Our aim was to try to analyze the prices of these ride-sharing apps and try to figure out what factors are driving the demand. Do Mondays have more demand than Sunday at 9 am? Do people avoid cabs on a sunny day? Was there a Red Sox match at Fenway that caused more people coming in? We have provided a small dataset as well as a mechanism to collect more data. We would love to see more conclusions drawn.

In [1]:
```
import pandas as pd
from sklearn.linear_model import LinearRegression
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import itertools
import gc
import os
import sys
%matplotlib inline
```

In [2]:
```
cab_data = pd.read_csv("cab_rides.csv")
weather_data = pd.read_csv("weather.csv")
```

In [3]:
```
cab_data
```

Out[3]:

	distance	cab_type	time_stamp	destination	source	price	surge_multiplier	id	product_id	name
0	0.44	Lyft	1544952607890	North Station	Haymarket Square	5.0	1.0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	lyft_line	Shared
1	0.44	Lyft	1543284023677	North Station	Haymarket Square	11.0	1.0	4bd23055-6827-41c6-b23b-3c491f24e74d	lyft_premier	Lux
2	0.44	Lyft	1543366822198	North Station	Haymarket Square	7.0	1.0	981a3613-77af-4620-a42a-0c0866077d1e	lyft	Lyft
3	0.44	Lyft	1543553582749	North Station	Haymarket Square	26.0	1.0	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	lyft_luxsuv	Lux Black XL
4	0.44	Lyft	1543463360223	North Station	Haymarket Square	9.0	1.0	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	lyft_plus	Lyft XL
...	...	...	...	...	...	...	...	...	...	...
693066	1.00	Uber	1543708385534	North End	West End	13.0	1.0	616d3611-1820-450a-9845-a9ff304a4842	6f72dfc5-27f1-42e8-84db-ccc7a75f6969	UberXL
693067	1.00	Uber	1543708385534	North End	West End	9.5	1.0	633a3fc3-1f86-4b9e-9d48-2b7132112341	55c66225-fbe7-4fd5-9072-eab1ece5e23e	UberX
693068	1.00	Uber	1543708385534	North End	West End	NaN	1.0	64d451d0-639f-47a4-9b7c-6fd92fbd264f	8cf7e821-f0d3-49c6-8eba-e679c0ebcf6a	Taxi
693069	1.00	Uber	1543708385534	North End	West End	27.0	1.0	727e5f07-a96b-4ad1-a2c7-9abc3ad55b4e	6d318bcc-22a3-4af6-bddd-b409bfce1546	Black SUV
693070	1.00	Uber	1543708385534	North End	West End	10.0	1.0	e7fdc087-fe86-40a5-a3c3-3b2a8badcbda	997acbb5-e102-41e1-b155-9df7de0a73f2	UberPool

693071 rows × 10 columns

In [4]:
```
weather_data
```

Out[4]:

	temp	location	clouds	pressure	rain	time_stamp	humidity	wind
0	42.42	Back Bay	1.00	1012.14	0.1228	1545003901	0.77	11.25
1	42.43	Beacon Hill	1.00	1012.15	0.1846	1545003901	0.76	11.32
2	42.50	Boston University	1.00	1012.15	0.1089	1545003901	0.76	11.07
3	42.11	Fenway	1.00	1012.13	0.0969	1545003901	0.77	11.09
4	43.13	Financial District	1.00	1012.14	0.1786	1545003901	0.75	11.49
...	...	...	...	...	...	...	...	...
6271	44.72	North Station	0.89	1000.69	NaN	1543819974	0.96	1.52
6272	44.85	Northeastern University	0.88	1000.71	NaN	1543819974	0.96	1.54
6273	44.82	South Station	0.89	1000.70	NaN	1543819974	0.96	1.54
6274	44.78	Theatre District	0.89	1000.70	NaN	1543819974	0.96	1.54
6275	44.69	West End	0.89	1000.70	NaN	1543819974	0.96	1.52

6276 rows × 8 columns

In [5]:
```
cab_data.columns
```

Out[5]:
```
Index(['distance', 'cab_type', 'time_stamp', 'destination', 'source', 'price',
       'surge_multiplier', 'id', 'product_id', 'name'],
      dtype='object')
```

In [6]:
```
weather_data.columns
```

Out[6]:
```
Index(['temp', 'location', 'clouds', 'pressure', 'rain', 'time_stamp',
       'humidity', 'wind'],
      dtype='object')
```

In [7]:
```
import datetime
cab_data['datetime'] = pd.to_datetime(cab_data['time_stamp'])
cab_data
weather_data['date_time'] = pd.to_datetime(weather_data['time_stamp'])
```

In [8]:
```
#Checking the shape of the data
cab_data.shape
```

Out[8]:
```
(693071, 11)
```

In [9]:
```
weather_data.shape
```

Out[9]:
```
(6276, 9)
```

In [10]:
```
#Complete description of the data
cab_data.describe()
```

Out[10]:

	distance	time_stamp	price	surge_multiplier
count	693071.000000	6.930710e+05	637976.000000	693071.000000
mean	2.189430	1.544046e+12	16.545125	1.013870
std	1.138937	6.891925e+08	9.324359	0.091641
min	0.020000	1.543204e+12	2.500000	1.000000
25%	1.280000	1.543444e+12	9.000000	1.000000
50%	2.160000	1.543737e+12	13.500000	1.000000
75%	2.920000	1.544828e+12	22.500000	1.000000
max	7.860000	1.545161e+12	97.500000	3.000000

In [11]:
```
weather_data.describe()
```

Out[11]:

	temp	clouds	pressure	rain	time_stamp	humidity	wind
count	6276.000000	6276.000000	6276.000000	894.000000	6.276000e+03	6276.000000	6276.000000
mean	39.090475	0.677777	1008.445209	0.057652	1.543857e+09	0.763985	6.802812
std	6.022055	0.314284	12.870775	0.100758	6.659340e+05	0.127340	3.633466
min	19.620000	0.000000	988.250000	0.000200	1.543204e+09	0.450000	0.290000
25%	36.077500	0.440000	997.747500	0.004900	1.543387e+09	0.670000	3.517500
50%	40.130000	0.780000	1007.660000	0.014850	1.543514e+09	0.760000	6.570000
75%	42.832500	0.970000	1018.480000	0.060925	1.544691e+09	0.890000	9.920000
max	55.410000	1.000000	1035.120000	0.780700	1.545159e+09	0.990000	18.180000

In [12]:
```
#Concating the weather and cab dataset
a = pd.concat([cab_data, weather_data])
```

In [13]:
```
a.head()
```

Out[13]:

	distance	cab_type	time_stamp	destination	source	price	surge_multiplier	id	product_id	name	datetime	temp	location	clouds	pressure	rain
0	0.44	Lyft	1544952607890	North Station	Haymarket Square	5.0	1.0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	lyft_line	Shared	1970-01-01 00:25:44.952607890	NaN	NaN	NaN	NaN	NaN
1	0.44	Lyft	1543284023677	North Station	Haymarket Square	11.0	1.0	4bd23055-6827-41c6-b23b-3c491f24e74d	lyft_premier	Lux	1970-01-01 00:25:43.284023677	NaN	NaN	NaN	NaN	NaN
2	0.44	Lyft	1543366822198	North Station	Haymarket Square	7.0	1.0	981a3613-77af-4620-a42a-0c0866077d1e	lyft	Lyft	1970-01-01 00:25:43.366822198	NaN	NaN	NaN	NaN	NaN
3	0.44	Lyft	1543553582749	North Station	Haymarket Square	26.0	1.0	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	lyft_luxsuv	Lux Black XL	1970-01-01 00:25:43.553582749	NaN	NaN	NaN	NaN	NaN
4	0.44	Lyft	1543463360223	North Station	Haymarket Square	9.0	1.0	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	lyft_plus	Lyft XL	1970-01-01 00:25:43.463360223	NaN	NaN	NaN	NaN	NaN

In [14]:
```
#getting particular day and time
a['day'] = a.date_time.dt.day
a['hour'] = a.date_time.dt.hour
```

In [15]:
```
a.head()
```

Out[15]:

	distance	cab_type	time_stamp	destination	source	price	surge_multiplier	id	product_id	name	...	temp	location	clouds	pressure	rain	humidity	wind
0	0.44	Lyft	1544952607890	North Station	Haymarket Square	5.0	1.0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	lyft_line	Shared	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	0.44	Lyft	1543284023677	North Station	Haymarket Square	11.0	1.0	4bd23055-6827-41c6-b23b-3c491f24e74d	lyft_premier	Lux	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	0.44	Lyft	1543366822198	North Station	Haymarket Square	7.0	1.0	981a3613-77af-4620-a42a-0c0866077d1e	lyft	Lyft	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	0.44	Lyft	1543553582749	North Station	Haymarket Square	26.0	1.0	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	lyft_luxsuv	Lux Black XL	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	0.44	Lyft	1543463360223	North Station	Haymarket Square	9.0	1.0	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	lyft_plus	Lyft XL	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 21 columns

In [16]:
```
#Most of the time and dates have NAN values so we need to solve that
```

In [17]:
```
#Checking the total null values
a.isnull().sum()
```

Out[17]:
```
distance                6276
cab_type                6276
time_stamp                 0
destination             6276
source                  6276
price                  61371
surge_multiplier        6276
id                      6276
product_id              6276
name                    6276
datetime                6276
temp                  693071
location              693071
clouds                693071
pressure              693071
rain                  698453
humidity              693071
wind                  693071
date_time             693071
day                   693071
hour                  693071
dtype: int64
```

In [18]:
```
#Filling the null values with 0
a.fillna(0,inplace=True)
```

In [19]:
```
a.head()
```

Out[19]:

	distance	cab_type	time_stamp	destination	source	price	surge_multiplier	id	product_id	name	...	temp	location	clouds	pressure	rain	humidity	wind
0	0.44	Lyft	1544952607890	North Station	Haymarket Square	5.0	1.0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	lyft_line	Shared	...	0.0	0	0.0	0.0	0.0	0.0	0.0
1	0.44	Lyft	1543284023677	North Station	Haymarket Square	11.0	1.0	4bd23055-6827-41c6-b23b-3c491f24e74d	lyft_premier	Lux	...	0.0	0	0.0	0.0	0.0	0.0	0.0
2	0.44	Lyft	1543366822198	North Station	Haymarket Square	7.0	1.0	981a3613-77af-4620-a42a-0c0866077d1e	lyft	Lyft	...	0.0	0	0.0	0.0	0.0	0.0	0.0
3	0.44	Lyft	1543553582749	North Station	Haymarket Square	26.0	1.0	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	lyft_luxsuv	Lux Black XL	...	0.0	0	0.0	0.0	0.0	0.0	0.0
4	0.44	Lyft	1543463360223	North Station	Haymarket Square	9.0	1.0	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	lyft_plus	Lyft XL	...	0.0	0	0.0	0.0	0.0	0.0	0.0

5 rows × 21 columns

In [20]:
```
a.columns
```

Out[20]:
```
Index(['distance', 'cab_type', 'time_stamp', 'destination', 'source', 'price',
       'surge_multiplier', 'id', 'product_id', 'name', 'datetime', 'temp',
       'location', 'clouds', 'pressure', 'rain', 'humidity', 'wind',
       'date_time', 'day', 'hour'],
      dtype='object')
```

In [21]:
```
#Group based on the columns
a.groupby('cab_type').count()
```

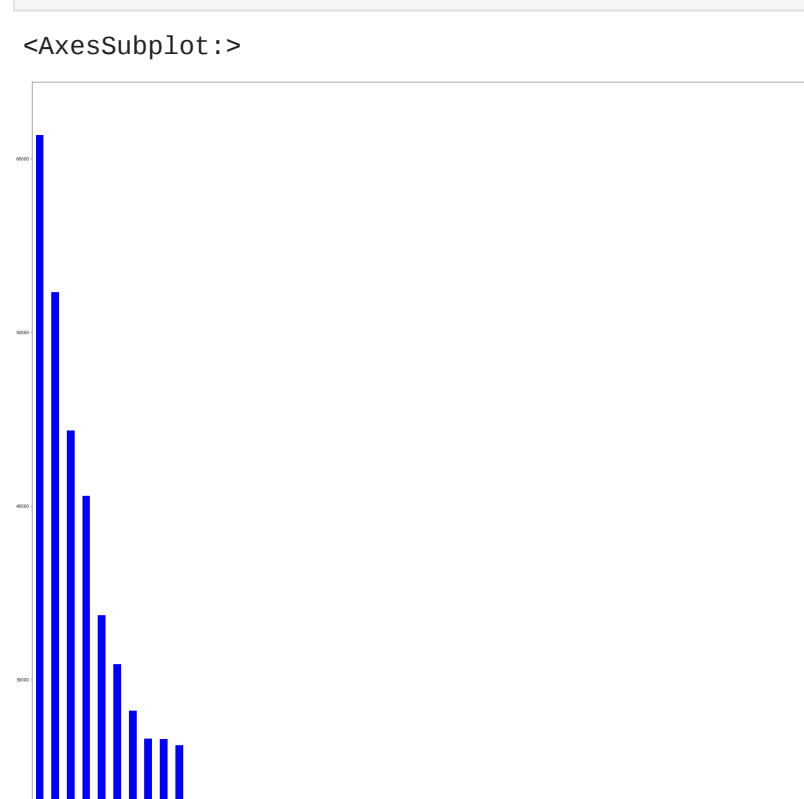
Out[21]:

cab_type	distance	time_stamp	destination	source	price	surge_multiplier	id	product_id	name	datetime	temp	location	clouds	pressure	rain	humidity	wind
0	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276	6276
Lyft	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408	307408
Uber	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663	385663

In [22]:
```
#So we have '307408' for Lyft cab and '385663' for Uber cab
```

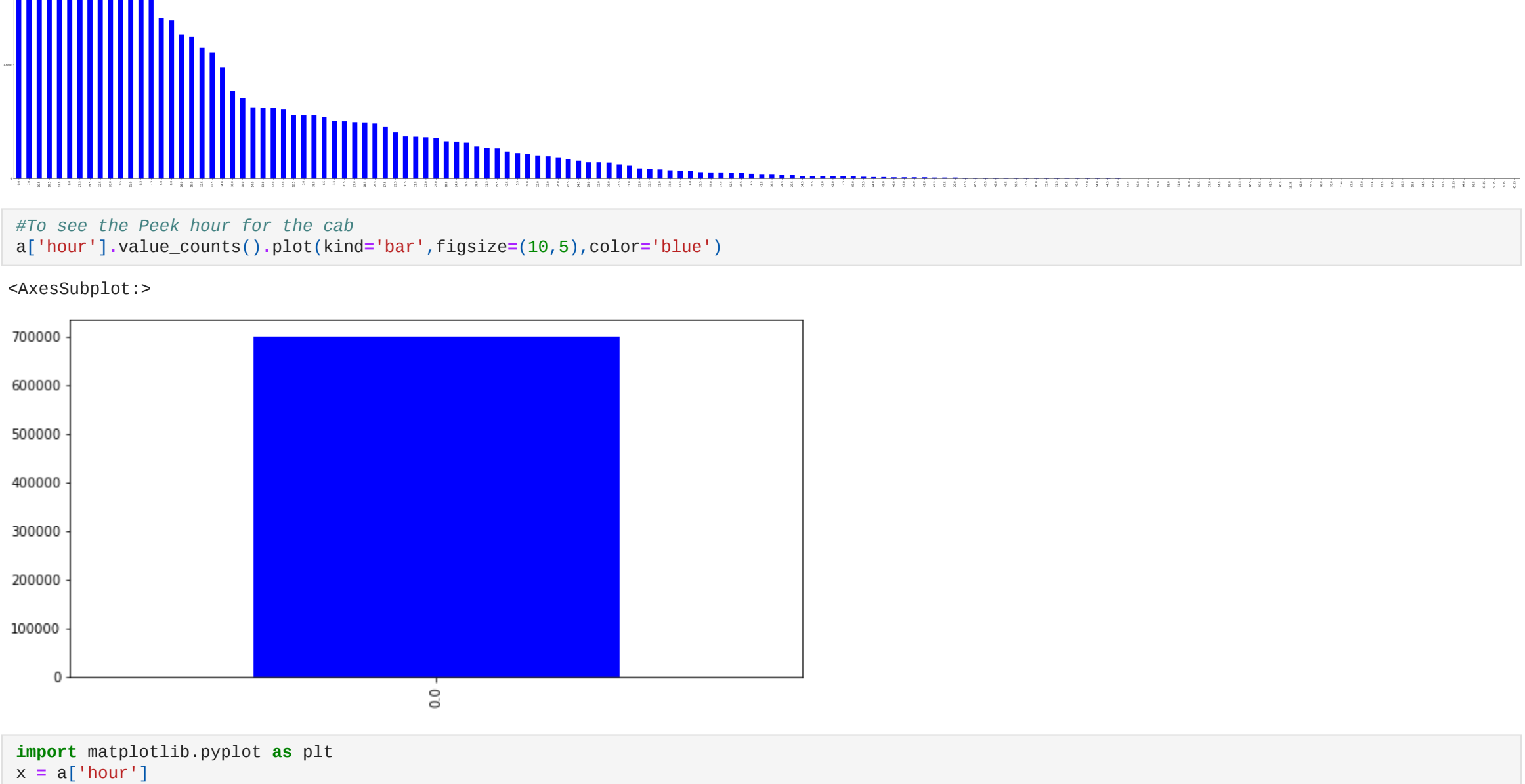
In [23]:
```
#Visualizing the data
a.groupby('cab_type').count().plot.bar()
```

Out[23]:
```
<AxesSubplot:xlabel='cab_type'>
```



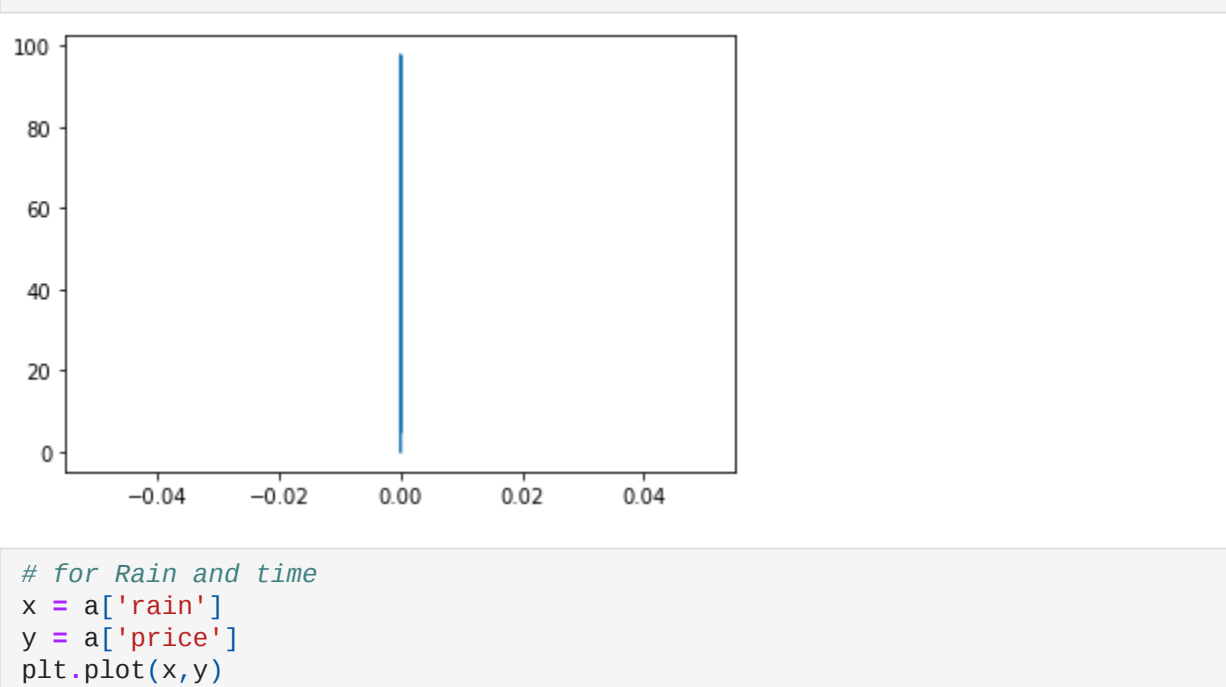
In [24]:
```
#To see what is fluction of the time
a['price'].value_counts().plot(kind='bar', figsize=(100,50), color='blue')
```

Out[24]:
```
<AxesSubplot:>
```

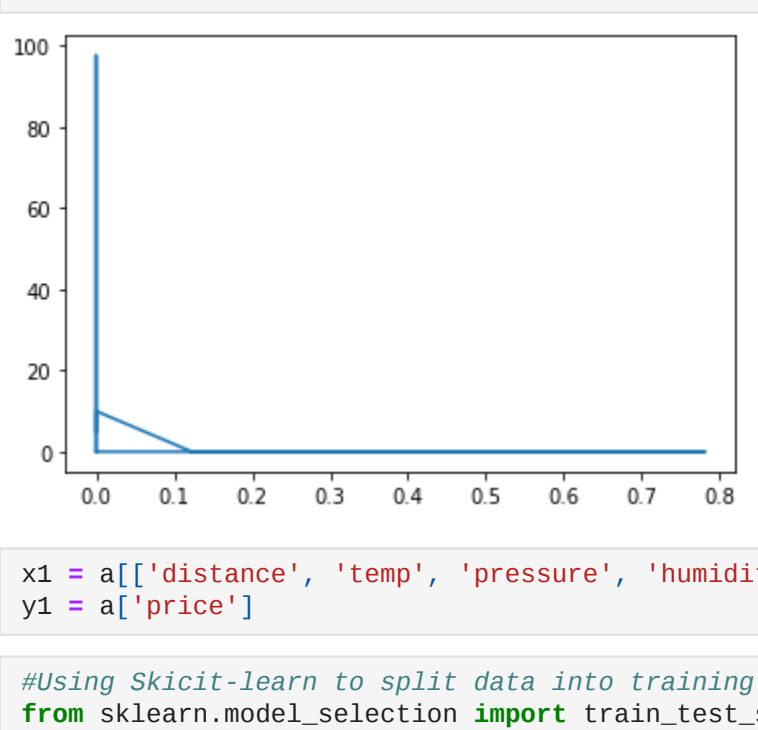


In [25]:
```
#To see the Peek hour for the cab
a['hour'].value_counts().plot(kind='bar',figsize=(10,5),color='blue')
```

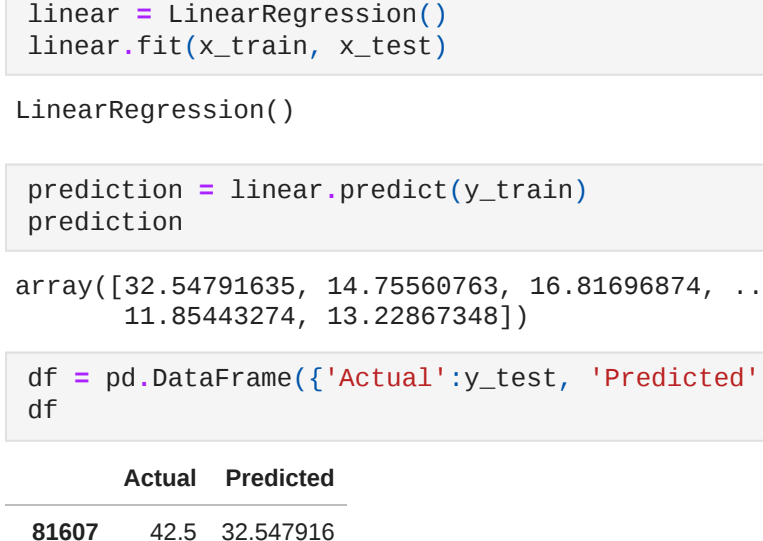
Out[25]:
```
<AxesSubplot:>
```



In [26]:
```
import matplotlib.pyplot as plt
x = a['hour']
y = a['price']
plt.plot(x,y)
plt.show()
```



In [27]:
```
# for Rain and time
x = a['rain']
y = a['price']
plt.plot(x,y)
plt.show()
```



In [28]:
```
x1 = a[['distance', 'temp', 'pressure', 'humidity', 'wind', 'rain', 'day', 'hour', 'surge_multiplier', 'clouds']]
y1 = a['price']
```

In [29]:
```
#Using Skicit-learn to split data into training and testing sets
from sklearn.model_selection import train_test_split

#Split the data into training and testing
x_train, y_train, x_test, y_test = train_test_split(x1, y1, test_size = 0.25, random_state = 42) #Random state takes the values randomly
```

In [30]:
```
linear = LinearRegression()
linear.fit(x_train, x_test)
```

Out[30]:
```
LinearRegression()
```

In [31]:
```
prediction = linear.predict(y_train)
prediction
```

Out[31]:
```
array([32.54791635, 14.75560763, 16.81696874, ..., 11.82898384,
       11.85443274, 13.22867348])
```

In [32]:
```
df = pd.DataFrame({'Actual':y_test, 'Predicted':prediction})
df
```

Out[32]:

	Actual	Predicted
81607	42.5	32.547916
354	0.0	14.755608
471801	32.5	16.816969
407645	27.5	10.480192
474760	27.5	10.556539
...	...	...
538489	7.5	10.709232
579511	13.5	15.519075
5421	9.0	11.828984
279982	8.0	11.854433
236315	27.5	13.228673

174837 rows × 2 columns

In [33]:
```
#See the visualization for accurate and predict

df1 = df.head(25)
df1.plot(kind = 'bar', figsize =(26,10))
plt.grid(which='major', linestyle = '-', linewidth = '0.5', color = 'green')
plt.grid(which='minor', linestyle = ':', linewidth = '0.5', color = 'black')
plt.show()
```


In [ ]: