PythonUebung2

October 14, 2021

1 Warum Pandas?

- Übung basierend auf Material von Prof. Dietzsch
- Pandas ist eine der leistungsfähigsten Bibliotheken, die man für Data Science in Python braucht
- Pandas bietet eine schnelle, flexible, leichte und intuitive Art der Manipulation von strukturierten/tabellenförmigen Daten (Data-Frames)
- Es ermöglicht die Arbeit mit großen Datensätzen und/oder mehreren Dateien zugleich.

1.1 Was zeichnet Pandas aus?

- `pandas` baut auf `numpy` auf und nutzt dieses als Backend
- Stellt eine High-Level Daten-Struktur zur Verfügung, die Spreadsheets von Tabellenkalkulationen ähnelt
- Verfügt über eingebaute Funktionen, um Daten zu bereinigen, zu filtern, zu gruppieren, zu kombinieren … etc.
- Es bietet die einfache Möglichkeit `numpy`- und `scipy`-Funktionen auf Datentabellen anzuwenden

1.1.1 Wie ist Pandas zu installieren?

Über den Anaconda-Propmpt und Eingabe von `conda install pandas`

oder über den `Anaconda Navigator` und der Auswahl von `pandas` installieren

oder öffnene einer Kommandozeile und Eingabe von `pip install pandas`

[ ]:
```
# Importieren von pandas
import pandas as pd
import numpy as np
import seaborn as sns
```

2 Matplotlib (Standard)

https://matplotlib.org/api/pyplot_summary.html https://matplotlib.org/examples/ https://matplotlib.org/gallery.html

Im weiteren Verlauf werden wir noch einige verschiede Grafik-Typen sehen bzw. benutzen.

[ ]:
```
# Einlesen der Daten aus einer CSV-Datei in eine Variable
pokemon_data = pd.read_csv('pokemon_data.csv')

#Überprüfen Sie Anfang und Ende
pokemon_data
```

[ ]:

	#	Name	Type 1	Type 2	HP	Attack	Defense	\
0	1	Bulbasaur	Grass	Poison	45	49	49	
1	2	Ivysaur	Grass	Poison	60	62	63	
2	3	Venusaur	Grass	Poison	80	82	83	
3	3	VenusaurMega Venusaur	Grass	Poison	80	100	123	
4	4	Charmander	Fire	NaN	39	52	43	
..	…	…	…	…	..	…	…	
795	719	Diancie	Rock	Fairy	50	100	150	
796	719	DiancieMega Diancie	Rock	Fairy	50	160	110	
797	720	HoopaHoopa Confined	Psychic	Ghost	80	110	60	
798	720	HoopaHoopa Unbound	Psychic	Dark	80	160	60	
799	721	Volcanion	Fire	Water	80	110	120	

	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	65	65	45	1	False
1	80	80	60	1	False
2	100	100	80	1	False
3	122	120	80	1	False
4	60	50	65	1	False
..	…	…	…	…	…
795	100	150	50	6	True
796	160	110	110	6	True
797	150	130	70	6	True
798	170	130	80	6	True
799	130	90	70	6	True

[800 rows x 12 columns]

`numpy.loadtxt` lädt Daten aus einem File in ein Numpy-Array. Standardmäßig wird von `float`-Werten ausgegangen. Dokumentation: https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.loadtxt.html

Anmerkung: Es gibt auch ein `numpy.savetxt(filename, array, ...)` zum Abspeichern Dokumentation: https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.savetxt.html

[ ]:
```
pokemon_data.columns              # Header der Daten
```

[ ]:
```
Index(['#', 'Name', 'Type 1', 'Type 2', 'HP', 'Attack', 'Defense', 'Sp. Atk',
       'Sp. Def', 'Speed', 'Generation', 'Legendary'],
      dtype='object')
```

[ ]:
```
pokemon_data.shape  # Größe der Datentabelle
```

[ ]: (800, 12)

[ ]:
```
pokemon_data.iloc[60] # Überprüfen des 60-sten Pokemons auf der Liste
```

[ ]:
```
#                 55
Name          Golduck
Type 1          Water
Type 2            NaN
HP                 80
Attack             82
Defense            78
Sp. Atk            95
Sp. Def            80
Speed              85
Generation          1
Legendary       False
Name: 60, dtype: object
```

[ ]:
```
pokemon_data[['Name','Attack','Speed']] # Untertabelle anhand mehrerer␣
↪selektierter Spalten (Slice-by-Columns)
```

[ ]:
```
                     Name  Attack  Speed
0               Bulbasaur      49     45
1                 Ivysaur      62     60
2                Venusaur      82     80
3    VenusaurMega Venusaur     100     80
4              Charmander      52     65
..                    …     …    …
795               Diancie     100     50
796    DiancieMega Diancie     160    110
797    HoopaHoopa Confined     110     70
798     HoopaHoopa Unbound     160     80
799              Volcanion     110     70

[800 rows x 3 columns]
```

[ ]:
```
pokemon_data.iloc[300:400] # Slice-by-Rows
```

[ ]:
```
       #             Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  \
300  277           Swellow   Normal  Flying  60      85       60       50
301  278           Wingull    Water  Flying  40      30       30       55
302  279          Pelipper    Water  Flying  60      50      100       85
303  280             Ralts  Psychic   Fairy  28      25       25       45
304  281            Kirlia  Psychic   Fairy  38      35       35       65
..   …                 …      …     …  ..     …       …       …
395  361           Snorunt      Ice     NaN  50      50       50       50
396  362            Glalie      Ice     NaN  80      80       80       80
```

```
397  362  GlalieMega Glalie      Ice     NaN  80     120       80       120
398  363             Spheal      Ice   Water  70      40       50        55
399  364             Sealeo      Ice   Water  90      60       70        75

     Sp. Def  Speed  Generation  Legendary
300       50    125           3      False
301       30     85           3      False
302       70     65           3      False
303       35     40           3      False
304       55     50           3      False
..       …    …          …         …
395       50     50           3      False
396       80     80           3      False
397       80    100           3      False
398       50     25           3      False
399       70     45           3      False

[100 rows x 12 columns]
```

Slicing: * Linke Seite → Zeilen * Rechte Seite → Spalten

[ ]:
```
pokemon_data.iloc[:,4:8] # alle Zeilen, Spalten 5 bis 9
```

[ ]:
```
     HP  Attack  Defense  Sp. Atk
0    45      49       49       65
1    60      62       63       80
2    80      82       83      100
3    80     100      123      122
4    39      52       43       60
..   ..      …       …       …
795  50     100      150      100
796  50     160      110      160
797  80     110       60      150
798  80     160       60      170
799  80     110      120      130

[800 rows x 4 columns]
```

[ ]:
```
pokemon_data.iloc[9:19,4:8] # Zeilen 10:20, Spalten 5 bis 9
```

[ ]:
```
    HP  Attack  Defense  Sp. Atk
9   44      48       65       50
10  59      63       80       65
11  79      83      100       85
12  79     103      120      135
13  45      30       35       20
14  50      20       55       25
```

```
15  60      45       50       90
16  40      35       30       20
17  45      25       50       25
18  65      90       40       45
```

[ ]:
```
pokemon_data.sort_values('Attack') # Sortieren nach Werten
```

[ ]:
```
       #                    Name   Type 1    Type 2   HP  Attack  Defense  \
488  440                 Happiny   Normal       NaN  100       5        5
121  113                 Chansey   Normal       NaN  250       5        5
230  213                 Shuckle      Bug      Rock   20      10      230
261  242                 Blissey   Normal       NaN  255      10       10
139  129                Magikarp    Water       NaN   20      10       55
..   …                       …       …       …  …     …       …
429  386       DeoxysAttack Forme  Psychic       NaN   50     180       20
426  384    RayquazaMega Rayquaza   Dragon    Flying  105     180      100
424  383    GroudonPrimal Groudon   Ground      Fire  100     180      160
232  214  HeracrossMega Heracross      Bug  Fighting   80     185      115
163  150      MewtwoMega Mewtwo X  Psychic  Fighting  106     190      100

     Sp. Atk  Sp. Def  Speed  Generation  Legendary
488       15       65     30           4      False
121       35      105     50           1      False
230       10      230      5           2      False
261       75      135     55           2      False
139       15       20     80           1      False
..       …      …    …          …        …
429      180       20    150           3       True
426      180      100    115           3       True
424      150       90     90           3       True
232       40      105     75           2      False
163      154      100    130           1       True

[800 rows x 12 columns]
```

[ ]:
```
pokemon_data.describe()
```

[ ]:
```
                #          HP      Attack     Defense     Sp. Atk     Sp. Def  \
count  800.000000  800.000000  800.000000  800.000000  800.000000  800.000000
mean   362.813750   69.258750   79.001250   73.842500   72.820000   71.902500
std    208.343798   25.534669   32.457366   31.183501   32.722294   27.828916
min      1.000000    1.000000    5.000000    5.000000   10.000000   20.000000
25%    184.750000   50.000000   55.000000   50.000000   49.750000   50.000000
50%    364.500000   65.000000   75.000000   70.000000   65.000000   70.000000
75%    539.250000   80.000000  100.000000   90.000000   95.000000   90.000000
max    721.000000  255.000000  190.000000  230.000000  194.000000  230.000000
```

```
              Speed  Generation
count  800.000000   800.00000
mean    68.277500     3.32375
std     29.060474     1.66129
min      5.000000     1.00000
25%     45.000000     2.00000
50%     65.000000     3.00000
75%     90.000000     5.00000
max    180.000000     6.00000
```

[ ]:
```
#pokemon_data.dropna(axis = 0) # alle Zeilen mit NaN's löschen
#pokemon_data.dropna(axis = 1) # alle Spalten mit NaN's löschen
```

[ ]:
```
pokemon_data['Type 2'].value_counts() # "unique Werte" einer Spalte
```

[ ]:
```
Flying      97
Ground      35
Poison      34
Psychic     33
Fighting    26
Grass       25
Fairy       23
Steel       22
Dark        20
Dragon      18
Water       14
Ghost       14
Ice         14
Rock        14
Fire        12
Electric     6
Normal       4
Bug          3
Name: Type 2, dtype: int64
```

[ ]:
```
pokemon_data['Type 2'].value_counts(normalize = True) # Anteil der "unique␣
↪Werte"
```

[ ]:
```
Flying      0.234300
Ground      0.084541
Poison      0.082126
Psychic     0.079710
Fighting    0.062802
Grass       0.060386
Fairy       0.055556
Steel       0.053140
Dark        0.048309
```

```
Dragon      0.043478
Water       0.033816
Ghost       0.033816
Ice         0.033816
Rock        0.033816
Fire        0.028986
Electric    0.014493
Normal      0.009662
Bug         0.007246
Name: Type 2, dtype: float64
```

[ ]:
```
pokemon_data['Attack'] + pokemon_data['Defense']
```

[ ]:
```
0       98
1      125
2      165
3      223
4       95
      ...
795    250
796    270
797    170
798    220
799    230
Length: 800, dtype: int64
```

[ ]:
```
overall = pokemon_data['HP'] + pokemon_data['Attack'] + pokemon_data['Defense']␣
 ↪+ pokemon_data['Speed']
pokemon_data['Overall'] = overall
pokemon_data.head(10)
```

[ ]:

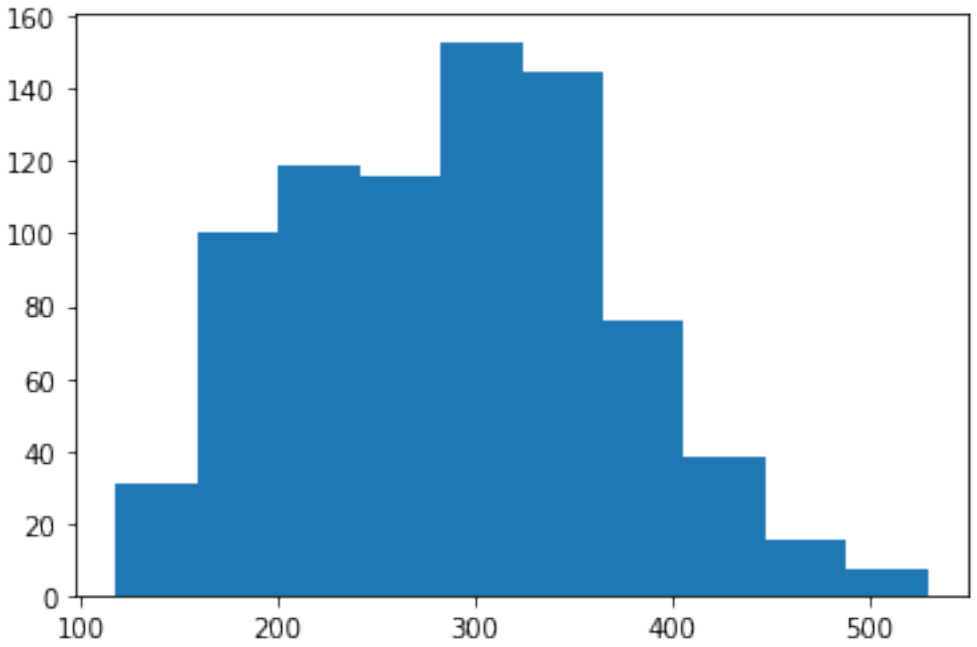
	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary	Overall
0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	False	188
1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	False	245
2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	False	325
3	3	VenusaurMega Venusaur	Grass	Poison	80	100	123	122	120	80	1	False	425
4	4	Charmander	Fire	NaN	39	52	43	60	50	65	1	False	309
5	5	Charmeleon	Fire	NaN	58	64	58	80	65	80	1	False	405
6	6	Charizard	Fire	Flying	78	84	78	109	85	100	1	False	534
7	6	CharizardMega Charizard X	Fire	Dragon	78	130	111	130	85	100	1	False	634
8	6	CharizardMega Charizard Y	Fire	Flying	78	104	78	159	115	100	1	False	634
9	7	Squirtle	Water	NaN	44	48	65	50	64	43	1	False	314

3	120	80	1	False	383
4	50	65	1	False	199
5	65	80	1	False	260
6	85	100	1	False	340
7	85	100	1	False	419
8	115	100	1	False	360
9	64	43	1	False	200

[ ]:
```
pokemon_data.to_csv('my_new_pokemon_table.txt', sep = '\t', index = False)
```

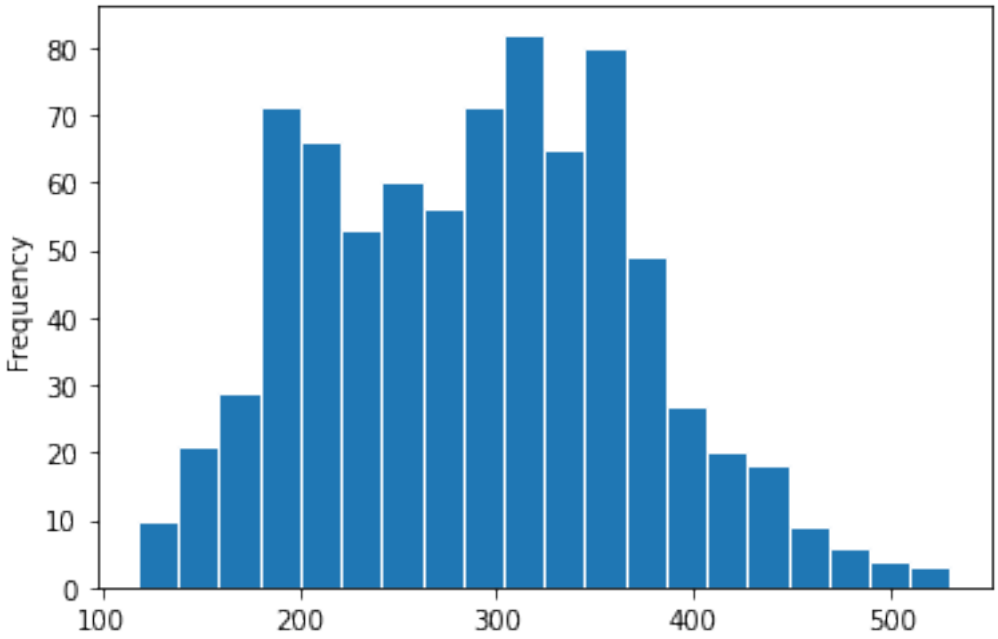
[ ]:
```
import matplotlib.pyplot as plt

plt.hist(pokemon_data['Overall']);
```



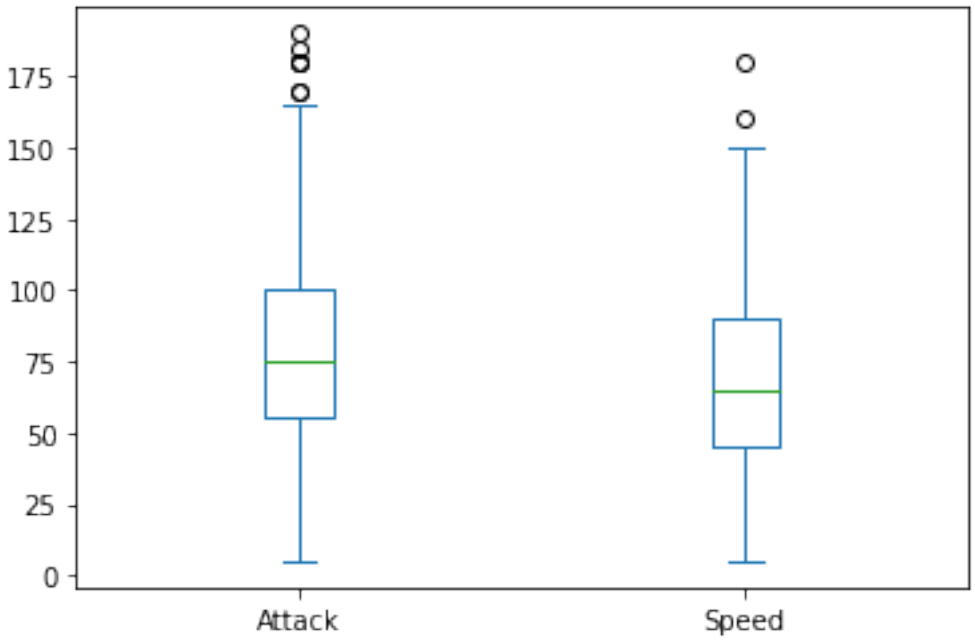
[ ]:
```
pokemon_data['Overall'].plot(kind = 'hist', bins = 20, edgecolor = 'white')
```

[ ]:
```
<AxesSubplot:ylabel='Frequency'>
```




```
[ ]: pokemon_data[['Attack','Speed']].plot(kind = 'box')
```

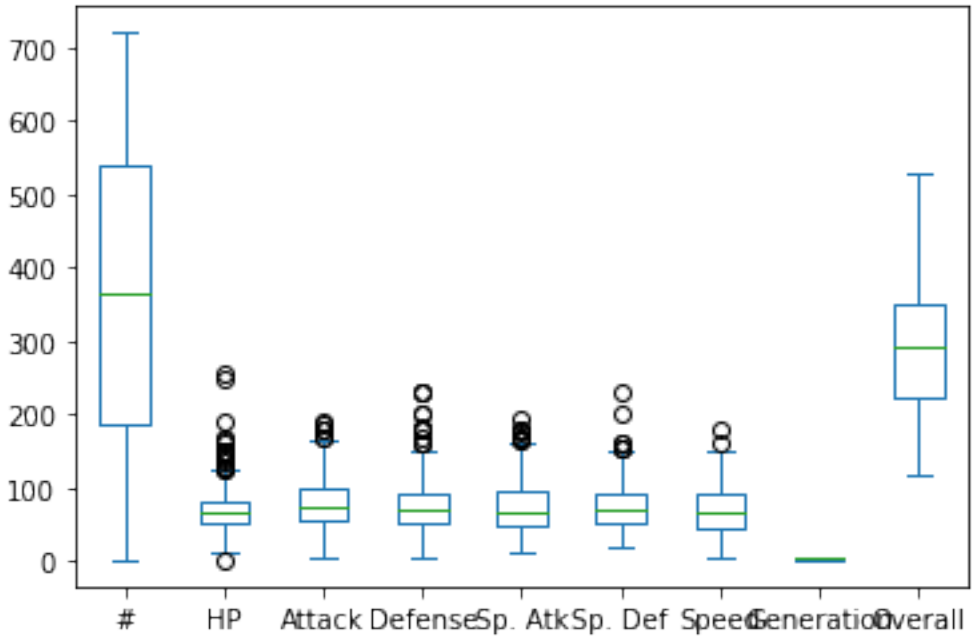
```
[ ]: <AxesSubplot:>
```



Aufgabe 1: Untersuchen Sie alle Stats mit Boxplots!

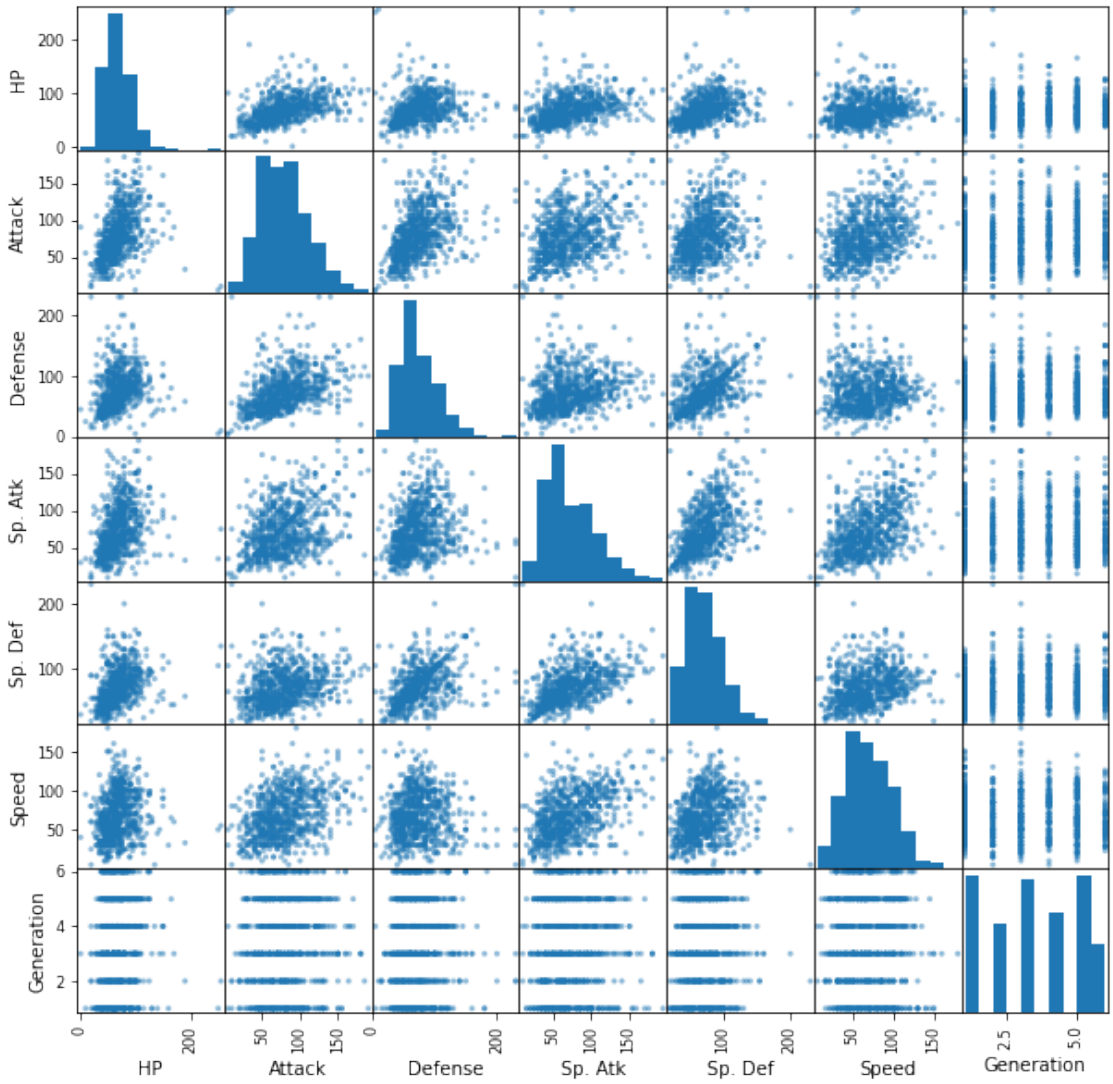
[ ]:
```
pokemon_data.plot(kind="box")
```

[ ]: <AxesSubplot:>



Aufgabe 2: Erstellen Sie eine Scatterplotmatrix!

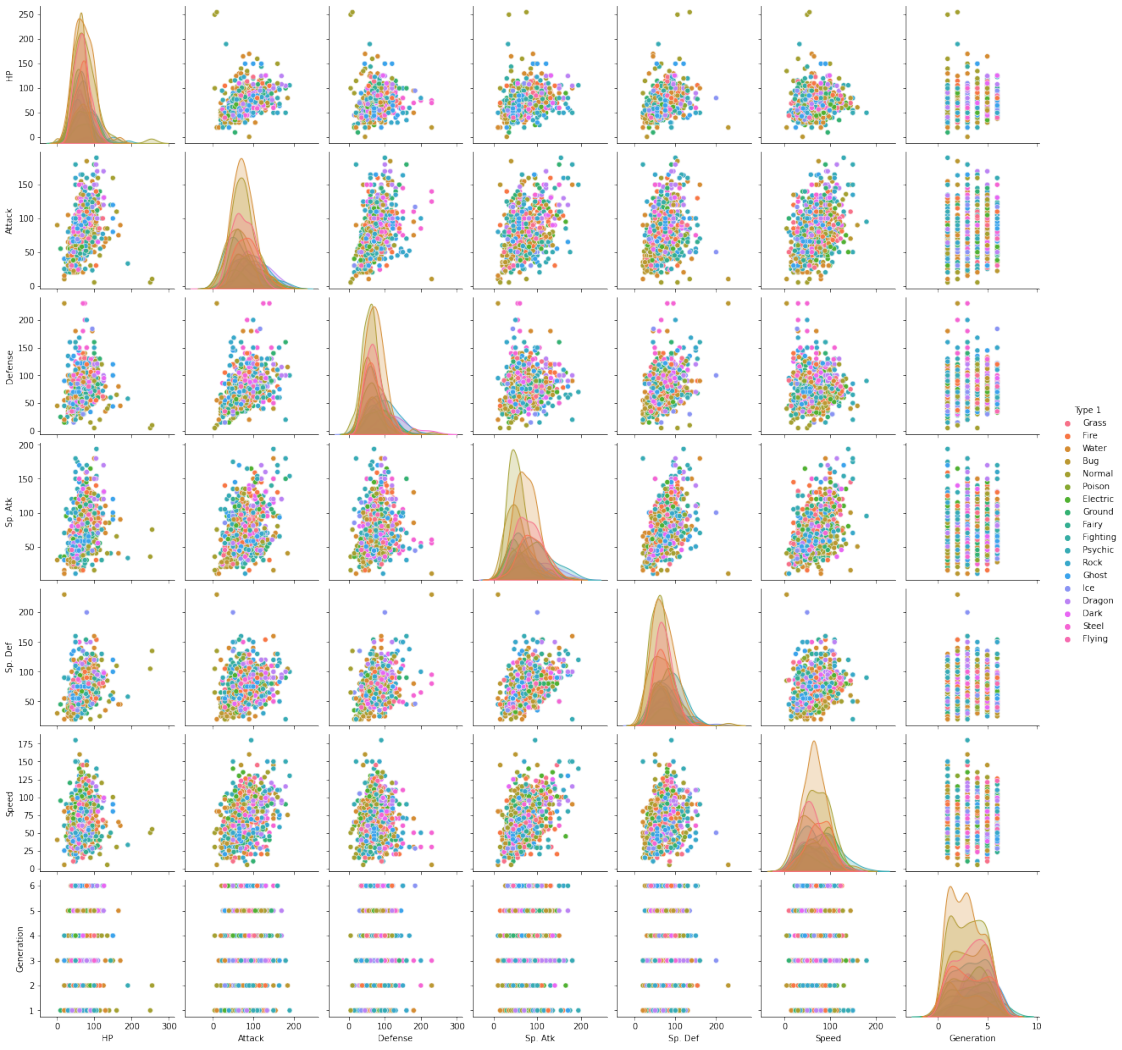
[ ]:
```
_ = pd.plotting.scatter_matrix(pokemon_data.iloc[:, 4:11], figsize=(10, 10))
```



Aufgabe 3: Erweitern Sie die Scatterplotmatrix um den “Type 1”-Typ (einfärben)!

[ ]:
```
sns.pairplot(pokemon_data.iloc[:, 2:11], hue="Type 1")
```

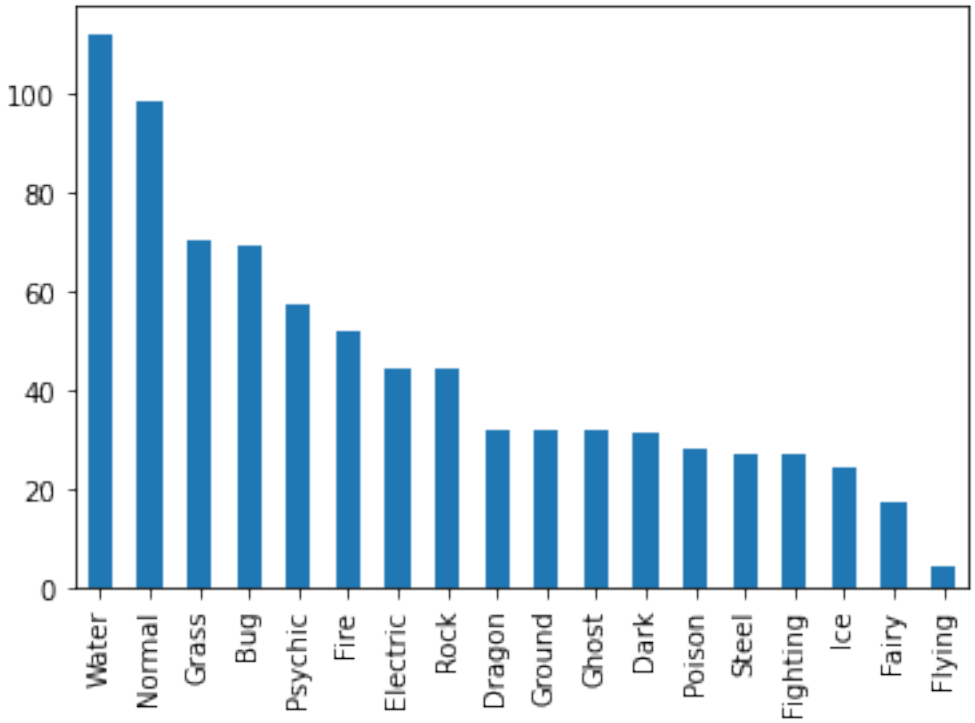
[ ]:
```
<seaborn.axisgrid.PairGrid at 0x7f4237c0ea90>
```



Aufgabe 4: Stellen Sie den “Type 1”-Typ dar vs. Anzahl der Pokemons hierfür.

[ ]:
```
pokemon_data['Type 1'].value_counts().plot(kind="bar")
```

[ ]: <AxesSubplot:>

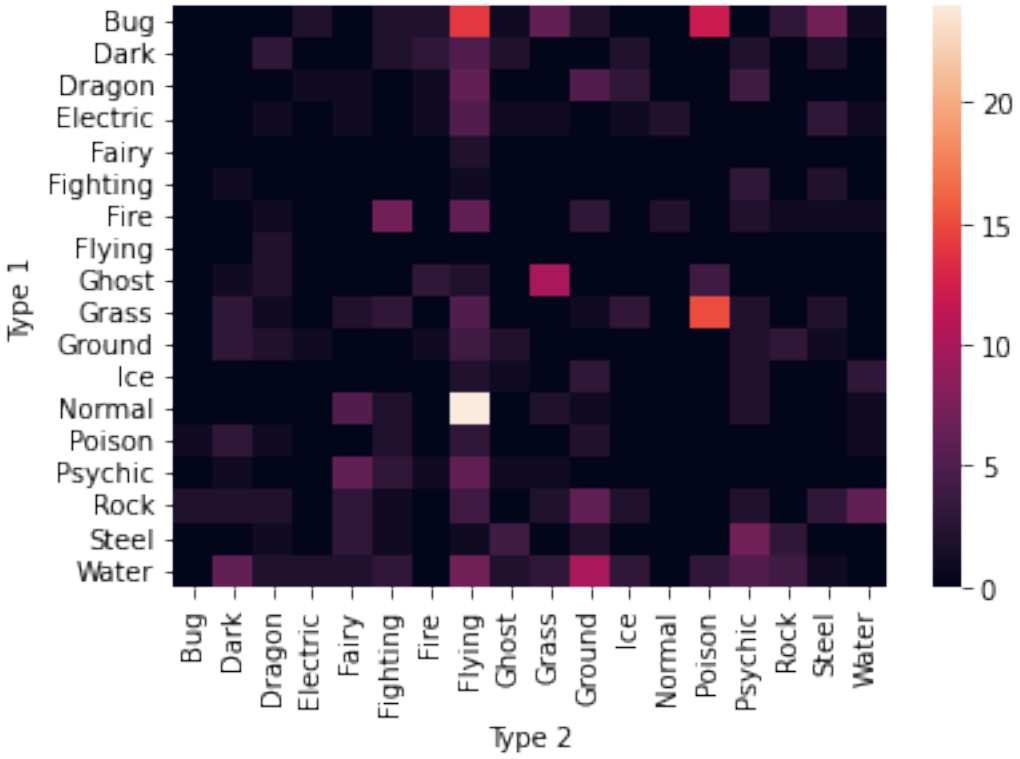


Aufgabe 5: Untersuchen Sie, wie sich Type1 zu Type2 verhält!

[ ]:
```
sns.heatmap(pd.crosstab(pokemon_data.get("Type 1"), pokemon_data.get("Type 2")))

# with numbers
# sns.heatmap(pokemon_data.groupby(['Type 1', 'Type 2']).size().unstack(),␣
 ↪linewidths=1, annot=True)
```

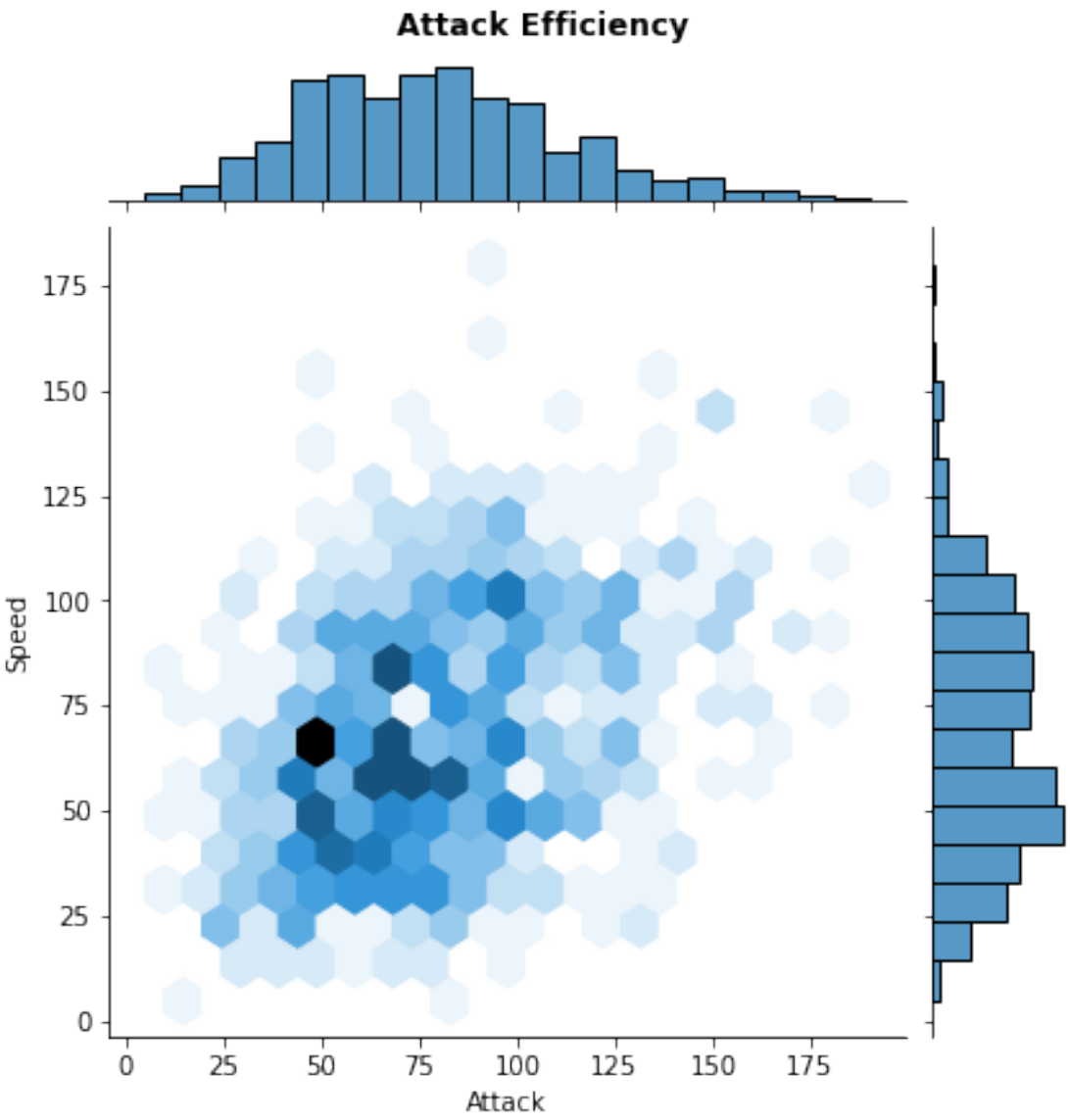
[ ]:
```
<AxesSubplot:xlabel='Type 2', ylabel='Type 1'>
```



Aufgabe 6: Erzeugen Sie eine Grafik für Attack Efficiency, welche Angriffsgeschwindigkeit und Angriffsstärke sinnvoll visualisiert.

[29]:
```
rel = sns.jointplot(x=pokemon_data.get("Attack"), y=pokemon_data.get("Speed"),␣
↪kind="hex")
rel.fig.suptitle("Attack Efficiency", fontweight="bold", y=1.02)
```

[29]:
```
Text(0.5, 1.02, 'Attack Efficiency')
```



Aufgabe 7: Stellen Sie eine Übersicht über defensive und angriffsfreudige Pokemons dar.

[30]:
```
# calculate overall attack and defense stats
pokemon_data["AttackOverall"] = pokemon_data["Attack"] + pokemon_data["Sp.␣
↪Atk"] + pokemon_data["Speed"]
pokemon_data["DefenseOverall"] = pokemon_data["Defense"] + pokemon_data["Sp.␣
↪Def"]
```

[31]:
```
# Best attack pokemon
pokemon_data.sort_values("AttackOverall").tail()
```

[31]:

	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary	Overall	AttackOverall	DefenseOverall
428	386	DeoxysNormal Forme	Psychic	NaN	50	150	50	150	50	150	3	True	400	450	100
163	150	MewtwoMega Mewtwo X	Psychic	Fighting	106	190	100	154	100	130	1	True	526	474	200
426	384	RayquazaMega Rayquaza	Dragon	Flying	105	180	100	180	100	115	3	True	500	475	200
164	150	MewtwoMega Mewtwo Y	Psychic	NaN	106	150	70	194	120	140	1	True	466	484	190
429	386	DeoxysAttack Forme	Psychic	NaN	50	180	20	180	20	150	3	True	400	510	40

[32]:
```
# best defense pokemon
pokemon_data.sort_values("DefenseOverall").tail()
```

[32]:

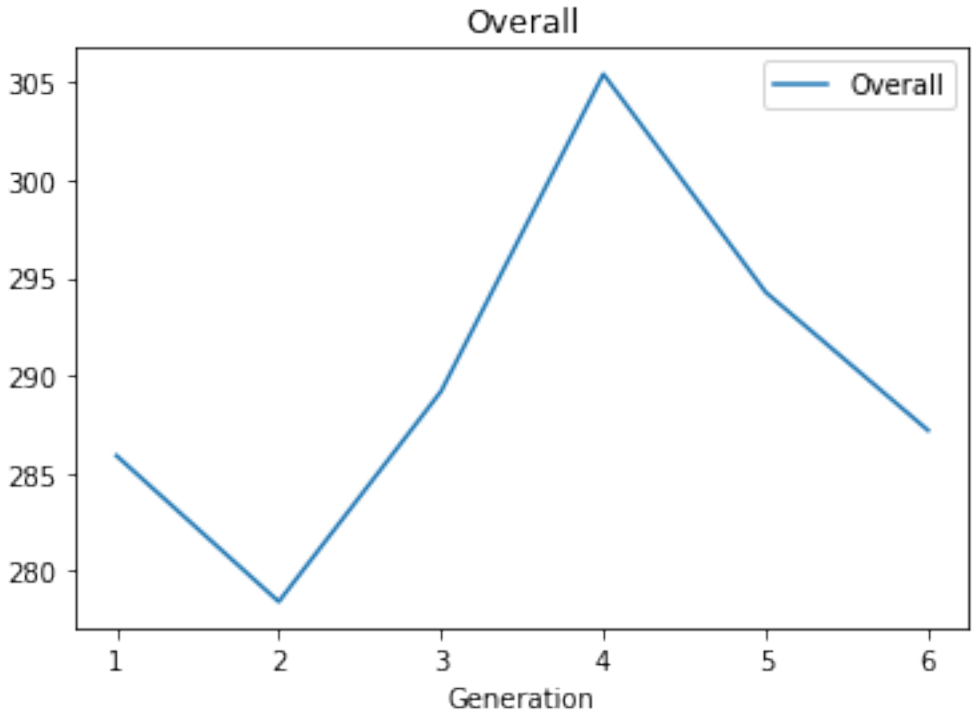
	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary	Overall	AttackOverall	DefenseOverall
456	411	Bastiodon	Rock	Steel	60	52	168	47	138	30	4	False	310	129	306
333	306	AggronMega Aggron	Steel	NaN	70	140	230	60	80	50	3	False	490	250	310
430	386	DeoxysDefense Forme	Psychic	NaN	50	70	160	70	160	90	3	True	370	230	320
224	208	SteelixMega Steelix	Steel	Ground	75	125	230	55	95	30	2	False	460	210	325
230	213	Shuckle	Bug	Rock	20	10	230	10	230	5	2	False	265	25	460

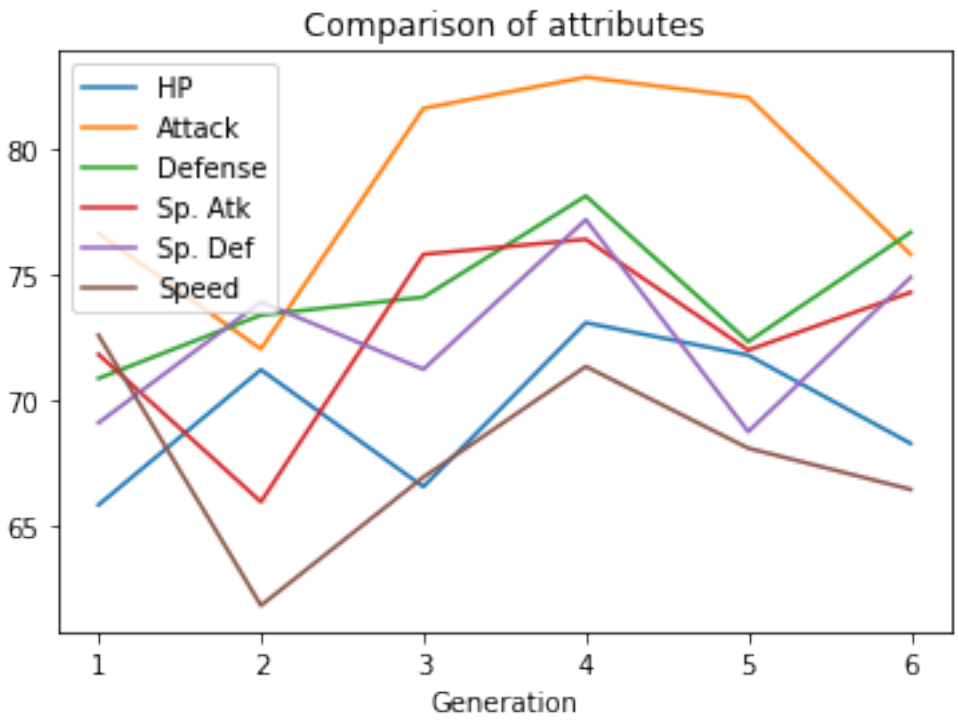
Aufgabe 8: Untersuchen Sie, wie sich die Pokemons über die Generationen verändert haben in den relevanten Werten. Werden diese besser? Schlechter? Sind bestimmte Elemente “in”?

[33]:
```
# Overall
pokemon_data.groupby("Generation").mean()[["Overall"]].plot(title="Overall")

# Overview on different attributes
pokemon_data.groupby("Generation").mean()[["HP", "Attack", "Defense", "Sp.␣
↪Atk", "Sp. Def", "Speed"]].plot(title="Comparison of attributes")
```

[33]:
```
<AxesSubplot:title={'center':'Comparison of attributes'}, xlabel='Generation'>
```

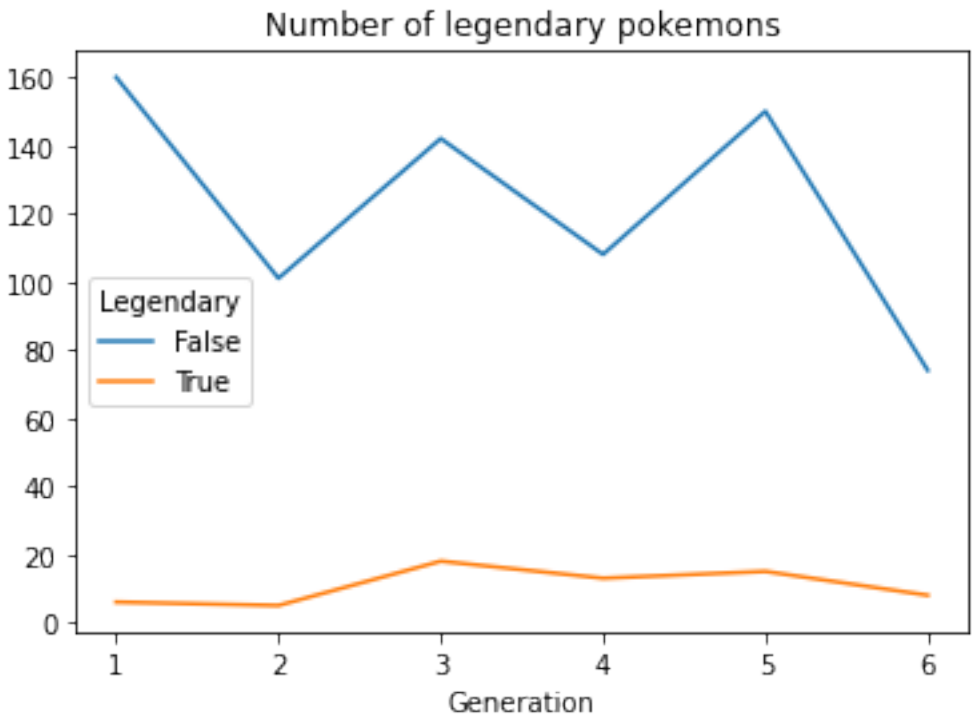




Aufgabe 9: Überlegen Sie sich mindestens 3 weitere sinnvolle Plots für diesen Datensatz.

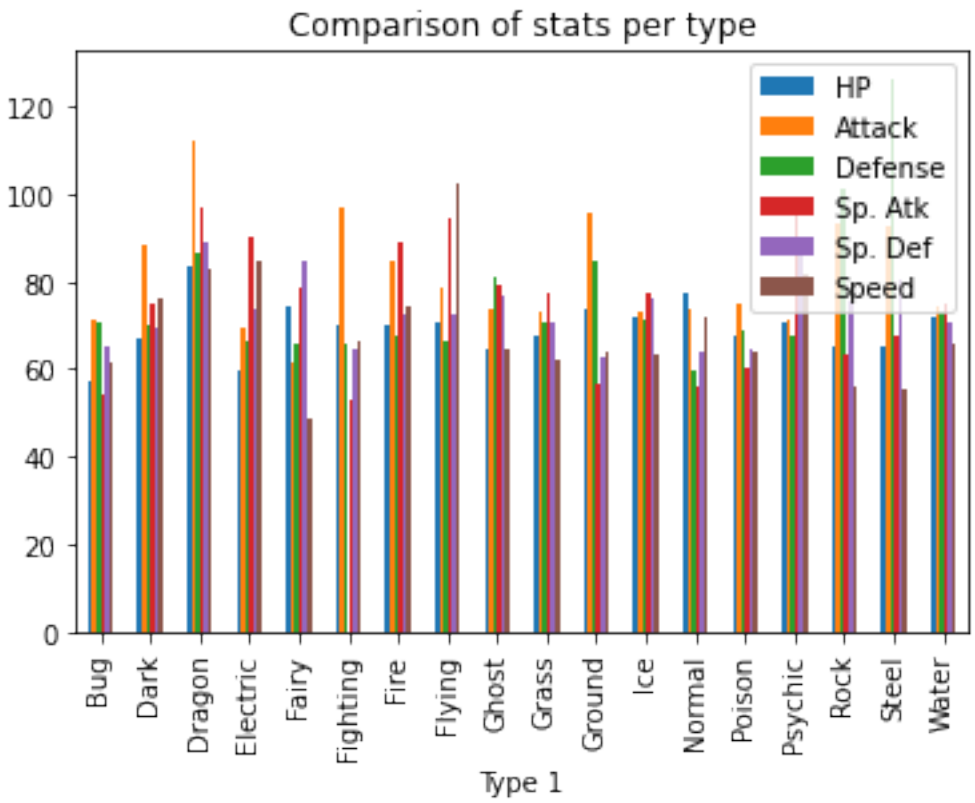
[34]:
```
# Legendary pokemons in different generations
pd.crosstab(pokemon_data.get("Generation"), pokemon_data.get("Legendary")).
 ↪plot().set(title="Number of legendary pokemons")
```

[34]:
```
[Text(0.5, 1.0, 'Number of legendary pokemons')]
```



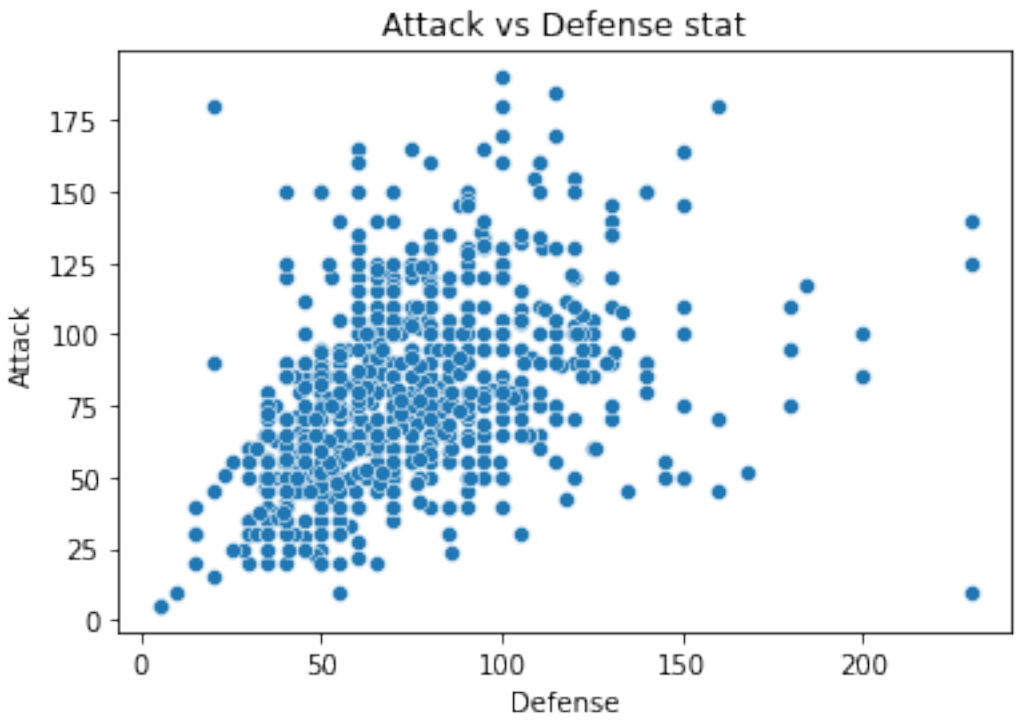
[35]:
```
pokemon_data.groupby("Type 1").mean()[["HP", "Attack", "Defense", "Sp. Atk",␣
↪"Sp. Def", "Speed"]].plot(kind="bar", title="Comparison of stats per type")
```

[35]: <AxesSubplot:title={'center':'Comparison of stats per type'}, xlabel='Type 1'>



[45]:
```
sns.scatterplot(data=pokemon_data[["Attack", "Defense"]], y="Attack",␣
↪x="Defense").set_title("Attack vs Defense stat")
```

[45]: Text(0.5, 1.0, 'Attack vs Defense stat')


Attack vs Defense stat
Attack
Defense
0
25
50
75
100
125
150
175
0
50
100
150
200

[ ]: