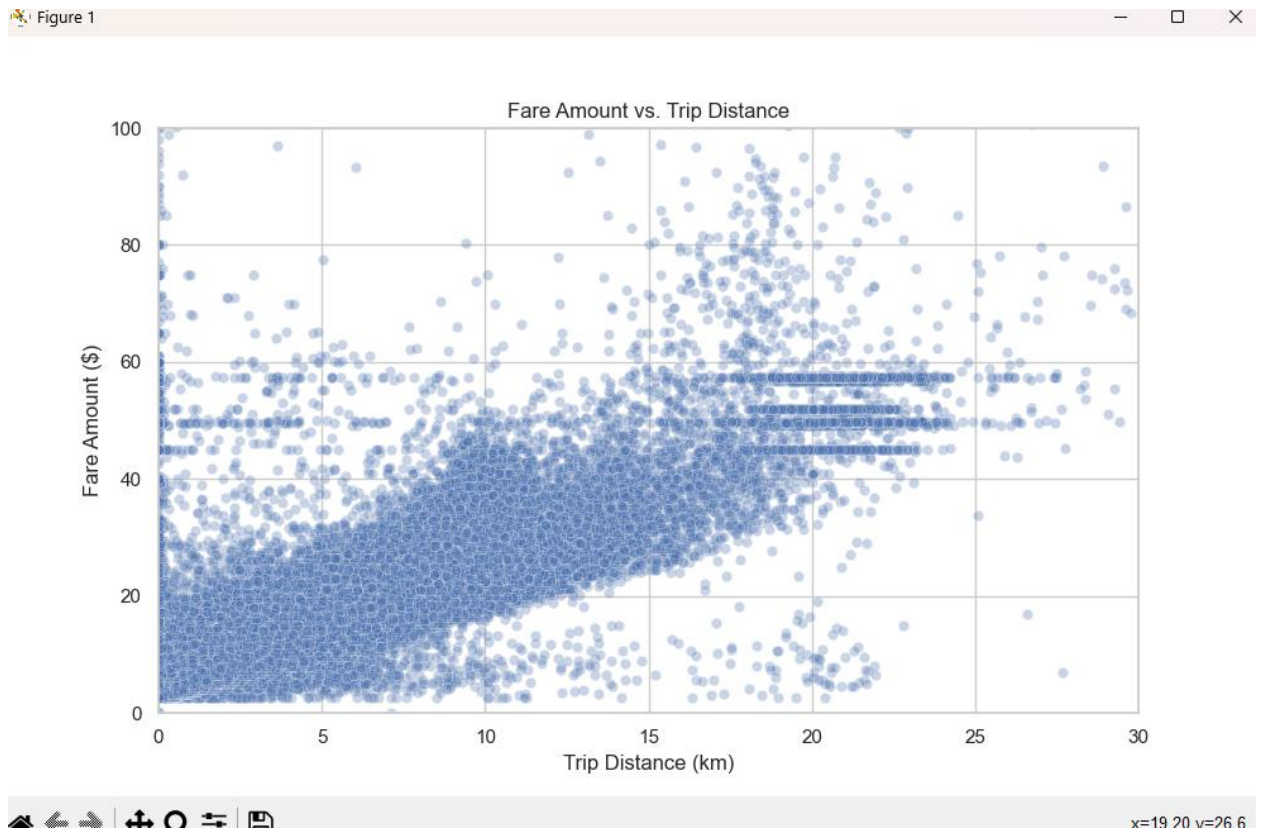


c. Analyze relationships between key variables:

- Fare amount vs. distance traveled

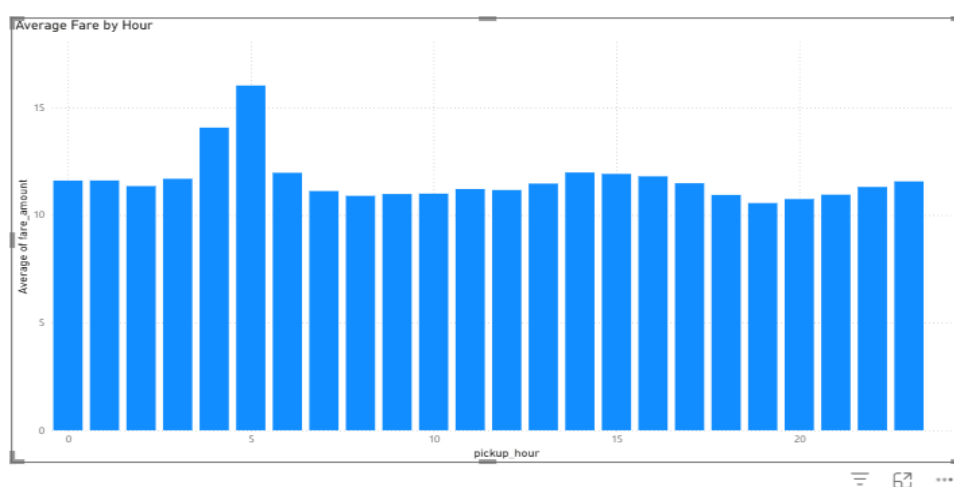


Fare amount vs. time of day



4. Data Analysis in Power BI

b. Create comprehensive visualizations analyzing:



2. Ride Patterns: Hourly, Daily, Monthly

Chart 1: Bar chart – Total rides per hour

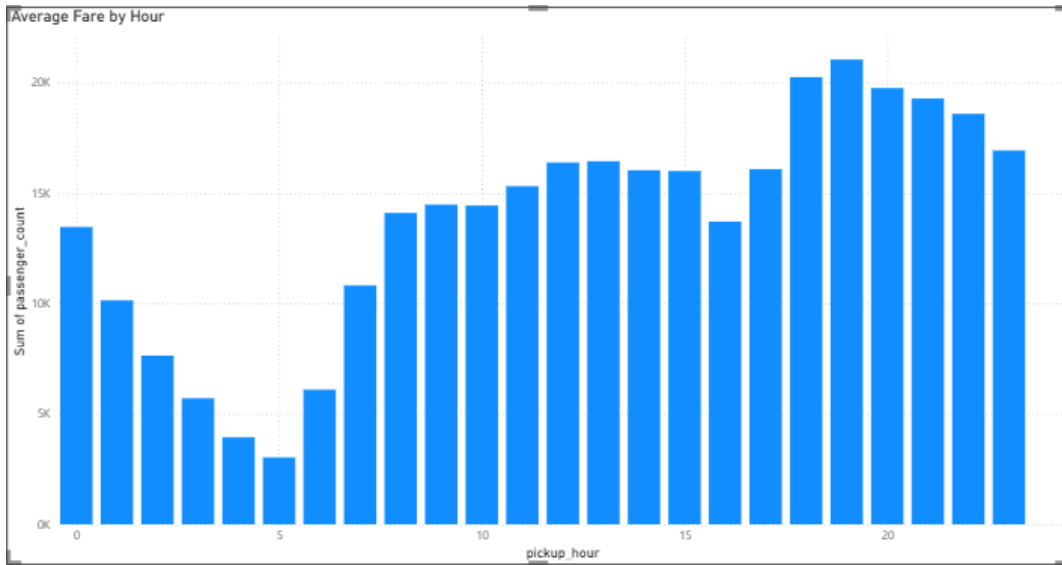


Chart 2: Bar chart – Rides per day of week

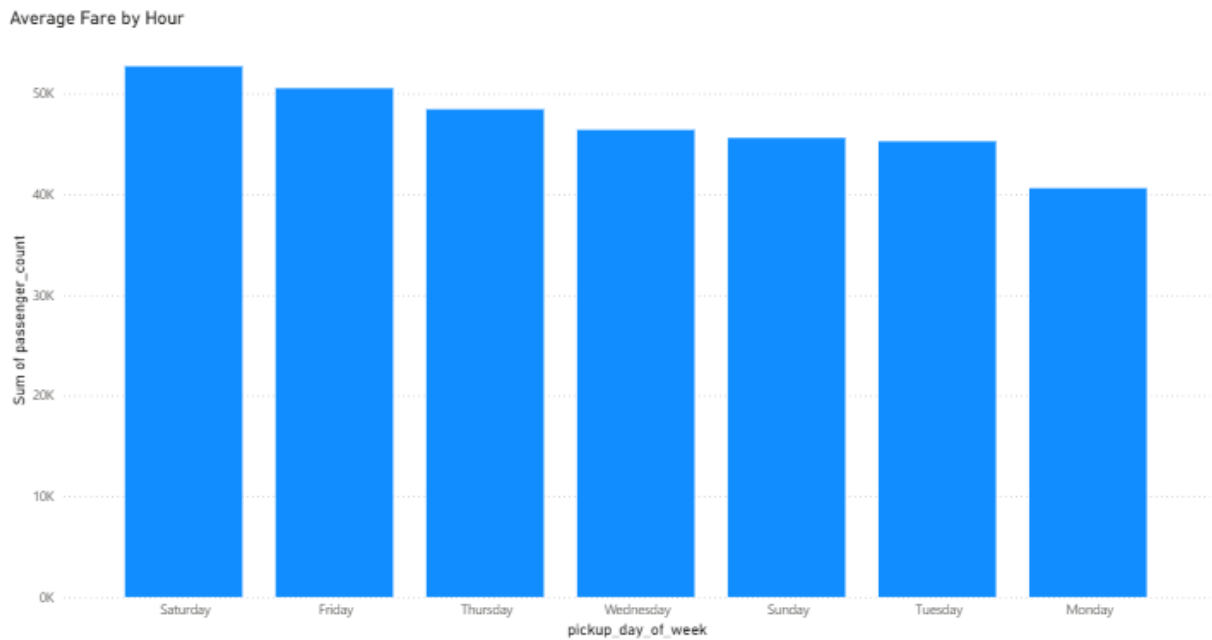
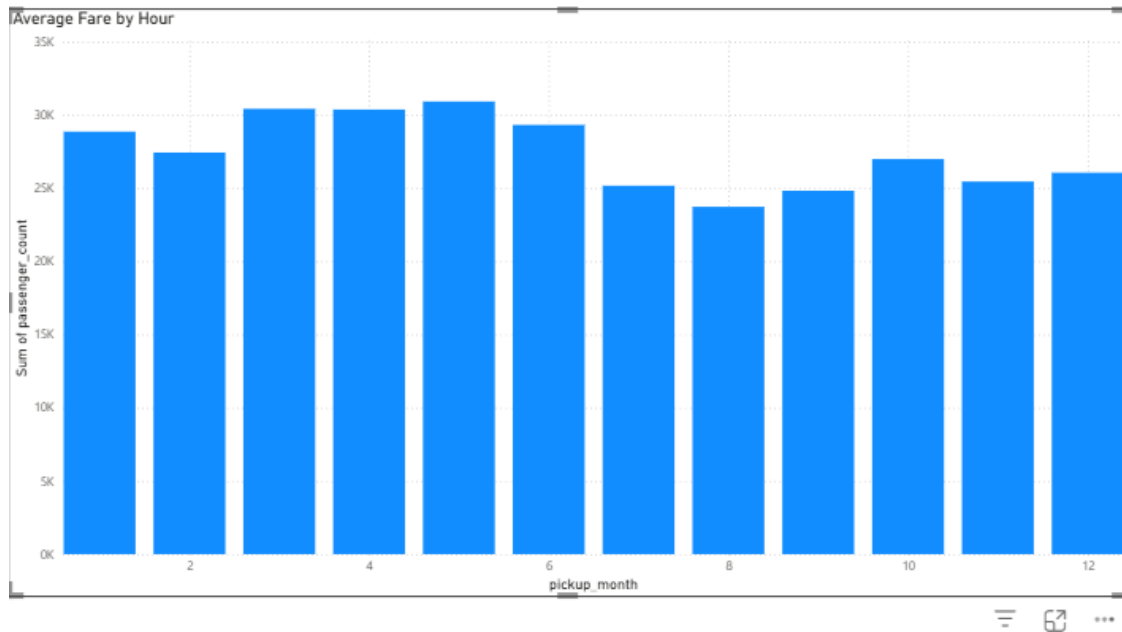


Chart 3: Line chart – Rides per month



Step 4c: Identify Busiest Periods

pickup_day_of_week_encoded	0	1	2	3	4	5
0	1/9/2009 12:27:56 AM	1/2/2009 1:30:25 AM	1/2/2009 2:16:47 AM	1/9/2009 3:33:00 AM	1/9/2009 4:04:00 AM	1/16/2009 5:56:40 AM
1	1/5/2009 12:21:00 AM	1/5/2009 1:08:00 AM	2/16/2009 2:15:09 AM	1/12/2009 3:09:51 AM	1/19/2009 4:00:00 AM	1/5/2009 5:06:40 AM
2	1/3/2009 12:06:00 AM	1/3/2009 1:25:00 AM	1/3/2009 2:06:01 AM	1/3/2009 3:52:00 AM	1/3/2009 4:21:00 AM	1/3/2009 5:00:00 AM
3	1/4/2009 12:10:49 AM	1/4/2009 1:01:00 AM	1/4/2009 2:18:28 AM	1/4/2009 3:03:50 AM	1/4/2009 4:06:19 AM	1/4/2009 5:14:27 AM
4	1/8/2009 12:09:57 AM	1/1/2009 1:15:22 AM	1/1/2009 2:05:03 AM	1/1/2009 3:15:17 AM	1/1/2009 4:04:08 AM	1/1/2009 5:30:40 AM
5	1/6/2009 12:54:15 AM	1/13/2009 1:17:58 AM	2/3/2009 2:44:00 AM	1/13/2009 3:05:30 AM	1/13/2009 4:14:44 AM	1/6/2009 5:23:00 AM
6	1/7/2009 12:21:00 AM	1/7/2009 1:56:40 AM	1/7/2009 2:00:00 AM	1/7/2009 3:58:19 AM	1/7/2009 4:47:58 AM	1/28/2009 5:06:19 AM
Total	1/3/2009 12:06:00 AM	1/1/2009 1:15:22 AM	1/1/2009 2:05:03 AM	1/1/2009 3:15:17 AM	1/1/2009 4:04:08 AM	1/1/2009 5:30:40 AM

PART 5: INTERACTIVE DASHBOARD CREATION

