

Full Stack AI Projects

Training, Debugging and Design Patterns

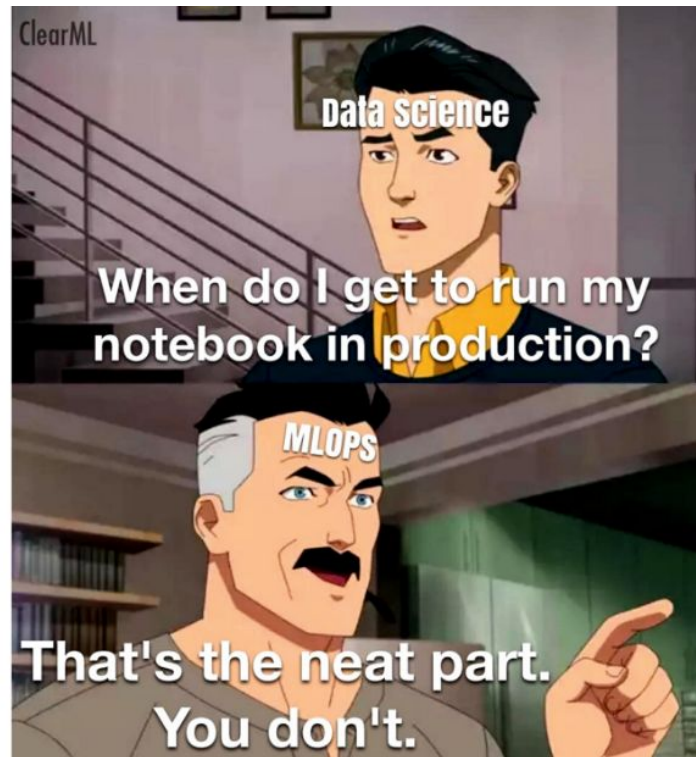


Henry Ruiz
GDE ML
@devharuiz
<https://haruiz.github.io/>

What we have discussed so far?

What is machine learning system design?

The process of **defining the interface, algorithms, data, infrastructure, and hardware** for a machine learning system to satisfy specified requirements.

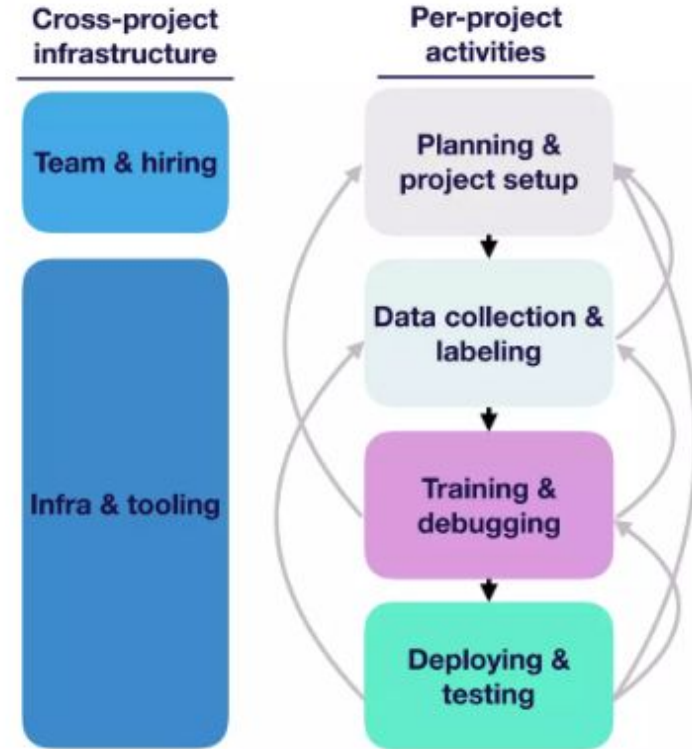


So, What is required to design and build an ML system?

What we have discussed so far?

Define a methodology

Like any other software solution, ML systems require a **well-structured methodology** to maximize the success rate of the implementation.



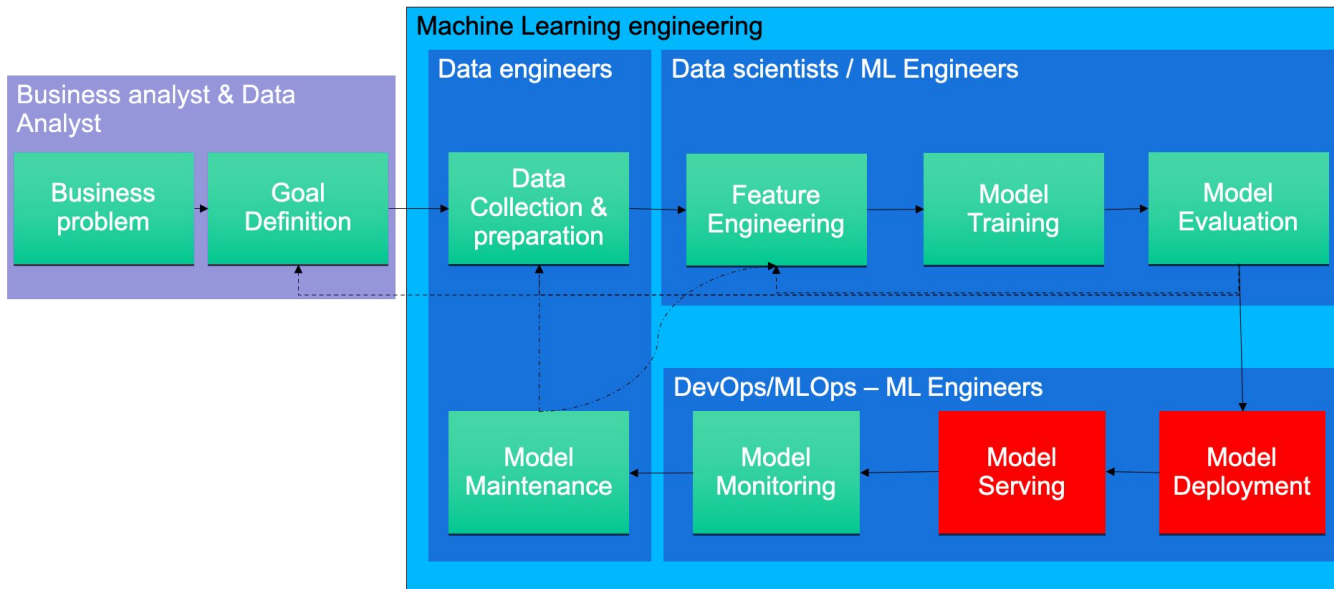
What we have discussed so far?

So, What is required to design and build an ML system?

Set up a team

Managing and leading ML and Data Science teams require unique skills.

ML teams have diverse roles.

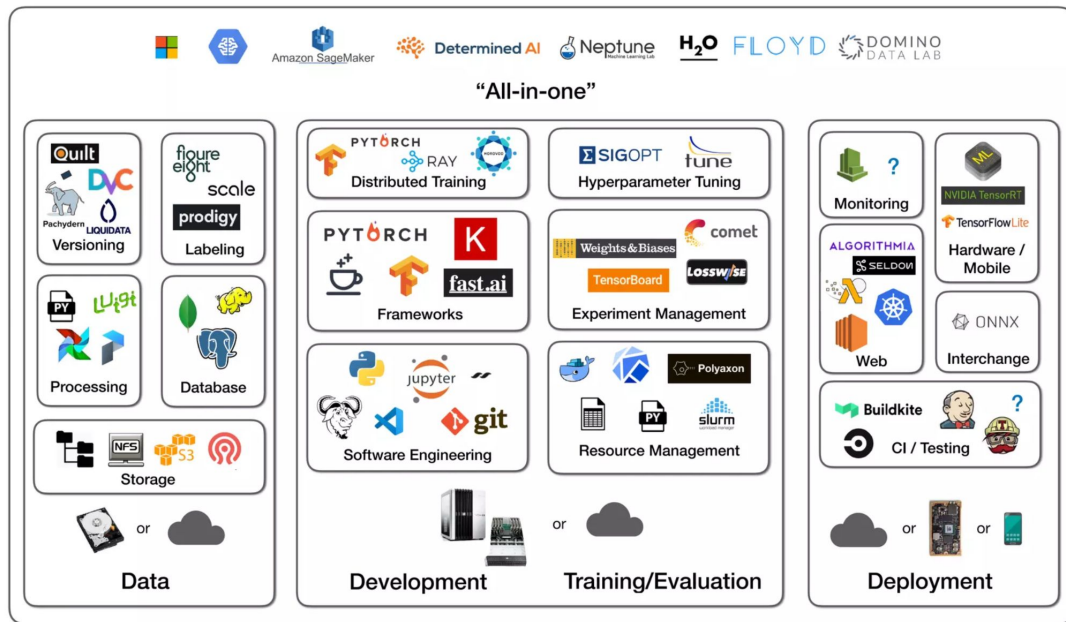


What we have discussed so far?

Define Development
infrastructure & Tooling

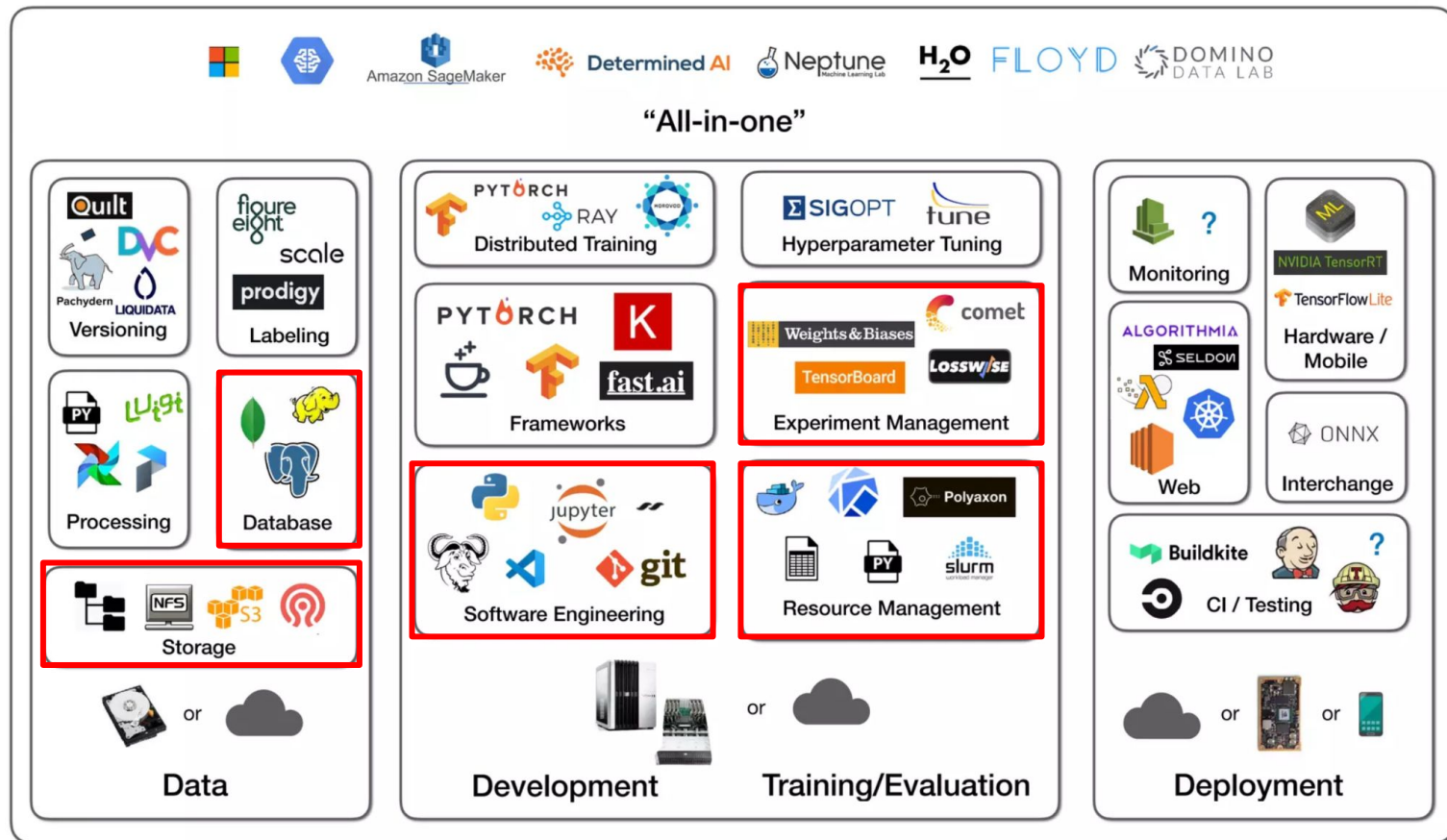
The number of **available tools to work with ML seems endless.**

Selecting the appropriate tools depends on:
the kind of problem, type of solution, deployment scenario, capacity building,
team experience, cost, hardware and software infrastructure, etc.



Data management tools

Credits: <https://fullstackdeeplearning.com/>



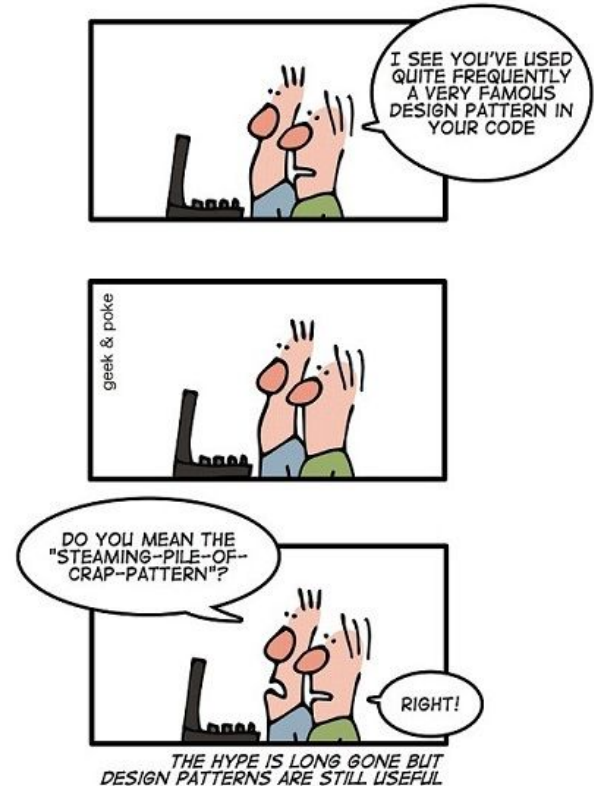
Training, Debugging and Design patterns

What are Design patterns?

In engineering disciplines, design patterns **capture best practices and solutions to commonly occurring problems** to provide **a standard approach to solve them**. They **codify the knowledge and experience of experts** into advice that all practitioners can follow.

Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

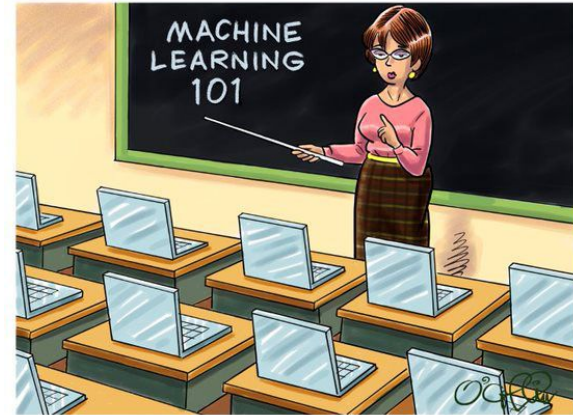
— *A Pattern Language* (Oxford University Press, 1977)



Design Patterns in ML

Developing machine learning models for production is becoming more of an engineering practice, where established ML techniques from research settings are utilized to solve business-related issues.

With the growing popularity of machine learning, it's crucial for professionals to utilize proven methods to tackle recurrent challenges.



Why Design Patterns Matter in Machine Learning?

Lets review the Machine Learning Process....

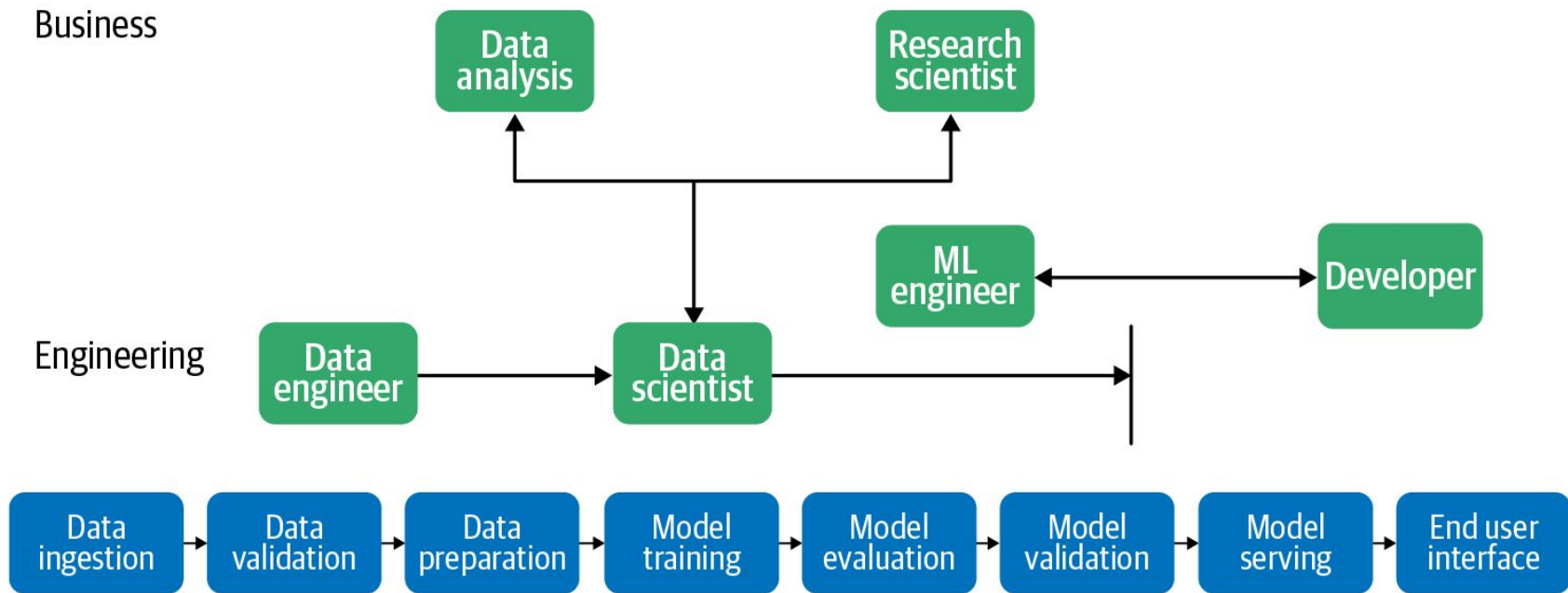
Machine Learning Process

The first step in a typical machine learning workflow is **training**—the process of passing training data to a model so that it can learn to identify patterns.

After training, the next step in the process is testing how your model performs on data outside of your training set. This is known as model **evaluation**. You might run training and evaluation multiple times, performing additional feature engineering and tweaking your model architecture.

Once you are happy with your model's performance during evaluation, you'll likely want to serve your model so that others can access it to make predictions. We use the term **serving** to refer to accepting incoming requests and sending back predictions by deploying the model as a microservice.

The process of sending new data to your model and making use of its output is called **prediction**.



Common Challenges in Machine Learning

Data Quality

The reliability of a machine learning model is entirely dependent on the quality of the data used to train it. When a machine learning model is trained on incomplete or poorly selected feature data, or when it is trained on data that does not accurately represent the population it aims to serve, the predictions generated by the model will be based on this deficient data. Therefore, machine learning models are commonly referred to as "**garbage in, garbage out.**" To ensure the quality of the data used to train machine learning models, it is essential to consider four critical components: accuracy, completeness, consistency, and timeliness.



Data Quality Components

Data accuracy is a crucial component of data quality in machine learning. It involves ensuring the accuracy of both the features in the training data and the corresponding ground truth labels. After data collection, a thorough analysis should be conducted to screen for errors such as typos, duplicates, measurement inconsistencies, and missing features.

Duplicates in the training dataset can be particularly problematic as they can cause the model to incorrectly assign more weight to these data points, resulting in inaccurate predictions.

Data completeness refers to having enough data for your model to be able to generalize to unseen data. For instance, if a machine learning model is trained to classify cat breeds, but the training data only includes ten specific breeds, the model won't be able to classify images of other cat breeds or even dogs. There's no way your model will be able to return "not a cat" if this data and label weren't included in the training dataset. Therefore, **it's crucial to ensure that your training data includes a wide range of examples for each label, as well as any possible outcomes that the model might encounter.**

Data Quality

Consistency: When dealing with large datasets, it's typical to distribute the tasks of collecting and labeling data among a team of individuals. To maintain consistency throughout the dataset, it's necessary to establish a set of guidelines for the process, as each person involved is likely to bring their own biases. For an example of inconsistent features, let's say you're collecting atmospheric data from temperature sensors. If each sensor has been calibrated to different standards, this will result in inaccurate and unreliable model predictions.

The concept of timeliness in data refers to the delay between the time when an event occurs and when it is added to the database. This delay can vary depending on the type of data being collected. For instance, error logs may take a few hours to show up in the log database, while credit card transactions may take up to a day to be reported in the system. To address this issue, it's important to capture as much information as possible about each data point, and to ensure that this information is reflected when the data is transformed into features for machine learning models.

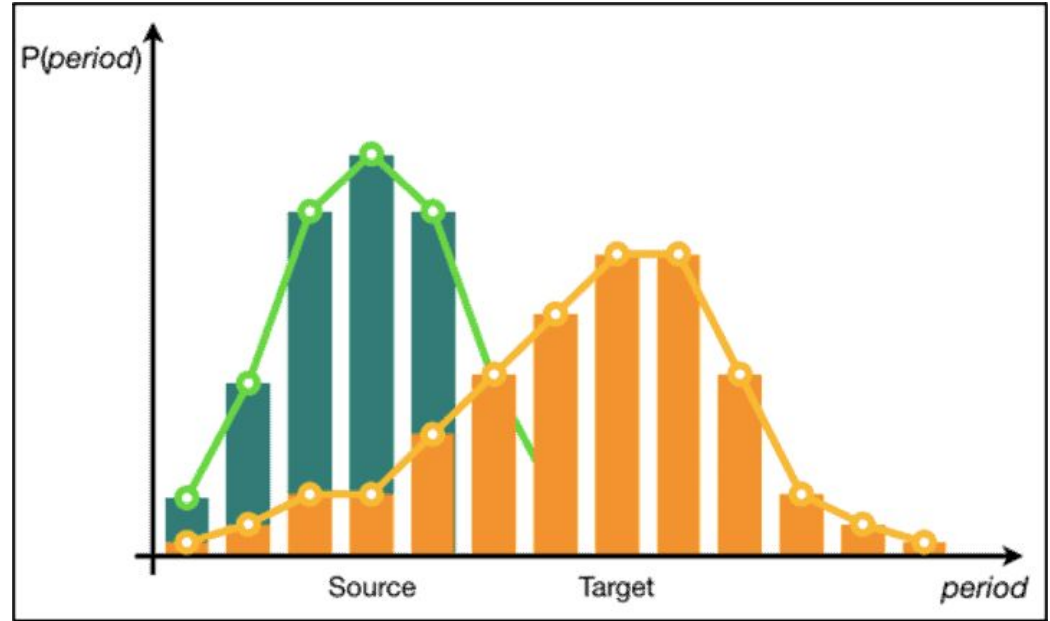
Reproducibility

The output of traditional programming is predictable and guaranteed, unlike machine learning models, which have an element of randomness. During training, machine learning model weights are initialized with random values that converge as the model iterates and learns from data. **As a result, the same model code given the same training data will produce slightly different results across training runs, making it challenging to ensure reproducibility.** This variability in results can hinder comparison across experiments, as repeated training runs are not guaranteed to reach the same level of accuracy.

The issue of repeatability in machine learning due to the inherent randomness of model training can be addressed by setting a random seed value. This ensures that the same randomness is applied every time the training is run. In TensorFlow, this can be accomplished by using the `tf.random.set_seed()` function at the beginning of the program

Data Drift

Data drift **refers to the phenomenon where the statistical properties of the target variable (or feature distribution) changes over time, which leads to a decrease in the performance of a machine learning model that was trained on an earlier dataset.** In other words, as time goes by, the data that is being collected may differ from the data that the model was originally trained on, causing the model's performance to deteriorate. This can happen due to various reasons such as changes in the business process, user behavior, or even external factors such as the weather. Detecting and addressing data drift is an important task in machine learning to maintain the accuracy and usefulness of a model over time.



Scaling

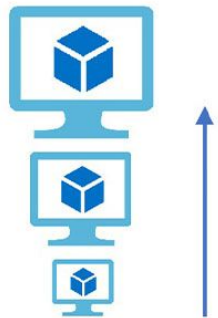
Scaling in ML systems refers to the process of **increasing the system's ability to handle more data, users, or computational resources. When an ML system is scaled, it can process more data, make more predictions, and handle more requests simultaneously.**

Scaling can be done in several ways, such as horizontal scaling or vertical scaling. Horizontal scaling involves adding more machines to distribute the workload across multiple nodes, while vertical scaling involves adding more resources to a single machine.

Scaling is important in ML systems because as the amount of data and the number of users increase, the system needs to be able to handle the additional workload without compromising performance or accuracy. Without scaling, the system may become slow, unresponsive, or even crash, which can lead to degraded performance and user dissatisfaction.

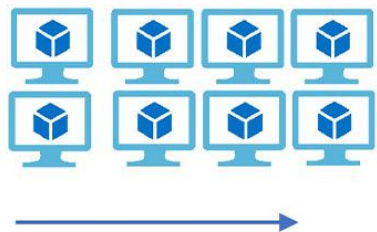
Vertical Scaling

(Increase size of instance (RAM , CPU etc.))



Horizontal Scaling

(Add more instances)



Multi Objectives

In the context of machine learning, multiple objectives refer to **the situation where a model needs to optimize multiple criteria or objectives simultaneously, instead of just one. These objectives may be in conflict with each other, meaning that optimizing for one may lead to worse performance on another.**

For example, in a classification problem, one objective might be to maximize overall accuracy, while another objective might be to minimize false negatives. Maximizing accuracy may lead to more false negatives, while minimizing false negatives may lead to lower overall accuracy. Thus, the model needs to balance both objectives and find a trade-off that satisfies both.

Dealing with multiple objectives in machine learning is an active research area, and there are many techniques and algorithms that have been developed to address this challenge. Some of these include multi-objective optimization, ensemble methods, and weighted loss functions. Ultimately, the approach used will depend on the specific problem and the trade-offs that need to be made between objectives.

Data patterns

Data is at the heart of any machine learning problem. When we talk about datasets, we're referring to the data used for training, validating, and testing a machine learning model.

*The bulk of your data will be **training data**: the data fed to your model during the training process.*

Validation data is data that is held out from your training set and used to evaluate how the model is performing after each training epoch (or pass through the training data). The performance of the model on the validation data is used to decide when to stop the training run, and to choose hyperparameters, such as the number of trees in a random forest model.

Test data is data that is not used in the training process at all and is used to evaluate how the trained model performs. Performance reports of the machine learning model must be computed on the independent test data, rather than the training or validation tests

– Machine learning Design Patterns Book

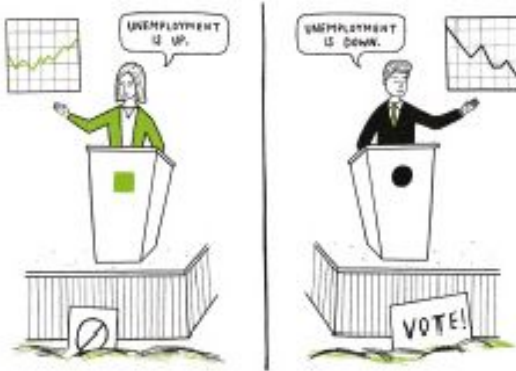
It's also important that the data be split in such a way that all three datasets (training, test, validation) have similar statistical properties



"If you don't reveal some insights soon, I'm going to be forced to slice, dice, and drill!"

Data fallacies

a mistaken belief, especially one based on unsound argument.



CHERRY PICKING

Selecting results that fit your claim and excluding those that don't.



DATA DREDGING

Repeatedly testing new hypotheses against the same set of data, failing to acknowledge that most correlations will be the result of chance.

Credits : <https://www.geckoboard.com/best-practice/statistical-fallacies/>

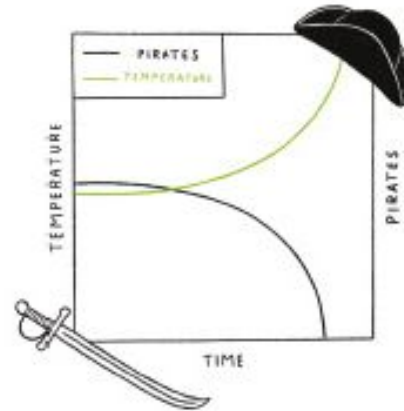
Data fallacies

a mistaken belief, especially one based on unsound argument.



COBRA EFFECT

Setting an incentive that accidentally produces the opposite result to the one intended. Also known as a Perverse Incentive.



FALSE CAUSALITY

Falsely assuming when two events appear related that one must have caused the other.

Data fallacies

a mistaken belief, especially one based on unsound argument.



SAMPLING BIAS

Drawing conclusions from a set of data that isn't representative of the population you're trying to understand.

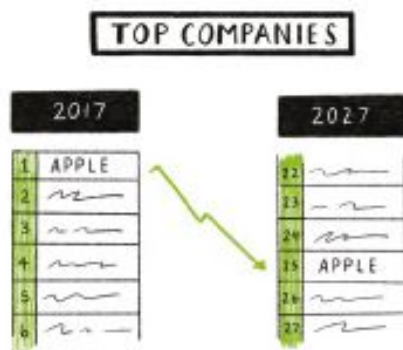


GAMBLER'S FALLACY

Mistakenly believing that because something has happened more frequently than usual, it's now less likely to happen in future (and vice versa).


Data fallacies

a mistaken belief, especially one based on unsound argument.



REGRESSION TOWARDS THE MEAN

When something happens that's unusually good or bad, it will revert back towards the average over time.

 APPLICATION SUCCESS RATE

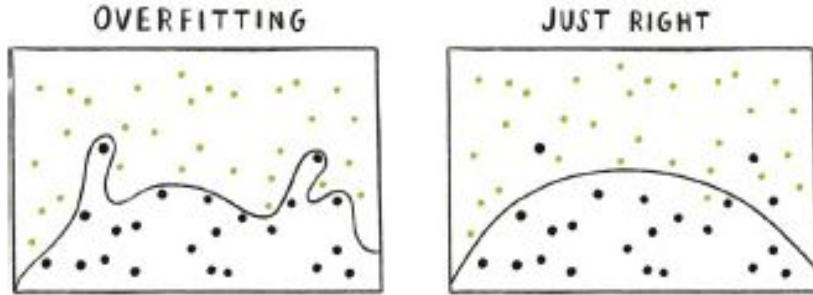
	MALE	FEMALE
SUBJECT 1	14 % (148 of 1200)	15 % (270 of 1800)
SUBJECT 2	50 % (400 of 800)	51 % (102 of 200)
TOTAL	28 % (548 of 2000)	19 % (272 of 1400) ??

SIMPSON'S PARADOX

When a trend appears in different subsets of data but disappears or reverses when the groups are combined.

Data fallacies

a mistaken belief, especially one based on unsound argument.



OVERFITTING

Creating a model that's overly tailored to the data you have and not representative of the general trend.

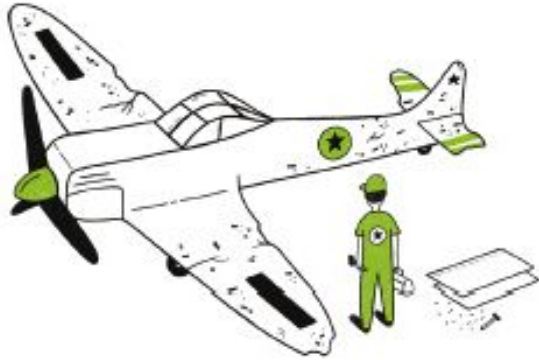


PUBLICATION BIAS

Interesting research findings are more likely to be published, distorting our impression of reality.

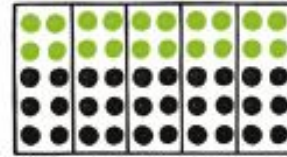
Data fallacies

a mistaken belief, especially one based on unsound argument.

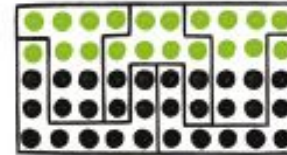


SURVIVORSHIP BIAS

Drawing conclusions from an incomplete set of data, because that data has 'survived' some selection criteria.



BLACK WINS



GREEN WINS

GERRYMANDERING

Manipulating the geographical boundaries used to group data in order to change the result.

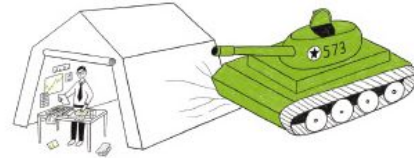
Data fallacies

a mistaken belief, especially one based on unsound argument.



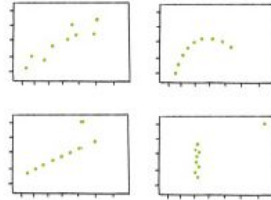
HAWTHORNE EFFECT

The act of monitoring someone can affect their behaviour, leading to spurious findings. Also known as the Observer Effect.



MCNAMARA FALLACY

Relying solely on metrics in complex situations and losing sight of the bigger picture.



DANGER OF SUMMARY METRICS

Only looking at summary metrics and missing big differences in the raw data.