

Heamnath

20104028

Basic Analysis using Numpy and Pandas

Importing libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

importing datasets

In [2]:

```
df=pd.read_csv("fiat.csv")
df
```

Out[2]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon
0	1.0	lounge	51.0	882.0	25000.0	1.0	44.907242	8.611559868
1	2.0	pop	51.0	1186.0	32500.0	1.0	45.666359	12.24188995
2	3.0	sport	74.0	4658.0	142228.0	1.0	45.503300	11.41784
3	4.0	lounge	51.0	2739.0	160000.0	1.0	40.633171	17.63460922
4	5.0	pop	73.0	3074.0	106880.0	1.0	41.903221	12.49565029
...
1544	NaN	NaN	NaN	NaN	NaN	NaN	NaN	length
1545	NaN	NaN	NaN	NaN	NaN	NaN	NaN	concat
1546	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Null values
1547	NaN	NaN	NaN	NaN	NaN	NaN	NaN	find
1548	NaN	NaN	NaN	NaN	NaN	NaN	NaN	search

1549 rows × 11 columns

To display first 10 rows

In [3]:

```
df.head(10)
```

Out[3]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	pric
0	1.0	lounge	51.0	882.0	25000.0		1.0	44.907242	8.611559868
1	2.0	pop	51.0	1186.0	32500.0		1.0	45.666359	12.24188995
2	3.0	sport	74.0	4658.0	142228.0		1.0	45.503300	11.41784
3	4.0	lounge	51.0	2739.0	160000.0		1.0	40.633171	17.63460922
4	5.0	pop	73.0	3074.0	106880.0		1.0	41.903221	12.49565029
5	6.0	pop	74.0	3623.0	70225.0		1.0	45.000702	7.68227005
6	7.0	lounge	51.0	731.0	11600.0		1.0	44.907242	8.611559868
7	8.0	lounge	51.0	1521.0	49076.0		1.0	41.903221	12.49565029
8	9.0	sport	73.0	4049.0	76000.0		1.0	45.548000	11.54946995
9	10.0	sport	51.0	3653.0	89000.0		1.0	45.438301	10.99170017

To display last 5 rows

In [4]:

```
df.tail(5)
```

Out[4]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnan
1544	NaN	NaN		NaN		NaN	NaN	length	5	I
1545	NaN	NaN		NaN		NaN	NaN	concat	lonprice	I
1546	NaN	NaN		NaN		NaN	NaN	Null values	NO	I
1547	NaN	NaN		NaN		NaN	NaN	find	1	I
1548	NaN	NaN		NaN		NaN	NaN	search	1	I

Statistical Summary

In [5]:

```
df.describe()
```

Out[5]:

	ID	engine_power	age_in_days	km	previous_owners	lat	Unnan
count	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	1538.000000	
mean	769.500000	51.904421	1650.980494	53396.011704		1.123537	43.541361
std	444.126671	3.988023	1289.522278	40046.830723		0.416423	2.133518
min	1.000000	51.000000	366.000000	1232.000000		1.000000	36.855839
25%	385.250000	51.000000	670.000000	20006.250000		1.000000	41.802990

	ID	engine_power	age_in_days	km	previous_owners	lat	Unnamed
50%	769.500000	51.000000	1035.000000	39031.000000	1.000000	44.394096	↑
75%	1150.750000	51.000000	2616.000000	70667.750000	1.000000	45.167060	↗

To print no of rows and columns

In [6]: `df.shape`

Out[6]: (1549, 11)

To print total no of elements

In [7]: `df.size`

Out[7]: 17039

To find the null value

In [8]: `df.isna()`

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon	price	Unnamed
0	False	False	False	False	False	False	False	False	False	True
1	False	False	False	False	False	False	False	False	False	True
2	False	False	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False	False	True
4	False	False	False	False	False	False	False	False	False	True
...
1544	True	True	True	True	True	True	True	True	False	True
1545	True	True	True	True	True	True	True	True	False	True
1546	True	True	True	True	True	True	True	True	False	True
1547	True	True	True	True	True	True	True	True	False	True
1548	True	True	True	True	True	True	True	True	False	True

1549 rows × 11 columns

To fill the missing value

In [9]:

```
df.fillna(value=0)
```

Out[9]:

	ID	model	engine_power	age_in_days	km	previous_owners	lat	lon
0	1.0	lounge		51.0	882.0	25000.0		
1	2.0	pop		51.0	1186.0	32500.0		
2	3.0	sport		74.0	4658.0	142228.0		
3	4.0	lounge		51.0	2739.0	160000.0		
4	5.0	pop		73.0	3074.0	106880.0		
...
1544	0.0	0		0.0	0.0	0.0	0.000000	length
1545	0.0	0		0.0	0.0	0.0	0.000000	concat loi
1546	0.0	0		0.0	0.0	0.0	0.000000	Null values
1547	0.0	0		0.0	0.0	0.0	0.000000	find
1548	0.0	0		0.0	0.0	0.0	0.000000	search

1549 rows × 11 columns

Print column names

In [10]:

```
df.columns
```

Out[10]:

```
Index(['ID', 'model', 'engine_power', 'age_in_days', 'km', 'previous_owners',
       'lat', 'lon', 'price', 'Unnamed: 9', 'Unnamed: 10'],
      dtype='object')
```

To print particular column names

In [11]:

```
data=df[["km","price"]]
data
```

Out[11]:

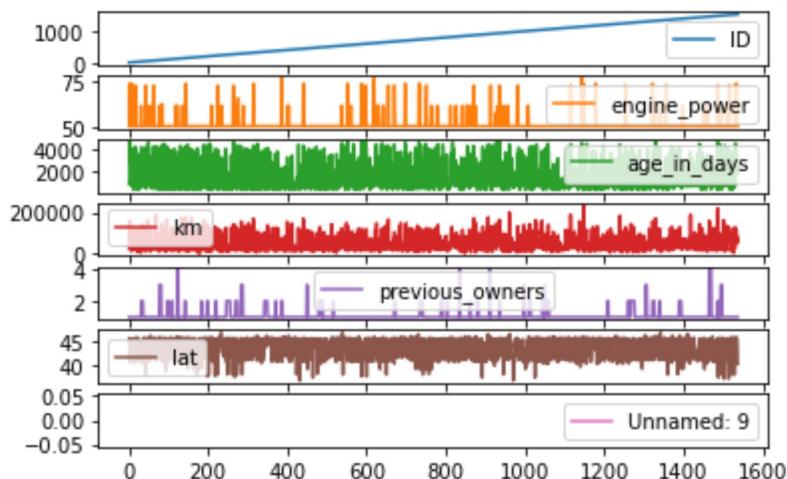
	km	price
0	25000.0	8900
1	32500.0	8800
2	142228.0	4200
3	160000.0	6000
4	106880.0	5700
...

	km	price
1544	NaN	5
1545	NaN	lonprice
1546	NaN	NO
1547	NaN	1
1548	NaN	1

Line chart with subplots

```
In [12]: df.plot.line(subplots=True)
```

```
Out[12]: array([<AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>,
   <AxesSubplot:>, <AxesSubplot:>, <AxesSubplot:>], dtype=object)
```



Line chart

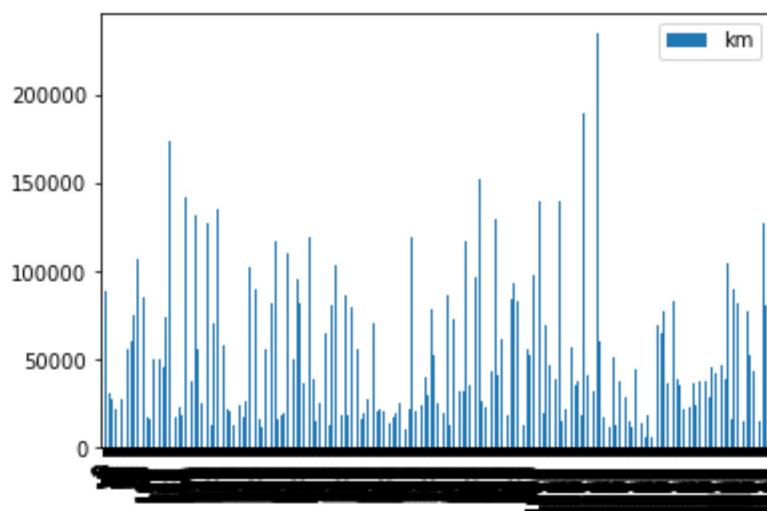
```
In [13]: df.plot.line()
```

```
Out[13]: <AxesSubplot:>
```

Bar chart

In [14]: `data.plot.bar()`

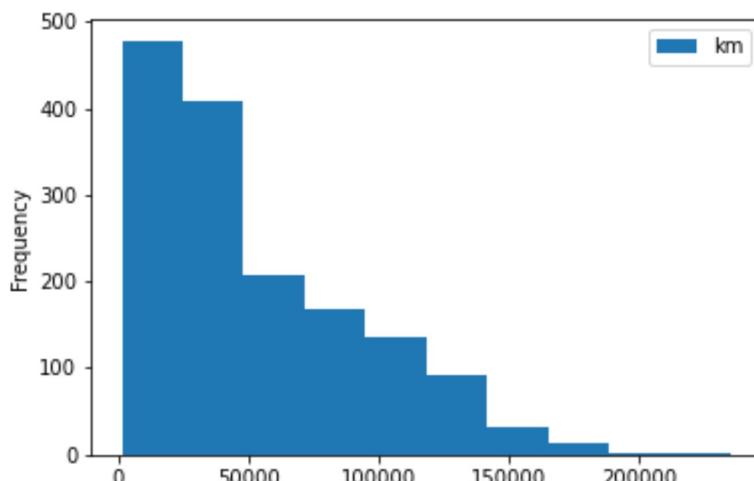
Out[14]: <AxesSubplot:>



Histogram

In [15]: `data.plot.hist()`

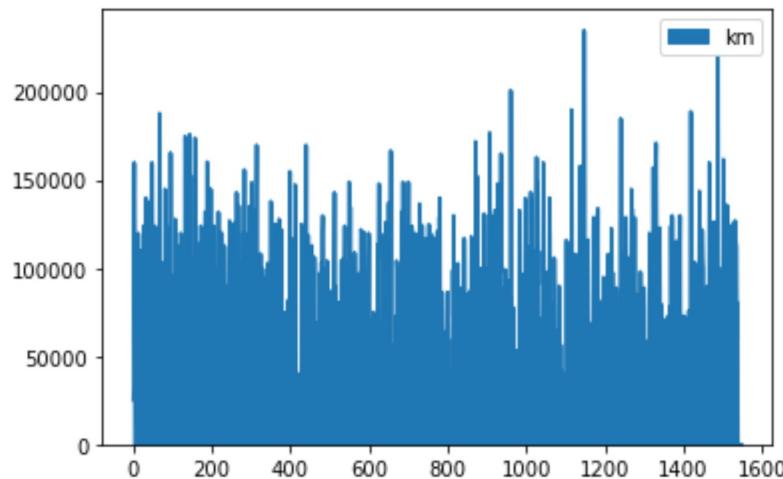
Out[15]: <AxesSubplot:ylabel='Frequency'>



Area chart

In [16]: `data.plot.area()`

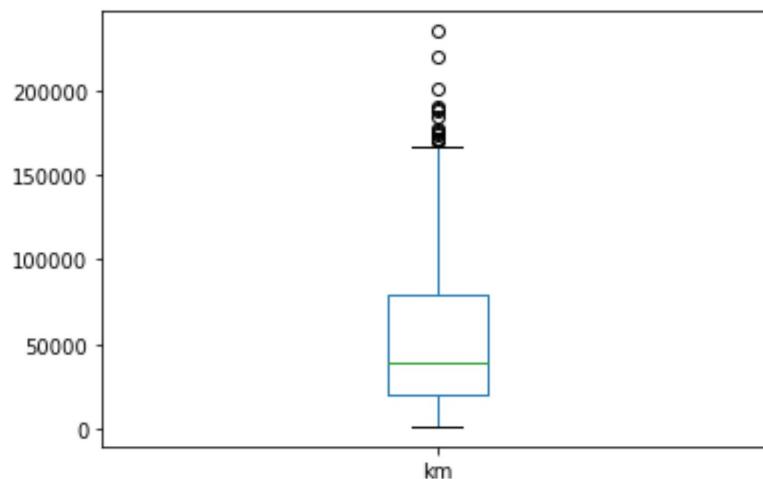
Out[16]: <AxesSubplot:>



Box chart

In [17]: `data.plot.box()`

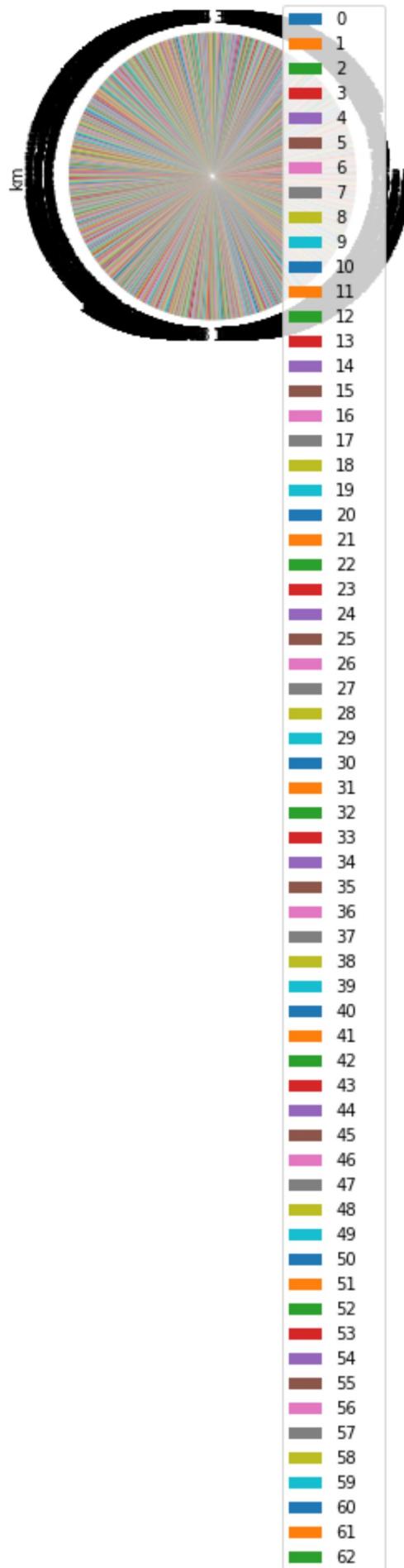
Out[17]: <AxesSubplot:>



Pie chart

In [18]: `data.plot.pie(y='km')`

Out[18]: <AxesSubplot:ylabel='km'>



■	63
■	64
■	65
■	66
■	67
■	68
■	69
■	70
■	71
■	72
■	73
■	74
■	75
■	76
■	77
■	78
■	79
■	80
■	81
■	82
■	83
■	84
■	85
■	86
■	87
■	88
■	89
■	90
■	91
■	92
■	93
■	94
■	95
■	96
■	97
■	98
■	99
■	100
■	101
■	102
■	103
■	104
■	105
■	106
■	107
■	108
■	109
■	110
■	111
■	112
■	113
■	114
■	115
■	116
■	117
■	118
■	119
■	120
■	121
■	122
■	123
■	124
■	125
■	126

■	127
■	128
■	129
■	130
■	131
■	132
■	133
■	134
■	135
■	136
■	137
■	138
■	139
■	140
■	141
■	142
■	143
■	144
■	145
■	146
■	147
■	148
■	149
■	150
■	151
■	152
■	153
■	154
■	155
■	156
■	157
■	158
■	159
■	160
■	161
■	162
■	163
■	164
■	165
■	166
■	167
■	168
■	169
■	170
■	171
■	172
■	173
■	174
■	175
■	176
■	177
■	178
■	179
■	180
■	181
■	182
■	183
■	184
■	185
■	186
■	187
■	188
■	189
■	190

■	191
■	192
■	193
■	194
■	195
■	196
■	197
■	198
■	199
■	200
■	201
■	202
■	203
■	204
■	205
■	206
■	207
■	208
■	209
■	210
■	211
■	212
■	213
■	214
■	215
■	216
■	217
■	218
■	219
■	220
■	221
■	222
■	223
■	224
■	225
■	226
■	227
■	228
■	229
■	230
■	231
■	232
■	233
■	234
■	235
■	236
■	237
■	238
■	239
■	240
■	241
■	242
■	243
■	244
■	245
■	246
■	247
■	248
■	249
■	250
■	251
■	252
■	253
■	254

■	255
■	256
■	257
■	258
■	259
■	260
■	261
■	262
■	263
■	264
■	265
■	266
■	267
■	268
■	269
■	270
■	271
■	272
■	273
■	274
■	275
■	276
■	277
■	278
■	279
■	280
■	281
■	282
■	283
■	284
■	285
■	286
■	287
■	288
■	289
■	290
■	291
■	292
■	293
■	294
■	295
■	296
■	297
■	298
■	299
■	300
■	301
■	302
■	303
■	304
■	305
■	306
■	307
■	308
■	309
■	310
■	311
■	312
■	313
■	314
■	315
■	316
■	317
■	318

319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382

■	383
■	384
■	385
■	386
■	387
■	388
■	389
■	390
■	391
■	392
■	393
■	394
■	395
■	396
■	397
■	398
■	399
■	400
■	401
■	402
■	403
■	404
■	405
■	406
■	407
■	408
■	409
■	410
■	411
■	412
■	413
■	414
■	415
■	416
■	417
■	418
■	419
■	420
■	421
■	422
■	423
■	424
■	425
■	426
■	427
■	428
■	429
■	430
■	431
■	432
■	433
■	434
■	435
■	436
■	437
■	438
■	439
■	440
■	441
■	442
■	443
■	444
■	445
■	446

■	447
■	448
■	449
■	450
■	451
■	452
■	453
■	454
■	455
■	456
■	457
■	458
■	459
■	460
■	461
■	462
■	463
■	464
■	465
■	466
■	467
■	468
■	469
■	470
■	471
■	472
■	473
■	474
■	475
■	476
■	477
■	478
■	479
■	480
■	481
■	482
■	483
■	484
■	485
■	486
■	487
■	488
■	489
■	490
■	491
■	492
■	493
■	494
■	495
■	496
■	497
■	498
■	499
■	500
■	501
■	502
■	503
■	504
■	505
■	506
■	507
■	508
■	509
■	510

■	511
■	512
■	513
■	514
■	515
■	516
■	517
■	518
■	519
■	520
■	521
■	522
■	523
■	524
■	525
■	526
■	527
■	528
■	529
■	530
■	531
■	532
■	533
■	534
■	535
■	536
■	537
■	538
■	539
■	540
■	541
■	542
■	543
■	544
■	545
■	546
■	547
■	548
■	549
■	550
■	551
■	552
■	553
■	554
■	555
■	556
■	557
■	558
■	559
■	560
■	561
■	562
■	563
■	564
■	565
■	566
■	567
■	568
■	569
■	570
■	571
■	572
■	573
■	574

■	575
■	576
■	577
■	578
■	579
■	580
■	581
■	582
■	583
■	584
■	585
■	586
■	587
■	588
■	589
■	590
■	591
■	592
■	593
■	594
■	595
■	596
■	597
■	598
■	599
■	600
■	601
■	602
■	603
■	604
■	605
■	606
■	607
■	608
■	609
■	610
■	611
■	612
■	613
■	614
■	615
■	616
■	617
■	618
■	619
■	620
■	621
■	622
■	623
■	624
■	625
■	626
■	627
■	628
■	629
■	630
■	631
■	632
■	633
■	634
■	635
■	636
■	637
■	638

639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702

■	703
■	704
■	705
■	706
■	707
■	708
■	709
■	710
■	711
■	712
■	713
■	714
■	715
■	716
■	717
■	718
■	719
■	720
■	721
■	722
■	723
■	724
■	725
■	726
■	727
■	728
■	729
■	730
■	731
■	732
■	733
■	734
■	735
■	736
■	737
■	738
■	739
■	740
■	741
■	742
■	743
■	744
■	745
■	746
■	747
■	748
■	749
■	750
■	751
■	752
■	753
■	754
■	755
■	756
■	757
■	758
■	759
■	760
■	761
■	762
■	763
■	764
■	765
■	766

■	767
■	768
■	769
■	770
■	771
■	772
■	773
■	774
■	775
■	776
■	777
■	778
■	779
■	780
■	781
■	782
■	783
■	784
■	785
■	786
■	787
■	788
■	789
■	790
■	791
■	792
■	793
■	794
■	795
■	796
■	797
■	798
■	799
■	800
■	801
■	802
■	803
■	804
■	805
■	806
■	807
■	808
■	809
■	810
■	811
■	812
■	813
■	814
■	815
■	816
■	817
■	818
■	819
■	820
■	821
■	822
■	823
■	824
■	825
■	826
■	827
■	828
■	829
■	830

■	831
■	832
■	833
■	834
■	835
■	836
■	837
■	838
■	839
■	840
■	841
■	842
■	843
■	844
■	845
■	846
■	847
■	848
■	849
■	850
■	851
■	852
■	853
■	854
■	855
■	856
■	857
■	858
■	859
■	860
■	861
■	862
■	863
■	864
■	865
■	866
■	867
■	868
■	869
■	870
■	871
■	872
■	873
■	874
■	875
■	876
■	877
■	878
■	879
■	880
■	881
■	882
■	883
■	884
■	885
■	886
■	887
■	888
■	889
■	890
■	891
■	892
■	893
■	894

■	895
■	896
■	897
■	898
■	899
■	900
■	901
■	902
■	903
■	904
■	905
■	906
■	907
■	908
■	909
■	910
■	911
■	912
■	913
■	914
■	915
■	916
■	917
■	918
■	919
■	920
■	921
■	922
■	923
■	924
■	925
■	926
■	927
■	928
■	929
■	930
■	931
■	932
■	933
■	934
■	935
■	936
■	937
■	938
■	939
■	940
■	941
■	942
■	943
■	944
■	945
■	946
■	947
■	948
■	949
■	950
■	951
■	952
■	953
■	954
■	955
■	956
■	957
■	958

959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022

■	1023
■	1024
■	1025
■	1026
■	1027
■	1028
■	1029
■	1030
■	1031
■	1032
■	1033
■	1034
■	1035
■	1036
■	1037
■	1038
■	1039
■	1040
■	1041
■	1042
■	1043
■	1044
■	1045
■	1046
■	1047
■	1048
■	1049
■	1050
■	1051
■	1052
■	1053
■	1054
■	1055
■	1056
■	1057
■	1058
■	1059
■	1060
■	1061
■	1062
■	1063
■	1064
■	1065
■	1066
■	1067
■	1068
■	1069
■	1070
■	1071
■	1072
■	1073
■	1074
■	1075
■	1076
■	1077
■	1078
■	1079
■	1080
■	1081
■	1082
■	1083
■	1084
■	1085
■	1086

■	1087
■	1088
■	1089
■	1090
■	1091
■	1092
■	1093
■	1094
■	1095
■	1096
■	1097
■	1098
■	1099
■	1100
■	1101
■	1102
■	1103
■	1104
■	1105
■	1106
■	1107
■	1108
■	1109
■	1110
■	1111
■	1112
■	1113
■	1114
■	1115
■	1116
■	1117
■	1118
■	1119
■	1120
■	1121
■	1122
■	1123
■	1124
■	1125
■	1126
■	1127
■	1128
■	1129
■	1130
■	1131
■	1132
■	1133
■	1134
■	1135
■	1136
■	1137
■	1138
■	1139
■	1140
■	1141
■	1142
■	1143
■	1144
■	1145
■	1146
■	1147
■	1148
■	1149
■	1150

■	1151
■	1152
■	1153
■	1154
■	1155
■	1156
■	1157
■	1158
■	1159
■	1160
■	1161
■	1162
■	1163
■	1164
■	1165
■	1166
■	1167
■	1168
■	1169
■	1170
■	1171
■	1172
■	1173
■	1174
■	1175
■	1176
■	1177
■	1178
■	1179
■	1180
■	1181
■	1182
■	1183
■	1184
■	1185
■	1186
■	1187
■	1188
■	1189
■	1190
■	1191
■	1192
■	1193
■	1194
■	1195
■	1196
■	1197
■	1198
■	1199
■	1200
■	1201
■	1202
■	1203
■	1204
■	1205
■	1206
■	1207
■	1208
■	1209
■	1210
■	1211
■	1212
■	1213
■	1214

■	1215
■	1216
■	1217
■	1218
■	1219
■	1220
■	1221
■	1222
■	1223
■	1224
■	1225
■	1226
■	1227
■	1228
■	1229
■	1230
■	1231
■	1232
■	1233
■	1234
■	1235
■	1236
■	1237
■	1238
■	1239
■	1240
■	1241
■	1242
■	1243
■	1244
■	1245
■	1246
■	1247
■	1248
■	1249
■	1250
■	1251
■	1252
■	1253
■	1254
■	1255
■	1256
■	1257
■	1258
■	1259
■	1260
■	1261
■	1262
■	1263
■	1264
■	1265
■	1266
■	1267
■	1268
■	1269
■	1270
■	1271
■	1272
■	1273
■	1274
■	1275
■	1276
■	1277
■	1278

■	1279
■	1280
■	1281
■	1282
■	1283
■	1284
■	1285
■	1286
■	1287
■	1288
■	1289
■	1290
■	1291
■	1292
■	1293
■	1294
■	1295
■	1296
■	1297
■	1298
■	1299
■	1300
■	1301
■	1302
■	1303
■	1304
■	1305
■	1306
■	1307
■	1308
■	1309
■	1310
■	1311
■	1312
■	1313
■	1314
■	1315
■	1316
■	1317
■	1318
■	1319
■	1320
■	1321
■	1322
■	1323
■	1324
■	1325
■	1326
■	1327
■	1328
■	1329
■	1330
■	1331
■	1332
■	1333
■	1334
■	1335
■	1336
■	1337
■	1338
■	1339
■	1340
■	1341
■	1342

■	1343
■	1344
■	1345
■	1346
■	1347
■	1348
■	1349
■	1350
■	1351
■	1352
■	1353
■	1354
■	1355
■	1356
■	1357
■	1358
■	1359
■	1360
■	1361
■	1362
■	1363
■	1364
■	1365
■	1366
■	1367
■	1368
■	1369
■	1370
■	1371
■	1372
■	1373
■	1374
■	1375
■	1376
■	1377
■	1378
■	1379
■	1380
■	1381
■	1382
■	1383
■	1384
■	1385
■	1386
■	1387
■	1388
■	1389
■	1390
■	1391
■	1392
■	1393
■	1394
■	1395
■	1396
■	1397
■	1398
■	1399
■	1400
■	1401
■	1402
■	1403
■	1404
■	1405
■	1406

■	1407
■	1408
■	1409
■	1410
■	1411
■	1412
■	1413
■	1414
■	1415
■	1416
■	1417
■	1418
■	1419
■	1420
■	1421
■	1422
■	1423
■	1424
■	1425
■	1426
■	1427
■	1428
■	1429
■	1430
■	1431
■	1432
■	1433
■	1434
■	1435
■	1436
■	1437
■	1438
■	1439
■	1440
■	1441
■	1442
■	1443
■	1444
■	1445
■	1446
■	1447
■	1448
■	1449
■	1450
■	1451
■	1452
■	1453
■	1454
■	1455
■	1456
■	1457
■	1458
■	1459
■	1460
■	1461
■	1462
■	1463
■	1464
■	1465
■	1466
■	1467
■	1468
■	1469
■	1470

■	1471
■	1472
■	1473
■	1474
■	1475
■	1476
■	1477
■	1478
■	1479
■	1480
■	1481
■	1482
■	1483
■	1484
■	1485
■	1486
■	1487
■	1488
■	1489
■	1490
■	1491
■	1492
■	1493
■	1494
■	1495
■	1496
■	1497
■	1498
■	1499
■	1500
■	1501
■	1502
■	1503
■	1504
■	1505
■	1506
■	1507
■	1508
■	1509
■	1510
■	1511
■	1512
■	1513
■	1514
■	1515
■	1516
■	1517
■	1518
■	1519
■	1520
■	1521
■	1522
■	1523
■	1524
■	1525
■	1526
■	1527
■	1528
■	1529
■	1530
■	1531
■	1532
■	1533
■	1534



Scatter chart

```
In [19]: data.plot.scatter(x='km',y='price')
```

```
Out[19]: <AxesSubplot:xlabel='km', ylabel='price'>
```

