

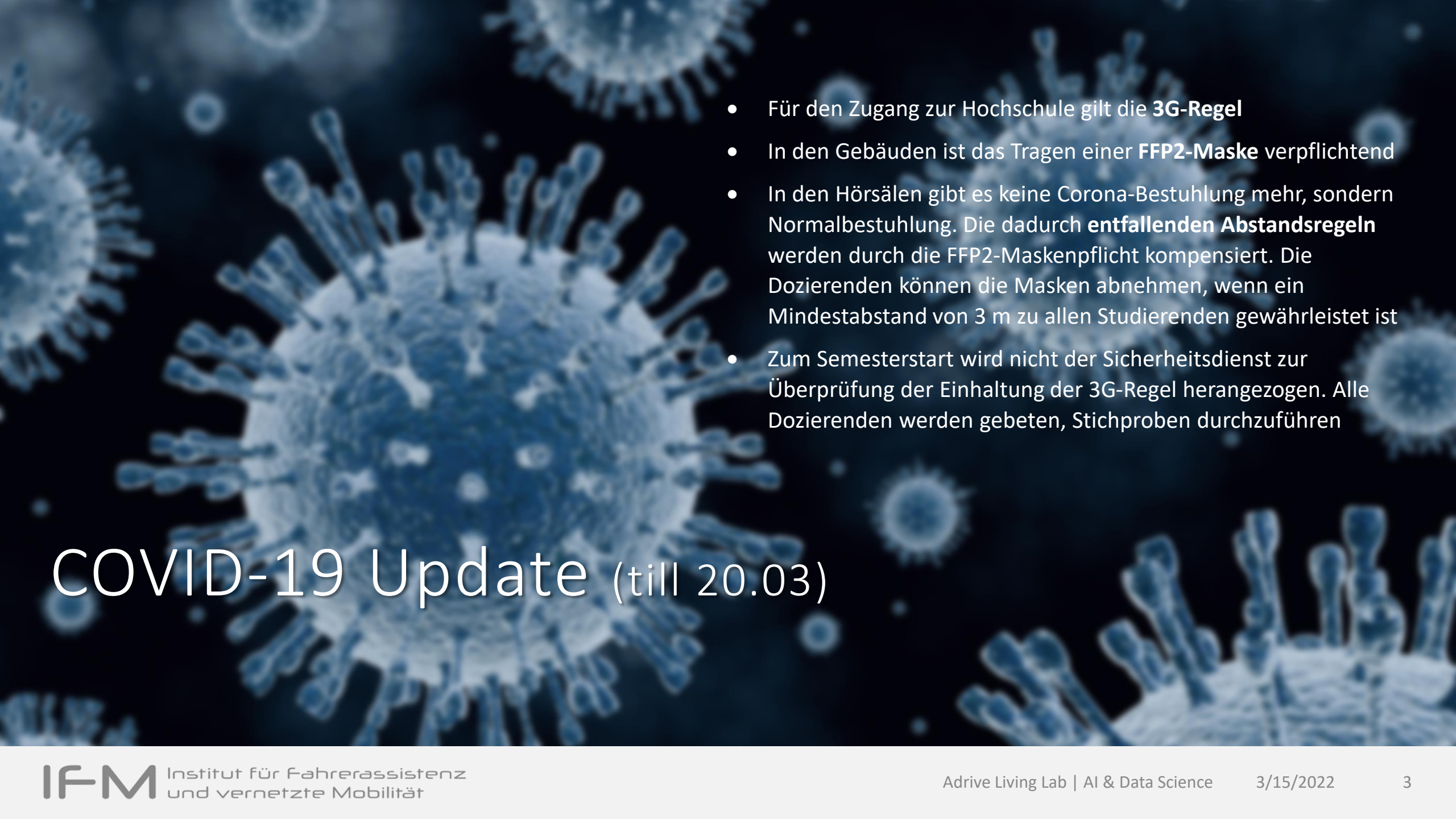
Data Science & Artificial Intelligence

Summer Term 2022



L01 Introduction

D. Schneider, B. Stuhr, J. Haselberger

- 
- Für den Zugang zur Hochschule gilt die **3G-Regel**
 - In den Gebäuden ist das Tragen einer **FFP2-Maske** verpflichtend
 - In den Hörsälen gibt es keine Corona-Bestuhlung mehr, sondern Normalbestuhlung. Die dadurch **entfallenden Abstandsregeln** werden durch die FFP2-Maskenpflicht kompensiert. Die Dozierenden können die Masken abnehmen, wenn ein Mindestabstand von 3 m zu allen Studierenden gewährleistet ist
 - Zum Semesterstart wird nicht der Sicherheitsdienst zur Überprüfung der Einhaltung der 3G-Regel herangezogen. Alle Dozierenden werden gebeten, Stichproben durchzuführen

COVID-19 Update (till 20.03)

Agenda

Theory (90 min)

Break (15 min)

Exercise (90 min)

L01.1 (15 min)

- Introduction
- Organizational
- Motivation

L01.2 (15 min)

- Definition Data Science
- Definition Artificial Intelligence

L01.3 (20 min)

- Data sets
- Iris
- MNIST
- CIFAR-10

L01.4 (40 min)

- Python
- Git
- Coding basics
- Environments

E01.1 (10 min)

- Google Colab

E01.2 (15 min)

- Python "hello world"

E01.3 (50 min)

- Python basics hands on

E01.4 (15 min)

- Data sets
- Scikit

Daniel Schneider

- PhD Student
 - Big Data & Objective Evaluation
- TU Graz, Austria
- AVL Deutschland, Karlsruhe
 - Objective Evaluation of lateral controlling ADAS
 - Fleet data evaluation
 - Knowledge Discovery



Bonifaz Stuhr

- PhD Student
 - Deep Learning
- Autonomous University of Barcelona, Spain
 - Self-supervised learning
 - Image-to-Image Translation
 - VDI Autonomous Driving Challenge



Johann Haselberger

- PhD Student
 - Deep Learning & Autonomous driving
- TU Berlin, Germany
- Porsche Engineering, Weissach
 - Deep learning-based lane keeping assistant
 - Driving-style detection
 - VDI Autonomous Driving Challenge

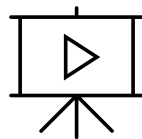


Theory/Lecture (L)

- $\approx 45\%$ of the time
- In person as well as recording/stream
- Definitions, descriptions and examples

Exercise (E)

- $\approx 45\%$ of the time
- In person as well as recording/stream
- Detailed realization in source code, examples and discussions



Recordings

Voices will be recorded during the lecture

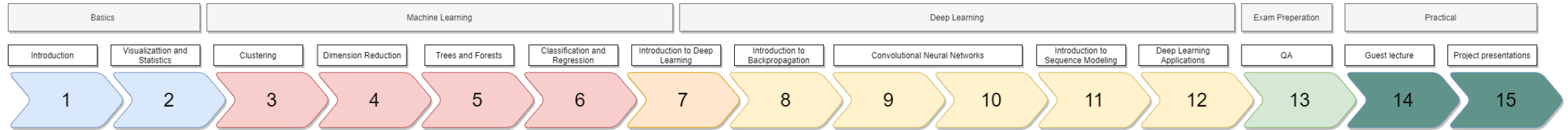
Home assignments (HA)

- $\approx 10\%$ of the time
- In person as well as stream \rightarrow No recoding!
- Discussions of given home assignments (all requirements provided during the lecture)

Final exam (EX)

- 90 min exam
- Fundamental and important concepts & definitions
- Important methods and models
- Basic mathematics
- Pseudo-code

Overview of the lecture in summer term 2022



Date	Content
16.03.2022	L01_Introduction
23.03.2022	L02_Visualization_and_Statistics
30.03.2022	L03_Clustering
06.04.2022	L04_Dimension_Reduction
13.04.2022	L05_Trees_and_Forests
20.04.2022	L06_Classification_and_Regression
27.04.2022	L07_Introduction_to_Deep_Learning
04.05.2022	L08_Introduction_to_Backpropagation
11.05.2022	L09_Convolutional_Neural_Networks_Part_1
18.05.2022	L10_Convolutional_Neural_Networks_Part_2
25.05.2022	L11_Introduction_to_Sequence_Modeling
01.06.2022	L12_Deep_Learning_Applications
08.06.2022	L13_Recap
15.06.2022	L14_Guest_Lecture
22.06.2022	L15_Project_Presentations

L01.1 Links and Further Information

Important links

- [Moodle](#) Data Science & KI (MB, EL, IF)
- [Google Drive](#)
- [GitHub](#)

Tools

[Xournal++](#)
[Rapid miner](#)
[Anaconda](#)

Helpful online lectures

- [MIT 6.S191: Introduction to Deep Learning – YouTube](#)
- [Introduction to Deep Learning \(I2DL\) - Technical University of Munich - Prof. Niessner, Prof. Leal-Taixe – YouTube](#)

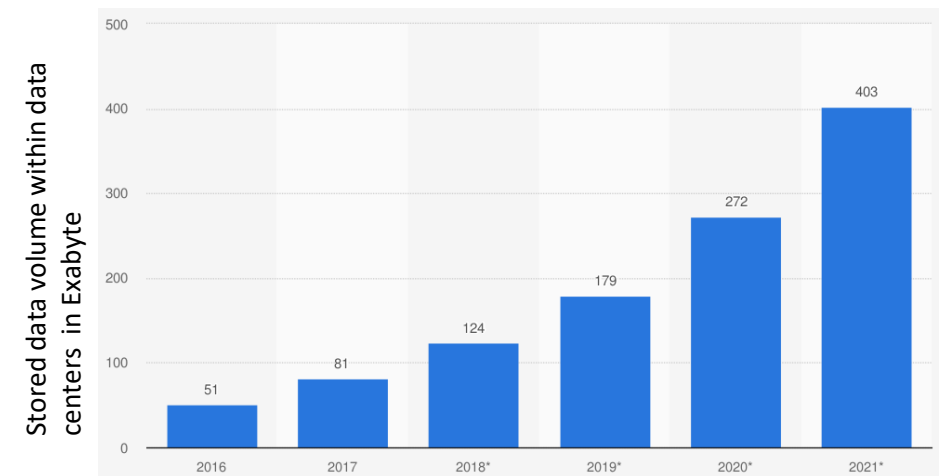
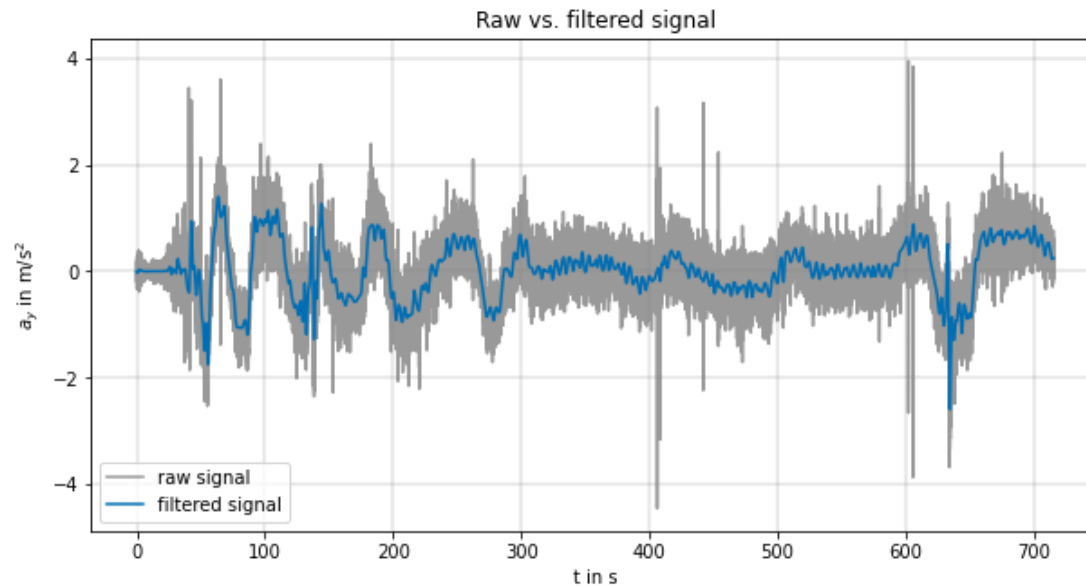
Books and further courses

- [Deep Learning \(deeplearningbook.org\)](#)
- [Artificial Intelligence: A Modern Approach \(berkeley.edu\)](#)
- [Learning From Data - Online Course \(MOOC\) \(caltech.edu\)](#)
- [Handbuch Data Science von Stefan Papp; Wolfgang Weidinger; Mario Meir-Huber; Bernhard Ortner; Georg Langs; Rania Wazir](#)
- [GitHub - instillai/deep-learning-roadmap: All You Need to Know About Deep Learning - A kick-starter](#)

L01.1 Motivation

- Total amount of data raised within the last decade exponentially
- In 2014 $2.5 \cdot 10^{30}$ bytes are produced per day [[Wu2013](#)]
- 2021 approx. 403 Exabyte stored in data centers [[Link](#)]
- Data itself are **useless without deeper understanding** and information extraction
- Not all data interesting/important (signal vs. noise) [[Papp2019](#)], thus extraction algorithms must be applied

- 1.024 Gigabyte = 1 Terabyte
- 1.024 Terabyte = 1 Petabyte
- 1.024 Petabyte = 1 Exabyte



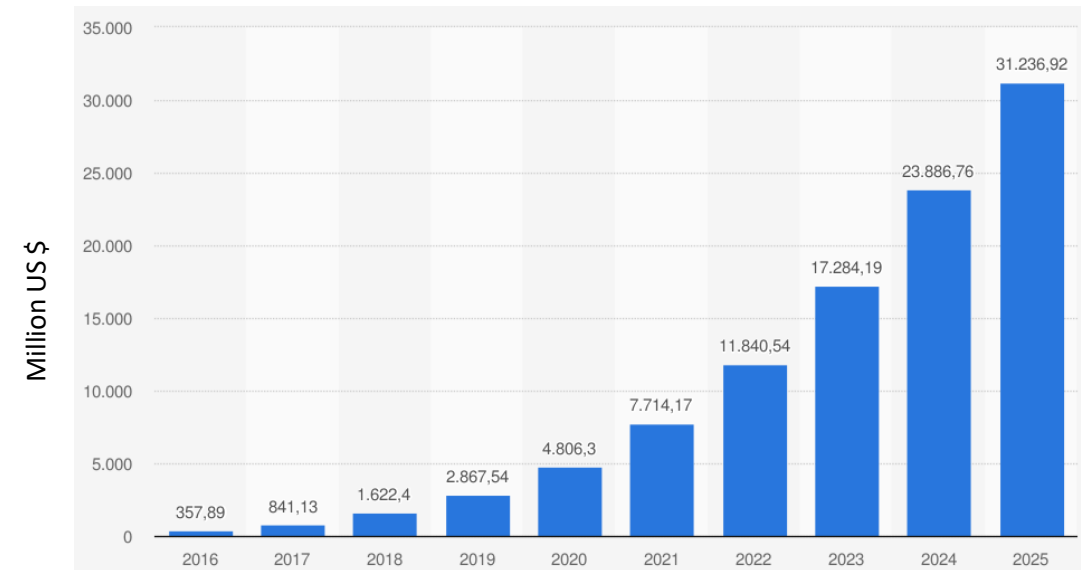
Data amount stored in data centers [[Link](#)]

L01.1 Motivation

Sales 2016-2025 in the field of AI

- Till 2025 over 31B \$ revenue in the field of AI [[Link](#)]
- Growth exponentially
- Huge market volume with a lot of applications available
- Many fields entered (medicine, automated driving, finance, ...)

Optical character recognition; Handwriting recognition; Speech recognition; Face recognition; Artificial creativity; **Computer vision**; Virtual reality; **Image processing**; Motion interpolation; Pixel-art scaling algorithms; Image scaling; Image restoration; Photo colorization; Film restoration; Photo tagging; Photo and video manipulation; Diagnosis; Game theory and strategic planning; Game artificial intelligence and computer game bot; **Natural language processing**, translation and chatterbots; **Nonlinear control** and robotics [[Application of AI, 28.05.2021](#)]



Forecast for revenue from enterprise artificial intelligence applications worldwide from 2016 to 2025 [[Link](#)]

Job perspective in the field of DS/AI and AD

- Data Scientists are **rare** and **needed** in a wide variety of fields
 - Good situation for you
 - You can often choose your employer
- Requirements:
 - Programming (**Python**, R, SQL, ...)
 - Libraries/Tools (Pandas, **PyTorch**, TensorFlow, Scikit, ...)
 - ML/AI (**mathematics**)
 - Visualization (Tableau, plotly, ...)
 - Docker
 - Often Linux knowledge



L01.1 Motivation

Job perspective in the field of DS/AI and AD

- Data Scientists are **rare** and **needed** in a wide variety of fields
 - Good situation for you
 - You can often choose your employer
- Requirements:
 - Programming (**Python**, R, SQL, ...)
 - Libraries/Tools (Pandas, **PyTorch**, TensorFlow, Scikit, ...)
 - ML/AI (**mathematics**)
 - Visualization (Tableau, plotly, ...)
 - Docker
 - Often Linux knowledge

Wie viel verdient man in welchem Berufsfeld?

IT & Development.

BERUFSFELD	EINSTIEGSGEHALT	JOBS	GEHALTSAUSSICHTEN
Datenbankentwicklung	39.100 Euro	Zu den Jobs	Zu den Infos
Hardwareentwicklung	50.855 Euro	Zu den Jobs	Zu den Infos
Mobile Development	55.797 Euro	Zu den Jobs	Zu den Infos
SAP-Beratung	47.580 Euro	Zu den Jobs	Zu den Infos
Softwareentwicklung	55.941 Euro	Zu den Jobs	Zu den Infos
Systemadministration	37.488 Euro	Zu den Jobs	Zu den Infos
Webentwicklung	39.842 Euro	Zu den Jobs	Zu den Infos
Wirtschaftsinformatik	65.206 Euro	Zu den Jobs	Zu den Infos

Ingenieurwesen & Technik.

BERUFSFELD	EINSTIEGSGEHALT	JOBS	GEHALTSAUSSICHTEN
Automatisierungstechnik	43.039 Euro	Zu den Jobs	Zu den Infos
Architektur	36.155 Euro	Zu den Jobs	Zu den Infos
Bauingenieurwesen	47.424 Euro	Zu den Jobs	Zu den Infos
Elektrotechnik	35.202 Euro	Zu den Jobs	Zu den Infos
Energietechnik	36.072 Euro	Zu den Jobs	Zu den Infos
Fahrzeugtechnik	30.764 Euro	Zu den Jobs	Zu den Infos
Fertigung & Produktion	39.018 Euro	Zu den Jobs	Zu den Infos
Mechatronik	33.994 Euro	Zu den Jobs	Zu den Infos
Maschinenbau	40.663 Euro	Zu den Jobs	Zu den Infos
Umweltingenieurwesen	46.958 Euro	Zu den Jobs	Zu den Infos
Wirtschaftsingenieurwesen	51.617 Euro	Zu den Jobs	Zu den Infos

Overview over the averaged income in field of AI, Germany in 2019 [[Link](#)]

L01.2 What a Data Scientist does

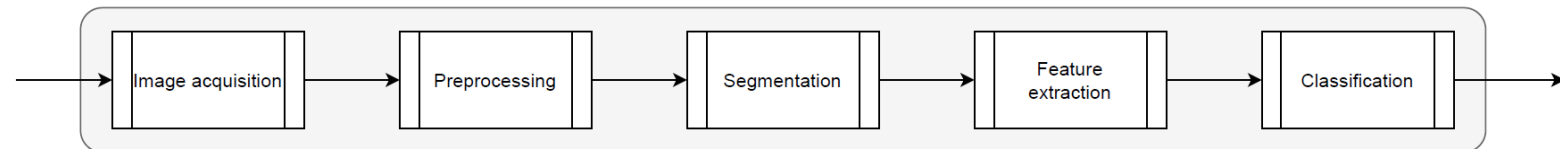
- Extract hidden information
- Build models
- Predict scenarios
- Summarize, prepare and deploy extracted information
- Data-driven decision making
- Business Intelligence (BI)
- ...

Create value from existing data by

- Combining computer science
- Modelling
- **Statistics**
- **Mathematic**
- Communication



Tesla AI based environmental perception [\[Link\]](#)



L01.2 Data Science Definition

Data

- At the beginning of your analysis
- Optimizing the data (deal with outliers, ...)
- No statements over the entire data set!



Data Science

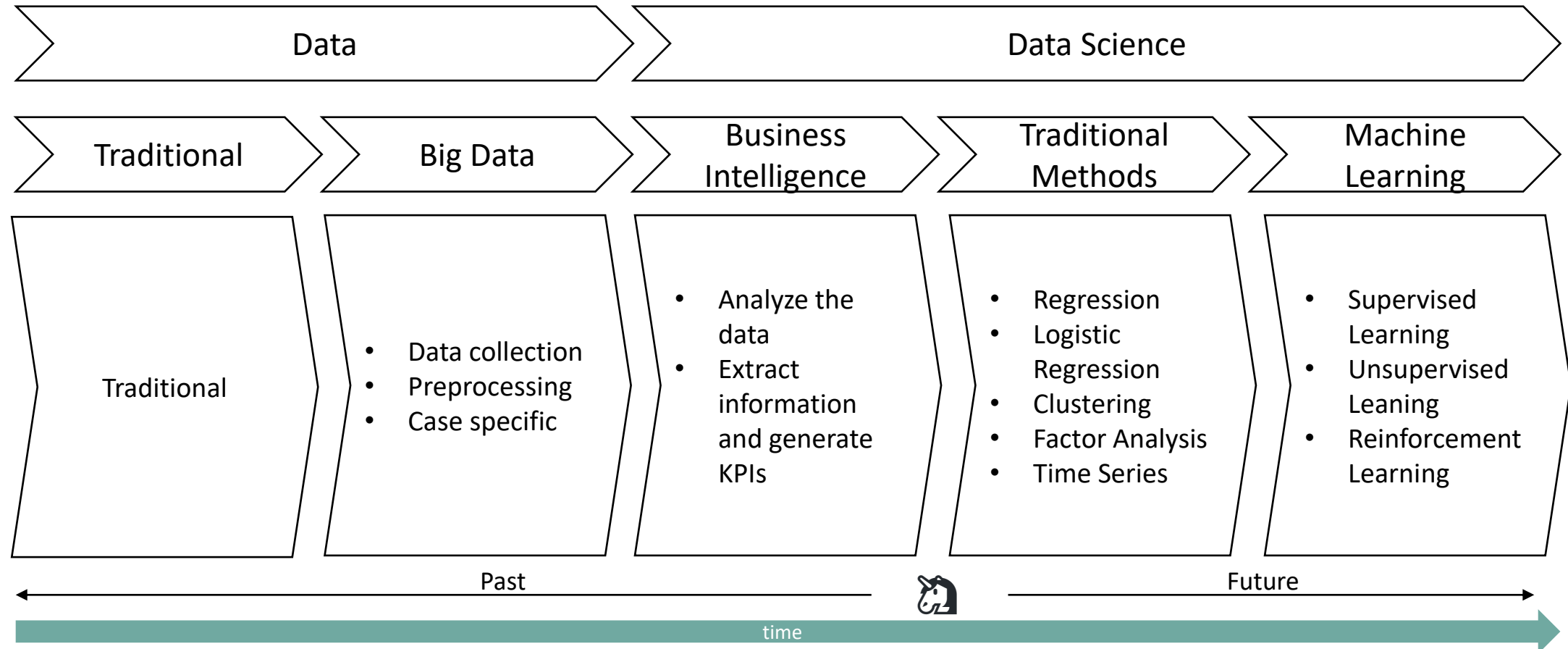
- Use data to create reports & dashboards
- Assess potential future scenarios
- Predict behavior and create forecasts

Time variant behavior

- Data transfer to Data Science over the time
- Data Science used to predict parts of the future

 = process/math

L01.2 Data Science Definition



L01.2 Artificial Intelligence Definition

Artificial Intelligence (AI)

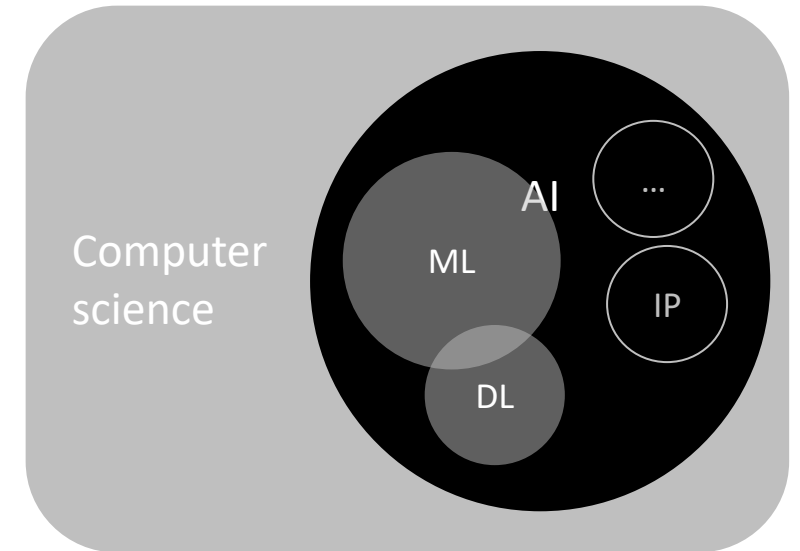
- Part of computer science
- Covers approaches such as [intelligent planning](#) (IP), searching, machine learning, deep learning, ...

Machine Learning (ML)

- Uses algorithms to analyze data
- Learn from that
- Make decisions based on what is learned

Deep Learning (DL)

- DL as a subset of ML
- Using approaches to imitate the structure of human brain (neurons, neuronal networks)
- DL is what drives the most human-like
- Deep-stacked blocks



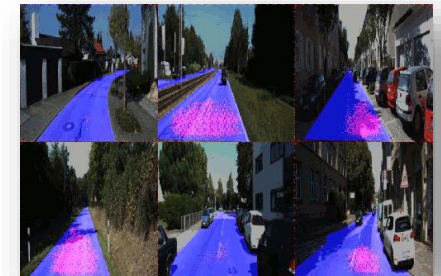
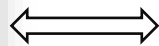
More information can be found here: [Deep Learning \(deeplearningbook.org\)](https://www.deeplearningbook.org)

L01.3 Data Sets

- Data collected from different sensors such as radar, camera, but also from surveys, business reports, stocks, ... and arranged systematically
- Distinguish between **labeled data sets** (e. g. [Iris](#), [KITTI](#), [ImageNet](#), ...) and **data collections** ([NYC Taxi](#), [S&P 500](#), ...)
- Public data sets accessible for example on [Kaggle](#)
- Often easily accessible via APIs or as in-build data sets (e. g. [TensorFlow](#), pulling from online source)

```
import torchvision
import torchvision.datasets as datasets
mnist_trainset = datasets.MNIST(root='./data', train=True,
                                download=True, transform=None) # Training
mnist_testset = datasets.MNIST(root='./data', train=False,
                                download=True, transform=None) # Testing
```

- Label often in additional file (for instance in KITTI) with link to input data



- [List of datasets for machine-learning research](#)

L01.3 MNIST

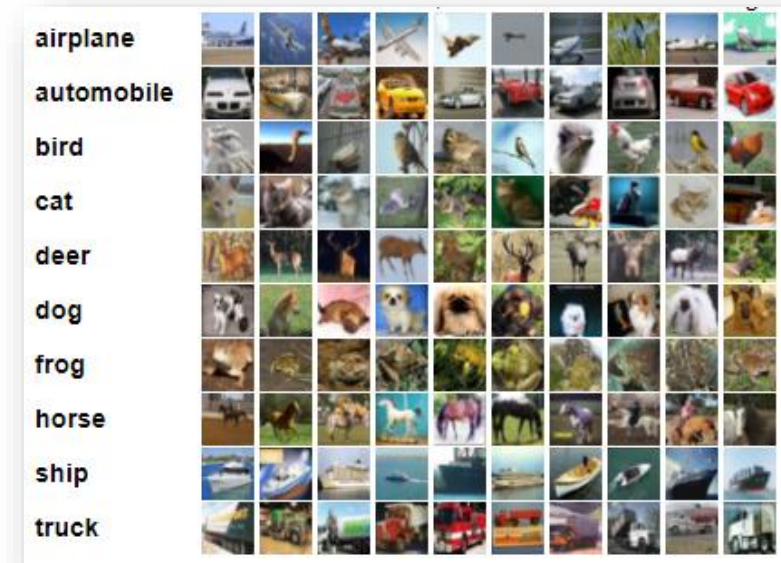
- **M**odified **N**ational Institute of **S**tandards and **T**echnology database (MNIST)
- [MNIST](#) often used for introduction to machine learning (we will also use this data set)
- Deals with complexity of handwritten letters
- MNIST consists of 60000 training data and 10000 testing data
- MNIST is a subset of larger NIST data set
- Samples already normalized and centered



MNIST example

L01.3 CIFAR-10

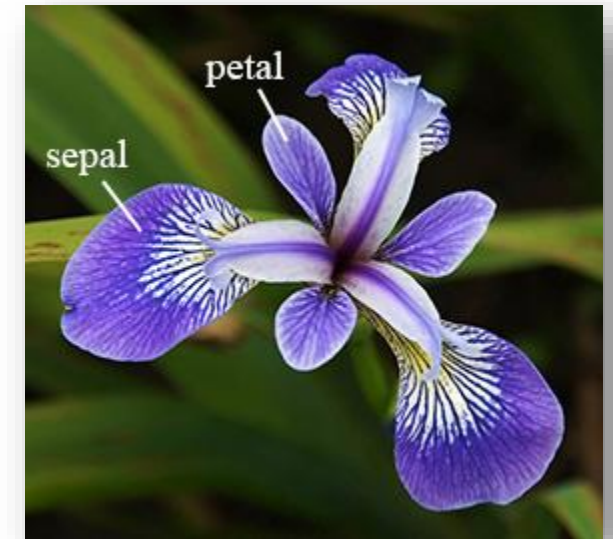
- Published by **Canadian Institute for Advanced Research** from [[Krizhevsky2009](#)]
- Consists of 60000 32x32x3 images in 10 classes
- CIFAR-10 consists of 50000 training data and 10000 testing data
- Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton
- Subset of CIFAR-100
- [CIFAR-100](#) consists of $80 \cdot 10^6$ images



CIFAR-10 example

L01.3 Iris

- Best known data set in literature (over 18500 citations, [published](#) in 1936)
- Consists of 3 classes (Iris Setosa, Iris Versicolour, Iris Virginica) with 50 instances each
- For each class, following parameters available:
 - sepal length in cm
 - sepal width in cm
 - petal length in cm
 - petal width in cm
- [The Iris Dataset · GitHub](#)
- [UCI Machine Learning Repository: Iris Data Set](#)



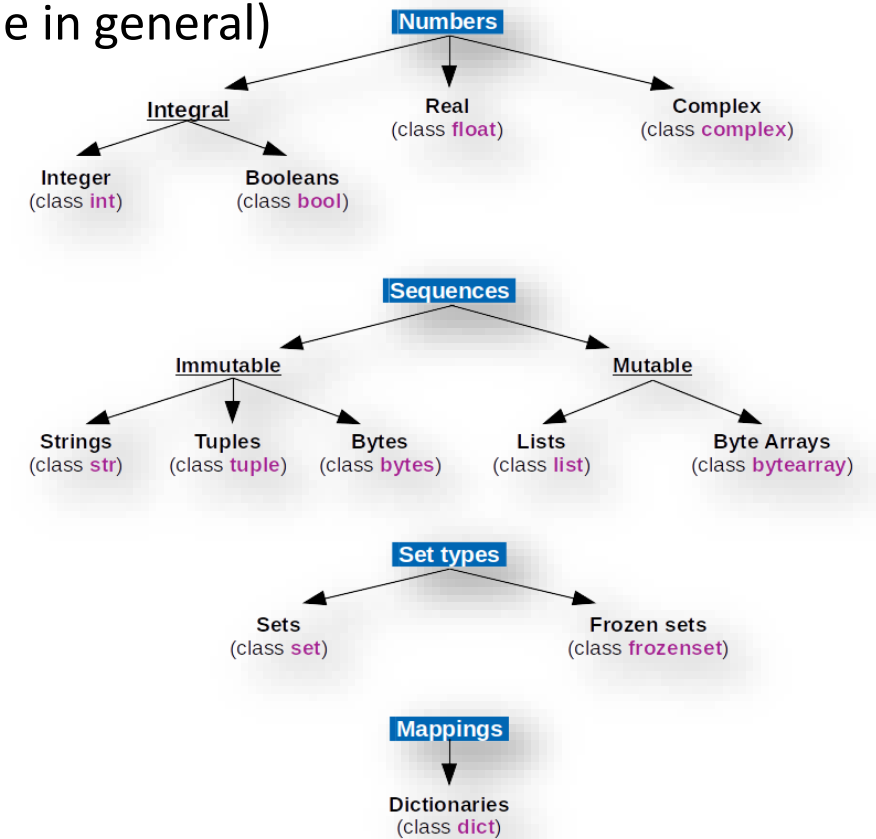
Iris example

L01.4 Python3

- Developed 1991 by Guido van Rossum
- Easy to learn **high-level** programming language
- Good readable due to indentation
- High impact in Data Science and Artificial Intelligence
- **Open Source which is predominant in AI-research**
- Large number of standard libraries available (Pandas, NumPy, PyTorch, ...)
- Large community
- **Object-orientated**
- GUI programming possible
- Runs on any platform (Linux, Windows, Mac, ...)
- Several IDEs available



- Use variables without declaring them
- Associativity: Left to right
- Call by “object reference” (neither call by reference, nor call by value in general)
- Do not have to deal with dereference etc. (compare to C/C++)
- Data types can be classified into:
 - Numbers
 - Sequences
 - Sets
 - Mappings
- Classes



Sequences

- Stores multiple numbers within one data sequence ($\mathbf{x} \in \mathbb{R}^{1 \times n}$)
 - Time series data for instance
 - `time = [0.1, 0.2, 0.3, 0.4, 0.5]`
- Tuple: `tup = (1, 2, 3)`
- List: `lis = [1, 2, 3]`
- Various in-built methods available
 - `append`, `pop`, `remove`, `del`, `index`, ...
- Lists of list also possible ($\mathbf{X} \in \mathbb{R}^{n \times m}$)
 - Greyscale image
 - `img = [[1, 2, 3], [1, 2, 3], [1, 2, 3]]`

```
## Lists
l = [1, '2', str(3), 4+1j, 55e-10] # pos: 0, 1, 2, 3, 4

# Length of a list (elements)
len(l)

# Slicing (access areas)
l[0] # first element, compare to position
l[-1] # last element (- operator equals to "from end")
l[1:4] # slice over an area within l

# Appending elements to list
l.append('sixth')
# Appending list to list
l.extend([7,8])

# Remove values
l.pop() # Pops last element
l.pop(2) # Pops third element (from start)
l.remove('sixth') # Remove element by specific key
del l[0]

# Get index (position) of specific value
l.index(55e-10)
```

What to do with python code? I want to run it!

- Using **interpreter** to run the code (idea: translating the source code while running into machine code)
- Python > 3.6 sufficient for our course

Jupyter Notebook

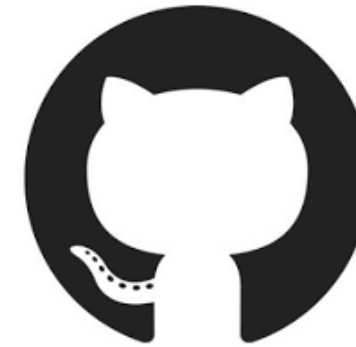
- Comes with [Anaconda | The World's Most Popular Data Science Platform](#)
 - *Jupyter* is a browser-based submodule of Anaconda
 - *Spyder* as IDE
 - ...
- Browser based frontend
- Step-by-step process
- Ideal for development
- Markdown to document
- Simple import from submodules



- No need for powerful computer
- No need to store large data sets etc.
- Easy to work on tablets/notebooks
- Using the computers from University

L01.4 Software Versioning (git)

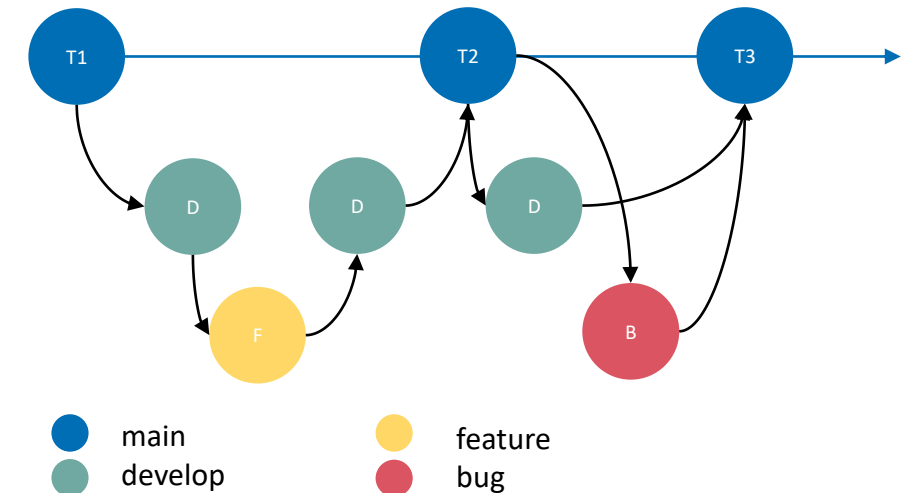
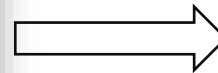
- To have control over existing software releases, version control is required
- Track changes and differentiate between multiple stages/versions
 - .gitconfig is used to exclude files, directions etc.
- Control different stages/branches of your software project
- Tagging of „final“ (release) versions
- [Git](#)Hub, GitLab, Gitea, Bitbucket, SVN, ...



```
Daniel.Schneider@IFMB56 MINGW64 ~/ADAS/adas (feature_error_model)
$ git status
On branch feature_error_model
Your branch is up to date with 'origin/develop'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   lkas_evaluation_dasense.ipynb
    new file:   paper_2021_lkas_evaluation.ipynb
    new file:   stationary_straight_driving.ipynb
    new file:   target_extraction.ipynb

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
  (commit or discard the untracked or modified content in submodules)
    modified:   behavior_model.ipynb
    modified:   cfg/settings.toml
    modified:   configuration (modified content)
    modified:   data_management (modified content, untracked content)
    modified:   data_types (modified content)
    modified:   data_visualization (modified content)
    modified:   doc/Architecture.drawio
```





Break

In the first practical session, we will:

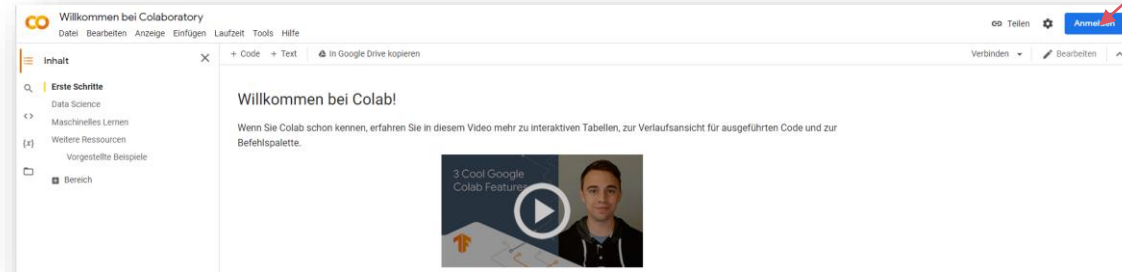
1. Access Google Colab
2. Run our first Python program
3. Check out existing repository in our Colab
4. Understand data types in Python, functions and libraries

E01.1 Google Colab

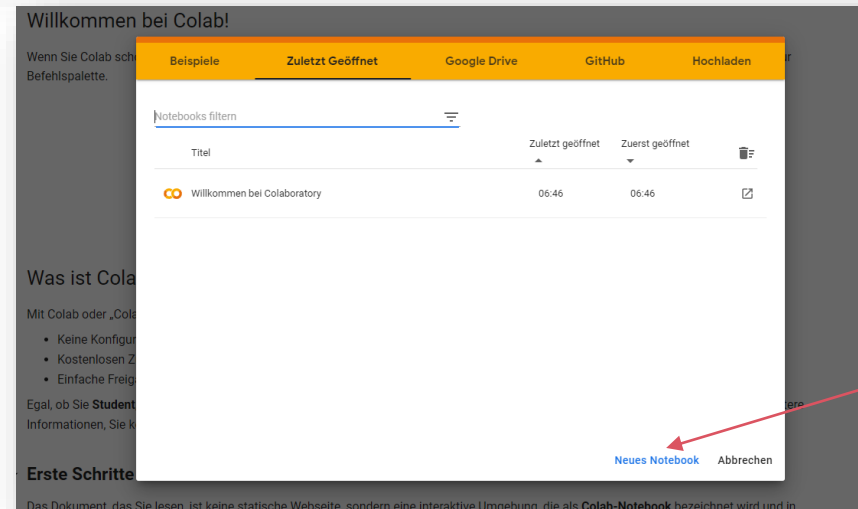
Getting started with our development environment

1. Access <https://colab.research.google.com/>
2. Sign in with Google account or create a new one for free

log/sign in



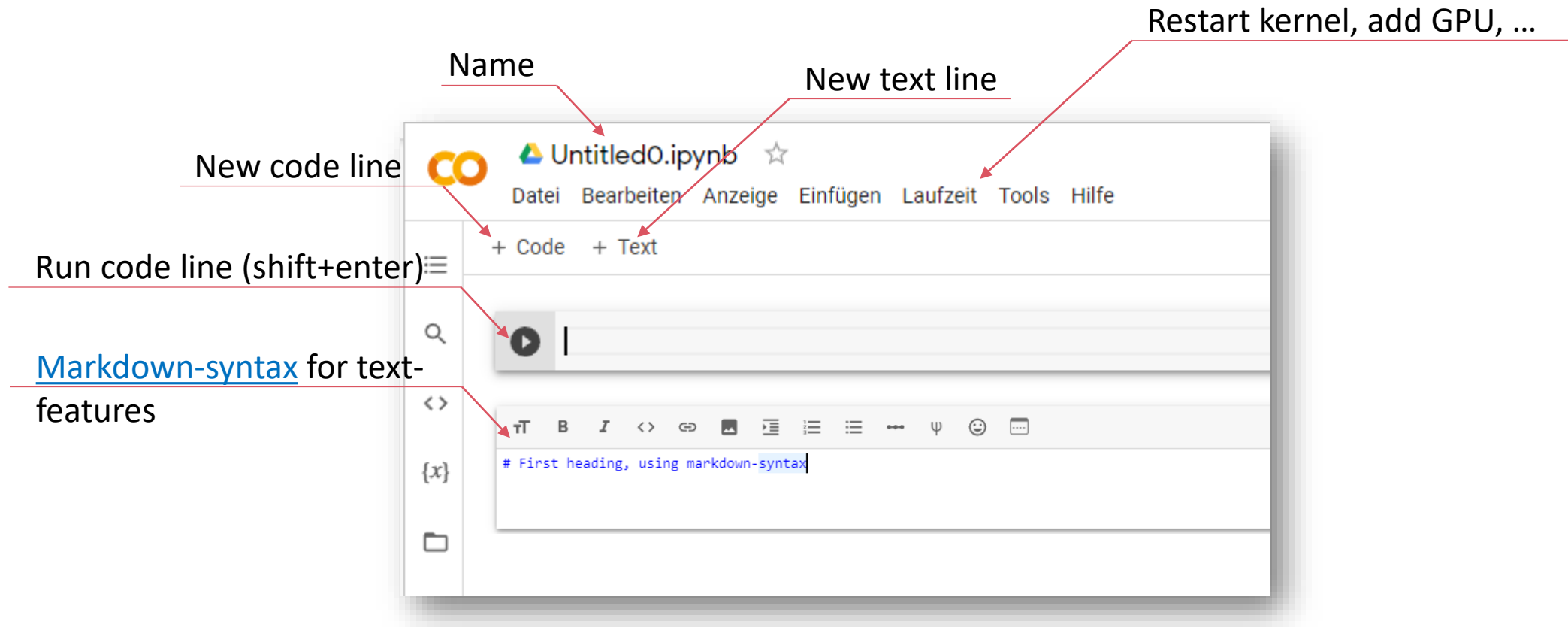
3. Create new notebook



New notebook

E01.1 Google Colab

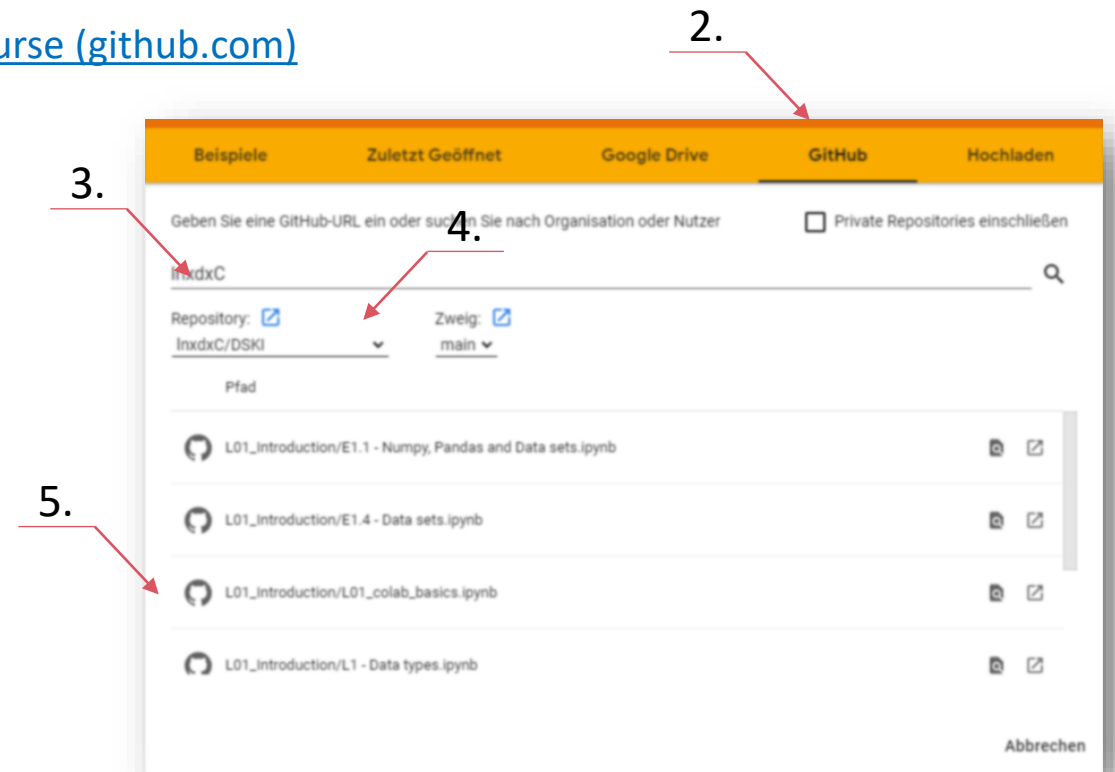
Once you have created a notebook, you see:



E01.1 Clone File from Repository in Colab

To check out an existing notebook (e. g. from GitHub) follow:

1. Ctrl+o to access the open-area
2. Select GitHub
3. Copy repo-link: [InxdxC/DSKI: Repository for the Data Science & KI course \(github.com\)](https://github.com/InxdxC/DSKI: Repository for the Data Science & KI course (github.com))
4. Select repository (DSKI) and branch (main)
5. Open notebook



HA01.1 Averaging

- Familiarize yourself with the Iris dataset and use the prepared script to load the data
- Use pandas and their DataFrame to work with, use the class names from the DataFrame, not from `iris.target_names`

Task 1:

- Since we have class IDs but no linkage between the class name, **associate the ID with the class names** for each class
- Use the field `df["Class Name"]`

Mapping:

$$C_{ID} \rightarrow C_{Name}$$

Task 2:

- Determine the **mean** of the sepal length **for each class** ($x \in \mathbb{R}^{1 \times n_{ID}}$) & store the result in a structure of your choice

Arithmetic mean:

$$\bar{x}_{ID} = \frac{1}{n_{ID}} \sum_{i=0}^{n_{ID}-1} x_{i,ID}$$

Hints

- Look deeper into the input data to get the classes by **key**
- Iterate over data or use list comprehensions and iterators
- Allocate the field `df["Class Name"]` before you assign values to it



www.hs-kempten.de/ifm