```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt

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

```
   Unnamed: 0                            key  fare_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   

           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   

   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  
```

```
df.columns
```

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

```
df.drop(columns=['Unnamed: 0','key'],inplace=True)
```

```
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
```

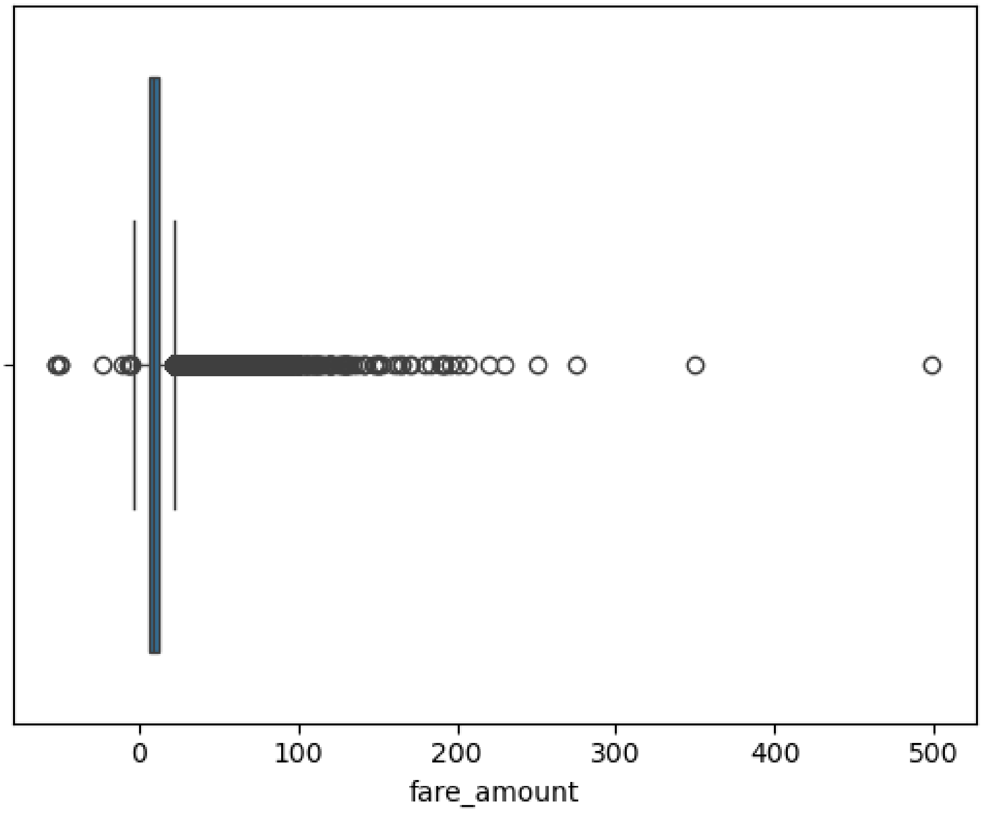
```
df.dropna(how='any',inplace=True)
```

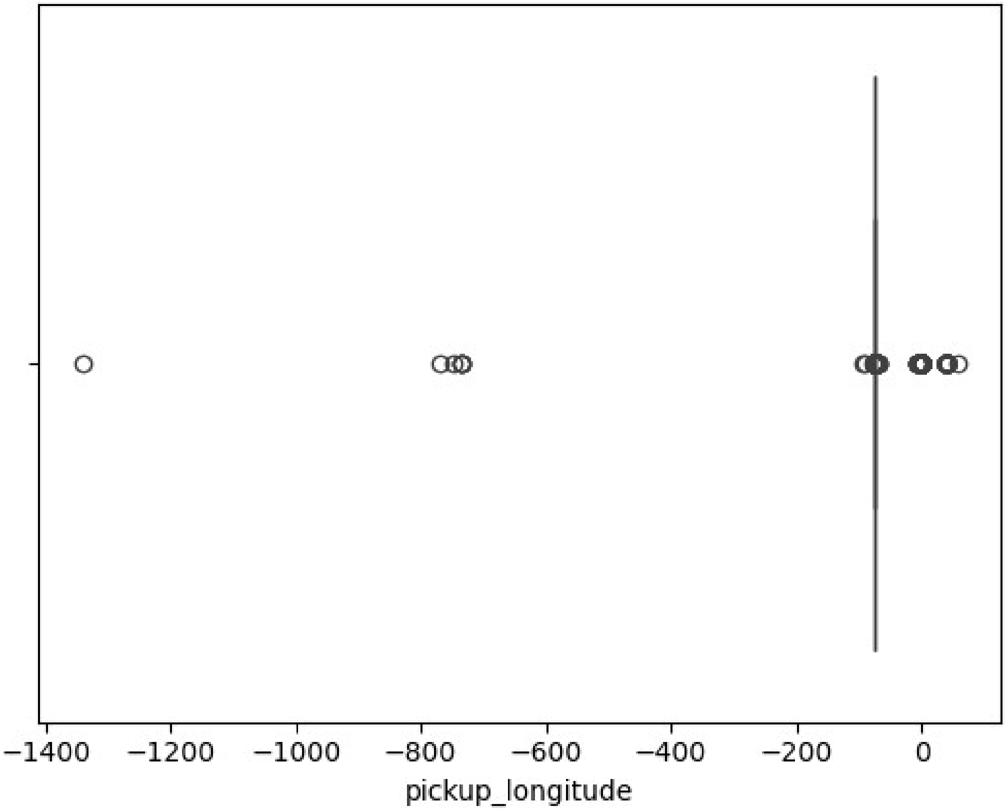
```
df.isnull().sum()
```

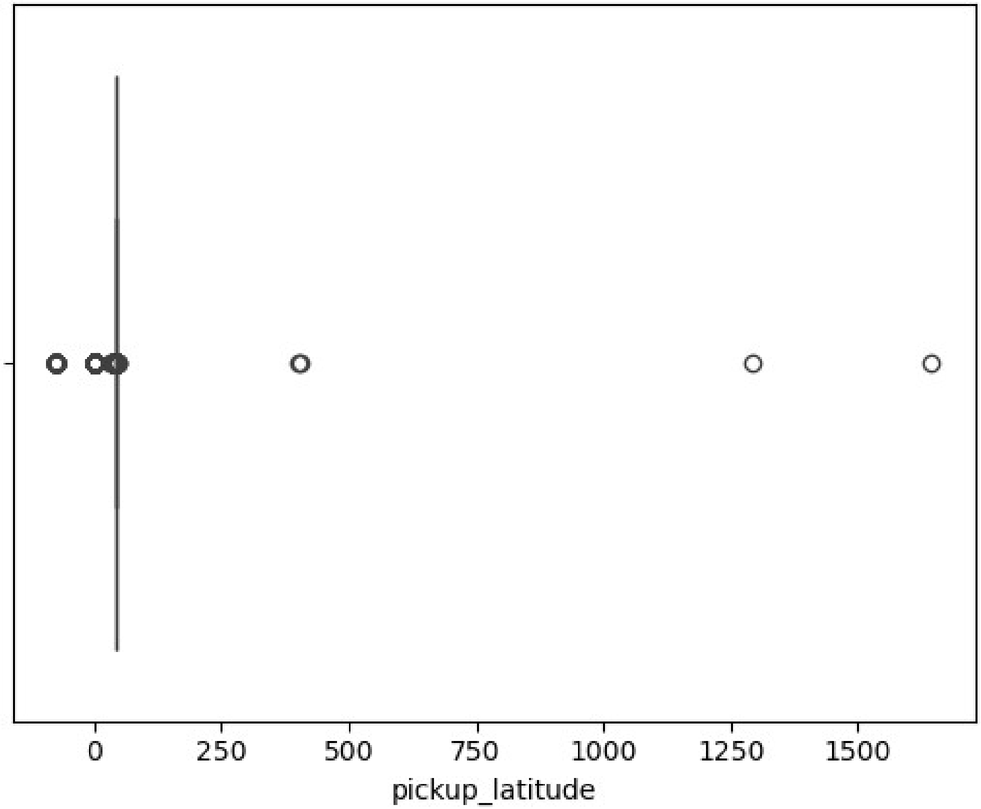
```
fare_amount          0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

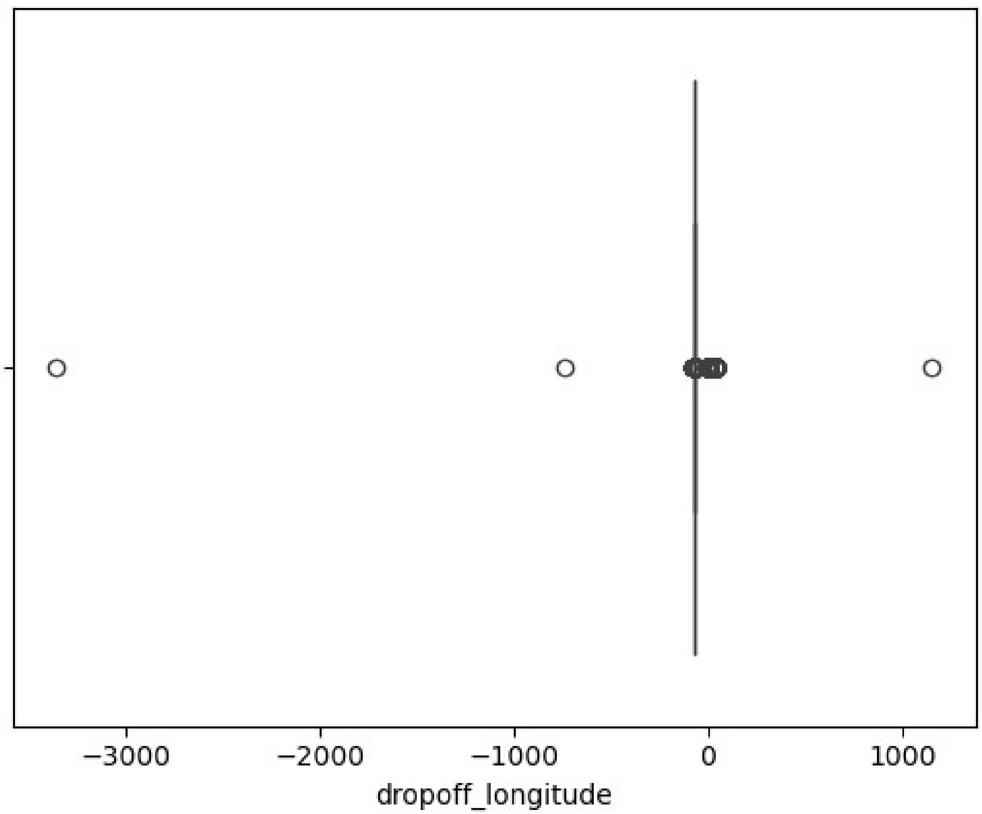
Box Plots

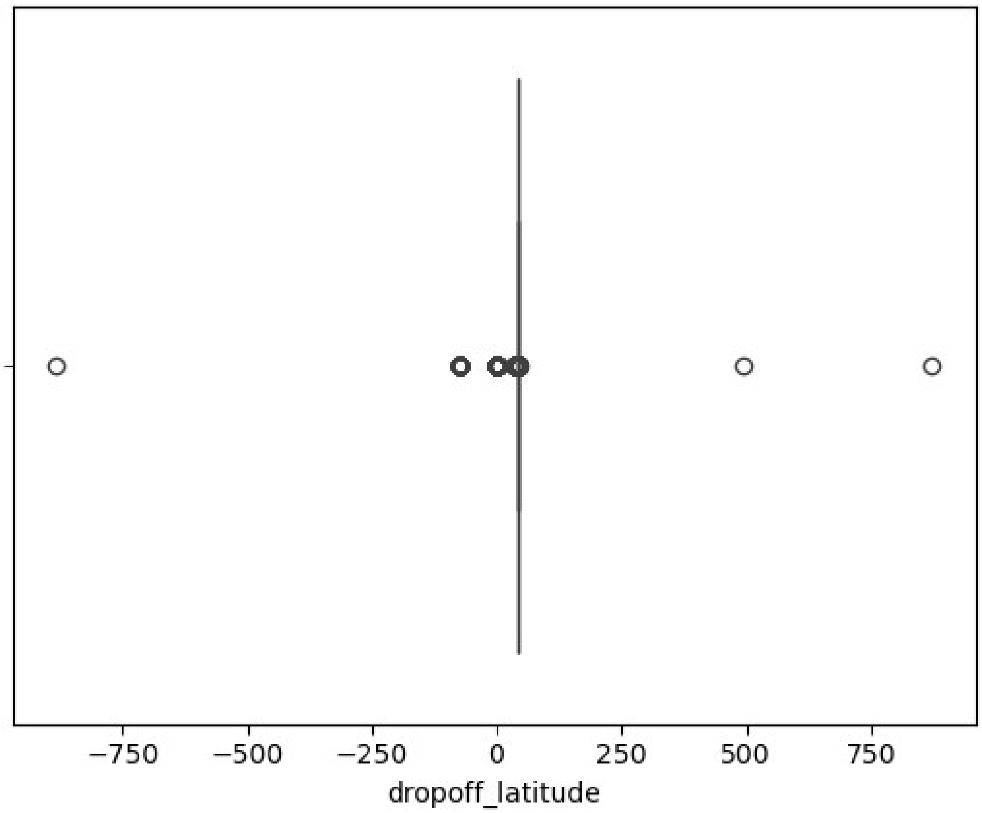
```
for col in df.select_dtypes(exclude=['object']):
    plt.figure()
    sns.boxplot(data=df, x=col)
```

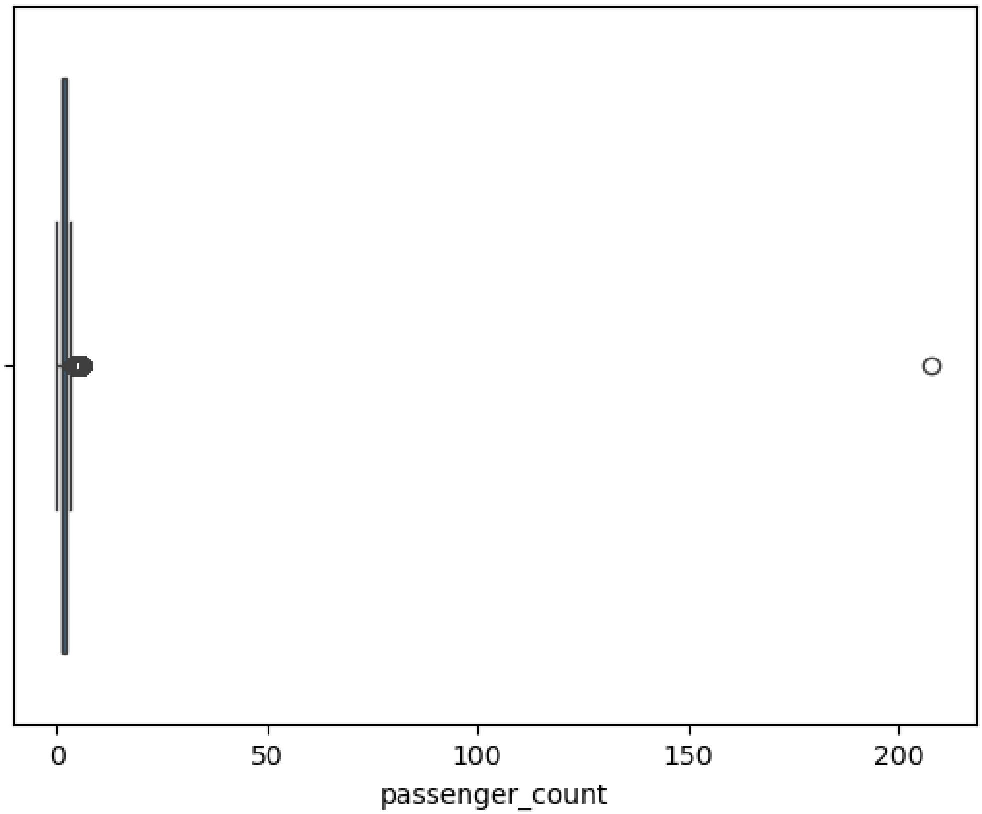



−1400
−1200
−1000
−800
−600
−400
−200
0
pickup_longitude


0
250
500
750
1000
1250
1500
pickup_latitude


−3000
−2000
−1000
0
1000
dropoff_longitude


−750
−500
−250
0
250
500
750
dropoff_latitude



```
df = df[
    (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 > -180) & (df.dropoff_longitude < 180) &
    (df.fare_amount > 0) & (df.passenger_count > 0) &
(df.passenger_count < 50)
]

from haversine import haversine, Unit

def haversine_distances(coord_list1, coord_list2):
    distances = [haversine(coord1, coord2, unit=Unit.KILOMETERS) for
coord1, coord2 in zip(coord_list1, coord_list2)]

    return distances

coords1 = list(zip(df['pickup_latitude'], df['pickup_longitude']))
coords2 = list(zip(df['dropoff_latitude'], df['dropoff_longitude']))

distances = haversine_distances(coords1, coords2)

distances
```

```
[1.6833250775073447,
 2.4575932783467835,
 5.036384146783453,
 1.661685753650294,
 4.47545614478295,
 0.0,
 11.731031087106315,
 0.0,
 2.332714231417754,
 4.889423641655177,
 2.2508607308770285,
 0.0,
 0.3022521108558365,
 3.5812557740132496,
 1.3099517093917648,
 1.716279792276335,
 0.7299665570466272,
 2.515953547298386,
 1.790321726187665,
 1.0347050399795192,
 2.4902472008677727,
 0.9594701844599927,
 1.2613897673437817,
 1.7517650017211177,
 6.1932445014761095,
 2.736192584061414,
 0.7232537124105735,
 3.229443537425455,
 1.4295172964395384,
 2.233699311547041,
 13.053330169234309,
 1.8950491608266506,
 1.9049353402365328,
 3.1821178242889583,
 11.356156384041455,
 2.9230236888626995,
 1.200213842045202,
 2.635790807404098,
 2.253613903554444,
 9.961510215975014,
 4.826533532872274,
 1.2502926870845612,
 0.7984742276755328,
 0.840396152683202,
 0.38519924001701444,
 2.434346176233359,
 3.127909533264099,
 3.7346507724526368,
 0.0,
 4.504360509008704,
1.5579008497303448,
4.923160610490316,
7.683158068167262,
1.8223765506663006,
4.523573083259081,
4.071896214031055,
1.1746737081325782,
1.2923819784535335,
1.21232151965934,
0.8729716749950138,
2.1074756058990856,
5.901643616945057,
0.6855640840334924,
5.634300221883286,
0.7550126930090186,
0.0,
1.0321749030315608,
5.847123145760689,
3.0962647003859574,
2.068044688771067,
1.5086978898301036,
3.310688313214333,
0.5099461974827887,
1.9072963928940811,
6.152304270973421,
6.100540899228414,
1.9571596722303939,
4.580228282398262,
4.247622321534987,
0.9043278463938618,
2.499149034842446,
0.41957484389859023,
15.902258366362949,
4.741483270405726,
0.0,
2.944307722325589,
1.2980360742853914,
6.903605106108037,
1.80509590056403,
1.3540037466619592,
1.3985793924855354,
1.0855348055513168,
0.0,
1.2560044656304787,
1.2322854358550752,
2.177184869699798,
0.5913097887913589,
0.7059208750601758,
11.095126148860693,
5.5182615502859935,
2.0175432998561025,
0.5730744516191166,
0.49252656363865255,
2.800572945460828,
11.877760859906285,
2.151452815480442,
0.9853760531912669,
1.705646050191744,
0.6325957503446071,
2.5205599224510924,
5.5555911761089485,
1.2902717916405408,
2.2039006742188842,
16.056289689784453,
1.3620086545357155,
0.8339963144694047,
2.339481564210149,
1.7643888489494255,
3.2125588562896175,
1.9504898596578084,
0.0,
2.3992175140695715,
1.4694773253693545,
1.9741348182430898,
3.740438452483117,
2.176311362468996,
1.2195234142647462,
1.1528728732020785,
2.1235004521220673,
3.5739610800397585,
1.5679009708942961,
1.0121554184708625,
3.513305615404955,
0.0,
1.7841984734351823,
3.6706958728660117,
3.7726640147517996,
5.060744411441399,
2.7231858480669056,
4.715419442881193,
2.1793115241552314,
1.70866368385628,
1.19476321862848,
1.6390222354302528,
3.0685608081368967,
9.923568648564544,
1.37341302240955,
3.481380394599612,
1.0186481929664464,
3.6116963717578914,
7.2130170185656395,
0.919827160362576,
3.5885450229857203,
3.4468771664826057,
0.6774823849168342,
1.166554730338164,
0.8456297568751322,
1.481729691783599,
0.5808106243688994,
1.4819618793333265,
0.9054249595313008,
2.6227549008377546,
9.360781463495947,
1.5240431957846732,
1.8947043339560108,
3.0793847352288597,
0.0,
1.066524360439298,
3.907631850108434,
0.21523498875516978,
1.323871725273009,
1.4225749614603436,
2.31597774099886,
1.6598673834539075,
2.3359208961641458,
3.6193655097463826,
2.9547192169738765,
1.0581972284125865,
5.259926514450871,
0.9517585737436477,
2.622033614711671,
3.6544207914251037,
5.744520049956267,
1.4099935576123503,
2.3731618371467786,
0.7459983249024945,
2.037028759274383,
4.447412149306085,
6.266021875809639,
6.102571795022024,
1.136059938462684,
0.4445695655507225,
8.305590458344271,
2.9969312113645437,
1.6880697796691957,
3.4192543921764096,
13.94870159920448,
3.239160591551795,
1.3160927544902608,
6.3239664101151,
1.403449368792598,
1.824875242257348,
8.69765646872847,
1.1540979512543523,
20.07108980263479,
9.509780988930771,
1.8044575499927282,
0.2630230396164467,
1.0843264058702755,
1.1339639614584733,
3.066968595549588,
0.8447879467847005,
2.4567303881601017,
1.9441282991504945,
7.7486482543894635,
0.7549463978280366,
6.340800401734032,
1.7484731673924465,
4.615772843385764,
1.7514377249484463,
1.9070604598703718,
5.0955040732855235,
6.88898020535702,
10.074405138041094,
0.8717589098431497,
4.946085236853659,
18.213917877092683,
0.8805574307190026,
1.4136419949939485,
2.1171056720053976,
2.396713554128903,
1.428890513842054,
1.2696816522951435,
3.0761080538691283,
0.777776683309271,
4.524011276386306,
1.8316384585992311,
0.0,
0.5878310700387969,
2.2671016634316494,
6.109066282333016,
1.8301186747611875,
1.0320970913885732,
3.287281125183714,
2.5309548788864404,
2.820714319336463,
5.153358154745971,
1.0514557394368378,
0.9370347202687204,
4.0076157324919,
3.407794584431871,
0.9069650672102905,
3.1957201482771658,
1.855751233786371,
1.7816896808136653,
3.2893642153446696,
1.1102900259853528,
1.5305966750547555,
5.669289630045369,
2.0619516487367537,
2.964787207618063,
1.37439119863637,
1.4867873316830493,
2.615256304607202,
5.307339210348947,
2.0551270047297665,
2.913775111717478,
2.2854336229377803,
6.862146431500417,
1.1321098421075944,
3.8980643064935903,
1.3483004540015668,
1.8778521168502242,
2.7086127461385794,
2.849660131367168,
0.7502259308508177,
7.816854975491772,
0.8197169165531962,
0.637783187568734,
6.019992146736847,
1.0740364899145673,
1.2700700374569274,
2.9734034401985547,
3.0116147920135834,
6.815085550014233,
2.3117681592280888,
2.6167108650408415,
3.8605586262495906,
0.8376659220622726,
2.2318566446333223,
6.870033039426992,
2.1701328664295403,
2.260245533570669,
2.0675428885425866,
0.45343004068156106,
5.197545351800834,
2.050534429744812,
0.7906358101792909,
5.633170841548833,
1.7345080962077137,
2.9310646591022587,
5.836506418085698,
1.69676275592188,
0.8182452803165237,
1.558254498123731,
1.9977641638140224,
1.8739592430616279,
10.860483797080812,
0.7442516783749616,
2.2216634205992745,
9.922346214564953,
1.2482518665688118,
2.7076963816300155,
2.15681425811478,
2.0354111961964683,
18.97541789383587,
0.0,
1.6792879719299658,
0.7726437078664028,
6.601001273588086,
2.794748182640361,
0.11662473456151841,
7.989963294804828,
1.4989314914213383,
1.1376001974132999,
1.768631576406727,
4.043076617608078,
6.106378475888705,
4.253132968700945,
5.328329723185175,
4.2795004379796735,
2.4309190094703865,
6.173318526640938,
3.152673004602956,
0.8871533865617451,
2.1021187803224657,
0.7016708392612302,
3.6039242909497586,
17.181152093458017,
1.7752639784781883,
2.892455314938483,
0.7123582388742966,
0.6329587547560238,
5.464886666552504,
1.7724901954681087,
8666.409982432475,
0.848050391473813,
8.347164009631284,
1.3814764234177208,
2.003784831804579,
2.8051963320067212,
1.278470666288321,
1.6347419897288369,
1.1709104013792118,
4.172434360901016,
2.165961078746152,
0.7345049093557242,
1.676533378261942,
1.8541170093620831,
3.424185644245665,
0.0,
2.6136025292117417,
0.19864745506307777,
0.3303543804390589,
1.8324676049519892,
19.646143556405914,
3.770414619529034,
1.3649227997441662,
2.282314718096799,
2.303596956224808,
1.8249615157515897,
1.0888024691132883,
2.831060645922549,
1.6703799542221531,
17.765102857144868,
1.9584552291998643,
3.418008322105889,
2.463244768079452,
4.8558516146941155,
1.650257559898209,
1.537444614708705,
0.2825599378027148,
0.3023112495565748,
3.3508800232590463,
1.7567159657582374,
3.1399660251977752,
3.8763297477390313,
4.4109768947999335,
0.0,
1.8870437474802155,
1.8754267410108403,
8.981676918718176,
1.795792427591927,
0.5878927920839351,
1.3369185040260436,
2.2472584882969744,
1.2986515625210666,
7.314251738097366,
3.1837289930871018,
2.557923445563547,
20.10906567774995,
2.0593281719808725,
1.7191102533721345,
4.83780843371231,
0.689300644079872,
0.6487597203396355,
2.4077790198451168,
3.4855635064245534,
4.286426008464891,
2.3513078928385456,
1.532139389138959,
0.0,
1.1877918931639615,
4.322649423125438,
3.565825156790359,
0.5891569005623903,
1.714143172025093,
18.773781959289273,
1.6024941617278432,
2.27087121603153,
1.0936375681686001,
5.016570592882959,
10.520330754220202,
5.2168277536888015,
3.422279491799945,
3.8769515468826867,
0.09726395497384312,
0.7240951406553017,
0.5204338393564141,
3.7533170584463833,
4.13885049561834,
0.6952572918176088,
2.3045712719846834,
1.3135581791415392,
1.5779749196009698,
6.144441005188224,
4.6888655751009995,
2.112291107300044,
8.238326728107612,
0.885665457677127,
3.8341329919278313,
13.12250565067841,
1.7309400715206562,
1.0125791809859863,
3.1809399052764062,
1.9477408522623545,
0.8459764927908993,
1.2687659135334046,
1.3414709430292162,
0.0,
1.4588232891326043,
2.2895923323056895,
8.281032134464457,
0.92604591608269,
2.86880046589127,
2.0441538459888444,
0.23956645135763113,
3.692351849835081,
2.019396504399618,
1.0099614279388553,
1.4917185101746704,
1.51978635774001,
3.6186879383790576,
4.915393538619769,
1.1679764088591194,
3.9509978615022536,
0.0,
1.0237567701848143,
1.3942824554541593,
0.8182888341956682,
1.2266922540603835,
0.6486966346700421,
2.0517326775865308,
4.465562222562157,
0.36798407759072044,
2.257552311699234,
1.2724626621587156,
1.5426496032860373,
2.7696182273659207,
3.0030659342015555,
0.5137198062988275,
0.47050099976415233,
1.152215578608276,
2.3595576810789183,
1.8238927179091515,
0.7301496430602897,
11.794648205867638,
1.1702807628426788,
4.278186898403274,
2.139020834850892,
1.9419779675245836,
3.2200924471559387,
4.815466874301063,
12.129538821325154,
14.020468401044491,
2.468053141177706,
3.926686002760724,
1.082968680897182,
1.908434319748239,
0.9767591601490874,
6.312720052945961,
6.671440855060088,
2.1674998278671826,
5.358575939695272,
8.569564470797777,
1.4984470715648295,
2.047379865183603,
2.014341745641009,
1.6805957719951525,
1.0114910223076548,
2.4469897836708903,
0.6288854558640983,
3.816324786242919,
3.860193932056304,
1.8677046164956665,
10.192836113251293,
4.7299648223310875,
1.1449848475485134,
4.94341894373907,
2.3853207884604952,
3.59905488314125,
1.7966384583858885,
0.9219224946595967,
4.859360909188374,
1.777950048066609,
1.6167585902738237,
4.373199866470946,
4.350333025395098,
7.627247373919188,
1.3296967160736641,
3.2128052234443634,
0.7369009671532577,
1.3935106503520962,
0.9556730444087008,
1.1105039204554548,
4.9497404432382055,
4.991862892928841,
0.6466676309960426,
4.657554246822394,
2.3135943540068773,
3.058593312489884,
2.8096621236304116,
16.14444554386304,
2.613176167892551,
10.200525378601803,
1.841827555746606,
1.5784920027165998,
6.520216485705921,
1.3788036841230704,
1.7718559472541635,
4.324522024442271,
1.2998178822150275,
6.8878940388607175,
2.5144596357259785,
6.559221973372529,
0.0,
3.214164805099608,
0.7117751106482453,
2.4111707644656954,
0.0,
2.6403859631565156,
3.0065595446660582,
1.151352611762345,
3.794471155550668,
5.73178248772352,
1.0972841005012344,
1.428788072292393,
3.8393914713678123,
5.620734537791365,
21.82538785945823,
2.328867189977751,
3.1788468764468965,
1.864766514057273,
2.963557269614453,
3.4143969112138826,
2.525972999216851,
3.396135340864232,
0.8089603221053214,
1.2626375215733299,
5.295749734128347,
0.5333099584649932,
3.759930274437996,
19.147835360447623,
6.669692568901135,
6.773948621639868,
0.5598729272408635,
0.8575046049062067,
2.445397820328373,
2.819299882067107,
0.6631050632140973,
1.529175531905398,
2.8065552316422444,
2.415435767100913,
14.12883228912143,
1.6709866693739857,
2.3089477018006606,
5.069408912952638,
2.279424174203932,
2.175872183824103,
1.7822549639345722,
2.027691556820331,
4.935236863266085,
1.769930606439239,
3.5464445522995054,
1.930228556402269,
2.8967472892238733,
11.94455270475773,
1.7356454118294677,
3.179668880507512,
21.583605121331544,
3.3392491828727233,
2.802434054464456,
0.7780875857644323,
1.0555368055749053,
19.688732094442646,
1.1033684919098945,
1.330409545845683,
1.0945881283615069,
1.1278515221295327,
0.07312531440803731,
7.139796982642585,
1.5250910735289303,
2.132863748846976,
1.217169659775424,
4.8891917255868105,
2.0582005327690176,
1.5933552497708745,
1.5271127136282383,
1.4425099381741628,
2.9763896465671817,
1.9064433193421977,
2.4352030459631324,
1.5220758462984687,
2.1145179355258907,
0.4245331575392342,
1.284338033040748,
7.293702402068946,
3.9565476409007556,
0.45435419424602635,
3.182334385545619,
15.613987722715919,
1.182727946057354,
1.0031399233590037,
7.065031578957336,
3.4830842837589238,
2.207251298003133,
4.7724404291016675,
11.345043930440726,
14.591721639886863,
2.666080402801578,
20.89058588507073,
5.55314298116919,
1.0131147993453922,
1.2246396471765753,
1.4105469835413413,
1.782242992232637,
1.887976791816379,
7.268713780045777,
5.532827433154519,
1.2364921515287135,
0.7549119616813444,
0.2017058942707934,
7.9865945024153575,
3.3762046294611587,
6.5406716356446735,
2.426261251065555,
9.022244046852048,
2.494627957066291,
1.4965143701203412,
2.311715660284443,
1.501226980843212,
2.866730457175025,
0.0,
1.303413803403511,
0.6665943697135664,
2.699298171593384,
4.6038445164375705,
2.0045432916833734,
2.123431317796762,
2.0471557363265602,
0.6356044232156017,
6.481609505674108,
3.3328148795753636,
2.053888426902192,
19.1619788232694,
1.9971729869288573,
0.3971168333138258,
1.8780531712593953,
0.7254565419178712,
2.0819031925372244,
3.550899328485873,
1.8263187880684726,
2.296210737760639,
4.874023545192515,
9.649731835138887,
0.0,
4.588853209670556,
3.5783986884586465,
3.436540345098955,
0.9312765398652313,
0.6826715332999583,
1.70332048812373,
4.8486555695047695,
1.5513312701575195,
1.0068073702535072,
1.8224928357945687,
2.8553297299716744,
0.4931170059278314,
15.338397572895529,
1.2713293208229537,
3.331561741870138,
3.9625167343368672,
1.5497326059200096,
5.152817856898846,
2.2364969655721434,
3.809148114825199,
3.0454453670685764,
3.4492974348802976,
2.0277384242189176,
2.9290610134344797,
2.3443416255896694,
1.8585713607209204,
0.6093896212130628,
0.8047490492658376,
3.304256051582717,
0.9185079416078408,
3.5363177950907168,
3.407801110430096,
1.2879034431578669,
0.31016035126772595,
21.161478340510467,
2.839134060354311,
0.0,
4.839891625106934,
7.494589083026297,
1.6212933234712446,
1.3404013731030586,
3.075714506482144,
8.225477129213049,
5.429864131246641,
2.479076237927193,
14.24551960982716,
1.8827128326093123,
17.601902863505323,
2.757322651253192,
2.770005031649036,
19.302163131933405,
1.0242935350137097,
1.0322114367941204,
1.3859234785861263,
0.6896975406083913,
1.125851875869708,
1.9641104557749733,
1.1436945655908968,
0.0,
1.350391816480814,
6.3601580530711574,
1.033316530682402,
5.578543275221338,
5.774863362904877,
2.090677676851702,
5.602092869892499,
7.161452992769449,
2.832460756471197,
2.0247851919397766,
4.0752079081654164,
1.6933716363352969,
1.7685964798353426,
3.2966683584597067,
0.6330598672139752,
0.6128430173625699,
0.9372042554319255,
5.417997173496873,
6.406606504026187,
2.335206712502547,
6.689555776387006,
5.413129768788556,
2.2314265324856306,
1.3649818172237111,
2.813700460878114,
2.262515664750233,
1.5414797447137452,
5.6046793110415285,
1.5229000197903686,
17.5320034843845,
0.7473281639199638,
5.014757908337086,
4.564305581115082,
2.8945506181928304,
2.131555536480811,
0.9856030297032413,
0.40503073577719884,
1.8108124245764523,
2.987086952763552,
3.32984254393489,
0.772834881151759,
0.5724932652355229,
2.731224317059544,
5.378025803249987,
1.7182794768417162,
7.80644373034564,
6.785429462047567,
1.4245655213651893,
1.6023858287615176,
2.23528044972027,
2.9886067281847546,
2.1394393129097398,
3.530043089623396,
1.7618994146890958,
0.9090491377711108,
1.3853815603152442,
4.586366881261418,
2.1023665139757393,
4.534256429097138,
1.3904209472479547,
0.7571467452313818,
5.136243742344894,
4.938556865065128,
3.6287690778314197,
0.29093105264477015,
1.8403084964090806,
1.824288691023579,
5.728428550499833,
2.0840566460800054,
1.5748103638724262,
2.2668017020507274,
2.125461484269435,
2.4428989788380493,
2.13954990258659,
2.712606747689459,
1.7376771124581445,
2.4503482806172467,
1.9358712394820639,
2.0531163251681153,
0.0,
0.6816597044927067,
3.265221205768907,
1.4339518997993392,
1.2418717726238544,
2.2232635821474296,
2.74312403285971,
1.396495653062777,
0.7533563429767167,
1.918366338563976,
3.276090734545401,
1.0220981275353787,
0.8028731813128359,
5.367201212772081,
1.1815959751383016,
1.75364153968604,
0.7402785252804821,
2.4152364904704666,
0.958111361314147,
1.3539310414657006,
3.2411526012417027,
4.66093964473386,
0.6784269362697066,
7.832362477593706,
1.2323472113721292,
1.5729504646537174,
0.0006881096783448881,
1.0194462591538884,
0.2673044885838518,
1.4923319965749529,
1.7413080398998093,
1.4677377563732839,
2.114199690018914,
6.300863261137619,
5.398460357329391,
4.857843833938755,
4.410081739428757,
1.984787243886089,
1.1980500351934138,
7.397915631399253,
9.358968396668198,
0.9298059679606306,
4.708010402865085,
0.36597878731840056,
3.8098111050268852,
32.77994869976646,
4.755259017839335,
2.101033210408425,
1.7796155439711778,
1.2416116683275618,
1.9655005971330346,
0.044975441822248746,
0.68184564424943,
1.7351825105424725,
0.7426908936122262,
4.94680694085774,
1.2615669163699652,
2.3497105174371966,
11.304816957682212,
9.957405114050669,
6.818246737854603,
3.6356623107458077,
2.3763138233998538,
1.2398187111762753,
7.507305415911398,
3.8766178679922474,
0.38987470455768436,
1.5771435499248874,
1.4710299431608118,
2.2043613706362843,
4.014265396642929,
2.444316628523308,
0.0011469893917261562,
4.234554310158971,
1.6649876952639204,
3.209033024155447,
6.1923121478466925,
4.736899083648394,
1.7046039223136809,
2.3395153521883296,
1.153296505750822,
19.57725252335396,
1.9894121213893772,
3.512904476068075,
3.792825880046498,
2.956198730866217,
2.229959454641979,
2.487535423486973,
12.237752321044281,
1.7229164226352438,
5.0716600856411285,
4.088464938992787,
6.632904063597387,
2.9518125984276957,
1.9424496727134426,
1.1476506080296858,
1.4214939464842302,
3.24626798590751,
1.4854335767232132,
0.7914244945415305,
8.823221045719995,
0.7748849127532341,
0.574224053339419,
1.5572587259249049,
9.581064557661062,
1.9910491217540927,
0.8163556242359274,
2.735692293927604,
2.1836766349334704,
1.6379171357573095,
6.610539990722355,
0.84176399432555,
3.8236706667296887,
7.131694041260707,
4.810743324475553,
0.8890510578085677,
9.446473270022405,
4.721744721434179,
2.6637162477718785,
1.0980991988116056,
10.599939313664162,
2.447573409792674,
3.689529829970618,
1.341043427245331,
0.6398617286820958,
1.5215388206847715,
7.69004437323511,
2.3379833439380504,
2.27566569572684,
19.49158292309222,
0.6661168931067618,
2.2536868002308568,
1.02463767282862,
8.242817591432255,
1.344639742223829,
4.493648352494523,
9.17072200497737,
2.7676433882404505,
1.7570889810476737,
2.9112630931231327,
1.3245466887599284,
1.9954416277497475,
1.3414797805465732,
3.0115765585384358,
5.8412108316709705,
1.3890144546952996,
1.971155434455477,
7.360919160254177,
21.144740719676903,
1.8998668113230914,
2.7330975056713473,
0.3046672682119567,
2.3681857721120796,
7.774143109251912,
10.303600252639225,
2.3364752879742516,
0.7200963018558747,
4.356159432256988,
1.7001217778094833,
2.368860127883199,
2.568767296328695,
1.1259514789242318,
3.2279354198279515,
2.413369187015248,
1.5199867955991746,
1.4010391366175086,
...]
```

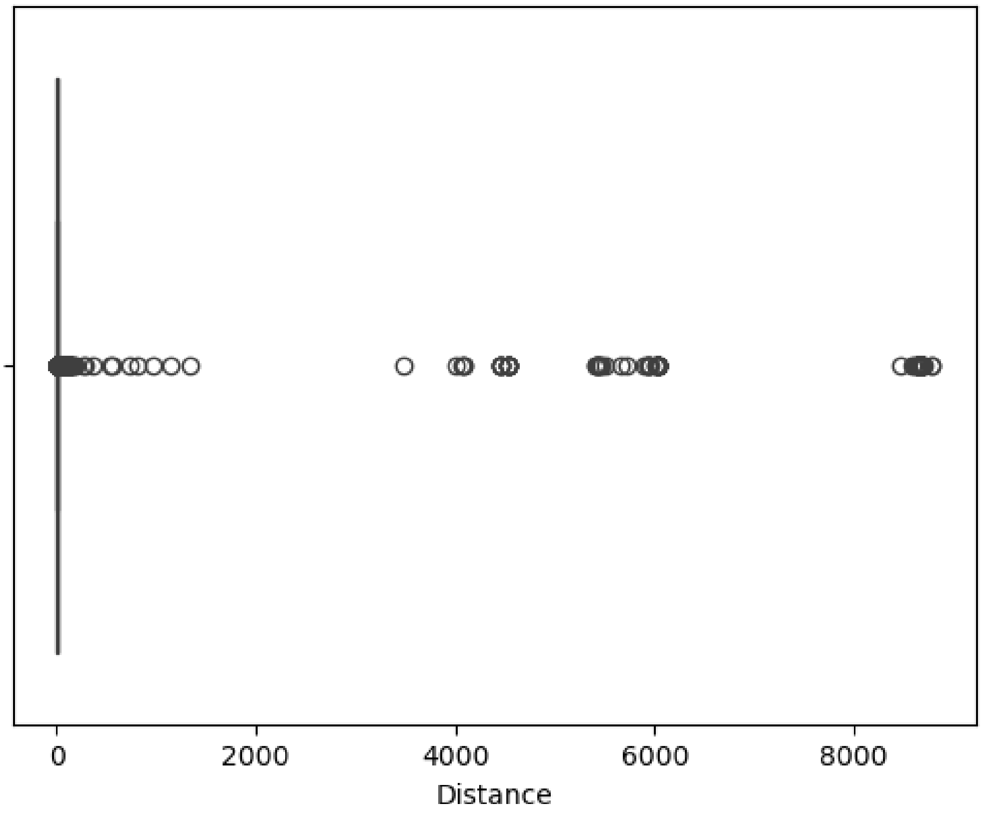
```python
df_new = df.copy()
df_new['Distance'] = distances
df = df_new
df.head()
```

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

   dropoff_longitude  dropoff_latitude  passenger_count  Distance
0         -73.999512         40.723217                1  1.683325
1         -73.994710         40.750325                1  2.457593
2         -73.962565         40.772647                1  5.036384
3         -73.965316         40.803349                3  1.661686
4         -73.973082         40.761247                5  4.475456
```

```python
sns.boxplot(data=df,x='Distance')
```

```
<Axes: xlabel='Distance'>
```



```
df = df[(df['Distance'] < 200) & (df['Distance'] > 0)]
```

```
df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])
```

```
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\1295461447.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])
```

```
df['week_day'] = df['pickup_datetime'].dt.day_name()
df['Year'] = df['pickup_datetime'].dt.year
df['Month'] = df['pickup_datetime'].dt.month
df['Hour'] = df['pickup_datetime'].dt.hour
```

```
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\2592915223.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
```

```
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['week_day'] = df['pickup_datetime'].dt.day_name()
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\2592915223.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Year'] = df['pickup_datetime'].dt.year
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\2592915223.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Month'] = df['pickup_datetime'].dt.month
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\2592915223.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Hour'] = df['pickup_datetime'].dt.hour
```

```python
df.drop(columns=['pickup_datetime','pickup_latitude','pickup_longitude
','dropoff_latitude','dropoff_longitude'],inplace=True)
```

```
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\3782303944.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy

df.drop(columns=['pickup_datetime','pickup_latitude','pickup_longitude
','dropoff_latitude','dropoff_longitude'],inplace=True)
```

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

```
   fare_amount  passenger_count  Distance  week_day  Year  Month  Hour
0          7.5                1  1.683325  Thursday  2015      5    19
1          7.7                1  2.457593    Friday  2009      7    20
2         12.9                1  5.036384    Monday  2009      8    21
```

```
3          5.3                3  1.661686    Friday  2009      6     8
4         16.0                5  4.475456  Thursday  2014      8    17
```

```python
distances = df.copy()

def convert_week_day(day):
    if day in ['Monday','Tuesday','Wednesday','Thursday','Friday']:
        return 0 # Weekday
    return 1 # Weekend

def convert_hour(hour):
    if 5 <= hour <= 12:
        return 1
    elif 12 < hour <= 17:
        return 2
    elif 17 < hour < 24:
        return 3
    return 0

df['week_day'] = distances['week_day'].apply(convert_week_day)
df['Hour'] = distances['Hour'].apply(convert_hour)
df.head()
```

```
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\3609834711.py:17:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['week_day'] = distances['week_day'].apply(convert_week_day)
C:\Users\Shubham\AppData\Local\Temp\ipykernel_28496\3609834711.py:18:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['Hour'] = distances['Hour'].apply(convert_hour)
```

```
   fare_amount  passenger_count  Distance  week_day  Year  Month  Hour
0          7.5                1  1.683325         0  2015      5     3
1          7.7                1  2.457593         0  2009      7     3
2         12.9                1  5.036384         0  2009      8     3
3          5.3                3  1.661686         0  2009      6     1
4         16.0                5  4.475456         0  2014      8     2
```

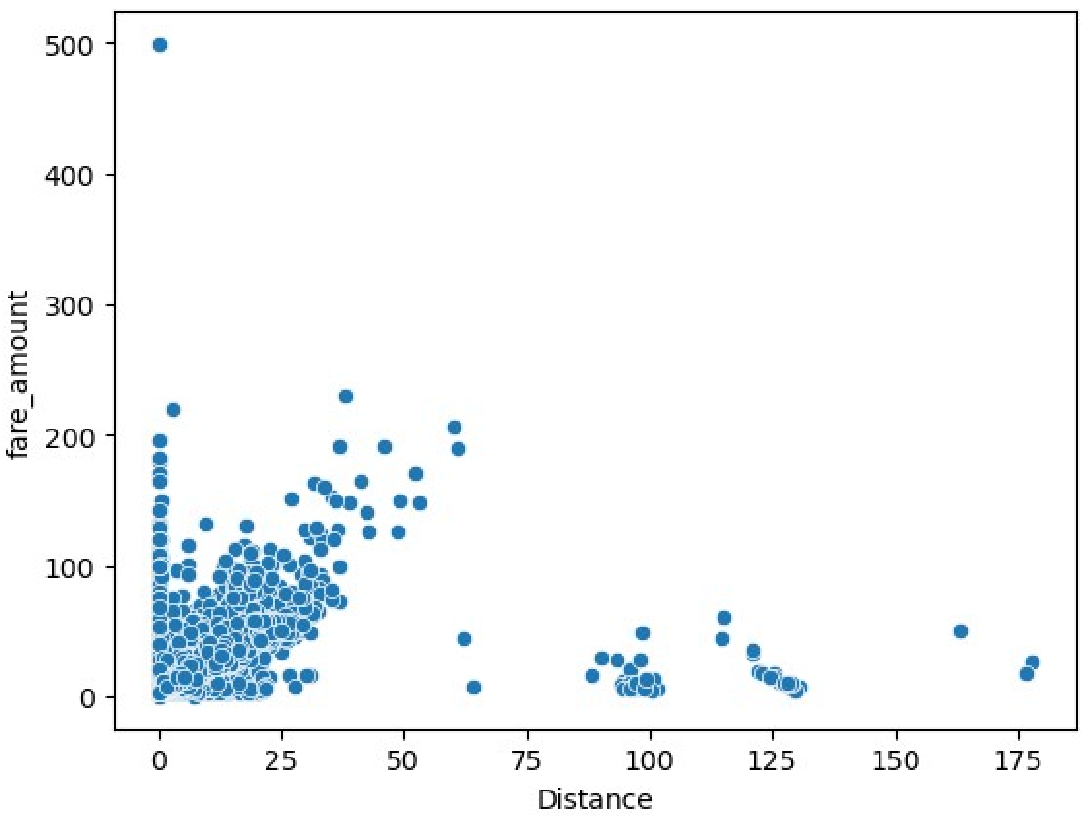
```python
df.corr()
```

```
                 fare_amount  passenger_count  Distance  week_day
Year  \
fare_amount         1.000000         0.011884  0.778667 -0.000165
0.120430
passenger_count     0.011884         1.000000  0.005112  0.042504
0.005339
Distance            0.778667         0.005112  1.000000  0.020587
0.018617
week_day           -0.000165         0.042504  0.020587  1.000000
0.010318
Year                0.120430         0.005339  0.018617  0.010318
1.000000
Month               0.024120         0.008818  0.007373 -0.006138 -
0.115182
Hour               -0.021078         0.013572 -0.022691 -0.092835
0.001131

                    Month      Hour
fare_amount      0.024120 -0.021078
passenger_count  0.008818  0.013572
Distance         0.007373 -0.022691
week_day        -0.006138 -0.092835
Year            -0.115182  0.001131
Month            1.000000 -0.005410
Hour            -0.005410  1.000000
```

```
sns.scatterplot(y=df['fare_amount'],x=df['Distance'])
```

```
<Axes: xlabel='Distance', ylabel='fare_amount'>
```



```
from sklearn.preprocessing import StandardScaler
x = df[['Distance']].values
y = df['fare_amount'].values.reshape(-1,1)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train,y_test =
train_test_split(x,y,random_state=10)

std_x = StandardScaler()
x_train = std_x.fit_transform(x_train)

x_test = std_x.transform(x_test)

std_y = StandardScaler()
y_train = std_y.fit_transform(y_train)

y_test = std_y.transform(y_test)

from sklearn.metrics import mean_squared_error,r2_score,
mean_absolute_error
def fit_predict(model):
    model.fit(x_train,y_train.ravel())
    y_pred = model.predict(x_test)
    r_squared = r2_score(y_test,y_pred)
    RMSE = mean_squared_error(y_test, y_pred,squared=False)
    MAE = mean_absolute_error(y_test,y_pred)
    print('R-squared: ', r_squared)
    print('RMSE: ', RMSE)
    print("MAE:  ",MAE)
```

```
from sklearn.linear_model import LinearRegression
```

```
fit_predict(LinearRegression())
```

```
R-squared:  0.604116792084117
RMSE:  0.6290054895695945
MAE:   0.2755232959095983
```

```
C:\Users\Shubham\anaconda3\Lib\site-packages\sklearn\metrics\
_regression.py:483: FutureWarning: 'squared' is deprecated in version
1.4 and will be removed in 1.6. To calculate the root mean squared
error, use the function'root_mean_squared_error'.
  warnings.warn(
```

```
from sklearn.ensemble import RandomForestRegressor
fit_predict(RandomForestRegressor())
```

```
----------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
Cell In[68], line 2
      1 from sklearn.ensemble import RandomForestRegressor
----> 2 fit_predict(RandomForestRegressor())

Cell In[62], line 3, in fit_predict(model)
      2 def fit_predict(model):
----> 3     model.fit(x_train,y_train.ravel())
      4     y_pred = model.predict(x_test)
      5     r_squared = r2_score(y_test,y_pred)

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1474, in
_fit_context.<locals>.decorator.<locals>.wrapper(estimator, *args,
**kwargs)
   1467     estimator._validate_params()
   1469 with config_context(
   1470     skip_parameter_validation=(
   1471         prefer_skip_nested_validation or
global_skip_validation
   1472     )
   1473 ):
-> 1474     return fit_method(estimator, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\ensemble\_forest.py:489, in
BaseForest.fit(self, X, y, sample_weight)
    478 trees = [
    479     self._make_estimator(append=False,
random_state=random_state)
    480     for i in range(n_more_estimators)
    481 ]
    483 # Parallel loop: we prefer the threading backend as the Cython
code
    484 # for fitting the trees is internally releasing the Python GIL
    485 # making threading more efficient than multiprocessing in
    486 # that case. However, for joblib 0.12+ we respect any
    487 # parallel_backend contexts set at a higher level,
    488 # since correctness does not rely on using threads.
--> 489 trees = Parallel(
    490     n_jobs=self.n_jobs,
    491     verbose=self.verbose,
    492     prefer="threads",
    493 )(
    494     delayed(_parallel_build_trees)(
    495         t,
    496         self.bootstrap,
    497         X,
    498         y,
    499         sample_weight,
    500         i,
    501         len(trees),
    502         verbose=self.verbose,
    503         class_weight=self.class_weight,
    504         n_samples_bootstrap=n_samples_bootstrap,
    505
missing_values_in_feature_mask=missing_values_in_feature_mask,
    506     )
    507     for i, t in enumerate(trees)
    508 )
    510 # Collect newly grown trees
    511 self.estimators_.extend(trees)

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:67, in
Parallel.__call__(self, iterable)
     62 config = get_config()
     63 iterable_with_config = (
     64     (_with_config(delayed_func, config), args, kwargs)
     65     for delayed_func, args, kwargs in iterable
     66 )
---> 67 return super().__call__(iterable_with_config)

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:1918, in
Parallel.__call__(self, iterable)
   1916     output = self._get_sequential_output(iterable)
   1917     next(output)
-> 1918     return output if self.return_generator else list(output)
```

```
   1920 # Let's create an ID that uniquely identifies the current
call. If the
   1921 # call is interrupted early and that the same instance is
immediately
   1922 # re-used, this id will be used to prevent workers that were
   1923 # concurrently finalizing a task from the previous call to run
the
   1924 # callback.
   1925 with self._lock:

File ~\anaconda3\Lib\site-packages\joblib\parallel.py:1847, in
Parallel._get_sequential_output(self, iterable)
   1845 self.n_dispatched_batches += 1
   1846 self.n_dispatched_tasks += 1
-> 1847 res = func(*args, **kwargs)
   1848 self.n_completed_tasks += 1
   1849 self.print_progress()

File ~\anaconda3\Lib\site-packages\sklearn\utils\parallel.py:129, in
_FuncWrapper.__call__(self, *args, **kwargs)
    127     config = {}
    128 with config_context(**config):
--> 129     return self.function(*args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\ensemble\_forest.py:192, in
_parallel_build_trees(tree, bootstrap, X, y, sample_weight, tree_idx,
n_trees, verbose, class_weight, n_samples_bootstrap,
missing_values_in_feature_mask)
    189     elif class_weight == "balanced_subsample":
    190         curr_sample_weight *=
compute_sample_weight("balanced", y, indices=indices)
--> 192     tree._fit(
    193         X,
    194         y,
    195         sample_weight=curr_sample_weight,
    196         check_input=False,
    197
missing_values_in_feature_mask=missing_values_in_feature_mask,
    198     )
    199 else:
    200     tree._fit(
    201         X,
    202         y,
   (...)
    205
missing_values_in_feature_mask=missing_values_in_feature_mask,
    206     )

File ~\anaconda3\Lib\site-packages\sklearn\tree\_classes.py:472, in
BaseDecisionTree._fit(self, X, y, sample_weight, check_input,
missing_values_in_feature_mask)
    461 else:
    462     builder = BestFirstTreeBuilder(
    463         splitter,
    464         min_samples_split,
   (...)
    469         self.min_impurity_decrease,
    470     )
--> 472 builder.build(self.tree_, X, y, sample_weight,
missing_values_in_feature_mask)
    474 if self.n_outputs_ == 1 and is_classifier(self):
    475     self.n_classes_ = self.n_classes_[0]

KeyboardInterrupt:
```