Untitled1

April 4, 2023

[1]:
```
!pip install pandas
```

```
Requirement already satisfied: pandas in /Users/h4/anaconda3/lib/python3.9/site-
packages (1.4.2)
Requirement already satisfied: python-dateutil>=2.8.1 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from pandas) (2021.3)
Requirement already satisfied: numpy>=1.20.0 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from pandas) (1.23.2)
Requirement already satisfied: six>=1.5 in
/Users/h4/anaconda3/lib/python3.9/site-packages (from python-
dateutil>=2.8.1->pandas) (1.16.0)
```

[2]:
```
import pandas as pd
```

[3]:
```
# wie definieren eine Variable, welche die Daten enthält in Form eines␣
↪Dataframes

df = pd.read_csv('/Users/h4/desktop/test.csv')
```

[4]:
```
df.head()
```

[4]:

	Spalte Kategorische;Spalte Numerisch;Spalte Numerisch 2;;;
0	Klasse A;12;120;;;
1	Klasse A;23;130;;;
2	Klasse B;31;89;;;
3	Klasse B;34;90;;;
4	Klasse C;1;7;;;

[5]:
```
df = pd.read_csv('/Users/h4/desktop/test.csv', sep=';')
```

[6]:
```
df.head()
```

[6]:

	Spalte Kategorische	Spalte Numerisch	Spalte Numerisch 2	Unnamed: 3	\
0	Klasse A	12	120	NaN	
1	Klasse A	23	130	NaN	
2	Klasse B	31	89	NaN	

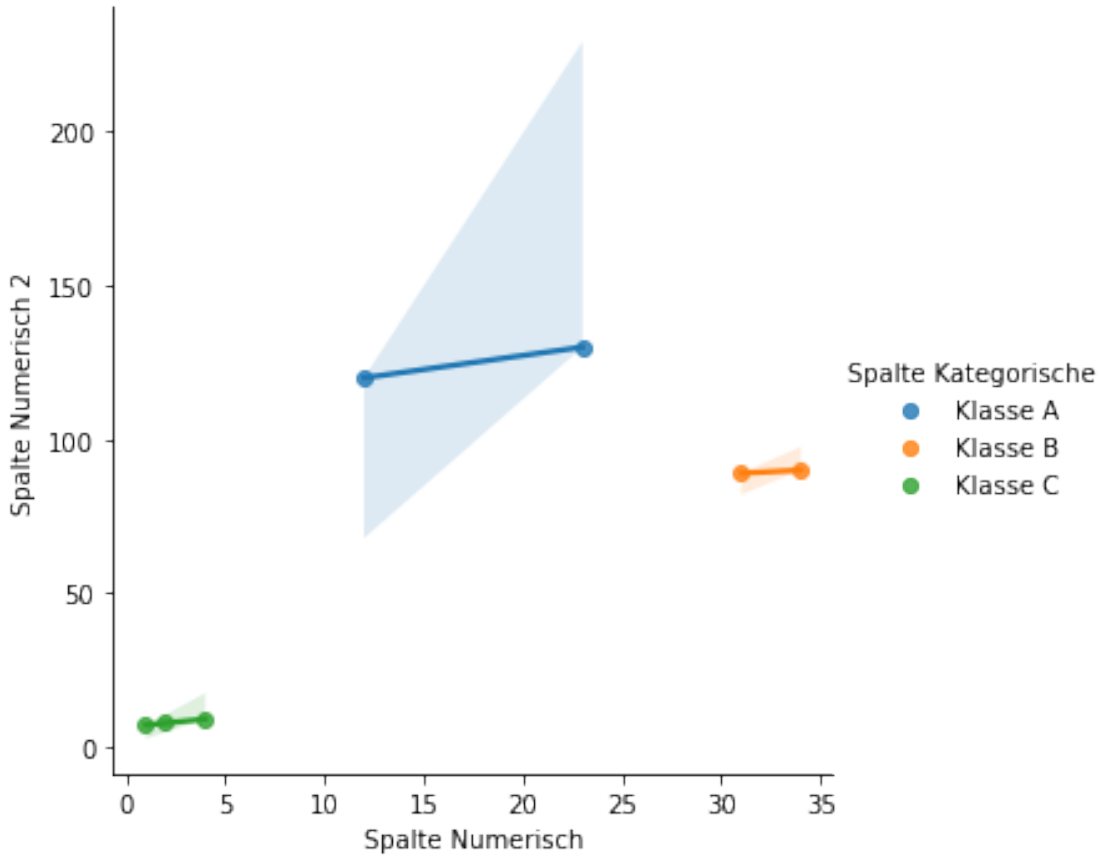
```
3            Klasse B                34                  90         NaN
4            Klasse C                 1                   7         NaN

   Unnamed: 4  Unnamed: 5
0         NaN         NaN
1         NaN         NaN
2         NaN         NaN
3         NaN         NaN
4         NaN         NaN
```

[8]:
```
import seaborn as sns
import matplotlib.pyplot as plt


sns.lmplot(x = 'Spalte Numerisch', y = 'Spalte Numerisch 2', data=df,␣
 ↪hue='Spalte Kategorische')
```

[8]:
```
<seaborn.axisgrid.FacetGrid at 0x2953a6070>
```



[9]:
```
# Dateien JSON = Wörterbuch

import pandas as pd

df = pd.read_json('/Users/h4/desktop/20230404_data.json')

print(df.to_string())
```

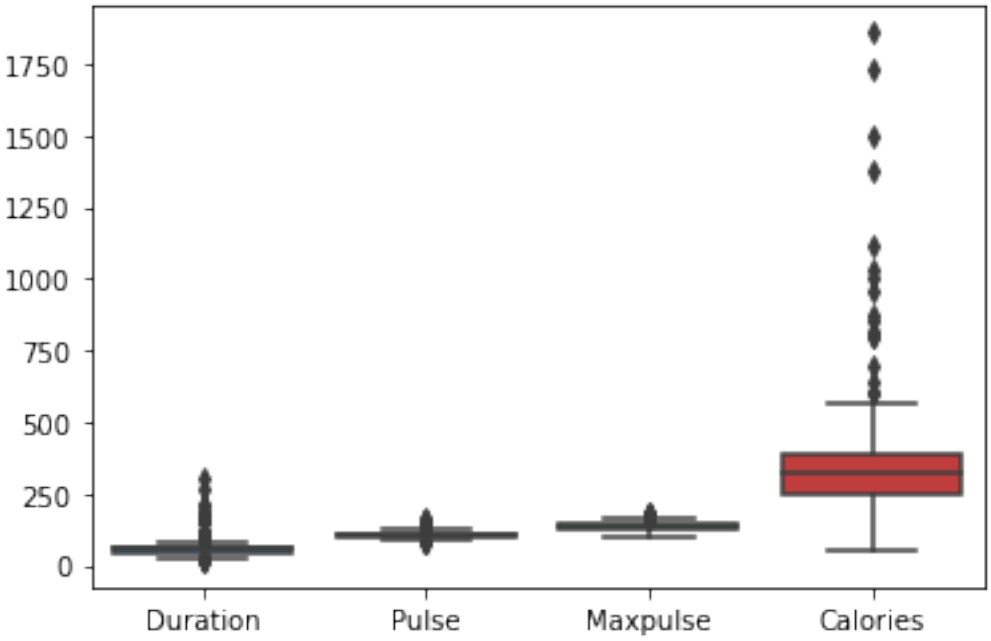
```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
5          60    102       127     300.5
6          60    110       136     374.0
7          45    104       134     253.3
8          30    109       133     195.1
9          60     98       124     269.0
10         60    103       147     329.3
11         60    100       120     250.7
12         60    106       128     345.3
13         60    104       132     379.3
14         60     98       123     275.0
15         60     98       120     215.2
16         60    100       120     300.0
17         45     90       112       NaN
18         60    103       123     323.0
19         45     97       125     243.0
20         60    108       131     364.2
21         45    100       119     282.0
22         60    130       101     300.0
23         45    105       132     246.0
24         60    102       126     334.5
25         60    100       120     250.0
26         60     92       118     241.0
27         60    103       132       NaN
28         60    100       132     280.0
29         60    102       129     380.3
30         60     92       115     243.0
31         45     90       112     180.1
32         60    101       124     299.0
33         60     93       113     223.0
34         60    107       136     361.0
35         60    114       140     415.0
36         60    102       127     300.5
37         60    100       120     300.1
```

38	60	100	120	300.0
39	45	104	129	266.0
40	45	90	112	180.1
41	60	98	126	286.0
42	60	100	122	329.4
43	60	111	138	400.0
44	60	111	131	397.0
45	60	99	119	273.0
46	60	109	153	387.6
47	45	111	136	300.0
48	45	108	129	298.0
49	60	111	139	397.6
50	60	107	136	380.2
51	80	123	146	643.1
52	60	106	130	263.0
53	60	118	151	486.0
54	30	136	175	238.0
55	60	121	146	450.7
56	60	118	121	413.0
57	45	115	144	305.0
58	20	153	172	226.4
59	45	123	152	321.0
60	210	108	160	1376.0
61	160	110	137	1034.4
62	160	109	135	853.0
63	45	118	141	341.0
64	20	110	130	131.4
65	180	90	130	800.4
66	150	105	135	873.4
67	150	107	130	816.0
68	20	106	136	110.4
69	300	108	143	1500.2
70	150	97	129	1115.0
71	60	109	153	387.6
72	90	100	127	700.0
73	150	97	127	953.2
74	45	114	146	304.0
75	90	98	125	563.2
76	45	105	134	251.0
77	45	110	141	300.0
78	120	100	130	500.4
79	270	100	131	1729.0
80	30	159	182	319.2
81	45	149	169	344.0
82	30	103	139	151.1
83	120	100	130	500.0
84	45	100	120	225.3
85	30	151	170	300.1

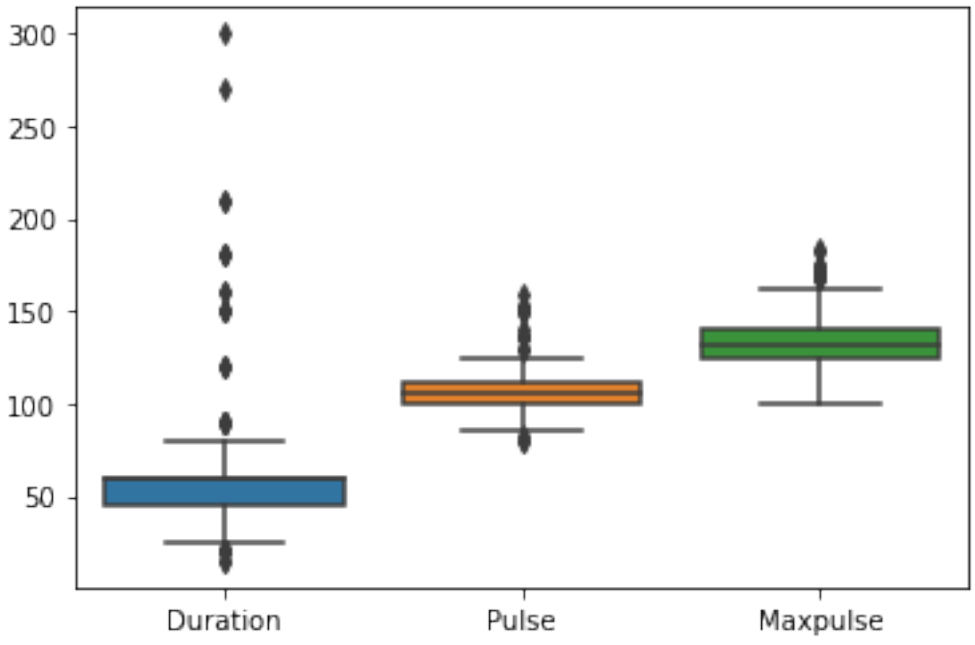
86	45	102	136	234.0
87	120	100	157	1000.1
88	45	129	103	242.0
89	20	83	107	50.3
90	180	101	127	600.1
91	45	107	137	NaN
92	30	90	107	105.3
93	15	80	100	50.5
94	20	150	171	127.4
95	20	151	168	229.4
96	30	95	128	128.2
97	25	152	168	244.2
98	30	109	131	188.2
99	90	93	124	604.1
100	20	95	112	77.7
101	90	90	110	500.0
102	90	90	100	500.0
103	90	90	100	500.4
104	30	92	108	92.7
105	30	93	128	124.0
106	180	90	120	800.3
107	30	90	120	86.2
108	90	90	120	500.3
109	210	137	184	1860.4
110	60	102	124	325.2
111	45	107	124	275.0
112	15	124	139	124.2
113	45	100	120	225.3
114	60	108	131	367.6
115	60	108	151	351.7
116	60	116	141	443.0
117	60	97	122	277.4
118	60	105	125	NaN
119	60	103	124	332.7
120	30	112	137	193.9
121	45	100	120	100.7
122	60	119	169	336.7
123	60	107	127	344.9
124	60	111	151	368.5
125	60	98	122	271.0
126	60	97	124	275.3
127	60	109	127	382.0
128	90	99	125	466.4
129	60	114	151	384.0
130	60	104	134	342.5
131	60	107	138	357.5
132	60	103	133	335.0
133	60	106	132	327.5

```
134        60    103        136     339.0
135        20    136        156     189.0
136        45    117        143     317.7
137        45    115        137     318.0
138        45    113        138     308.0
139        20    141        162     222.4
140        60    108        135     390.0
141        60     97        127       NaN
142        45    100        120     250.4
143        45    122        149     335.4
144        60    136        170     470.2
145        45    106        126     270.8
146        60    107        136     400.0
147        60    112        146     361.9
148        30    103        127     185.0
149        60    110        150     409.4
150        60    106        134     343.0
151        60    109        129     353.2
152        60    109        138     374.0
153        30    150        167     275.8
154        60    105        128     328.0
155        60    111        151     368.5
156        60     97        131     270.4
157        60    100        120     270.4
158        60    114        150     382.8
159        30     80        120     240.9
160        30     85        120     250.4
161        45     90        130     260.4
162        45     95        130     270.0
163        45    100        140     280.9
164        60    105        140     290.8
165        60    110        145     300.4
166        60    115        145     310.2
167        75    120        150     320.4
168        75    125        150     330.4
```

[10]:
```
sns.boxplot(data=df)
plt.show()
```



[12]:
```
stats_df = df.drop(['Calories'], axis = 1)
sns.boxplot(data=stats_df)
plt.show()
```



[13]:
```
df
```

[13]:
```
     Duration  Pulse  Maxpulse  Calories
0          60    110       130     409.1
1          60    117       145     479.0
2          60    103       135     340.0
3          45    109       175     282.4
4          45    117       148     406.0
..        ...    ...       ...       ...
164        60    105       140     290.8
165        60    110       145     300.4
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4

[169 rows x 4 columns]
```

[14]:
```
print(df.head(10)) # es werden die ersten 10 Zeilen dargestellt
```

```
   Duration  Pulse  Maxpulse  Calories
0        60    110       130     409.1
1        60    117       145     479.0
2        60    103       135     340.0
3        45    109       175     282.4
4        45    117       148     406.0
5        60    102       127     300.5
6        60    110       136     374.0
7        45    104       134     253.3
8        30    109       133     195.1
9        60     98       124     269.0
```

[15]:
```
print(df.tail(10)) # es werden die letzten 10 Zeilen dargestellt
```

```
     Duration  Pulse  Maxpulse  Calories
159        30     80       120     240.9
160        30     85       120     250.4
161        45     90       130     260.4
162        45     95       130     270.0
163        45    100       140     280.9
164        60    105       140     290.8
165        60    110       145     300.4
166        60    115       145     310.2
167        75    120       150     320.4
168        75    125       150     330.4
```

[17]:
```
# um bestimmte Zeilen oder Spalten Positionen zu zeigen, nutzen die Funktion␣
↪von Pandas "iloc"
```

```
# df.iloc[row, column]

df.iloc[8:14, :] # zeigt Zeile 9 bis 15 und Alle Spalten
```

[17]:

	Duration	Pulse	Maxpulse	Calories
8	30	109	133	195.1
9	60	98	124	269.0
10	60	103	147	329.3
11	60	100	120	250.7
12	60	106	128	345.3
13	60	104	132	379.3

[19]:
```
# Zeile 12 bis 23 und Spalte 2 bis 3

df.iloc[11:22, 1:3]
```

[19]:

	Pulse	Maxpulse
11	100	120
12	106	128
13	104	132
14	98	123
15	98	120
16	100	120
17	90	112
18	103	123
19	97	125
20	108	131
21	100	119

[21]:
```
# ein Element vom Dataframe

df.iloc[12, 2]
```

[21]: 128

[22]:
```
# Informationen vom Dataframe darstellen lassen

print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 169 entries, 0 to 168
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Duration  169 non-null    int64
 1   Pulse     169 non-null    int64
```

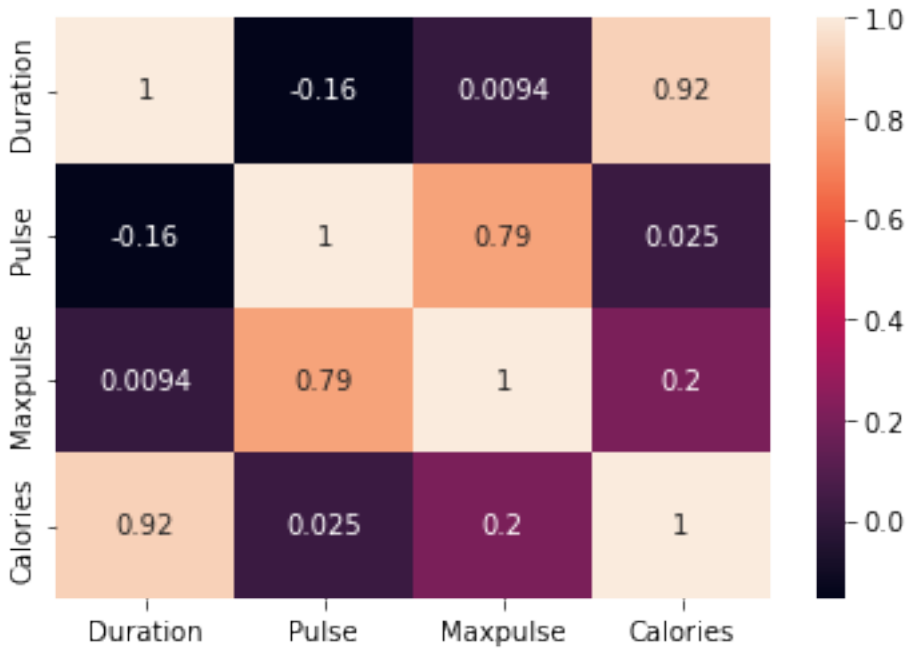
```
 2   Maxpulse   169 non-null    int64
 3   Calories   164 non-null    float64
dtypes: float64(1), int64(3)
memory usage: 6.6 KB
None
```

[23]:
```
df.corr() # Korrelationsmatrix
```

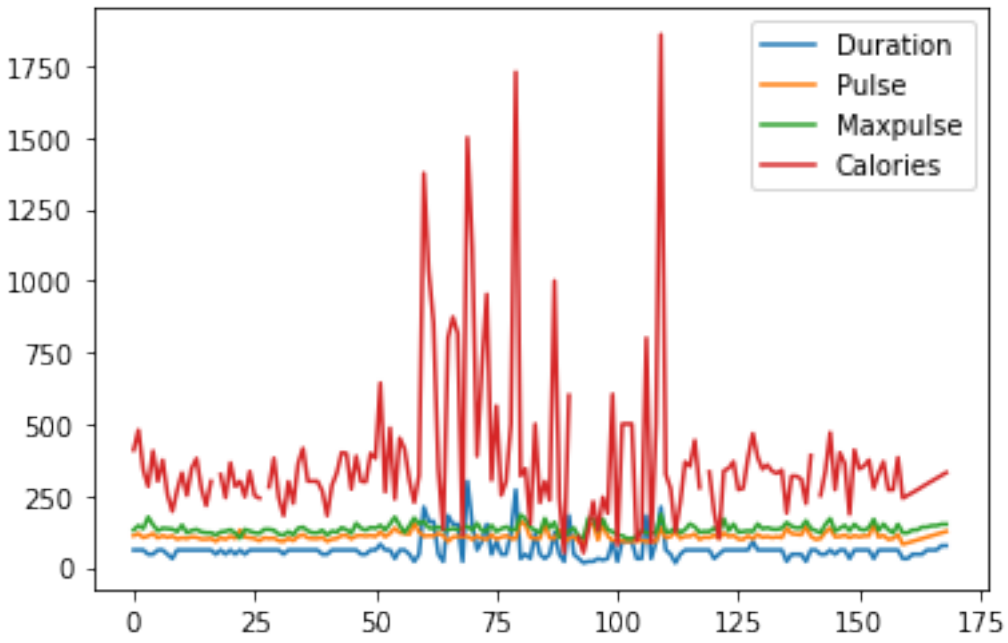
[23]:

	Duration	Pulse	Maxpulse	Calories
Duration	1.000000	-0.155408	0.009403	0.922721
Pulse	-0.155408	1.000000	0.786535	0.025120
Maxpulse	0.009403	0.786535	1.000000	0.203814
Calories	0.922721	0.025120	0.203814	1.000000

[26]:
```
sns.heatmap(df.corr(), annot=True) # korrelationsmatrix graphisch als "heatmap"
plt.show()
```



[27]:
```
df.plot() #dataframe plot in Linien
plt.show()
```



[ ]:

[28]:
```
# netwerken
```

[30]:
```
!pip install networkx
```

```
Requirement already satisfied: networkx in
/Users/h4/anaconda3/lib/python3.9/site-packages (2.7.1)
```

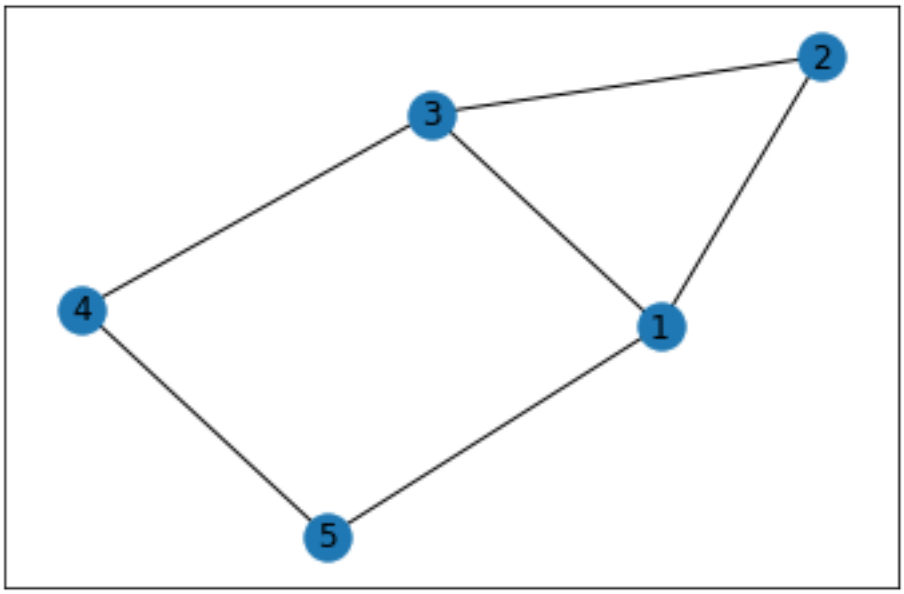
[31]:
```
import networkx
```

[32]:
```
import networkx as nx
import matplotlib.pyplot as plt

G = nx.Graph()

G.add_edge(1,2)
G.add_edge(1,3)
G.add_edge(1,5)
G.add_edge(2,3)
G.add_edge(3,4)
G.add_edge(4,5)

nx.draw_networkx(G)
plt.show()
```



[35]:
```
list(G.nodes)
```

[35]: [1, 2, 3, 5, 4]

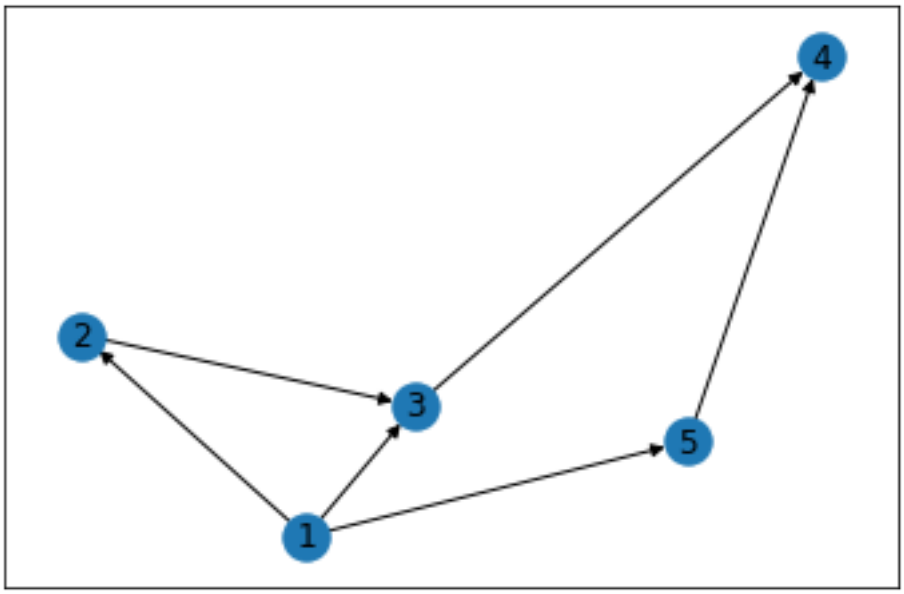
[36]:
```
list(G.edges)
```

[36]: [(1, 2), (1, 3), (1, 5), (2, 3), (3, 4), (5, 4)]

[37]:
```
G = nx.DiGraph([(1, 2), (1, 3), (1, 5), (2, 3), (3, 4), (5, 4)])

nx.draw_networkx(G)

plt.show()
```


4
2
3
5
1

[ ]: