SpekIt Technical Challenge

Business Problem

Pull the following from this link https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

- Taxi trip records from 4 sources
- Associated data dictionaries
- Taxi zone lookups

Answer the following questions about the effects of the Corona Virus on taxi transportation in NYC:

- What has been the economic impact of the pandemic on NYC taxi revenue over time?
- What was the most expensive trip? Between what zones? Conversely, what was the cheapest trip? What is the difference between these before and during the peak impact of the pandemic?
- What's the most popular payment method? Did this change because of the pandemic?
- What's the most expensive day of the week to travel on? Did this change because of the pandemic?

Submit your answers in the form of a GitHub repository, with additional instructions on how to reproduce them. Store the data into an AWS Postgres instance (free tier) or equivalent and provide access to us to review.

Pre-processing of the original data

- Only files from 2018, 2019, 2020 and 2021 were used
- Only filed with the Yellow and the Green tripdata were used
- Rows with VendorID equal NULL were dropped
- Only 75% of the records chosen randomly were included in the final analysis
- Duplicate rows were dropped
- Rows with timestamps with errors we dropped
- The final sqlite Database has two tables

Most of the modifications to the original data were done to reduce the volume of the records unmanageable on my home machine. The usage of AWS PostgreSQL instance was cost prohibitive.

Visualization:

Number of trips between 2018 and 2021 (July). The plot is interactive in Jupyter notebook but GitHub does not accept plotly images

```
# # This cell is commented out because
# # it takes too long to run and the DataFrame is saved from the previous runs into a pickle file

# freq='W'

# df_x = df_tripdata.groupby(['File', pd.Grouper(key='timestamp',
#                                                freq=freq)])['File'].agg(['count']).reset_index()
# df_x = df_x.sort_values(by=['timestamp', 'count'])
```

executed in 9m 33s, finished 17:55:59 2021-12-22

```
# # This cell is commented out to not overwrite the previously pickled file


# # Pickling the DF for the future use

# with open('data/pickled_dataframes/df_weekly_volume.pickle', 'wb') as f:
#     pickle.dump(df_x, f)
```

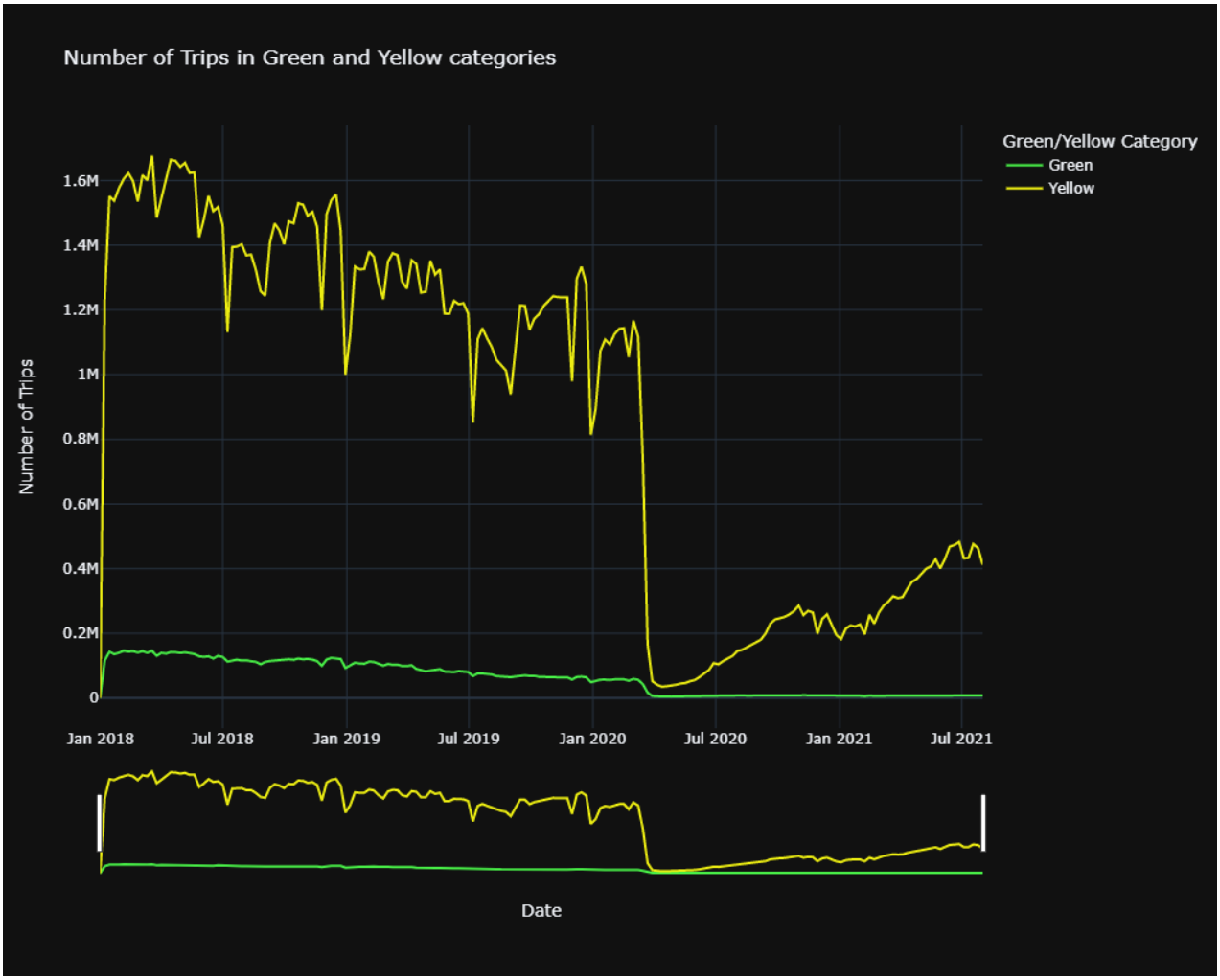
executed in 84ms, finished 17:56:45 2021-12-22

```
#UnPickling the DF for the future use

with open('data/pickled_dataframes/df_weekly_volume.pickle', 'rb') as f:
    df_x=pickle.load(f)
```

executed in 104ms, finished 17:56:51 2021-12-22

```
#Visualizing weekly volume of trips

color_grn_yllw=['#48ED47','#ECED12']
grn_yllw_categories=['Green', 'Yellow']

color_discrete_map_=dict(zip(grn_yllw_categories,color_grn_yllw))

fig1 = px.line(df_x, x='timestamp', y='count', color='File',
               color_discrete_map=color_discrete_map_,
labels={ "timestamp": "Date",  "count": "Number of Trips", "File": "Green/Yellow Category"},
      title='Number of Trips in Green and Yellow categories',
template="plotly_dark"
             )

fig1.update_layout(width=1000,
                   height=800)

fig1.update_layout(
    xaxis=dict(
        rangeslider=dict(
            visible=True
        ),
    )
)
fig1.show()
```

executed in 2.06s, finished 17:57:00 2021-12-22



This visualization shows a pretty dramatic impact of the pandemic on the weekly trip volume of NYC taxi services both in Yellow and in Green categories.

- While the volume of the trips is significantly more significant in the Yellow category, the trend in both categories is the same. Dramatic drop at the beginning of the pandemic in March 2020 with a slow recovery in Yellow category up to about 35-40% of the pre-pandemic volume. The Green category remains almost flat at probably 10% of the pre-pandemic level.

It is also worth mentioning that both categories were experiencing a downward trend over the last two pre-pandemic years. Possibly due to the competition from Uber, Lyft, and alike

- .

Weekly income of NYC taxi over the period of 2018 (January) and 2021 (July)

```
## This cell is commented out because
## it takes too long to run and the DataFrame is saved from the previous runs into a pickle file

# freq='W'

# df_x = df_tripdata.groupby(['File', pd.Grouper(key='timestamp',
#                                                freq=freq)])['total_amount'].sum().reset_index()
# df_x = df_x.sort_values(by=['timestamp', 'total_amount'])

```

executed in 5m 59s, finished 17:45:29 2021-12-22

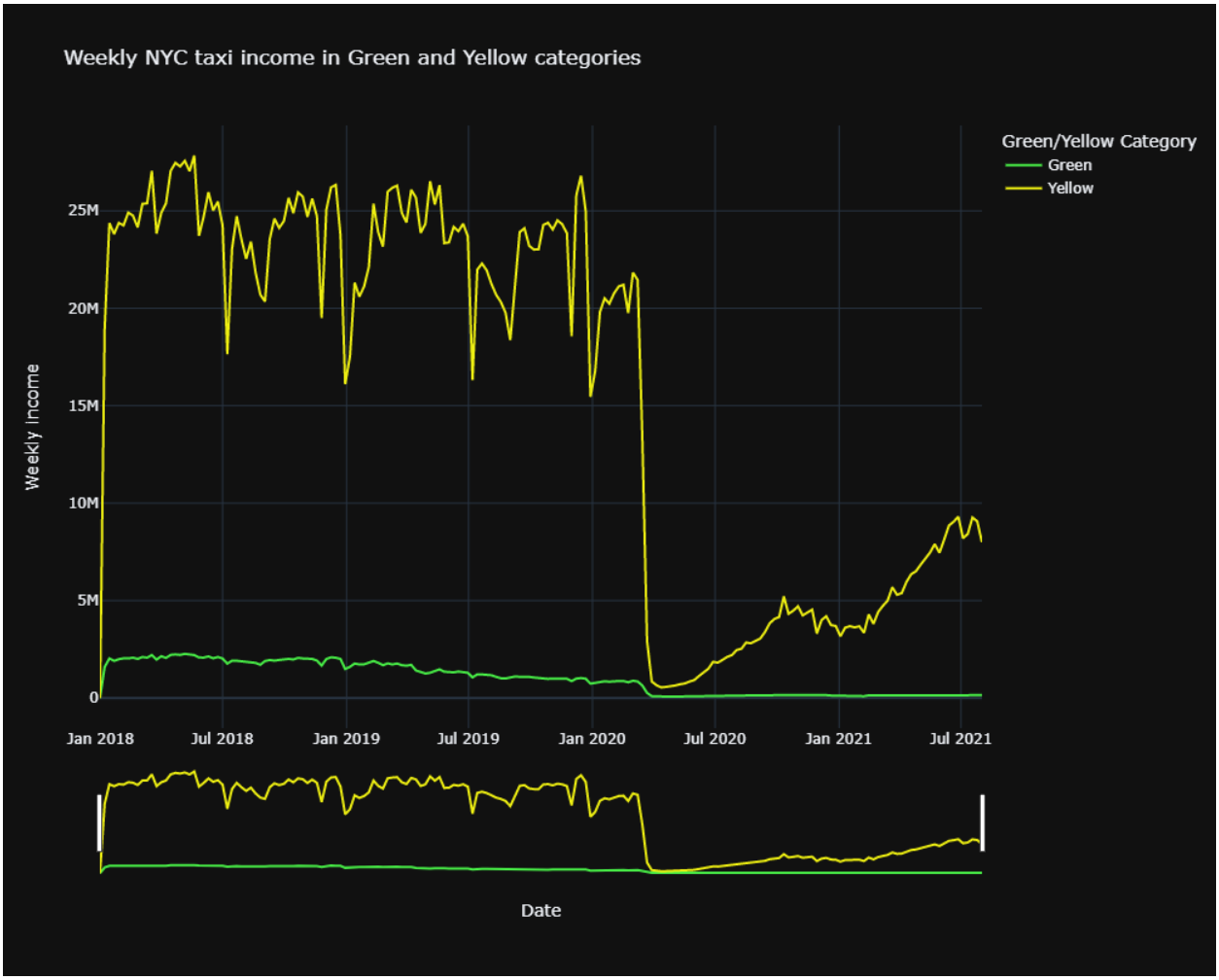
```
## This cell is commented out to not overwrite the previously pickled file


# Pickling the DF for the future use

# with open('data/pickled_dataframes/df_weekly_income.pickle', 'wb') as f:
#     pickle.dump(df_x, f)
```

executed in 5m 59s, finished 17:45:29 2021-12-22

```
#UnPickling the DF for the future use

with open('data/pickled_dataframes/df_weekly_income.pickle', 'rb') as f:
    df_x=pickle.load(f)
```

executed in 107ms, finished 17:57:32 2021-12-22

```
#Visualizing weekly income of trips

color_grn_yllw=['#48ED47','#ECED12']
grn_yllw_categories=['Green', 'Yellow']

color_discrete_map_=dict(zip(grn_yllw_categories,color_grn_yllw))

fig1 = px.line(df_x, x='timestamp', y='total_amount', color='File',
               color_discrete_map=color_discrete_map_,
labels={ "timestamp": "Date",  "total_amount": "Weekly income", "File": "Green/Yellow Category"},
      title='Weekly NYC taxi income in Green and Yellow categories',
template="plotly_dark"
              )

fig1.update_layout(width=1000,
                   height=800)

fig1.update_layout(
    xaxis=dict(
        rangeslider=dict(
            visible=True
        ),
    )
)
fig1.show()
```



This visualization shows a dramatic impact of the pandemic on the gross earning of NYC taxi services in the Yellow and Green categories. The trend follows the previous visualization of the volume of trips.

- While the earnings are significantly more significant in the Yellow category, the trend in both categories is the same. Dramatic drop at the beginning of the pandemic in March 2020 with a slow recovery in Yellow category up to about 30% of the pre-pandemic volume. The Green category remains almost flat at probably about 20% of the pre-pandemic level.
-
-
- It is also worth mentioning that both categories were experiencing a downward trend over the last two pre-pandemic years. Possibly due to the competition from Uber, Lyft, and alike. However, the Yellow category felt the competing market's financial impact less than the Green category. Possibly their strategy to remain competitive working better.

Number of transactions in each payment category between 2018 (January) and 2021 (July)

```
## This cell is commented out because
## it takes too long to run and the DataFrame is saved from the previous runs into a pickle file

# freq='W'

# df_x = df_tripdata.groupby(['payment_type', pd.Grouper(key='timestamp',
#                                                        freq=freq)])['payment_type'].agg(['count']).reset_index()
# df_x = df_x.sort_values(by=['timestamp', 'count'])
# df_x.head(25)
```

executed in 5m 59s, finished 17:45:29 2021-12-22

```
## This cell is commented out to not overwrite the previously pickled file

## Pickling the DF for the future use

# with open('data/pickled_dataframes/df_weekly_type_of_payments_volume.pickle', 'wb') as f:
#     pickle.dump(df_x, f)
```

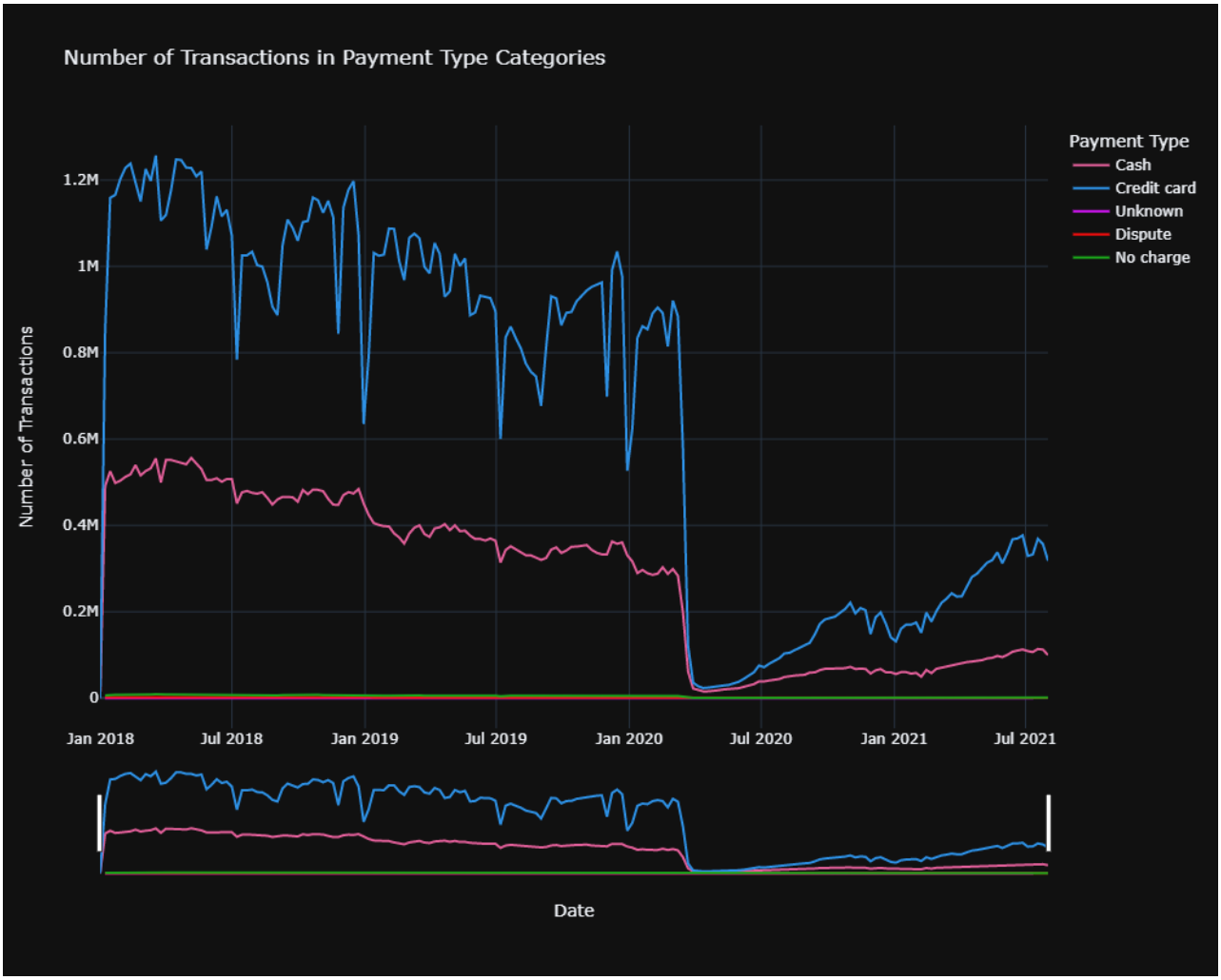
executed in 5m 59s, finished 17:45:29 2021-12-22

```
#UnPickling the DF for the future use

with open('data/pickled_dataframes/df_weekly_type_of_payments_volume.pickle', 'rb') as f:
    df_x=pickle.load(f)
```

executed in 120ms, finished 17:58:05 2021-12-22

```
#Visualizing weekly volume of payments in payment type categories


colors_dark24=px.colors.qualitative.Dark24
colors_dark24=colors_dark24[:-1]
payment_categories=['Credit card','Cash','No charge','Dispute','Unknown','Voided trip']

color_discrete_map_=dict(zip(payment_categories,colors_dark24))

fig1 = px.line(df_x, x='timestamp', y='count', color='payment_type',
              color_discrete_map=color_discrete_map_,
labels={ "timestamp": "Date",  "count": "Number of Transactions", "payment_type": "Payment Type"},
      title='Number of Transactions in Payment Type Categories',
template="plotly_dark"
             )

fig1.update_layout(width=1000,
                   height=800)

fig1.update_layout(
    xaxis=dict(
        rangeslider=dict(
            visible=True
        ),
    )
)
fig1.show()

```

executed in 180ms, finished 17:58:07 2021-12-22



The most transactions are made with credit cards followed by cash transactions. This tendency remained for after the pandemic onset with number of credit card transactions growing faster than the number of cash transactions

Weekly NYC taxi income per days of the week:

```
## This cell is commented out because
## it takes too long to run and the DataFrame is saved from the previous runs into a pickle file

# freq='W'

# df_x = df_tripdata.groupby(['Day of the week', pd.Grouper(key='timestamp',
#                                                           freq=freq)])['total_amount'].sum().reset_index()
# df_x = df_x.sort_values(by=['timestamp', 'total_amount'])
# df_x.head(25)
```

executed in 5m 59s, finished 17:45:29 2021-12-22

```
## This cell is commented out to not overwrite the previously pickled file

## Pickling the DF for the future use

# with open('data/pickled_dataframes/df_weekday_income.pickle', 'wb') as f:
#     pickle.dump(df_x, f)
```

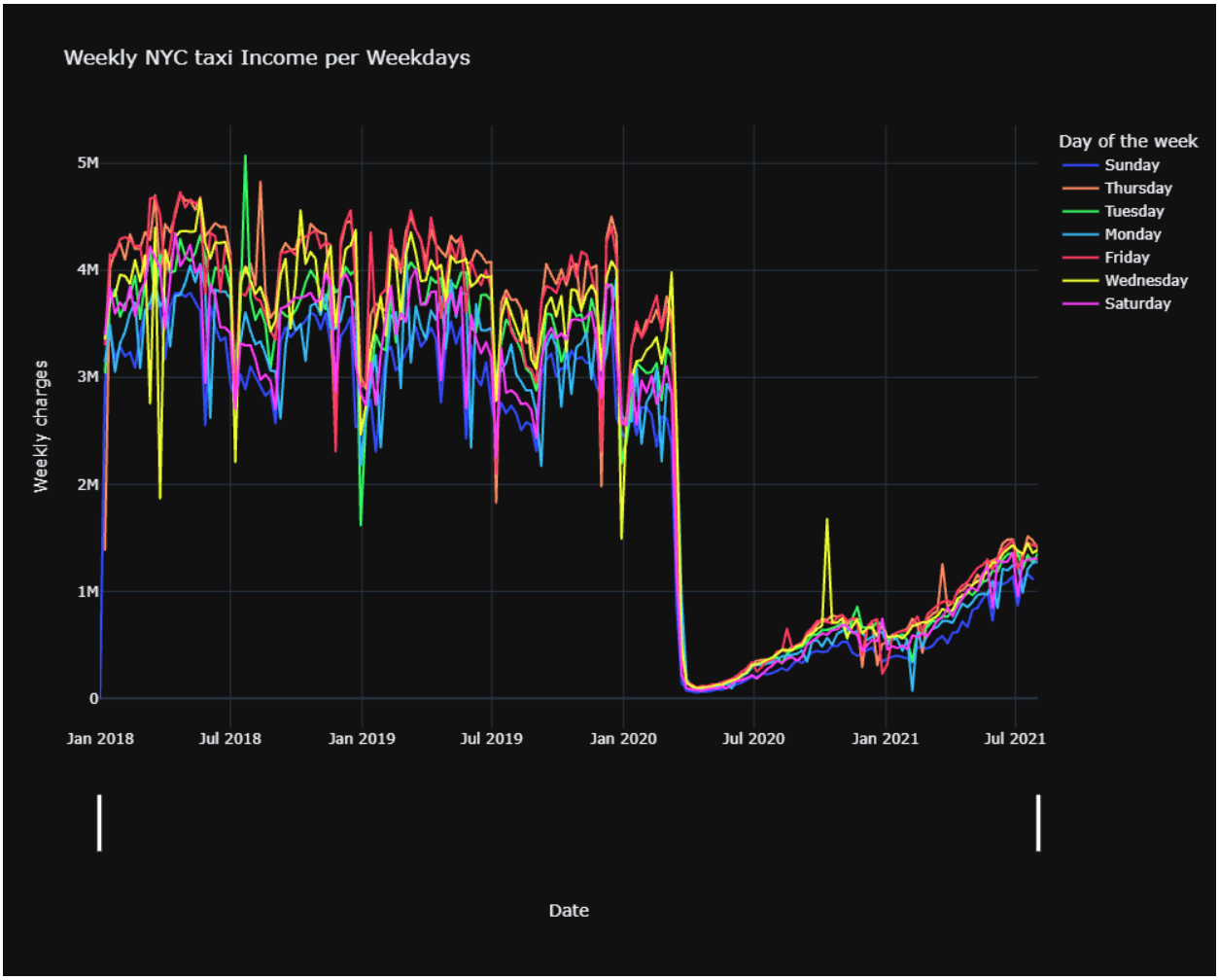
executed in 5m 59s, finished 17:45:29 2021-12-22

```
#UnPickling the DF for the future use

with open('data/pickled_dataframes/df_weekday_income.pickle', 'rb') as f:
    df_x=pickle.load(f)
```

executed in 101ms, finished 17:58:14 2021-12-22

```
color_weekdays=['#37BBFA','#37FA62', '#EFFF34','#FF8C63','#FF3C63', '#FF3CFE','#334BFF']
weekdays=['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday']

color_discrete_map_=dict(zip(weekdays,color_weekdays))


fig1 = px.line(df_x, x='timestamp', y='total_amount', color='Day of the week',
               color_discrete_map=color_discrete_map_,
labels={ "timestamp": "Date",  "total_amount": "Weekly charges", "Day of the week": "Day of the week"},
      title='Weekly NYC taxi Income per Weekdays',
template="plotly_dark"
             )

fig1.update_layout(width=1000,
                   height=800)

fig1.update_layout(
    xaxis=dict(
        rangeslider=dict(
            visible=True
        ),
    )
)
fig1.show()

```

executed in 231ms, finished 17:58:21 2021-12-22



The visualization is too close to make any conclusions. It seems that Friday and Thursday are the days with the most income for NYC taxi days, and Sunday rides bring the least income. However, I will address this issue in the Numeric analysis section.

Numeric Analysis:

What has been the economic impact of the pandemic on NYC taxi income over time?

Averaging daily income in all categories over two periods:

- It is impossible to calculate the revenue for this problem because we don't know the budget of NYC taxi; therefore I am calculating the income.
- the number of days since the onset of the pandemic measures on March 1, 2020 (approximate) and the last day in the data files provided
- the same number of days before March 1, 2020

The total number of days after the onset of the pandemic and the last data point: 517. That is the number of days I will be using to average income and volume of transactions

```
# Averaging daily income over a number of days
# Sum all income over a period of time/number of days

#Calculating number of days between since the pandemic started and the last data point
diff = a - datetime.strptime('3/1/2020', '%m/%d/%Y')
print('Total number of days after the onset of the pandemic and the last data point: {}'.format(diff.days))

# Datastamp 517 days before the inset of the pandemic
date_before = datetime.strptime('3/1/2020', '%m/%d/%Y') - timedelta(diff.days)
print('Pandemic onset timestamp: {}'.format(datetime.strptime('3/1/2020', '%m/%d/%Y')))
print('Timestamp {} days before pandemic onset : {}'.format(diff.days,date_before))
```

executed in 119ms, finished 18:06:37 2021-12-22

```
Total number of days after the onset of the pandemic and the last data point: 517
Pandemic onset timestamp: 2020-03-01 00:00:00
Timestamp 517 days before pandemic onset: 2018-10-01 00:00:00
```

```
# Creating two dataframes for before and after the pandemic onset (517 days)
df_temp_after=df_tripdata.loc[df_tripdata.timestamp > '2020-03-01']
b=df_temp_after.total_amount.sum()

df_temp_before=df_tripdata.loc[(df_tripdata.timestamp > datetime.strptime(str(date_before),"%Y-%m-%d %H:%M:%S")) & (df_tripd
d=df_temp_before.total_amount.sum()

#Printing averagedayli income for both dataframes and the percentage of the after
average_daily_after=b/diff.days
average_daily_before=d/diff.days
perc=average_daily_after/average_daily_before*100

print('Average daily income after the onset of the pandemic ({} days): ${}'.format(diff.days,round(average_daily_after)))
print('Average daily income before the onset of the pandemic ({} days): ${}'.format(diff.days,round(average_daily_before)))
print('Average daily income as a percentage of the prior to pandemic income: {}%'.format(round(perc)))
```

executed in 12.9s, finished 18:06:51 2021-12-22

```
Average daily income after the onset of the pandemic (517 days): $675301
Average daily income before the onset of the pandemic (517 days): $3474778
Average daily income as a percentage of the prior to pandemic income: 19%
```

Pandemic impact on the daily income in green and yellow categories

```
# Creating two dataframes for before and after the pandemic onset (517 days) in Yellow category
df_temp_after_Y=df_tripdata_yellow.loc[df_tripdata_yellow.timestamp > '2020-03-01']
b=df_temp_after_Y.total_amount.sum()

df_temp_before_Y=df_tripdata_yellow.loc[(df_tripdata_yellow.timestamp > datetime.strptime(str(date_before),"%Y-%m-%d %H:%M:%
d=df_temp_before_Y.total_amount.sum()

#Printing averagedayli income for both dataframes and the percentage of the after
average_daily_after=b/diff.days
average_daily_before=d/diff.days
perc=average_daily_after/average_daily_before*100

print('Average daily income after the onset of the pandemic ({} days) in Yellow Category: ${}'.format(diff.days,round(averag
print('Average daily income before the onset of the pandemic ({} days) in Yellow Category: ${}'.format(diff.days,round(avera
print('Average daily income as a percentage of the prior to pandemic income in Yellow Category: {}%'.format(round(perc)))
```

executed in 2m 6s, finished 18:09:50 2021-12-22

```
Average daily income after the onset of the pandemic (517 days) in Yellow Cat
egory: $653354
Average daily income before the onset of the pandemic (517 days) in Yellow Ca
tegory: $3280259
Average daily income as a percentage of the prior to pandemic income in Yello
w Category: 20%
```

```
# Creating two dataframes for before and after the pandemic onset (517 days) in Yellow category
df_temp_after_G=df_tripdata_green.loc[df_tripdata_green.timestamp > '2020-03-01']
b=df_temp_after_G.total_amount.sum()

df_temp_before_G=df_tripdata_green.loc[(df_tripdata_green.timestamp > datetime.strptime(str(date_before),"%Y-%m-%d %H:%M:%S"
d=df_temp_before_G.total_amount.sum()

#Printing averagedayli income for both dataframes and the percentage of the after
average_daily_after=b/diff.days
average_daily_before=d/diff.days
perc=average_daily_after/average_daily_before*100

print('Average daily income after the onset of the pandemic ({} days) in Green Category: ${}'.format(diff.days,round(average
print('Average daily income before the onset of the pandemic ({} days) in Green Category: ${}'.format(diff.days,round(averag
print('Average daily income as a percentage of the prior to pandemic income in Yellow Category: {}%'.format(round(perc)))
```

executed in 2m 39s, finished 18:13:45 2021-12-22

```
Average daily income after the onset of the pandemic (517 days) in Green C
ategory: $21947
Average daily income before the onset of the pandemic (517 days) in Green
Category: $194519
Average daily income as a percentage of the prior to pandemic income in Ye
llow Category: 11%
```

Maximum and minimum charges before and after the pandemic?

AFTER the onset of the pandemic

```
# Min amount charged after the pundemic

min_after=df_temp_after.total_amount.min()
df=df_temp_after.loc[df_temp_after.total_amount==min_after]
df
```

executed in 366ms, finished 18:14:43 2021-12-22

	PULocationID	DOLocationID	total_amount	payment_type	timestamp	File	Day of the week
12041018	264	264	0.01	Credit card	2021-01-30 14:50:20	Yellow	Saturday
12310715	264	264	0.01	Credit card	2021-03-25 13:09:22	Yellow	Thursday
12583717	264	264	0.01	Credit card	2021-05-01 09:35:59	Yellow	Saturday
12615567	264	264	0.01	Credit card	2021-04-06 09:48:27	Yellow	Tuesday
15887621	264	264	0.01	Credit card	2021-07-23 18:14:30	Yellow	Friday
16137149	264	264	0.01	Credit card	2021-07-05 18:34:02	Yellow	Monday
16354685	264	264	0.01	Credit card	2021-02-24 11:46:34	Yellow	Wednesday
17582591	264	264	0.01	Credit card	2021-06-25 17:14:04	Yellow	Friday
18279532	264	264	0.01	Credit card	2021-06-25 18:15:59	Yellow	Friday
19468845	264	264	0.01	Credit card	2021-05-24 17:56:25	Yellow	Monday
19554930	265	264	0.01	Cash	2021-03-18 00:21:44	Yellow	Thursday
19684855	48	233	0.01	Cash	2021-01-23 17:37:11	Yellow	Saturday
19911632	264	264	0.01	Credit card	2021-02-19 14:13:19	Yellow	Friday
21505630	264	264	0.01	Credit card	2021-06-21 15:29:08	Yellow	Monday
21717923	264	264	0.01	Credit card	2021-03-06 13:07:48	Yellow	Saturday
22159563	264	264	0.01	Credit card	2021-03-31 10:09:14	Yellow	Wednesday
22224882	264	264	0.01	Credit card	2021-07-24 17:37:30	Yellow	Saturday
24010104	264	264	0.01	Credit card	2020-10-19 14:21:36	Yellow	Monday
26538122	264	264	0.01	Credit card	2020-09-02 13:14:58	Yellow	Wednesday
32043091	264	264	0.01	Credit card	2020-12-12 12:52:25	Yellow	Saturday
33319549	264	264	0.01	Credit card	2020-10-13 09:15:05	Yellow	Tuesday
35035185	264	264	0.01	Credit card	2020-10-28 10:57:35	Yellow	Wednesday
35194801	264	264	0.01	Credit card	2020-09-21 07:06:49	Yellow	Monday
36708790	264	264	0.01	Credit card	2020-09-28 10:35:13	Yellow	Monday
37379403	264	264	0.01	Credit card	2020-10-23 11:08:49	Yellow	Friday
38421532	264	264	0.01	Credit card	2020-06-29 10:16:48	Yellow	Monday

There are 26 trips after the onset of the pandemic with $0.01 charges; most of them are between 264-264 zones (the same zone), with one standing out, it's between zones 48 and 233.

```
# Max amount charged after the pandemic

max_after=df_temp_after.total_amount.max()
df=df_temp_after.loc[df_temp_after.total_amount==max_after]
df
```

executed in 183ms, finished 18:14:51 2021-12-22

	PULocationID	DOLocationID	total_amount	payment_type	timestamp	File	Day of the week
29934266	41	42	998325.61	No charge	2020-10-07 10:35:56	Yellow	Wednesday

There is just one trip after the onset of the pandemic with a charge of $998325.61; it's between zones 41 and 42. It is not clear why this insane amount has been charged. Possible to look at the original data for this date.

BEFORE the onset of the pandemic

```
# Min amount charged before the pandemic

min_before=df_temp_before.total_amount.min()
df=df_temp_before.loc[df_temp_before.total_amount==min_before]
df
```

executed in 489ms, finished 18:14:57 2021-12-22

	PULocationID	DOLocationID	total_amount	payment_type	timestamp	File	Day of the week
1139435	81	264	0.01	Cash	2019-08-27 06:37:18	Green	Tuesday
1143221	131	173	0.01	Credit card	2019-01-05 04:21:38	Green	Saturday
1144254	69	69	0.01	Cash	2019-03-15 06:55:39	Green	Friday
1146872	160	157	0.01	Credit card	2019-09-07 04:53:51	Green	Saturday
1168941	57	57	0.01	Credit card	2019-05-04 05:59:12	Green	Saturday
...	...	...	...	...	...	...	...
101406899	264	264	0.01	Cash	2019-11-13 18:56:42	Yellow	Wednesday
124772477	168	264	0.01	Cash	2018-10-13 19:43:42	Yellow	Saturday
128664365	138	264	0.01	Cash	2018-10-14 22:27:26	Yellow	Sunday
153897757	264	264	0.01	Cash	2018-12-16 14:54:11	Yellow	Sunday
166273362	230	264	0.01	Cash	2018-10-23 16:24:22	Yellow	Tuesday

672 rows × 7 columns

There are 672 trips before the onset of the pandemic with $0.01 charges, and therefore a combination of a variety of zones.

```
(df.groupby(['PULocationID', 'DOLocationID']).size()
   .sort_values(ascending=False)
   .reset_index(name='count')
   .drop_duplicates(subset='DOLocationID'))
```

executed in 182ms, finished 18:15:04 2021-12-22

	PULocationID	DOLocationID	count
0	247	247	41
1	82	82	28
2	193	193	17
3	191	191	14
4	235	235	13
...	...	...	...
250	161	162	1
261	153	200	1
262	152	248	1
265	152	132	1
272	2	2	1

117 rows × 3 columns

There are 117 combinations of zones before the onset of the pandemic with $0.01 charges.

There is just one trip after the onset of the pandemic with a charge of $671124.94; it's between zones 138 and 181. It is not clear why this insane amount has been charged. Possible to look at the original data for this date.

What's the most popular payment method? Did this change because of the pandemic?

AFTER the onset of the pandemic

```
#Creating dataframes for type of payments for after the pandemic onset

df_tripdata_credit_card_after=df_temp_after.loc[df_temp_after.payment_type=='Credit card']
df_tripdata_cash_after=df_temp_after.loc[df_temp_after.payment_type=='Cash']
df_tripdata_nocharge_after=df_temp_after.loc[df_temp_after.payment_type=='No charge']
df_tripdata_dispute_after=df_temp_after.loc[df_temp_after.payment_type=='Dispute']
df_tripdata_unknown_after=df_temp_after.loc[df_temp_after.payment_type=='Unknown']
df_tripdata_voidedtrip_after=df_temp_after.loc[df_temp_after.payment_type=='Voided trip']

#Calculating number of transactions in each category
num_w_credit_card_after=df_tripdata_credit_card_after.timestamp.count()
num_w_cash_after=df_tripdata_cash_after.timestamp.count()
num_nocharge_after=df_tripdata_nocharge_after.timestamp.count()
num_dispute_after=df_tripdata_dispute_after.timestamp.count()
num_unknown_after=df_tripdata_unknown_after.timestamp.count()
num_voided_after=df_tripdata_voidedtrip_after.timestamp.count()

print('Number of credit card transactions for {} days after the pandemic onset: {}'.format(diff.days,num_w_credit_card_after
print('Number of cash transactions for {} days after the pandemic onset: {}'.format(diff.days,num_w_cash_after))
print('Number of no charge transactions for {} days after the pandemic onset: {}'.format(diff.days,num_nocharge_after))
print('Number of disputed transactions for {} days after the pandemic onset: {}'.format(diff.days,num_dispute_after))
print('Number of transactions unknow type of payment for {} days after the pandemic onset: {}'.format(diff.days,num_unknown_
print('Number of voided trip transactions for {} days after the pandemic onset: {}'.format(diff.days,num_voided_after))

```

executed in 6.99s, finished 18:15:20 2021-12-22

```
Number of credit card transactions for 517 days after the pandemic onset: 14452618
Number of cash transactions for 517 days after the pandemic onset: 5035772
Number of no charge transactions for 517 days after the pandemic onset: 93188
Number of disputed transactions for 517 days after the pandemic onset: 33981
Number of transactions unknow type of payment for 517 days after the pandemic onset: 27
Number of voided trip transactions for 517 days after the pandemic onset: 0
```

```
Number of credit card transactions for 517 days after the pandemic onset:
14452618
Number of cash transactions for 517 days after the pandemic onset: 5035772
Number of no charge transactions for 517 days after the pandemic onset: 93
188
Number of disputed transactions for 517 days after the pandemic onset: 339
81
Number of transactions unknown type of payment for 517 days after the pand
emic onset: 27
Number of voided trip transactions for 517 days after the pandemic onset:
0
```

Credit card is the most popular type of payment followed by cash payments after the pandemic onset.

BEFORE the onset of the pandemic

```
#Creating dataframes for type of payments for after the pandemic onset

df_tripdata_credit_card_before=df_temp_before.loc[df_temp_before.payment_type=='Credit card']
df_tripdata_cash_before=df_temp_before.loc[df_temp_before.payment_type=='Cash']
df_tripdata_nocharge_before=df_temp_before.loc[df_temp_before.payment_type=='No charge']
df_tripdata_dispute_before=df_temp_before.loc[df_temp_before.payment_type=='Dispute']
df_tripdata_unknown_before=df_temp_before.loc[df_temp_before.payment_type=='Unknown']
df_tripdata_voidedtrip_before=df_temp_before.loc[df_temp_before.payment_type=='Voided trip']

#Calculating number of transactions in each category

num_w_credit_card_before=df_tripdata_credit_card_before.timestamp.count()
num_w_cash_before=df_tripdata_cash_before.timestamp.count()
num_nocharge_before=df_tripdata_nocharge_before.timestamp.count()
num_dispute_before=df_tripdata_dispute_before.timestamp.count()
num_unknown_before=df_tripdata_unknown_before.timestamp.count()
num_voided_before=df_tripdata_voidedtrip_before.timestamp.count()

print('Number of credit card transactions for {} days before the pandemic onset: {}'.format(diff.days,num_w_credit_card_befo
print('Number of cash transactions for {} days before the pandemic onset: {}'.format(diff.days,num_w_cash_before))
print('Number of no charge transactions for {} before after the pandemic onset: {}'.format(diff.days,num_nocharge_before))
print('Number of disputed transactions for {} before after the pandemic onset: {}'.format(diff.days,num_dispute_before))
print('Number of transactions unknow type of before for {} days after the pandemic onset: {}'.format(diff.days,num_unknown_b
print('Number of voided trip transactions for {} days before the pandemic onset: {}'.format(diff.days,num_voided_before))
```

executed in 34.4s, finished 18:16:11 2021-12-22

```
Number of credit card transactions for 517 days before the pandemic onset: 69261198
Number of cash transactions for 517 days before the pandemic onset: 27623982
Number of no charge transactions for 517 before after the pandemic onset: 414838
Number of disputed transactions for 517 before after the pandemic onset: 131225
Number of transactions unknow type of before for 517 days after the pandemic onset: 214
Number of voided trip transactions for 517 days before the pandemic onset: 0
```

Number of credit card transactions for 517 days before the pandemic onset: 69261198
Number of cash transactions for 517 days before the pandemic onset: 27623982
Number of no charge transactions for 517 before after the pandemic onset: 414838
Number of disputed transactions for 517 before after the pandemic onset: 131225
Number of transactions unknow type of before for 517 days after the pandemic onset: 214
Number of voided trip transactions for 517 days before the pandemic onset: 0

Credit card is the most popular type of payment followed by cash payments after the pandemic onset

What's the most expensive day of the week to travel on? Did this change because of the pandemic?

It was unclear what it means "most expensive day of the week. Therefore, I calculated both the total amount of charges per day of the week for both periods

AND

An average trip cost per day of the week for both periods

1.

AFTER the onset of the pandemic

```
#Creating dataframes for type of payments for after the pandemic onset

df_tripdata_Monday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Monday']
df_tripdata_Tuesday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Tuesday']
df_tripdata_Wednesday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Wednesday']
df_tripdata_Thursday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Thursday']
df_tripdata_Friday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Friday']
df_tripdata_Saturday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Saturday']
df_tripdata_Sunday_after=df_temp_after.loc[df_temp_after['Day of the week']=='Sunday']

#Calculating number of transactions in each category
exp_Monday_after=round(df_tripdata_Monday_after.total_amount.sum(),2)
exp_Tuesday_after=round(df_tripdata_Tuesday_after.total_amount.sum(),2)
exp_Wednesday_after=round(df_tripdata_Wednesday_after.total_amount.sum(),2)
exp_Thursday_after=round(df_tripdata_Thursday_after.total_amount.sum(),2)
exp_Friday_after=round(df_tripdata_Friday_after.total_amount.sum(),2)
exp_Saturday_after=round(df_tripdata_Saturday_after.total_amount.sum(),2)
exp_Sunday_after=round(df_tripdata_Sunday_after.total_amount.sum(),2)

print('Total amount of charges on Monday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Monday_after))
print('Total amount of charges on Tuesday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Tuesday_after))
print('Total amount of charges on Wednesday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Wednesday_after
print('Total amount of charges on Thursday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Thursday_after))
print('Total amount of charges on Friday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Friday_after))
print('Total amount of charges on Saturday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Saturday_after))
print('Total amount of charges on Sunday over {} days after the pandemic onset: ${}'.format(diff.days,exp_Sunday_after))
```

executed in 8.34s, finished 18:17:04 2021-12-22

```
Total amount of charges on Monday over 517 days after the pandemic onset: $46851069.18
Total amount of charges on Tuesday over 517 days after the pandemic onset: $51740456.04
Total amount of charges on Wednesday over 517 days after the pandemic onset: $55562527.58
Total amount of charges on Thursday over 517 days after the pandemic onset: $55111901.04
Total amount of charges on Friday over 517 days after the pandemic onset: $54907415.91
Total amount of charges on Saturday over 517 days after the pandemic onset: $46423669.32
Total amount of charges on Sunday over 517 days after the pandemic onset: $38533511.46
```

- Total amount of charges on Monday over 517 days after the pandemic onset: $46851069.18
- Total amount of charges on Tuesday over 517 days after the pandemic onset: $51740456.04
- Total amount of charges on Wednesday over 517 days after the pandemic onset: $55562527.58
- Total amount of charges on Thursday over 517 days after the pandemic onset: $55111901.04
- Total amount of charges on Friday over 517 days after the pandemic onset: $54907415.91
- Total amount of charges on Saturday over 517 days after the pandemic onset: $46423669.32
- Total amount of charges on Sunday over 517 days after the pandemic onset: $38533511.46

Over the 517 days since the onset of the pandemic the income per each day of the week is:

- Monday: $46851069
- Tuesday: $51740456
- Wednesday: $55562528
- Thursday: $55111901
- Friday: $54907416

- Saturday: $46423669
- Sunday: $38533511

The most expensive (bringing the most money) day of the week is Wednesday and the least expensive is Sunday

BEFORE the onset of the pandemic

```
#Creating dataframes for type of payments for before the pandemic onset

df_tripdata_Monday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Monday']
df_tripdata_Tuesday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Tuesday']
df_tripdata_Wednesday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Wednesday']
df_tripdata_Thursday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Thursday']
df_tripdata_Friday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Friday']
df_tripdata_Saturday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Saturday']
df_tripdata_Sunday_before=df_temp_before.loc[df_temp_before['Day of the week']=='Sunday']

#Calculating number of transactions in each category
exp_Monday_before=round(df_tripdata_Monday_before.total_amount.sum(),2)
exp_Tuesday_before=round(df_tripdata_Tuesday_before.total_amount.sum(),2)
exp_Wednesday_before=round(df_tripdata_Wednesday_before.total_amount.sum(),2)
exp_Thursday_before=round(df_tripdata_Thursday_before.total_amount.sum(),2)
exp_Friday_before=round(df_tripdata_Friday_before.total_amount.sum(),2)
exp_Saturday_before=round(df_tripdata_Saturday_before.total_amount.sum(),2)
exp_Sunday_before=round(df_tripdata_Sunday_before.total_amount.sum(),2)

print('Total amount of charges on Monday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Monday_before))
print('Total amount of charges on Tuesday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Tuesday_before))
print('Total amount of charges on Wednesday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Wednesday_befo
print('Total amount of charges on Thurday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Thursday_before)
print('Total amount of charges on Friday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Friday_before))
print('Total amount of charges on Saturday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Saturday_before
print('Total amount of charges on Sunday over {} days before the pandemic onset: ${}'.format(diff.days,exp_Sunday_before))

executed in 41.0s, finished 18:17:51 2021-12-22
```

```
Total amount of charges on Monday over 517 days before the pandemic onset: $238247633.69
Total amount of charges on Tuesday over 517 days before the pandemic onset: $257295459.35
Total amount of charges on Wednesday over 517 days before the pandemic onset: $271477547.33
Total amount of charges on Thurday over 517 days before the pandemic onset: $283509485.42
Total amount of charges on Friday over 517 days before the pandemic onset: $280996827.36
Total amount of charges on Saturday over 517 days before the pandemic onset: $245110746.74
Total amount of charges on Sunday over 517 days before the pandemic onset: $219822658.5
```

- Total amount of charges on Monday over 517 days before the pandemic onset: $238247633.69
- Total amount of charges on Tuesday over 517 days before the pandemic onset: $257295459.35
- Total amount of charges on Wednesday over 517 days before the pandemic onset: $271477547.33
- Total amount of charges on Thurday over 517 days before the pandemic onset: $283509485.42
- Total amount of charges on Friday over 517 days before the pandemic onset: $280996827.36
- Total amount of charges on Saturday over 517 days before the pandemic onset: $245110746.74
- Total amount of charges on Sunday over 517 days before the pandemic onset: $219822658.5

Over the 517 days before the onset of the pandemic the income per each day of the week is:

- Monday: $238247633
- Tuesday: $257295459
- Wednesday: $271477547
- Thursday: $283509485
- Friday: $280996827
- Saturday: $245110747

- Sunday: $219822659

The most expensive (bringing the most money) day of the week is Thursday and the least expensive is Sunday

2.

Average trip cost

AFTER the onset of the pandemic

```
#Calculating amount charged in each category
exp_Monday_after=round(df_tripdata_Monday_after.total_amount.sum(),2)
exp_Tuesday_after=round(df_tripdata_Tuesday_after.total_amount.sum(),2)
exp_Wednesday_after=round(df_tripdata_Wednesday_after.total_amount.sum(),2)
exp_Thursday_after=round(df_tripdata_Thursday_after.total_amount.sum(),2)
exp_Friday_after=round(df_tripdata_Friday_after.total_amount.sum(),2)
exp_Saturday_after=round(df_tripdata_Saturday_after.total_amount.sum(),2)
exp_Sunday_after=round(df_tripdata_Sunday_after.total_amount.sum(),2)

#Calculating number of trips in each category
num_Monday_after=df_tripdata_Monday_after.timestamp.count()
num_Tuesday_after=df_tripdata_Tuesday_after.timestamp.count()
num_Wednesday_after=df_tripdata_Wednesday_after.timestamp.count()
num_Thursday_after=df_tripdata_Thursday_after.timestamp.count()
num_Friday_after=df_tripdata_Friday_after.timestamp.count()
num_Saturday_after=df_tripdata_Saturday_after.timestamp.count()
num_Sunday_after=df_tripdata_Sunday_after.timestamp.count()

#Calculating average cost in each category
avg_cost_Monday_after=round(exp_Monday_after/num_Monday_after,2)
avg_cost_Tuesday_after=round(exp_Tuesday_after/num_Tuesday_after,2)
avg_cost_Wednesday_after=round(exp_Wednesday_after/num_Wednesday_after,2)
avg_cost_Thursday_after=round(exp_Thursday_after/num_Thursday_after,2)
avg_cost_Friday_after=round(exp_Friday_after/num_Friday_after,2)
avg_cost_Saturday_after=round(exp_Saturday_after/num_Saturday_after,2)
avg_cost_Sunday_after=round(exp_Sunday_after/num_Sunday_after,2)


print('Average trip cost on Monday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Monday_after))
print('Average trip cost on Tuesday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Tuesday_after))
print('Average trip cost on Wednesday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Wednesday_after)
print('Average trip cost on Thursday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Thursday_after))
print('Average trip cost on Friday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Friday_after))
print('Average trip cost on Saturday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Saturday_after))
print('Average trip cost on Sunday over {} days after the pandemic onset: ${}'.format(diff.days,avg_cost_Sunday_after))
```

executed in 221ms, finished 18:25:08 2021-12-22

```
Average trip cost on Monday over 517 days after the pandemic onset: $17.7
Average trip cost on Tuesday over 517 days after the pandemic onset: $17.48
Average trip cost on Wednesday over 517 days after the pandemic onset: $17.99
Average trip cost on on Thursday over 517 days after the pandemic onset: $17.86
Average trip cost on on Friday over 517 days after the pandemic onset: $17.82
Average trip cost on on Saturday over 517 days after the pandemic onset: $17.51
Average trip cost on on Sunday over 517 days after the pandemic onset: $18.34
```

Average trip cost on Monday over 517 days after the pandemic onset: $17.7
Average trip cost on Tuesday over 517 days after the pandemic onset: $17.48
Average trip cost on Wednesday over 517 days after the pandemic onset: $17.99
Average trip cost on Thursday over 517 days after the pandemic onset: $17.86
Average trip cost on Friday over 517 days after the pandemic onset: $17.82
Average trip cost on Saturday over 517 days after the pandemic onset: $17.51
Average trip cost on Sunday over 517 days after the pandemic onset: $18.34

The most expensive day of the week after the pandemic onset is Wednesday and the least expensive is Tuesday. H the difference is negligible.

BEFORE the onset of the pandemic

```
#Calculating amount charged in each category
exp_Monday_before=round(df_tripdata_Monday_before.total_amount.sum(),2)
exp_Tuesday_before=round(df_tripdata_Tuesday_before.total_amount.sum(),2)
exp_Wednesday_before=round(df_tripdata_Wednesday_before.total_amount.sum(),2)
exp_Thursday_before=round(df_tripdata_Thursday_before.total_amount.sum(),2)
exp_Friday_before=round(df_tripdata_Friday_before.total_amount.sum(),2)
exp_Saturday_before=round(df_tripdata_Saturday_before.total_amount.sum(),2)
exp_Sunday_before=round(df_tripdata_Sunday_before.total_amount.sum(),2)

#Calculating number of trips in each category
num_Monday_before=df_tripdata_Monday_before.timestamp.count()
num_Tuesday_before=df_tripdata_Tuesday_before.timestamp.count()
num_Wednesday_before=df_tripdata_Wednesday_before.timestamp.count()
num_Thursday_before=df_tripdata_Thursday_before.timestamp.count()
num_Friday_before=df_tripdata_Friday_before.timestamp.count()
num_Saturday_before=df_tripdata_Saturday_before.timestamp.count()
num_Sunday_before=df_tripdata_Sunday_before.timestamp.count()

#Calculating average cost in each category
avg_cost_Monday_before=round(exp_Monday_before/num_Monday_before,2)
avg_cost_Tuesday_before=round(exp_Tuesday_before/num_Tuesday_before,2)
avg_cost_Wednesday_before=round(exp_Wednesday_before/num_Wednesday_before,2)
avg_cost_Thursday_before=round(exp_Thursday_before/num_Thursday_before,2)
avg_cost_Friday_before=round(exp_Friday_before/num_Friday_before,2)
avg_cost_Saturday_before=round(exp_Saturday_before/num_Saturday_before,2)
avg_cost_Sunday_before=round(exp_Sunday_before/num_Sunday_before,2)


print('Average trip cost on Monday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Monday_before))
print('Average trip cost on Tuesday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Tuesday_before))
print('Average trip cost on Wednesday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Wednesday_befor
print('Average trip cost on on Thursday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Thursday_befo
print('Average trip cost on Friday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Friday_before))
print('Average trip cost on Saturday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Saturday_before)
print('Average trip cost on Sunday over {} days before the pandemic onset: ${}'.format(diff.days,avg_cost_Sunday_before))

```

executed in 598ms, finished 18:21:45 2021-12-22

```
Average trip cost on Monday over 517 days before the pandemic onset: $18.7
Average trip cost on Tuesday over 517 days before the pandemic onset: $18.38
Average trip cost on Wednesday over 517 days before the pandemic onset: $18.67
Average trip cost on on Thursday over 517 days before the pandemic onset: $18.92
Average trip cost on on Friday over 517 days before the pandemic onset: $18.69
Average trip cost on on Saturday over 517 days before the pandemic onset: $17.35
Average trip cost on on Sunday over 517 days before the pandemic onset: $18.31
```

Average trip cost on Monday over 517 days before the pandemic onset: $18.7
Average trip cost on Tuesday over 517 days before the pandemic onset: $18.38
Average trip cost on Wednesday over 517 days before the pandemic onset: $18.6
7
Average trip cost on Thursday over 517 days before the pandemic onset: $18.92
Average trip cost on Friday over 517 days before the pandemic onset: $18.69
Average trip cost on Saturday over 517 days before the pandemic onset: $17.35
Average trip cost on Sunday over 517 days before the pandemic onset: $18.31

The most expensive day of the week prior to pandemic is Thursday and the least expensive is Saturday. However, the difference is negligible.