Fact\_Sheet Pandas

KI\_4

2020/2021

Tobias Steiner

Inhalt

[Install and Import: 3](#_Toc56069005)

[Series and DataFrames: 3](#_Toc56069006)

[Creating DataFrames from scratch: 3](#_Toc56069007)

[Daten von einer CSV Datei lessen 3](#_Toc56069008)

[Daten einlesen von JSON 4](#_Toc56069009)

[Daten einlesen von SQL 4](#_Toc56069010)

[Zurück zu CSV, JSON oder SQL konvertieren 5](#_Toc56069011)

[Wichtigsten DataFrame Operationen 5](#_Toc56069012)

[Daten ansehen 5](#_Toc56069013)

[Info über die Datei bekommen 5](#_Toc56069014)

[Duplikate 5](#_Toc56069015)

[Column Cleanup 6](#_Toc56069016)

# Install and Import:

conda install pandas *🡪* **mit diesem Command installiert man Panda**

import pandas as pd 🡪**mit dem Command wird Pandas in Python Datei importiert**

# Series and DataFrames:

**Series** sind Spalten und **DataFrames** sind multi-dimensionale Tabellen.

# Creating DataFrames from scratch:

**Ein einfacher weg, ein DataFrame zu erstellen ist es, ein einfaches „dict“ zu verwenden.**

**Bsp:**

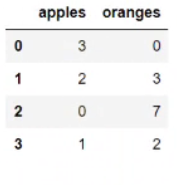
data = {

‘apples‘ : [3, 2 , 0 ,1]

‘orange‘ : [3, 2 , 0 ,1]

}

**Nachdem man es erstellt hat, muss man es in den Konstruktor des pandas DataFrames geben**

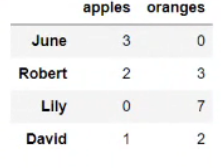
**Bsp:**

purchase = pd.DateFrame(data)

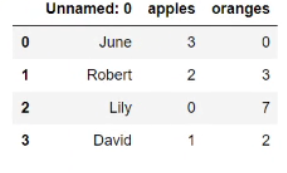
pruchase 🡪 mit dem Befehl gibst du es dann aus

**Mit dem nächsten Befehl, können wir den Index umändern:**

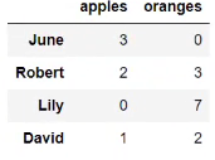
purchases = pd.DataFrame(data, index= ['June', 'Robert', 'Lily', 'David'])

purchases

# Daten von einer CSV Datei lessen

df = pd.read\_csv(‘purchuases.csv’)

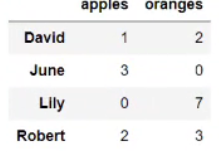
df **🡪 mit dem command gibt man es wieder aus**

**Als nächstes können wir unseren Index ändern:**

df = pd.read\_csv(‘purchuases.csv’, index\_col = 0)

df

# Daten einlesen von JSON

df = pd.read\_json('purchases.json')

df

**Manchmal funktioniert das einlesen der JSON Datei nicht und dazu muss man dann das Keyword „orient“ benutzen.**

**Link für eine Erklärung:** [**https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read\_json.html**](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_json.html)

# Daten einlesen von SQL

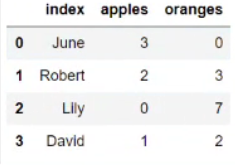
**Zuerst müssen wir den Command:** pip install pysqlite3, **in unserem Terminal eingeben.**

**Oder wir lassen den Command in einer Zelle unseres Notebooks laufen**: !pip install pysqlite3

**Um das dann in Python verwenden zu können muss man folgendes eingeben:**

import sqlite3

con = sqlite3.connect(„database.db”) **🡪 Eine Verbindung mit der Datenbank wird hergestellt.**

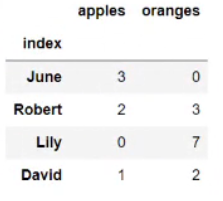


df = pd.read\_sql\_query(“SELECT \* FROM pruchase”, con)

df

**Mit dem folgenden command kann man den Index auf die Spalte festlegen:**

df = df.set\_index('index')

df

# Zurück zu CSV, JSON oder SQL konvertieren

df.to\_csv('new\_purchases.csv')

df.to\_json('new\_purchases.json')

df.to\_sql('new\_purchases', con)

# Wichtigsten DataFrame Operationen

movies\_df = pd.read\_csv("IMDB-Movie-Data.csv", index\_col="Title")

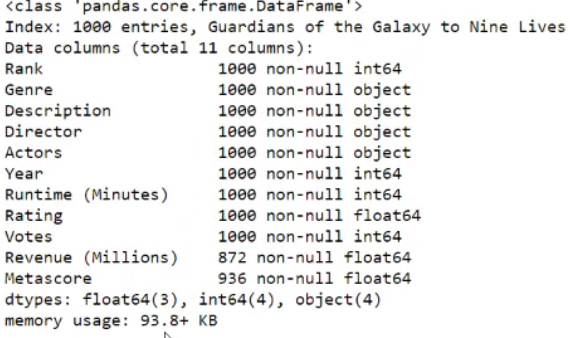
# Daten ansehen

**Der folgende Command gibt die ersten 5 Zeilen per default aus und als Parameter kann man auch andere Mengen festlegen, das funktioniert auch mit tail(das Ende)**

movies\_df.head()

movies\_df.tail()

# Info über die Datei bekommen

mavies\_df.info()

Mit dem Befehl sieht man wie viele Spalten es gibt und ihre Eigenschaften (Name, Anzahl der Einträge, Eigenschaft, Datentyp)

**Mit dem shape Command gibt man aus wie viele Reihen und Spalten es gibt.**

movies\_df.sjape()

# Duplikate

**Mit dem nächsten Command duplizieren wir die Reihen:**

temp\_df = movies\_df.append(movies\_df)

temp\_df.shape

**Und mit dem Command dropen wir sie wieder:**

temp\_df = temp\_df.drop\_duplicates()

temp\_df.shape

**Jedoch geht es mit dem folgenden Command wieder leichter:**

temp\_df.drop\_duplicates(inplace = True)

**Es gibt dann für die drop Methode 3 verschiedene Zustände bei dem Keyword „keep“**

* First
  + Es werden alle Duplikate bis auf das erste gedropped
* last
  + Das letzte Duplikat wird gedropped
* False
  + alle Duplikate werden gedropped

# Column Cleanup

Mit dem Command: movies\_df.columns kann man die einzelnen Spalten anzeigen.

**Mit dem Keyword „rename“ kann man es dann umbenennen.**

**Bsp:** movies\_df.rename(columns={

'Runtime (Minutes)': 'Runtime',

'Revenue (Millions)': 'Revenue\_millions'

}, inplace=True)

movies\_df.columns

**Man kann es auch anders hinzufügen:**

movies\_df.columns = ['rank', 'genre', 'description', 'director', 'actors', 'year', 'runtime', 'rating', 'votes', 'revenue\_millions', 'metascore']

**Mit dem Command ändert man die Namen der Spalten zu Kleinbuchstaben:**

movies\_df.columns = [col.lower() for col in movies\_df]

**Mit dem Command:** movie\_df.isnull() **kann man sich ausgeben lassen welche Zeilen keinen Wert beinhalten.**

**Das Gleiche kann man auch mit Spalten machen:** movies\_df.isnull().sum()

**Wenn man die NullWerte droppen möchte benutzt man folgenden Command:** movies\_df.dropna()

**Mit** inplace = true **verändert man dann wirklich den DataFrame.**

**Wenn man eine ganze Spalte löschen will benutz man folgenden Command:**

movies\_df.dropna(axis = 1)