Noch mehr zu Farben

by Woche 12

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

Farbe ist ein wesentliches ästhetisches Element in der Datenvisualisierung und verdient eine detaillierte Betrachtung. Die richtige Farbwahl kann die Verständlichkeit und Ästhetik einer Grafik erheblich verbessern. In diesem Kapitel werden wir die Nuancen der Farbwahl in Python mit Seaborn und Matplotlib erkunden. Wir nutzen dazu wieder den Pinguin-Datensatz.

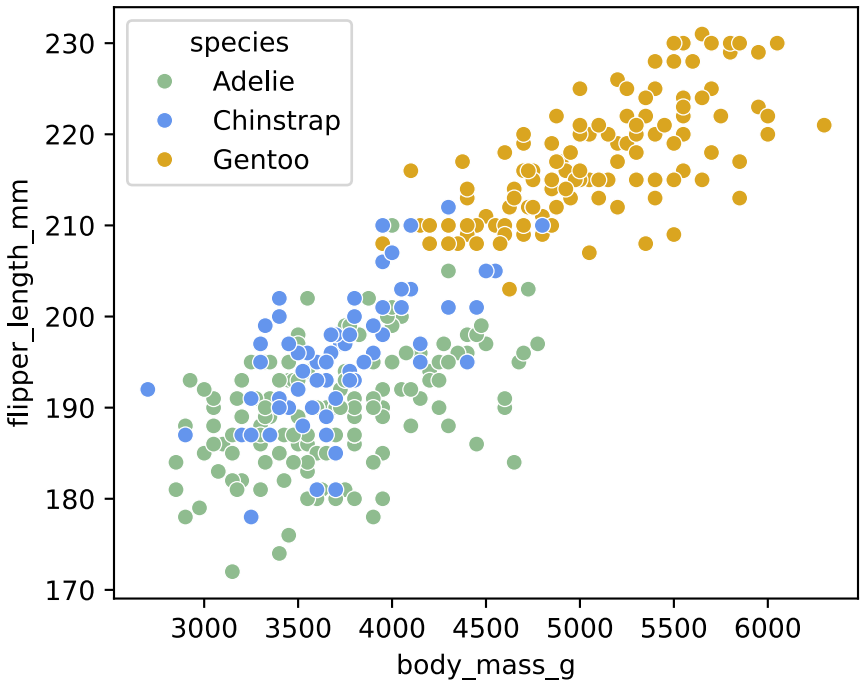
```
csv_url='https://raw.githubusercontent.com/SchmidtPaul/ExampleData/main/
palmer_penguins/palmer_penguins.csv'
df=pd.read_csv(csv_url)

# Konvertiere alle 'object'-Spalten in 'category'
for col in df.select_dtypes(include='object').columns:
    df[col] = df[col].astype('category')
```

Manuelle Farbauswahl

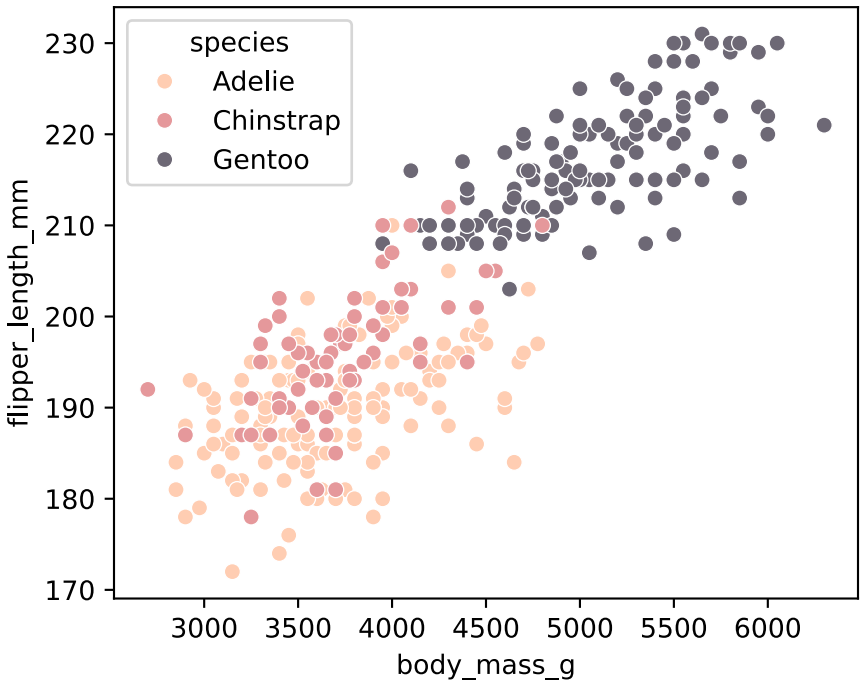
Wir haben bereits gelernt, dass es eine Reihe Farben gibt, die anhand ihres Namens verwendet werden können. Die volle Flexibilität erhalten wir jedoch, wenn wir HEX-Codes verwenden. Diese Codes sind eine Kombination von Buchstaben und Zahlen, die eine Farbe eindeutig identifizieren. Zum Beispiel ist der HEX-Code für das oft auf dieser Webseite verwendete grün `#00923f`. Wir können also genau so gut einfach HEX-Codes als strings übergeben.

```
palette1 = {
    'Adelie': 'darkseagreen',
    'Chinstrap': 'cornflowerblue',
    'Gentoo': 'goldenrod'
}

plt.figure(figsize=(5, 4))
sns.scatterplot(
  data=df,
  x='body_mass_g',
```

```
  y='flipper_length_mm',
  hue='species',
  palette=palette1
)
plt.show()
```




```
palette1 = {
    'Adelie': '#ffcdb2',
    'Chinstrap': '#e5989b',
    'Gentoo': '#6d6875'
}

plt.figure(figsize=(5, 4))
sns.scatterplot(
  data=df,
  x='body_mass_g',
  y='flipper_length_mm',
  hue='species',
  palette=palette1
)
plt.show()
```



> **RGB-Werte**
>
> Eine weitere Möglichkeit Farben zu definieren sind die RGB-Werte. Diese bestehen aus drei Zahlen, die den Anteil von Rot, Grün und Blau in der Farbe angeben. So wäre die Farbe `#00923f` auch als `rgb(0, 146, 63)` definiert. Python versteht beide und tatsächlich erzeugen die folgenden Funktionen die Farben im RGB-Format. Ich bevorzuge jedoch die HEX-Codes, da sie kürzer sind.

Da man so also die volle Freiheit hat, braucht man ggf. Hilfe bzw. Inspiration um Farben zu finden, die gut zueinander passen. Hierfür gibt es verschiedene Tools, wie z.B. Coolors. Man bekommt also Vorschläge zu beliebten Farbpaletten und kann speziell mit dem Coolors Palette Generator auch recht komfortabel eigene Farbpaletten ausprobieren und schließlich speichern. Außerdem kann auch mit dem Coolors Image Picker eine Farbpalette aus einem Bild extrahiert werden.

Vorhandene Farbpaletten

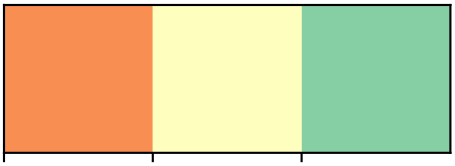
Seaborn bietet aber auch eine Reihe von vorgefertigten Farbpaletten, die wir verwenden können. Diese z.B. hier aufgelistet. So können wir mit `sns.color_palette()` eine Farbpalette auswählen und angeben wie viele Farben wir aus dieser Palette benötigen. Außerdem können wir uns dann diese Farben mit `sns.palplot()` anzeigen lassen.

```
farben = sns.color_palette(palette='Spectral', n_colors=3)
```

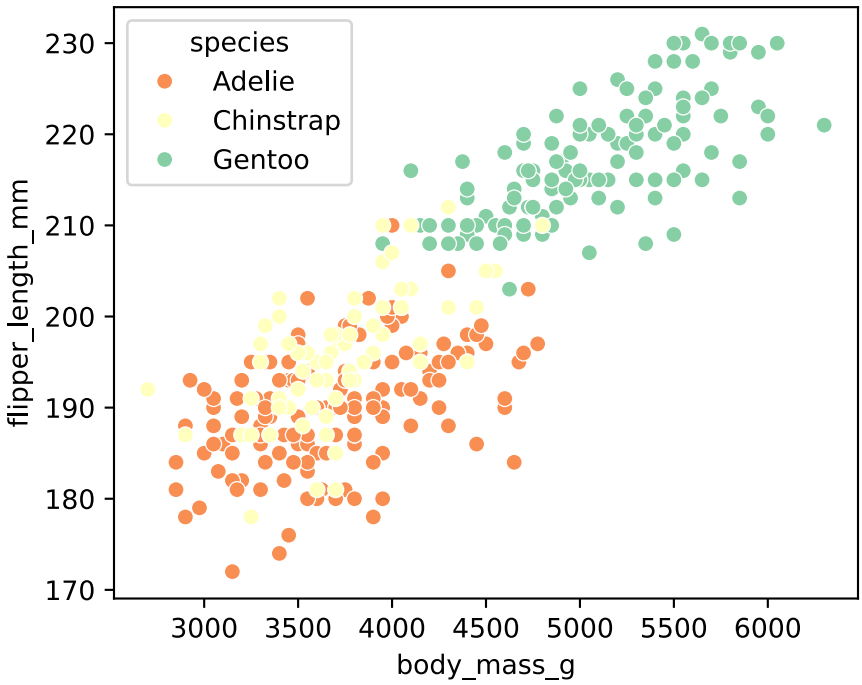
```
print(farben)
```

```
[(0.9748558246828143, 0.5574009996155325, 0.32272202998846594),
(0.998077662437524, 0.9992310649750096, 0.7460207612456747),
(0.5273356401384084, 0.8106113033448674, 0.6452133794694349)]
```

```
sns.palplot(farben)
```



```
plt.figure(figsize=(5, 4))
sns.scatterplot(
  data=df,
  x='body_mass_g',
  y='flipper_length_mm',
  hue='species',
  palette='Spectral' # oder: farben
)
plt.show()
```



Wer mit den vorgefertigten Paletten aus Seaborn nicht zufrieden ist, kann auf zusätzliche Bibliotheken wie palettable und colorcet zurückgreifen.

Geeignetere Farbpaletten

Abgesehen von persönlichem Geschmack gibt es allerdings auch zwei nennenswerte Aspekte, die ggf. dazu führen, dass bestimmten Farbpaletten besser geeignet sind als andere.

Grund 1: Corporate Design

Es kann sein, dass ein Plot letztendlich für ein Unternehmen bzw. eine Institution angefertigt wird. Dann ist es gut möglich, dass es ein Corporate Design gibt, das vorschreibt, welche Farben verwendet werden sollen. In diesem Fall ist es natürlich sinnvoll, sich an diese Vorgaben zu halten. Beispielsweise gibt es das Coporate Design des Umweltbundesamtes, welches genau 12 Farben erlaubt:

	Farbname	Euroskala	RGB	Pantone	Hex	RAL
1	UBA Grün	68/0/95/0	94/173/53	368	#5EAD35	6018
	UBA Dunkelgrün	85/0/100/35	0/118/38	356	#007626	6001
2	UBA Blau	80/20/0/0	0/155/213	2925	#0b90d5	5012
	UBA Dunkelblau	80/20/0/50	0/95/133	2955	#005f85	5001
3	UBA Flieder	45/75/0/0	157/87/154			
	UBA Dunkelflieder	50/80/0/45	98/47/99			
4	UBA Fuchsia	15/95/40/0	206/31/94			
	UBA Dunkelfuchsia	20/100/40/45	131/5/60			
5	UBA Ocker	0/30/100/0	250/187/0			
	UBA Dunkelocker	0/50/100/15	215/132/0			
6	UBA Hellgrau*	0/0/0/10	240/241/241			
	UBA Dunkelgrau	0/0/0/85	75/75/77			

Und tatsächlich wirken Visualisierungen schlichtweg professioneller und harmonischer, wenn sie sich an ein Farbschema halten, das nicht nur zwischen den verschiedenen Grafiken einheitlich ist, sondern auch beispielsweise zu den Farben der Webseite passt, auf der die Grafiken eingebettet sind, oder zu den Farben des Unternehmenslogos, welches im Kopf jeder Berichtseite prangt, auf dem die Grafiken abgedruckt sind.

Grund 2: Barrierefreiheit

Ein weiterer wichtiger Aspekt ist die Farbenblindheit. Etwa 8% der Männer und 0.5% der Frauen sind zumindest teilweise farbenblind. Die häufigste Form der Farbenblindheit ist die Rot-Grün-Schwäche. Das bedeutet, dass die Betroffenen Schwierigkeiten haben, Rot und Grün zu unterscheiden. Es ist also wichtig, dass Farben in Grafiken so gewählt werden, dass sie auch von farbenblinden Menschen gut unterschieden werden können. Früher wurde auch als Grund angeführt, dass Dokumente ggf. auch auf Schwarz-Weiß-Druckern ausgedruckt werden, sodass auch danach die Abbildung weiterhin interpretierbar sein sollte.

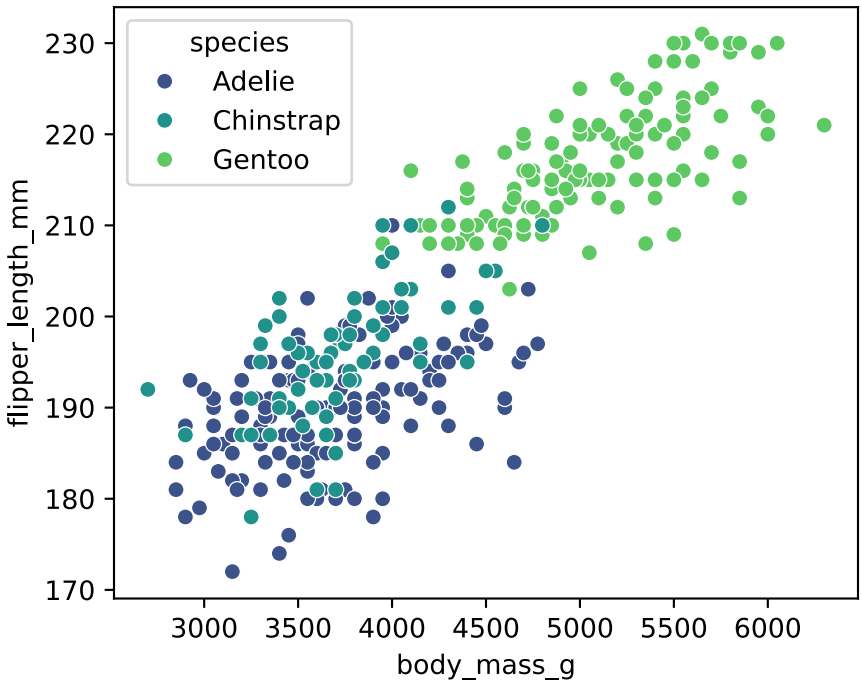
Als Lösung gibt es spezielle Farbpaletten, die so gewählt sind, dass sie auch von farbenblinden Menschen gut unterschieden werden können. Eine der populärsten Farbpaletten dieser Art heißt *Viridis* und ist auch über Seaborn verfügbar:

```
sns.palplot(sns.color_palette(palette="viridis", n_colors=25))
```

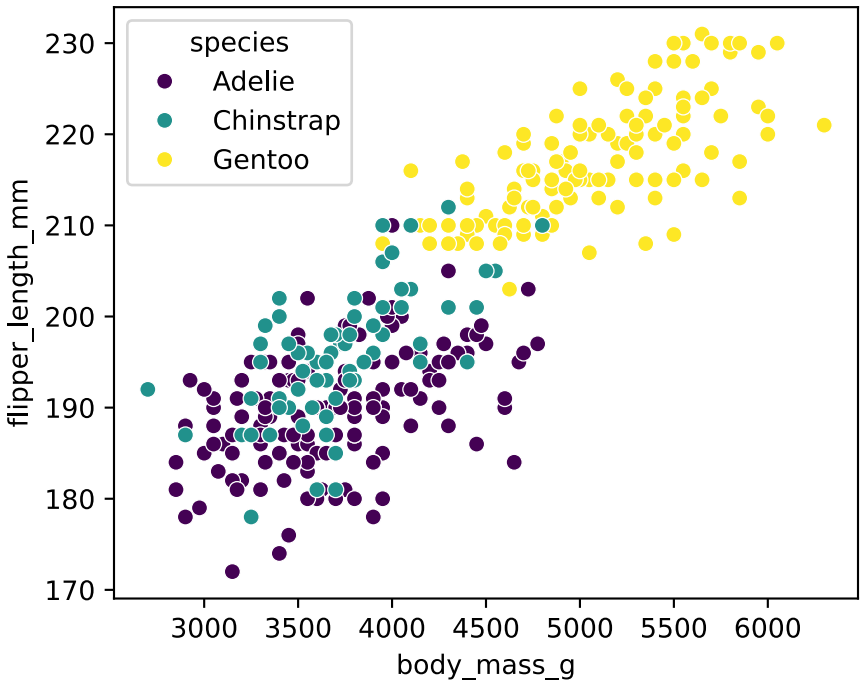
Um zu sehen wie gut die Farben von *Viridis* von Menschen mit verschiedenen Formen der Farbenblindheit wahrgenommen werden, kann die Aufbereitung auf dieser Website betrachtet werden. Im Prinzip wird dort gezeigt, dass das was für Menschen mit normaler Farbwahrnehmung wie ein Gradient von Gelb über Grün zu Blau zu Lila aussieht, selbst ohne Farbwahrnehmung noch als Gradient von hell zu dunkel wahrgenommen wird.

Bezüglich der Anwednung von *Viridis* in Seaborn gibt es sogar die sehr einfache Möglichkeit `palette='viridis'` zu schreiben. Allerdings ist diese Funktionalität meines Erachtens nicht optimal umgesetzt, da dann anscheinend (?!) nicht die gesamte Palette verwendet wird, sondern nur ein Teil davon. Demnach gebe ich hier eine zweite Möglichkeit an, wo mit extra Schritten dafür gesorgt wird, dass möglichst die Extreme der Farbpalette - also so unterschiedliche Farben wie möglich - verwendet werden.

```
#



plt.figure(figsize=(5, 4))
sns.scatterplot(
  data=df,
  x='body_mass_g',
  y='flipper_length_mm',
  hue='species',
  palette='viridis'
)
plt.show()
```




```
viridis = sns.color_palette("viridis", as_cmap=True)
n = 3
mein_viridis = [viridis(i / (n - 1)) for i in range(n)]

plt.figure(figsize=(5, 4))
sns.scatterplot(
  data=df,
  x='body_mass_g',
  y='flipper_length_mm',
  hue='species',
  palette=mein_viridis
)
plt.show()
```



Hier noch eine abschließende Bemerkung: Wir haben nur im allerersten Abschnitt die Farben als Dictionary übergeben. In allen anderen Fällen haben wir die Farben als Liste übergeben. Wie man sieht reicht es aus, Farben als Liste zu übergeben, allerdings ist dann unter Umständen nicht gewährleistet, dass die Farben auch wirklich den “richtigen” Kategorien zugeordnet werden. So kann es passieren, dass in einem Plot die Art “Adelie” plötzlich die Farbe von “Gentoo” hat. Mithilfe der Legende wären beide Plots natürlich weiterhin korrekt interpretierbar, aber intuitiver wäre es, wenn dieselbe Art über alle Plots hinweg dieselbe Farbe hat. Deshalb ist es ratsam, möglichst noch vor Erzeugung des ersten Plots die genauen Farben je Kategorie als Dictionary zu definieren und dieses dann im gesamten Projekt immer wieder zu verwenden.

Weitere Ressourcen

- Seaborn Color Palette Basics | Using named and custom color palettes in Python seaborn
- Advanced Seaborn Color Palettes | Cubehelix palette, xkcd colors, choose_colorbrewer_palette

Übungen

Nutze ein Tool (z.b. Coolors Image Picker) um eine Farbpalette aus einem Logo oder einem Bild zu extrahieren und verwende diese Farbpalette dann in einem Plot.

- (A) Geschafft