Data Science - Working with Data
Pre-processing, Explorative Data Analysis

After-Work-Drink? Wunschgetränk

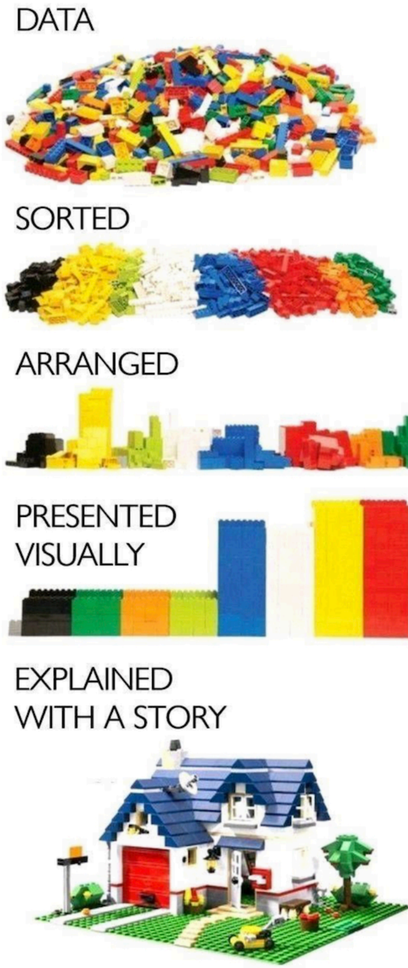
Data Science: But why?

[1]

[2]

Raw Data

Knowlege



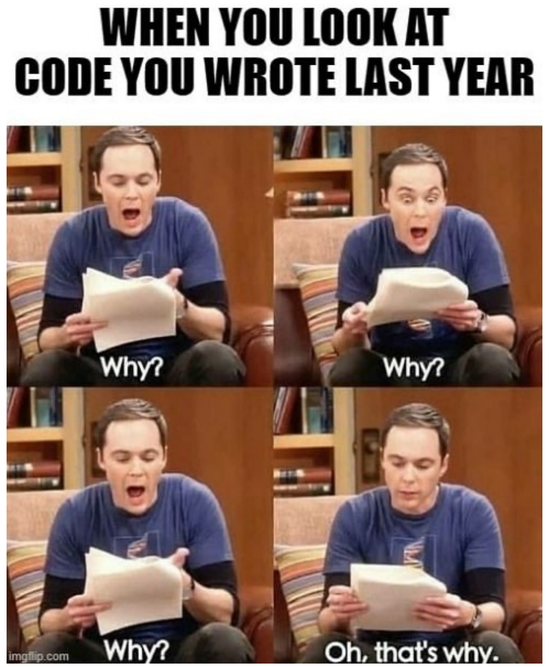
[1] https://www.google.com/url?sa=i&url=https%3A%2F%2Flustich.de%2Fbilder%2Fandere%2Fpapierberg%2F&psig=AOvVaw1shfkEZVN9kWASoXICbwAi&ust=1601110332932000&source=images&cd=vfe&ved=2ahUKEwiMxrL29oPsAhVUNewKHdqPD0MQr4kDegUIARDIAQ
[2] https://www.youtube.com/watch?v=n0ql-yeY9u0

Recap: A Code Dev...

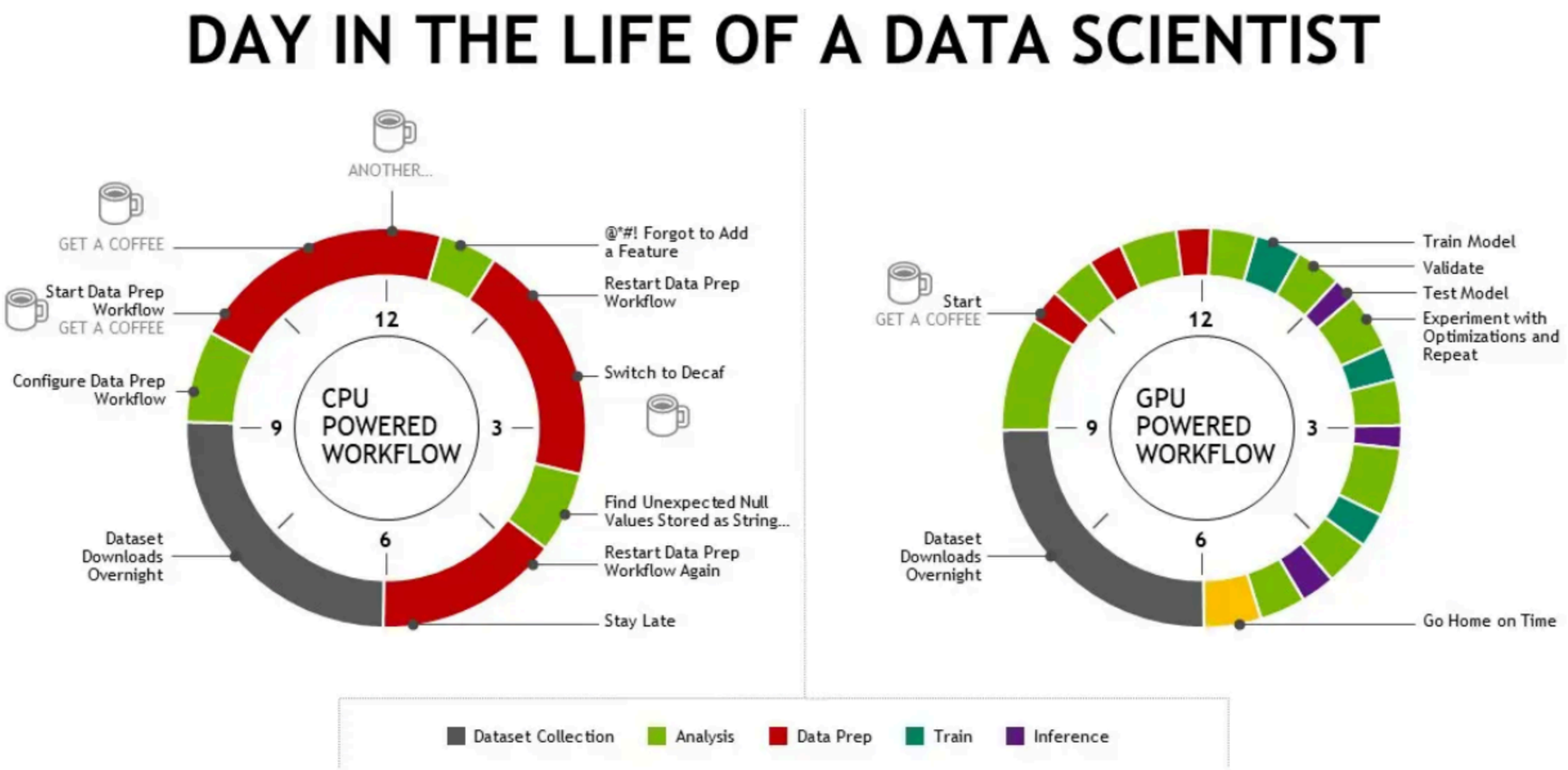
A good programmer **plans ahead, problem solves, builds a collection of useful resources**, and **creates modular systems** that **are easy to iterate upon and debug,** all while **focusing on optimization** and **potential automation…**

Why am I telling you this?

Because it ties into the **skills required to excel at Data Science Project!**



Pre-Processing: Critical Reflection



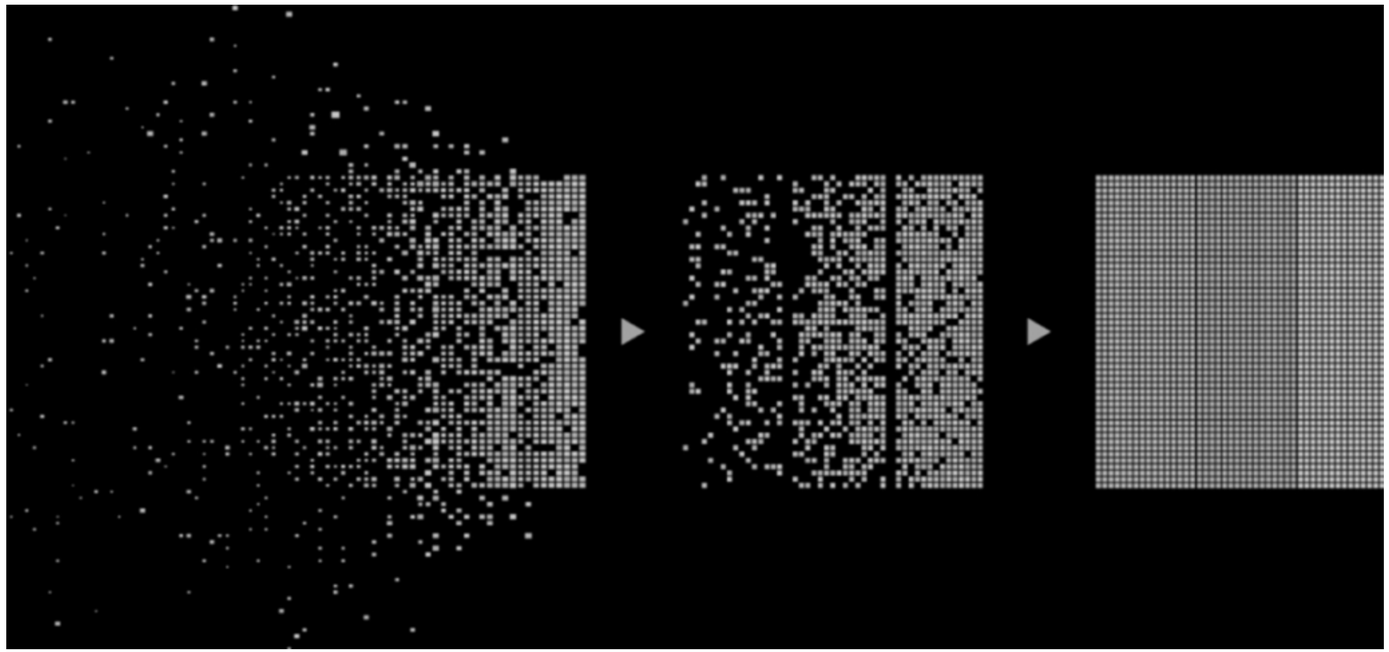
Therefore, it is important to realize that it is common practice to dedicate about 80% of the labor and time within the KDD process to data preparation, as a representative survey on a popular data science portal shows. https://bit.ly/2WwVPho.

In: Abdel-Karim, B. M., Pfeuffer, N., & Hinz, O. (2021). Machine learning in information systems-a bibliographic review and open research issues. Electronic Markets, 31(3), 643-670

https://blogs.nvidia.com/blog/2018/11/15/accelerated-data-science-hpc/

Pre-Processing: What is That?

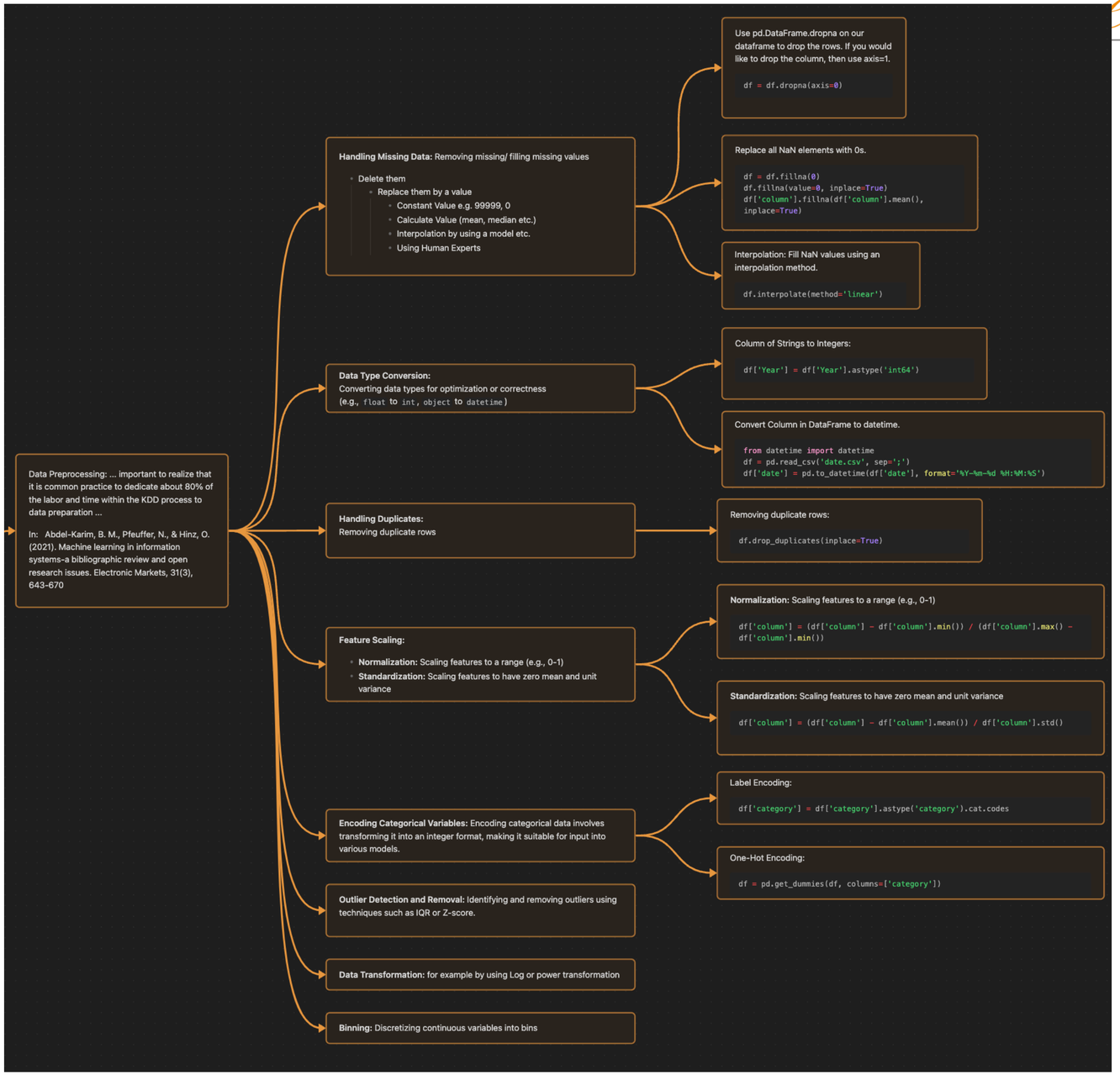
- Data preprocessing is a major and essential stage whose main goal is to obtain final data sets which can be considered correct and useful for further data mining algorithms [1].
- The idea is to filter the data according to quality criteria, selected according to possible target values, and transformed correspondingly [2] .



Raw Data Pre-Processing Clean Data

[1] García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. Big Data Analytics, 1(1), 1-22.
[2] Abdel-Karim, B. M., Pfeuffer, N., & Hinz, O. (2021). Machine learning in information systems-a bibliographic review and open research issues. Electronic Markets, 31, 643-670.

- **Handling Missing Data**
  - Removing missing values
  - Filling missing values
- **Data Type Conversion**
- **Handling Duplicates**
- **Feature Scaling**
  - Normalization
  - Standardization
- **Encoding Categorical Variables**
  - Label Encoding
  - One-Hot Encoding
- **Feature Engineering: Create new ones**
- **Outlier Detection and Removal**
- **Data Transformation**
- **Binning**
- ...



Alasadi, S. A., & Bhaya, W. S. (2017). Review of data preprocessing techniques in data mining. Journal of Engineering and Applied Sciences, 12(16), 4102-4107.
Fan, C., Chen, M., Wang, X., Wang, J., & Huang, B. (2021). A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. Frontiers in energy research, 9, 652801.
García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. Big data analytics, 1, 1-22.

Handling Missing Data: Removing

```python
import pandas as pd
import numpy as np

# Sample DataFrame with missing values
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [55, np.nan, 35, 45, np.nan],
    'City': ['London', 'Los Angeles', np.nan, 'Chicago', 'New York']
}

df = pd.DataFrame(data)
print(df)
```

	Name	Age	City
0	Alice	55.0	London
1	Bob	NaN	Los Angeles
2	Charlie	35.0	NaN
3	David	45.0	Chicago
4	Eve	NaN	New York

`df_dropped_rows = df.dropna()` Deleting:

Remove columns with any missing values:

`df_dropped_columns = df.dropna(axis=1)` Deleting:

Handling Missing Data: Fill Values

```python
import pandas as pd
import numpy as np

# Sample DataFrame with missing values
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [55, np.nan, 35, 45, np.nan],
    'City': ['London', 'Los Angeles', np.nan, 'Chicago', 'New York']
}

df = pd.DataFrame(data)
print(df)
```

	Name	Age	City
0	Alice	55.0	London
1	Bob	NaN	Los Angeles
2	Charlie	35.0	NaN
3	David	45.0	Chicago
4	Eve	NaN	New York

Fill with a specific value:

```python
df_filled = df.fillna(0)
```

NaN => 0

Forward fill (propagate last valid observation forward):

```python
df_filled_forward = df.fillna(method='ffill')
```

Bob, **55.0**, Los Angeles
Charlie, 35.0, **Los Angeles**
Eve, **45.0**, New York

Backward fill (propagate next valid observation backward):

```python
df_filled_backward = df.fillna(method='bfill')
```

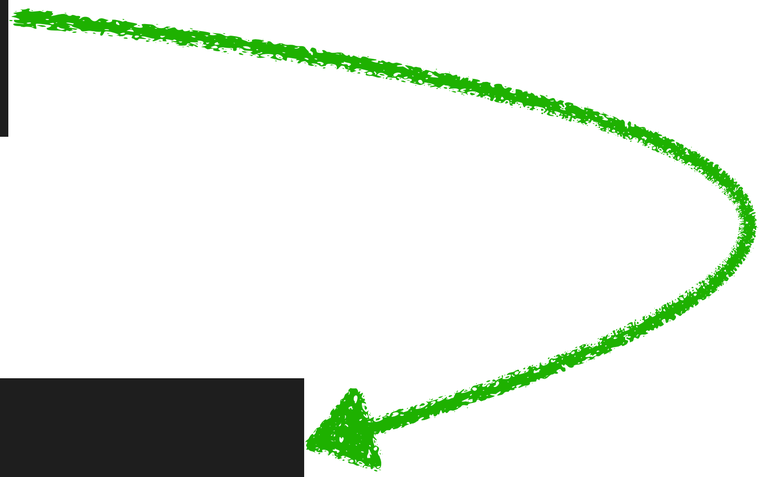
Bob, **35.0,** Los Angeles
Charlie, 35.0, **Chicago**
Eve, **NaN**, New York

Careful

Note: Interpolate missing data, which is especially useful for time series:

Data Type Conversion (Data Types of Each Column)

```
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ID        4 non-null      object
 1   Price     4 non-null      object
 2   Date      4 non-null      object
 3   Category  4 non-null      int64
dtypes: int64(1), object(3)
memory usage: 260.0+ bytes
```



```
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ID        4 non-null      int64
 1   Price     4 non-null      float64
 2   Date      4 non-null      datetime64[ns]
 3   Category  4 non-null      category
dtypes: category(1), datetime64[ns](1), float64(1), int64(1)
memory usage: 356.0 bytes
None
```

```python
data = {
    'ID': ['1', '2', '3', '4'],  # IDs stored as strings
    'Price': ['99.5', '20.99', '15.75', '40'],  # Prices stored as strings
    'Date': ['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04'],  # Dates stored as strings
    'Category': [1, 2, 1, 2]  # Categories stored as integers
}

df = pd.DataFrame(data)
print("Original DataFrame:\n", df)

print(df.info())

# Convert ID from string to integer:
df['ID'] = df['ID'].astype(int)
print("\nDataFrame after converting ID to integer:\n", df)

# Convert Price from string to float:
df['Price'] = df['Price'].astype(float)
print("\nDataFrame after converting Price to float:\n", df)

# Convert Date from string to datetime:
df['Date'] = pd.to_datetime(df['Date'])
print("\nDataFrame after converting Date to datetime:\n", df)

# Convert Category from integer to category:
df['Category'] = df['Category'].astype('category')
print("\nDataFrame after converting Category to categorical:\n", df)

print(df.info())
```

Data Type Conversion (Data Types of Each Column)

Correct Data Representation:

- Data types ensure that each column in a DataFrame is stored in the most appropriate format. For example:
- Numerical Data: Converting columns to int or float ensures that numerical operations (like addition or averaging) are performed correctly.
- Categorical Data: Converting columns to category type can save memory and speed up operations if the column has a limited number of unique values.
- Date/Time Data: Converting columns to datetime type allows for time-based operations and manipulations.

Performance Optimization:

- Correct data types can lead to more efficient storage and faster computation. For instance:
- Memory Efficiency: Using category for columns with a limited number of unique values reduces memory usage compared to using object (which is typically a catch-all for string data).
- Speed: Numerical operations are faster with int or float types compared to object types, which are more general-purpose and less optimized for numeric computations.

Handling Duplicates

Why handling duplicates could be important:

- **Data Integrity:** Duplicates can compromise the accuracy and integrity of the data.
- **Accurate Analysis:** impact on statistical analysis and metrics.
- **Efficient Ressourcen Usage**

Option to Handle:

- **Removing Duplicates:** Remove duplicate rows based on all columns or specific ones.
- **Identifying Duplicates:** Find duplicate rows before deciding how to handle them.

Ideas:

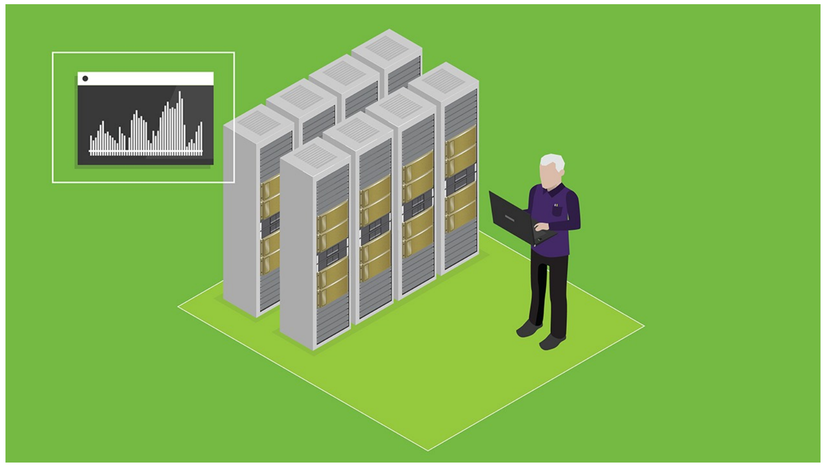
- Delete them
- Replace them by
- Constant Value e.g. 99999, 0
- Calculate Value (mean, median etc.)
- Interpolation by using a model etc.
- Using Human Experts

The Matrix: Agent Smith's Replicating

https://www.cbr.com/the-matrix-agent-smith-replicating-powers-mysteries-explained/

Pre-Processing: Challenge Workflow

„Local“ Development



Server Deployment

Use in Production

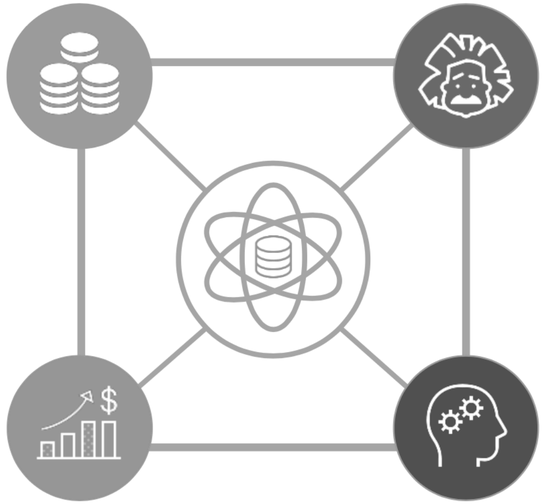
From Data to Machine Learning … Easy?

Database Management	Machine Learning
Database is an active, evolving entity	Database is just a static collection of data
Records may contain missing or erroneous information	Instances are usually complete and noise-free
A typical field is numeric	A typical feature is binary
A database typically contains millions of records	Data sets typically contain several hundred instances
AI should get down to reality	"Databases" is a solved problem and is therefore uninteresting

Table 2. Conflicting Viewpoints between Database Management and Machine Learning. [1]

Needs to shape the data into the right form

Data Engineer
- → Responsible for the acquisition, organization and storage of data
- → Possible skills: Software Engineering, SQL, Hadoop, Python(Pandas)



General Data Scientist
- → Coding Generalist who extracts knowledge from data and creates business value with the full data science stack
- → Possible Skills: Domain Knowledge, Statistics, Python/R, Pandas/Numpy, Machine Learning, Data Visualization, etc.

Data Analyst
- → Responsible for the creation of business value with data
- → Possible Skills: Domain Knowledge, Python/R, SQL, Excel, Data Visualization

Machine Learning Engineer
- → Specialist in the field of Machine Learning; Creator of Algorithms
- → Possible Skills: Mathematics/Statistics, Software Engineering, Python, C++, Pandas/Numpy/Pytorch, Machine Learning, Neural Networks, etc.

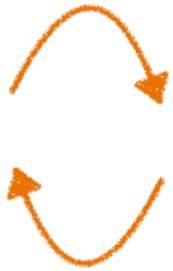
[1] Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge discovery in databases: An overview. AI magazine, 13(3), 57-57.

Domain Knowledge: Why this is Important?

Domain knowledge can be define as knowledge about the environment in which the target implementation operates [1].

Data Science Team

Data Scientist/ Data Analyst



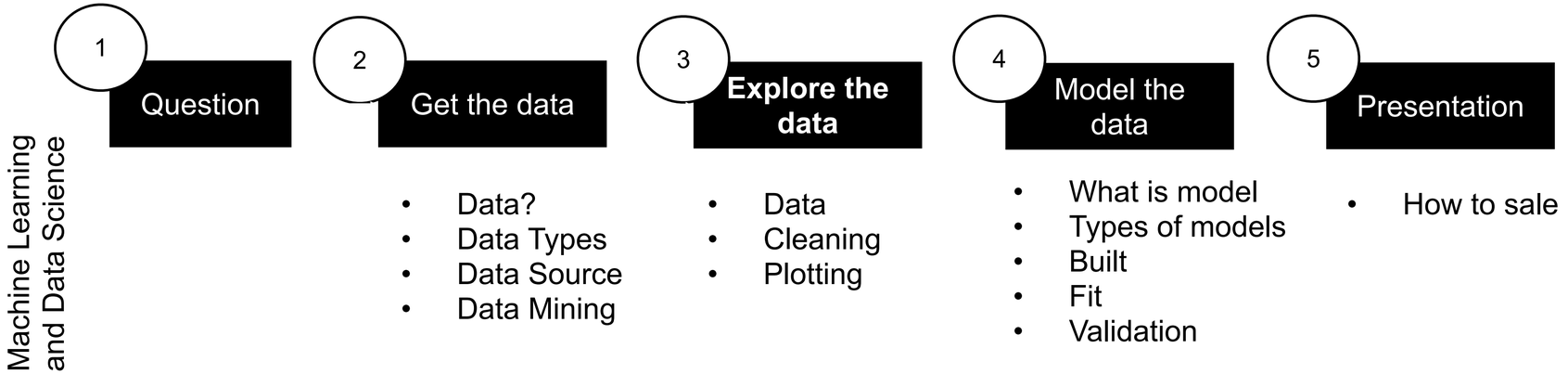
Need each other

Business side

[1] Hjørland, B. & Albrechtsen, H. (1995). Toward A New Horizon in Information Science: Domain Analysis. Journal of the American Society for Information Science, 1995, 46(6), p. 400–425.

Images:
https://www.theforage.com/blog/careers/investment-banking
https://unsplash.com/de/s/fotos/Geschäft
https://stock.adobe.com/de/images/nerd-at-the-computer-side-view-of-young-nerd-man-working-at-the/55780370

Concept: Data Science Process



KDD Model: Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI magazine*, *17*(3), 37-37.

Note: The KDD framework is not the only approach for executing data science projects professionally. Other widely used frameworks include CRISP-DM (Wirth & Hipp, 2000), which emphasizes business understanding, data preparation, modeling, evaluation, and deployment, and Agile Data Science / Agile ML (Saltz & Hotz, 2019), which focuses on iterative development, rapid prototyping, and continuous integration. KDD was selected here due to its practical efficiency and clear stepwise structure for knowledge discovery.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From Data Mining to Knowledge Discovery in Databases. AI Magazine, 17(3), 37–54.
Wirth, R., & Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining.
Saltz, J., & Hotz, N. (2019). Adopting agile principles for data science project success. International Journal of Information Management, 49, 34–42.

Case Study

What is Steam

• […] one of the most popular digital game delivery platform (Lin et. al., 2019, p. 1).

• Steam, one of the largest gaming networks for computer games in the world (Baumann et. Al., 2018, p. 1).

Baumann, F., Emmert, D., Baumgartl, H., & Buettner, R. (2018). Hardcore Gamer Profiling: Results from an unsupervised learning approach to playing behavior on the Steam platform. Procedia Computer Science, 126, 1289-1297.
Lin, D., Bezemer, C. P., Zou, Y., & Ha

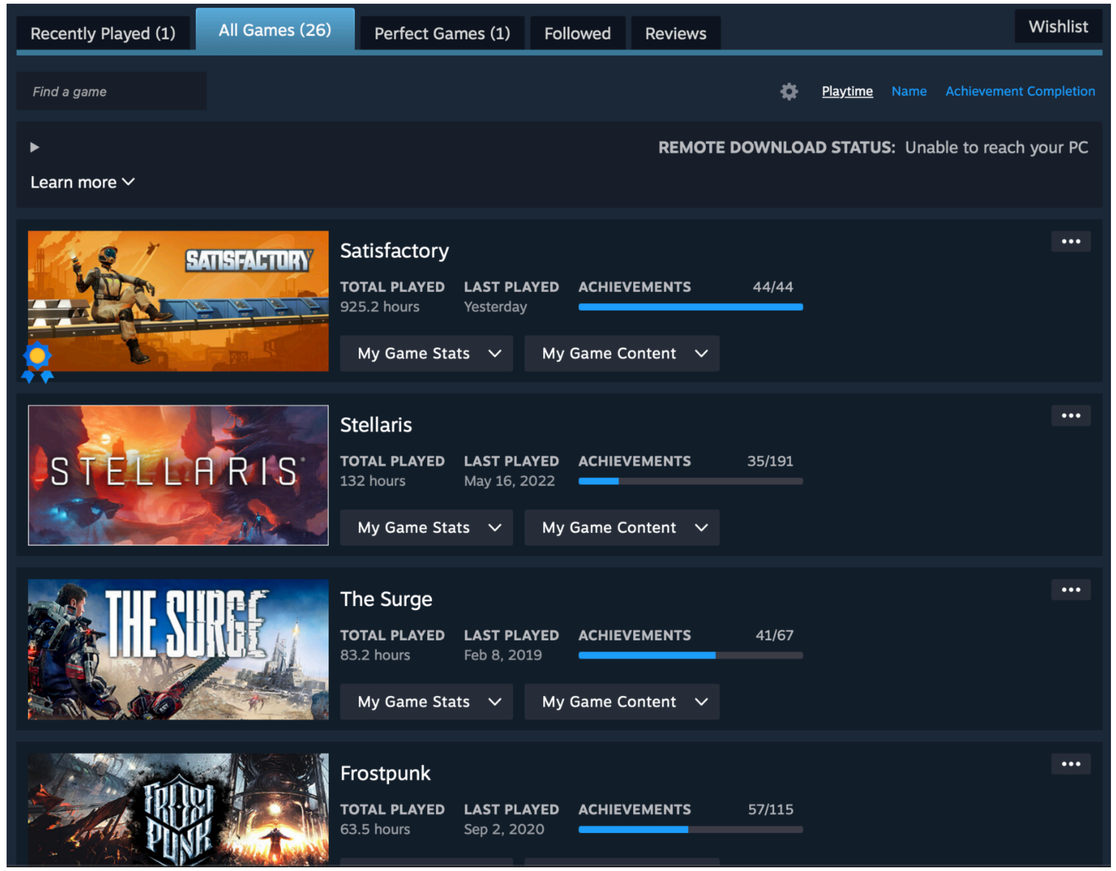


Figure: Steam Licensed Games

Data Sets: High Level View

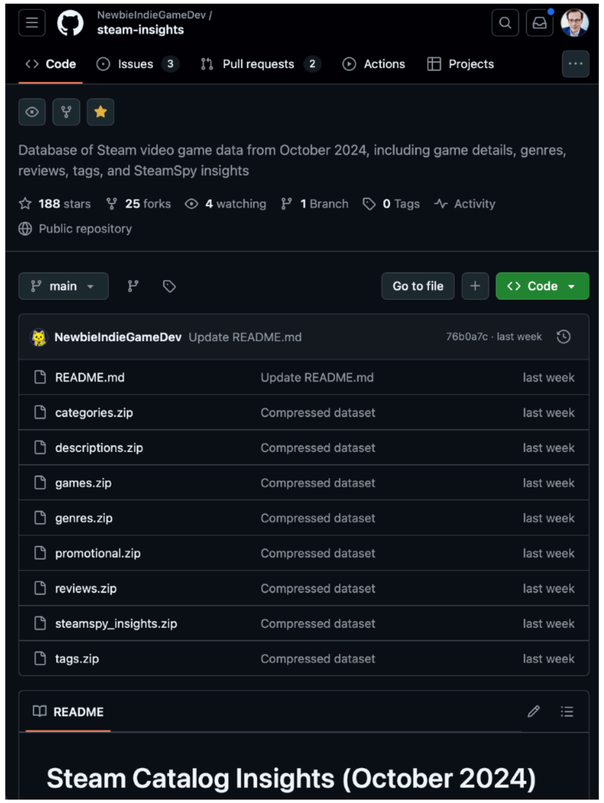


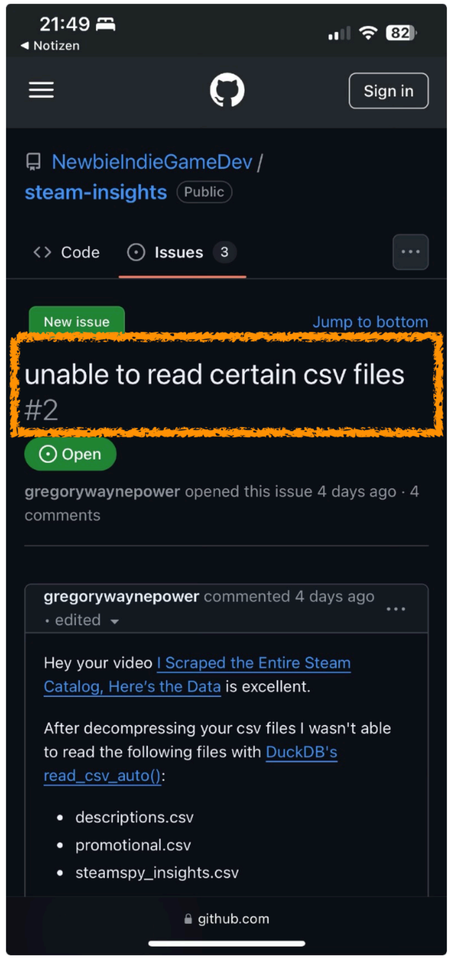
Figure: Git Repo for the Raw Data

https://github.com/NewbieIndieGameDev/steam-insights

Figure: Corresponding YouTube Video

https://www.youtube.com/watch?v=qiNv3qv-YbU

Limitation: Data Set is not Perfect
Do you have any idea why we see this kind of raw data in the .csv?



Pseudo „Jason“/ Dictionary stored in the excel!

games.csv

games.csv

```
"app_id","name","release_date","is_free","price_overview","languages","type"
"10","Counter-Strike","2000-11-01","0","{\"final\": 819, \"initial\": 819, \"currency\": \"EUR\", \"final_formatted\": \"8,19€\", \"discount_percent\": 0, \"initial_formatted\": \"\"}","English<strong>*</strong>, French<strong>*</strong>, German<strong>*</strong>, Italian<strong>*</strong>, Spanish - Spain<strong>*</strong>, Simplified Chinese<strong>*</strong>, Traditional Chinese<strong>*</strong>, Korean<strong>*</strong><br><strong>*</strong>languages with full audio support","game"
```

HTML Code stored as well

Task and Context

Task

Supporting an M&A Initiative in the Gaming Sector by Using Real World Data!

Context

Under Phil Spencer’s leadership, Microsoft shifted its Xbox strategy to focus on content and ecosystem strength rather than just console sales. To boost market share and compete with rivals like Sony and Nintendo, Microsoft began acquiring major game studios and publishers. From Mojang (Minecraft) to ZeniMax (Bethesda) and eventually Activision Blizzard. These acquisitions aimed to secure exclusive content, expand the Xbox Game Pass library, and position Xbox as a leading platform for both console and cloud gaming. But success is missing...

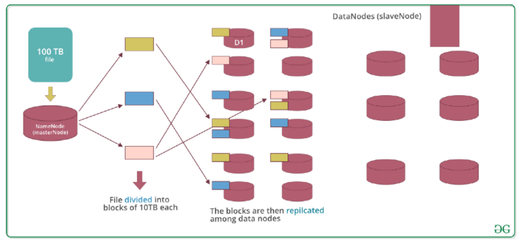
Lets have a Look into the Data!

But why?
Computing Power (Moore‘s Law), e.g., Lundstrom (2003)
Economies of Scale (Wrights’s Law), e.g., Stringham et al. (2015)
Non-linear (exponential) increase in knowledge (Kurzweil‘s Law), e.g., Cyranoski et al. (2011)

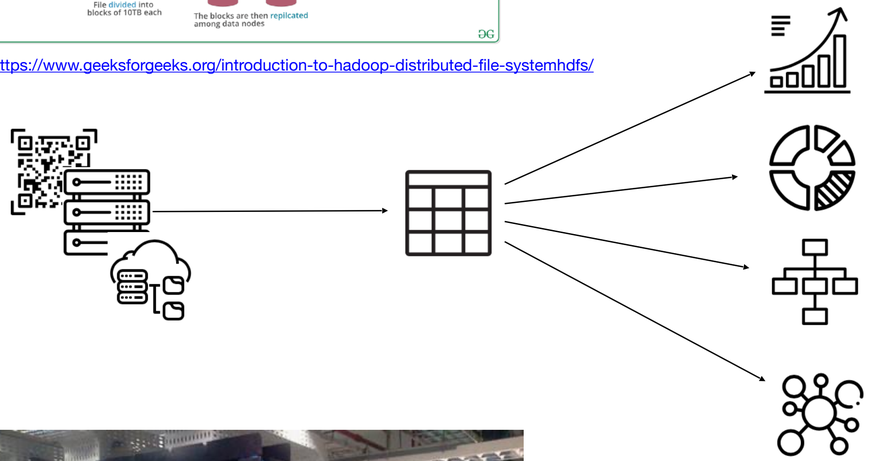
Lundstrom, M. (2003). Moore’s Law Forever? Science, 299(5604), 210–211.
Cyranoski, D., Gilbert, N., Ledford, H., Nayar, A., & Yahia, M. (2011). Education: The PhD Factory. Nature, 472(1), 276–279.
Stringham, E. P., Miller, J. K., & Clark, J. (2015). Overcoming Barriers to Entry in an Established Industry: Tesla Motors. California Management Review, 57(4), 85–103.

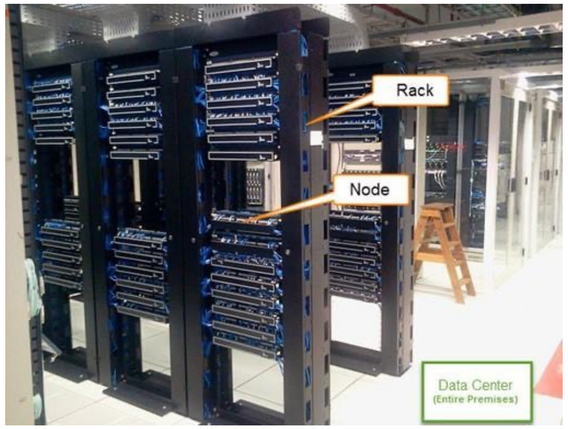
Out Look

Pre-Processing: Challenge Data



https://www.geeksforgeeks.org/introduction-to-hadoop-distributed-file-systemhdfs/





https://www.guru99.com/learn-hadoop-in-10-minutes.html

Data Mesh

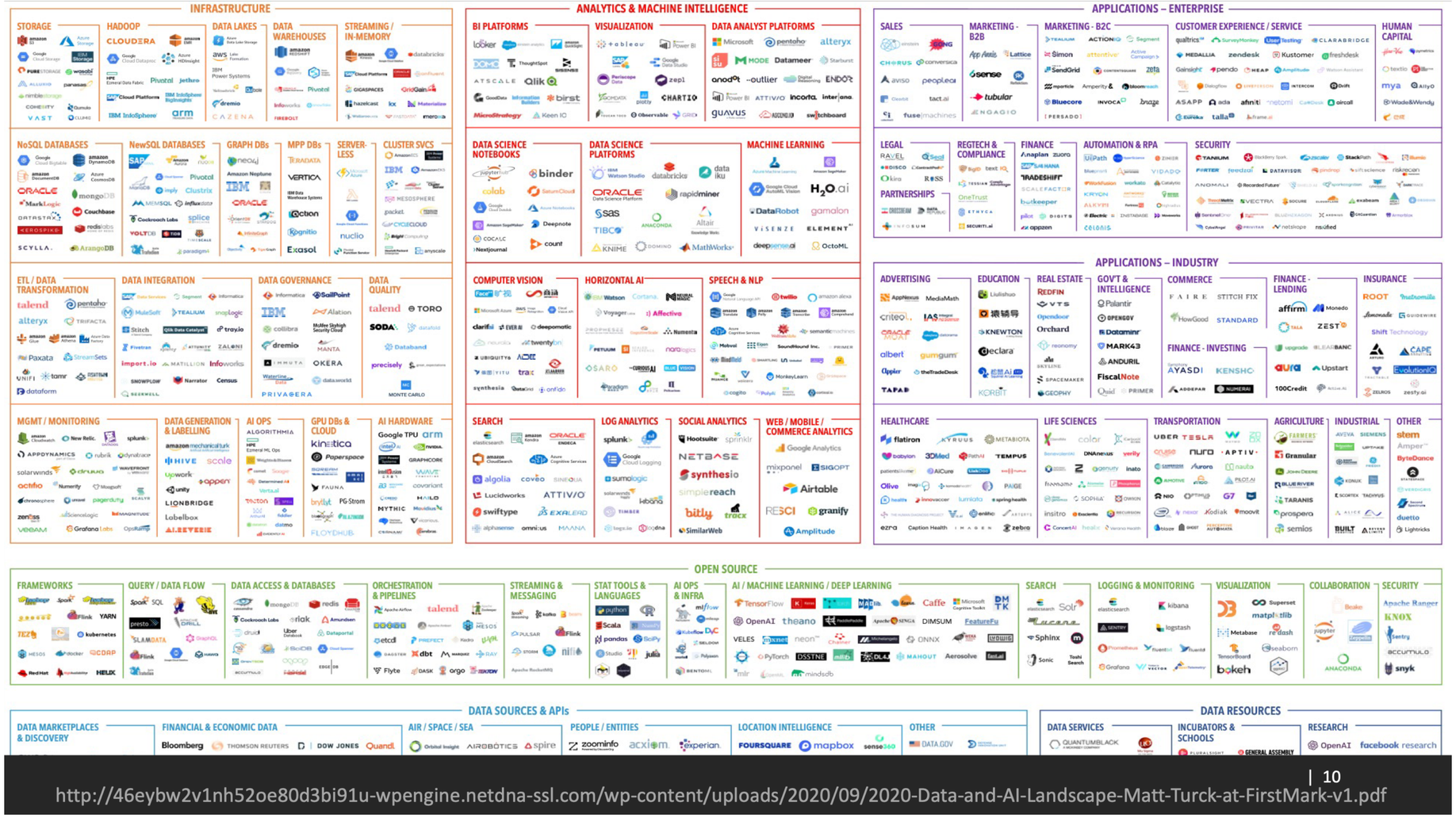
Cloud Data

Data Warehouse

Data Mining

Local Data

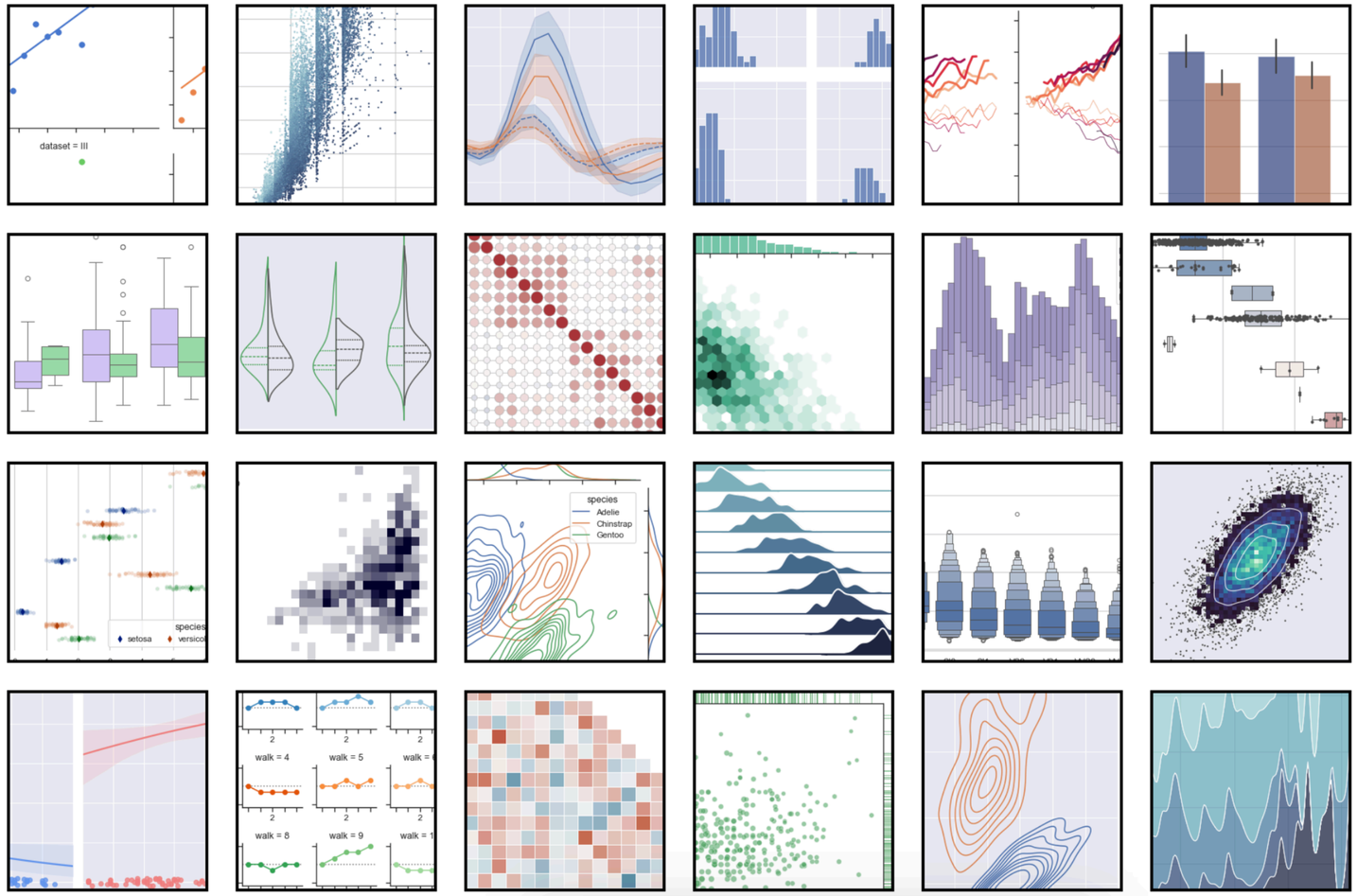
Pre-Processing: Interface Challenge to Application



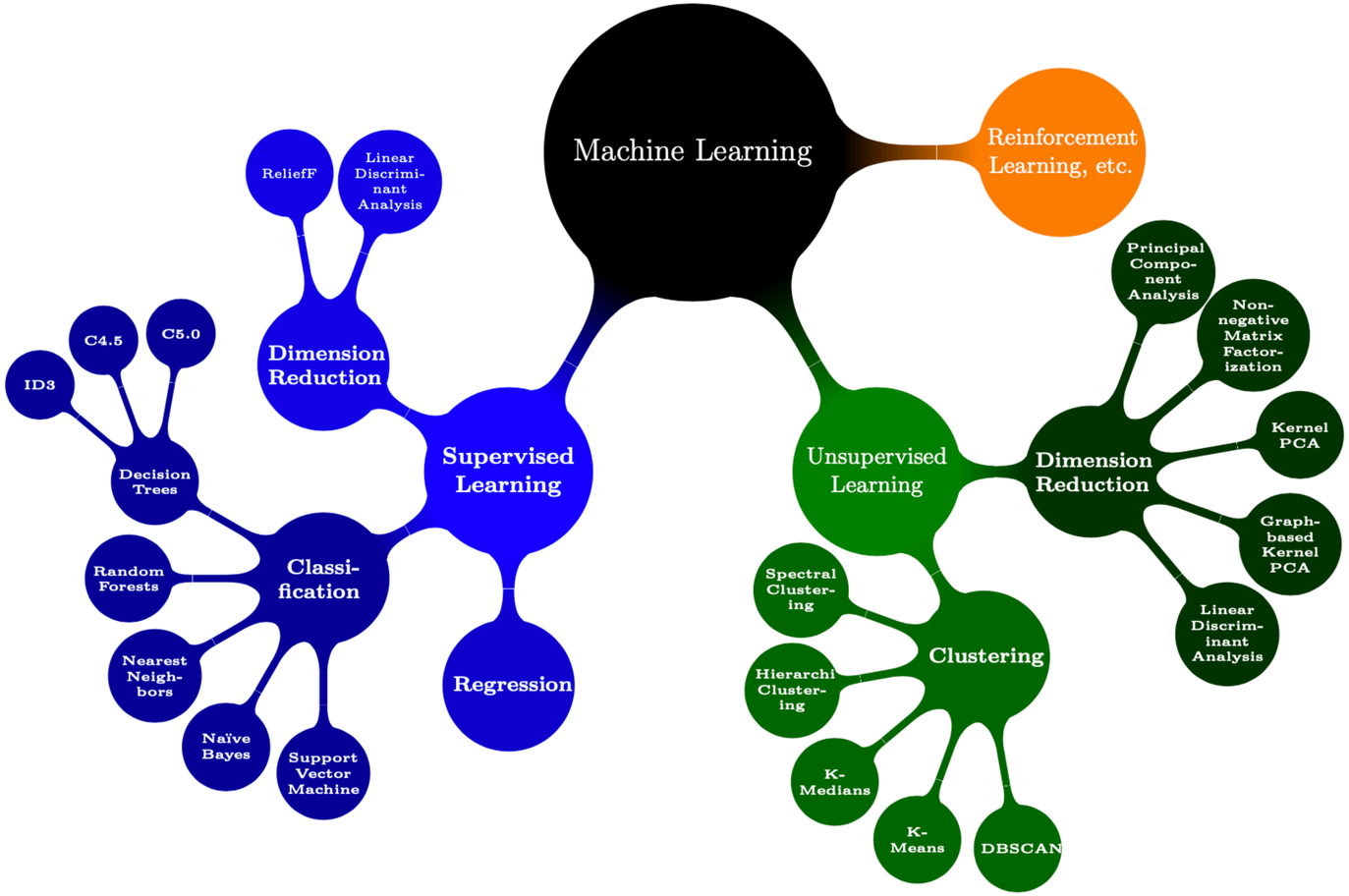
http://46eybw2v1nh52oe80d3bi91u-wpengine.netdna-ssl.com/wp-content/uploads/2020/09/2020-Data-and-AI-Landscape-Matt-Turck-at-FirstMark-v1.pdf

Data Visualization: Seaborn

- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. https://seaborn.pydata.org/
- An extension of the matplotlib is Seaborn.
- This library can operate directly with data frame objects. For data
science, a useful tool!
- It provides a high-level interface for drawing attractive and informative statistical graphics.
- Easy to use with DataFrames



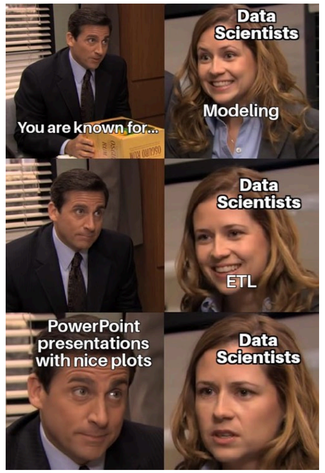
Models



A Schematic Illustration of the Taxonomy and Example Algorithms in Machine Learning in Accordance With the Works of Russell & Norvig (2016) and Bishop (2006)

Roweis, S. T., & Saul, L. K. (2000). Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, 290(5500), 2323–2326.

Data Science in Practice



Tip: Work Iterative

Get the Data → **Explore the data** → **Model the data** → **Presentation**

Rebalancing → **Feature Selection** → **Scaling** → **Checking Model with 10 K**

Downsizing

Using Correlation
Using Shape Values

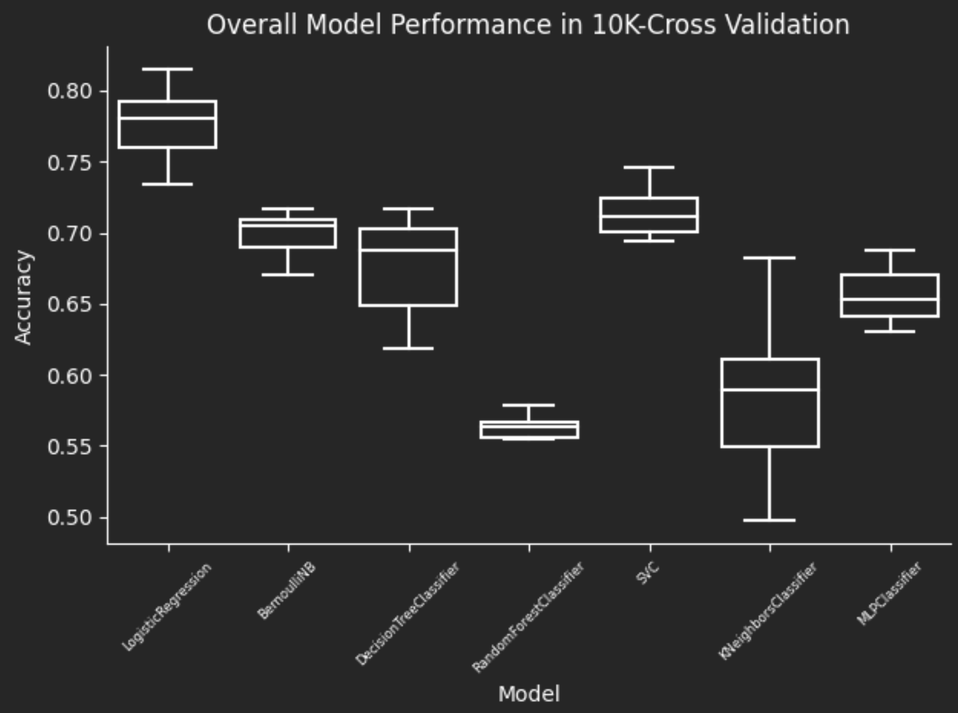
Using Min/Max

	precision
no_bankrupt	0.61
bankrupt	0.64

	precision
no_bankrupt	0.60
bankrupt	0.70

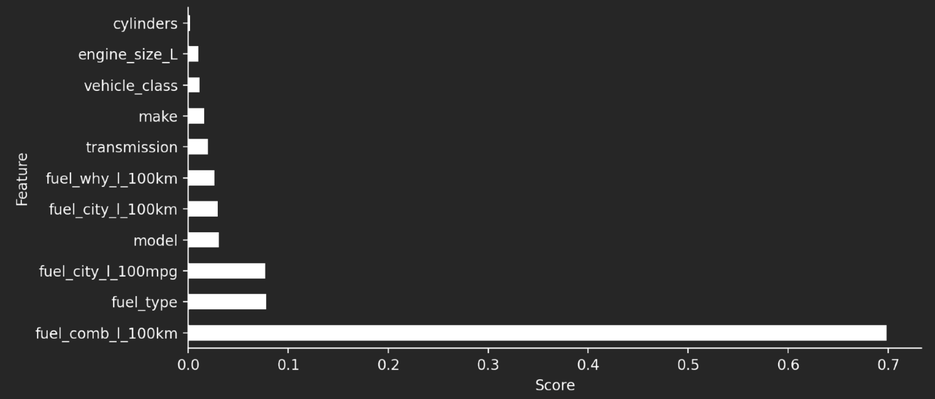
	precision
no_bankrupt	0.62
bankrupt	0.81

Improvements by using different data pre-processing techniques.



Data Optimization

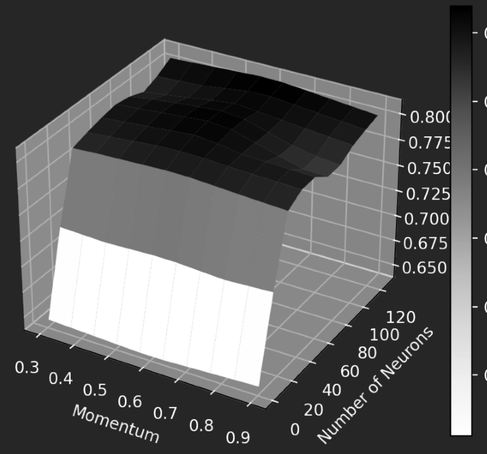
Model Optimization



More Features != Better Performance

Scaling Feature

Starting



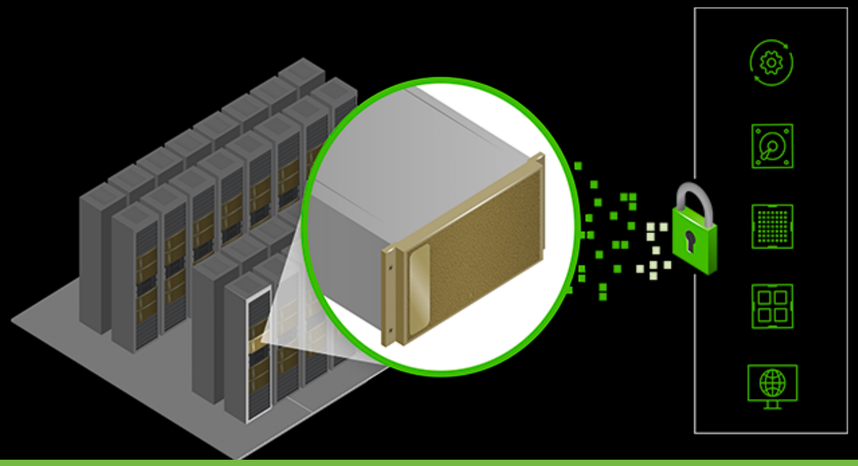
1

2

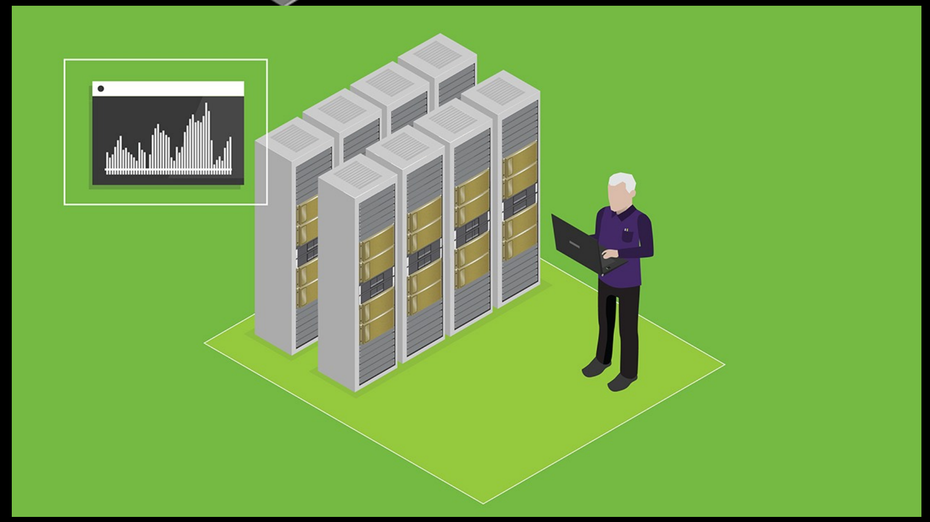
Steps

Research protocol based on Python 3.10 for ML Optimizationunder a MacBook Pro (13", 2018), 2,7 GHz Quad-Core Intel Core i7, 8 GB RAM.

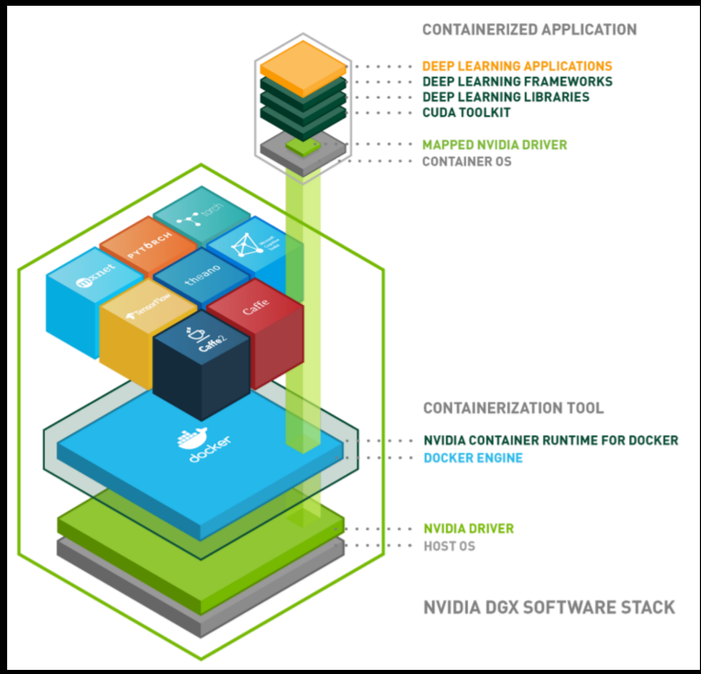
Approach



„Data“ Scientist writing code locally



Executing code on DGX Station via deployed with Container (e.g., **Docker**)!



Code and (the rest) is running

Bilder von Nividia https://www.google.com/imgres?imgurl=x-raw-image%3A%2F%2F9c6b15d4b89aab119a5ff1f75089aa8bf7c7bab17d11c8cbed53f57769bd7349&imgrefurl=https%3A%2F%2Fwww.nvidia.cn%2Fcontent%2Fdam%2Fen-zz%2FSolutions%2FData-Center%2Fdgx-station%2Fdgx-station-print-explorer-datasheet-letter-final-web.pdf&tbnid=EauauNQVa5w1mM&vet=10CAkQxiAoCmoXChMIyO6LoprU9AIVAAAAAB0AAAAAEA0..i&docid=gskwpG4XnBhBYM&w=499&h=473&itg=1&q=DGX%20station%20grafikkarten&client=safari&ved=0CAkQxiAoCmoXChMIyO6LoprU9AIVAAAAAB0AAAAAEA0