YASH DHUMAL C3 BATCH 14369

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_csv("uber.csv")
df
```

Out[2]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dro
0	24238194	52:06.0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	
1	27835199	04:56.0	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	
2	44984355	45:00.0	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	
3	25894730	22:21.0	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	
4	17610152	47:00.0	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	
...	...	...	...	...	...	...	
199995	42598914	49:00.0	3.0	2012-10-28 10:49:00 UTC	-73.987042	40.739367	
199996	16382965	09:00.0	7.5	2014-03-14 01:09:00 UTC	-73.984722	40.736837	
199997	27804658	42:00.0	30.9	2009-06-29 00:42:00 UTC	-73.986017	40.756487	
199998	20259894	56:25.0	14.5	2015-05-20 14:56:25 UTC	-73.997124	40.725452	
199999	11951496	08:00.0	14.1	2010-05-15 04:08:00 UTC	-73.984395	40.720077	

200000 rows × 9 columns

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_lo
0	24238194	52:06.0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-7
1	27835199	04:56.0	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-7
2	44984355	45:00.0	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-7
3	25894730	22:21.0	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-7
4	17610152	47:00.0	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-7

In [4]:
```
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 [5]:
```
df.columns
```

Out[5]:
```
Index(['Unnamed: 0', 'key', 'fare_amount', 'pickup_datetime',
       'pickup_longitude', 'pickup_latitude', 'dropoff_longitude',
       'dropoff_latitude', 'passenger_count'],
      dtype='object')
```

In [6]:
```
df =df.drop(['Unnamed: 0', 'key'], axis =1)
```

In [7]:
```
df.head()
```

Out[7]:

	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_lat
0	7.5	2015-05-07 19:52:06 UTC	-73.999817	40.738354	-73.999512	40.7
1	7.7	2009-07-17 20:04:56 UTC	-73.994355	40.728225	-73.994710	40.7
2	12.9	2009-08-24 21:45:00 UTC	-74.005043	40.740770	-73.962565	40.7
3	5.3	2009-06-26 08:22:21 UTC	-73.976124	40.790844	-73.965316	40.8
4	16.0	2014-08-28 17:47:00 UTC	-73.925023	40.744085	-73.973082	40.7

In [8]:
```
df.shape
```

Out[8]:
```
(200000, 7)
```

In [9]:
```
df.dtypes
```

Out[9]:
```
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

In [10]:
```
df.describe()
```

Out[10]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passe
count	200000.000000	200000.000000	200000.000000	199999.000000	199999.000000	20
mean	11.359955	-72.527638	39.935885	-72.525292	39.923890	
std	9.901776	11.437787	7.720539	13.117408	6.794829	
min	-52.000000	-1340.648410	-74.015515	-3356.666300	-881.985513	
25%	6.000000	-73.992065	40.734796	-73.991407	40.733823	
50%	8.500000	-73.981823	40.752592	-73.980093	40.753042	
75%	12.500000	-73.967153	40.767158	-73.963659	40.768001	
max	499.000000	57.418457	1644.421482	1153.572603	872.697628	

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [12]:
```
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(), inplace=True)
```

In [13]:
```
df.isnull().sum()
```

Out[13]:
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     0
passenger_count      0
dtype: int64
```

In [14]:
```
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].median(), inplace=True
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

In [16]:
```
df.dtypes
```

Out[16]:
```
fare_amount          float64
pickup_datetime       object
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
dtype: object
```

In [17]:
```
df.pickup_datetime = pd.to_datetime(df.pickup_datetime, errors='coerce')
```

In [18]:
```
df.dtypes
```

Out[18]:
```
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 [19]:
```
df= df.assign(hour = df.pickup_datetime.dt.hour,
              day = df.pickup_datetime.dt.day,
              month = df.pickup_datetime.dt.month,
              year = df.pickup_datetime.dt.year,
              dayofweek = df.pickup_datetime.dt.dayofweek)
```

In [20]:
```
df.head
```

Out[20]:
```
<bound method NDFrame.head of         fare_amount           pickup_datetime  picku
p_longitude  \
0                7.5 2015-05-07 19:52:06+00:00        -73.999817
1                7.7 2009-07-17 20:04:56+00:00        -73.994355
2               12.9 2009-08-24 21:45:00+00:00        -74.005043
3                5.3 2009-06-26 08:22:21+00:00        -73.976124
4               16.0 2014-08-28 17:47:00+00:00        -73.925023
...              ...                       ...               ...
199995           3.0 2012-10-28 10:49:00+00:00        -73.987042
199996           7.5 2014-03-14 01:09:00+00:00        -73.984722
199997          30.9 2009-06-29 00:42:00+00:00        -73.986017
199998          14.5 2015-05-20 14:56:25+00:00        -73.997124
199999          14.1 2010-05-15 04:08:00+00:00        -73.984395

        pickup_latitude  dropoff_longitude  dropoff_latitude  passenger_count  \
0             40.738354         -73.999512         40.723217                1
1             40.728225         -73.994710         40.750325                1
2             40.740770         -73.962565         40.772647                1
3             40.790844         -73.965316         40.803349                3
4             40.744085         -73.973082         40.761247                5
...                 ...                ...               ...              ...
199995        40.739367         -73.986525         40.740297                1
199996        40.736837         -74.006672         40.739620                1
199997        40.756487         -73.858957         40.692588                2
199998        40.725452         -73.983215         40.695416                1
199999        40.720077         -73.985508         40.768793                1

        hour  day  month  year  dayofweek
0         19    7      5  2015          3
1         20   17      7  2009          4
2         21   24      8  2009          0
3          8   26      6  2009          4
4         17   28      8  2014          3
...      ...  ...    ...   ...        ...
199995    10   28     10  2012          6
199996     1   14      3  2014          4
199997     0   29      6  2009          0
199998    14   20      5  2015          2
199999     4   15      5  2010          5

[200000 rows x 12 columns]>
```

In [21]:
```
df = df.drop('pickup_datetime', axis =1)
```

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

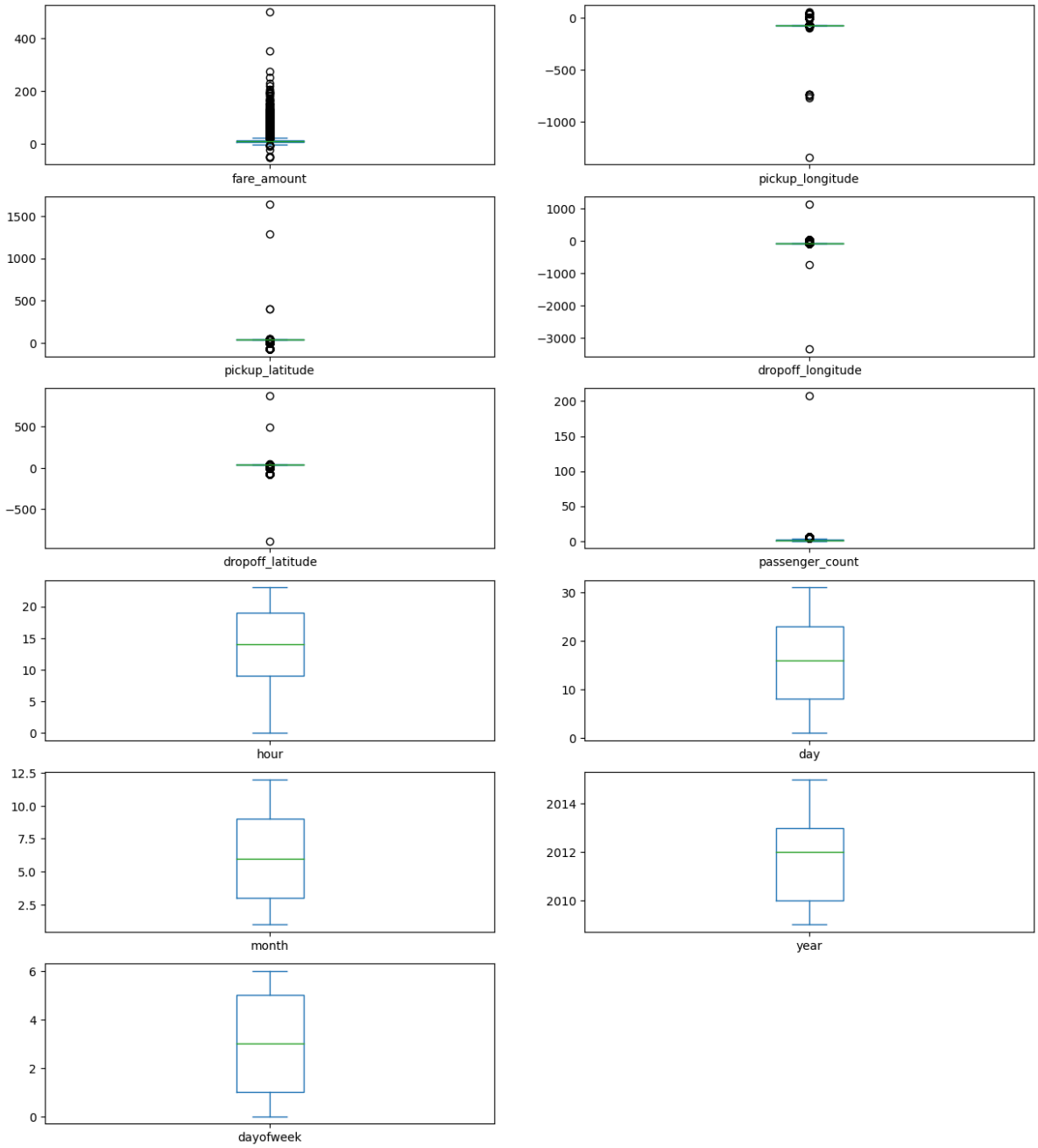
Out[22]:
```
<bound method NDFrame.head of         fare_amount  pickup_longitude  pickup_latitude  dropoff_longitude  \
0                7.5        -73.999817        40.738354         -73.999512
1                7.7        -73.994355        40.728225         -73.994710
2               12.9        -74.005043        40.740770         -73.962565
3                5.3        -73.976124        40.790844         -73.965316
4               16.0        -73.925023        40.744085         -73.973082
...              ...               ...              ...                ...
199995           3.0        -73.987042        40.739367         -73.986525
199996           7.5        -73.984722        40.736837         -74.006672
199997          30.9        -73.986017        40.756487         -73.858957
199998          14.5        -73.997124        40.725452         -73.983215
199999          14.1        -73.984395        40.720077         -73.985508

        dropoff_latitude  passenger_count  hour  day  month  year  dayofweek
0              40.723217                1    19    7      5  2015          3
1              40.750325                1    20   17      7  2009          4
2              40.772647                1    21   24      8  2009          0
3              40.803349                3     8   26      6  2009          4
4              40.761247                5    17   28      8  2014          3
...                  ...              ...   ...  ...    ...   ...        ...
199995         40.740297                1    10   28     10  2012          6
199996         40.739620                1     1   14      3  2014          4
199997         40.692588                2     0   29      6  2009          0
199998         40.695416                1    14   20      5  2015          2
199999         40.768793                1     4   15      5  2010          5

[200000 rows x 11 columns]>
```

In [23]:
```
df.dtypes
```

Out[23]:
```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count        int64
hour                   int32
day                    int32
month                  int32
year                   int32
dayofweek              int32
dtype: object
```

In [25]:
```
df.plot(kind="box", subplots = True,layout =(7,2),figsize=(15,20))
```

Out[25]:
```
fare_amount            Axes(0.125,0.786098;0.352273x0.0939024)
pickup_longitude     Axes(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude        Axes(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude    Axes(0.547727,0.673415;0.352273x0.0939024)
dropoff_latitude       Axes(0.125,0.560732;0.352273x0.0939024)
passenger_count      Axes(0.547727,0.560732;0.352273x0.0939024)
hour                   Axes(0.125,0.448049;0.352273x0.0939024)
day                  Axes(0.547727,0.448049;0.352273x0.0939024)
month                  Axes(0.125,0.335366;0.352273x0.0939024)
year                 Axes(0.547727,0.335366;0.352273x0.0939024)
dayofweek              Axes(0.125,0.222683;0.352273x0.0939024)
dtype: object
```

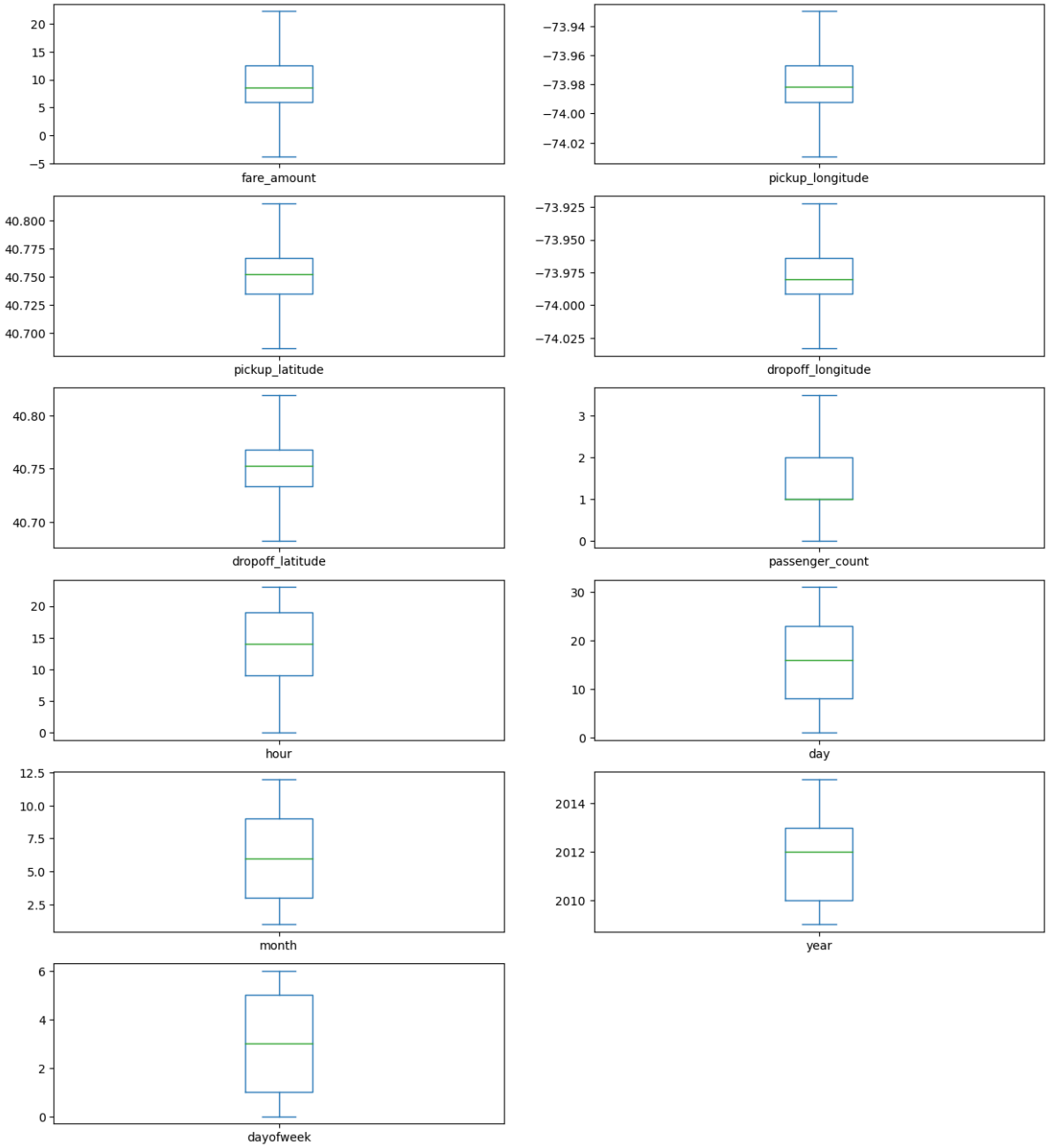


In [27]:
```
def remove_outlier(df1, col):
    Q1 =df1[col].quantile(0.25)
    Q3= df1[col].quantile (0.75)
    IQR= Q3 - Q1
    lower_whisker= Q1-1.5*IQR
    upper_whisker= Q3+1.5*IQR
    df[col] = np.clip(df1[col], lower_whisker, upper_whisker)
    return df1
def treat_outliers_all(df1, col_list):
    for c in col_list:
        df1 = remove_outlier(df, c)
    return df1
```

In [28]:
```
df = treat_outliers_all(df,df.iloc[:, 0::])
```

In [29]:
```
df.plot(kind = "box",subplots = True,layout=(7,2),figsize=(15,20))
```

Out[29]:
```
fare_amount          Axes(0.125,0.786098;0.352273x0.0939024)
pickup_longitude     Axes(0.547727,0.786098;0.352273x0.0939024)
pickup_latitude      Axes(0.125,0.673415;0.352273x0.0939024)
dropoff_longitude    Axes(0.547727,0.673415;0.352273x0.0939024)
dropoff_latitude     Axes(0.125,0.560732;0.352273x0.0939024)
passenger_count      Axes(0.547727,0.560732;0.352273x0.0939024)
hour                 Axes(0.125,0.448049;0.352273x0.0939024)
day                  Axes(0.547727,0.448049;0.352273x0.0939024)
month                Axes(0.125,0.335366;0.352273x0.0939024)
year                 Axes(0.547727,0.335366;0.352273x0.0939024)
dayofweek            Axes(0.125,0.222683;0.352273x0.0939024)
dtype: object
```



In [30]:
```
pip install haversine
```

```
Collecting haversine
  Obtaining dependency information for haversine from https://files.pythonhosted.o
rg/packages/5b/f1/b7274966f0b5b665d9114e86d09c6bc87d241781d63d8817323dcfa940c6/hav
ersine-2.8.1-py2.py3-none-any.whl.metadata
  Downloading haversine-2.8.1-py2.py3-none-any.whl.metadata (5.9 kB)
Downloading haversine-2.8.1-py2.py3-none-any.whl (7.7 kB)
Installing collected packages: haversine
Successfully installed haversine-2.8.1
Note: you may need to restart the kernel to use updated packages.
```

In [38]:

```
import haversine as hs
travel_dist = []
for pos in range(len(df['pickup_longitude'])):
    long1,lati1,long2,lati2 = [df['pickup_longitude'][pos],df['pickup_latitude'][pc
    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[38]:
```
<bound method NDFrame.head of         fare_amount  pickup_longitude  pickup_latitude  dropoff_longitude  \
0               7.50        -73.999817        40.738354         -73.999512
1               7.70        -73.994355        40.728225         -73.994710
2              12.90        -74.005043        40.740770         -73.962565
3               5.30        -73.976124        40.790844         -73.965316
4              16.00        -73.929786        40.744085         -73.973082
...              ...               ...              ...                ...
199995          3.00        -73.987042        40.739367         -73.986525
199996          7.50        -73.984722        40.736837         -74.006672
199997         22.25        -73.986017        40.756487         -73.922036
199998         14.50        -73.997124        40.725452         -73.983215
199999         14.10        -73.984395        40.720077         -73.985508

        dropoff_latitude  passenger_count  hour  day  month  year  dayofweek  \
0              40.723217              1.0    19    7      5  2015          3
1              40.750325              1.0    20   17      7  2009          4
2              40.772647              1.0    21   24      8  2009          0
3              40.803349              3.0     8   26      6  2009          4
4              40.761247              3.5    17   28      8  2014          3
...                  ...              ...   ...  ...    ...   ...        ...
199995         40.740297              1.0    10   28     10  2012          6
199996         40.739620              1.0     1   14      3  2014          4
199997         40.692588              2.0     0   29      6  2009          0
199998         40.695416              1.0    14   20      5  2015          2
199999         40.768793              1.0     4   15      5  2010          5

        dist_travel_km
0             1.683325
1             2.457593
2             5.036384
3             1.661686
4             4.116088
...                ...
199995        0.112210
199996        1.875053
199997        8.919323
199998        3.539720
199999        5.417791

[200000 rows x 12 columns]>
```

In [39]:
```
df = df.loc[(df.dist_travel_km>=1) |(df.dist_travel_km<=130) ]
print("Remaining obervation:" , df.shape)
```

```
Remaining obervation: (200000, 12)
```

In [47]:
```
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_latitude>90) | (df.dropoff_latitude< -90)
                             ]
```

In [48]:
```
df.drop(incorrect_coordinates, inplace = True, errors ='ignore')
```

In [49]:
```
df.head()
```

Out[49]:

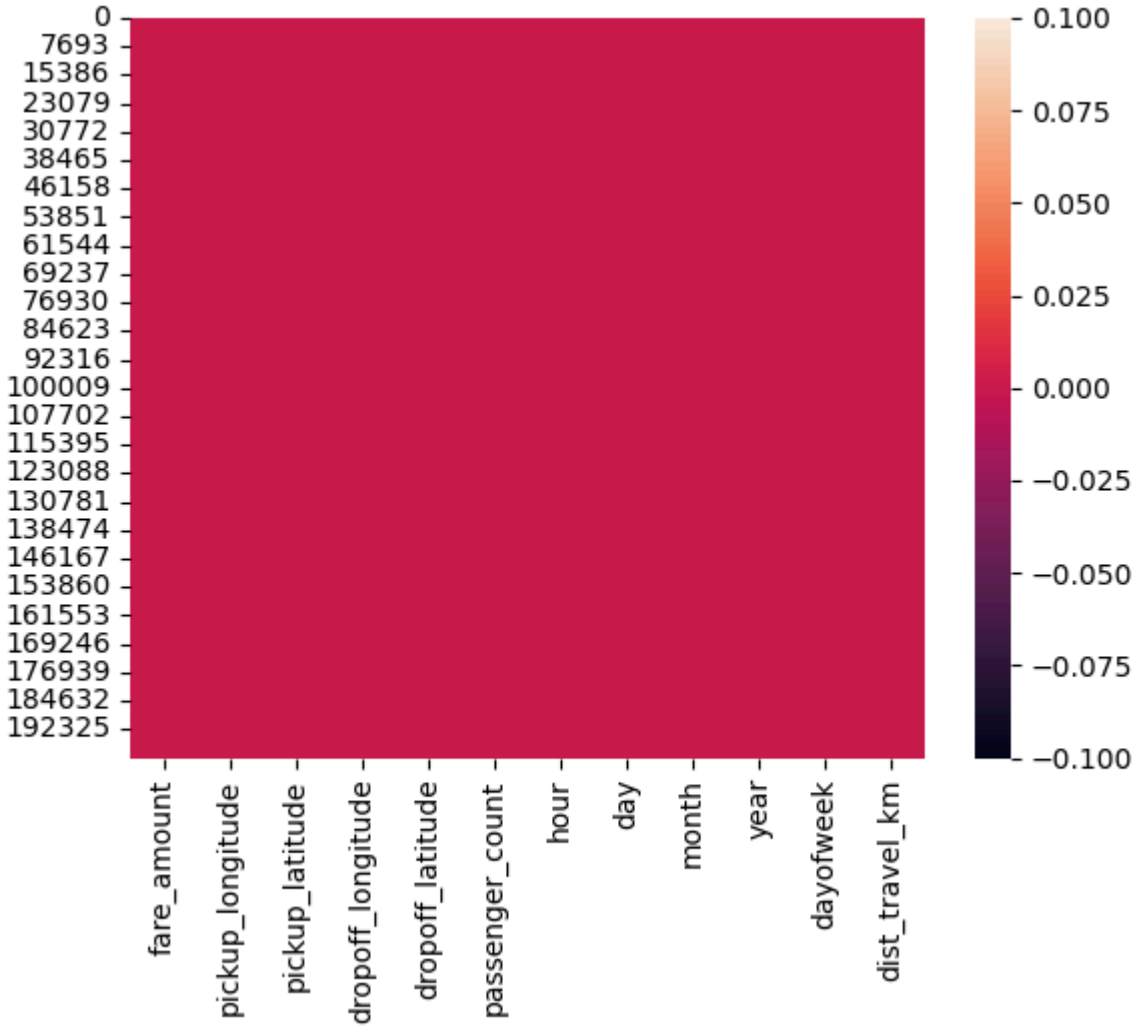
	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_
0	7.5	-73.999817	40.738354	-73.999512	40.723217	
1	7.7	-73.994355	40.728225	-73.994710	40.750325	
2	12.9	-74.005043	40.740770	-73.962565	40.772647	
3	5.3	-73.976124	40.790844	-73.965316	40.803349	
4	16.0	-73.929786	40.744085	-73.973082	40.761247	

In [50]:
```
df.isnull().sum()
```

Out[50]:
```
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 [51]:
```
sns.heatmap(df.isnull())
```

Out[51]:
```
<Axes: >
```



In [52]:
```
corr = df.corr()
```

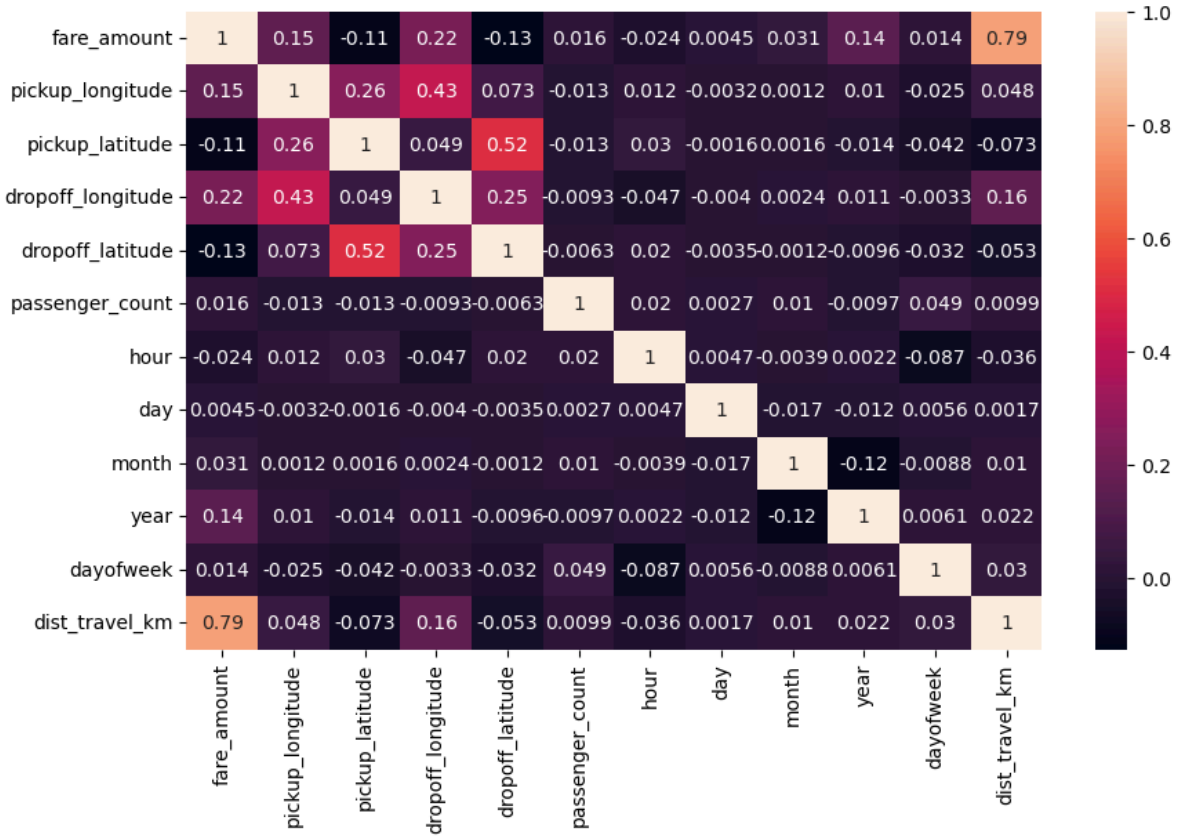
In [53]:
```
corr
```

Out[53]:

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latit
fare_amount	1.000000	0.154069	-0.110842	0.218675	-0.125
pickup_longitude	0.154069	1.000000	0.259497	0.425619	0.073
pickup_latitude	-0.110842	0.259497	1.000000	0.048889	0.515
dropoff_longitude	0.218675	0.425619	0.048889	1.000000	0.245
dropoff_latitude	-0.125898	0.073290	0.515714	0.245667	1.000
passenger_count	0.015778	-0.013213	-0.012889	-0.009303	-0.006
hour	-0.023623	0.011579	0.029681	-0.046558	0.019
day	0.004534	-0.003204	-0.001553	-0.004007	-0.003
month	0.030817	0.001169	0.001562	0.002391	-0.001
year	0.141277	0.010198	-0.014243	0.011346	-0.009
dayofweek	0.013652	-0.024652	-0.042310	-0.003336	-0.031
dist_travel_km	0.786385	0.048446	-0.073362	0.155191	-0.052

In [55]:
```
fig,axis = plt.subplots(figsize= (10,6))
sns.heatmap(df.corr(), annot = True)
```

Out[55]:
```
<Axes: >
```



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

Out[56]:
```
fare_amount          float64
pickup_longitude     float64
pickup_latitude      float64
dropoff_longitude    float64
dropoff_latitude     float64
passenger_count      float64
hour                   int32
day                    int32
month                  int32
year                   int32
dayofweek              int32
dist_travel_km       float64
dtype: object
```

In [58]:
```
x = df[['pickup_longitude', 'pickup_latitude','dropoff_longitude','dropoff_latitude
```

In [59]:
```
y = df['fare_amount']
```

In [60]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train,y_test = train_test_split(x,y,test_size = 0.33)
```

In [61]:
```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()
```

In [62]:
```
regression.fit(x_train,y_train)
```

Out[62]:
```
▾ LinearRegression

LinearRegression()
```

In [64]:
```
regression.intercept_
```

Out[64]:
```
3615.5861559214154
```

In [65]:
```
regression.coef_
```

Out[65]:
```
array([ 2.49068742e+01, -6.89454719e+00,  2.01504052e+01, -1.81286709e+01,
        5.98985068e-02,  6.08043699e-03,  2.92131129e-03,  5.97438413e-02,
        3.68715386e-01, -3.72169236e-02,  1.85329279e+00])
```

In [66]:
```
prediction = regression.predict(x_test)
```

In [67]:
```
print(prediction)
```

```
[ 9.21179204  8.91436439  5.47686097 ...  7.52253896 14.2121657
  5.73971605]
```

In [68]:
```
y_test
```

Out[68]:
```
134041    13.5
125453     8.5
151639     3.0
174239    12.9
165574     5.7
          ...
73077      4.9
91779     14.0
176467     5.7
152546    17.7
194320     6.1
Name: fare_amount, Length: 66000, dtype: float64
```

In [69]:
```
from sklearn.metrics import r2_score
```

In [70]:
```
r2_score(y_test,prediction)
```

Out[70]:
```
0.6599227836474304
```

In [71]:
```
from sklearn.metrics import mean_squared_error
```

In [72]:
```
MSE = mean_squared_error(y_test,prediction)
```

In [73]:
```
MSE
```

Out[73]:
```
10.009319812626584
```