In [33]:
```
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [34]:
```
#import dataset
df=pd.read_csv(r"E:\154\uber - uber.csv")
df
```

Out[34]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dr
0	24238194	2015-05-07 19:52:06	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	
1	27835199	2009-07-17 20:04:56	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	
2	44984355	2009-08-24 21:45:00	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	
3	25894730	2009-06-26 08:22:21	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	
4	17610152	2014-08-28 17:47:00	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	
...	...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739367	
199996	16382965	2014-03-14 01:09:00	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736837	
199997	27804658	2009-06-29 00:42:00	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756487	
199998	20259894	2015-05-20 14:56:25	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725452	
199999	11951496	2010-05-15 04:08:00	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720077	

200000 rows × 9 columns

In [ ]:

In [35]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   Unnamed: 0         200000 non-null  int64  
 1   key                200000 non-null  object 
 2   fare_amount        200000 non-null  float64
 3   pickup_datetime    200000 non-null  object 
 4   pickup_longitude   200000 non-null  float64
 5   pickup_latitude    200000 non-null  float64
 6   dropoff_longitude  199999 non-null  float64
 7   dropoff_latitude   199999 non-null  float64
 8   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

In [36]:
```
#to display top 5 rows
df.head()
```

Out[36]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_
0	24238194	2015-05-07 19:52:06	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-
1	27835199	2009-07-17 20:04:56	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-
2	44984355	2009-08-24 21:45:00	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-
3	25894730	2009-06-26 08:22:21	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-
4	17610152	2014-08-28 17:47:00	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-

Data cleaning and Pre-Processing

In [37]:
```
#To find null values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 9 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   Unnamed: 0         200000 non-null  int64  
 1   key                200000 non-null  object 
 2   fare_amount        200000 non-null  float64
 3   pickup_datetime    200000 non-null  object 
 4   pickup_longitude   200000 non-null  float64
 5   pickup_latitude    200000 non-null  float64
 6   dropoff_longitude  199999 non-null  float64
 7   dropoff_latitude   199999 non-null  float64
 8   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(2), object(2)
memory usage: 13.7+ MB
```

In [38]:
```
# To display summary of statistics
df.describe()
```

Out[38]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff
count	2.000000e+05	200000.000000	200000.000000	200000.000000	199999.000000	19999
mean	2.771250e+07	11.359955	-72.527638	39.935885	-72.525292	3
std	1.601382e+07	9.901776	11.437787	7.720539	13.117408	
min	1.000000e+00	-52.000000	-1340.648410	-74.015515	-3356.666300	-88
25%	1.382535e+07	6.000000	-73.992065	40.734796	-73.991407	4
50%	2.774550e+07	8.500000	-73.981823	40.752592	-73.980093	4
75%	4.155530e+07	12.500000	-73.967153	40.767158	-73.963659	4
max	5.542357e+07	499.000000	57.418457	1644.421482	1153.572603	87

In [39]:
```
#To Display column heading
df.columns
```

Out[39]:
```
Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
       'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
       'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

EDA and VISUALIZATION

In [ ]:
```
sns.pairplot(df)
```

In [ ]:
```
sns.distplot(df['passenger_count'])
```

In [26]:
```
df1=df[['MonthYear', 'Time index', 'StoreID', 'City', 'Dept_ID', 'HoursOwn', '
```

Plot Using Heat Map

In [ ]:
```
sns.heatmap(df1.corr())
```

To Train The Model-Model Building

we are going to train Linera Regression Model;We need to split out data into two variables x and y where x is independent variable(input) and y is dependent on x(output) we could ignore address column as it required for our model

In [29]:
```
df1=df[['MonthYear', 'Time index', 'StoreID', 'City', 'Dept_ID', 'HoursOwn', '

y=df1['Sales units']
```

To Split my dataset into training and test data

In [30]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [31]:
```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-31-ea6401f380cc> in <module>
      1 from sklearn.linear_model import LinearRegression
      2 lr= LinearRegression()
----> 3 lr.fit(x_train,y_train)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_base.py in f
it(self, X, y, sample_weight)
    516         accept_sparse = False if self.positive else ['csr', 'csc', 'c
oo']
    517
--> 518         X, y = self._validate_data(X, y, accept_sparse=accept_sparse,
    519                                    y_numeric=True, multi_output=True)
    520

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py in _validate_data
(self, X, y, reset, validate_separately, **check_params)
    431                 y = check_array(y, **check_y_params)
    432             else:
--> 433                 X, y = check_X_y(X, y, **check_params)
    434             out = X, y
    435

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inn
er_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in che
ck_X_y(X, y, accept_sparse, accept_large_sparse, dtype, order, copy, force_al
l_finite, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_f
eatures, y_numeric, estimator)
    812         raise ValueError("y cannot be None")
    813
--> 814     X = check_array(X, accept_sparse=accept_sparse,
    815                     accept_large_sparse=accept_large_sparse,
    816                     dtype=dtype, order=order, copy=copy,

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inn
er_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in che
ck_array(array, accept_sparse, accept_large_sparse, dtype, order, copy, force
_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, es
timator)
    614                     array = array.astype(dtype, casting="unsafe", cop
y=False)
    615                 else:
```

```
--> 616                     array = np.asarray(array, order=order, dtype=dtyp
e)
    617             except ComplexWarning as complex_warning:
    618                 raise ValueError("Complex data not supported\n"

C:\ProgramData\Anaconda3\lib\site-packages\numpy\core\_asarray.py in asarray
(a, dtype, order, like)
    100         return _asarray_with_like(a, dtype=dtype, order=order, like=l
ike)
    101
--> 102     return array(a, dtype, copy=False, order=order)
    103
    104


C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py in __array_
_(self, dtype)
   1897
   1898     def __array__(self, dtype=None) -> np.ndarray:
-> 1899         return np.asarray(self._values, dtype=dtype)
   1900
   1901     def __array_wrap__(

C:\ProgramData\Anaconda3\lib\site-packages\numpy\core\_asarray.py in asarray
(a, dtype, order, like)
    100         return _asarray_with_like(a, dtype=dtype, order=order, like=l
ike)
    101
--> 102     return array(a, dtype, copy=False, order=order)
    103
    104

ValueError: could not convert string to float: '- - - -'
```

In [32]:
```
lr.intercept_
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-32-e33b6092e6ca> in <module>
----> 1 lr.intercept_

AttributeError: 'LinearRegression' object has no attribute 'intercept_'
```

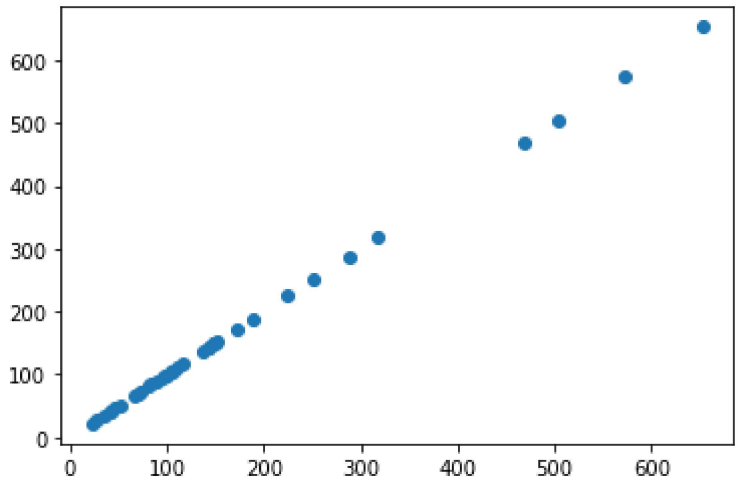
In [17]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[17]:

	Co-efficient
Impressions	6.000135e-16
From Home	-7.460040e-16
From Hashtags	-7.173340e-16
From Explore	-6.060430e-16
From Other	3.668616e-15
Saves	1.000000e+00
Comments	8.391696e-16
Shares	-1.180882e-16
Likes	-2.075960e-18
Profile Visits	3.936108e-16
Follows	-3.795219e-16

In [18]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[18]:
```
<matplotlib.collections.PathCollection at 0x199e0f55400>
```



In [19]:
```
lr.score(x_test,y_test)
```

Out[19]:
```
1.0
```

In [ ]: