Matthias Mruzek-Vering, M.Sc.

Hochschule Karlsruhe
University of
Applied Sciences

DSCB 230 - Aufgabenblatt 7

Die Aufgabenblätter werden in dieser Veranstaltung in Jupyter Notebooks veröffentlicht und bearbeitet. Diese finden Sie in der Github Organisation für Data Science 2 unter dem Repository *dscb230-tutorial* (https://github.com/hka-mmv/dscb230-tutorial). Die Musterlösung wird ebenfalls in Form eines Jupyter Notebook in Github hochgeladen.

Aufgabenteil 1: Unit Test

Schreibt ein Programm welches Pi annähert und testet dieses anschließend

Hierfür gibt es viele verschiedene Ansätze, es kann Pi mittels *math* oder auch *numpy* abgefragt werden. Versucht nun mittels der Leibniz Formel (https://en.wikipedia.org/wiki/Leibniz_formula_for_%CF%80) Pi eigenständig zu errechnen. Anschließend soll das Ergebnis eurer eigenen Rechnung in einem Unit-Test überprüft werden. Wie kann nun eure Lösung überprüft werden, ohne Pi *hardcodiert* in den Code reinzuschreiben? Überprüfen Sie die ersten n Nachkommastellen.

Fortgeschritte Aufgabe: Versucht ebenfalls Euler und Lambert zu implementieren, erweitert entsprechend die Test Fälle.

Tim Ehmann
Fabian Juny
Raphaele Salvatore Licciardo, B.Sc.

Aufgabenteil 2: Fehlerbehandlung

Ein Streaming Anbieter strukturiert seine Daten in zwei Klassen. Die Klasse Account repräsentiert einen Account eines Nutzers, in dem Name, Alter und über den Account angemeldete Geräte gespeichert werden. Über die Methode device_hinzufuegen kann ein Gerät den angemeldeten Geräten hinzugefügt werden.

Die Klasse Film repräsentiert einen Film mit Name, Rating und FSK. Über die Methode alter_ueberpruefen kann geprüft werden, ob ein Account das Mindestalter für den Film unterschreitet. Mit der Methode lade_rating wird das Rating des Film aus einem separaten df geladen.

Ihre Aufgabe ist es drei Fehler abzufangen, bzw. zu handeln:

1. Erstellen Sie eine eigene Exception, die aufgerufen wird, wenn die Maximalanzahl von Geräten (5) erreicht ist, jedoch trotzdem versucht wird ein weiteres Gerät hinzuzufügen. Die Fehlermeldung soll zurückgeben: "Die Maximalanzahl an Geräten für den Account von {Account.name} ist erreicht". Ersetzen Sie {Account.name} mit dem Accountnamen.

2. Erstellen Sie eine eigene Exception, die aufgerufen wird, wenn das Mindestalter für einen Film unterschritten wird. Die Fehlermeldung soll zurückgeben: "Das Mindesalter für den Film {Film.name} wurde um {Differenz} Jahre unterschritten". Ersetzen Sie {Film.name} mit dem Accountnamen und {Differenz} mit der Differenz des Mindestalters und dem im Account angegebnen Alter.

3. Es kann sein, dass zu einem Film kein Rating gespeichert ist, aber trotzdem versucht wird, dieses zu laden. Fangen Sie den Fehler mit einem try/except ab. Sollte das Rating nicht geladen werden können, soll ein Rating von 5 festgelegt werden.

Hinweis: https://www.programiz.com/python-programming/user-defined-exception

```
class Device:
    def __init__(self, name, ip):
        self.name = name
        self.ip = ip

class Account:
    def __init__(self, name, alter):
        self.name = name
        self.alter = alter
        self.devices = []

    def device_hinzufuegen(self, device):
        if len(self.devices) == 5:
            # Hier die Exception auslösen
            pass

        else:
            self.devices.append(device)

class Film:
    def __init__(self, name, fsk):
        self.name = name
        self.rating = None
        self.fsk = fsk

    def alter_ueberpruefen(self, account):
        if self.fsk >= account.alter:
            # Hier die Exception auslösen
            pass

    def lade_rating(self, df_rating):
        self.rating = df_rating.loc[self.name][„rating"]

import pandas as pd
film_rating = {'name': ['Titanic', 'Der_Pate', 'Pulp_Fiction',
'Inception', 'Rocky'],
               'rating': [7.9, 9.2, 8.9, 8.7, 8.0]}
df_filme = pd.DataFrame(film_rating,columns = ["name", "rating"])
df_filme.set_index("name", inplace=True)
df_filme
```

```
handy = Device("Handy", "164.117.163.174")
acc = Account("Bob", 11)
matrix = Film("Matrix", 16)
for _ in range(5):
    acc.device_hinzufuegen(handy)



Folgende Zellen sollten die selbsterstellten Exceptions auslösen:

# 1
acc.device_hinzufuegen(handy)

# 2
matrix.alter_ueberpruefen(acc)

Folgende Zelle sollte nach der Bearbeitung der Aufgabe keine
Fehler auslösen:

# 3
matrix.lade_rating(df_filme)
```

Aufgabenteil 3: Python 3.10 Features

Für for Schleifen ist die zip Methode in vielen Fällen sehr hilfreich, um mehrere Listen überschaulich zu kombinieren. Dabei kann es jedoch zu nervigen und schwer zu findenden Bugs kommen, wenn die Listen unterschiedlich lang sind, besonders bei größeren Datensätzen.

```
l1 = ["Martin", "Paul", "Sahra", "Justin", "Emmy"]
l2 = ["Stuttgart", "Karlsruhe", "Berlin", "Frankfurt", "München"]
l3 = [70173, 76187, 10115, 60306]

for name, ort, plz in zip(l1, l2, l3):
    print (f"{name} aus {plz}-{ort} ")
```

Wenn Sie den Code ausführen, merken Sie, dass die Informationen zu Emmy aus München verloren gegangen ist, da die Postleitzahl von ihr in der Liste fehlt. Diesen Bug abzufangen würde in 3.9 oder älteren Versionen einen Vergleich der Listenlängen benötigen. In 3.10 kam der Parameter strict zur Zip-Methode hinzu. Hierbei können nur noch gleichlange Iterables gezippt werden, wenn strict auf True gesetzt ist.

```
for name, ort, plz in zip(l1, l2, l3, strict = True):
    print (f"{name} aus {plz}-{ort} ")
```

Wenn Sie versuchen, diesen Code nun auszuführen, wird ein Fehler geworfen.

Aus den Statistik Vorlesungen von letztem Jahr sollte Ihnen der Pearsonsche Korrelationskoeffizient noch ein Begriff sein. Diesen zu berechnen ist jetzt seit Python 3.10 direkt implementiert und kann über das statistics modul mithilfe von statistics.correlation() verwendet werden. Diese Zahl beschreibt aber nur den Grad einer linearen Korrelation. Zudem kann mit statistics.covariance() die Kovarianz zweier Merkmale und mit statistics.linear_regression() die Parameter einer Regressiongeraden berechnet werden.
Berechnen Sie zuerst den Pearsonschen Korrelationsgroße zwischen Alter und Größe bei Jungs und Mäödchen zwischen dem Alter 1 und 14. Stellen Sie danach die Größe von Mädchen und Jungs abhängig vom Alter mithilfe von matplotlib dar sowie die durch das statistics Modul berechnete Lineare Regressionsgerade.

```
import statistics
import matplotlib.pyplot as plt
```