databricks NYC_Taxi_Data_Analysis

(https://databricks.com/)

Part A: Basic Prescriptive Analytics:

Step 1 & 2: Data Ingestion

```
3
    df = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/yellow_tripdata_2015_01.csv")
```

```
4
    df.show(5)
```

```
+--------+--------------------+--------------------+---------------+-------------+--------------------+-------------------+----------+--------------------+--------------------+-------------------+------------+------------+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|   pickup_longitude|    pickup_latitude|RateCodeID|store_and_fwd_flag|   dropoff_longitude|   dropoff_latitude|payment_type|fare_amount|extra|mta_tax|tip_amount|tolls_amount|improvement_surcharge|total_amount|
+--------+--------------------+--------------------+---------------+-------------+--------------------+-------------------+----------+--------------------+--------------------+-------------------+------------+------------+-----+-------+----------+------------+---------------------+------------+
|       2| 2015-01-15 19:05:39|  2015-01-15 19:23:42|              1|         1.59|   -73.993896484375|40.750110626220703|         1|                   N|-73.974784851074219|40.750617980957031|           1|         12|    1|    0.5|      3.25|           0|                  0.3|       17.05|
|       1| 2015-01-10 20:33:38|  2015-01-10 20:53:28|              1|         3.30| -74.00164794921875|  40.7242431640625|         1|                   N|-73.994415283203125|40.759109497070313|           1|       14.5|  0.5|    0.5|         2|           0|                  0.3|        17.8|
|       1| 2015-01-10 20:33:38|  2015-01-10 20:43:41|              1|         1.80|-73.963340759277344|40.802787780761719|         1|                   N|-73.951820373535156|40.824413299560547|           2|        9.5|  0.5|    0.5|         0|           0|                  0.3|        10.8|
|       1| 2015-01-10 20:33:39|  2015-01-10 20:35:31|              1|          .50|-74.009086608886719|40.713817596435547|         1|                   N|-74.004325866699219|40.719985961914063|           2|        3.5|  0.5|    0.5|         0|           0|                  0.3|         4.8|
|       1| 2015-01-10 20:33:39|  2015-01-10 20:52:58|              1|         3.00|-73.971176147460938|40.762428283691406|         1|                   N|-74.004180908203125|40.742652893066406|           2|         15|  0.5|    0.5|         0|           0|                  0.3|        16.3|
+--------+--------------------+--------------------+---------------+-------------+--------------------+-------------------+----------+--------------------+--------------------+-------------------+------------+------------+-----+-------+----------+------------+---------------------+------------+
only showing top 5 rows
```

Step 3: Initial Data Exploration

```
6
    df.count()
```

```
12748986
```

```
7
    df.columns
```

```
['VendorID',
 'tpep_pickup_datetime',
 'tpep_dropoff_datetime',
 'passenger_count',
 'trip_distance',
 'pickup_longitude',
 'pickup_latitude',
 'RateCodeID',
 'store_and_fwd_flag',
 'dropoff_longitude',
 'dropoff_latitude',
 'payment_type',
```

```
 'fare_amount',
 'extra',
 'mta_tax',
 'tip_amount',
 'tolls_amount',
 'improvement_surcharge',
 'total_amount']
```

8

```
df.summary()
```

```
DataFrame[summary: string, VendorID: string, tpep_pickup_datetime: string, tpep_dropoff_datetime: string, passenger_count: string, trip_
distance: string, pickup_longitude: string, pickup_latitude: string, RateCodeID: string, store_and_fwd_flag: string, dropoff_longitude:
string, dropoff_latitude: string, payment_type: string, fare_amount: string, extra: string, mta_tax: string, tip_amount: string, tolls_a
mount: string, improvement_surcharge: string, total_amount: string]
```

9

```
# Display the schema of the dataset
df.printSchema()
```

```
root
 |-- VendorID: string (nullable = true)
 |-- tpep_pickup_datetime: string (nullable = true)
 |-- tpep_dropoff_datetime: string (nullable = true)
 |-- passenger_count: string (nullable = true)
 |-- trip_distance: string (nullable = true)
 |-- pickup_longitude: string (nullable = true)
 |-- pickup_latitude: string (nullable = true)
 |-- RateCodeID: string (nullable = true)
 |-- store_and_fwd_flag: string (nullable = true)
 |-- dropoff_longitude: string (nullable = true)
 |-- dropoff_latitude: string (nullable = true)
 |-- payment_type: string (nullable = true)
 |-- fare_amount: string (nullable = true)
 |-- extra: string (nullable = true)
 |-- mta_tax: string (nullable = true)
 |-- tip_amount: string (nullable = true)
 |-- tolls_amount: string (nullable = true)
 |-- improvement_surcharge: string (nullable = true)
 |-- total_amount: string (nullable = true)
```

10

```
# Display basic statistics of numerical columns, such as trip distance, passenger count, and fare amount
df.select("trip_distance", "passenger_count", "fare_amount").describe().show()
```

```
+-------+------------------+------------------+------------------+
|summary|     trip_distance|   passenger_count|       fare_amount|
+-------+------------------+------------------+------------------+
|  count|          12748986|          12748986|          12748986|
|   mean|13.459129611562718|1.6814908260154964|11.905659425776989|
| stddev| 9844.094218468374|1.3379235172874737|10.302537135952232|
|    min|               .00|                 0|             -0.01|
|    max|             99.90|                 9|            999.99|
+-------+------------------+------------------+------------------+
```

STEP 4 : Data Cleaning

12

```
# Removing missing values of column "fare_amount", "trip_distance" & "passenger_count"
df_cleaned = df.na.drop(subset=["fare_amount", "trip_distance", "passenger_count"])
df_cleaned.describe().show()
```

```
+-------+------------------+--------------------+--------------------+------------------+------------------+--------------------+-------
-----------+------------------+------------------+------------------+------------------+------------------+--------
-----------+------------------+------------------+------------------+---------------------+------------------+
```

```
|summary|          VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|   passenger_count|     trip_distance|   pickup_longitude|   pick
up_latitude|        RateCodeID|store_and_fwd_flag|  dropoff_longitude|   dropoff_latitude|      payment_type|       fare_amount|
extra|           mta_tax|        tip_amount|       tolls_amount|improvement_surcharge|      total_amount|
+-------+------------------+--------------------+---------------------+------------------+------------------+-------------------+-------
-----------+------------------+------------------+-------------------+-------------------+------------------+-------------------+--------
-----------+------------------+------------------+-------------------+---------------------+------------------+
|  count|          12748986|            12748986|             12748986|          12748986|          12748986|           12748986|
12748986|          12748986|          12748986|           12748986|           12748983|          12748986|
12748986|          12748986|          12748986|           12748986|            12748983|          12748986|
|   mean|1.5214373127400094|                NULL|                 NULL|1.6814908260154964|13.459129611562718| -72.56183777902534| 39.972
82304763482|1.0369007386156044|              NULL| -72.60903923063492|   39.9996144802455|1.3867115392549652|11.905659425776989|0.308278
95724412907|0.4977986092384132|1.8538136460419994|0.24349839430352666|  0.28314307893811447|15.108294537401271|
| stddev|0.4995402498256225|                NULL|                 NULL|1.3379235172874737|  9844.094218468374| 10.125103592972911|   5.57
86905190884|0.6732239779497589|              NULL|   9.96603703803103|   5.48774188661968|0.4988610635053929|10.302537135952232| 0.59166
43112912818|0.0353422867098315|1106.4323141838747| 1.5271714003797854|  0.06908632935830779| 1106.503246710499|
|    min|                 1| 2015-01-01 00:00:00|  2015-01-01 00:00:00|                 0|              .00|-1.3151819705963135|
0|                1|                N|-0.1166670024394989|-9.0291566848754883|                 1|            -0.01|             -0.0
9|             -0.5|            -0.01|            -10.66|                    0|            -0.31|
|    max|                 2| 2015-01-31 23:59:59|  2016-02-02 16:30:52|                 9|             99.90| 78.662651062011719|9.58784
67559814453|                99|                Y| 85.274024963378906| 9.9809532165527344|                 5|            999.99|
999.99|               0.5|            99.96|             999.99|                 0.3|             99.99|
+-------+------------------+--------------------+---------------------+------------------+------------------+-------------------+-------
-----------+------------------+------------------+-------------------+-------------------+------------------+-------------------+--------
-----------+------------------+------------------+-------------------+---------------------+------------------+
```

13

```
from pyspark.sql.functions import col, regexp_extract
# Filter out rows with invalid data (e.g., fare_amount < 0 or trip_distance = 0)
df_cleaned = df_cleaned.filter((df_cleaned["fare_amount"] > 0) & (df_cleaned["trip_distance"] > 0) & (df_cleaned["pickup_longitude"
(df_cleaned["dropoff_longitude"] != 0) & (df_cleaned["dropoff_latitude"] != 0))
valid_fare_amount_regex = "^[0-9]+(\.[0-9]{1,2})?$"
df_cleaned = df_cleaned.filter(regexp_extract(col("fare_amount").cast("string"), valid_fare_amount_regex, 0) != "")
df_cleaned.describe().show()
```

```
+-------+------------------+--------------------+---------------------+------------------+------------------+-------------------+-------
-----------+------------------+------------------+-------------------+-------------------+------------------+-------------------+--------
-----------+------------------+------------------+-------------------+---------------------+------------------+
|summary|          VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|   passenger_count|     trip_distance|   pickup_longitude|    pic
kup_latitude|        RateCodeID|store_and_fwd_flag|  dropoff_longitude|  dropoff_latitude|      payment_type|       fare_amount|
extra|           mta_tax|        tip_amount|      tolls_amount|improvement_surcharge|      total_amount|
+-------+------------------+--------------------+---------------------+------------------+------------------+-------------------+-------
-----------+------------------+------------------+-------------------+-------------------+------------------+-------------------+--------
-----------+------------------+------------------+-------------------+---------------------+------------------+
|  count|           9474464|             9474464|              9474464|           9474464|           9474464|            9474464|
9474464|           9474464|           9474464|            9474464|            9474464|           9474464|            9
474464|            9474464|           9474464|           9474464|              9474464|           9474464|
|   mean|1.5275759135292508|                NULL|                 NULL|1.6921189420319713|17.829444945907543| -73.97325995357461| 40.750
038711870616| 1.034897910847516|              NULL| -73.97297563495772| 40.75096497538369|  1.35772577741601|13.910922983084133| 0.31404
57919308153|  0.4985670429482871|2.2235511032600423|0.3158089069733504|   0.2828745562794641|17.562448306351715|
| stddev|0.4992390162031469|                NULL|                 NULL|1.3442813854320428|11419.212112398436|0.07848886714950458|0.12801
451565479846|0.5417347281929257|              NULL| 0.3861348522326615|0.1887650711938517|0.4870103396253098|10.313916600093934|0.456318
40124394124|0.026728734263891794|1283.4682079132479|1.5335839535427085|  0.06960138507560257| 1283.531044935015|
|    min|                 1| 2015-01-01 00:00:00|  2015-01-01 00:00:00|                 0|              1.00|-1.3151819705963135| 18.625
944137573242|                 1|                N|-1.2284070253372192|18.625944137573242|                 1|                  1|
-0.4|                 0|                0|                0|                  0|            -69.7|
|    max|                 2| 2015-01-31 23:59:59|  2015-02-15 21:09:34|                 9|             99.90| 78.662651062011719| 9.5878
467559814453|                99|                Y| 78.662651062011719|9.9809532165527344|                 4|             99.99|
7|               0.5|            98.65|             95.5|                 0.3|            99.99|
+-------+------------------+--------------------+---------------------+------------------+------------------+-------------------+-------
-----------+------------------+------------------+-------------------+-------------------+------------------+-------------------+--------
-----------+------------------+------------------+-------------------+---------------------+------------------+
```

14

```
from pyspark.sql.functions import col

# Convert the pickup_datetime and dropoff_datetime columns to timestamp data types
df_cleaned = df_cleaned.withColumn("pickup_datetime", col("tpep_pickup_datetime").cast("timestamp"))
df_cleaned = df_cleaned.withColumn("dropoff_datetime", col("tpep_dropoff_datetime").cast("timestamp"))

df_cleaned.show(5)
```

```
+--------+--------------------+--------------------+---------------+-------------+-------------------+------------------+----------+---
---------------+------------------+------------------+------------+-----------+-----+-------+----------+------------+------------------
---+------------+-------------------+-------------------+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|   pickup_longitude|   pickup_latitude|RateCodeID|sto
re_and_fwd_flag|  dropoff_longitude|  dropoff_latitude|payment_type|fare_amount|extra|mta_tax|tip_amount|tolls_amount|improvement_surcha
rge|total_amount|    pickup_datetime|   dropoff_datetime|
+--------+--------------------+--------------------+---------------+-------------+-------------------+------------------+----------+---
---------------+------------------+------------------+------------+-----------+-----+-------+----------+------------+------------------
---+------------+-------------------+-------------------+
|       2| 2015-01-15 19:05:39|  2015-01-15 19:23:42|              1|         1.59|   -73.993896484375|40.750110626220703|         1|
N|-73.974784851074219|40.750617980957031|           1|         12|    1|    0.5|      3.25|           0|                  0.3|       17.
05|2015-01-15 19:05:39|2015-01-15 19:23:42|
|       1| 2015-01-10 20:33:38|  2015-01-10 20:53:28|              1|         3.30| -74.00164794921875|  40.7242431640625|         1|
N|-73.994415283203125|40.759109497070313|           1|       14.5|  0.5|    0.5|         2|           0|                  0.3|        1
7.8|2015-01-10 20:33:38|2015-01-10 20:53:28|
|       1| 2015-01-10 20:33:38|  2015-01-10 20:43:41|              1|         1.80|-73.963340759277344|40.802787780761719|         1|
N|-73.951820373535156|40.824413299560547|           2|        9.5|  0.5|    0.5|         0|           0|                  0.3|        1
0.8|2015-01-10 20:33:38|2015-01-10 20:43:41|
|       1| 2015-01-10 20:33:39|  2015-01-10 20:52:58|              1|         3.00|-73.971176147460938|40.762428283691406|         1|
N|-74.004180908203125|40.742652893066406|           2|         15|  0.5|    0.5|         0|           0|                  0.3|        1
6.3|2015-01-10 20:33:39|2015-01-10 20:52:58|
|       1| 2015-01-10 20:33:39|  2015-01-10 20:53:52|              1|         9.00|-73.874374389648438|  40.7740478515625|         1|
N|-73.986976623535156|40.758193969726563|           1|         27|  0.5|    0.5|       6.7|        5.33|                  0.3|       40.
33|2015-01-10 20:33:39|2015-01-10 20:53:52|
+--------+--------------------+--------------------+---------------+-------------+-------------------+------------------+----------+---
---------------+------------------+------------------+------------+-----------+-----+-------+----------+------------+------------------
---+------------+-------------------+-------------------+
only showing top 5 rows
```

15

```
from pyspark.sql.functions import unix_timestamp, col

# Create new columns, such as trip duration (in minutes) and trip speed (in miles per hour)
df_cleaned = df_cleaned.withColumn("trip_duration_min",
                                   (unix_timestamp("dropoff_datetime") - unix_timestamp("pickup_datetime")) / 60)


df_cleaned = df_cleaned.withColumn("trip_speed_mph",
                                   col("trip_distance") / (col("trip_duration_min") / 60))

df_cleaned.select("pickup_datetime", "dropoff_datetime", "trip_duration_min", "trip_speed_mph").show(5)
```

```
+-------------------+-------------------+------------------+------------------+
|    pickup_datetime|   dropoff_datetime| trip_duration_min|    trip_speed_mph|
+-------------------+-------------------+------------------+------------------+
|2015-01-15 19:05:39|2015-01-15 19:23:42|             18.05| 5.285318559556787|
|2015-01-10 20:33:38|2015-01-10 20:53:28|19.833333333333332| 9.983193277310924|
|2015-01-10 20:33:38|2015-01-10 20:43:41|             10.05|10.746268656716417|
|2015-01-10 20:33:39|2015-01-10 20:52:58|19.316666666666666| 9.318377911993098|
|2015-01-10 20:33:39|2015-01-10 20:53:52|20.216666666666665|26.710634789777412|
+-------------------+-------------------+------------------+------------------+
only showing top 5 rows
```

STEP 5: Exploratory Data Analysis

17

```
# avg fare and avg distance group by passenger count
df_grouped_by_passenger = df_cleaned.groupBy("passenger_count").agg(
    {"fare_amount": "avg", "trip_distance": "avg"}
)

# renaming temporarily
df_grouped_by_passenger = df_grouped_by_passenger.withColumnRenamed("avg(fare_amount)", "avg_fare")\
                                                 .withColumnRenamed("avg(trip_distance)", "avg_trip_distance")

# result
df_grouped_by_passenger.show()
```

```
+---------------+------------------+------------------+
|passenger_count|          avg_fare| avg_trip_distance|
+---------------+------------------+------------------+
|              3|14.042959587975478| 3.502365063959862|
|              0|12.731226210551675| 2.901132257287401|
|              5|14.038445848844917| 3.540712301877328|
|              6| 13.88652208851831| 3.472525537512198|
|              1|13.790829828712587| 19.80164590310544|
|              4|  14.0730966302188| 3.509825008583094|
|              2|14.390632541239809|23.563276703570445|
|              9|              69.7|            15.962|
|              7|              15.6|              4.28|
|              8|              33.5| 7.263333333333333|
+---------------+------------------+------------------+
```

18

```
from pyspark.sql.functions import hour
# taking hour out of the datetime
df_with_hour = df_cleaned.withColumn("pickup_hour", hour(col("pickup_datetime")))

# group by hour
df_busiest_hours = df_with_hour.groupBy("pickup_hour").count()

df_busiest_hours = df_busiest_hours.orderBy(col("count").desc())

df_busiest_hours.show()
```

```
+-----------+------+
|pickup_hour| count|
+-----------+------+
|         19|592172|
|         18|584945|
|         20|557565|
|         21|554629|
|         22|544294|
|         17|489310|
|         23|473549|
|         14|470190|
|         15|465655|
|         13|450447|
|         12|447838|
|         11|422559|
|         16|417757|
|          9|407165|
|         10|402276|
|          8|400821|
|          0|375346|
|          7|341971|
|          1|284178|
|          2|214544|
+-----------+------+
only showing top 20 rows
```

19

```
# group by location
df_avg_fare_by_location = df_cleaned.groupBy("pickup_longitude", "pickup_latitude").agg(
    {"fare_amount": "avg"}
)

# renaming tepmorarily
df_avg_fare_by_location = df_avg_fare_by_location.withColumnRenamed("avg(fare_amount)", "avg_fare")


df_avg_fare_by_location = df_avg_fare_by_location.orderBy(col("avg_fare").desc())


df_avg_fare_by_location.show()
```

```
+------------------+-----------------+--------+
|   pickup_longitude|   pickup_latitude|avg_fare|
+------------------+-----------------+--------+
|-73.961532592773438|40.770637512207031|  4008.0|
|-73.950325012207031|40.752861022949219|   800.0|
|-73.942741394042969|40.790802001953125|   780.0|
|-73.950279235839844|40.777347564697266|  760.01|
|-73.826431274414063|40.833961486816406|   600.0|
|-73.925872802734375|40.743618011474609|   525.0|
|-73.807296752929688|40.656135559082031|   489.5|
|-73.993026733398437|40.757881164550781|  467.54|
|  -73.9478759765625|40.583606719970703|   450.0|
|-73.977920532226563|  40.7623291015625|   448.0|
|-73.974945068359375|40.760028839111328|   440.0|
|-74.000579833984375|40.722129821777344|   435.0|
|-73.781707763671875|40.644550323486328|   434.5|
|-73.789047241210937|40.647251129150391|   420.0|
| -73.98858642578125|40.768974304199219|  414.44|
|-73.873367309570313|40.774147033691406|   405.0|
|-73.973373413085937|40.746353149414063|   400.0|
|-73.789115905761719|  40.6422119140625|   375.5|
|-73.788642883300781|40.641929626464844|   370.0|
|-73.978889465332031|40.749427795410156|   370.0|
+------------------+-----------------+--------+
only showing top 20 rows
```

STEP 6: Visualizing the Data

21

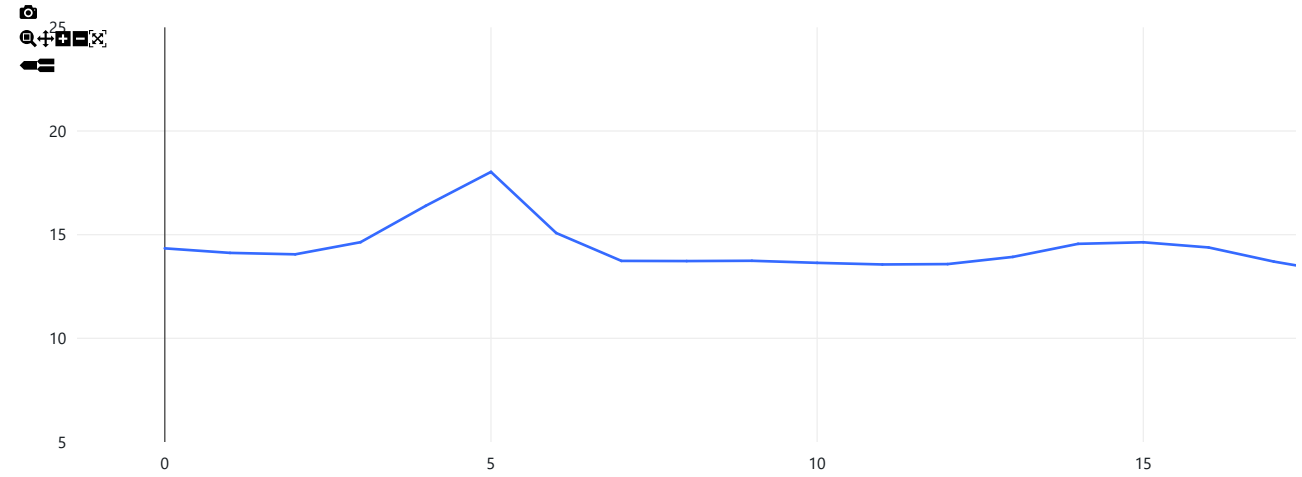
```
# Use display function to Visualize trip distance data
df_cleaned = df_cleaned.withColumn("trip_distance", col("trip_distance").cast("float"))
display(df_cleaned.select("trip_distance"))
```

Table Trip Distance Line Graph

1,081+ rows | Truncated data

```
22
    from pyspark.sql.functions import hour, avg

    # Calculate the average fare by hour
    avg_fare_by_hour = df_cleaned.groupBy(hour(col("pickup_datetime")).alias("pickup_hour")).agg(avg("fare_amount").alias("avg_fare_amou

    display(avg_fare_by_hour)
```

Table Average Fare Line Graph



24 rows

STEP 7: Summary and Insights

In step 5, we can see that passenger count is clearly related to the average fare, since as the passenger count increases so does the fare, though there isn't a notable difference in fare for passengers between 2-5. After the passenger count increases past 6, the fare seems to increase exponentially. The trip distance does not seem to have a notable correlation to passenger count or fare. However, trips with a passenger count of 1-2 and 8-9 have a much larger than usual average trip distance, with 2 passengers having the highest average trip distance. In summary, the average fair increases as the passenger count increases, and the passenger count with the highest expectable trip distance is 2 passengers.

From the second table in part 5, we can make the conclusion that the most frequent pickup hours are between 6 and 10 PM, as they are the all the top 5 pick up hours with the highest count of pick ups, and they are all over 540 thousand pick ups, unlike the rest of the hours which are all 480 thousand and smaller in pick up count.

In step 6, from the trip distance line graph we can see that the frequency of trips decreases as the distance increases, or in other words, trips of a small distance are much more frequent (which is not surprising). Trips are the most frequent between a distance of 1-2.1, miles as those distances are all much larger in count than the rest of the distances, and the count approaches 1 as the distance reaches 40 miles.

In the Average Fare Line graph, we can see that the average fare sits between 14 and 15, with it being less than 15 except on hours 4-6, with highest fare being at 5 am, and the cheapest at 7 pm, which makes sense since 7 pm is in the range of most frequent pick up hours from the second table in part 5, we could make the conclusion that the price is lowest at that time since there is the most competition and probably more traffic, so the distance would be less and theres more options so setting a higher fare price would lower the number of potential customers. The opposite also explains the highest rate being at 5 am, since there is likely the fewest taxi's around, they could more easily get away with charging higher fares due to less competition.

PART B: Advanced Prescriptive Analytics

1. Trip Duration:

```
27
    from pyspark.sql.functions import col

    # Calculate the trip duration by subtracting the pickup time from the drop- off time.
    df_cleaned = df_cleaned.withColumn("trip_duration_min",
                                       (col("dropoff_datetime").cast("long") - col("pickup_datetime").cast("long")) / 60)

    df_cleaned.select("pickup_datetime", "dropoff_datetime", "trip_duration_min").show(5)
```

```
+-------------------+-------------------+------------------+
|    pickup_datetime|   dropoff_datetime| trip_duration_min|
+-------------------+-------------------+------------------+
|2015-01-15 19:05:39|2015-01-15 19:23:42|             18.05|
|2015-01-10 20:33:38|2015-01-10 20:53:28|19.833333333333332|
|2015-01-10 20:33:38|2015-01-10 20:43:41|             10.05|
|2015-01-10 20:33:39|2015-01-10 20:52:58|19.316666666666666|
|2015-01-10 20:33:39|2015-01-10 20:53:52|20.216666666666665|
+-------------------+-------------------+------------------+
only showing top 5 rows
```

2. Hour and Day:

```
29
    from pyspark.sql.functions import hour, dayofmonth

    #  Extract hour and day from the pickup_datetime to analyze hourly and daily patterns
    df_cleaned = df_cleaned.withColumn("pickup_hour", hour(col("pickup_datetime")))

    df_cleaned = df_cleaned.withColumn("pickup_day", dayofmonth(col("pickup_datetime")))


    df_cleaned.select("pickup_datetime", "pickup_hour", "pickup_day").show(5)
```

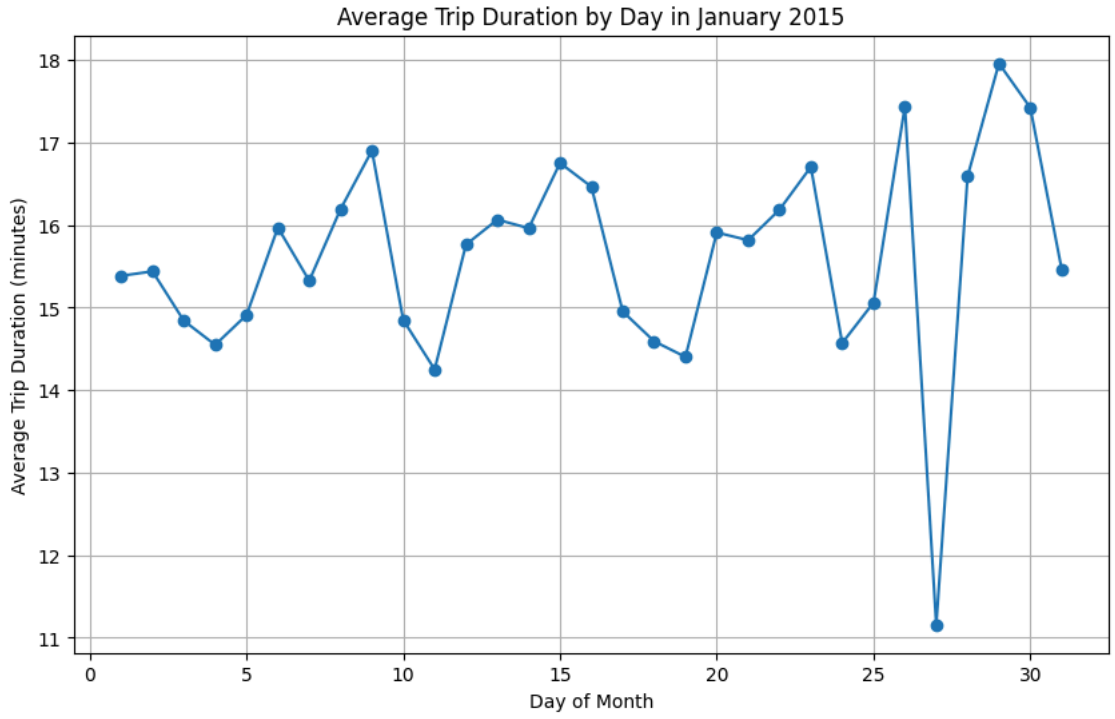
```
+-------------------+-----------+----------+
|    pickup_datetime|pickup_hour|pickup_day|
+-------------------+-----------+----------+
|2015-01-15 19:05:39|         19|        15|
|2015-01-10 20:33:38|         20|        10|
|2015-01-10 20:33:38|         20|        10|
|2015-01-10 20:33:39|         20|        10|
|2015-01-10 20:33:39|         20|        10|
+-------------------+-----------+----------+
only showing top 5 rows
```

3. Trend Over Years: Analyze how the trip duration changes over the years. Plot the results.

```
31
```

```
import matplotlib.pyplot as plt

df_cleaned.select("pickup_datetime").agg({"pickup_datetime": "min", "pickup_datetime": "max"}).show()

#since max pickup date is 31st jan 2015 , we will do how how the trip duration changes over each day of january

from pyspark.sql.functions import dayofmonth

# dayofmonth
df_cleaned = df_cleaned.withColumn("pickup_day", dayofmonth(col("pickup_datetime")))

# grouping by day
df_duration_by_day = df_cleaned.groupBy("pickup_day").agg(avg("trip_duration_min").alias("avg_trip_duration"))
df_duration_by_day = df_duration_by_day.orderBy("pickup_day")

# pandas df
df_duration_by_day_pd = df_duration_by_day.toPandas()

# plot
plt.figure(figsize=(10,6))
plt.plot(df_duration_by_day_pd["pickup_day"], df_duration_by_day_pd["avg_trip_duration"], marker='o')
plt.title("Average Trip Duration by Day in January 2015")
plt.xlabel("Day of Month")
plt.ylabel("Average Trip Duration (minutes)")
plt.grid(True)
plt.show()
```

```
+--------------------+
|max(pickup_datetime)|
+--------------------+
| 2015-01-31 23:59:59|
+--------------------+
```

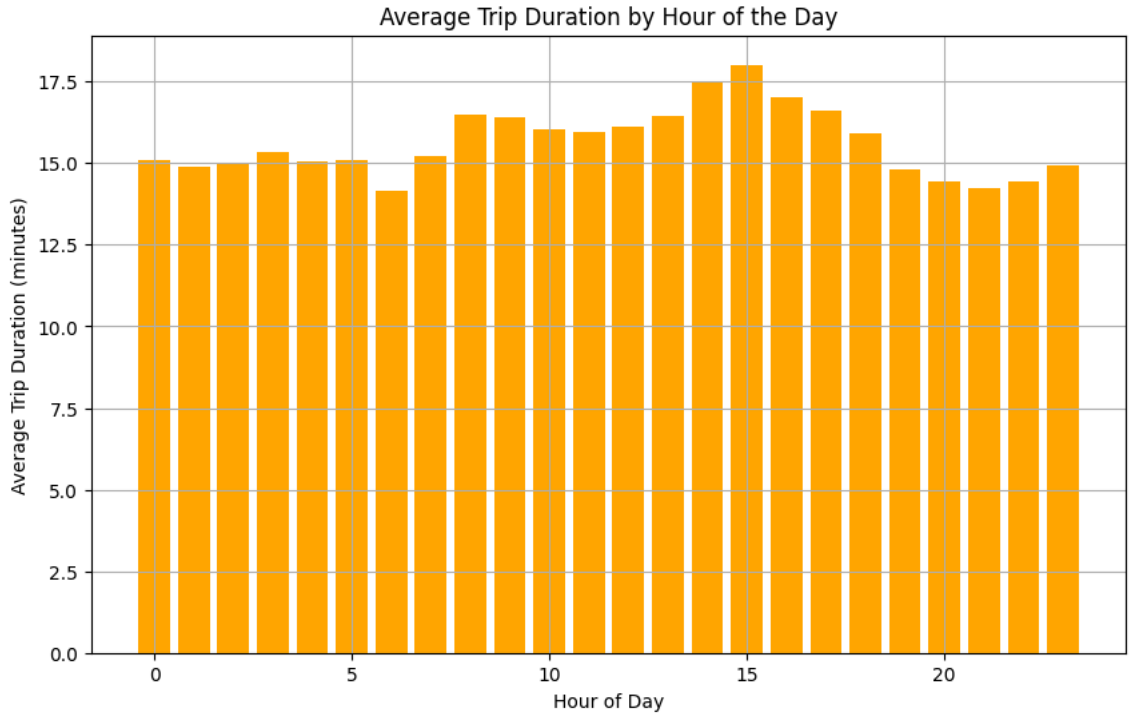


4. Hourly Analysis: Check how the trip duration varies throughout the day. Plot the results

33

```
# Group by hour and calculate the average trip duration
df_duration_by_hour = df_cleaned.groupBy("pickup_hour").agg(avg("trip_duration_min").alias("avg_trip_duration"))

# Order the results by hour
df_duration_by_hour = df_duration_by_hour.orderBy("pickup_hour")

# Show the result
df_duration_by_hour.show()

# Convert the result to Pandas DataFrame for plotting
df_duration_by_hour_pd = df_duration_by_hour.toPandas()

# Plot the average trip duration by hour of the day
plt.figure(figsize=(10,6))
plt.bar(df_duration_by_hour_pd["pickup_hour"], df_duration_by_hour_pd["avg_trip_duration"], color='orange')
plt.title("Average Trip Duration by Hour of the Day")
plt.xlabel("Hour of Day")
plt.ylabel("Average Trip Duration (minutes)")
plt.grid(True)
plt.show()
```

```
+-----------+------------------+
|pickup_hour| avg_trip_duration|
+-----------+------------------+
|          0|15.094587482838069|
|          1|14.884602197683598|
|          2|15.016721822407833|
|          3|15.335224492536504|
|          4|15.069514610764612|
|          5|15.088893829183478|
|          6|14.159144579124913|
|          7|15.208690064362195|
|          8| 16.47289180623437|
|          9|16.393476600395424|
|         10|16.045915407995178|
|         11| 15.94066714155104|
|         12|16.122697679369175|
|         13| 16.43576062592638|
|         14| 17.52447918217457|
|         15| 18.01168547529826|
|         16| 17.01756547466589|
|         17|16.594882078845707|
|         18|15.898487607096948|
|         19|14.822664755059892|
+-----------+------------------+
only showing top 20 rows
```

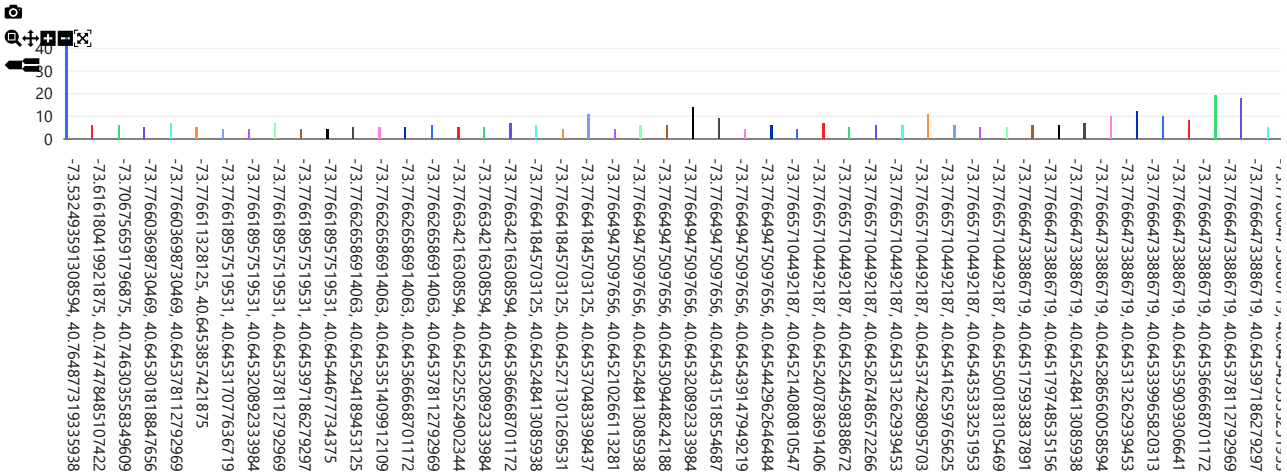


5.Identify Hotspots:

35

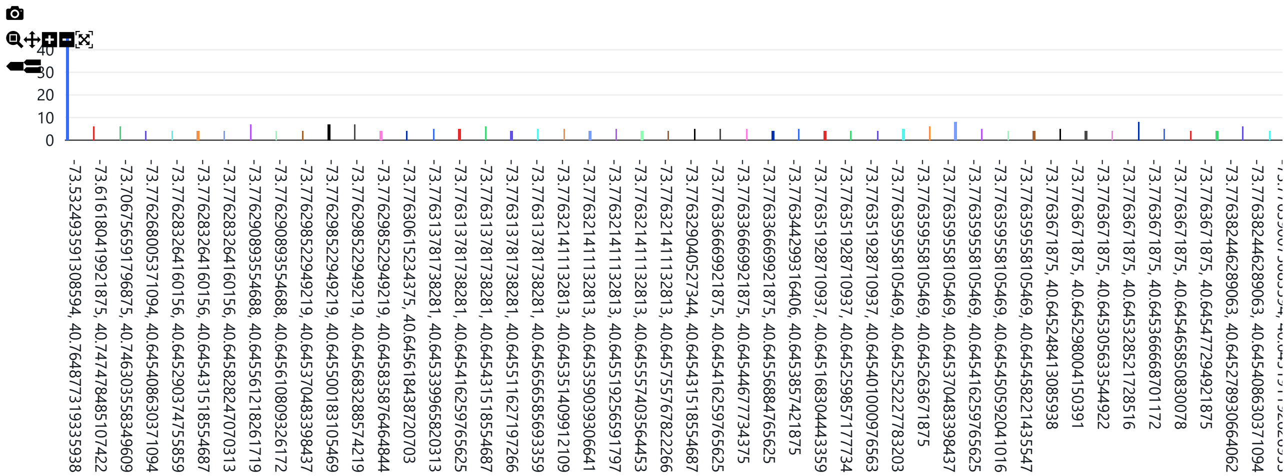
```
from pyspark.sql.functions import count

# Create 'pickup_location' and 'dropoff_location' in new dataframe
df_hotspots = df_cleaned.select("pickup_longitude", "pickup_latitude", "dropoff_longitude", "dropoff_latitude").toPandas()
df_hotspots['pickup_location'] = df_hotspots.apply( lambda row: f"{row['pickup_longitude']}, {row['pickup_latitude']}", axis=1)
df_hotspots['dropoff_location'] = df_hotspots.apply( lambda row: f"{row['dropoff_longitude']}, {row['dropoff_latitude']}", axis=1)

# Display Pickup hotspots
pickup_hotspots = df_hotspots.groupby('pickup_location').size().reset_index(name='trip_count').sort_values(by='trip_count', ascendi
display(pickup_hotspots)

# Display dropoff hotspots
dropoff_hotspots = df_hotspots.groupby('dropoff_location').size().reset_index(name='trip_count').sort_values(by='trip_count', ascen
display(dropoff_hotspots)
```

Table Pickup Hotspots



50+ rows | Truncated data

Table | Dropoff Hotspots



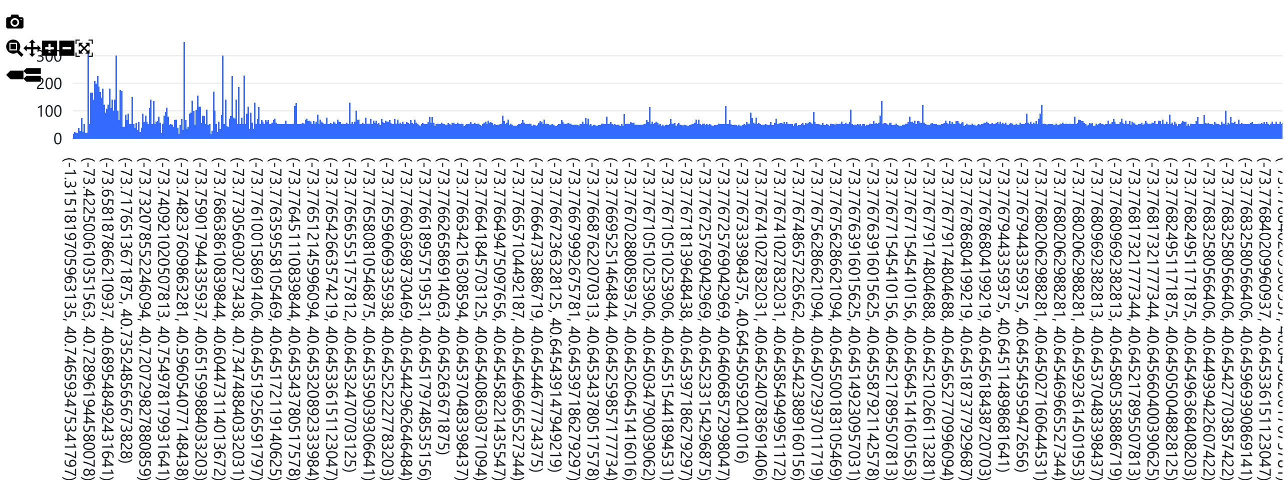
50+ rows | Truncated data

6. Average Fair by Location:

37

```
import pandas as pd
# Create 'pickup_location'
df_locations = df_cleaned.select("pickup_longitude", "pickup_latitude", "fare_amount").toPandas()
df_locations['pickup_location'] = df_locations.apply(lambda row: f"({row['pickup_longitude']}, {row['pickup_latitude']})", axis=1)
df_locations['fare_amount'] = pd.to_numeric(df_locations['fare_amount'], errors='coerce')

# Find average fare based on pickup location
avg_fare_by_location = df_locations.groupby('pickup_location')['fare_amount'].mean().reset_index()
avg_fare_by_location.columns = ['Pickup Location', 'Average Fare Amount']

# Display results
display(avg_fare_by_location)
```

Table | Average Fare by Location



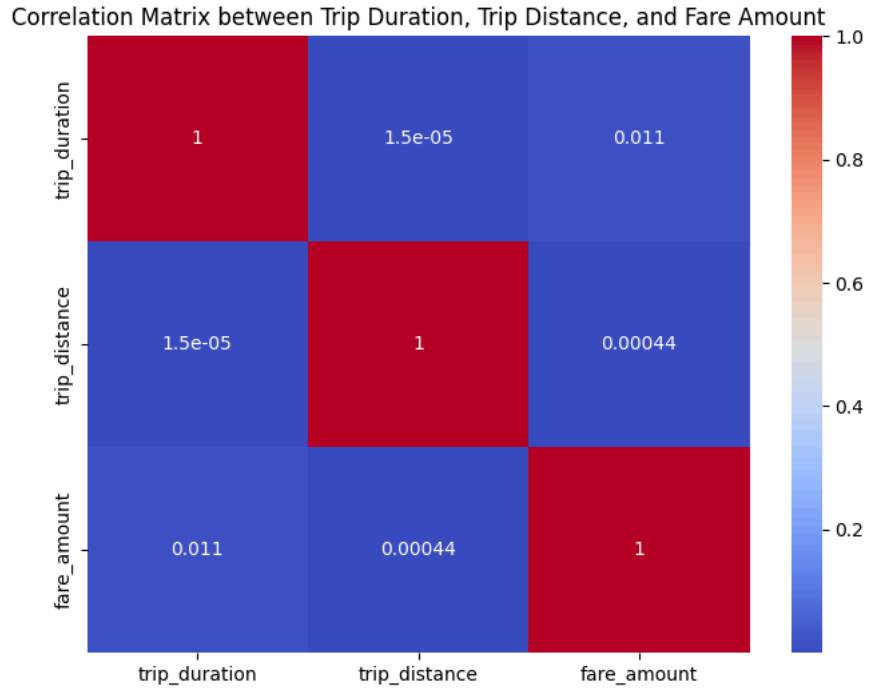
10,000+ rows | Truncated data

7. Correlation Analysis

39

```
# Calculate trip duration in minutes
df = df.withColumn('trip_duration',
                   ((col('tpep_dropoff_datetime').cast('long') - col('tpep_pickup_datetime').cast('long')) / 60).cast('double'))

# Convert columns to double type for consistency
df = df.withColumn('trip_distance', col('trip_distance').cast('double'))
df = df.withColumn('fare_amount', col('fare_amount').cast('double'))

# Filter out rows with null values in key columns
df = df.filter((col('trip_duration').isNotNull()) &
               (col('trip_distance').isNotNull()) &
               (col('fare_amount').isNotNull()))

# Calculate correlations
correlation = df.stat.corr('trip_duration', 'trip_distance')
fare_correlation = df.stat.corr('trip_duration', 'fare_amount')

# Print correlation results
print(f"Correlation between trip duration and trip distance: {correlation}")
print(f"Correlation between trip duration and fare amount: {fare_correlation}")

# Create a pandas DataFrame and correlation matrix
corr_df = df.select('trip_duration', 'trip_distance', 'fare_amount').toPandas()
corr_matrix = corr_df.corr()

# Plot the correlation matrix heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix between Trip Duration, Trip Distance, and Fare Amount')
plt.show()
```

```
Correlation between trip duration and trip distance: 1.5306018579757404e-05
Correlation between trip duration and fare amount: 0.011440083866136857
```

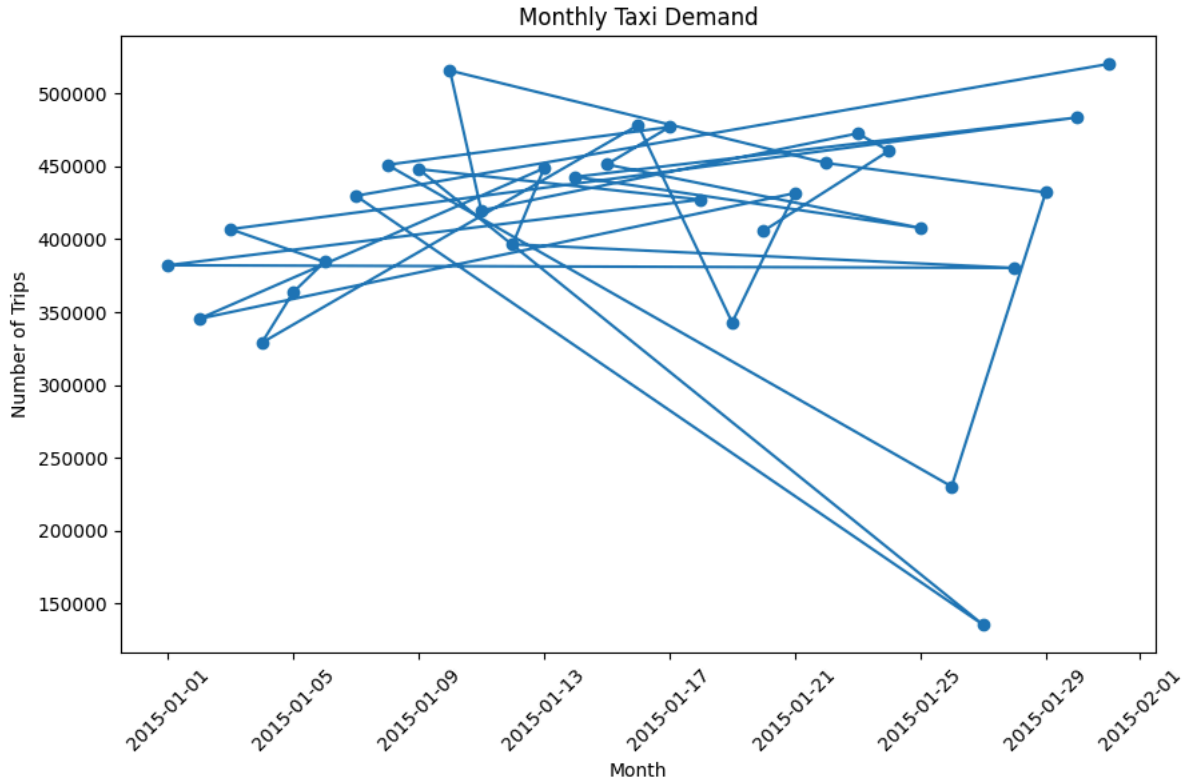


The trip duration and fare amount have a correlation (since how long the trip takes will influence the fare). The fare amount also has a correlation to trip distance (which also makes sense for the same reason. Trip distance and trip duration have the least correlation (which is likely influenced by traffic).

8. Examine taxi demand

42

```
# Cast pickup datetime column to timestamp type
df = df.withColumn('tpep_pickup_datetime', df['tpep_pickup_datetime'].cast('timestamp'))

# Extract the pickup month (date) from the pickup datetime
df = df.withColumn('pickup_month', df['tpep_pickup_datetime'].cast('date'))

# Group by pickup month and count the number of trips per month
monthly_trips = df.groupby('pickup_month').count()

# Convert the grouped data to pandas for plotting
plt.figure(figsize=(10, 6))
monthly_trips_df = monthly_trips.toPandas()

# Plot the number of trips by month
plt.plot(monthly_trips_df['pickup_month'], monthly_trips_df['count'], marker='o')
plt.title('Monthly Taxi Demand')
plt.xlabel('Month')
plt.ylabel('Number of Trips')
plt.xticks(rotation=45)
plt.show()
```



Taxi demand appears to be at its highest near the middle of the month (days 14-18), and in the last day of the month, and seems low at start of the month (days 1-5), and at its lowest on the 27th.