Explore NYC taxi trips dataset

In [1]:
```
import pandas as pd
from bokeh.plotting import figure,show
from bokeh.models import FactorRange,ColumnDataSource,NumeralTickFormatter
import numpy as np
from bokeh.io import output_notebook
import bokeh.palettes
from bokeh.transform import factor_cmap
from bokeh.layouts import row
```

In [2]:
```
#load a subset of the new york city TLC taxi trips,which is the yellow taxi trip records in May,2021
ny_taxi_raw=pd.read_parquet('https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_tripdata_2021-05.par
```

In [3]:
```
#only consider the actual taxi trips and trip amount>0
ny_taxi_raw=ny_taxi_raw[ny_taxi_raw['tpep_pickup_datetime']!=ny_taxi_raw['tpep_dropoff_datetime']]
ny_taxi_raw=ny_taxi_raw[ny_taxi_raw['total_amount']>0]
```

In [4]:
```
# get the overall description
ny_taxi_raw.info(show_counts=True)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2493735 entries, 0 to 2507108
Data columns (total 19 columns):
 #   Column                 Non-Null Count    Dtype
---  ------                 --------------    -----
 0   VendorID               2493735 non-null  int64
 1   tpep_pickup_datetime   2493735 non-null  datetime64[ns]
 2   tpep_dropoff_datetime  2493735 non-null  datetime64[ns]
 3   passenger_count        2366710 non-null  float64
 4   trip_distance          2493735 non-null  float64
 5   RatecodeID             2366710 non-null  float64
 6   store_and_fwd_flag     2366710 non-null  object
 7   PULocationID           2493735 non-null  int64
 8   DOLocationID           2493735 non-null  int64
 9   payment_type           2493735 non-null  int64
 10  fare_amount            2493735 non-null  float64
 11  extra                  2493735 non-null  float64
 12  mta_tax                2493735 non-null  float64
 13  tip_amount             2493735 non-null  float64
 14  tolls_amount           2493735 non-null  float64
 15  improvement_surcharge  2493735 non-null  float64
 16  total_amount           2493735 non-null  float64
 17  congestion_surcharge   2366710 non-null  float64
 18  airport_fee            2366683 non-null  float64
dtypes: datetime64[ns](2), float64(12), int64(4), object(1)
memory usage: 380.5+ MB
```

In [5]:
```
#None is also considered a missing value in pandas. value_counts count the Number of non-NA elements in
ny_taxi_raw['store_and_fwd_flag'].value_counts()
```

Out[5]:
```
N    2339340
Y      27370
Name: store_and_fwd_flag, dtype: int64
```

In [6]:
```
#drop the na values in the dataframe. there are fewer NA values, so we could drop them.
ny_taxi=ny_taxi_raw.dropna(how='any')
ny_taxi
```

Out[6]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwc
0	1	2021-05-01 00:37:18	2021-05-01 00:41:07	2.0	0.70	1.0	
1	1	2021-05-01 00:43:01	2021-05-01 00:49:19	1.0	1.40	1.0	
2	1	2021-05-01 00:05:54	2021-05-01 00:31:46	1.0	5.70	1.0	
3	2	2021-05-01 00:08:21	2021-05-01 00:19:20	1.0	3.04	1.0	
4	2	2021-05-01 00:32:44	2021-05-01 00:48:44	1.0	4.04	1.0	
...	...	...	...	...	...	...	
2379808	2	2021-05-31 23:20:04	2021-05-31 23:20:52	1.0	0.13	1.0	
2379809	2	2021-05-31 23:35:23	2021-05-31 23:42:32	1.0	1.08	1.0	
2379810	2	2021-05-31 23:45:25	2021-06-01 00:03:38	1.0	5.23	1.0	
2379811	1	2021-05-31 23:10:46	2021-05-31 23:27:35	1.0	4.30	1.0	
2379812	2	2021-05-31 23:05:56	2021-05-31 23:11:35	1.0	1.06	1.0	

2366683 rows × 19 columns

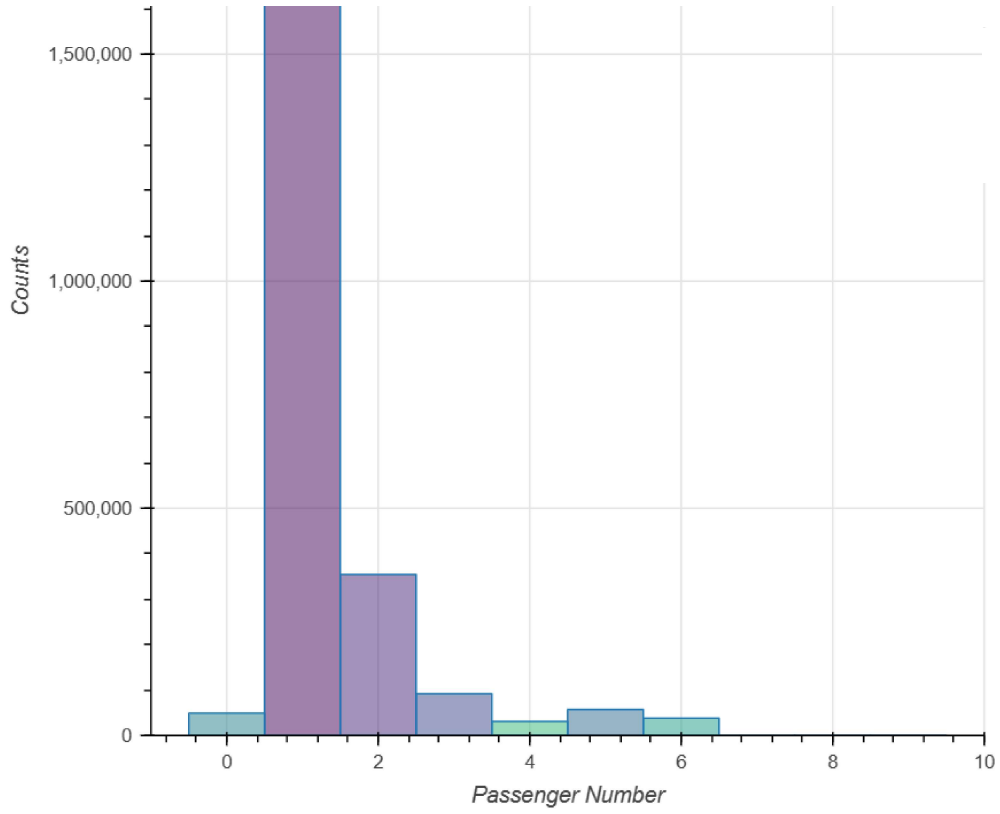
In [7]:
```
#Displaying in a Jupyter notebook
output_notebook()
```

BokehJS 2.4.1 successfully loaded.

In [8]:
```
# what is the count frequency of passenger number in the taxi trips?
passenger_count=ny_taxi['passenger_count'].value_counts()
passenger_number=np.array(passenger_count.index)
counts=np.array(passenger_count)
counts
```

Out[8]:
```
array([1745265,  354411,   91769,   57100,   49322,   37904,   30900,
             5,       4,       3], dtype=int64)
```

In [9]:
```
#color value of the bars
fill_color =bokeh.palettes.viridis(10)

# create ColumnDataSource based on a dict
source = ColumnDataSource(data=dict(passenger_number=passenger_number,counts=counts,fill_color=fill_colo

# create a new plot with a title and axis labels
p1 = figure(title="Passenger Counts", x_axis_label="Passenger Number", y_axis_label="Counts",y_range=[0,

#format axes ticks
p1.yaxis[0].formatter = NumeralTickFormatter(format="0,0")

# add a bar
p1.vbar(x='passenger_number', top='counts',source=source,fill_color='fill_color',fill_alpha=0.5)

# show the results
show(p1)
```



Most of the time there are 1 or 2 passengers.

In [10]:
```
#what are the first 10 frequent pickup locations?
pick_up_locations_count=ny_taxi['PULocationID'].value_counts()[ny_taxi['PULocationID'].value_counts()>=6

#change the location_id to categorical data
location_id=list(map(str,pick_up_locations_count.index))

#use the palette for color filing
fill_color_2 =bokeh.palettes.Category20c[10]
# create ColumnDataSource from Dict

pick_up_locations_count=np.array(pick_up_locations_count)
source2=ColumnDataSource(data=dict(location_id=location_id,pick_up_locations_count=pick_up_locations_cou

#reverse the order of y axis
y_cat_range = FactorRange(factors=list(reversed(location_id)))

# create a new plot with a title and axis labels
p2 = figure(title="Pick Up Locations Frequency", y_range=y_cat_range,y_axis_label="Location Id", x_axis_

#format axes ticks
p2.xaxis[0].formatter = NumeralTickFormatter(format="0,0")

# add a horizontal bar
p2.hbar(y='location_id',right='pick_up_locations_count',source=source2,height=0.7,fill_color='fill_color
```

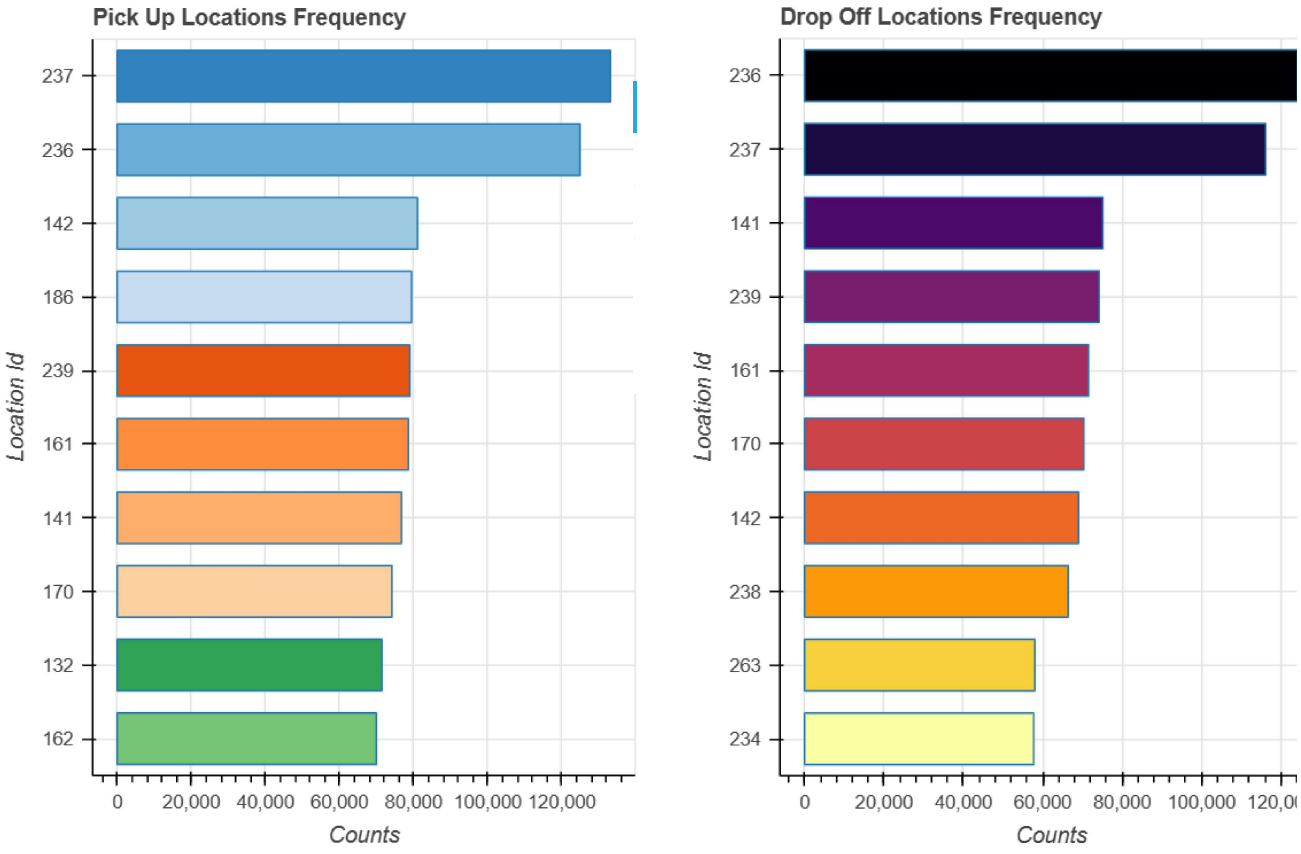
Out[10]: **GlyphRenderer**(id = '1138', ...)

In [11]:
```
#what are the first 10 frequent drop off locations?
drop_off_locations_count=ny_taxi['DOLocationID'].value_counts()[ny_taxi['DOLocationID'].value_counts()>=

#change the location_id to categorical data
location_id_1=list(map(str,drop_off_locations_count.index))

#use the palette for color filing
fill_color_2 =bokeh.palettes.Inferno[10]
```

```
# create ColumnDataSource from Dict
drop_off_locations_count=np.array(drop_off_locations_count)
source4=ColumnDataSource(data=dict(location_id=location_id_1,drop_off_locations_count=drop_off_locations_

#reverse the order of y axis
y_cat_range = FactorRange(factors=list(reversed(location_id_1)))

# create a new plot with a title and axis labels
p4 = figure(title="Drop Off Locations Frequency", y_range=y_cat_range,y_axis_label="Location Id", x_axis_

#format axes ticks
p4.xaxis[0].formatter = NumeralTickFormatter(format="0,0")

# add a horizontal bar
p4.hbar(y='location_id',right='drop_off_locations_count',source=source4,height=0.7,fill_color='fill_colo
```

Out[11]: **GlyphRenderer**(id = '1181', ...)

In [12]:
```
# show the results
show(row(p2,p4))
```

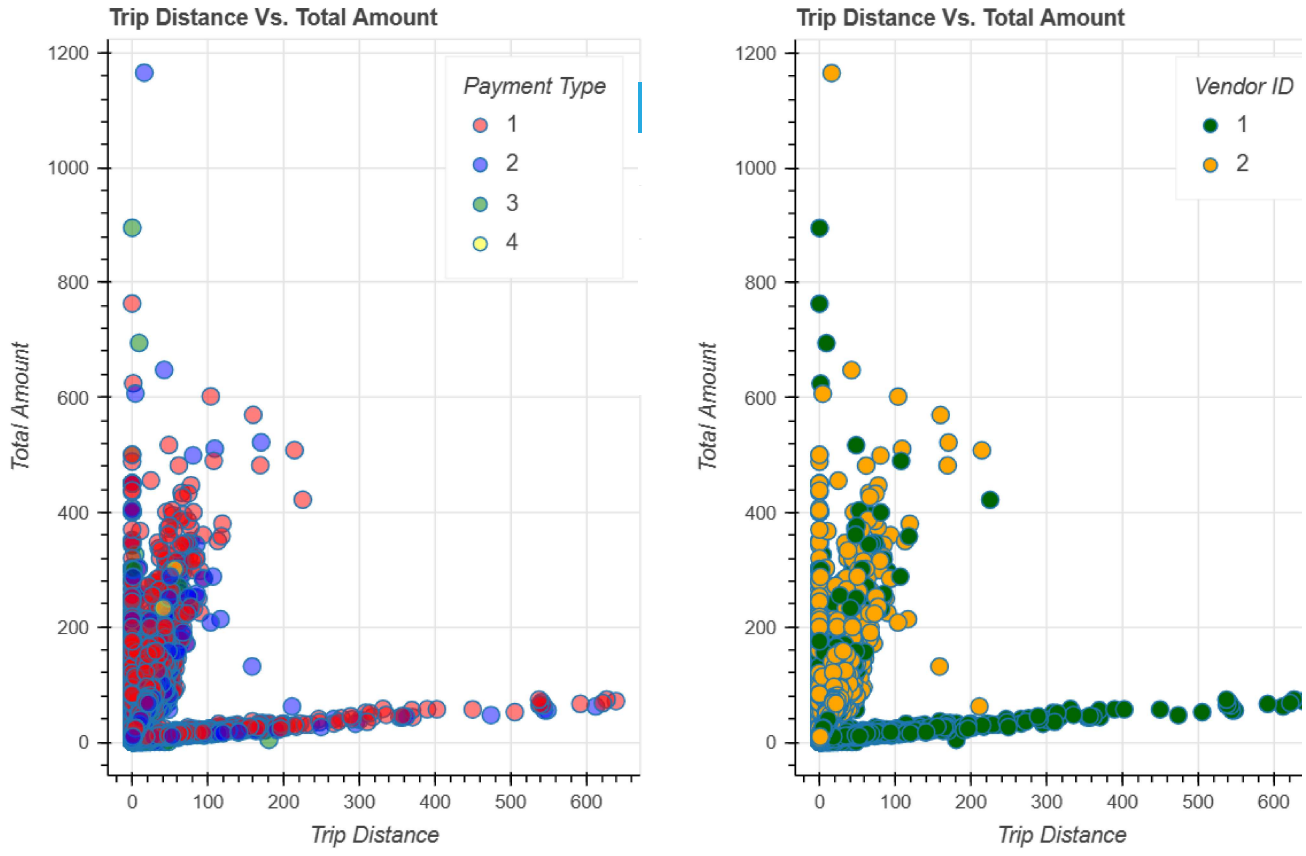


Locations with ID 236 and 237 are the most frequent pick-up and drop-off locations.

In [13]:
```
# explore the relationship between trip_distance and total_amount
trip_distance=np.array(ny_taxi.trip_distance)
total_amount=np.array(ny_taxi.total_amount)

ny_taxi.payment_type=ny_taxi.payment_type.astype(str)
ny_taxi.VendorID=ny_taxi.VendorID.astype(str)
index_cmap = factor_cmap('payment_type', palette=['red', 'blue','green','yellow'],
                         factors=sorted(ny_taxi.payment_type.unique()))
index_cmap_1 = factor_cmap('VendorID', palette=['darkgreen', 'orange'],
                         factors=sorted(ny_taxi.VendorID.unique()))
```

```
# create ColumnDataSource based on a DataFrame
source3 = ColumnDataSource(data=ny_taxi)

# create a new plot with a title and axis labels
p3 = figure(title="Trip Distance Vs. Total Amount", y_axis_label="Total Amount", x_axis_label="Trip Dist
p5 = figure(title="Trip Distance Vs. Total Amount", y_axis_label="Total Amount", x_axis_label="Trip Dist
# add a circle
p3.scatter(y="total_amount", x="trip_distance",source=source3,size=10,fill_color=index_cmap,fill_alpha=0
p5.scatter(y="total_amount", x="trip_distance",source=source3,size=10,fill_color=index_cmap_1,legend_gro
# add a title to your legend
p3.legend.title = "Payment Type"
p5.legend.title = "Vendor ID"
# show the results
show(row(p3,p5))
```

```
C:\Users\Rachel\anaconda3\lib\site-packages\pandas\core\generic.py:5516: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.ht
ml#returning-a-view-versus-a-copy
  self[name] = value
```



We could see that Vendor 1 has more trips with longer distances. In general, there are two types of trips for vendor 1.Type I trips are similar to those of vendor 2, type II trips are exclusive to vendor 1 which have longer distances and less total amount comparing to type I.