j&s-soft GmbH

Whitepaper

Machine Learning with SAP

An Overview and Best Practices Guide

v1.0.275-0f57611

SUMMARY / TOC

Over the course of the last three decades, machine learning has evolved from a mere academic exercise with a few niche applications into a dominant topic whose advances are at the core of many recent technological innovations and breakthroughs. A vastly expanded volume of available development data, increasingly affordable computing power and betterments of algorithms and related software have made machine learning accessible and applicable to a wide spectrum of industries and businesses - in theory.

Notwithstanding its popularity and apparent pervasiveness, machine learning remains a technically challenging subject. Many businesses struggle with the practical aspects of interfacing their existing processes with machine learning applications and face considerable expenses for the endeavor to merge novel technologies with their existing tech-stack. To make matters worse, the highly convoluted technological landscape complicates orientation and introduces a substantial entry barrier.

Addressing these matters this whitepaper provides a set of guiding principles and best practices for businesses that seek to venture into the field of machine learning. We provide a synopsis of the most relevant machine learning technologies provided by SAP. Although the focus of our market overview lies in SAP-related technologies the predominant majority of the information presented in this document is universally valid and applicable.

Chapter I introduces basic concepts and gives an overview of the current state of affairs of machine learning. Chapter II details the anatomy of a fully-featured machine learning application and helps to gauge the overall effort involved in its realization. Chapter III lists SAP's most relevant technologies and tools related to machine learning and offers insights into possible application scenarios. Finally, Chapter IV presents the recommended best practices derived from the expositions of the preceding chapters and concludes this report.

Machine Learning Fundamentals

If and how well machine learning may be employed in an organization depends on the degree of understanding of what it is and where it is headed. This chapter provides a quick economic outlook and introduces basic terminology pertaining to machine learning and artificial intelligence and serves as a brief glossary for the notions most relevant to this report. We prefer conciseness and clarity over an attempt to provide a rigorous and exhaustive set of definitions which are bound to be complex and unwieldy. As this is a 'practical' report we are accordingly concerned with those aspects of terminology which turn out to be relevant for practical application.

Economic Outlook

In this section we consult several recent surveys and reports to establish a baseline of how machine learning is perceived by the industry and what expectations it elicits in its early adopters. While machine learning may seem to be pervasive and to be used across all industries, as its extensive publicity in the technologically oriented media may suggest, the reality is very different. Although surveys tend to exaggerate towards the positive many participants are willing to concede that their own experience paints a much more humbling picture.

McKinsey estimates that artificial intelligence techniques, which are currently mainly based on machine learning, have the potential to create between $3.5 trillion and $5.8 trillion in annual value across industries [1]. Similar estimates were published by Gartner [3]. Driven by such high expectations, many companies reach for new opportunities and, depending on the survey, between 33% and 63% of enterprises worldwide claim to have already adopted machine learning in production today [2, 5, 6].

In a survey of 1.100 IT and line-of-business executives from US-based companies, 63% state that their artificial intelligence adoption aims to catch up with competitors or to establish a lead [6]. 42% even conjecture that artificial intelligence technologies will become critical in the next one to two years [6], which is a comparatively short period of time to establish expertise given the complexity of these technologies.

In practice, any potential benefits of adoption are contrasted with a high amount of time and effort to build up expertise and to overcome machine learning specific limitations and obstacles [1]. Although two-thirds of enterprises believe in the importance of artificial intelligence and machine learning, at most one-third trust in their competence and experience in dealing with these new technologies [7]. Besides model building, a lack of experience in model deployment and consumption in an application poses a major problem. The result is an apparent gap between model experimentation and models that end up in production. Gartner predicts that most artificial intelligence projects will not make it to production until 2022 [4].

Based on these observations, **Machine learning as a service** providers increasingly supply enterprises with implemented or pre-configured functionality which serves to facilitate the adoption of machine

Back to ToC

learning and to narrow its technical entry barrier. New services continue to spawn and even though machine learning on an enterprise-scale is certainly still in its early stages the machine learning market is already very rich. Among the providers of machine learning services is SAP, which is the focus of this report and to which Chapter III and Chapter IV are dedicated.

Areas in Artificial Intelligence

The terms *artificial intelligence*, *machine learning* and *deep learning* are often confused and are utilized interchangeably. The following paragraphs clarify and differentiate the three concepts and characterize their mutual relationship which is summarized by the symbolic diagram in Figure I.1.

Artificial Intelligence

Artificial intelligence denotes the concept of computers performing tasks which normally require human intelligence and has established itself as an indiscriminately used umbrella term encompassing everything a computer can be made to do that might be perceived as 'smart' by a human. We explicitly stress that the term is nebulous and that it is best to conceive of artificial intelligence in accordance with how it is contemporarily used and understand it as a vague notion in the above sense as opposed to attempting to find a rigorous definition for it.

In addition to the strict subset of machine learning techniques characterized in the next section artificial intelligence includes rule-based programming techniques which are sometimes referred to as classical programming techniques. Rule-based programming techniques can be used to construct sophisticated expert systems and rules engines which may justifiably be perceived as smart. The characteristic which distinguishes artificial intelligence from machine learning is that the 'intelligence' in classical AI is realized by the manual construction of a decision tree which maps the understanding of a human agent into the computer program whereas machine learning techniques are applied to construct the decision tree semi-autonomously.

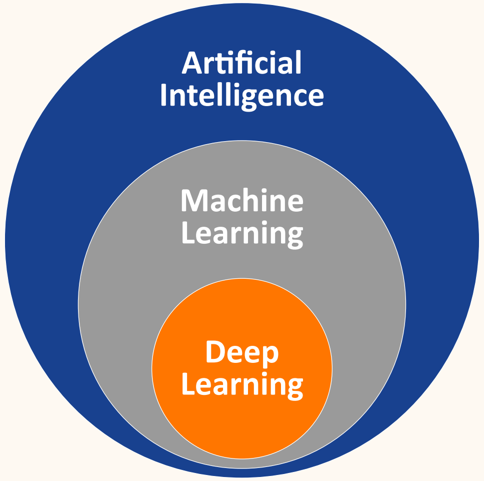

Figure I.1: **Symbolic set diagramm highlighting the relationship between artificial intelligence, machine learning and deep learning.** Artificial intelligence encompasses machine learning and classical, rule-based programming techniques. Deep learning is distinguished from machine learning by the application of multi-layered neural networks.

It has become a widespread custom to attach the AI-label to almost everything in an attempt to elicit a perception of grandeur and importance in the target audience. Artificial intelligence has a long and interesting history and its recent technological developments have ignited the imaginations of many which has brought the term back into vogue. This is very relevant in the corporate context of enterprise software with which this report is partly concerned. Also, note that whenever breakthroughs in or novel applications of artificial intelligence are reported they apply exclusively to machine learning or deep learning.

Machine Learning

Machine learning is the science of providing computers with the ability to learn and automatically improve from experience without being explicitly programmed. The focal point of this definition lies in its final part which stresses that, in contrast to classical programming techniques, there is no need to manually construct the reasoning logic. Instead, machine learning techniques are used to autonomously adapt to a provided set of data and to 'learn' its characteristics. A machine learning program alters itself.

The learning procedure corresponds to an optimization problem in which the loss function, a task-specific metric, is minimized. During this process internal variables are continuously adjusted until a stable point is arrived at. The entirety of the internal variables and the applied algorithms are referred to as the **machine learning model**. After the conclusion of the training phase the model's parameters reflect the internal state of the reasoning logic which has been extracted from the data.

By definition, machine learning applies where rule-based reasoning is unsuitable, be it due to the fact that the rules are unknown or because the rules are prohibitively copious and intricate. Many tasks which are performed based on human intuition fall into this category such as the ability of every child to distinguish daisies from sunflowers or to differentiate the sounds produced by different musical instruments.

Deep Learning

Deep learning is a subset of machine learning which is concerned with **neural networks**. Neural networks are a concept loosely inspired by biology in which *artificial neurons* are arranged in multiple interconnected layers as shown below in Figure I.2. The naming of deep learning simply stems from the circumstance that the tree graph spanned by multi-layered neural networks is *deep*, as opposed to architectures with a single or two layers which are referred to as *shallow*.

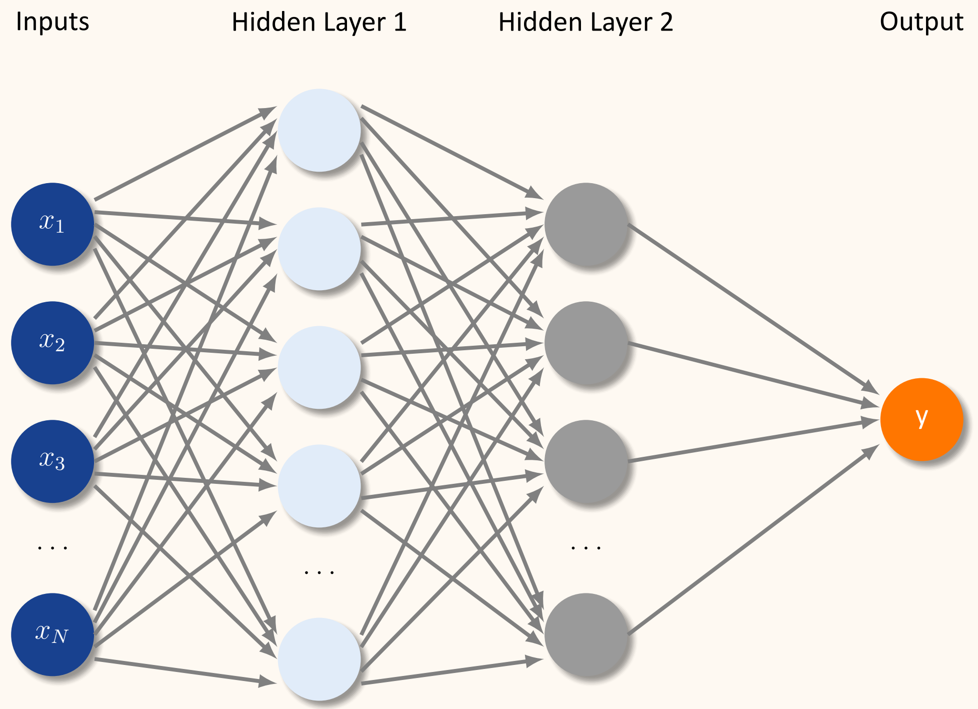

Figure I.2: **Multi-layer Neural Network** Multi-layer networks refer to architectures with multiple layers between the input and output layers, which are as referred to as *hidden layers*. The resulting tree diagram grows in depth with every layer which gives *deep learning* its name.

The class of models comprised of neural networks and the associated algorithms are at the heart of the many breakthroughs of recent years and have made it possible to surpass human performance levels in areas such as image and audio recognition and natural language processing, i.e. in the very fields of applications which are notoriously difficult to manage using rule-based systems and in which human intuition reigned supreme over computer agents.

An essential advantage of deep learning is that neural networks can be applied to unstructured data.

Back to ToC

Many statistical algorithms of machine learning require structured data, i.e data that conforms to a simple format such as tabular data which makes said algorithms inapplicable to fields where unstructured data is prevalent such as in the fields of application mentioned above.

An important disadvantage of neural networks is the requirement of very large amounts of data. Due to this high data volume and the complex network topology, training deep neural networks usually requires a lot of computing power and training time. Another drawback of deep learning is the **'black box' problem**. The trained networks are rarely interpretable and it is nearly impossible to reason in human terms why a certain conclusion was drawn.

Basic Machine Learning Terminology

Model Constituents

As stated above, machine learning is based on the idea of automated parameter fitting to adapt an algorithm to the information provided by a given dataset. The overall aim is to arrive at an approximation of the distribution underlying the data. If the data is representative and the approximation is accurate it can be used to provide predictions on new, unseen data. This process involves constituents which we briefly describe here for later reference. See Figure I.3 for a symbolic overview.

The algorithm is a class of rules or functions that are parameterized during the training process. It determines the totality of all distributions which may be learned from the provided data set. Algorithms are chosen based on the characteristics of the data and assumptions about the problem at hand. An example of an algorithm is polynomial regression.

The hyperparameters are the static subset of an algorithm's variables, which is not estimated during training. Hyperparameters determine the shape and character of a model and are specified manually prior to training. Their choice selects a specific function from of the algorithm's class of functions to be trained on the data. The hyperparameters for a polynomial regression include the degree of the polynomial.

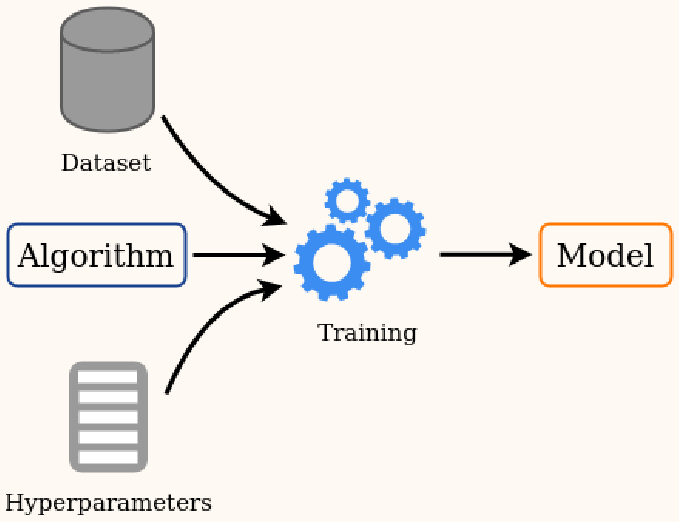

Figure I.3: **Constituents of a machine learning model.** The generic algorithm and a set of hyperparameters comprise the model whose internal parameters are adjusted according to a training data set.

The parameters are the internal variables of a specific function determined by a choice of hyperparameters which are estimated and incrementally adjusted during the training process. Their values capture the internalized representation of the data set which was utilized for training. The parameters for polynomial regression correspond to the weights of the polynomial.

A model is an umbrella term describing a selected algorithm and a concrete choice of hyperparameters. A model is said to have been trained if it has undergone the training process and its parameters have been determined accordingly. The term is sometimes used more broadly referring to algorithms or whole classes of algorithms.

Learning Types

The broad field of machine learning can be subdivided into categories according to the type of training scheme which is applied to a model for a given task. Distinctions are made between supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is utilized when the aim is to make predictions about input data which may either be assigned to a member of a set of categories or a continuous numerical value. Such tasks are referred to as **classification tasks** or **regression tasks**, respectively. Supervised learning uses labeled input data for the training process, i.e. each piece of input data on which the model is trained is accompanied by the 'true' value according to which the model's parameters are steered stepwise in the right direction. Exemplary supervised learning tasks are given in Table I.1.

Table I.1: **Exemplary supervised learning tasks: Classification vs. regression tasks**

	Classification	Regression
Task description	The output variable is a category or an integer	The output variable is a real value or continuous quantity
Task examples	Music genre classification	Housing price prediction
	Spam email detection (spam vs. no spam)	Predicting how much a customer will spend
	Predicting failure of machines (failure/no failure)	Predicting the remaining lifetime of a machine

Unsupervised learning is applied to autonomously uncover hidden structures in the data. In contrast to supervised learning, unsupervised learning attempts to recognize previously unknown patterns in the data without the explicit requirement for labels. Two important applications of unsupervised learning are **clustering** and **dimensionality reduction**. Clustering, or cluster analysis, divides the data into groups of similar data points, whereby the measure of similarity is determined by the choice of the model. Dimensionality reduction aims to remove portions of data which are redundant or may be expressed more concisely in order to find a minimal representation of one's data set.

Reinforcement learning is an additional type of machine learning problem which is less frequently used in enterprise contexts but shall be mentioned here for the sake of completeness. Reinforcement learning corresponds to the approach that most resembles the intuitive understanding of learning as perceived in daily life. It is a sequential decision-making method that associates an underlying **state** and an **action** that was performed based on this state with an **outcome**, i.e. a reward or a penalty. While the initial actions are random, they are refined throughout the training process with the overall aim to maximize the **cumulative reward** over a series of states by choosing the action which yields the maximum reward for each state.

References

[1] Michael Chui et al. *Notes from the ai frontier - Insights from hundreds of use cases*. Tech. rep. 2018.

[2] Louis Columbus. *The State of AI and Machine Learning*. Tech. rep. 2019. URL: Link.

[3] Gartner. *Gartner Says Global Artificial Intelligence Business Value to Reach $1.2 Trillion in 2018*. 2018. URL: Link.

[4] Gartner. *Gartner Says Nearly Half of CIOs Are Planning to Deploy Artificial Intelligence*. 2018. URL: Link.

[5] Ben Lorica and Paco Nathan. "The state of machine learning adoption in the enterprise". In: *O'Reilly* (2018).

[6] Jeff Loucks, Tom Davenport, and David Schatsky. "State of AI in the Enterprise, 2nd Edition". In: (2018), p. 26.

[7] Kevin Poskitt. *SAP Data Intelligence: Enterprise AI Meets Intelligent Information Management | SAP Blogs*. 2019. URL: Link.

The Machine Learning Life Cycle

Building a mature, real-world machine learning application involves many facets beyond the mere choice of a suitable model or algorithm and its training. Model deployment, evaluation and monitoring comprise essential components of a machine learning project's pipeline and require just as much thoughtful consideration. This chapter aims to provide a structured blueprint of a typical machine learning project. Its individual components are laid out sequentially as a series of logically consecutive steps which need to be implemented one way or another.

In spite of the sequential structure, many components exhibit mutual dependencies that require incremental fine-tuning of previously implemented components, going back and forth multiple times. In practice, a machine learning project is not carried out in a top-down fashion but instead is realized by iterating over the individual steps of implementation multiple times, incorporating acquired information and understanding which oftentimes calls for readjustments to the project's original goals. We thus refer to this iterative and cyclical process as the machine learning life cycle.

The schema of the machine learning life cycle presented herein shall provide a cognitive aid rather than a rigorous set of principles. We aim to capture the crucial aspects and commonalities of real-life machine learning applications. The machine learning life cycle's steps are introduced as an overview in Figure II.1 with further detail being provided in the subsequent paragraphs.

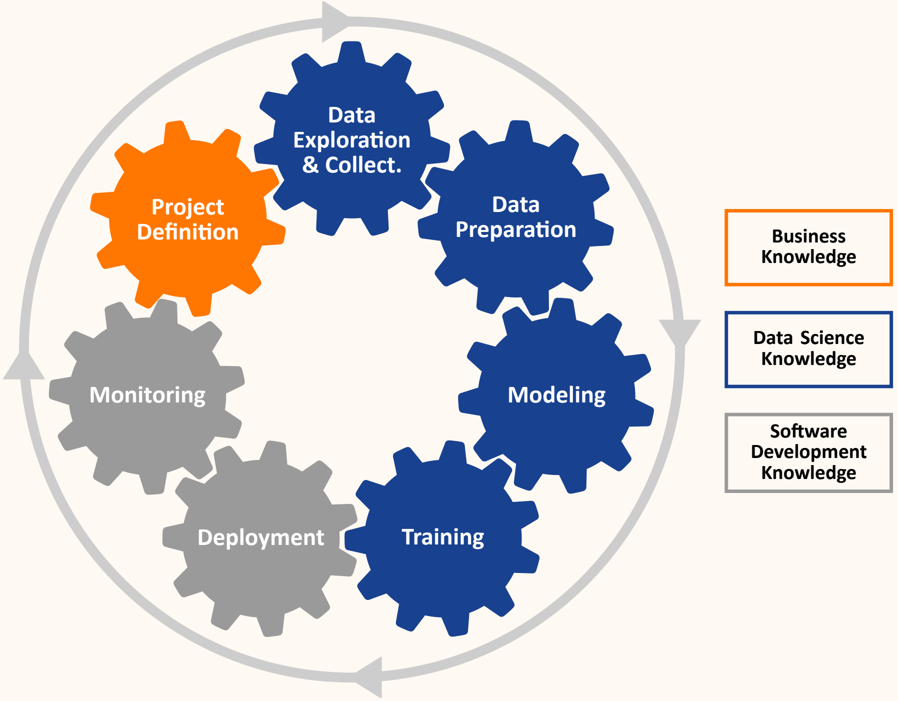

Figure II.1: **A machine learning project's life cycle.** A project's individual steps are coarsely color-coded according to the area of competence most relevant to them. The cycle is entered at *project definition*.

Project Definition

The pivotal question for any machine learning task is whether it is advisable to use machine learning in the first place. As the contents of this chapter will demonstrate a machine learning application entails unique aspects of great complexity which may be elided altogether by the usage of classical programming techniques if the problem at hand so allows.

Problems that may qualify for machine learning are distinguished from classical problems of statistics or programming by the impossibility to formulate a deterministic and simple **rule-based solution** or where the exhaustive set of such rules would be unfeasibly complex and large. The rise of machine learning is not the least due to having made practical to systematically address this class of problems at all. Where sufficient quantities of data can be provided a problem's characteristics may be deduced by the application of the techniques of machine learning.

Given a task has been determined to be solved using machine learning, the subsequent project definition phase should establish which subset of components of the machine learning lifecycle will have to be implemented manually and which are available as more or less ready-to-use modules or services from the ever-growing machine learning marketplace. Many frameworks and tools are available which encapsulate interrelated functionality of a machine learning project and provide control via well-defined interfaces which reduce the exposure to technical details to the degree that is actually required for the problem to be solved.

Data Exploration and Collection

The first hands-on step in a machine learning project is to explore the available data which might be potentially relevant to the task at hand. Establishing a proper overview of one's data set is an essential prerequisite and sets the stage for the rest of the project as the choice of a model or algorithm is intimately linked to the available data.

Generally speaking, the available corpus of data must be sufficiently representative of the problem domain to allow an algorithm or a model to internalize or 'learn' its characteristics. It is hence useful to think in terms of **quality vs. quantity of input data** during data collection. The less relevant information any given piece of gathered data contains the greater the required total volume of data will be for the trained model to properly generalize to previously unseen information. While most machine learning tasks involving unstructured data such as imagery or human speech require prohibitively copious amounts of input data for a model to be trained from scratch, **transfer learning** can drastically reduce this required volume if a suitable pre-trained model is available.

After anonymizing potentially sensitive information the definitive corpus of data should be **stored in a consistent format** and should be readily **accessible and expandable**. Viable storage options include relational database systems and structured directory trees in a local file system which are often used when data is present in the form of individual files.

Data Preparation

The gathered data is then subjected to multiple steps of data preparation. Redundant or irrelevant information is removed and the data may be expanded by augmenting it with domain-specific knowledge and mathematical transformations that present the data in a way that is more 'understandable' by the model. These collective measures are referred to as **feature engineering** and aim to produce a representation of one's data which is optimal with respect to the set of algorithmic tools and models to be used for solving the problem at hand. Feature engineering is best performed utilizing a mixture of common sense and domain expertise and involves manipulations as simple as removing the color channels from images for object classification tasks and more sophisticated transformations such as dimensionality reduction.

Conditioning refers to transformations of one's numerical data such that rounding errors inherent to calculations performed by computer hardware do not build up. A processor's floating-point arithmetic

Back to ToC

exhibits unavoidable rounding errors which, although minute individually, can accumulate and produce erroneous computations and can prevent the convergence of iterative methods unless special care is taken to control their propagation. Note that conditioning is sometimes considered to be a part of the learning algorithm itself.

Adding to the above, supervised learning tasks entail additional work. An accurate set of labels must accompany the data which, if absent, will have to be prepared by manual labor. If the data corpus is sufficiently large to allow for hold-out validation a small portion of the data volume is put aside as a test set in order to periodically estimate the progress in model training by gauging the performance on the test set. If possible, another small subset of data referred to as the validation set should be reserved in order to compare different settings of model hyperparameters. Special care needs to be taken to ensure sufficiently similar statistical distributions of the data subsets, i.e. the subsets must exhibit the same characteristics. A careless sampling of the total data volume will introduce hidden biases.

Modeling

The choice of a machine learning model must be made in consideration of the data set's volume and degree of intricacy. The informal notion of capacity, sometimes called a model's complexity, describes a model's ability to internalize and express the complex relationships contained in data. An overly simplistic model, i.e. a model whose capacity is too small, is unable to grasp the characteristics of a complicated data set and will thus perform poorly which is referred to as underfitting. On the flip side, an exceedingly complex model requires much more data to work properly or it will otherwise exhibit overfitting and will internalize the data's noise.

Where the hypotheses of machine learning models are used in safety-relevant or delicate matters the introspectability of a model becomes relevant which refers to the possibility to comprehend why a model arrived at a prediction or an estimation. While it's usually of no concern how an image classifier reasons a model's decision about a company's employee leading to termination must be understood by a human. In such cases, a black-box model should be considered unacceptable regardless of how well the model ostensibly performed on historical data during its training process. The introspectability loosely correlates with a models complexity. While simpler models such as linear or polynomial regressors readily expose their internal perception of the importance of each feature, models on the opposite end of the complexity spectrum such as the family of neural networks are virtually opaque.

Training

The training process of a machine learning model follows a simple scheme: After initializing the model with a random set of its parameters the training data is used to produce optimum parameter values by solving an optimization task based on the minimization of a domain-specific metric. For supervised learning tasks the metric correlates strongly with how well the model performs on the training set while for unsupervised learning tasks the metric is of a more abstract nature.

Generally, the resulting parameters correspond to local optima, i.e. there may exist other sets of model parameters whose performance is better in terms of the metric. As it is unfeasible to scan for a global optimum the aim is thus to find a good enough approximation to serve as a solution to one's problem. Additionally, every choice of a metric entails a certain degree of arbitrariness in much the same way any attempt to precisely quantify beauty would be arbitrary. The final decision over the progress of the training process should thus be made by a human.

Where the amount of data is large the training procedure is very computationally heavy and ought to be executed on dedicated training hardware to achieve tolerable training durations. For many models and types of data the majority of involved computations consists of dense matrix arithmetic which is better dealt with by GPUs than by CPU compute nodes. The bottleneck for these operations is the memory bandwidth, i.e. the speed at which data can be read and written to and from main memory. A GPU's memory bandwidth exceeds that of a CPU at least tenfold while it also possesses advanced techniques to hide the latency of memory accesses. It has thus become standard practice to train on dedicated GPU hardware when working with large amounts of data or complex models, notably any flavor of neural network.

In stark contrast to the training itself the evaluation of a trained model is readily performed by standard commodity hardware. There exist many services that allow to perform the training step on remote hardware rented for the duration of training. Speaking in modern terms, one is able to **train in the cloud** while deploying a service based on the trained model independently.

Deployment

As with any other business application the ultimate objective is to create a service available for consumption. With regards to a machine learning project this stage of deployment refers to **making the model available within its target environment**. To this end, a wrapper application is set up which exposes the trained model's acquired wisdom through a well-defined interface that can be subsequently used to return predictions. As application deployment is generally well understood and not specific to machine learning we will keep this section accordingly brief and refer the reader to the existing literature for further reading.

A sizeable fraction of the plethora of technological novelties pertaining to machine learning is related to the deployment process. Many vendors of machine learning solutions provide custom tools and solutions which allow the user to deploy a machine learning model with little effort. Of the different strategies containerized deployments are the most popular due to their light weight, the ease with which the application can be moved around, and their inherent ability to scale. Important aspects of a deployment solution to consider include the **latency** at which requests can be served, i.e. the delay between a query and the response containing the model's prediction. **Auditability** may be desirable where information about model accesses and its usage patterns are of interest. Where the user base of one's application is expected to grow over time **scalable** deployment solutions become relevant.

Monitoring

Whenever the deployed model is intended to be used continuously for an extended period of time, as opposed to a one-shot usage scenario, it will necessarily have to be monitored. Unless working in a controlled environment changes in the input data's characteristics constitute a serious threat to the integrity of the whole application. Such changes, called **concept drift** or data drift, refer to a systematic shifting of the input data's distribution over time and may be produced by a multitude of causes, such as wear of mechanical or eletronic components or changes to the environment, i.e. alteration of hidden variables whose variation had not been foreseen at the time of modeling.

Whichever circumstance may be the reason, concept drift causes the implicit and explicit assumptions made during modeling and training to no longer hold and will cause the model's performance to deteriorate over time with deleterious consequences for every dependent and user of the application.

A periodical sampling of the input data can be used to ascertain the degree of data drift and whether readjustments to the model or full retraining may be required. Where the model is continuously learning from newly gathered information, i.e. when dealing with online learning systems, the model may be probed with a stored set of labeled data to determine if it is still well-behaved or whether it has silently adjusted to rogue input data.

The SAP Machine Learning Landscape

The SAP machine learning landscape comprises many technologies and frameworks encapsulated into separate applications and frameworks. They provide the means to solve many common and recurring machine learning problems using different approaches with differing degrees of exposure to technical details.

Owed to the rapid and heterogeneous development machine learning has experienced in the past decade, many related ideas and technologies have been developed in separate environments. In their pursuit to embrace machine learning, SAP has, in part, attempted to develop their machine learning software internally, has externally acquired existing tools, and has integrated uprising players in the field. The SAP machine learning landscape is therefore highly affected by this compartmentalization of technologies and rather than exhibiting a well-thought-out partitioning of the full spectrum of possible machine learning problems to solve, its organization is reflective of its own organic development and thus certainly is imperfect. This chapter aims to provide a structured overview of the existing SAP machine learning technologies to find a suitable solution to a machine learning problem within the SAP cosmos.

The following list summarizes the relevant technologies.

- SAP Data Intelligence aims to provide an end-to-end solution covering the full machine learning life-cycle including all relevant aspects of a data science workflow.
- SAP Leonardo Machine Learning Foundation is a cloud-based platform offering consumption of configurable services and deployment of custom Tensorflow models.
- SAP In-Database Machine Learning comprises libraries implementing native execution of machine learning algorithms on SAP HANA systems and capabilities to interface external Tensorflow models.
- SAP Conversational AI is a framework for the synthesis of chatbots. The bots use a blend of rule-based reasoning and machine-learning techniques to generate replies from natural language.
- HANA Spatial Services comprise services and applications focused on the collection, processing, and consumption of georeferenced data. Machine learning techniques are applied for object recognition in satellite and drone imagery.

SAP Data Intelligence

SAP Data Intelligence constitutes a fully-featured development environment for machine learning and data science intended to cover most aspects of the machine learning life-cycle. SAP Data Intelligence is an extension to the well-established SAP Data Hub and introduces new features especially useful to a machine learning environment such as the ability to work within Jupiter notebooks which are pervasively used in machine learning.

Back to ToC

Further notable features include data science tools to explore and prepare data, built-in versioning of source code and data, and automated logging of model metrics. The retraining, testing, and deployment of finalized models are performed from a central cockpit. Monitoring services are available to infer precise information about how the model is used and how it performs 'out in the wild' to facilitate further fine-tuning of one's pipeline. Data Intelligence allows to integrate or import data from either arbitrary, external sources or existing SAP systems and provides the option to explore and visualize the data from within the Data Intelligence UI itself.

Data Intelligence includes a graphical workflow modeling tool which is used to incrementally define the project's flow of information. The modeler provides a clear and intuitive overview of the individual constituents of the full pipeline, an exemplary snapshot of which is given in Figure III.1. In addition, according to its roadmap, SAP Data Intelligence will eventually incorporate the SAP Leonardo Machine Learning Foundation inheriting all its features and consumable services.

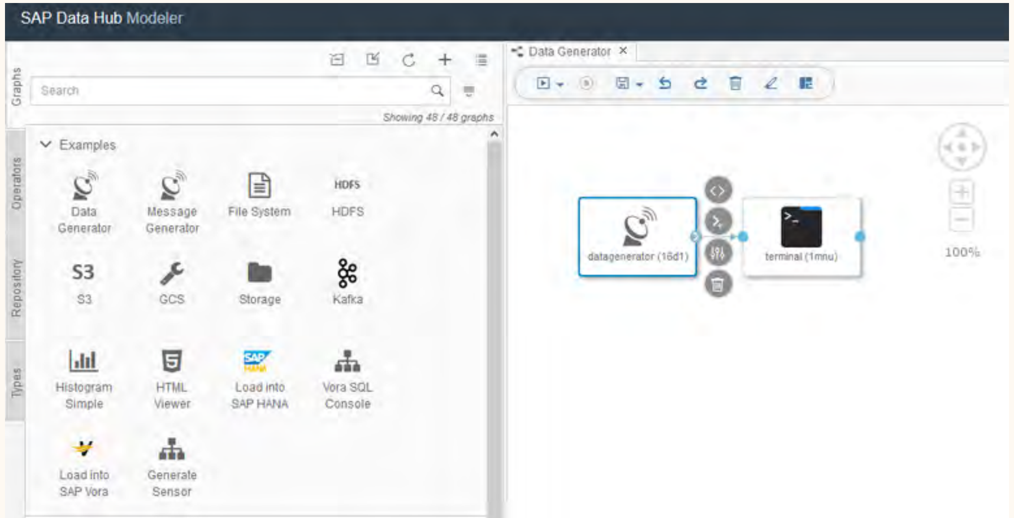

Figure III.1: **View of the Data Hub modeler.** Individual nodes represent steps of processing and can be freely interconnected, incrementally constructing a machine learning pipeline graphically.

As of the writing of this report, no Data Intelligence trial is available free of charge but has been announced to be delivered alongside the next major release. SAP Data Intelligence is available on-premise or as a cloud version via a paid subscription to SAP Cloud Platform. The SAP Data Intelligence openSAP course can be browsed to gain insights into usage and possible application scenarios without cost. Current information about the relationship between Data Intelligence, Leonardo Machine Learning and related technologies can be retrieved from the Data Intelligence FAQ.

SAP Leonardo Machine Learning Foundation

The SAP Leonardo Machine Learning Foundation (MLF) is a cloud-based platform for machine learning which provides several machine learning products loosely separated into two groups. The pre-configured **consumable services** allow to tackle certain classes of universal and widespread machine learning challenges at a comparatively low dosage of exposure to technical details. The consumable services are tailored to suit generic and recurring problems from the realms of image processing, speech processing, and document processing. Exemplary use cases include object classification in images, sentiment analysis in documents, and conversion from text to speech and vice versa.

In contrast, the second group of machine learning products provides direct interfacing for Tensorflow models. These **model-based services** are aimed at savvy machine learning engineers who seek a custom solution to the particular problem at the expense of higher development effort. The model-based services allow the deployment of Tensorflow models of an arbitrary origin and include the option to train a vanilla model on dedicated hardware in the cloud.

In general, working directly with the machine learning models provides significantly more control over the implementation details at the cost of proportionally increased complexity. If your specific challenge fits one of the generic classes of machine learning problems detailed below, the usage of the corresponding consumable service is to be preferred. Conceptualizing, implementing and tuning a custom machine learning model involves a considerable amount of time and requisite knowledge, as elucidated in the previous chapters.

The subsequent sections provide an overview of the most notable services. Please take note, that, as mentioned above, the MLF will eventually cease to exist as a standalone bundle of services and will be incorporated fully into SAP Data Intelligence.

Consumable Services

Image Processing Services

Several ready-to-use image processing services are available and include **face-recognition**, which returns the position and size of faces' bounding boxes in the image. **Image classification** categorizes images according to their content by computing a measure of affiliation for a given set of 1000 generic categories. This algorithm can be customized to classify objects according to a different set of categories.

Optical character recognition and **scene text recognition** are used to extract arbitrary text from document scans and images, respectively. **Image feature extraction** retrieves the position and size of various geometric shapes of interest in the image and is typically used as a precursor for more advanced applications such as optical character recognition and similarity analysis.

Text Processing Services

Text and document processing services solve common tasks such as **language detection** and can perform simple **translations**. **Topic detection** extracts relevant keywords from a text to determine the main topic and possible related subjects.

Analogously to image feature extraction, **text feature extraction** characterizes a document by distillation of a generic, abstract feature vector which may be used for other dependent tasks, such as similarity analysis.

Speech Processing Services

Speech processing services translate between audio-streams of speech and written text and v.v. **Speech-to-text services** can be used to generate subtitles for videos or to establish voice-based controls, among other things. The complimentary **Text-to-speech services** attempt to recreate natural-sounding human speech from text.

Model-Based Services

Bring-Your-Own-Model Service

The Bring-Your-Own-Model (BYOM) service, fittingly, provides support for interfacing custom Tensorflow models, which must be developed and trained separately. An API is automatically generated for the model using the Tensorflow Serving toolkit and is made available as a so-called model server. The model server can henceforth be called upon to retrieve information and predictions from the model.

In contrast to current state-of-the-art machine learning platforms the BYOM service limits model deployment to the MLF's shared cloud infrastructure and requires the usage of Tensorflow models. Provided a trained model is available, the workflow focuses on quick deployment with a very low additional effort required to create a fully functional application. Due to the limitations pertaining to the advanced aspects of the machine learning lifecycle like scaling or automated load balancing the BYOM service is suitable only for quick prototyping of projects, proof-of-concept solutions, and small scale applications with predictable performance requirements.

In addition to a Tensorflow model, the implementation of a client application is required which establishes the connection to the model server and performs any data conversions between the user's input and the Tensorflow model's expected numerical data format. The communication between client and model server is performed via REST- or RPC-API calls, depending on the configuration of the model server. Although the application host may be chosen freely, SAP suggests running the application in its Cloud Foundry environment which may benefit the integration with other existing applications.

Train-Your-Own-Model Service

Complementing the Bring-Your-Own-Model service the Train-Your-Own-Model (TYOM) service may be used to perform a model's actual training in the cloud on more powerful hardware than may be available to you in your development environment. To speed up the training phase utilizing the TYOM service the model alongside the labeled training data need to be provided.

Some machine learning models and architectures, chiefly among them neural networks, are notorious for high computational performance requirements during training. A full training run for a small model paired with moderate amounts of training data may easily exhaust a modern desktop machine for many hours and up to days.

Luckily, much of the training's work can be distributed across multiple compute nodes and all major machine learning frameworks provide GPU support, including Tensorflow. This circumstance allows you to design your models locally and commit them to the cloud for rapid training on suitable hardware using the TYOM service.

Configurable Model Services

The configurable model services constitute a half-way solution between the model-based services and the consumable services. Akin to the consumable services no explicit handling of models is required, with the additional feature that the generic model running in the background can be adjusted to one's use case through training on labeled data.

This is achieved by a technique referred to as **transfer learning** which uses models which have been pretrained to recognize the training data set's common generic features. The model may then be exported to be used in specific application scenarios by performing a final training step using domain-specific data while keeping the model's knowledge by retaining previously learned parameters.

Transfer learning's canonical example is the customizable image classification where a complex convolutional neural network is trained to classify images from a large generic data set. The neural network may then be tuned to a specific data set by training only the last few layers while leaving the others frozen. It is understood that the earlier layers of a neural network refer to generic and problem independent features and will hence internalize structural information relevant to a wide variety of objects. Rather than relearning this information repeatedly for every new data set it is advisable to reuse a pre-trained generic model. This can dramatically reduce training times and the required amount of labeled data.

Many of the consumable services listed above ship with an adjustable pendant which is tunable in the above manner. The available configurable model services can be explored at the SAP API Business Hub in a sandbox environment.

SAP In-Database Machine Learning

Classically, to train or query machine learning systems the data is required to be moved to and from the model while the model is typically run as a service on dedicated hardware. This poses a problem if the data volume is exceedingly large or whenever data privacy policies apply. Moving the model closer to its data sources has thus become a regular practice for machine learning engineers.

Naturally, the aspired ideal configuration has its data sources and its compute nodes share the same system. SAP HANA comprises such a system and includes libraries that implement machine learning routines natively on the database. Such techniques, referred to as **in-database machine learning**, can dramatically improve query latencies and allow to facilitate a set of machine learning and data analytics routines without the usage of external frameworks or programming languages if so desired.

Libraries that implement routines executed natively on SAP HANA systems are referred to as **Application Function Libraries**. Concerning machine learning the most important are the Automated Predictive Library, the Predictive Analysis Library, and the External Machine Learning Library.

Predictive Analysis Library

The **PAL**, or Predictive Analysis Library, is a library of machine learning and statistics routines implemented natively for SAP HANA systems. The PAL includes many of the most common algorithms for clustering, classification, time-series forecasting, and regression as well as general statistics routines. The collection of algorithms is aimed at predictive analysis for business contexts. Provided a current release, these routines ship with the SAP HANA system and can be called from within a standard development environment using SAP HANA SQLScript. An exhaustive list of algorithms can be found in the PAL reference.

For data scientists with different machine learning backgrounds, the PAL provides **Python and R Client APIs** for SAP HANA systems. They constitute wrappers around the PAL algorithms and allow working in the respective language directly on the SAP HANA system using the HANA dataframe abstraction. Optionally, data can be exported into regular Python and R dataframes.

While the PAL is very straightforward to use a possible downside is its fixed implementation of the algorithms. The underlying models cannot be adjusted or tweaked beyond the parameters which their respective interfaces expose. Thus working in-database with custom models is impossible and requires a different approach.

Automated Predictive Library

The Automated Predictive Library, or **APL**, is another application function library and is related to the PAL with regards to its usage in a machine learning context. The APL aims to automate the process of data preparation and filtering, proper encoding, and choice of model and strives to be applicable by non-technical users.

In contrast to the PAL, which exposes a rich set of algorithms and models and their hyperparameters and settings, the APL constitutes a black-box approach whereby the user is only asked to provide data and a minimum of configuration while the APL takes care of everything else by the application of the proper algorithms. Naturally, the scope of the APL is limited to simpler classification and regression tasks and provides general-purpose data mining capabilities rather than advanced machine learning features.

A listing of technical prerequisites and an exhaustive functional description of the APL can be found in its library reference.

External Machine Learning Library

The External Machine Learning Library, or **EML**, is an application function library complementing the machine learning capabilities of the predictive analysis and the automated predictive libraries. It provides interfacing of trained Tensorflow models with SAP HANA systems via remote procedure calls to the corresponding model's server. As elucidated previously Tensorflow models are usually hosted using Tensorflow Serving, Google's model deployment system. Live model servers are queried by Tensorflow Serving clients via gRPC, Googles RPC framework, which offers server and client implementations for many programming languages. Given this background, the EML may be conceived as a Tensorflow Serving client implementation for SAP HANA allowing to retrieve inferences and predictions from trained models using regular SAP HANA SQLScript.

In contrast to the actual in-database machine learning procedures provided by the PAL, a model server is a separate entity, inevitably external to the SAP HANA system which requires communication to leave the database. Nevertheless, as the RPC-based communication is agnostic towards the location of the endpoint, hosting the model on the database's host machine will partly retain the advantages related to the locality of data and model. Details and instructions are listed in the EML reference.

SAP Conversational AI

Introduced into the SAP machine learning landscape in 2019 Conversational AI is an acquired software toolkit to build chatbots. Chatbots are a conversational frontend connecting the user with the information he or she is seeking using natural human language. As such, chatbots are subject to the principles of user interface design and hence are science and art at the same time.

Although being part of the rapidly developing machine learning cosmos, the basic scheme of operation of chatbots has not undergone any significant changes since their initial inception.[1] The chatbot attempts to classify the user's input phrase according to a finite set of preset categories referred to as **intents**, which are specific to the application. Depending on the intent, the chatbot will then either initiate an action, reply with information, or ask a question to narrow down the search, prompting the user for further input and restarting the cycle.

Intent classification has historically been performed by simple word mapping, whereby the user expression is matched against a list of words, each of which is assigned to one or more intent categories. Advancements in natural language processing allow recognizing intents at more depth using classical machine learning like Bayesian classification or neural networks tailored to language processing. These modern methods allow extracting more fine-grained aspects from the user input which greatly help the chatbot choose an appropriate reply. **Sentiment analysis** distinguishes between different moods of the user input and may provide a quick lead to an angry customer while another, more even-tempered user might prefer more general information about the same topic. Similarly, the **semantic type** of a user expression helps distinguish statements from requests and will quickly identify a question as such, even in the case of missing punctuation.

As far as machine learning is concerned, intent analysis is an ordinary classification problem that hence inherits its quirks and issues to the development of a chatbot. A domain-specific sophisticated chatbot will require large amounts of labeled data. Without it, the final product will amount to little more than a gimmick as useful as the simple flowchart it represents.

As suggested above, due to the complexity of the human language the entry barrier to a capable chatbot is fairly high. Nevertheless, even simple chatbots may be of use as digital receptionists, prompting the user for more detailed information before forwarding to a human agent[2]. Other chatbots present possible answers to their questions for the user to choose from as an alternative to traditional site navigation[3]. State-of-the-art chatbot creation software hides many of the technical details and allows a developer to incrementally build and improve a chatbot using a fairly intuitive set of tools. Although the total amount of effort required for a capable chatbot is substantial, virtually no knowledge about machine learning is required to create it.

SAP Conversational AI constitutes one such end-to-end chatbot building platform which allows developers to create chatbots from scratch and to improve them incrementally. Intents can be defined at arbitrary levels of detail using the builder-tools and labeled input phrases are used to train and improve intent recognition. Built-in analytics provide introspection about the bot's usage, vital for its further development. Furthermore, Conversational AI makes a point of simple integration of their chatbots into existing SAP software. SAP Conversational AI can be tested with offline bots free of charge. Extensive tutorials and walk-throughs can be found at the Conversational AI blog that aid the synthesis of a first chatbot. Deploying production-ready bots requires a license.

[1] See ELIZA, a chatbot from the 1960s, which mimicks a psychotherapist by examining the user input for keywords and parroting them back in the form of superficial questions or utterances.

[2] See, for example, ROOF AI, a real-estate customer chatbot

[3] An exemplary TIDIO chatbot can be tested here

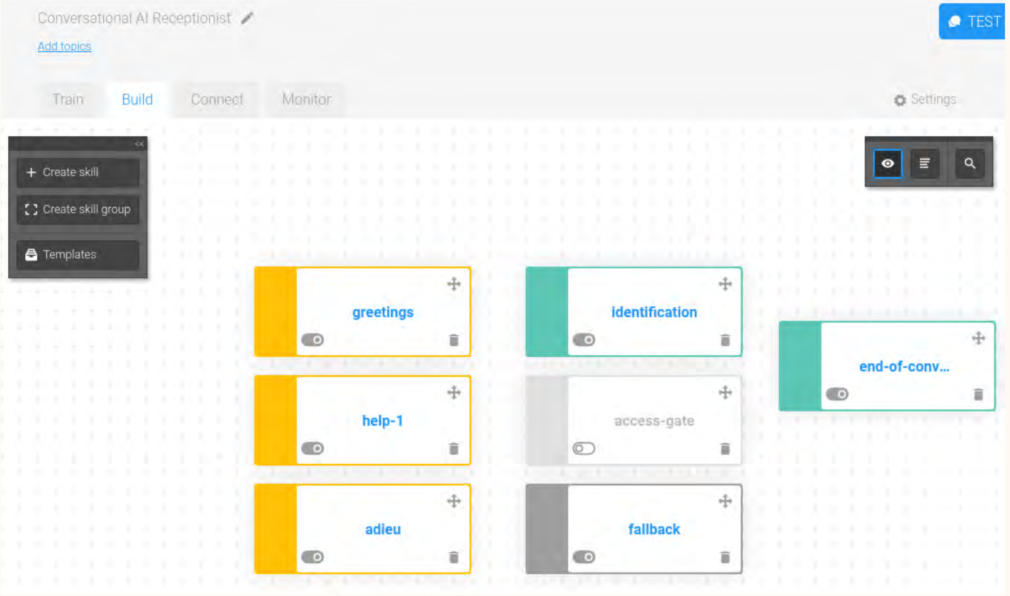

Figure III.2: **The SAP Conversational AI user interface.** The clearly structured, tab-based interface allows to build a chatbot's flowgraph, to train the underlying model on labeled data, to connect the chatbot with the outside world and to monitor the usage and performance of the deployed chatbot.

HANA Spatial Services

The HANA Spatial Services are a bundle of services which focus on the handling of geographical information. Services are available for retrieving, processing and visualizing cartographical and geographical data. Data sources include national weather observatories, map services and institutions which provide access to satellite imagery supplying copious amounts of data to explore and work with. HANA Spatial Services may be of interest to customers who seek further insight into their field of action from such information or who plan to make regular strategic business decisions based on intelligence from cartographical and geographical data. Conversely, teams whose sphere of expertise is situated within or close to the domain of geographical information systems will likely have a different, already established solution. Whether a switch to HANA Spatial Services is a viable option is best evaluated by assessing the usefulness of the available services, an overview of which is presented herein.

The **Object Detection Service** is used to scan satellite or drone imagery for objects of interest by running it through a pre-trained classification system. Provided a sufficiently high resolution of the images the objects' positions are retrieved as georeferenced locations. This data may then be used elsewhere. The **Point of Interest Service** provides the locations of and basic information about important places in the vicinity of a geographical location of interest. Among other things the locations of city theaters, parks, restaurants, and town halls may be retrieved. **Weather forecasting** and more specialized services such as **wildfire risk analysis** are also included.

Although all of the services described above are consumable via REST APIs and their results may be freely processed and utilized using any suitable tools, HANA Spatial Services provides its own application which integrates seamlessly with these services. The **HANA Spatial Services App** provides visualization of geographical data and further basic geographical information systems software functionality such as map layering, which is used to add individual pieces of data and geographical information as overlays onto a map or satellite imagery. Many of the spatial services can be consumed from within the application and their results can be promptly viewed and explored as additional map layers. Especially SAP customers who do not have knowledge about geographical information systems and who are unacquainted with the handling and visualization of geographical data will benefit greatly from the application as it provides an easily accessible gateway to interactively examining the data.

Among more features, the HANA Spatial Services App's built-in **map labeling tool** performs surface area classification of map imagery into custom classes based on a few hand-picked samples of polygon-shaped map areas labeled via the application GUI. A classification of the full map is then created by the classifier and can be exported.

A full list of services is found in the reference section of the SAP HANA Spatial Services help page. Pricing information is available at the SAP Store. A demo of the HANA Spatial Services App is presented at the SAP YouTube channel.

Recommended Best Practices

In light of the volume of information pertaining to machine learning and the copious amounts of available platforms, technologies, and software gaining an initial foothold in machine learning seems a daunting task. Assuming a machine learning problem has been systematically identified this chapter suggests heuristics and general principles by which to abide in order to arrive at a suitable solution. For SAP customers or possible clients who are considering a solution within the SAP cosmos further criteria apply and are analyzed.

General Guidelines

As outlined in this report a fully-featured machine learning application consists of a multitude of components and comprises a highly convoluted system. It is thus virtually always an impractical and prohibitively complex endeavor to build machine learning systems from the ground up. Fortunately, such an effort is unlikely to be truly required as the past decade has seen the development of a myriad of machine learning related software, frameworks, and platforms to choose from. Awareness about the shape of the machine learning landscape and about the availability of technologies related to one's problem comprises a vital step towards a polished solution.

A machine learning system can be envisioned and developed at different levels of abstraction depending on the granularity of control that is required to implement its aspired functionality. Although at the heart of every machine learning service there lies a trained model it is unadvised to work with models directly whenever it may be avoided. Many existing technologies hide much of the complexity and expose its users to a much simpler interface and oftentimes only require the user to provision reference data with which the system can adapt to the application-specific task. If the problem is relatively generic a ready-to-consume service might be available.

Examples for this are amply available in every domain: Aforementioned chatbots can be created using intuitive building platforms without mandatory exposure to the details of natural language processing. Object classification engines are based on generic classifiers and utilize transfer learning relieving the user form the need to develop a model architecture from scratch and optimize it for image recognition. Existing optical character recognition and text processing engines are, for all practical purposes, perfect and are readily available as consumable services. Frameworks for the manual creation of models provide the means to incrementally build, test, and improve a model and bring it to maturity in a well-supported development environment.

Time perusing the rich machine learning landscape for technologies even remotely relevant to the task at hand is certainly well spent. The chances that a machine learning problem has an existing solution are highly favorable and even where manual development is required the take-off point may be shifted by using existing technology. Where multiple equivalent options are available choose the most mature and

Back to ToC

widespread technology. Developing a machine learning solution is an engineering problem which profits immensely from experience and user feedback.

Cloud vs. Local

Another highly topical question to answer is whether to cloud or not to cloud. Many applications and tools are available as cloud-based services in addition to on-site versions. Where the two options are otherwise equivalent we advise, as a default, to choose the on-site version. Due to its complex and oftentimes convoluted structure the set-up or even the development of a machine learning application is performed much more easily with full administrator-level control over the development system or the deployment system. A cloud-based application cannot be inspected or manipulated beyond the means provided by its default setting which is a major disadvantage when working with any technical system. Very often significant insights are hidden just behind the curtain of the interface exposed by the application.

Additional Criteria for SAP Customers

Should SAP Technologies Be Preferred?

An obvious question relevant to SAP customers or clients working with SAP technology remains to be answered: Given the option of two apparently equivalent solutions to a machine learning problem, one SAP-based and the other external to the SAP realm, are there advantages to the former? And under which circumstances does an SAP-based solution warrant deviating from the above general principles?

Naturally, one may be enticed to prefer solutions from his own technological neighborhood over alternatives by outside providers. The knowledge and awareness about components and interfaces in one's technology stack facilitate the development of machine learning solutions that integrate well with the existing software landscape. Whether these potential advantages actually manifest in the final product remains to be determined individually for each case, however. As stated before, some of SAP's machine learning technologies are in active development while others were acquired in the very recent past and of such novel technologies no profound advantages related to integration should be assumed in blind faith.

While it is inadvisable to unquestioningly prefer SAP technologies in order to solve a specific machine learning problem, the aim to make machine learning a staple of one's technology stack does introduce a bias in favor of a SAP-based solution. It is reasonable to assume that SAP-based technologies will develop in a manner as to improve integration with other SAP-platforms much more than any alternative solutions from external providers. The adhesion between currently disjunct technologies and platforms can be expected to strengthen and grow towards a unified experience. In order to gauge whether an SAP-based solution is best it is thus prudent to distinguish one-shot machine learning projects from scenarios that entail a prolonged or continuous investment of time and effort. The latter can justify an initial investment while the former is best addressed as a self-contained and time-limited project.

Assessment of Individual SAP Technologies

Given a sufficiently recent version of HANA, the technologies related to in-database machine learning, i.e. the PAL, APL, and EML application function libraries, can be manually installed onto existing systems where they aren't by default. The libraries are available for the freely obtainable HANA Express platform and may thus be evaluated extensively free of charge before any financial commitment is required for production systems. The libraries serve a clearly defined purpose and are modular in their design which is highly favorable towards their ease of use and the ability to extend the functionality of running systems without interference with their moving parts. On the whole, we recommend giving the in-database machine learning capabilities of SAP HANA a thorough trial where a possible use case has been identified.

Data Intelligence, aspiring to be a full-pipeline machine learning solution, provides a large set of features and components which, as a whole, comprise a potent machine learning environment based on the capabilities of the SAP Leonardo Machine Learning Foundation and the SAP Data Hub. Naturally, for users of SAP systems whose machine learning problem or use case requires a granular degree of control over the machine learning lifecycle Data Intelligence should be considered a candidate technology. However, in view of the drastic changes undergone by Data Intelligence and the Machine Learning Foundation in the very recent future we judge these technologies to not yet be stable and mature enough to warrant a determined commitment to this novel technology. We advise waiting for the announced release of a free trial of Data Intelligence which, in addition to the obvious opportunity to practically assess the software, serves as a genuine signal of commitment towards and maturity of the product by SAP itself.

Conversational AI and HANA Spatial Services, the remainder of SAP-related machine learning technologies referenced in this report, are niche technologies serving a narrowly specialized purpose. Albeit being SAP technologies both may technically be used outside of an SAP context. Conversational AI, SAP's chatbot building platform, is a potent development suite that is being applied by companies to partially automate their customer support. Conversational AI is sufficiently mature with well-developed features and an intuitive interface which allow building chatbots incrementally. Thus we suggest trying it out in order to estimate its usefulness for one's purpose which is possible free of charge. In contrast, no free trial is available for HANA Spatial Services which, in addition to its particular nature, makes it difficult to ascertain a general degree of usefulness without financial commitment. We advise comparing HANA Spatial Services to other services for georeferenced data and geographic information systems software available on the market.

j&s-soft GmbH
Max-Jarecki-Straße 21
69115 Heidelberg
Germany

js-soft.com

This document has been compiled with the greatest care and the information contained therein has been taken from sources that are considered reliable. Nevertheless, no liability is assumed for the accuracy, completeness and timeliness of the information and no claims can be derived from it. The information and the opinions presented in this document are subject to change without notice.