# Introduction

In reinforcement learning algorithms the human is more or less excluded from the learning process as well as from the decision making when using an already trained agent. This creates new questions:

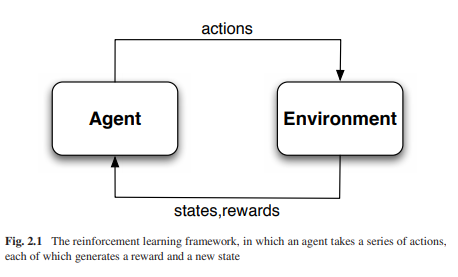
* How does my agent make its decisions?
* Does it use a different strategy than this other agent?
* Which agent is the best for my use case?
* Is another Agent more fault reliable?
* Are Agents trained with an specific algorithm better than the other ones?
* Is the agent applicable for different environments or is it a specific solution for exactly the environment it was trained on?

To answer at least some of these questions we will discuss different approaches to compare agents which were trained in the same environment but with different algorithms. Because environments and their observations and actions have values as well as their variation in time are very different from environment to environment. Comparing time series data is a big field in data science and uses many different approaches depending on the use case. Therefore our solution will also not be the one solution for every agent comparison and will be more or less specific for our use cases. Therefore this paper will focus more on the different possibilities as well as their pros and cons. If you want to use our solution for your agents we recommend to read our “concept evaluating” chapter and evaluate the discussed options for your use case.

# State of the art

## Reinforcement learning

Reinforcement learning is a subtype of machine learning. The Agent only interacts with the environment through the state in which he currently is(which we will call observation), an action he takes and the reward the last action produced. The agent only “learns” which behavior is good or bad by a reward the environment gives the agent. The agent tries to connect his actions and states with the rewards he got to understand which paths produced the most rewards in order to maximize it.



In comparison to supervised machine learning in reinforcement learning the agent the agent does not have an underlying dataset based on which he can make his own decisions. This leads to new possibilities but also new problems. First of all can supervised machine learning algorithms by definition only be as good in making decisions as the creator of the data set the algorithm was trained on.

Reinforcement learning algorithms on the other hand are often able to find new ways to solve a task because there are not bound to previous solutions. The downsides are that they often take longer to train and it is hard to comprehend the decision making by the agent.

## Literature Reinforcement Learning

-<https://doi.org/10.1007/978-1-4899-7687-1_720>

Stone P. (2017) Reinforcement Learning. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, MA

This is more non mathematical and where the picture is from

-Whiteson S. (2010) Reinforcement Learning. In: Adaptive Representations for Reinforcement Learning. Studies in Computational Intelligence, vol 291. Springer, Berlin, Heidelberg

# [Procedure](https://www.dict.cc/englisch-deutsch/procedure.html)

## Creating the datasets

The first obvious task is to get data from different models that were trained in the same environment. To generate data we use the already trained agents from the GitHub repository “arrafin/rl-baselines-zoo”. This repository implements the “stable-baselines” repository and its 7 different algorithms and provides one model for each algorithm with tuned hyperparameters. We chose the Bipedal Walker to have an complex enough environment so that different agents will have different approaches. With simpler environments it is often the case that after enough training they are all perfect. We use the “enjoy” script of the zoo (which means we let him take actions in the environment but he does not adjust his neural network) on each agent for 100´000 timesteps. To actually get the data we save the observations array with the actions array and the reward as a row for each timestep in a .csv file. This required some simple changes in the “enjoy.py” script from the rl-baselines-zoo.

## preparing the data

To analyze the data we use the well established python library “pandas”.

To make it easier to analyse the datasets we want to simplify the data. We want to do this by finding smaller subsets that are still representative for our dataset. For example this could be a subtask which has to be completed over and over again. In our Bipedal Walker example this may be a single or a number of steps. So instead of comparing the hole runs of different agents we may just want to compare how each agents makes its steps.

In order to generalize we wanted to use pattern recognition algorithms, so it could be easily used for other problems too. In the financial sector are plenty of pattern recognition algorithms used but unfortunately they are not very use full to us. They mostly explicitly search for “stock market”patterns and we can not use parts of the algorithms because their pattern recognition heavily relies on minima and maxima which are not that distinct in our datasets.