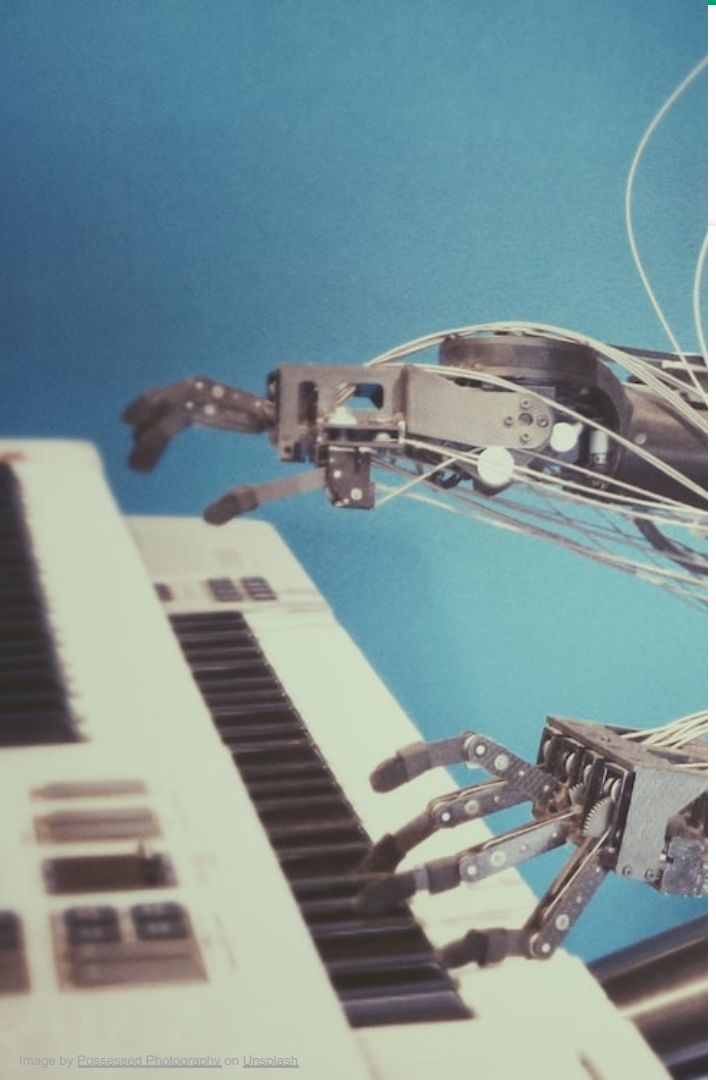


Applied Artificial Intelligence
04 - AI Lifecycle: Deployment

Univ.-Prof. Dr-Ing. habil. Niklas Kühl
www.niklas.xyz

University of Bayreuth

Karlsruhe Institute of Technology

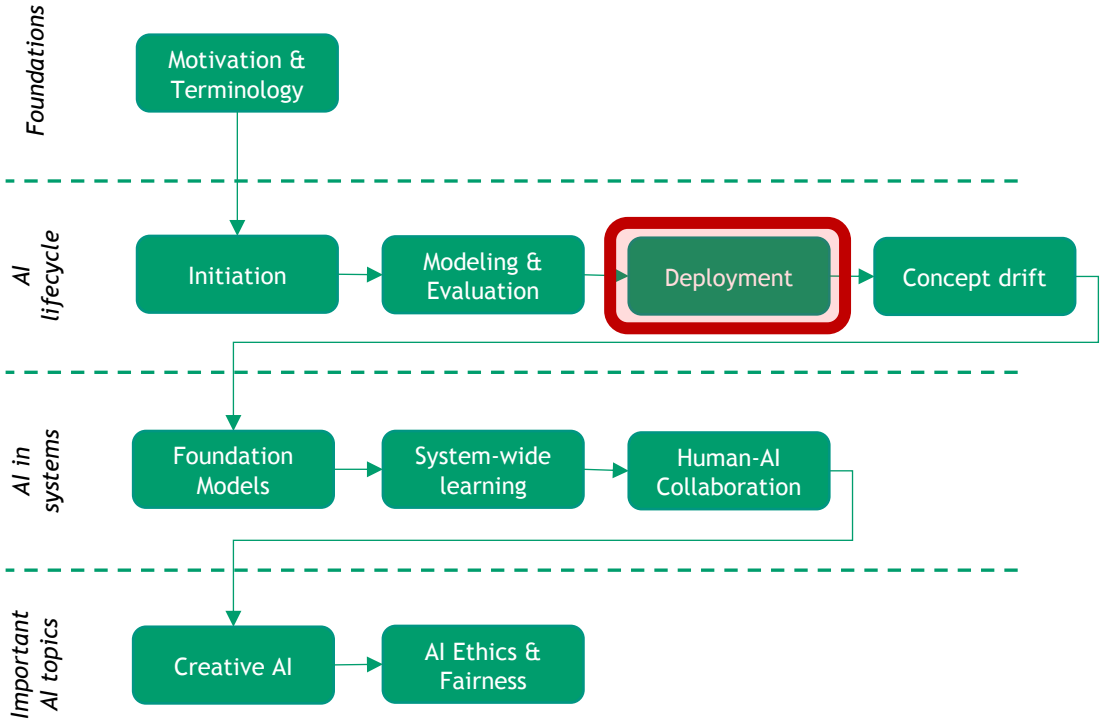
TUM School of Management

www.uni-bayreuth.de | www.kit.edu | www.tum.de | www.fim-rc.de | www.wirtschaftsinformatik.fraunhofer.de

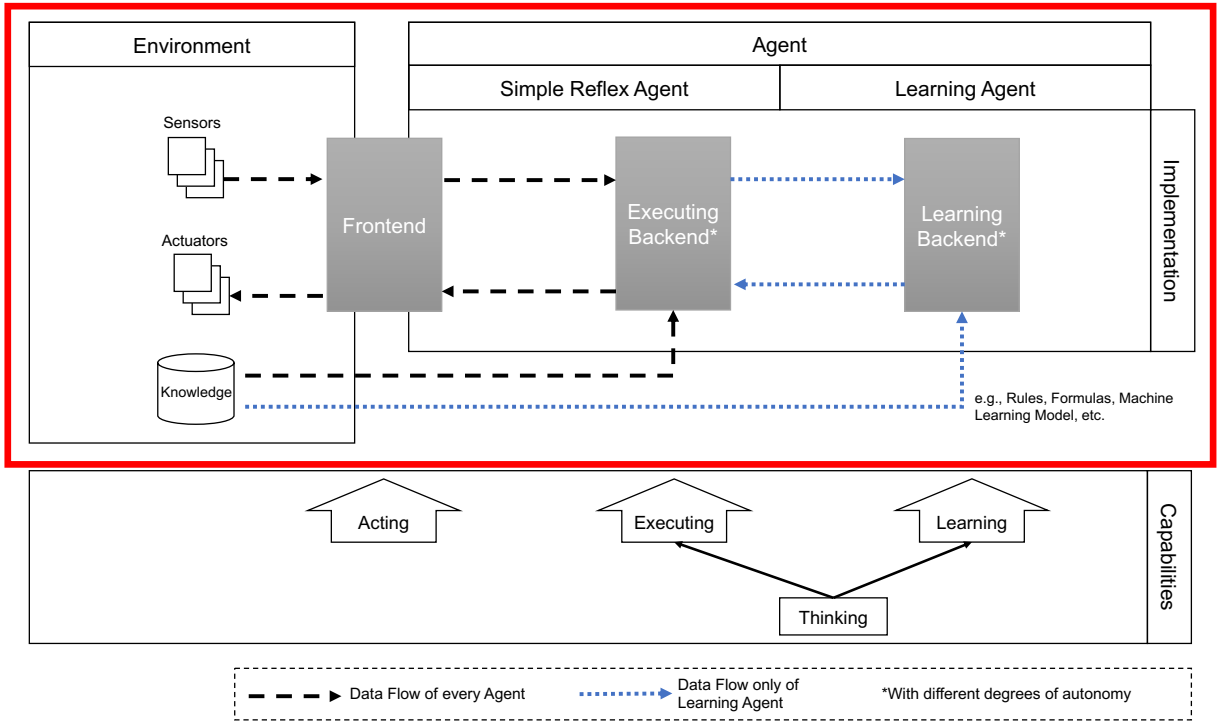
Repetition

Capability of data & algorithm	There are multiple concepts for supervised machine learning. The performance of each algorithm is determined by the **data and** the **algorithms' capabilities** to capture patterns in that very data.
Model optimization	**Models** can be **optimized** by tuning the **parameters** of the underlying algorithm.
Performance selection	The **selection** of the performance metric sets the **aim** for the model.
Interplay of components	It is important to understand the **interplay** of **problem statement**, **data (distribution)**, **algorithm** and **performance metric**.

Organizational
The story of the lecture



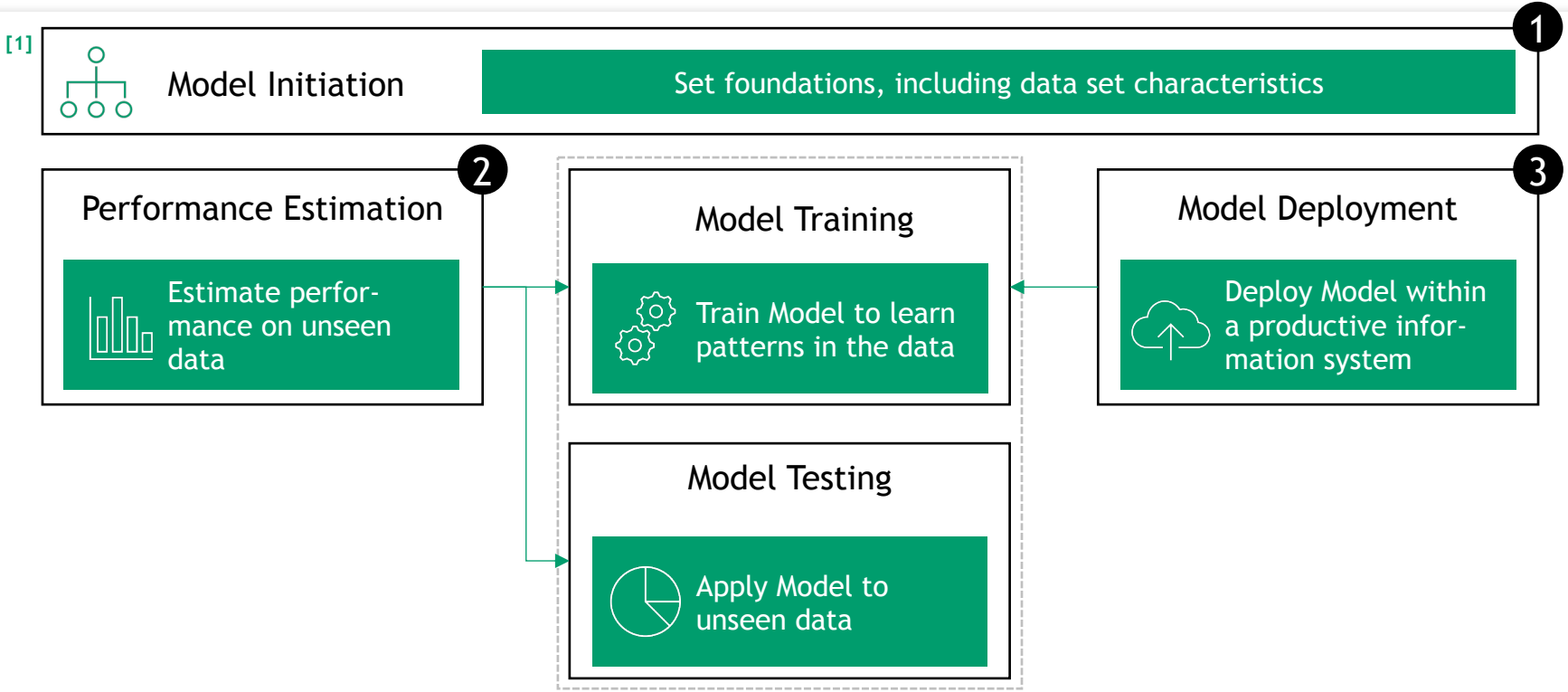
Positioning of the “deployment” step within an intelligent agent’s architecture



[1]

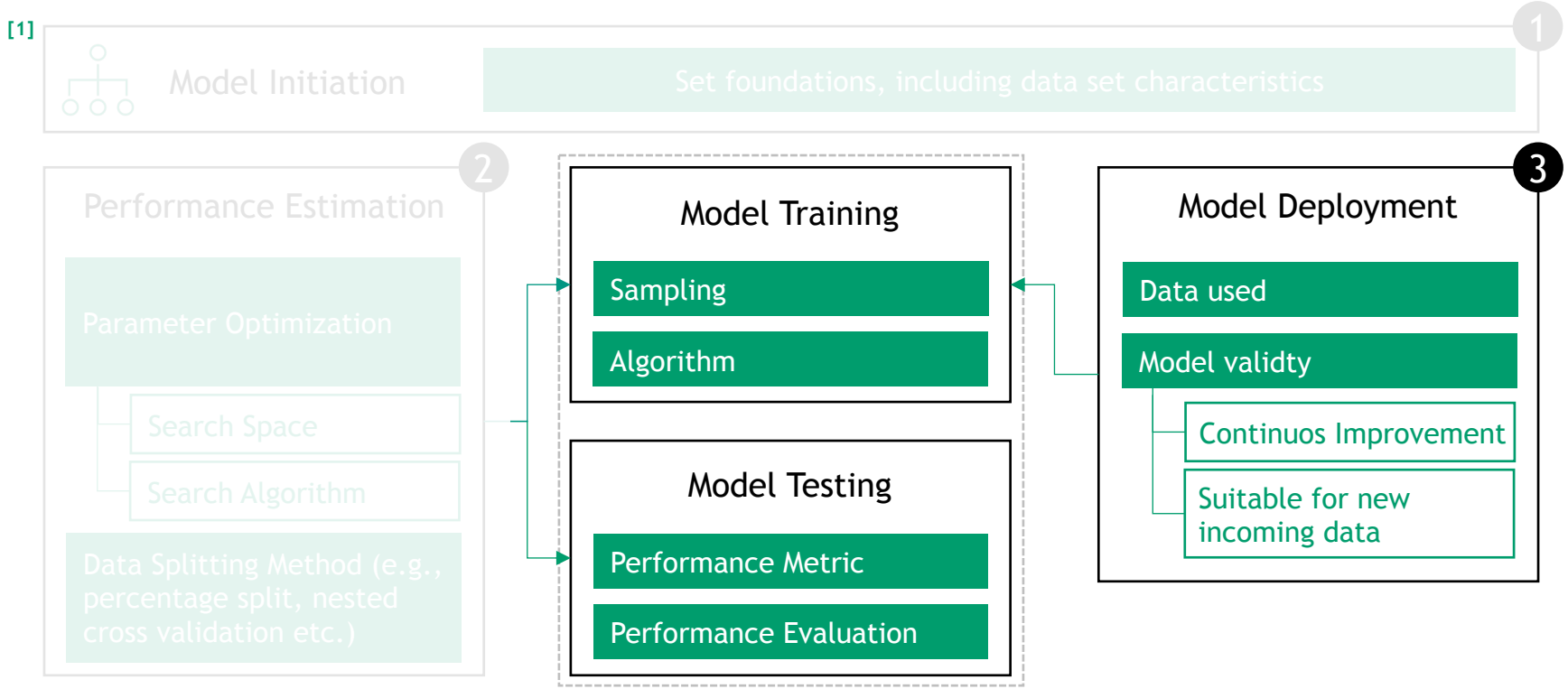
Kühl et al. (2022), Artificial intelligence and machine learning [1]

The AI Lifecycle: Required steps for supervised machine learning in the deployment step



Kühl et al. (2021), How to conduct rigorous supervised machine learning in information systems research: the supervised machine learning report card [1]

The AI Lifecycle: Required steps for supervised machine learning in the deployment step



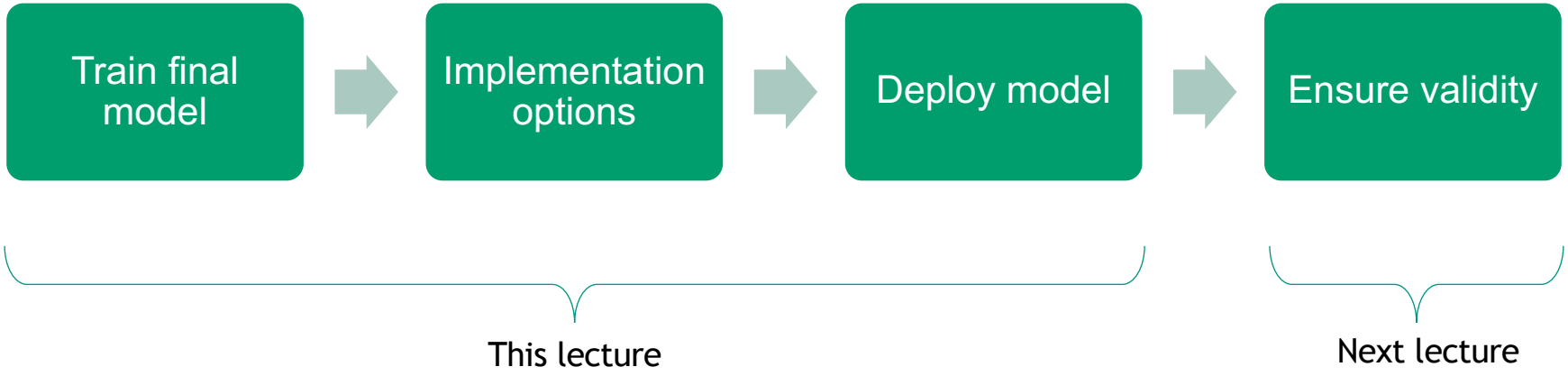
Kühl et al. (2021), How to conduct rigorous supervised machine learning in information systems research: the supervised machine learning report card [1]

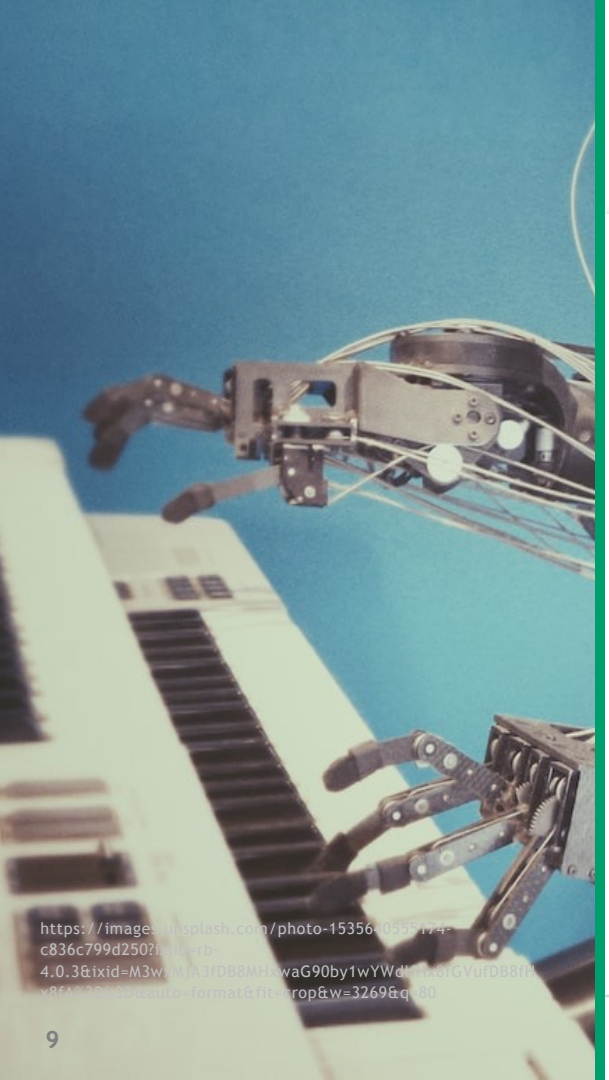
Objectives
What are the learning goals of this lecture?

EXPLORE	UNDERSTAND	INTENSIFY	APPLY
Discover the deployment phase of AI	Understand how to train the final model for deployment	Familiarize with different possibilities of deployment	be able to illustrate different implementation options




Process
This lecture focuses on deployment of final model.



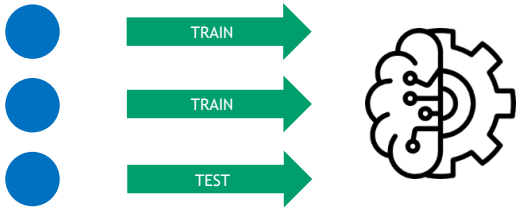


1. Final Training
2. Implementation options
3. APIs
4. Deployment options
5. MLOps

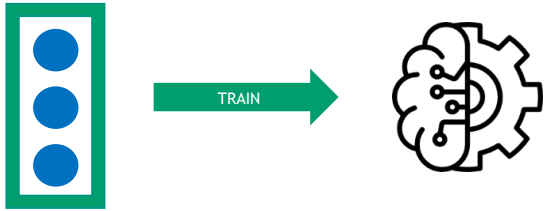
1. Final Training
2. Implementation options
3. APIs
4. Deployment options
5. MLOps

Final Training
All available data is needed to finalize the training.

- Before: (Iterative) model training and testing to estimate future performance, e.g., gain approximations of overfitting tendencies



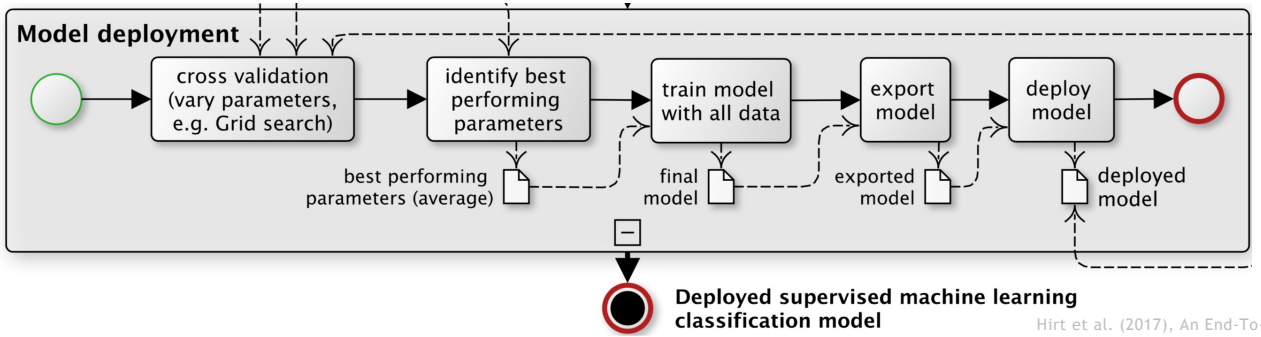
- Now: Use all data available and build model (to the best of your knowledge) for integration into a productive environment



Final Training
The final model is trained using the best performing parameters.

- Select parameters for final training
  - Perform final grid search [optional]: Perform a cross validation with a Grid Search to identify the best performing parameters out of the same parameter search space from the evaluation steps
  - Identify best performing parameters (on average): Check whether performance result lays inside the confidence interval identified during the model evaluation phase
  - Use best performing parameters from evaluation phase

- Train model with all data: All ground truth data is used to train the final model



[1]

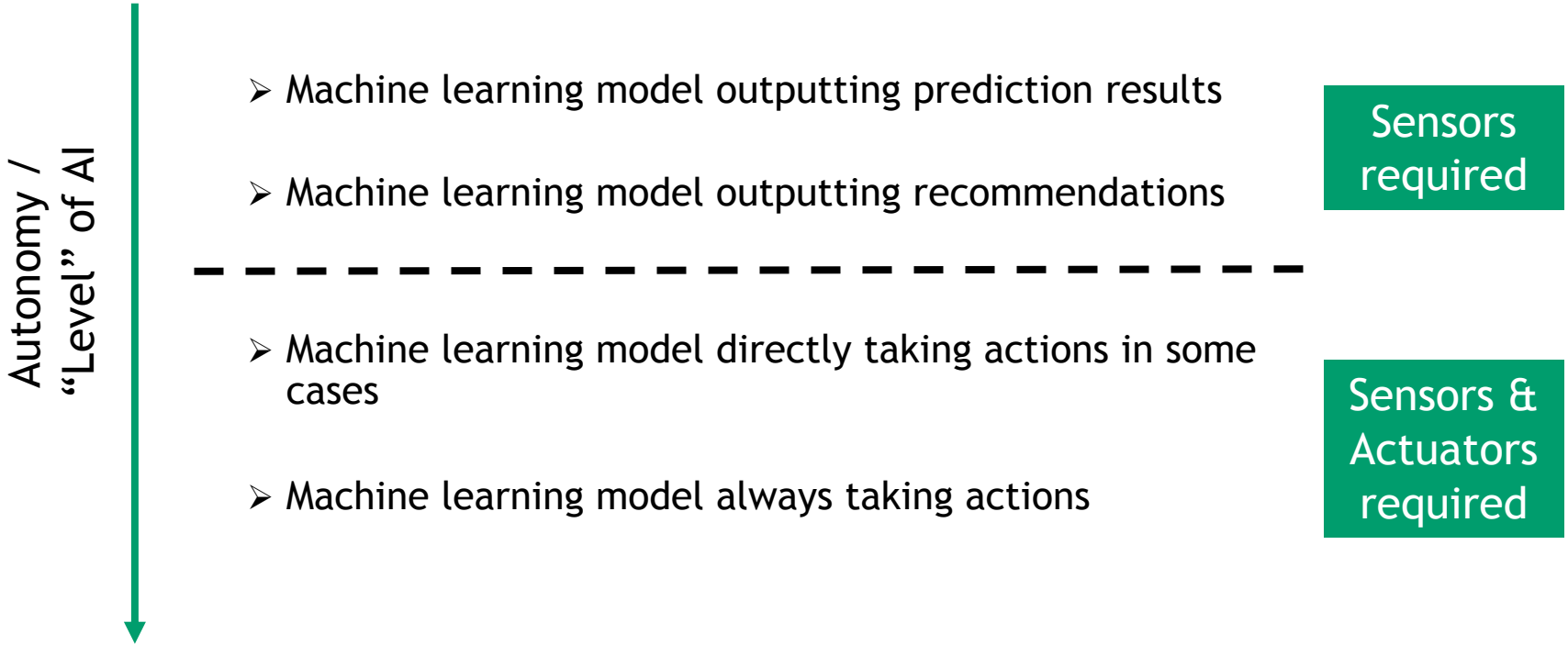
Hirt et al. (2017), An End-To-End Process Model for Supervised Machine Learning Classification [1]

https://images.unsplash.com/photo-1628258334105-2a0b3d6efee1?auto=format&fit=crop&q=80&w=3615&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D

1. Final Training
2. Implementation options
3. APIs
4. Deployment options
5. MLOps

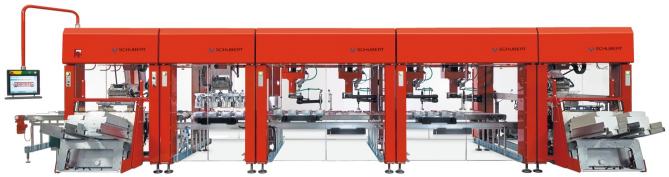
Implementation options

How to distinguish between levels of AI.



Implementation options
An example from practice.

[1]



Case A: No AI
10% of products faulty

Exemplary outcomes		
Output	Shipped to customer	Claimed by customer
100	100	-10

https://commons.wikimedia.org/wiki/File:TLM-Verpackungsmaschine_neueste_Generation.jpg [1]

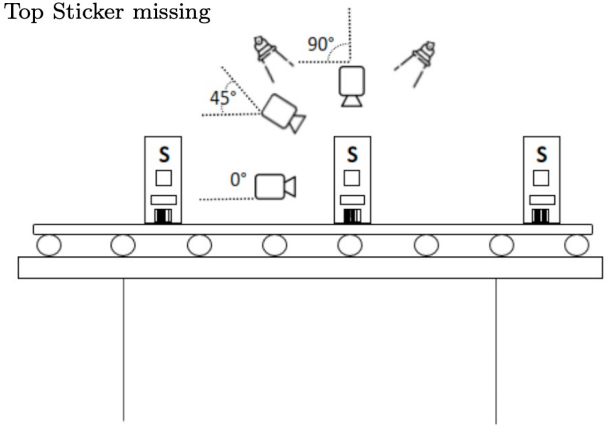
Implementation options
Example from Schwarz/Kaufland

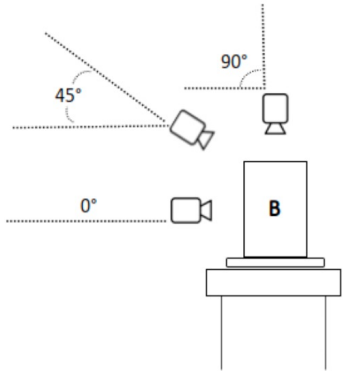
[1]

Fig. 7 Expiry Date, lot number and Barcode defect

Fig. 8 Expiry Date and lot number wrong

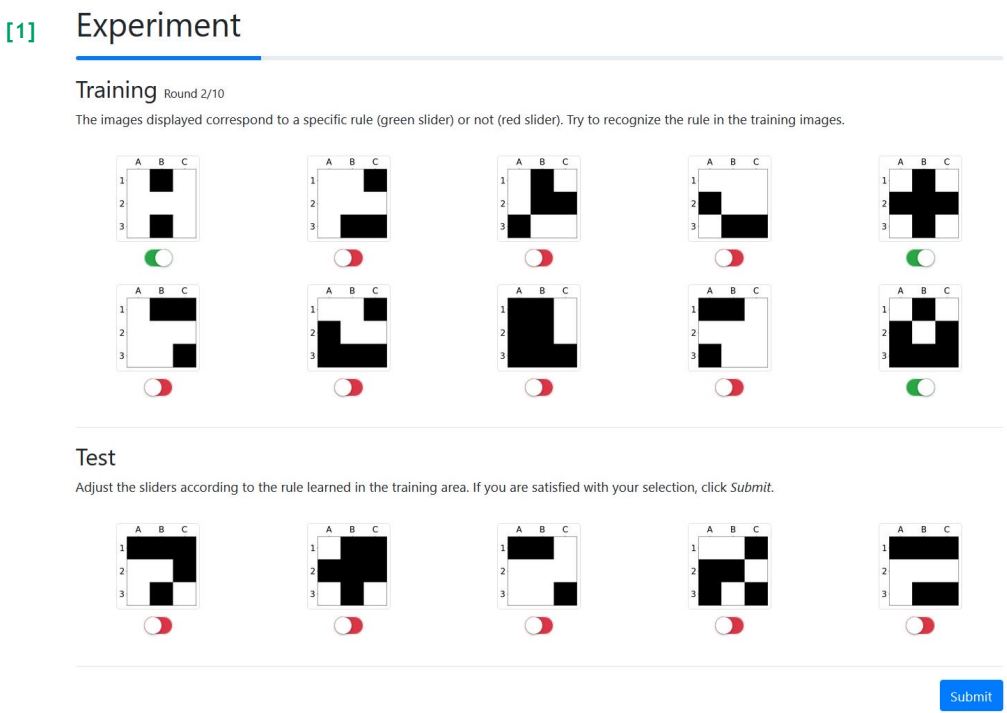
Fig. 9 Top Sticker missing





Xiong, Kühl, Stauder (2023), Designing a computer-vision-based artifact for automated quality control: A case study in the food industry [1]

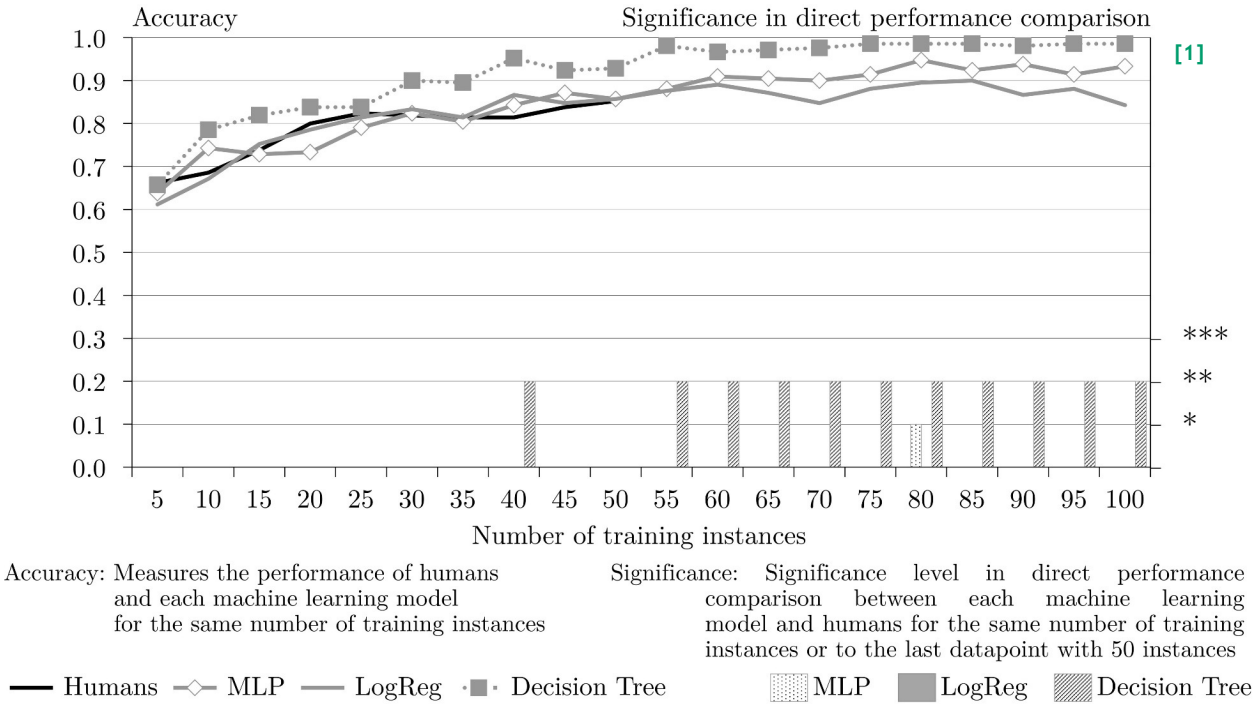
Implementation options
Excursion: Who is better at a task—A deployed AI or a human?



What's the pattern here?

Kühl et al. (2022): Human vs. machine learning: Who learns patterns (faster)? [1]

Implementation options
Excursion: Who is better at a task—A deployed AI or a human?



Kühl et al. (2022): Human vs. machine learning: Who learns patterns (faster)? [1]

Implementation options
What are challenges in deployment?

- Multiple Languages: (Python, R, Scala) are routinely used, even for different parts of the same model. [1]
  → APIs

- Parallel GPU Usage: Deep Learning often requires a lot parallelization, like what's offered by GPUs. These are rarely part of your existing core infrastructure.
  → Cloud

- Unpredictable Costs: Usage of Machine Learning models for inference can follow a spiked and unpredictable pattern, making cost-efficient scaling difficult.
  → Serverless

https://cdn2.hubspot.net/hubfs/4485839/Whitepapers/Enterprise-WP-Deploying-Machine-Learning-With-Serverless-Microservices.pdf [1]

1. Final Training
2. Implementation options
3. APIs
4. Deployment options
5. MLOps

APIs
Example of an Application Programming Interface (API)

- The API expresses a (service) component in terms of its functionalities, inputs, outputs, and underlying types.
- ChatGPT Example:

python Copy [1]

```python
response = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Who won the world series in 2020?"},
        {"role": "assistant", "content": "The Los Angeles Dodgers won the World Series in 20
        {"role": "user", "content": "Where was it played?"}
    ]
)
```

Singh, N. et al. (2024). A Hybrid Ayurvedic Drug Recommendation System with Generative AI. In: Hassanien, A.E., Anand, S., Jaiswal, A., Kumar, P. (eds) Innovative Computing and Communications. [1]

APIs
Documentation of an Application Programming Interface (API)

- The API documentation provides information to other programmers regarding the expected input and output of certain functionalities

The chat completion object

Represents a chat completion response returned by model, based on the provided input.

id string

A unique identifier for the chat completion.

choices array

A list of chat completion choices. Can be more than one if `n` is greater than 1.

+ Show properties

created integer

The Unix timestamp (in seconds) of when the chat completion was created.

model string

The model used for the chat completion.

object string

The object type, which is always `chat.completion`.

usage object

Usage statistics for the completion request.

+ Show properties

The chat completion object — Copy

```
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1677652288,
  "model": "gpt-3.5-turbo-0613",
  "choices": [{
    "index": 0,
    "message": {
      "role": "assistant",
      "content": "\n\nHello there, how may I assist
    },
    "finish_reason": "stop"
  }],
  "usage": {
    "prompt_tokens": 9,
    "completion_tokens": 12,
    "total_tokens": 21
  }
}
```

[1]

https://platform.openai.com/docs/api-reference/chat/object [1]

APIs
A brief look into history.

- Does not include non-public APIs

Roy Fielding
originator of REST

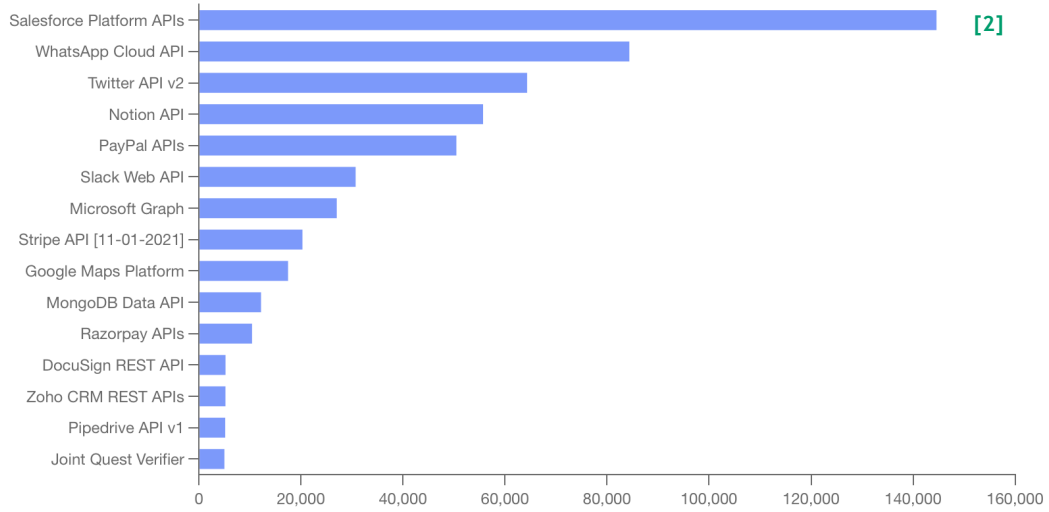
{ REST }

[1] Sales Force Automation API
first complicated XML API

salesforce | Cloud Alliance Partner

2000

Most popular APIs



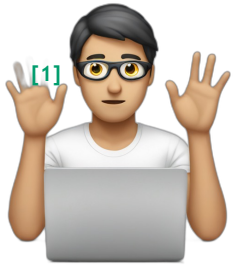
[2]

05/2023, Postman

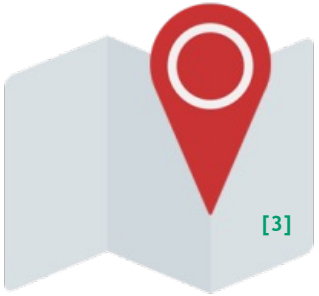
https://commons.wikimedia.org/wiki/File:Salesforce_Cloud_Alliance.jpg [1]
https://www.postman.com/state-of-api/api-global-growth/#api-global-growth [2]

APIs
Example from Uber Eats

- Real-world example of API/Service Mashup



[2]



End User
(The Buyer)

User Application
(Uber Eats)

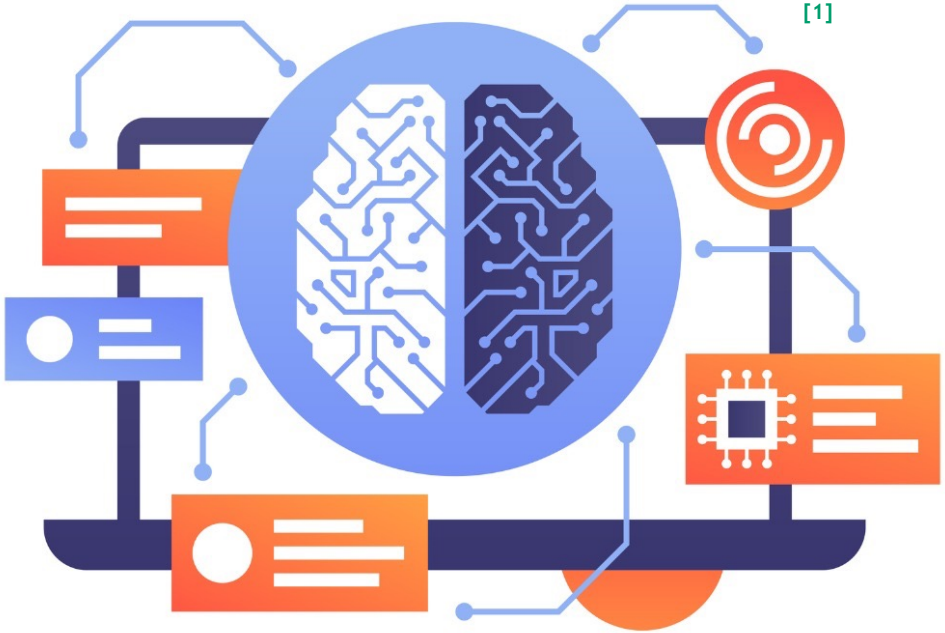
API Service Provider
(Google Maps)

https://emojis.sh/emoji/hacker-person-doing-stop-by-his-hands-KNEHHWVYMq [1]
https://ccnull.de/foto/hand-holding-a-logo-of-famous-takeaway-delivery-service-uber-eats-on-green-background/1063147 [2]
https://icon-icons.com/de/symbol/google-maps/41220 [3]

APIs
Further examples from industry.

… but what about our own-built models?

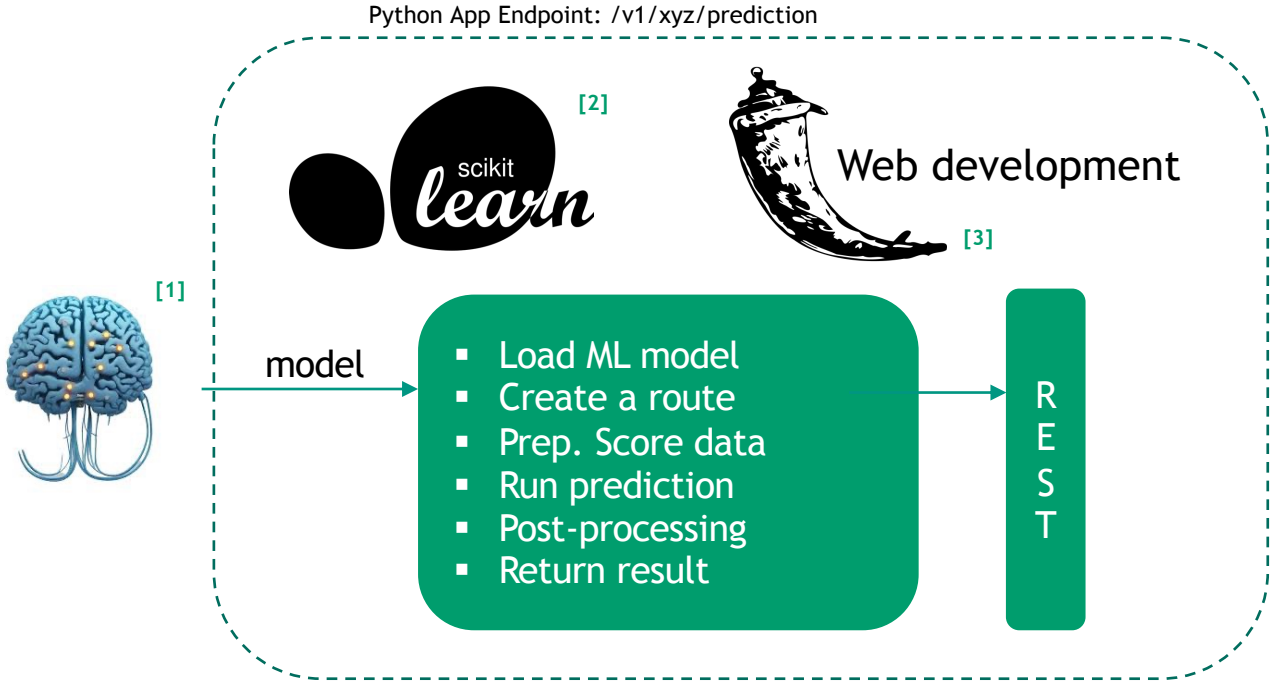
- BigML
- IBM Watson API
- Google Cloud Vision API



[1]

https://www.deanlong.io/blog/the-google-search-machine-learning-we-trust [1]

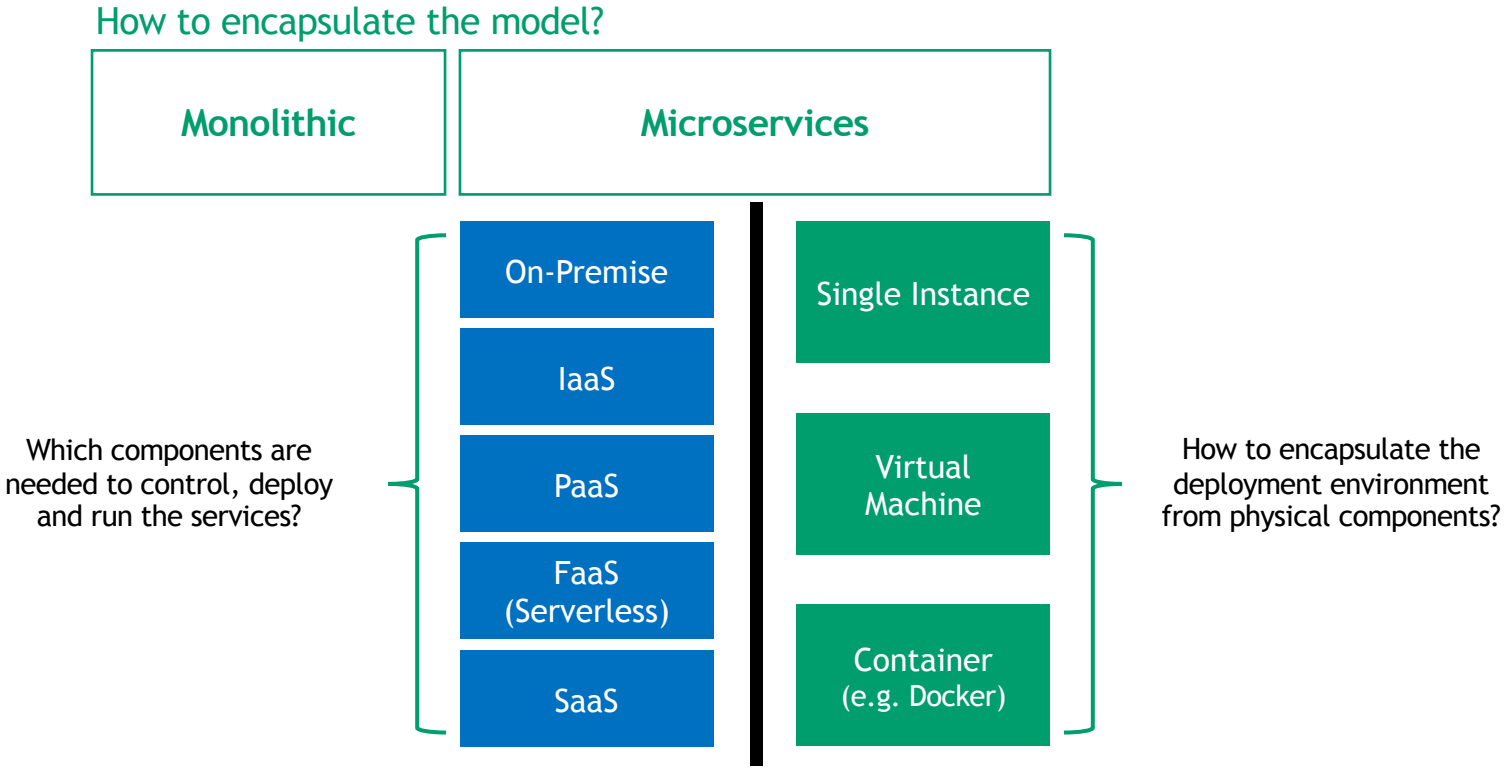
API
Design your customized API.



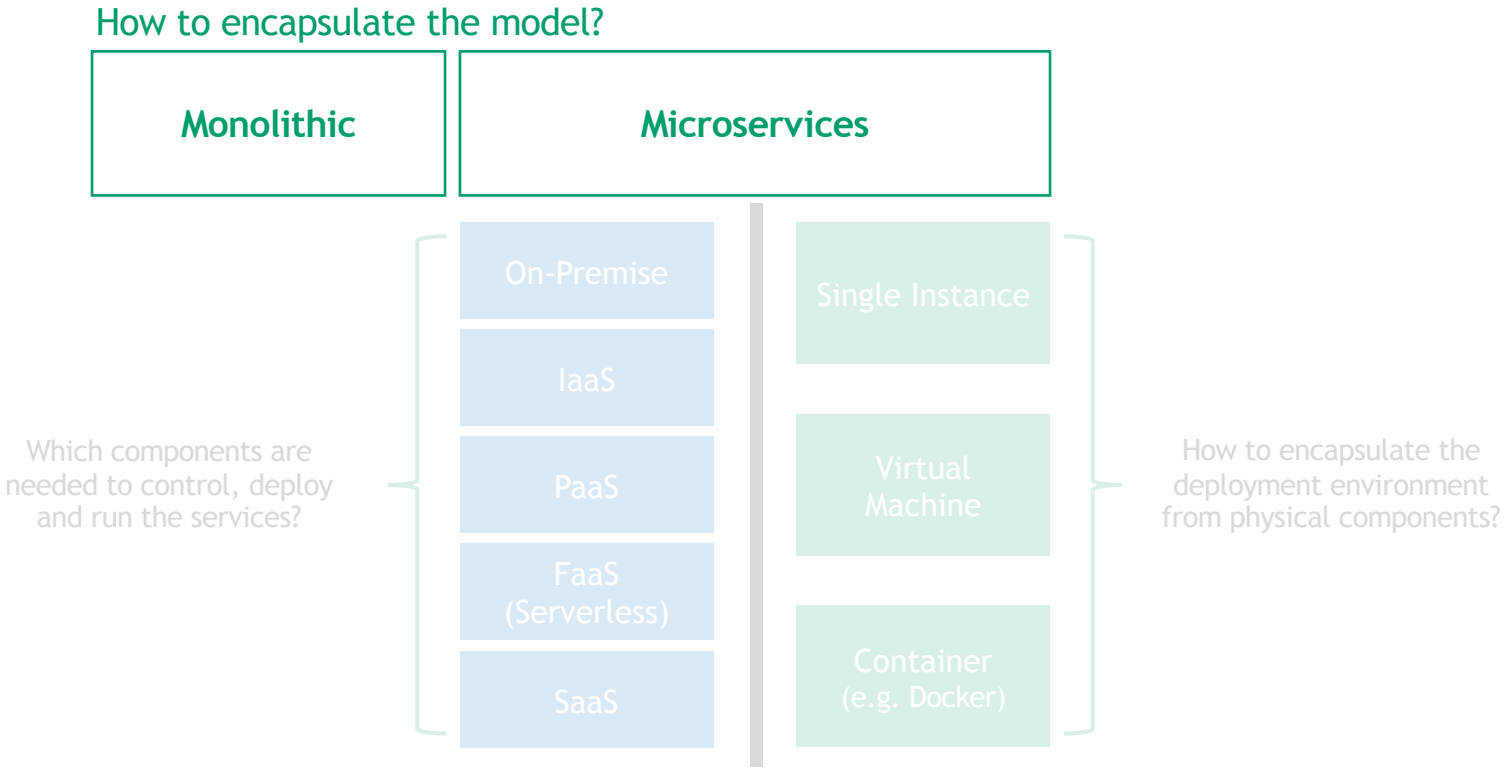
https://emojis.sh/emoji/ai-brain-wires-circuit-ezeGZJl9aq [1]
https://de.slideshare.net/slideshow/how-to-deploy-machine-learning-models-into-production/102834895 [2]
https://creazilla.com/media/icon/3270584/scikit-learn [3]

1. Final Training
2. Implementation options
3. APIs
4. **Deployment options**
5. MLOps

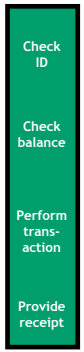
Deployment options
What are the model deployment decisions?



Deployment options
What are the model deployment decisions?

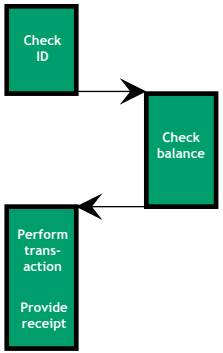


Deployment options
Service-oriented and model-driven programming



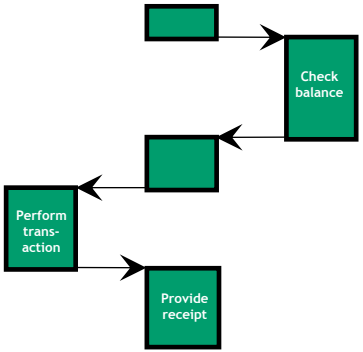
“Monolithic“ Program

One component



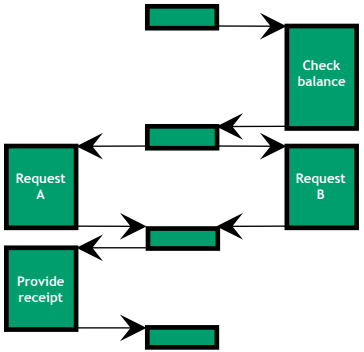
Programming using reusable, pre-programmed components (services)

Two components



Less programming, more use of components (services)

„Many“ components

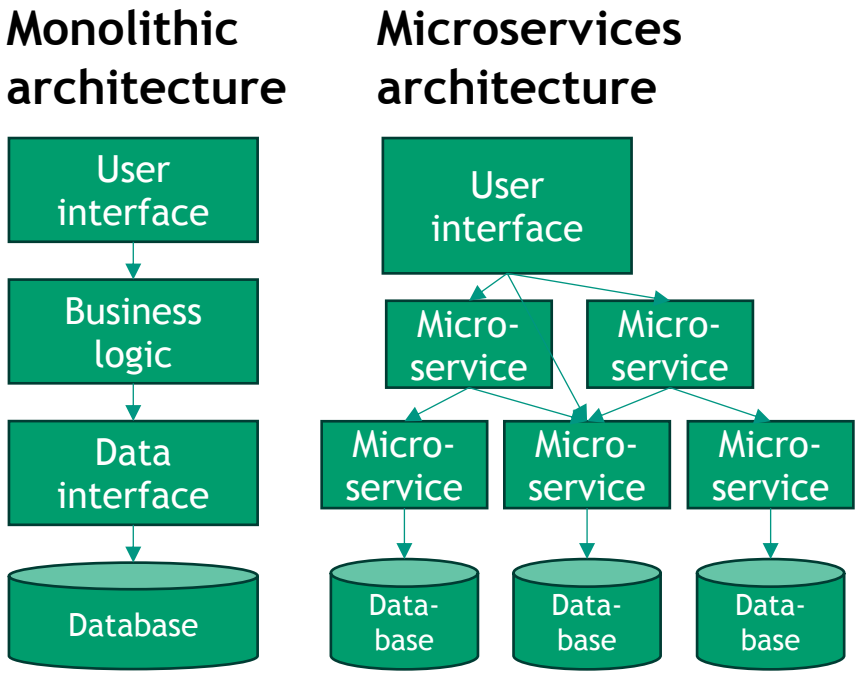


Extensive use of components (services), programming has become “modeling“ or “choreographing“

Simplified example: Banking transaction

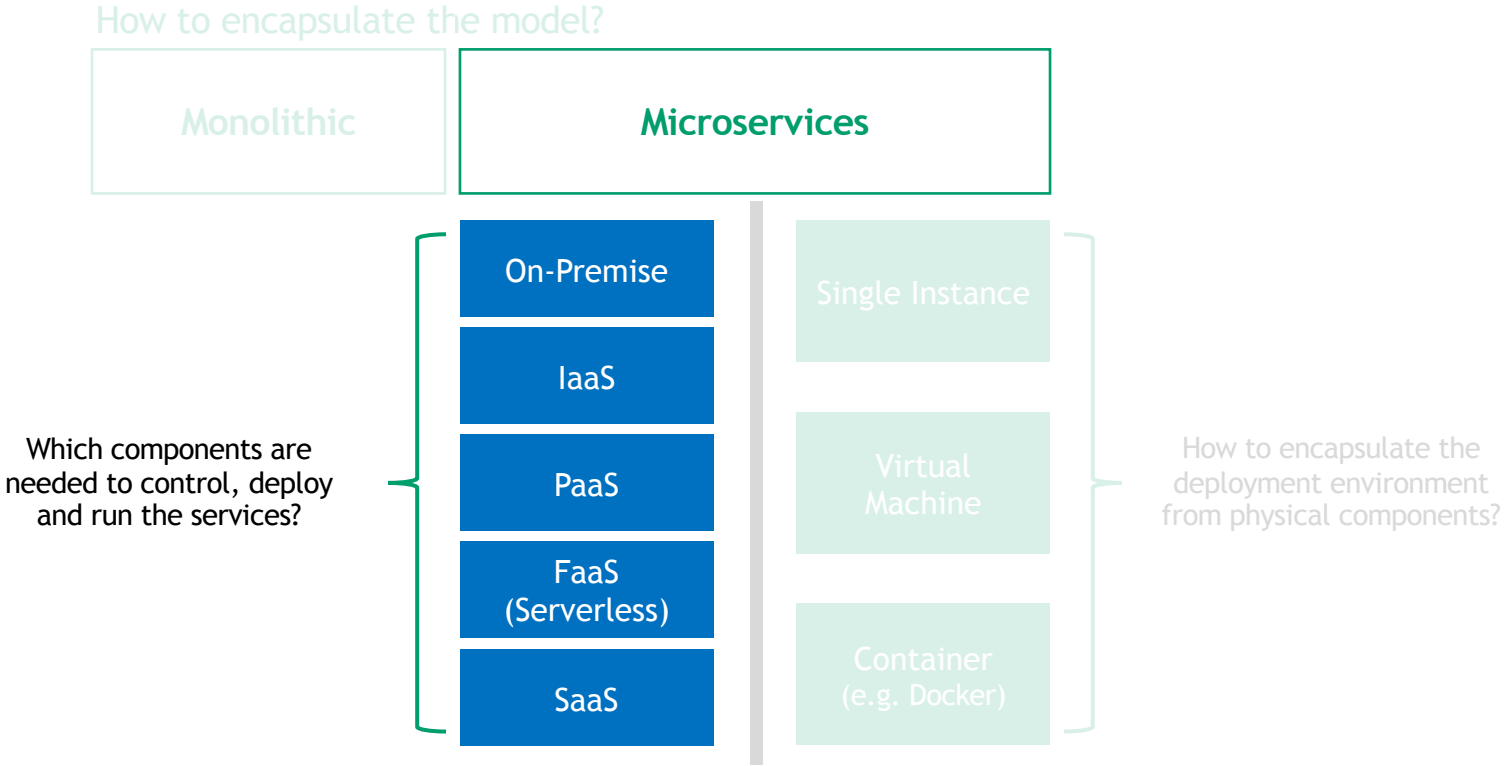
Deployment options
What are Microservices?

- Microservices – are the next step in the evolution of Service Oriented Architectures. They serve only one specific business function and are completely independent of each other. Microser2vices use lightweight HTTP, REST, or Thrift APIs for communicating among themselves. [1]
- "Almost three-quarters (74%) of respondent organizations are currently using microservices architecture." — Gartner (2023) [2]
- "Microservices are a natural paradigm for developing a machine learning pipeline." — Nitin Bandugula (MapR)

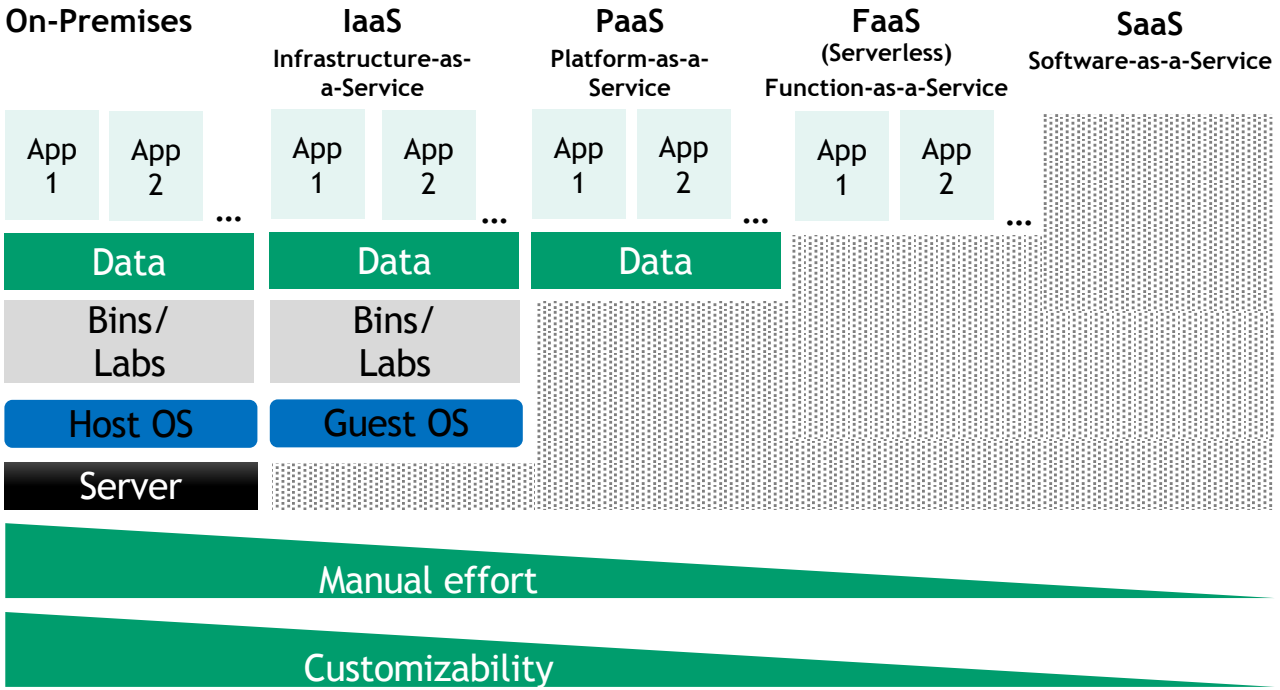


Despodovski (2017), Microservices vs. SOA (Blog) ; Deploying Machine Learning with Serverless Microservices [1]
https://www.gartner.com/peer-community/oneminuteinsights/microservices-architecture-have-engineering-organizations-found-success-u6b [2]

Deployment options
Which components are needed?



Deployment options
Options for (micro)services are manifold.



Peipman (2017), A Short Introduction to Serverless Architecture (Blog) ; Tozzi (2017), Why PaaS vs. CaaS is the wrong Question to ask (Blog) [1]

Deployment options
Infrastructure-as-a-Service (IaaS)

- Provision, processing, storage, networks, and other fundamental computing resources [1]
- Consumer is able to deploy and run arbitrary software, including operating systems and applications
- Consumer has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components
- Consumer does not manage or control the underlying cloud infrastructure

[2]

[3]

Cloud


[4]

NIST, The NIST Definition of Cloud Computing [1]
https://commons.wikimedia.org/wiki/File:Amazonwebservices-original-wordmark.svg [2]
https://icon-icons.com/de/symbol/ibm/130909 [3]
https://commons.wikimedia.org/wiki/File:New-azure-logo-square.png [4]

Deployment options
Platform-as-a-Service (PaaS) – Cloud Platforms

- Capability provided to the consumer to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider
- User has control over the deployed applications and possibly configuration settings for the application-hosting environment
- Consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage

[1]


[2]

Google App Engine

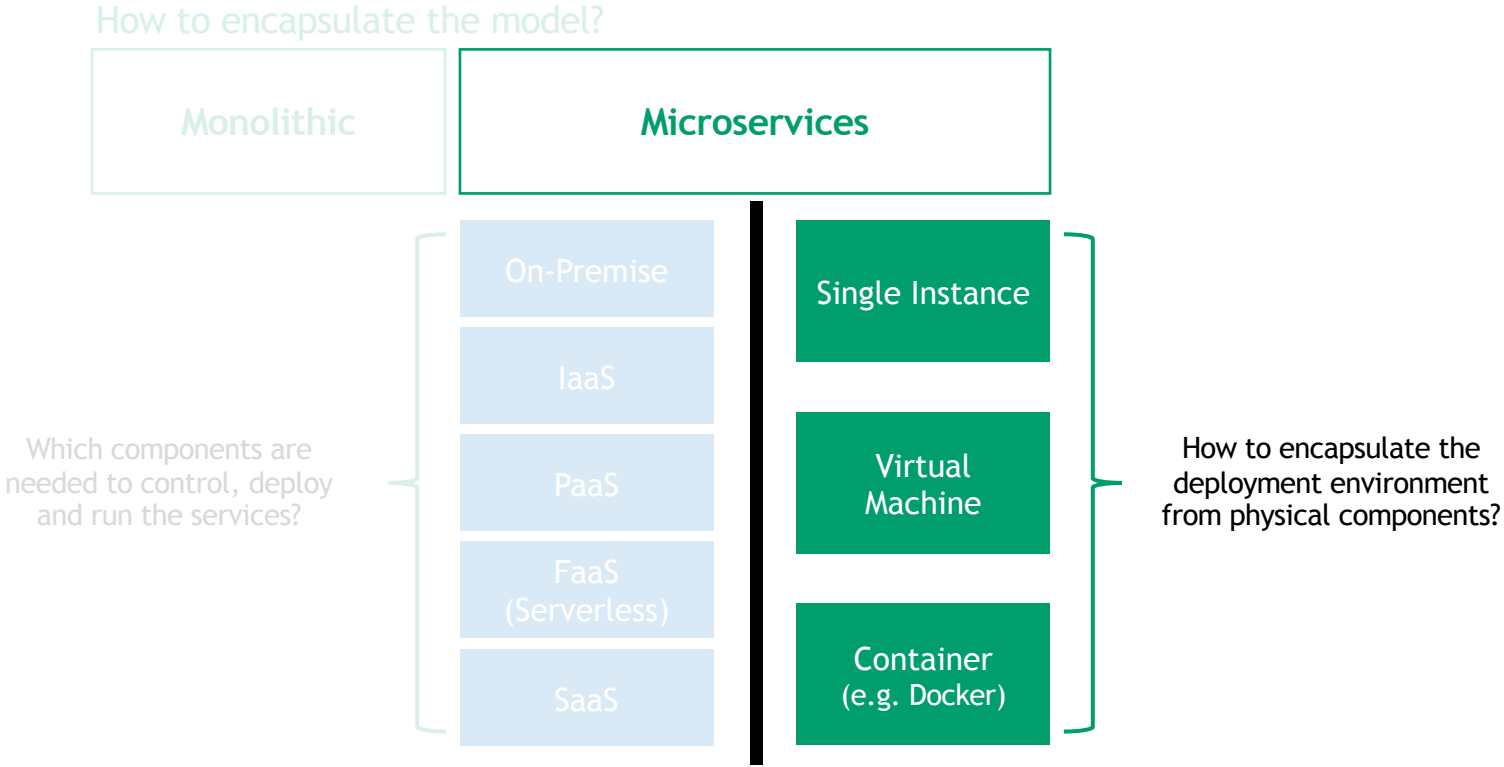
https://commons.wikimedia.org/wiki/File:Salesforce_Cloud_Alliance.jpg [1]
https://icon-icons.com/de/symbol/google/30031 [2]

Deployment options
FaaS (Serverless) – Advantages

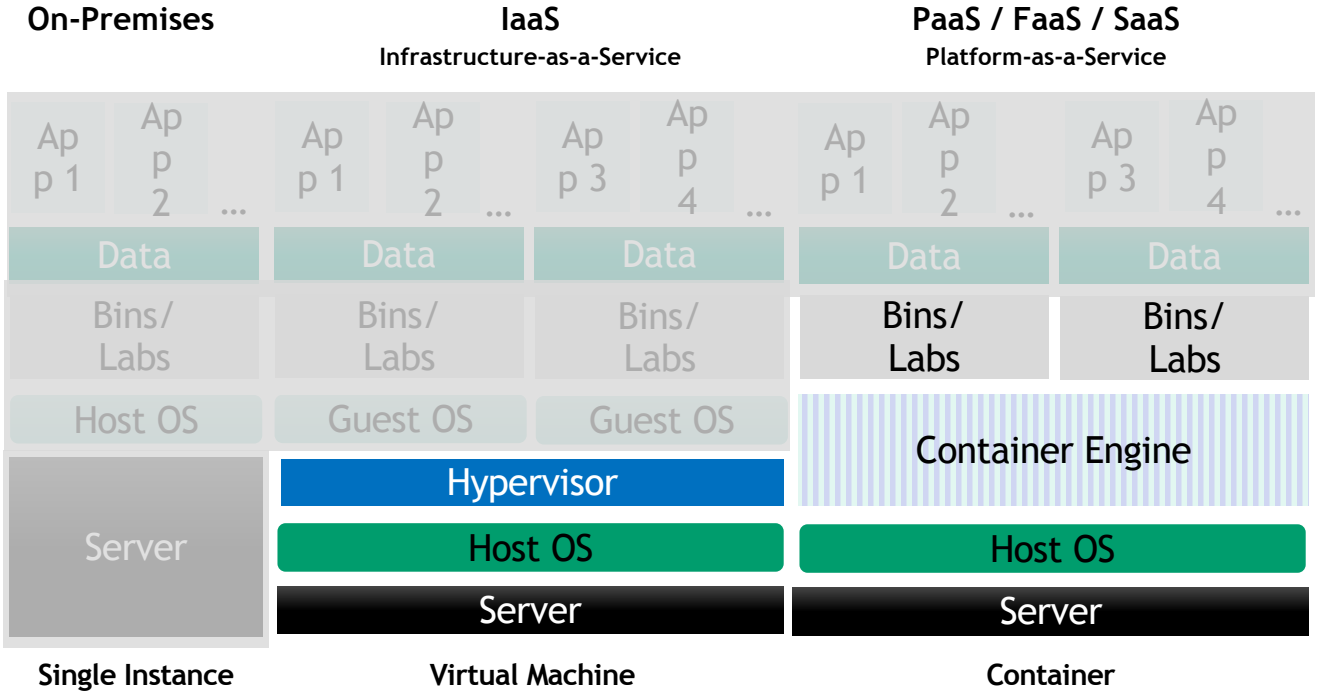
- Function-as-a-Service – is a model to run pure programming code. The service provider deals with every physical or software component besides the actual function.
  - Pay – only when the function is running. No costs for unused server.
  - Server management and security – is provided by the service provider.
  - Deployment – is simple, only the code snippet is needed.
  - Scale – the function seamlessly and almost unlimited.

Further reading: L Schuler, S Jamil, N Kühl: AI-based Resource Allocation: Reinforcement Learning for Adaptive Auto-scaling in Serverless Environments

Deployment options
What are the model deployment decisions?



Deployment options
Virtual Machines & Containers to encapsulate deployment.



[1]

Peipman (2017), A Short Introduction to Serverless Architecture (Blog) ; Tozzi (2017), Why PaaS vs. CaaS is the wrong Question to ask (Blog) [1]

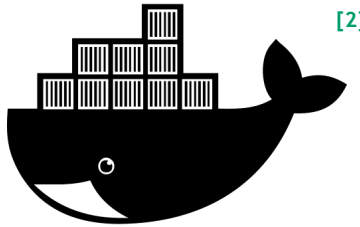
Deployment options
Advantages of Container

- Container – enables an abstraction of applications from the environment in which they run. A container contains a single application including its dependencies but without the full OS to run on. [1]

  - Portability – of containers are given because container contains every dependency and configuration.
  - Efficiency – by not using an entire OS, emulated by a virtual machine
  - Isolation – provides a consistent environment. Every application runs in its own sandbox
  - Scalability – enables to assign more resources to one container but also adding more containers to the same container engine

- Docker – is the best-known container provider. Its standardized container file format further helps portability.

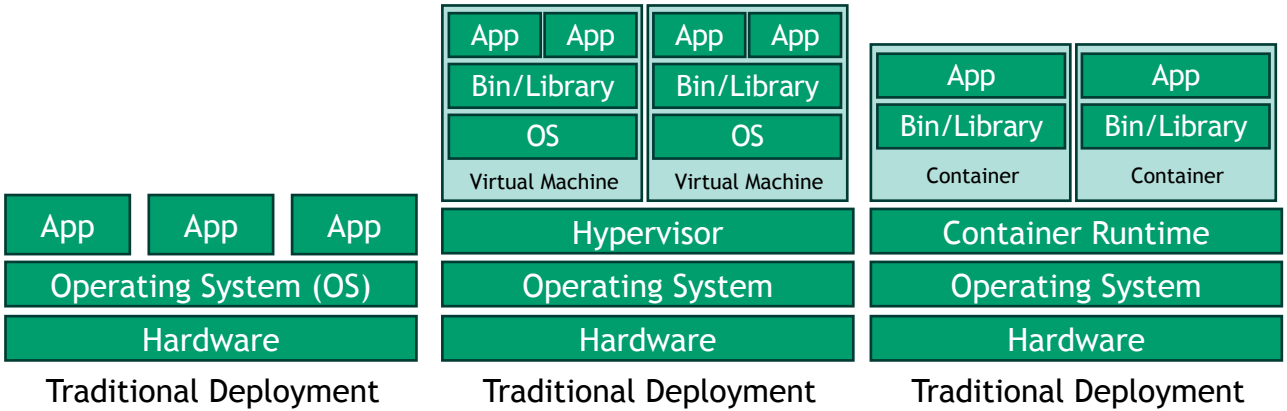
[2]



Peipman (2017), A Short Introduction to Serverless Architecture (Blog) ; Tozzi (2017), Why PaaS vs. CaaS is the wrong Question to ask (Blog) [1]
https://icon-icons.com/icon/Docker/132435 [2]

Deployment options
Exemplary deployment with Kubernetes

- Kubernetes – is a system for managing containers. It enables deploying, maintenance and scaling of containerized applications often across multiple host servers.
  - Pods – bundle several containers on the same host server.
  - Nodes – bundle several pods. All nodes form a cluster.
  - Master Plane – controls the cluster. It communicates with the cluster and schedules tasks.



Bernstein (2014), From lxc to docker to Kubernetes, Kubernetes.io, What is Kubernetes? [1]

Deployment options
Exemplary deployment with Cloud Foundry

- Cloud Foundry – is a platform-as-a-service to develop and scale applications. [1]
  - External resources – such as databases, storage or communication are provided by Cloud foundry as services. Applications have to use these services to e.g. store their internal state. External provides can register their API in the cloud foundry marketplace to be used by applications deployed by Cloud Foundry.
  - Support for programming languages and frameworks – is built in for many common use cases.
  - Scalability – ensued automatically for resources and CPU requirement
  - Logging and monitoring – is provided by Cloud Foundry.

[2]

CLOUD **FOUNDRY**

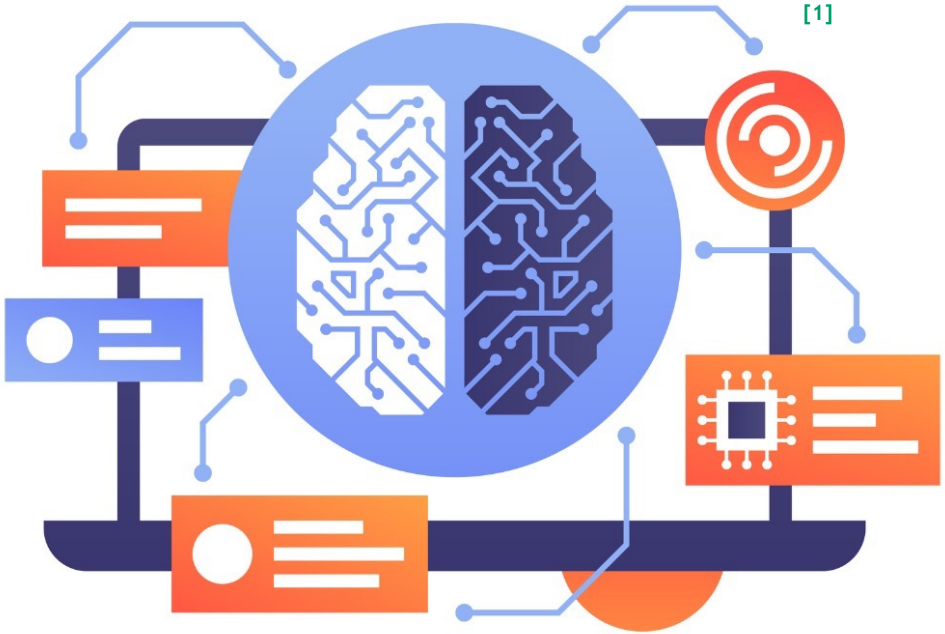
Bernstein (2014), Cloud foundry aims to become the OpenStack of PaaS [1]
https://icon-icons.com/icon/file-type-light-cloudfoundry/130474 [2]

Deployment options
Deployment with managed services (MLaaS = SaaS)

Azure Machine Learning services

Amazon SageMaker

IBM Watson Studio



[1]

https://www.deanlong.io/blog/the-google-search-machine-learning-we-trust [1]

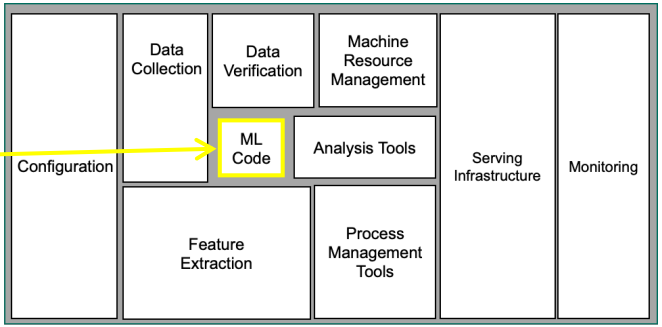
1. Final Training
2. Implementation options
3. APIs
4. Deployment options
5. MLOps

MLOps
Why is Machine Learning Operations (MLOps) interesting?

"What Data Scientists are taught"

```
In [24]: # from sklearn import DecisionTreeClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.metrics import accuracy_score

         my_random_seed = 42

         # Initialize the DecisionTreeClassifier
         model = DecisionTreeClassifier(random_state=my_random_seed)

         # train the classifier
         model.fit(X_train, y_train)

         # Predict with the test data
         y_predict = model.predict(X_test)

         # Determine the accuracy score
         accuracy_score = accuracy_score(y_test, y_predict)

         print ("Accuracy score for the trained model: {:.4f}.".format(accuracy_score))

         Accuracy score for the trained model: 1.0000.
```

"What reality looks like "



[2]

Source: Hidden Technical Debt in Machine Learning Systems | (D. Sculley et. al., 2015)

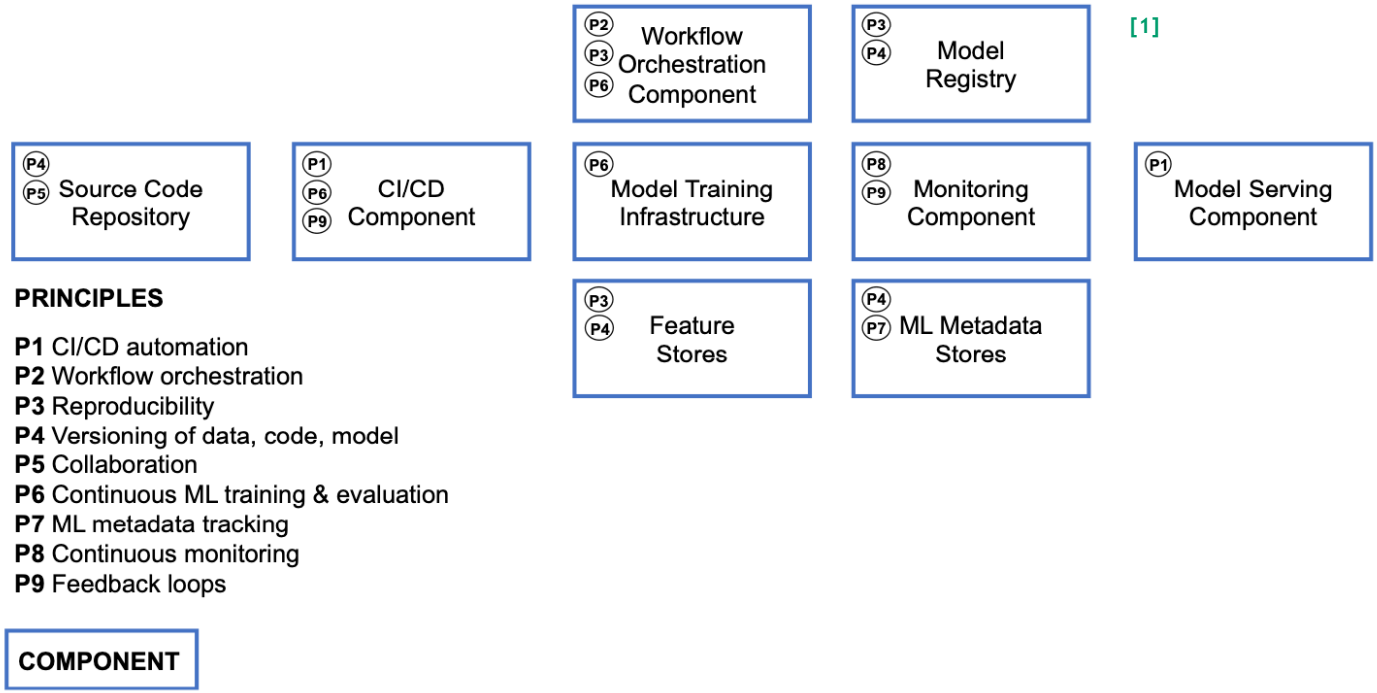
Solution: MLOps

"

[1] *MLOps (Machine Learning Operations) is a paradigm, including aspects like best practices, sets of concepts, as well as a development culture when it comes to the end-to-end conceptualization, implementation, monitoring, deployment, and scalability of machine learning products.*
Most of all, it is an engineering practice that leverages three contributing disciplines: machine learning, software engineering (especially DevOps), and data engineering. MLOps is aimed at productionizing machine learning systems by bridging the gap between development (Dev) and operations (Ops). Essentially, MLOps aims to facilitate the creation of machine learning products by leveraging these principles: CI/CD automation, workflow orchestration, reproducibility; versioning of data, model, and code; collaboration; continuous ML training and evaluation; ML metadata tracking and logging; continuous monitoring; and feedback loops.

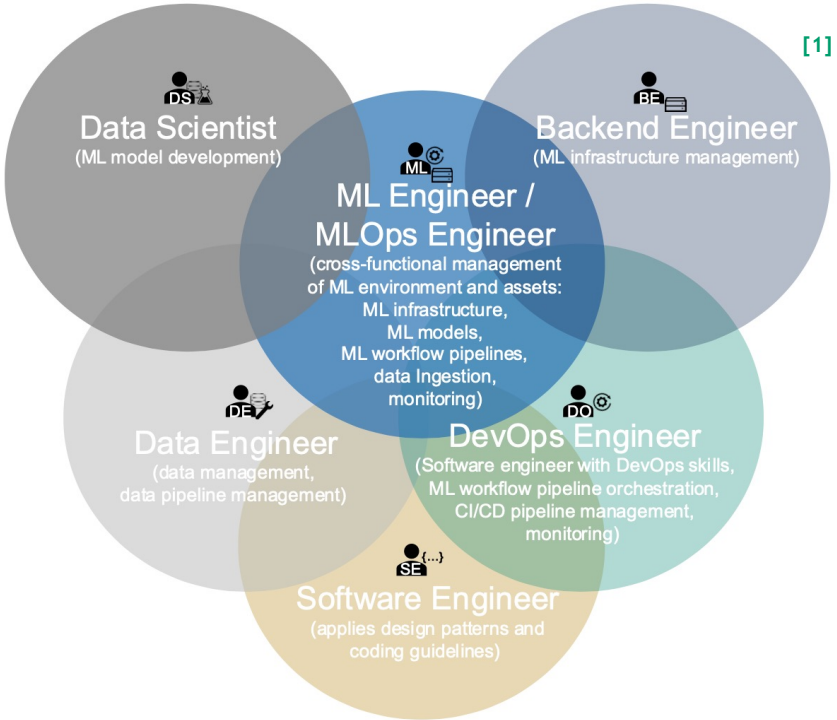
Kreuzberger, Kühl, Hirschl (2023): Machine Learning Operations (MLOps): Overview, Definition, and Architecture [1]
Iliadis, Andrew & Acker, Amelia. (2022). The Seer and the Seen: Surveying Palantir's Surveillance Platform. 10.13140/RG.2.2.34071.85927. [2]

MLOps
Based on principles to be implemented as components.



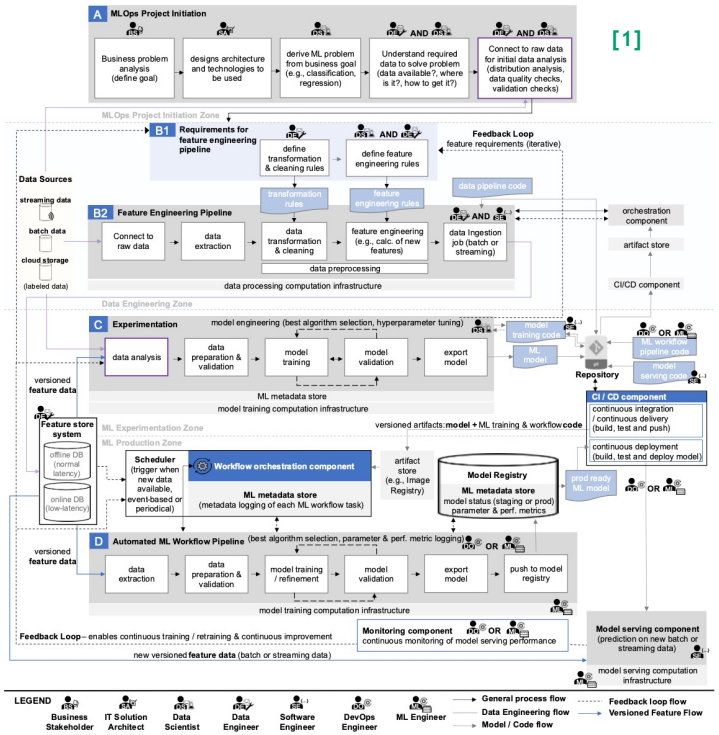
Kreuzberger, Kühl, Hirschl (2023): Machine Learning Operations (MLOps): Overview, Definition, and Architecture [1]

MLOps
Successful MLOps requires multiple roles – not only Data Scientists



Kreuzberger, Kühl, Hirschl (2023): Machine Learning Operations (MLOps): Overview, Definition, and Architecture [1]

MLOps
Architecture and Workflow



Kreuzberger, Kühl, Hirschl (2023): Machine Learning Operations (MLOps): Overview, Definition, and Architecture [1]

Summary

Final Training	Degree/ Autonomy	Deployment Options	Existing Platforms	MLOps
A final training is required for an operational, to-be-deployed model.	There are different options on the “degree/autonomy ” of a ML model within an AI, e.g., whether this model can make decisions in the real world.	The deployment options are manifold, and different choices must be made in its design.	There are existing platforms supporting an AI deployment endeavor, e.g. Kubernetes, Cloud Foundry or even holistic MLaaS solutions.	MLOps is a paradigm when it comes to the E2E conceptualization, implementation, monitoring, deployment, and scalability of ML products.