In [56]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [57]:
```
df = pd.read_csv('uber.csv')
```

In [58]:
```
df.head
```

Out[58]:
```
<bound method NDFrame.head of         Unnamed: 0                            key  f
are_amount  \
0         24238194    2015-05-07 19:52:06.0000003          7.5
1         27835199    2009-07-17 20:04:56.0000002          7.7
2         44984355   2009-08-24 21:45:00.00000061         12.9
3         25894730    2009-06-26 08:22:21.0000001          5.3
4         17610152  2014-08-28 17:47:00.000000188         16.0
...            ...                           ...          ...
199995    42598914   2012-10-28 10:49:00.00000053          3.0
199996    16382965    2014-03-14 01:09:00.0000008          7.5
199997    27804658   2009-06-29 00:42:00.00000078         30.9
199998    20259894    2015-05-20 14:56:25.0000004         14.5
199999    11951496   2010-05-15 04:08:00.00000076         14.1

                pickup_datetime  pickup_longitude  pickup_latitude  \
0       2015-05-07 19:52:06 UTC        -73.999817        40.738354
1       2009-07-17 20:04:56 UTC        -73.994355        40.728225
2       2009-08-24 21:45:00 UTC        -74.005043        40.740770
3       2009-06-26 08:22:21 UTC        -73.976124        40.790844
4       2014-08-28 17:47:00 UTC        -73.925023        40.744085
...                         ...               ...              ...
199995  2012-10-28 10:49:00 UTC        -73.987042        40.739367
199996  2014-03-14 01:09:00 UTC        -73.984722        40.736837
199997  2009-06-29 00:42:00 UTC        -73.986017        40.756487
199998  2015-05-20 14:56:25 UTC        -73.997124        40.725452
199999  2010-05-15 04:08:00 UTC        -73.984395        40.720077

        dropoff_longitude  dropoff_latitude  passenger_count
0              -73.999512         40.723217                1
1              -73.994710         40.750325                1
2              -73.962565         40.772647                1
3              -73.965316         40.803349                3
4              -73.973082         40.761247                5
...                   ...               ...              ...
199995         -73.986525         40.740297                1
199996         -74.006672         40.739620                1
199997         -73.858957         40.692588                2
199998         -73.983215         40.695415                1
199999         -73.985508         40.768793                1

[200000 rows x 9 columns]>
```

In [59]:
```
df.head()
```

Out[59]:

	Unnamed: 0	key	fare_amount	pickup_datetime	pickup_longitude	pickup_
0	24238194	2015-05-07 19:52:06.0000003	7.5	2015-05-07 19:52:06 UTC	-73.999817	40
1	27835199	2009-07-17 20:04:56.0000002	7.7	2009-07-17 20:04:56 UTC	-73.994355	40
2	44984355	2009-08-24 21:45:00.00000061	12.9	2009-08-24 21:45:00 UTC	-74.005043	40
3	25894730	2009-06-26 08:22:21.0000001	5.3	2009-06-26 08:22:21 UTC	-73.976124	40
4	17610152	2014-08-28 17:47:00.000000188	16.0	2014-08-28 17:47:00 UTC	-73.925023	40

In [60]:
```
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 [61]:
```
df.shape
```

Out[61]:
```
(200000, 9)
```

In [62]:
```
df.isnull().any()
```

Out[62]:
```
Unnamed: 0           False
key                  False
fare_amount          False
pickup_datetime      False
pickup_longitude     False
pickup_latitude      False
dropoff_longitude     True
dropoff_latitude      True
passenger_count      False
dtype: bool
```

In [63]:
```
df.isnull().sum()
```

Out[63]:
```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [64]:
```
df = df.drop('pickup_datetime',axis = 1)
```

In [65]:
```
df.isnull().sum()
```

Out[65]:
```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    1
dropoff_latitude     1
passenger_count      0
dtype: int64
```

In [66]:
```
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].mean(), inplace = True
```

In [67]:
```
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(), inplace = True)
```

In [68]:
```
df.isnull().sum()
```

Out[68]:
```
Unnamed: 0           0
key                  0
fare_amount          0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

In [69]:
```
df = df.loc[df['pickup_longitude'] >= -180]
df = df.loc[df['pickup_longitude'] <= 180]
df.shape
```

Out[69]:
```
(199993, 8)
```

In [70]:
```
df = df.loc[df['dropoff_longitude'] >= -180]
df = df.loc[df['dropoff_longitude'] <= 180]
df.shape
```

Out[70]:
```
(199991, 8)
```

In [71]:
```
df = df.loc[df['pickup_latitude'] >= -90]
df = df.loc[df['pickup_latitude'] <= 90]
df.shape
```

Out[71]:
```
(199989, 8)
```

In [72]:
```
df = df.loc[df['dropoff_latitude'] >= -90]
df = df.loc[df['dropoff_latitude'] <= 90]
df.shape
```

Out[72]:
```
(199988, 8)
```

In [73]:
```
df.isnull().any()
```

Out[73]:
```
Unnamed: 0           False
key                  False
fare_amount          False
pickup_longitude     False
pickup_latitude      False
dropoff_longitude    False
dropoff_latitude     False
passenger_count      False
dtype: bool
```

In [74]:
```
from math import *
def distance_formula(long1, long2, lat1, lat2):
    dist_long = long2 - long1
    dist_lat = lat2 - lat1

    c = np.sin(dist_long/2)**2+np.cos(long1)*np.cos(long2)*np.sin(dist_lat/2)**2

    c = np.arcsin(np.sqrt(c))

    c = 2*6371*c

    return c
```

In [75]:
```
df['distance'] = distance_formula(df['pickup_longitude'],
                   df['dropoff_longitude'],df['pickup_latitude'],
                   df['dropoff_latitude'])
```

In [76]:
```
df
```

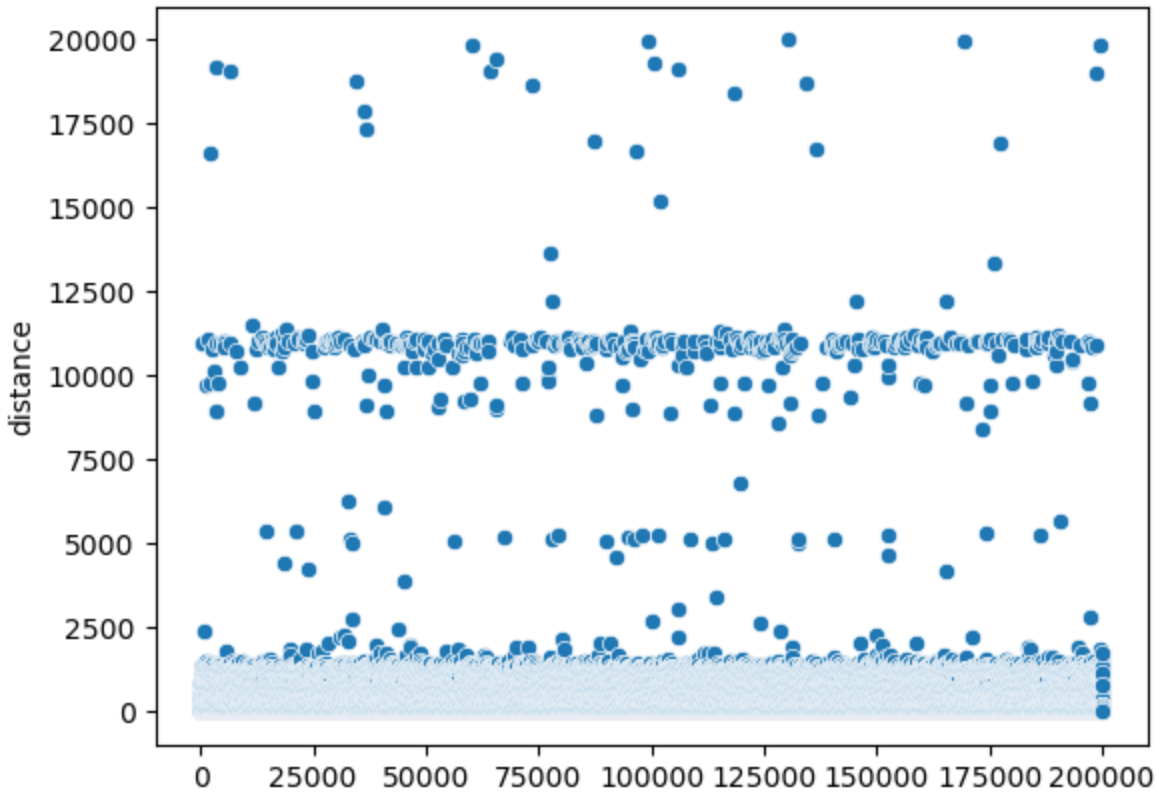
Out[76]:

	Unnamed: 0	key	fare_amount	pickup_longitude	pickup_latitude	dr
0	24238194	2015-05-07 19:52:06.0000003	7.5	-73.999817	40.738354	
1	27835199	2009-07-17 20:04:56.0000002	7.7	-73.994355	40.728225	
2	44984355	2009-08-24 21:45:00.00000061	12.9	-74.005043	40.740770	
3	25894730	2009-06-26 08:22:21.0000001	5.3	-73.976124	40.790844	
4	17610152	2014-08-28 17:47:00.000000188	16.0	-73.925023	40.744085	
...	...	...	...	...	...	...
199995	42598914	2012-10-28 10:49:00.00000053	3.0	-73.987042	40.739367	
199996	16382965	2014-03-14 01:09:00.0000008	7.5	-73.984722	40.736837	
199997	27804658	2009-06-29 00:42:00.00000078	30.9	-73.986017	40.756487	
199998	20259894	2015-05-20 14:56:25.0000004	14.5	-73.997124	40.725452	
199999	11951496	2010-05-15 04:08:00.00000076	14.1	-73.984395	40.720077	

199988 rows × 9 columns

In [77]:
```
import seaborn as sns
sns.scatterplot(df['distance'])
```

Out[77]:
```
<Axes: ylabel='distance'>
```



In [78]:
```
df = df.loc[df['distance'] > 0]
df.shape
```

Out[78]:
```
(194356, 9)
```

In [79]:
```
df = df.loc[df['fare_amount'] > 0]
df.shape
```

Out[79]:
```
(194336, 9)
```

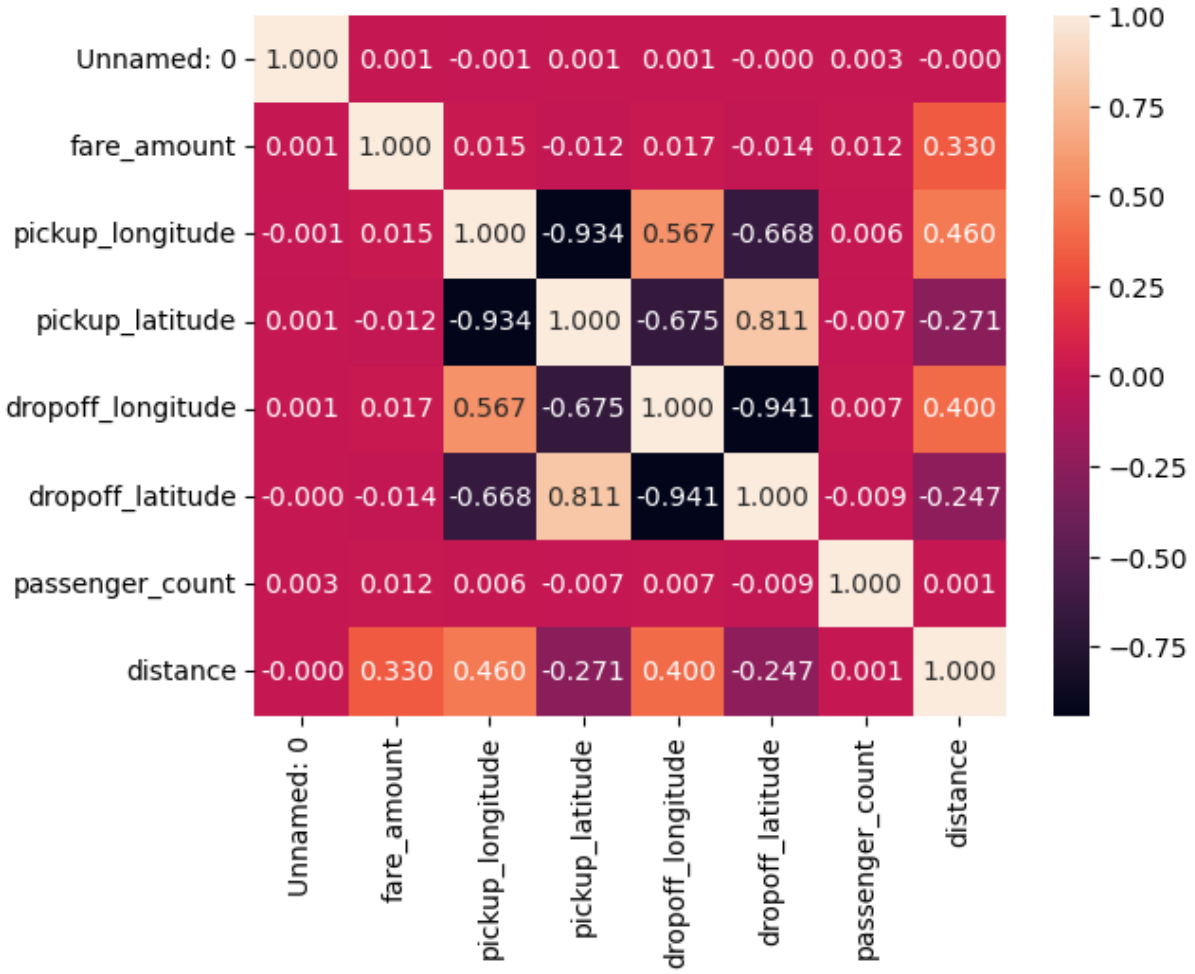
In [80]:
```
df.isnull().any()
```

Out[80]:
```
Unnamed: 0          False
key                 False
fare_amount         False
pickup_longitude    False
pickup_latitude     False
dropoff_longitude   False
dropoff_latitude    False
passenger_count     False
distance            False
dtype: bool
```

In [81]:
```
df = df.drop('key', axis = 1)
```

In [82]:
```
sns.heatmap(df.corr(), annot = True, fmt = '.3f')
```

Out[82]:
```
<Axes: >
```



In [83]:
```
x = df['distance'].values.reshape([-1,1])
```

In [84]:
```
x
```

Out[84]:
```
array([[ 16.64165387],
       [ 23.52739069],
       [272.43560947],
       ...,
       [810.00993188],
       [ 93.86877183],
       [ 49.19746499]])
```

In [85]:
```
y = df['fare_amount'].values.reshape([-1,1])
y
```

Out[85]:
```
array([[ 7.5],
       [ 7.7],
       [12.9],
       ...,
       [30.9],
       [14.5],
       [14.1]])
```

In [86]:
```
# model 1 : Linear Regression
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

In [87]:
```
x = sc.fit_transform(x)
y = sc.fit_transform(y)
```

In [88]:
```
x
```

Out[88]:
```
array([[-0.27925081],
       [-0.26705972],
       [ 0.17362855],
       ...,
       [ 1.12539582],
       [-0.14252136],
       [-0.22161124]])
```

In [89]:
```
y
```

Out[89]:
```
array([[-0.39555097],
       [-0.37503504],
       [ 0.15837935],
       ...,
       [ 2.00481375],
       [ 0.32250685],
       [ 0.28147498]])
```

In [90]:
```
from sklearn.model_selection import train_test_split as tts
from sklearn.linear_model import LinearRegression
xtrain, xtest, ytrain, ytest = tts(x, y, test_size=0.2)
```

In [91]:
```
lm = LinearRegression()
```

In [92]:
```
lm = lm.fit(xtrain, ytrain)
y_pred = lm.predict(xtest)
```

In [93]:
```
y_pred
```

Out[93]:
```
array([[ 0.12258958],
       [-0.04857723],
       [-0.05178543],
       ...,
       [-0.07565463],
       [-0.0803088 ],
       [-0.05782254]])
```

In [94]:
```
ytest
```

Out[94]:
```
array([[ 1.79965437],
       [ 0.3737967 ],
       [ 0.73282561],
       ...,
       [-0.44684082],
       [-0.74432192],
       [-0.60071035]])
```

In [95]:
```
print(float(lm.intercept_))
```

```
-0.0007198632214074118
```

In [96]:
```
print(float(lm.coef_))
```

```
0.32050989031359595
```

In [97]:
```
from sklearn.metrics import r2_score,mean_squared_error
import math
print("The rmse is :", math.sqrt(mean_squared_error(ytest, y_pred)))
```

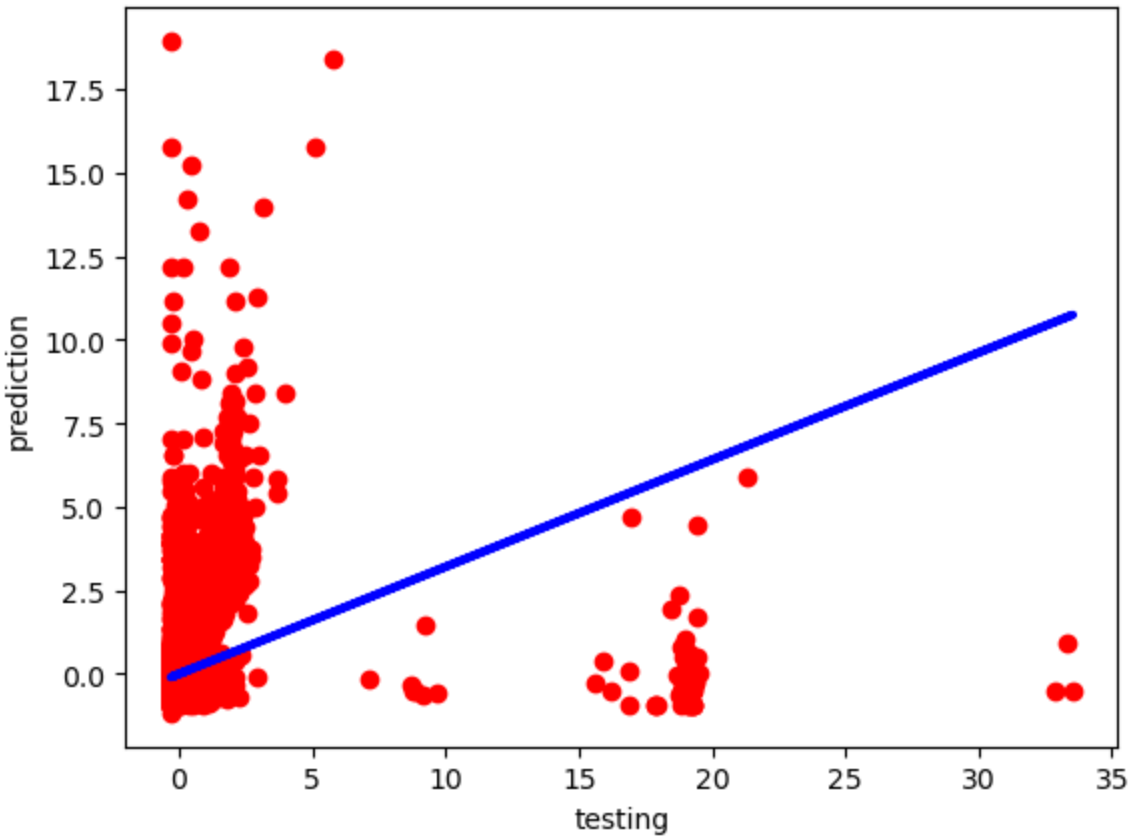
```
The rmse is : 0.9464872900001531
```

In [98]:
```
print("The mse is : ", mean_squared_error(ytest, y_pred))
```

```
The mse is :  0.895838190131834
```

In [99]:
```
plt.scatter(xtest, ytest, color = "red")
plt.plot(xtest, y_pred, color = "blue", linewidth = 3)
plt.xlabel("testing")
plt.ylabel("prediction")
```

Out[99]:
```
Text(0, 0.5, 'prediction')
```



In [100…
```
# implementation of random forest
x = df[['pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitu
y = df['fare_amount']
```

In [104…
```
x_train, x_test, ytrain, ytest = tts(x, y, test_size = 0.25)
```

In [105…
```
df
```

Out[105]:

	Unnamed: 0	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	drc
0	24238194	7.5	-73.999817	40.738354	-73.999512	
1	27835199	7.7	-73.994355	40.728225	-73.994710	
2	44984355	12.9	-74.005043	40.740770	-73.962565	
3	25894730	5.3	-73.976124	40.790844	-73.965316	
4	17610152	16.0	-73.925023	40.744085	-73.973082	
…	…	…	…	…	…	
199995	42598914	3.0	-73.987042	40.739367	-73.986525	
199996	16382965	7.5	-73.984722	40.736837	-74.006672	
199997	27804658	30.9	-73.986017	40.756487	-73.858957	
199998	20259894	14.5	-73.997124	40.725452	-73.983215	
199999	11951496	14.1	-73.984395	40.720077	-73.985508	

194336 rows × 8 columns

In [106…
```
from sklearn.ensemble import RandomForestRegressor as rfr
rf = rfr(n_estimators = 100)
rf.fit(x_train, ytrain)
```

Out[106]:
```
RandomForestRegressor
RandomForestRegressor()
```

In [107…
```
y_pred = rf.predict(x_test)
y_pred
```

Out[107]:
```
array([19.711,  5.021,  6.59 , ...,  9.869, 10.978,  6.628])
```

In [108…
```
ytest
```

Out[108]:
```
55822      4.9
15350      4.5
93329      8.5
42342      4.1
6737      10.5
          ...
83681      7.0
2869       6.0
92967      8.0
67499      9.3
145687     8.5
Name: fare_amount, Length: 48584, dtype: float64
```

In [109…
```
print(r2_score(ytest, y_pred))
```

```
0.7585451554161666
```

In [110…
```
print(math.sqrt(mean_squared_error(ytest, y_pred)))
```

```
4.867767302363608
```

In [ ]:
```

```