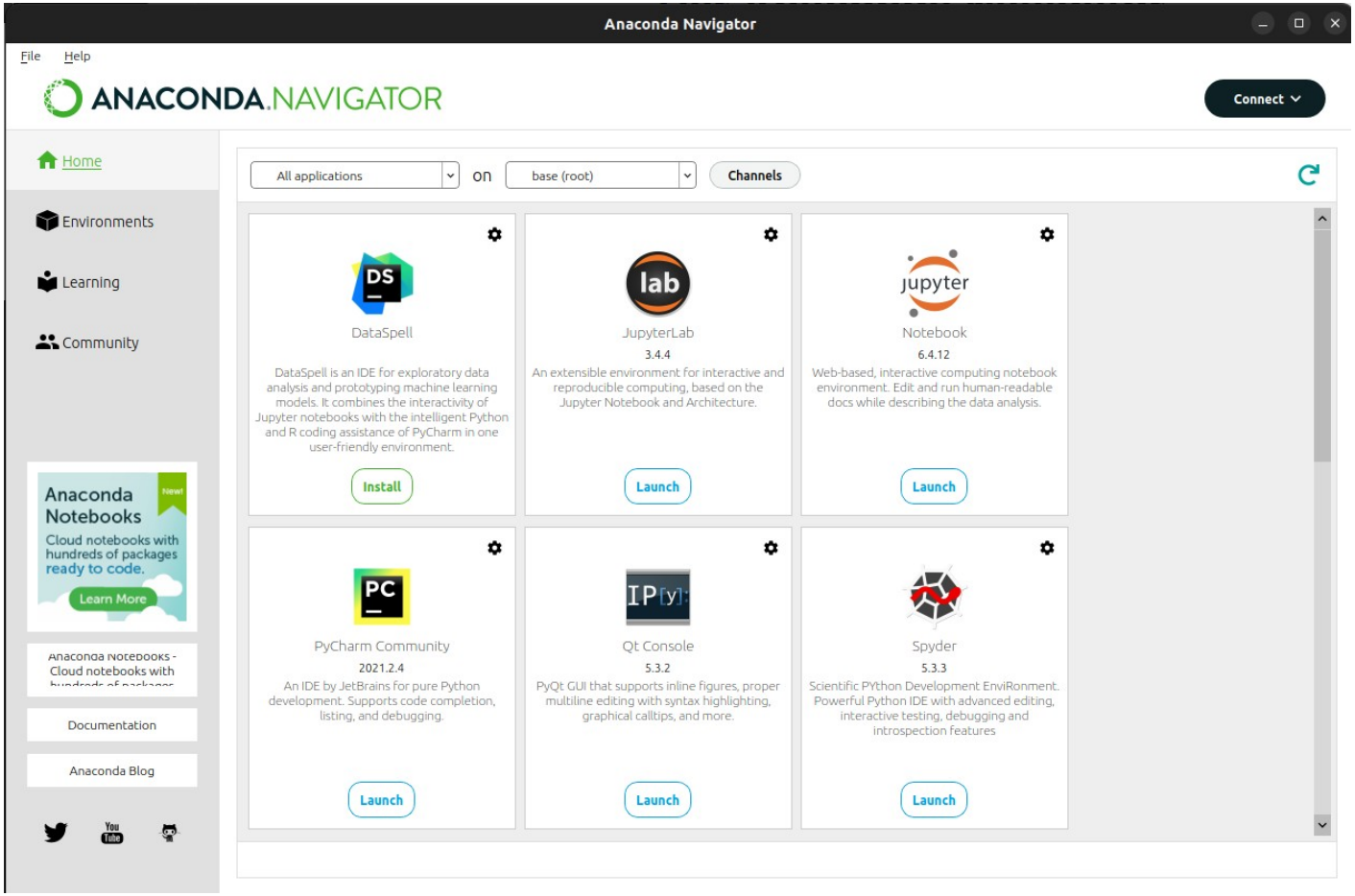
Tag 7
Inhaltsverzeichnis

Referenzen
Viele davon auf Englisch

- Offizielle Dokumentation: https://jupyter-notebook.readthedocs.io/en/stable/
- Miniconda Installationsanleitung: hier
- Eine gute Einführung von Jupyter Notebook: https://www.dataquest.io/blog/jupyter-notebook-tutorial/
- Daten mit Plotly visualisieren: https://plotly.com/python/
- Markdown: https://de.wikipedia.org/wiki/Markdown
- Pandas 10 Min. Guide hier
  - Siehe *Notebook_ITP_DMOD7_Pandas10MinGuide_V1.0.ipynb*
- Blick ins CheatSheets – Verzeichnis zu werfen, lohnt sich

Der Anaconda Navigator
Übersicht

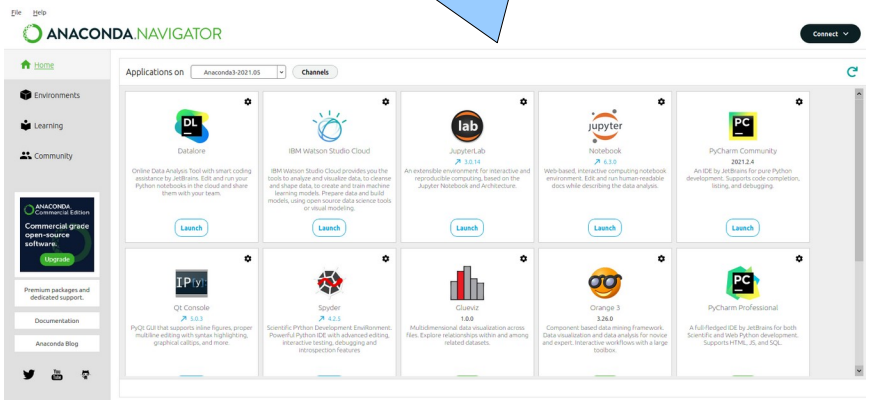


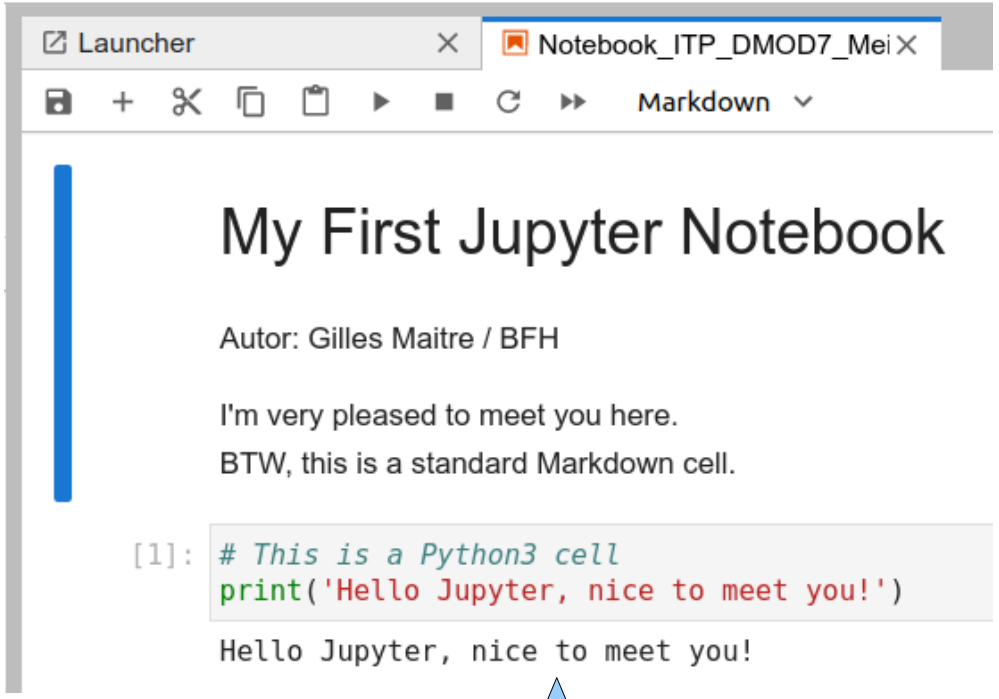
- Davon brauchen wir nur Spyder und Jupyter Lab.
- Beide kann man auch im Anaconda Prompt von Hand starten.
- *Achtung: Nicht alle Browser machen mit. Firefox und Chrome OK.*

Was ist ein Jupyter Notebook?
Prinzip

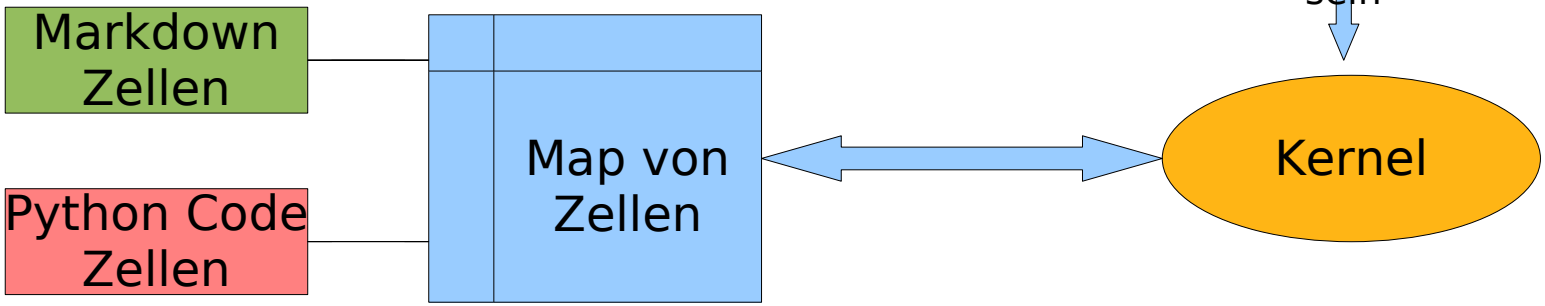
Im Anaconda Navigator
Jupyter Lab starten
oder
Anaconda Prompt starten,
und "jupyter lab" eintippen

Interaktive Ausführung im Default-Browser

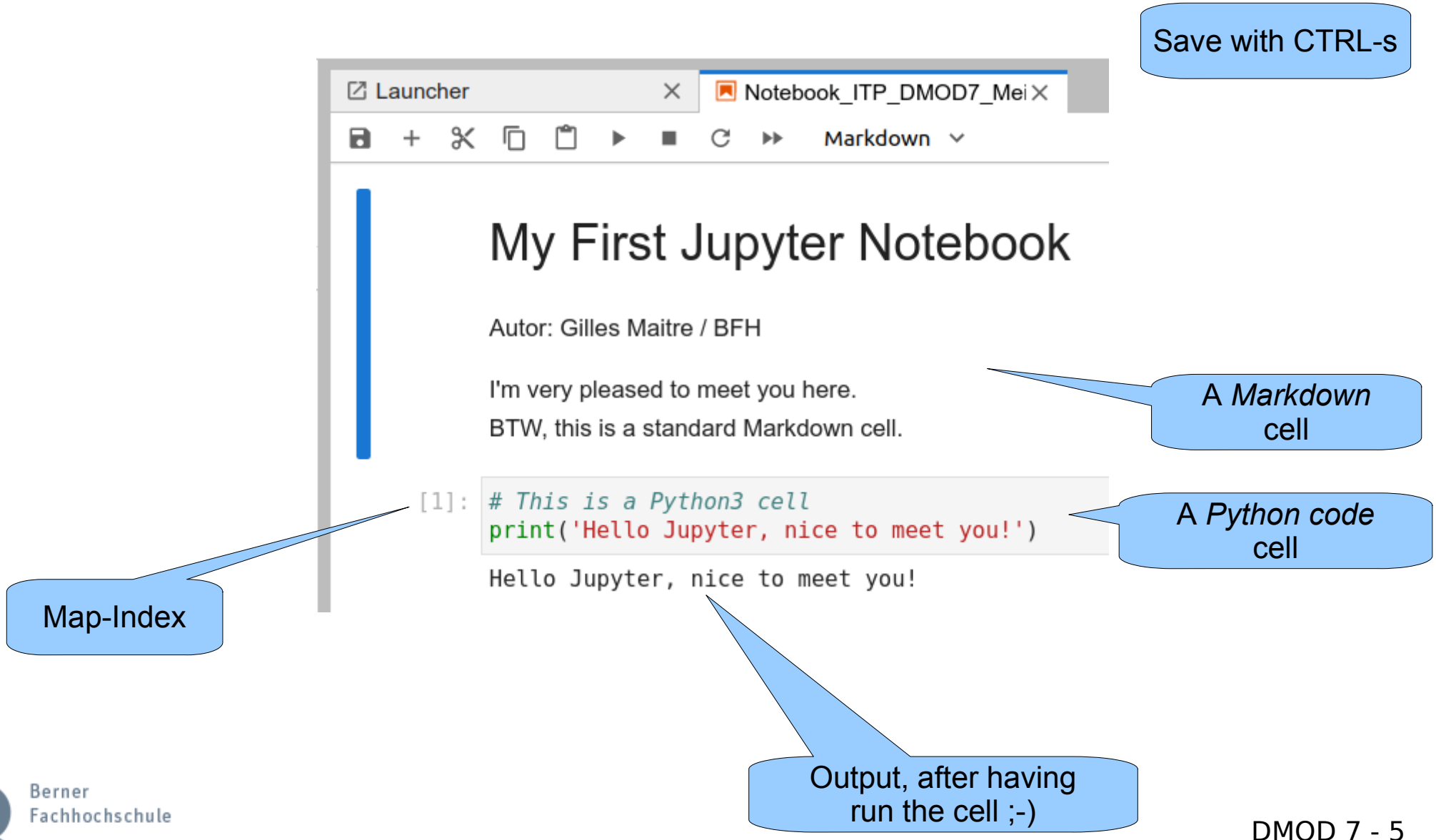




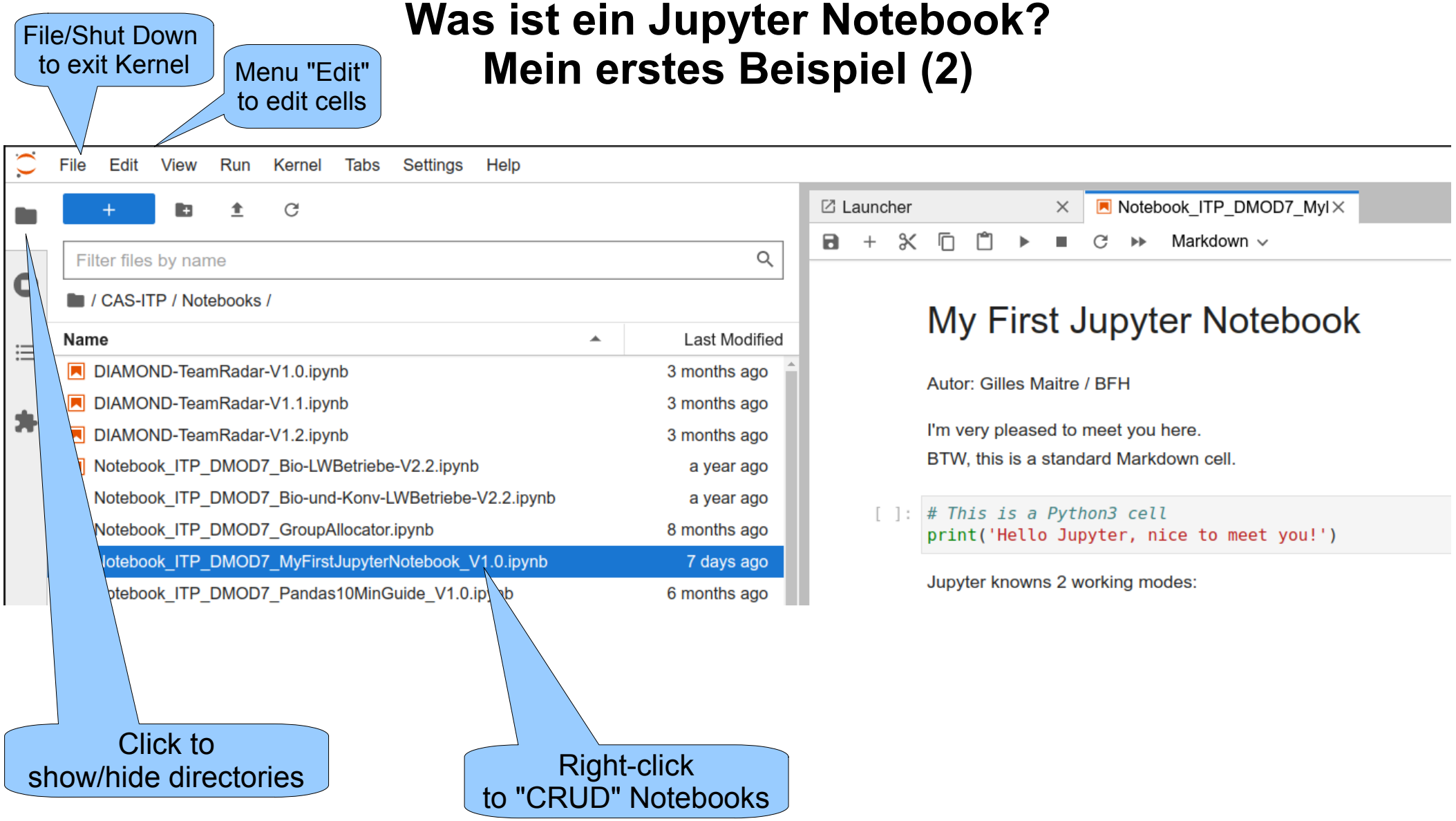
Kann "remote" sein



Was ist ein Jupyter Notebook?
Mein erstes Beispiel (1)



Was ist ein Jupyter Notebook?
Mein erstes Beispiel (2)



Was ist ein Jupyter Notebook?
Mein erstes Beispiel (3)

Let's have
a demo

Was ist ein Jupyter Notebook?
Notebook- und Daten-Verzeichnis

Es gibt ein Verzeichnis für die Notebooks und ein für die Daten.

1) Die Zip-Datei mit dem Name *CAS-ITP_DMOD_NotebooksAndData_V*.zip* aus Moodle herunterladen.

2) Diese Zip-Datei entpacken und die beiden Verzeichnisse in Ihr CAS-ITP Verzeichnis kopieren (am besten unter C:\Users\<IhrUsername>*CAS-ITP_DMOD_NotebooksAndData*).

3) Jetzt sollen Sie in diesem Verzeichnis beiden Unterverzeichnisse haben:
   1) Notebooks
   2) Data

4) Anaconda Navigator starten und Jupyter Lab auswählen
   oder
   Anaconda Prompt starten und darin "jupyter lab" eintippen.

5) Ihr Internet-Browser soll jetzt Jupyter Lab anzeigen.

Falls Sie die Zip-Datei anderswo speichern wollen, sollen Sie:
- Anaconda Prompt starten.
- Darin folgenden Befehl eintippen: "*cd <Verzeichnis, wo Sie die Zip-Datei entzippt haben>*".
- Dann "jupyter lab" eintippen.

Was ist ein Jupyter Notebook?
Vergleich Skript vs Notebook-Entwicklung

Skript/Programm (zB. Spyder)

- **Lineare** Entwicklung "Edit/(Compile)/Debug"-Zyklus
- Debugger/Tracing gehört dazu
- Ideal für "**produktive**" Entwicklung
- Nicht ideal für "**prospektive**" Entwicklung

Notebook (zB. Jupyter)

- **Interaktive** / **Inkrementelle** Entwicklung
- Kein Debugger => Arbeit mit den Zellen
- Nicht ideal für "**produktive**" Entwicklung
- Ideal für "**prospektive**" Entwicklung

Anwendungsbeispiel bei der SBB-Infrastruktur:
1) Data-Scientists/Bahn-Ingenieure entwickeln Prototypen mit Notebooks
2) Software-Ingenieure *integrieren/weiterentwickeln* diese für TEST/PROD-Systeme

Was ist ein Jupyter Notebook?
Spickzettel

- Hier der Link auf den Spickzettel
  - <SHIFT>-<RETURN> Aktuelle Zelle(n) ausführen
  - <CTRL>-<RETURN> Aktuelle Zelle(n) ausführen
  - a oder b Code-Zelle "**a**bove" / "**b**elow" addieren (in Command-Modus)
  - d d aktuelle Zelle löschen (in Command-Modus)
  - z Gelöschte Zelle wieder beleben (in Command-Modus)
  - <CTRL>-s Notebook ins .checkpoint Verzeichnis speichern
  - <TAB> "Code completion", je nach Kontext
  - ? Help anzeigen (Funktion oder Objekt)

- Wichtigste Pandas *DataFrame*-Befehle
  - df # Am Schluss einer Zelle: DataFrame anzeigen
  - df.shape # DataFrame-Dimensionen (Zeilen x Spalten)
  - df.head() # Erste 5 Zeilen anzeigen
  - df.info() # DateFrame-Datentypen anzeigen
  - df.columns # DataFrame-Spalten (ohne "()") anzeigen
  - df.index # DataFrame-Index (ohne "()") anzeigen
  - df.describe() # Einfache Statistik durchführen

Was ist ein Jupyter Notebook?
jupyter notebook vs. jupyter lab

jupyter notebook

- "Altes" Werkzeug für die Arbeit mit Notebooks
- Sehr gut bekannt und dokumentiert
- Ende der Entwicklung absehbar
- Rudimentär, weit weg einer Entwicklungsumgebung, wie Spyder

jupyter lab

- Künftiges Werkzeug für die Arbeit mit Jupyter Notebooks
- 100% kompatibel mit den Jupyter Notebooks
- Weniger bekannt und dokumentiert
- In der Entwicklung, Überraschungen nicht ausgeschlossen ;-)
- "Dark-mode"-Theme vorhanden ;-)
  - Siehe "Setting / Themes"

Im Unterricht brauchen wir jupyter lab

Pandas in a Nutshell
Was ist Pandas?

pandas is a Python package (…) [which] (…) aims to be the fundamental high-level building block for doing **practical, real-world data analysis in Python**.

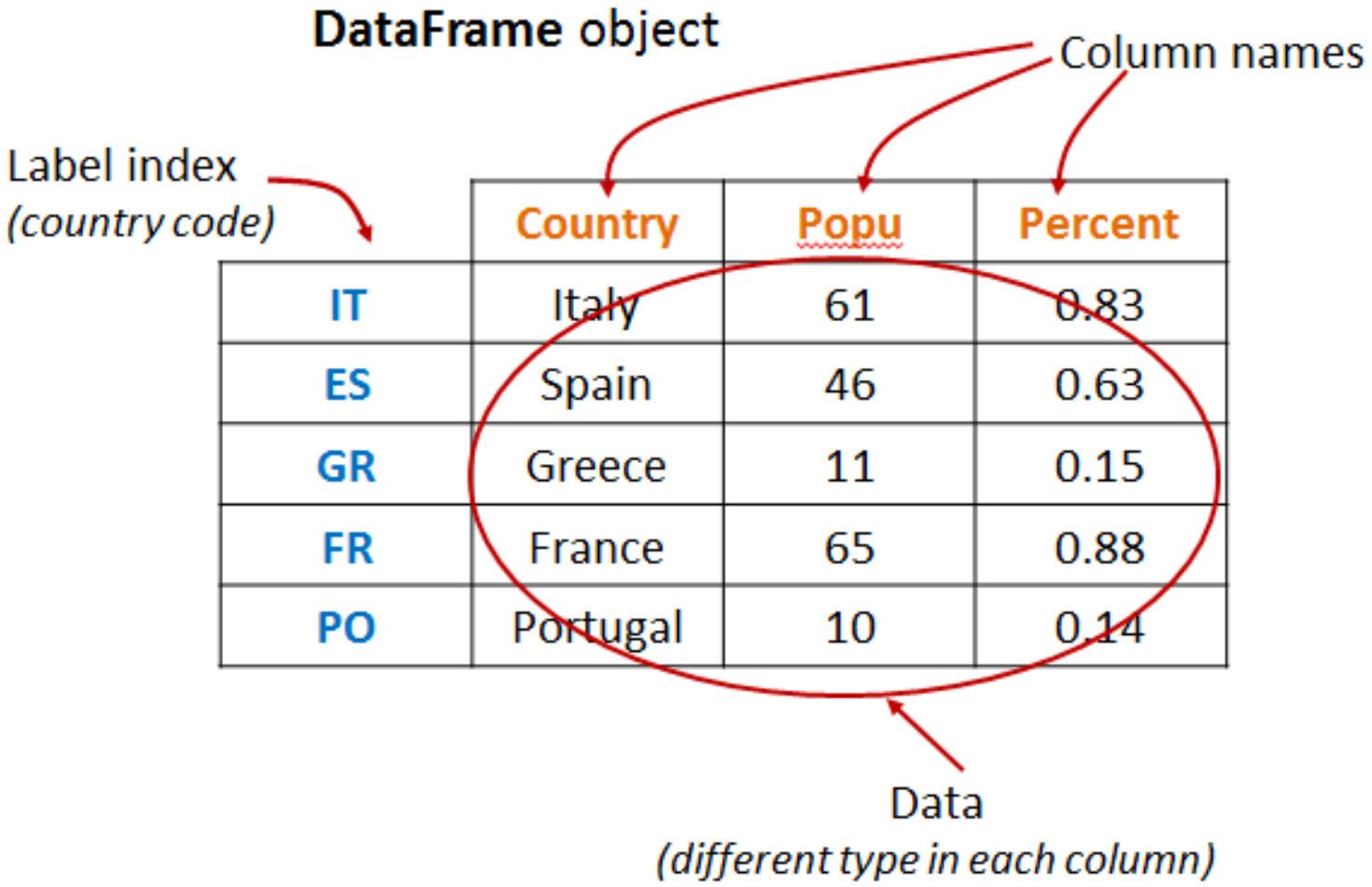
In one word:
a fantastic *data engineering* tool

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels

- Basic data structures: Series (1-dimensional) and DataFrame (2-dimensional)
- Handling of missing data
- Columns and rows CRUD-ing
- Reshaping, pivoting data
- Merging, joining data
- Powerful groupBy (split-apply-combine)
- IO tools to load data from CSV, JSON, Excel, HDF5

Pandas in a Nutshell
Was sind DataFrames?



Source: http://pytolearn.csd.auth.gr/b4-pandas/40/dataframe.html

Pandas in a Nutshell
Typen von DataFrames

Excel

- Not really a DataFrame tool, but...
- **Local** computations on **one** CPU (?)

Python / Pandas

- Classical / well known
- **Local** computations on **one or more** CPUs
- Used by Anaconda / Data Scientists

Python / Spark (PySpark)

- **Distributed** computations on **a cluster of** CPUs (BigData)
- Used by Spark / Hadoop / Data Scientists

Pandas in a Nutshell
Ein Reference Notebook

- Object creation
- Viewing data
- Selection
- Handling missing data
- Operations / Applying
- Merging / Joining
- Grouping
- Reshaping (stacking and pivoting)
- Time series
- Data frames
- Reading CSV, Excel and JSON files

Das alles habe ich in einen Notebook gepackt

Sources: https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html
und *Notebook_ITP_DMOD7_Pandas10MinGuide_V1.0.ipynb*

Analyse Bio-LWBetriebe 1990-2021
Suchen nach Daten

- Ein erster Versuch: OpenData und dann Wiki Quickstart
  - Hier sieht man, welche Daten verfügbar sind.
  - Und spezifisch für die Schweiz. Gut zu wissen!

- Ein anderer Weg: Das Bundesamt für Statistik
  - Datenüberblick hier
  - Ich suche nach "landwirtschaft betriebe"
  - Unter Landwirtschafliche Betriebe und Beschäftigte nach Kanton finden wir etwas
  - Die Daten kann man hier runterladen, aber man muss sagen, welche =>
  - Folgende Felder wähle ich:
    - Beschäftigte Total, alle Kantone (ohne Schweiz), "Bio-Betriebe" als Betriebssystem, für die Jahre 1990, 2000, 2010, 2020
    - Dateiformat: Tabelle
    - Speichern unter / "CSV" anklicken
  - Die Datei mit Excel öffnen... Kodierung: "ISO-8859-1" wählen

Analyse Bio-LWBetriebe 1990-2020
Analyse im Notebook / Überblick

Folgende Operationen führen wir jetzt im Notebook aus:

1) Daten in ein Pandas *Data-Frame* laden und anzeigen

2) Ein paar Spalten löschen

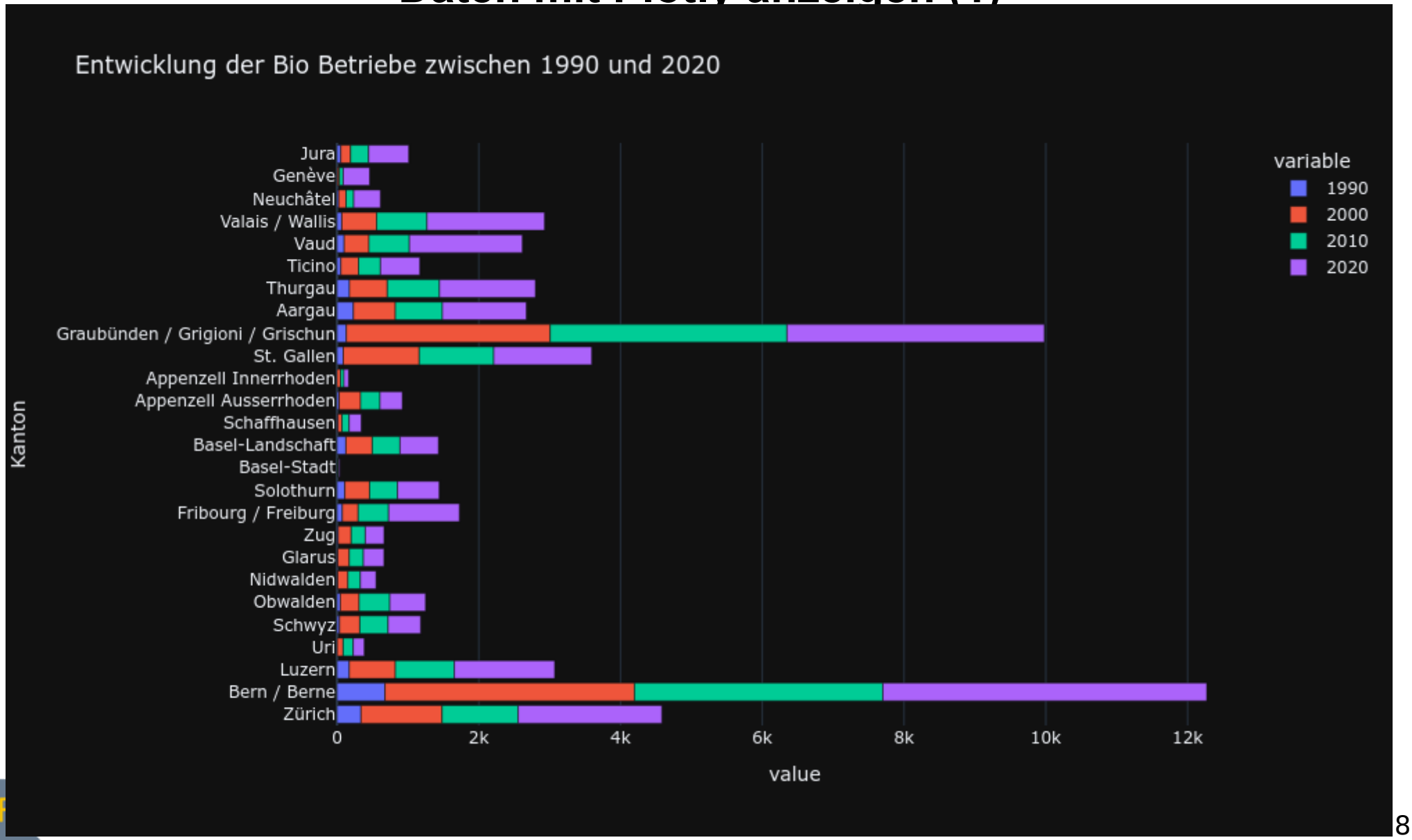
3) Einfache Statistik ausführen, mit *describe()*

4) Die Daten grafisch mit Plotly darstellen

5) Entwicklung zwischen 1990 und 2020 in % anzeigen

Jetzt ins Notebook umsteigen...

Analyse Bio-LWBetriebe 1990-2020
Daten mit Plotly anzeigen (1)



Analyse Bio-LWBetriebe 1990-2019
Daten mit Plotly anzeigen (2)

Plotly ist die *grosse Revolution*, um Daten zu visualisieren:

1) Arbeitet Hand in Hand mit Pandas *Data-Frames.*

2) Haufen von "Traces". Siehe hier.

3) Einfache Scatter-, Bar- und Line-Plots.

4) Diese Diagramme sind interaktiv... und farbig
Tolle Beispiele hier. (jemand farbenblind?)

5) Die Diagramme kann man als PNG (statisch) oder HTML (mit Animation) speichern.

Analyse Bio-LWBetriebe 1990-2020
Analyse im Notebook, Entwicklung in %

Jetzt noch die Entwicklung zwischen 1990 und 2020 in %

Dafür addieren wir eine neue Spalte *diff1990_2020*, die diese %-Zahlen enthält. Und sortieren das DataFrame nach dieser Spalte, aufsteigend.

Der Kanton Appenzell Innerrhoden hat eine unendliche Entwicklung hinter sich ;-)

```
biobetriebe_df['diff1990_2020'] = biobetriebe_df['2020']*100/biobetriebe_df['1990'] - 100
biobetriebe_df.sort_values(by = 'diff1990_2020')
```

	Kanton	2020	2010	2000	1990	diff1990_2020
11	Basel-Stadt	15	11	8	7	114.285714
12	Basel-Landschaft	543	391	372	125	334.400000
18	Aargau	1184	664	591	232	410.344828
10	Solothurn	589	394	348	111	430.630631
0	Zürich	2028	1073	1144	338	500.000000
1	Bern / Berne	4569	3504	3520	677	574.889217
19	Thurgau	1353	736	534	176	668.750000
2	Luzern	1416	832	649	174	713.793103
20	Ticino	555	309	249	55	909.090909
25	Jura	566	254	138	54	948.148148
14	Appenzell Ausserrhoden	315	273	303	30	950.000000
5	Obwalden	504	433	265	47	972.340426

Analyse Bio-LWBetriebe 1990-2020
Fazit

1) Wir haben mit wenig Python-Code eine einfache Analyse der Daten durchgeführt.

2) Mit Excel hätten wir das auch geschafft.

3) Wird die Analyse komplexer, ist IMHO die Jupyter-Lösung wesentlich stärker.

4) Ihre Meinung?

Notebooks vergleichen/teilen
Wie geht das?

1) Jupyter Notebooks sind dafür nicht gut geeignet.

2) Pragmatische Lösung dazu:
    1) Das Notebook als Python-Skript (Menü "File") herunterladen.

    2) Mit einem Dateivergleichsprogramm (z.B. kdiff3) die Python-Skripts vergleichen.
3) Um Notebooks zu verteilen, geht es *am einfachsten, die .ipynb-Dateien* herumzuschicken.

Notebook Beispiel
GroupAllocator

[3]:
```
studies = {'Vorname':['Silvy', 'StefanA', 'Dominic', 'Cora', 'Sabrina', 'Ursina', 'Astrid', 'Davide', 'André', 'ChristianC
                      'Cornelia', 'Fränzie', 'Patrick', 'ChristianL', 'Andreas', 'Alex', 'ThomasM', 'Jean-Louis', 'Miriam', '
data_df = pd.DataFrame(studies)
data_df.head()
```

[3]:

	Vorname
0	Silvy
1	StefanA
2	Dominic
3	Cora
4	Sabrina

Zufallzahlen + Gruppen erstellen und übernehmen

[4]:
```
# generate random integer values and groups
from random import seed
from random import randint
rnd_list = []
group_nr = []
counter = 0
for i in range(data_df.shape[0]):
    rnd_list.append(randint(0, 100))
    if (i % GROUP_SIZE == 0):
        counter += 1
    group_nr.append(counter)
print(rnd_list)
print(group_nr)
data_df['rnd'] = rnd_list
data_df.head()
```

```
[4, 41, 38, 11, 96, 21, 11, 72, 61, 32, 32, 84, 54, 93, 42, 67, 72, 62, 85, 47, 28, 63, 82, 61]
[1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10, 10, 11, 11, 12, 12]
```

[4]:

	Vorname	rnd
0	Silvy	4
1	StefanA	41
2	Dominic	38
3	Cora	11
4	Sabrina	96

Magic... ;-)

[5]:
```
data_df.sort_values(by=['rnd'], inplace=True)
data_df['group_nr'] = group_nr
data_df.drop(['rnd'], axis=1, inplace=True)
data_df
```

[5]:

	Vorname	group_nr
0	Silvy	1
3	Cora	1
6	Astrid	2
5	Ursina	2

Übungen
Analyse der konventionellen Betriebe

1) Wenn Sie die Softwares noch nicht installiert haben, ist es jetzt höchste Zeit… Siehe Seite 2.

2) Laden Sie aus Moodle die Zip-Datei mit Notebooks und Daten herunter (siehe Slide *Notebook- und Daten-Verzeichnis*).

3) Führen Sie selber eine Analyse der *konventionellen* Landwirtschaftsbetriebe am Ende des Notebooks aus.

Tipp: Machen Sie eine *Kopie* meines originalen Notebooks.