In [39]:

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as pl
```

In [40]:

```python
df = pd.read_csv("uber.csv")
```

In [41]:

```python
df.head()
```

Out[41]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passen
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217	
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325	
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647	
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349	
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247	

```
In [42]: 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 [43]: df.columns
```

```
Out[43]: Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
               'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
               'dropoff_latitude', 'passenger_count'],
              dtype='object')
```

```
In [44]: df = df.drop(['Unnamed: 0', 'key'], axis= 1)
In [45]: df.head()
```

Out[45]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.723217	1
1	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.750325	1
2	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.772647	1
3	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.803349	3
4	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.761247	5

```
In [46]: df.shape

Out[46]: (200000, 7)
```

```
In [47]: df.dtypes

Out[47]: fare_amount          float64
         pickup_datetime       object
         pickup_longitude     float64
         pickup_latitude      float64
         dropoff_longitude    float64
         dropoff_latitude     float64
         passenger_count        int64
         dtype: object
```

In [48]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200000 entries, 0 to 199999
Data columns (total 7 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   fare_amount        200000 non-null  float64
 1   pickup_datetime    200000 non-null  object 
 2   pickup_longitude   200000 non-null  float64
 3   pickup_latitude    200000 non-null  float64
 4   dropoff_longitude  199999 non-null  float64
 5   dropoff_latitude   199999 non-null  float64
 6   passenger_count    200000 non-null  int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 10.7+ MB
```

In [49]:
```
df.describe() #To get statistics of each columns
```

Out[49]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	200000.000000	200000.000000	200000.000000	199999.000000	199999.000000	200000.000000
mean	11.359955	-72.527638	39.935885	-72.525292	39.923890	1.684535
std	9.901776	11.437787	7.720539	13.117408	6.794829	1.385997
min	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.985513	0.000000
25%	6.000000	-73.992065	40.734796	-73.991407	40.733823	1.000000
50%	8.500000	-73.981823	40.752592	-73.980093	40.753042	1.000000
75%	12.500000	-73.967154	40.767158	-73.963658	40.768001	2.000000
max	499.000000	57.418457	1644.421482	1153.572603	872.697628	208.000000

In [50]:

```
df.isnull().sum()
```

Out[50]:

```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [5]:

```
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(),inplace = True)
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].median(),inplace = True)
```

In [51]:

```
df.isnull().sum()
```

Out[51]:

```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [8]:

```
df.isnull().sum()
```

Out[8]:

```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

In [52]:

```
df.dtypes
```

Out[52]:

```
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

In [53]:

```
df.pickup_datetime = pd.to_datetime(df.pickup_datetime, errors='coerce')
df.dtypes
```

Out[53]:

```
fare_amount                      float64
pickup_datetime      datetime64[ns, UTC]
pickup_longitude                 float64
pickup_latitude                  float64
dropoff_longitude                float64
dropoff_latitude                 float64
passenger_count                    int64
dtype: object
```

In [54]:
```
df= df.assign(hour = df.pickup_datetime.dt.hour,day= df.pickup_datetime.dt.day,month = df.pickup_datetime.dt.month
df.head()
```

Out[54]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month
0	7.5	2015-05-07 19:52:06+00:00	-73.999817	40.738354	-73.999512	40.723217	1	19	7	5
1	7.7	2009-07-17 20:04:56+00:00	-73.994355	40.728225	-73.994710	40.750325	1	20	17	7
2	12.9	2009-08-24 21:45:00+00:00	-74.005043	40.740770	-73.962565	40.772647	1	21	24	8
3	5.3	2009-06-26 08:22:21+00:00	-73.976124	40.790844	-73.965316	40.803349	3	8	26	6
4	16.0	2014-08-28 17:47:00+00:00	-73.925023	40.744085	-73.973082	40.761247	5	17	28	8

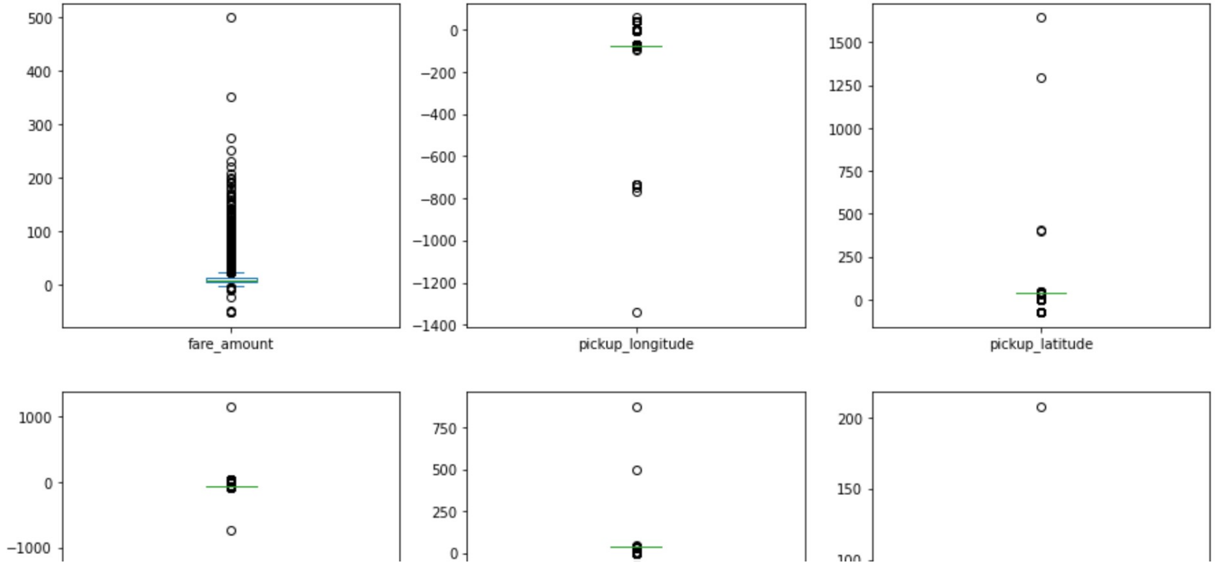
In [55]:
```
df = df.drop('pickup_datetime',axis=1)
```

In [56]:
```
df.dtypes
```

Out[56]:
```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
hour                   int64
day                    int64
month                  int64
year                   int64
dayofweek              int64
dtype: object
```

```
In [65]: df.plot(kind = "box",subplots =True ,layout = (4,3),figsize=(15,20))
```

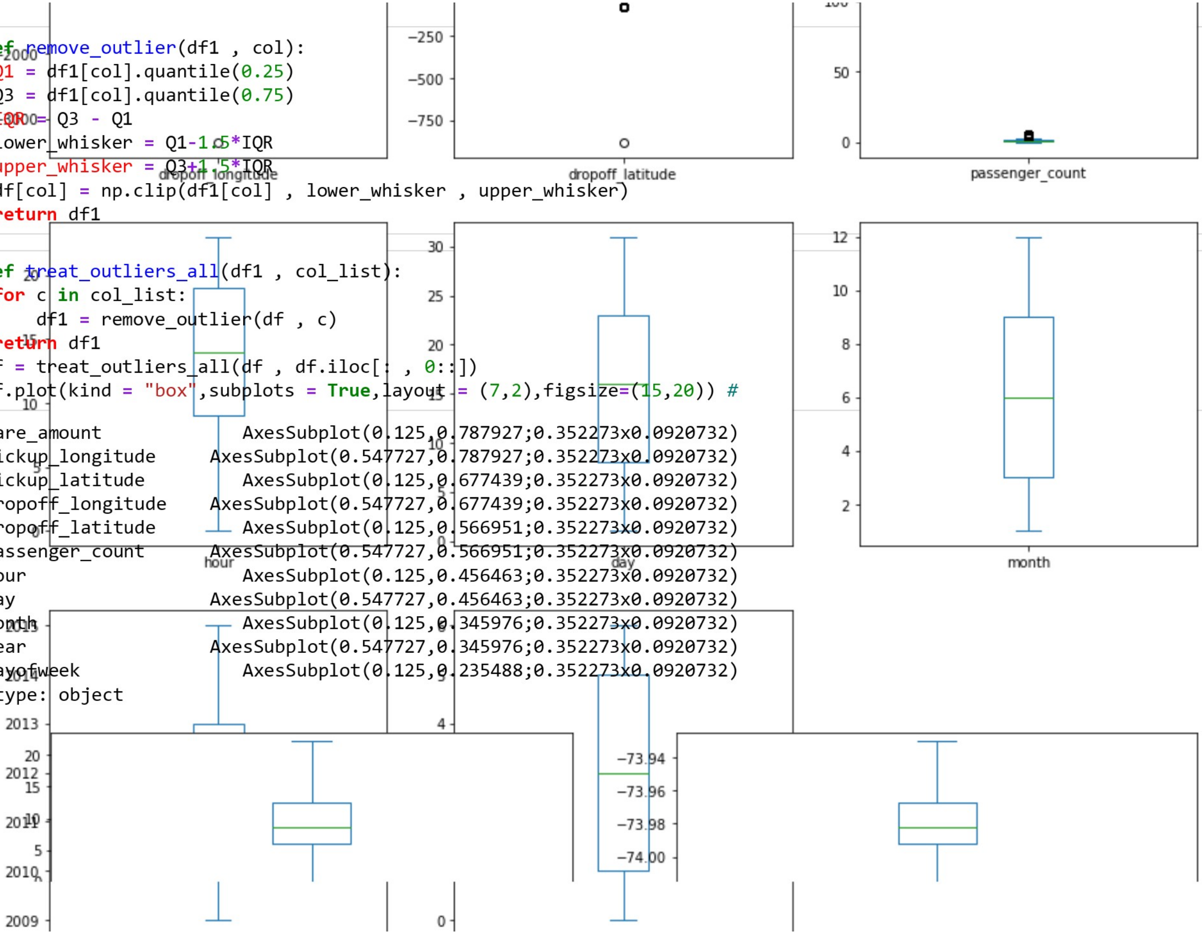
```
Out[65]: fare_amount            AxesSubplot(0.125,0.71587;0.227941x0.16413)
         pickup_longitude    AxesSubplot(0.398529,0.71587;0.227941x0.16413)
         pickup_latitude     AxesSubplot(0.672059,0.71587;0.227941x0.16413)
         dropoff_longitude     AxesSubplot(0.125,0.518913;0.227941x0.16413)
         dropoff_latitude   AxesSubplot(0.398529,0.518913;0.227941x0.16413)
         passenger_count    AxesSubplot(0.672059,0.518913;0.227941x0.16413)
         hour                  AxesSubplot(0.125,0.321957;0.227941x0.16413)
         day                AxesSubplot(0.398529,0.321957;0.227941x0.16413)
         month              AxesSubplot(0.672059,0.321957;0.227941x0.16413)
         year                     AxesSubplot(0.125,0.125;0.227941x0.16413)
         dayofweek             AxesSubplot(0.398529,0.125;0.227941x0.16413)
         dtype: object
```



In [15]:
```
def remove_outlier(df1 , col):
 Q1 = df1[col].quantile(0.25)
 Q3 = df1[col].quantile(0.75)
 IQR = Q3 - Q1
 lower_whisker = Q1-1.5*IQR
 upper_whisker = Q3+1.5*IQR
 df[col] = np.clip(df1[col] , lower_whisker , upper_whisker)
 return df1
```

In [16]:
```
def treat_outliers_all(df1 , col_list):
 for c in col_list:
     df1 = remove_outlier(df , c)
 return df1
df = treat_outliers_all(df , df.iloc[: , 0::])
df.plot(kind = "box",subplots = True,layout = (7,2),figsize=(15,20)) #
```

Out[16]:
```
fare_amount             AxesSubplot(0.125,0.787927;0.352273x0.0920732)
pickup_longitude     AxesSubplot(0.547727,0.787927;0.352273x0.0920732)
pickup_latitude         AxesSubplot(0.125,0.677439;0.352273x0.0920732)
dropoff_longitude    AxesSubplot(0.547727,0.677439;0.352273x0.0920732)
dropoff_latitude        AxesSubplot(0.125,0.566951;0.352273x0.0920732)
passenger_count      AxesSubplot(0.547727,0.566951;0.352273x0.0920732)
hour                    AxesSubplot(0.125,0.456463;0.352273x0.0920732)
day                  AxesSubplot(0.547727,0.456463;0.352273x0.0920732)
month                   AxesSubplot(0.125,0.345976;0.352273x0.0920732)
year                 AxesSubplot(0.547727,0.345976;0.352273x0.0920732)
dayofweek               AxesSubplot(0.125,0.235488;0.352273x0.0920732)
dtype: object
```



In [21]:

```
pip install haversine
```

```
Collecting haversine
  Using cached haversine-2.7.0-py2.py3-none-any.whl (6.9 kB)
Installing collected packages: haversine
Successfully installed haversine-2.7.0
Note: you may need to restart the kernel to use updated packages.
```

In [17]:

```
import haversine as hs
```

In [18]:

```python
travel_dist = []
for pos in range(len(df['pickup_longitude'])):
    long1,lati1,long2,lati2 = [df['pickup_longitude'][pos],df['pickup_latitude'][pos],df['dropoff_longitude'][pos
    loc1=(lati1,long1)
    loc2=(lati2,long2)
    c = hs.haversine(loc1,loc2)
    travel_dist.append(c)
print(travel_dist)
df['dist_travel_km'] = travel_dist
df.head()
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

Out[18]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	year	dayofweek
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1.0	19	7	5	2015	3
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1.0	20	17	7	2009	4
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1.0	21	24	8	2009	0
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3.0	8	26	6	2009	4
4	16.0	-73.929786	40.744085	-73.973082	40.761247	3.5	17	28	8	2014	3

In [20]:

```python
df= df.loc[(df.dist_travel_km >= 1) | (df.dist_travel_km <= 130)]
print("Remaining observastions in the dataset:", df.shape)
```

```
Remaining observastions in the dataset: (199999, 12)
```

In [19]:
```
incorrect_coordinates = df.loc[(df.pickup_latitude > 90) |(df.pickup_latitude < -90) |
 (df.dropoff_latitude > 90) |(df.dropoff_latitude < -90) |
 (df.pickup_longitude > 180) |(df.pickup_longitude < -180) |
 (df.dropoff_longitude > 90) |(df.dropoff_longitude < -90)
 ]
```

In [20]:
```
df.drop(incorrect_coordinates, inplace = True, errors = 'ignore')
df.head()
```

Out[20]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	year	dayofweek
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1.0	19	7	5	2015	3
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1.0	20	17	7	2009	4
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1.0	21	24	8	2009	0
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3.0	8	26	6	2009	4
4	16.0	-73.929786	40.744085	-73.973082	40.761247	3.5	17	28	8	2014	3

In [21]:
```
df.drop(incorrect_coordinates, inplace = True, errors = 'ignore')
```

In [22]:
```
df.head()
```

Out[22]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day	month	year	dayofweek
0	7.5	-73.999817	40.738354	-73.999512	40.723217	1.0	19	7	5	2015	3
1	7.7	-73.994355	40.728225	-73.994710	40.750325	1.0	20	17	7	2009	4
2	12.9	-74.005043	40.740770	-73.962565	40.772647	1.0	21	24	8	2009	0
3	5.3	-73.976124	40.790844	-73.965316	40.803349	3.0	8	26	6	2009	4
4	16.0	-73.929786	40.744085	-73.973082	40.761247	3.5	17	28	8	2014	3

In [23]:
```
df.isnull().sum()
```

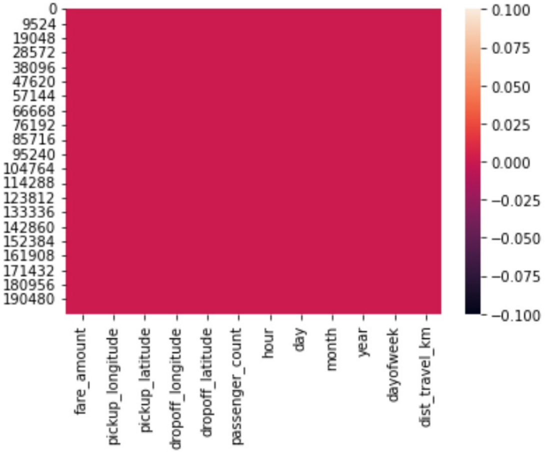
Out[23]:
```
fare_amount          0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
hour                 0
day                  0
month                0
year                 0
dayofweek            0
dist_travel_km       0
dtype: int64
```

In [24]:

```
sns.heatmap(df.isnull())
```

Out[24]:

```
<AxesSubplot:>
```



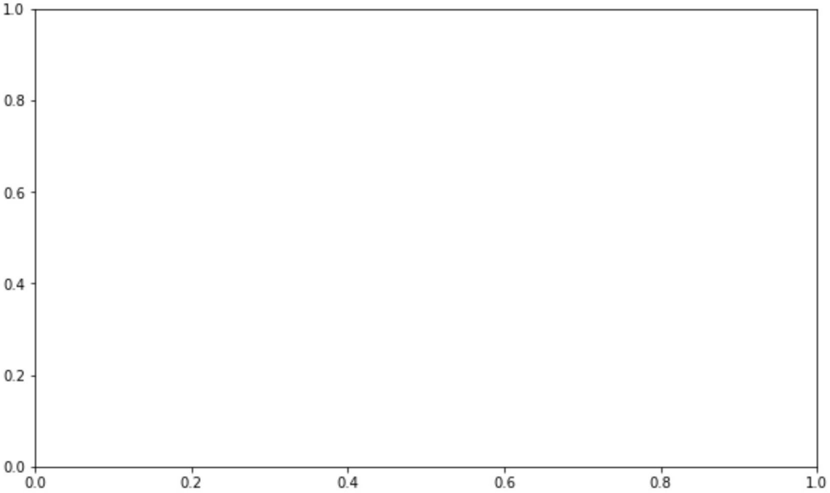
In [25]:

```
corr = df.corr() #Function to find the correlation
corr
```

Out[25]:

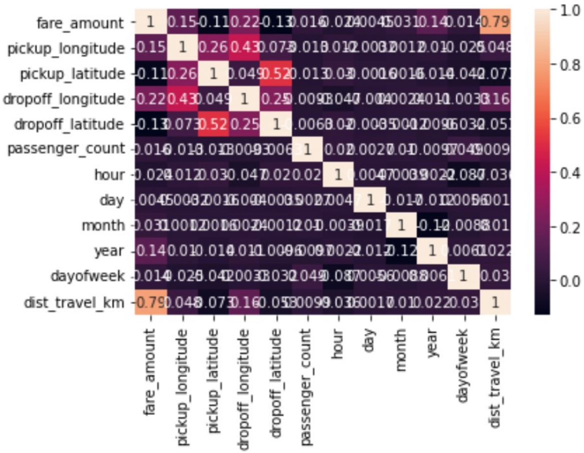
	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count	hour	day
fare_amount	1.000000	0.154069	-0.110842	0.218675	-0.125898	0.015778	-0.023623	0.004534
pickup_longitude	0.154069	1.000000	0.259497	0.425619	0.073290	-0.013213	0.011579	-0.003204
pickup_latitude	-0.110842	0.259497	1.000000	0.048889	0.515714	-0.012889	0.029681	-0.001553
dropoff_longitude	0.218675	0.425619	0.048889	1.000000	0.245667	-0.009303	-0.046558	-0.004007
dropoff_latitude	-0.125898	0.073290	0.515714	0.245667	1.000000	-0.006308	0.019783	-0.003479
passenger_count	0.015778	-0.013213	-0.012889	-0.009303	-0.006308	1.000000	0.020274	0.002712
hour	-0.023623	0.011579	0.029681	-0.046558	0.019783	0.020274	1.000000	0.004677
day	0.004534	-0.003204	-0.001553	-0.004007	-0.003479	0.002712	0.004677	1.000000
month	0.030817	0.001169	0.001562	0.002391	-0.001193	0.010351	-0.003926	-0.017360
year	0.141277	0.010198	-0.014243	0.011346	-0.009603	-0.009749	0.002156	-0.012170
dayofweek	0.013652	-0.024652	-0.042310	-0.003336	-0.031919	0.048550	-0.086947	0.005617
dist_travel_km	0.786385	0.048446	-0.073362	0.155191	-0.052701	0.009884	-0.035708	0.001709

```
In [26]: fig,axis = pl.subplots(figsize = (10,6))
```



```
In [27]: sns.heatmap(df.corr(),annot = True)
```

```
Out[27]: <AxesSubplot:>
```



```
In [28]: x = df[['pickup_longitude','pickup_latitude','dropoff_longitude','dropoff_latitude','passenger_count','hour','day'
         y = df['fare_amount']
```

```
In [29]: from sklearn.model_selection import train_test_split
```

```
In [30]: X_train,X_test,y_train,y_test = train_test_split(x,y,test_size = 0.33)
```

```
In [31]: from sklearn.linear_model import LinearRegression
```

In [32]:

```
regression = LinearRegression()
regression.fit(X_train,y_train)
regression.coef_ #To find the linear coeeficient
regression.intercept_ #To find the linear intercept
prediction = regression.predict(X_test) #To predict the target values
print(prediction)
y_test
```

```
[ 6.59280783 14.51769546  9.4935111  ...  7.63448304 12.46094817
 20.32750706]
```

Out[32]:

```
28360      3.50
4943      14.10
35866     11.50
14219      5.50
76522     14.90
          ...  
22002      8.50
82027      4.50
199627     6.00
64668      9.30
193832    22.25
Name: fare_amount, Length: 66000, dtype: float64
```

In [33]:

```
from sklearn.metrics import r2_score
```

In [34]:

```
r2_score(y_test,prediction)
from sklearn.metrics import mean_squared_error
MSE = mean_squared_error(y_test,prediction)
MSE
RMSE = np.sqrt(MSE)
RMSE
```

Out[34]:

```
3.16894186838267
```

In [35]:

```
from sklearn.ensemble import RandomForestRegressor
```

```
In [36]: rf = RandomForestRegressor(n_estimators=100)
         rf.fit(X_train,y_train)
         y_pred = rf.predict(X_test)
         y_pred

Out[36]: array([ 5.374 , 13.5905,  9.834 , ...,  8.27   , 12.8815, 20.8375])
```

```
In [37]: R2_Random = r2_score(y_test,y_pred)
         R2_Random

Out[37]: 0.7957099739429692
```

```
In [46]: MSE_Random = mean_squared_error(y_test,y_pred)
         MSE_Random

Out[46]: 5.995060403951849
```

```
In [47]: RMSE_Random = np.sqrt(MSE_Random)
         RMSE_Random

Out[47]: 2.4484812443537014
```