uber-data-visualization

May 16, 2023

1 @**CodeClause Project : Uber Data Analysis (Visualization)**

1.1 Summary To Explain Project (Keypoints)

- Import Modules
- Load Dataset
- Data Preparation
- Visualization
  1. Number of trips by hour
  2. Number of trips by month
  3. Analysis of Week Day and Running Day
  4. Ratio of the increase from August to September
  5. Number of trips by weekday
  6. Lowest number of trips by weekday
  7. Trips Ratio Working Days and Weekends
  8. Number of trips by day
  9. Number of trips by hour and month
  10. Trips by Hour and Weekday

1.1.1 1. Import Modules

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

%matplotlib inline
```

1.1.2 2. Load Dataset

[2]:
```
#Load the datasets
try:
    df_apr14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-apr14.csv",error_bad_lines=False,engine="python")
    df_may14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-may14.csv",error_bad_lines=False,engine="python")
    df_jun14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-jun14.csv",error_bad_lines=False,engine="python")
```

```
    df_jul14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-jul14.csv",error_bad_lines=False,engine="python")
    df_aug14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-aug14.csv",error_bad_lines=False,engine="python")
    df_sep14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/
 ↪uber-raw-data-sep14.csv",error_bad_lines=False,engine="python")
except pd.errors.ParserError as e:
    print("Error occurred while parsing CSV:", e)

#Merge the dataframes into one

df = df_apr14.append([df_may14,df_jun14,df_jul14,df_aug14,df_sep14],␣
 ↪ignore_index=True)
```

```
<ipython-input-2-11b64f844646>:3: FutureWarning: The error_bad_lines argument
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_apr14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-apr14.csv",error_bad_lines=False,engine="python")
Skipping line 68535: unexpected end of data
<ipython-input-2-11b64f844646>:4: FutureWarning: The error_bad_lines argument
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_may14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-may14.csv",error_bad_lines=False,engine="python")
Skipping line 91516: unexpected end of data
<ipython-input-2-11b64f844646>:5: FutureWarning: The error_bad_lines argument
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_jun14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-jun14.csv",error_bad_lines=False,engine="python")
Skipping line 68622: unexpected end of data
<ipython-input-2-11b64f844646>:6: FutureWarning: The error_bad_lines argument
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_jul14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-jul14.csv",error_bad_lines=False,engine="python")
Skipping line 68469: unexpected end of data
<ipython-input-2-11b64f844646>:7: FutureWarning: The error_bad_lines argument
```

```
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_aug14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-aug14.csv",error_bad_lines=False,engine="python")
Skipping line 68616: unexpected end of data
<ipython-input-2-11b64f844646>:8: FutureWarning: The error_bad_lines argument
has been deprecated and will be removed in a future version. Use on_bad_lines in
the future.


df_sep14=pd.read_csv("/content/drive/MyDrive/MyDataSet/Uber_Visialization/uber-
raw-data-sep14.csv",error_bad_lines=False,engine="python")
Skipping line 68583: unexpected end of data
<ipython-input-2-11b64f844646>:14: FutureWarning: The frame.append method is
deprecated and will be removed from pandas in a future version. Use
pandas.concat instead.
  df = df_apr14.append([df_may14,df_jun14,df_jul14,df_aug14,df_sep14],
ignore_index=True)
```

1.1.3 3. Data Preparation

[3]:
```
df.head()
```

[3]:
```
            Date/Time      Lat      Lon    Base
0  4/1/2014 0:11:00  40.7690 -73.9549  B02512
1  4/1/2014 0:17:00  40.7267 -74.0345  B02512
2  4/1/2014 0:21:00  40.7316 -73.9873  B02512
3  4/1/2014 0:28:00  40.7588 -73.9776  B02512
4  4/1/2014 0:33:00  40.7594 -73.9722  B02512
```

[4]:
```
df.shape
```

[4]:
```
(434329, 4)
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 434329 entries, 0 to 434328
Data columns (total 4 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   Date/Time  434329 non-null  object
 1   Lat        434329 non-null  float64
 2   Lon        434329 non-null  float64
 3   Base       434329 non-null  object
```

```
dtypes: float64(2), object(2)
memory usage: 13.3+ MB
```

[6]:
```
df.describe()
```

[6]:
```
                 Lat            Lon
count  434329.000000  434329.000000
mean       40.741484     -73.975492
std         0.042399       0.061789
min        39.656900     -74.703900
25%        40.723500     -73.997500
50%        40.744700     -73.983800
75%        40.762500     -73.968100
max        41.373000     -72.299900
```

[7]:
```
#Renaming the Date/Time Colomn
df = df.rename(columns={'Date/Time': 'Date_time'})

#Converting the Date_time type into Datetime
df['Date_time'] = pd.to_datetime(df['Date_time'])

#Adding usufull colomns
df['Month'] = df['Date_time'].dt.month_name()
df['Weekday'] = df['Date_time'].dt.day_name()
df['Day'] = df['Date_time'].dt.day
df['Hour'] = df['Date_time'].dt.hour
df['Minute'] = df['Date_time'].dt.minute
df['weekno']=df['Date_time'].dt.weekofyear - 13
```

```
<ipython-input-7-13cce4626a62>:13: FutureWarning: Series.dt.weekofyear and
Series.dt.week have been deprecated. Please use Series.dt.isocalendar().week
instead.
  df['weekno']=df['Date_time'].dt.weekofyear - 13
```

[8]:
```
df.head()
```

[8]:
```
            Date_time      Lat      Lon    Base  Month  Weekday  Day  Hour  \
0 2014-04-01 00:11:00  40.7690 -73.9549  B02512  April  Tuesday    1     0
1 2014-04-01 00:17:00  40.7267 -74.0345  B02512  April  Tuesday    1     0
2 2014-04-01 00:21:00  40.7316 -73.9873  B02512  April  Tuesday    1     0
3 2014-04-01 00:28:00  40.7588 -73.9776  B02512  April  Tuesday    1     0
4 2014-04-01 00:33:00  40.7594 -73.9722  B02512  April  Tuesday    1     0

   Minute  weekno
0      11       1
1      17       1
2      21       1
3      28       1
```

```
4       33        1
```

[9]:
```
df.isnull().sum()
```

[9]:
```
Date_time    0
Lat          0
Lon          0
Base         0
Month        0
Weekday      0
Day          0
Hour         0
Minute       0
weekno       0
dtype: int64
```

[10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 434329 entries, 0 to 434328
Data columns (total 10 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   Date_time  434329 non-null  datetime64[ns]
 1   Lat        434329 non-null  float64
 2   Lon        434329 non-null  float64
 3   Base       434329 non-null  object
 4   Month      434329 non-null  object
 5   Weekday    434329 non-null  object
 6   Day        434329 non-null  int64
 7   Hour       434329 non-null  int64
 8   Minute     434329 non-null  int64
 9   weekno     434329 non-null  int64
dtypes: datetime64[ns](1), float64(2), int64(4), object(3)
memory usage: 33.1+ MB
```

[11]:
```
df.describe(include = 'all')
```

```
<ipython-input-11-74aa2f970831>:1: FutureWarning: Treating datetime data as
categorical rather than numeric in `.describe` is deprecated and will be removed
in a future version of pandas. Specify `datetime_is_numeric=True` to silence
this warning and adopt the future behavior now.
  df.describe(include = 'all')
```

[11]:
```
                  Date_time            Lat            Lon    Base   Month  \
count                434329  434329.000000  434329.000000  434329  434329
unique               147306            NaN            NaN       2       6
```

```
top     2014-07-02 18:26:00            NaN            NaN  B02598     May
freq                     33            NaN            NaN  228656   91514
first   2014-04-01 00:00:00            NaN            NaN     NaN     NaN
last    2014-09-30 22:59:00            NaN            NaN     NaN     NaN
mean                    NaN      40.741484     -73.975492     NaN     NaN
std                     NaN       0.042399       0.061789     NaN     NaN
min                     NaN      39.656900     -74.703900     NaN     NaN
25%                     NaN      40.723500     -73.997500     NaN     NaN
50%                     NaN      40.744700     -73.983800     NaN     NaN
75%                     NaN      40.762500     -73.968100     NaN     NaN
max                     NaN      41.373000     -72.299900     NaN     NaN

        Weekday            Day           Hour         Minute         weekno
count    434329  434329.000000  434329.000000  434329.000000  434329.000000
unique        7            NaN            NaN            NaN            NaN
top     Tuesday            NaN            NaN            NaN            NaN
freq      72549            NaN            NaN            NaN            NaN
first       NaN            NaN            NaN            NaN            NaN
last        NaN            NaN            NaN            NaN            NaN
mean        NaN       9.018081      14.170981      29.407636      12.415837
std         NaN       8.631915       5.747129      17.320751       7.508438
min         NaN       1.000000       0.000000       0.000000       1.000000
25%         NaN       3.000000      10.000000      14.000000       6.000000
50%         NaN       5.000000      15.000000      29.000000      12.000000
75%         NaN      14.000000      19.000000      44.000000      19.000000
max         NaN      31.000000      23.000000      59.000000      27.000000
```

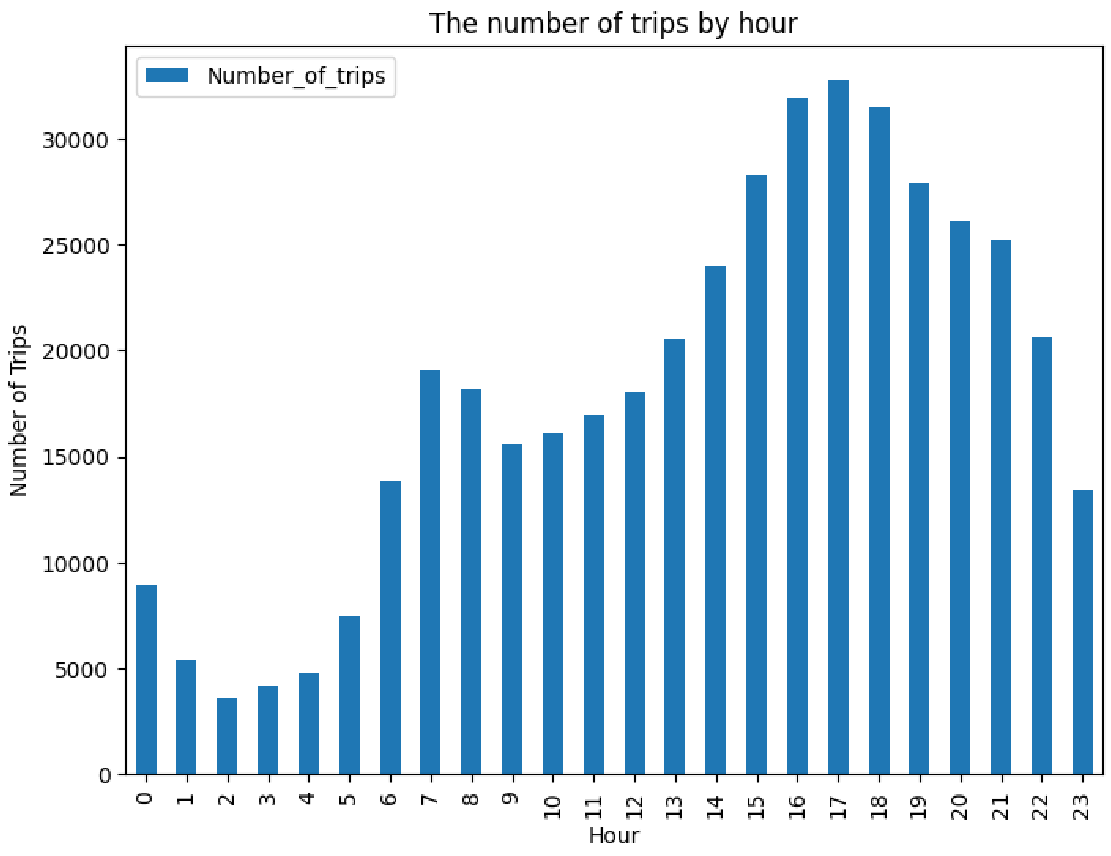
1.2 4 ——— Visualization ———-

1.2.1 4.1 Number of trips by hour

[12]:
```python
df_hour_grouped = df.groupby(['Hour']).count()
df_hour = pd.DataFrame({'Number_of_trips':df_hour_grouped.values[:,0]}, index =␣
  ↪df_hour_grouped.index)
df_hour.head()
```

[12]:
```
      Number_of_trips
Hour
0                8924
1                5381
2                3584
3                4144
4                4750
```

[13]:
```python
df_hour.plot(kind='bar', figsize=(8,6))
plt.ylabel('Number of Trips')
plt.title('The number of trips by hour')
plt.show()
```

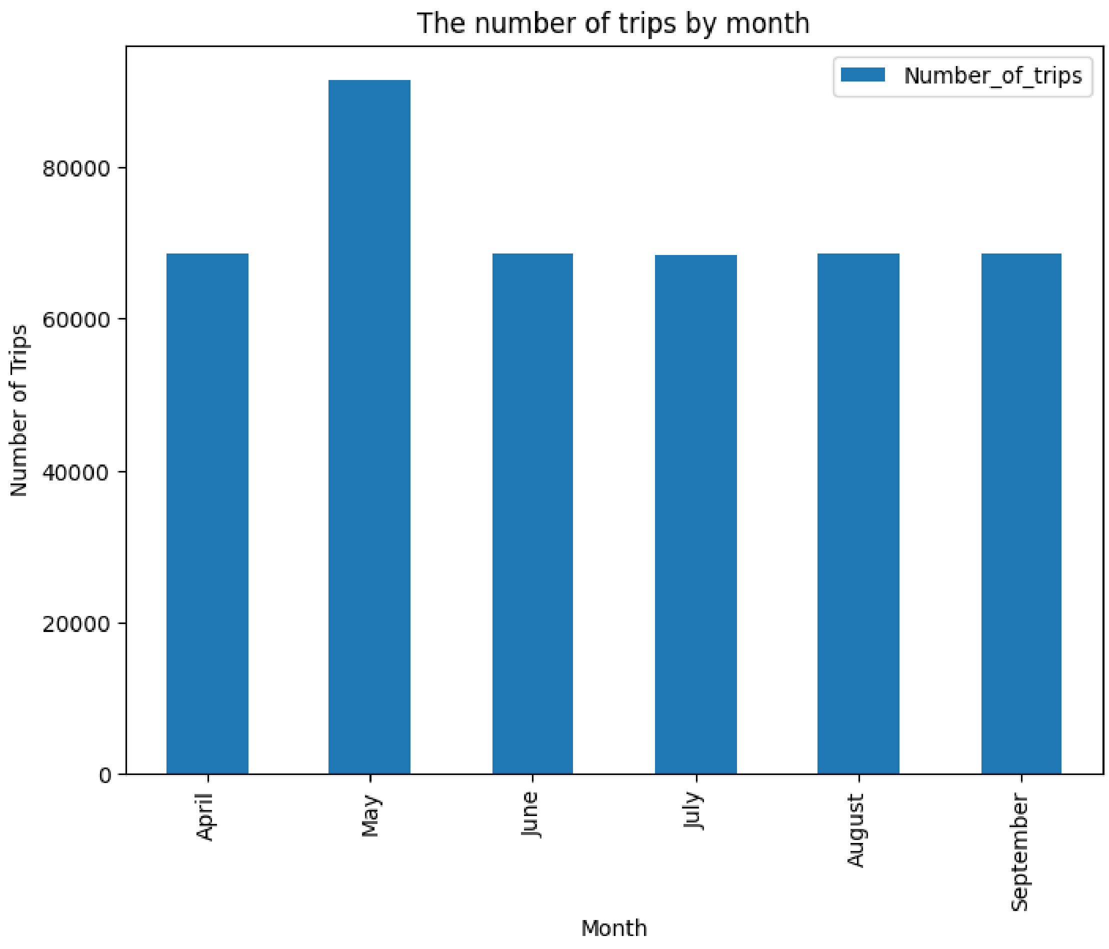


1.2.2 4.2 Number of trips by month

[14]:
```
df_month_grouped = df.groupby(['Month'], sort=False).count()
df_month = pd.DataFrame({'Number_of_trips':df_month_grouped.values[:,0]}, index␣
↪= df_month_grouped.index)
df_month
```

[14]:
```
           Number_of_trips
Month
April                68533
May                  91514
June                 68620
July                 68467
August               68614
September            68581
```

[15]:
```
df_month.plot(kind='bar', figsize=(8,6))
plt.ylabel('Number of Trips')
plt.title('The number of trips by month')
plt.show()
```



1.2.3 4.3 Analysis of Week Day and Running Day

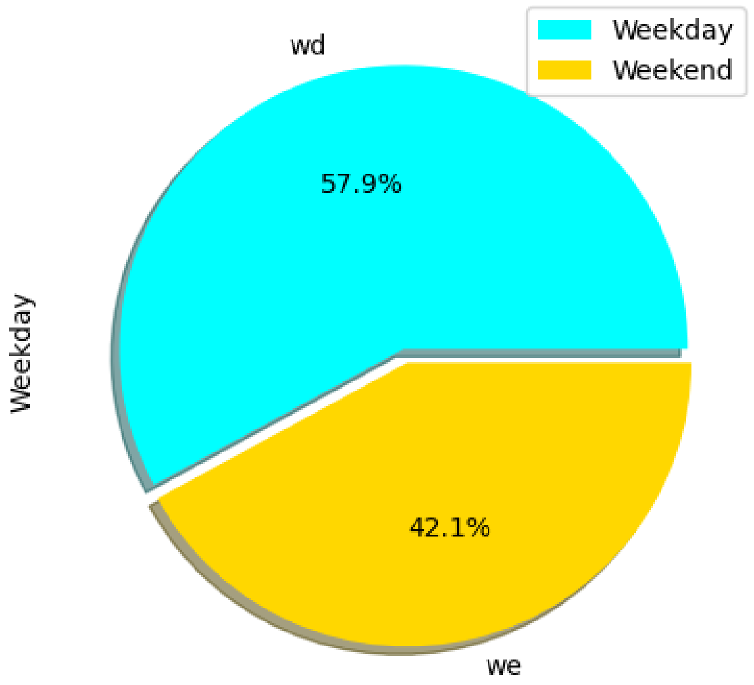
[16]:
```
week_day=pd.DataFrame(df['Weekday'].value_counts())

week_day['day_type']=['wd','wd','wd','wd','we','wd','we']
```

[17]:
```
pie=week_day.groupby('day_type').sum()
pie.iloc[0,0]=pie.iloc[0,0]/5
pie.iloc[1,0]=pie.iloc[1,0]/2
pie
```

[17]:
```
          Weekday
day_type
wd          67315
we          48877
```

[18]:
```
explode=[0,0.05]
colors=['cyan','gold']
labels=['Weekday','Weekend']
pie.plot.pie(autopct = '%1.1f%%',shadow=True,subplots=True,
        colors=colors,explode=explode)
plt.legend(labels=labels)
plt.show()
```



1.2.4 4.4 Ratio of the increase from August to September

[19]:
```
number_of_trips_may = df_month.loc['May'].values
number_of_trips_sep = df_month.loc['September'].values

ratio_month = (((number_of_trips_may - number_of_trips_sep) /␣
 ↪number_of_trips_may) * 100)[0]
ratio_month = round(ratio_month)
```

```
print('The ratio of the increase from August to September is {} %.'.
  ↪format(ratio_month))
```

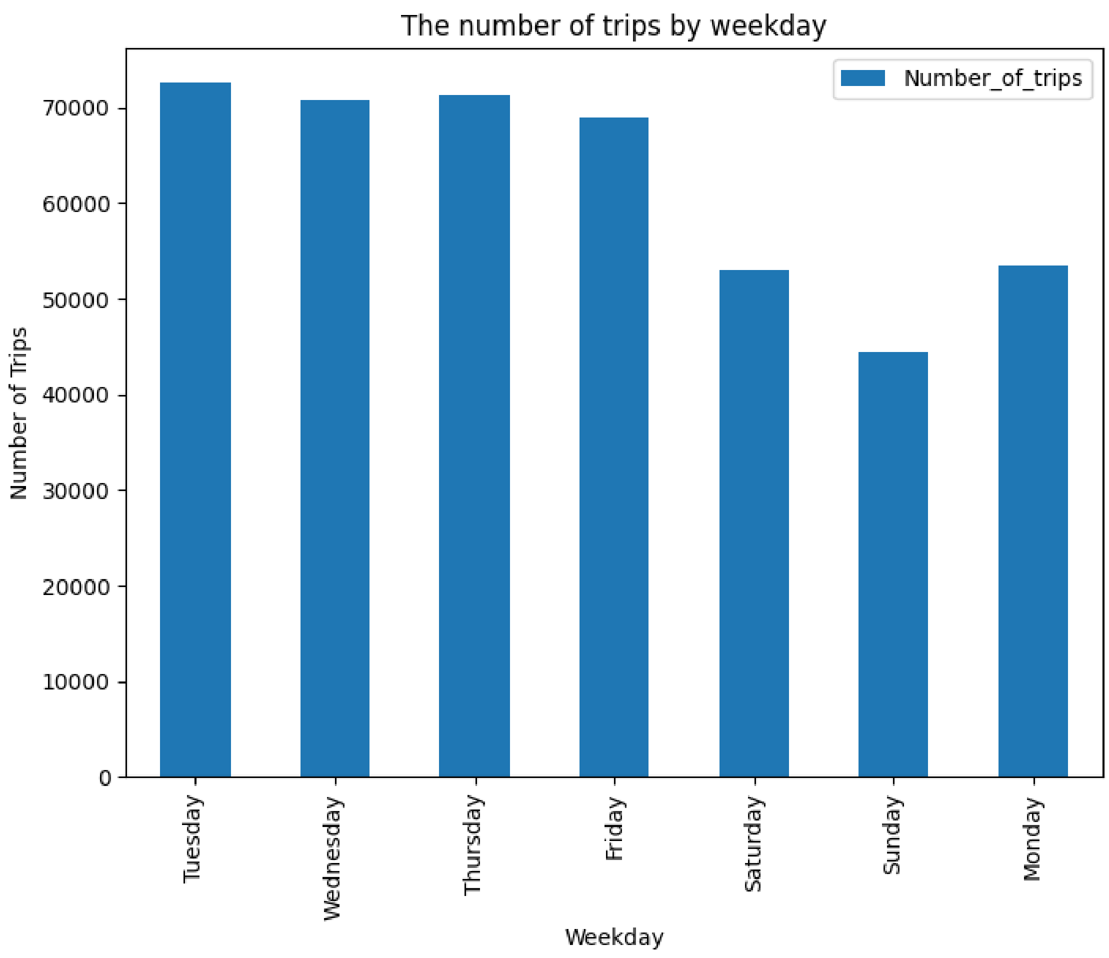
```
The ratio of the increase from August to September is 25 %.
```

1.2.5 4.5 Number of trips by weekday

[20]:
```
df_weekday_grouped = df.groupby(['Weekday'], sort = False).count()
df_weekday = pd.DataFrame({'Number_of_trips':df_weekday_grouped.values[:,0]},␣
  ↪index = df_weekday_grouped.index)
df_weekday
```

[20]:
```
           Number_of_trips
Weekday
Tuesday              72549
Wednesday            70827
Thursday             71312
Friday               68895
Saturday             52992
Sunday               44345
Monday               53409
```

[21]:
```
df_weekday.plot(kind='bar', figsize=(8,6))
plt.ylabel('Number of Trips')
plt.title('The number of trips by weekday')
plt.show()
```



1.2.6 4.6 Lowest number of trips by weekday

[22]:
```
min_number_of_trips_weekday = min(df_weekday['Number_of_trips'])
min_weekday = df_weekday[df_weekday['Number_of_trips'] ==␣
 ↪min_number_of_trips_weekday].index[0]
print('The lowest number of trips by weekday is {} trip, that corresponds to {}.
 ↪'.format(min_number_of_trips_weekday, min_weekday))
```

```
The lowest number of trips by weekday is 44345 trip, that corresponds to Sunday.
```

1.2.7 4.7 Trips Ratio Working Days and Weekends

[23]:
```
mean_number_of_trips_weekend = ((df_weekday.loc['Saturday'] + df_weekday.
 ↪loc['Sunday']) / 2).values
mean_number_of_trips_workday = (((df_weekday.loc['Monday'] + df_weekday.
 ↪loc['Tuesday'] + df_weekday.loc['Wednesday'] + df_weekday.loc['Thursday'] +␣
 ↪df_weekday.loc['Friday'])/ 5).values)[0]
```

```
ratio_weekday = (((mean_number_of_trips_workday - mean_number_of_trips_weekend)␣
↪/ mean_number_of_trips_weekend) * 100)[0]
ratio_weekday = round(ratio_weekday, 1)
print('The mean number of trips during working days is {}% higher than the mean␣
↪number of trips during weekends.'.format(ratio_weekday))
```

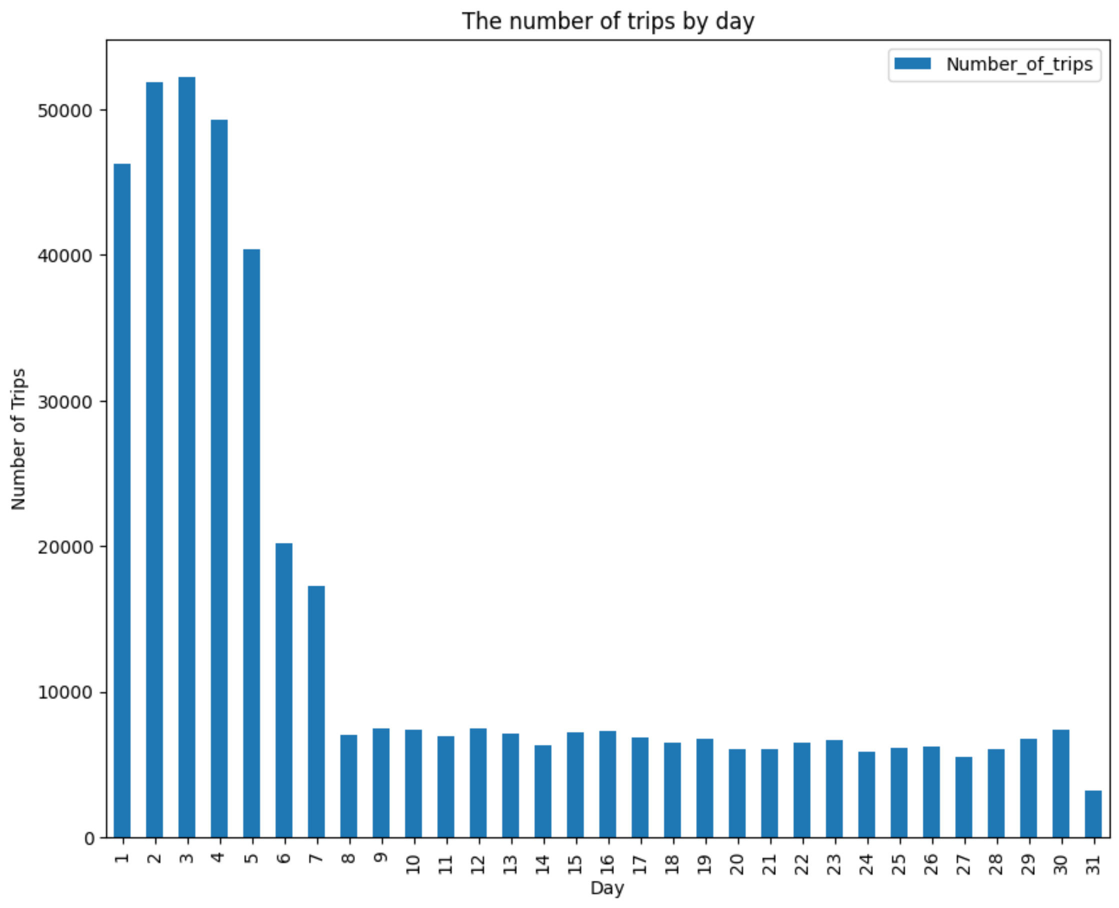
```
The mean number of trips during working days is 38.5% higher than the mean
number of trips during weekends.
```

1.2.8 4.8 Number of trips by day

[24]:
```
df_day_grouped = df.groupby(['Day']).count()
df_day = pd.DataFrame({'Number_of_trips':df_day_grouped.values[:,0]}, index =␣
↪df_day_grouped.index)
df_day.head()
```

[24]:
```
     Number_of_trips
Day
1              46267
2              51907
3              52199
4              49294
5              40404
```

[25]:
```
df_day.plot(kind='bar', figsize=(10,8))
plt.ylabel('Number of Trips')
plt.title('The number of trips by day')
plt.show()
```



1.2.9 4.9 Number of trips by hour and month

[26]:
```
df_hour_month_grouped = df.groupby(['Hour','Month']).count()
df_hour_month = pd.DataFrame({'Number_of_trips':df_hour_month_grouped.values[:
  ↪,1]}, index = df_hour_month_grouped.index)
df_hour_month.head(10)
```

[26]:
```
                 Number_of_trips
Hour Month
0    April                  1477
     August                 1717
     July                   1294
     June                   1313
     May                    1904
     September              1219
1    April                   877
     August                 1136
     July                    824
     June                    775
```

[27]:
```
df_hour_month.reset_index(inplace= True)
df_hour_month.head()
```

[27]:
```
   Hour   Month  Number_of_trips
0     0   April             1477
1     0  August             1717
2     0    July             1294
3     0    June             1313
4     0     May             1904
```

[28]:
```
data_hour_month = df_hour_month['Number_of_trips'].values.reshape(24,6)
data_hour_month
```

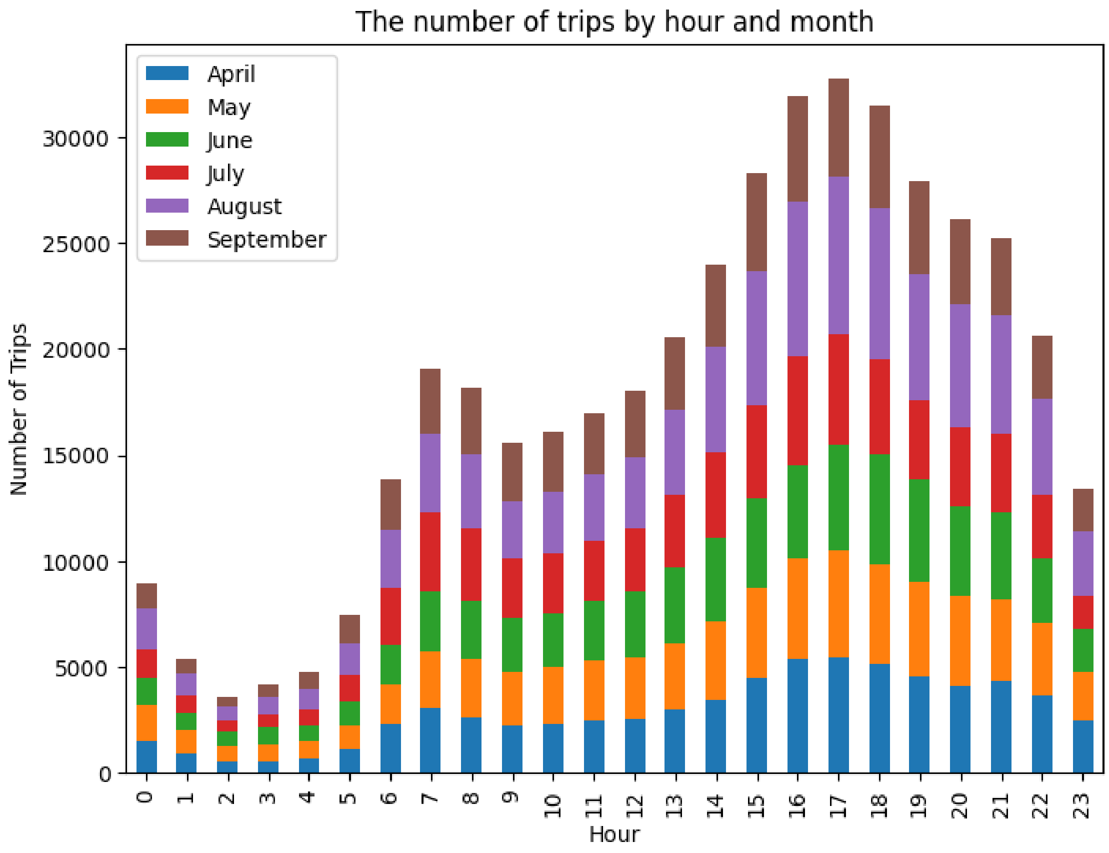
[28]:
```
array([[1477, 1717, 1294, 1313, 1904, 1219],
       [ 877, 1136,  824,  775, 1097,  672],
       [ 518,  752,  668,  533,  683,  430],
       [ 539,  824,  773,  603,  835,  570],
       [ 694,  827,  731,  753,  941,  804],
       [1082, 1176, 1073, 1300, 1483, 1351],
       [2282, 1889, 1834, 2691, 2799, 2331],
       [3041, 2701, 2823, 3695, 3730, 3076],
       [2622, 2745, 2753, 3427, 3457, 3189],
       [2221, 2565, 2491, 2815, 2729, 2738],
       [2313, 2665, 2527, 2828, 2951, 2776],
       [2486, 2799, 2806, 2817, 3165, 2928],
       [2565, 2860, 3137, 2960, 3382, 3107],
       [2971, 3151, 3569, 3404, 4056, 3402],
       [3443, 3696, 3985, 3967, 4991, 3918],
       [4458, 4222, 4286, 4388, 6343, 4597],
       [5345, 4754, 4454, 5067, 7298, 4990],
       [5437, 5075, 4971, 5212, 7473, 4593],
       [5100, 4745, 5211, 4421, 7210, 4837],
       [4550, 4490, 4814, 3682, 5983, 4364],
       [4127, 4210, 4240, 3750, 5780, 4046],
       [4332, 3889, 4051, 3718, 5607, 3668],
       [3620, 3421, 3111, 2922, 4549, 3001],
       [2433, 2305, 2041, 1579, 3068, 1974]])
```

[29]:
```
df_hour_month = pd.DataFrame(data = data_hour_month, index =␣
 ↪df_hour_month['Hour'].unique(), columns = df['Month'].unique())
df_hour_month.head()
```

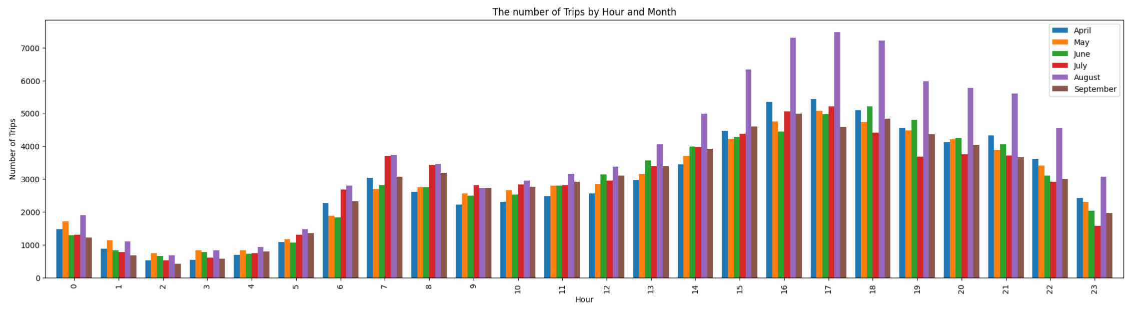
[29]:
```
   April   May  June  July  August  September
0   1477  1717  1294  1313    1904       1219
1    877  1136   824   775    1097        672
```

```
2     518    752    668    533      683          430
3     539    824    773    603      835          570
4     694    827    731    753      941          804
```

[30]:
```
df_hour_month.plot(kind='bar', figsize=(8,6), stacked=True)
plt.xlabel('Hour')
plt.ylabel('Number of Trips')
plt.title('The number of trips by hour and month')
plt.show()
```



[31]:
```
df_hour_month.plot(kind='bar', figsize=(25,6),width=0.8)
plt.xlabel('Hour')
plt.ylabel('Number of Trips')
plt.title('The number of Trips by Hour and Month')
plt.show()
```



1.2.10 4.10 Trips by Hour and Weekday

[32]:
```
df_weekday_hour_grouped = df.groupby(['Weekday','Hour'], sort = False).count()
df_weekday_hour = pd.DataFrame({'Number_of_trips':df_weekday_hour_grouped.
 ↪values[:,1]}, index = df_weekday_hour_grouped.index)
df_weekday_hour
```

[32]:
```
                Number_of_trips
Weekday Hour
Tuesday 0                   615
        1                   351
        2                   231
        3                   483
        4                   775
…                         …
Monday  19                 3268
        20                 3101
        21                 2691
        22                 1832
        23                 1007

[168 rows x 1 columns]
```

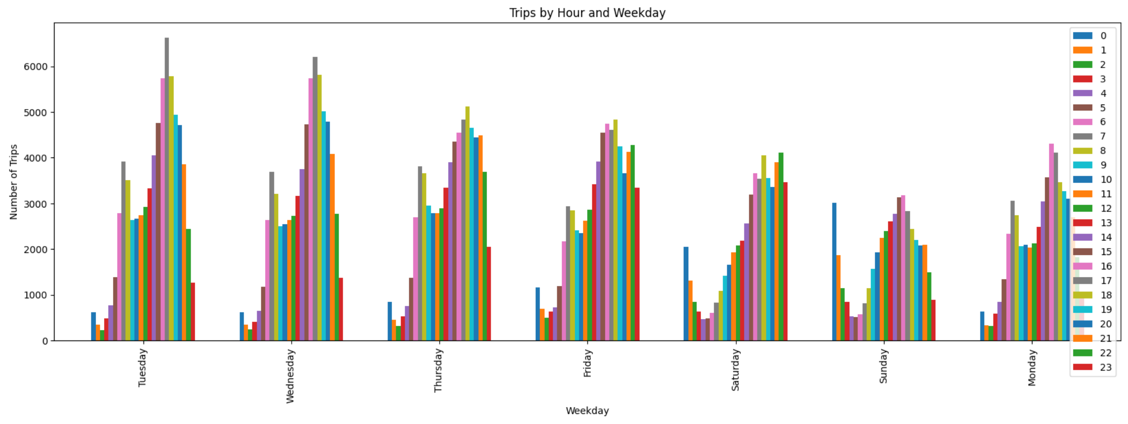
[33]:
```
df_weekday_hour.reset_index(inplace= True)
data_weekday_hour = df_weekday_hour['Number_of_trips'].values.reshape(7,24)
df_weekday_hour = pd.DataFrame(data = data_weekday_hour, index =␣
 ↪df_weekday_hour['Weekday'].unique(), columns = df['Hour'].unique())
df_weekday_hour.head()
```

[33]:
```
              0     1    2    3    4     5     6     7     8     9  …    14  \
Tuesday     615   351  231  483  775  1385  2791  3920  3506  2644  …  4056
Wednesday   615   345  246  411  644  1180  2637  3690  3210  2496  …  3753
Thursday    841   459  315  536  754  1380  2700  3814  3657  2949  …  3896
Friday     1163   701  494  641  732  1189  2178  2943  2844  2419  …  3917
Saturday   2043  1318  839  632  468   478   606   826  1091  1413  …  2561
```

```
              15    16    17    18    19    20    21    22    23
Tuesday     4764  5731  6623  5783  4933  4711  3863  2443  1263
Wednesday   4736  5733  6206  5813  5010  4796  4082  2770  1371
Thursday    4351  4544  4837  5128  4662  4441  4483  3693  2051
Friday      4549  4749  4605  4839  4249  3662  4135  4276  3350
Saturday    3189  3658  3543  4059  3560  3367  3909  4110  3466

[5 rows x 24 columns]
```

[34]:
```python
df_weekday_hour.plot(kind='bar', figsize=(20,6), width = 0.7)
plt.xlabel('Weekday')
plt.ylabel('Number of Trips')
plt.title('Trips by Hour and Weekday')
plt.show()
```



[35]:
```python
df_month_weekday_grouped = df.groupby(['Month','Weekday'], sort=False).count()
df_month_weekday = pd.DataFrame({'Number_of_trips':df_month_weekday_grouped.
  ↪values[:,1]}, index = df_month_weekday_grouped.index)
df_month_weekday.head(10)
```

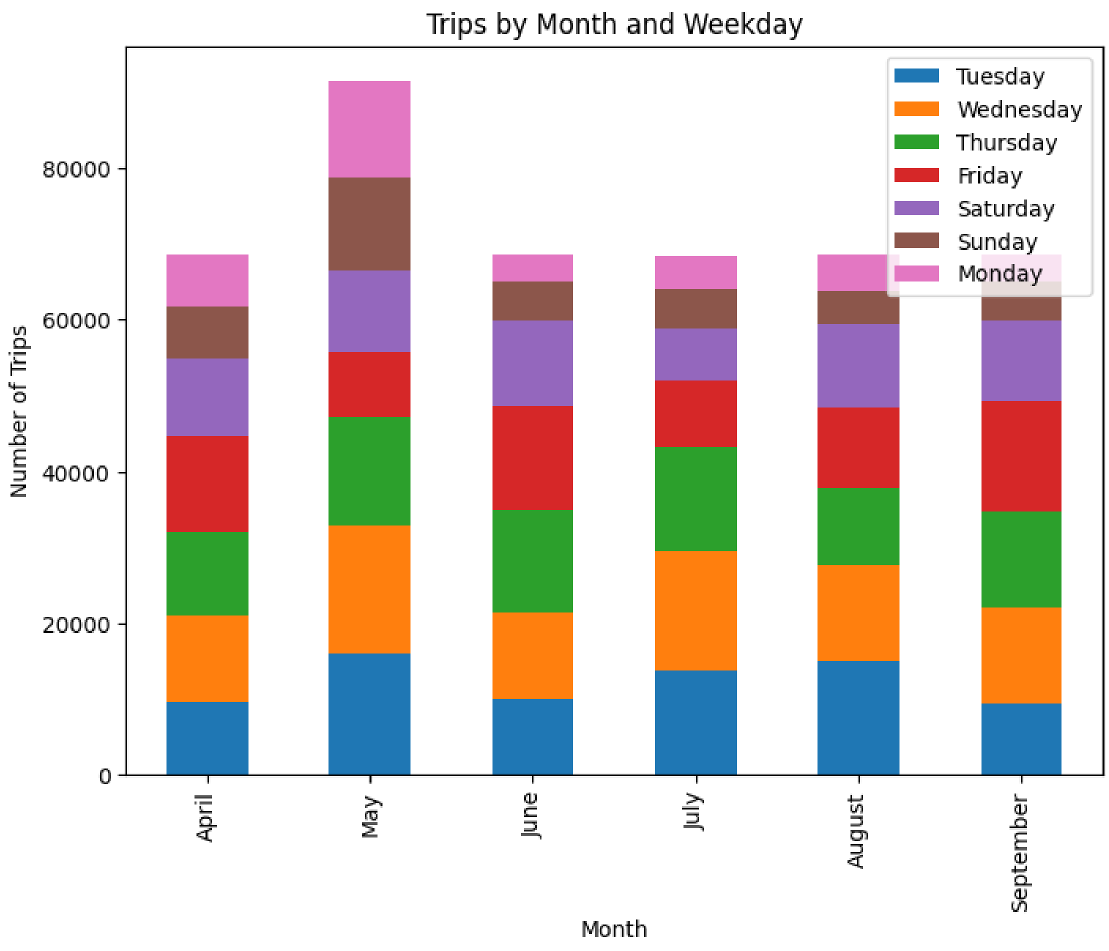
[35]:
```
                 Number_of_trips
Month Weekday
April Tuesday               9609
      Wednesday            11470
      Thursday             10992
      Friday               12703
      Saturday             10173
      Sunday                6894
      Monday                6692
May   Thursday             15986
      Friday               16807
```

```
    Saturday              14450
```

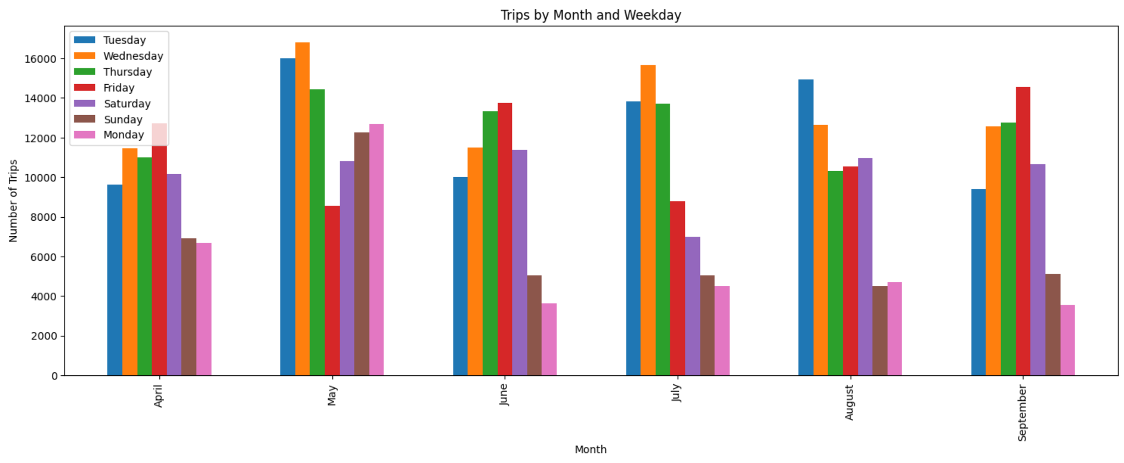
[36]:
```
df_month_weekday.reset_index(inplace= True)
data_month_weekday = df_month_weekday['Number_of_trips'].values.reshape(6,7)
df_month_weekday = pd.DataFrame(data = data_month_weekday, index =␣
 ↪df_month_weekday['Month'].unique(), columns = df['Weekday'].unique())
df_month_weekday.head()
```

[36]:
```
        Tuesday  Wednesday  Thursday  Friday  Saturday  Sunday  Monday
April      9609      11470     10992   12703     10173    6894    6692
May       15986      16807     14450    8570     10791   12247   12663
June       9993      11502     13346   13764     11370    5024    3621
July      13816      15659     13703    8779      6976    5043    4491
August    14926      12657     10307   10553     10946    4516    4709
```

[37]:
```
df_month_weekday.plot(kind='bar', figsize=(8,6), stacked = True)
plt.xlabel('Month')
plt.ylabel('Number of Trips')
plt.title('Trips by Month and Weekday')
plt.show()
```



[38]:
```python
df_month_weekday.plot(kind='bar', figsize=(18,6), width = 0.6)
plt.xlabel('Month')
plt.ylabel('Number of Trips')
plt.title('Trips by Month and Weekday')
plt.show()
```


Trips by Month and Weekday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Monday
Number of Trips
April
May
June
July
August
September
Month