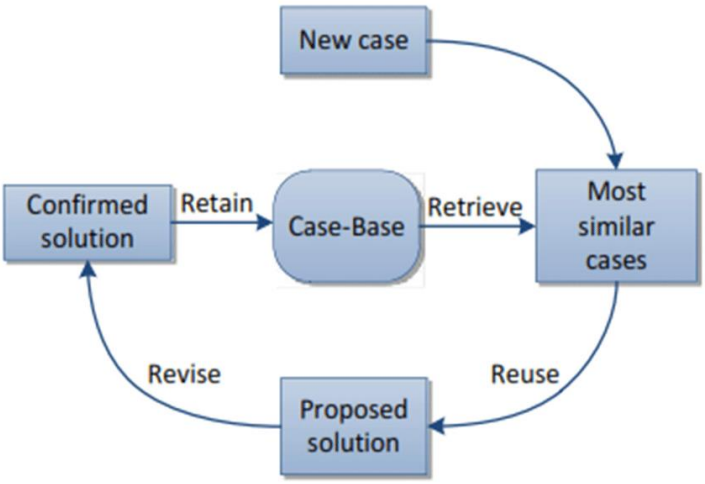
# Assignment 4

## Theory

1. First of all, case-based reasoning is more of a paradigm than an method or algorithm. It is an idea of how to structure machine learning procedures and previously learned knowledge. One can apply the algorithms of choice to implement any of the underlying parts of the CBR-cycle. CBR is characterized by using symbol-based processing, that is it handles explicitly encoded features in a domain. The CBR is defined by the CBR-cycle which is shown below




The middle component, often referred to as the Case-Base is where all previous knowledge is stored. When a new case appears, the first thing to do is to lookup the most similar looking case from the case-base. Apply the same solution to the current problem. If the problem was solved with this solution, it might also be added to the case-base for future cases. If the solution did not work, sometimes one wants tools to alter the applied solution to find a solution that works and then add the new solution to the case-base. As mentioned before, this is the general architecture and it is up to the programmer to decide on how to implement the different modules and how to handle and maintain the case-base.
To compare with other machine learning techniques it is interesting to note that this is placed in the symbolic paradigm, rather than the sub-symbolic ones that has emerged the recent years with deep learning. Where many other machine learning methods is more like a statistical machine that searched patterns, CBR is concerned with similarities from instance to instance.

2. From cognitive science we learn more about how the human brain structures and stores information and memories. One of the things that intrigued me the most when learning about cognitive science is how powerful analogies are to us humans. CBR tries to exploit the same effects with similarities that humans do with analogies. Often quite different domains can have some underlying similarities which are important for solving issues. In CBR the key idea is *Similar problems have similar solutions*, the same way we humans experience and solves problems, even in different domains. The CBR cycle is also trying to mimic the problem solving method that a human goes through, one experience the problem, then tries to find similarities with problems one has faced before, and hopefully one can apply a solution to the new problem because similar problems was solved before. At some point one revises the solutions applied, because sometimes they wont work and then you know that you cant use that same solution next time this kind of problem arrives. I also want to mention that the CBR-method is very experience based, rather than analytical and logical based. The lecturer put it like this; "*CBR is more like the mechanic than it is alike the engineer*".
3. When looking for similarities, there are mainly two different types of similarities that one could evaluate. Surface Similarities, for instance that the two cases have the same type of

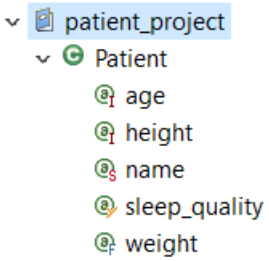
features and that the features are the same or much alike. Structural similarities is more concerned with the connections of the features, the structure of the features and the domain, how the different features are connected and what the hierarchy looks like. Sometimes it is valuable or even sufficient to compare only surface features, for example lets say a system is diagnosing cars, often we can make the assumption that these cars have very similar structures, so comparing them based on structural similarities is just unnecessary. However the surface similarities may help because one can look at which year the car was produced, how may miles it has ran since last service and what kind of car it is, which seems like insightful information for this case. If one makes a system that should handle cases that perhaps is present in different domains, then the surface similarities may be completely absent, but hopefully the structural similarities in the two domains calls for similar solutions. Its hard for me to come up with an example about this, but I can add that it is in these cases where we as humans often find analogies helpful. Two cases might have similar structures even though they are in completely different domains, with just a few or even none similar features.

4. When dealing with a number of features with different datatypes one has to encode the datatypes into comparable numbers. For instance, if one compares to strings there might be several ways to break this into numbers, for example the length of the string and the number of words, perhaps the number of words that are similar between the two cases etc. When a class has many different features, it is also helpful to weight each feature. In most domains, some features is more important than others, so they should be weighted accordingly.
5. Knowledge containers is containers that keeps all our knowledge (duh!). In the image above it is shown as the “Case-Base”, but in fact the case-base might be only one of several containers that define the knowledge of our system. For instance we might a container that has explicitly encoded rules that was put there by a programmer, or another container with rules for how to compare different cases (lets call it a similarity container) and a third that just keeps track of our vocabulary for instance. There are no restrictions for how many or how few knowledge containers there should be in a system. The more complex domain, the more explicitly made knowledge perhaps.

# Practical

This task is done in the tool “myCBR”.

## 1. Case Modelling




I created the concept of patient which has the attributes shown to the left. The age is restricted between 0 and 120. The height is restricted between 60cm and 240cm, and defined as an integer. Weight is a float restricted between 0 and 300 and supposed to be encoded as kilograms. Name is just a string which will be discriminated in the retrieval process. Sleep quality is a symbol which can be either low, medium or high.

Below are 3 screenshots of three of the ten instances I added to my case-base.

**Instance**

**Instance information**

| Name | Patient #0 |
|---|---|

**Attributes**

| | |
|---|---|
| age | 42 |
| height | 192 |
| name | Ken Ronny Børresen |
| sleep_quality | low |
| weight | 105.0 |

**Instance**

**Instance information**

| Name | Patient #1 |
|---|---|

**Attributes**

| | |
|---|---|
| age | 16 |
| height | 165 |
| name | Lise Kleiva |
| sleep_quality | medium |
| weight | 58.0 |

**Instance**

**Instance information**

| Name | Patient #9 |
|---|---|

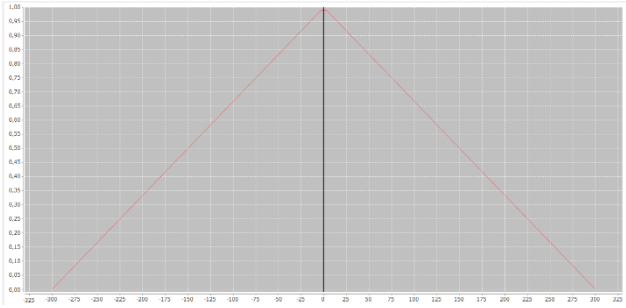
**Attributes**

| | |
|---|---|
| age | 44 |
| height | 180 |
| name | Frida Frik |
| sleep_quality | medium |
| weight | 72.0 |

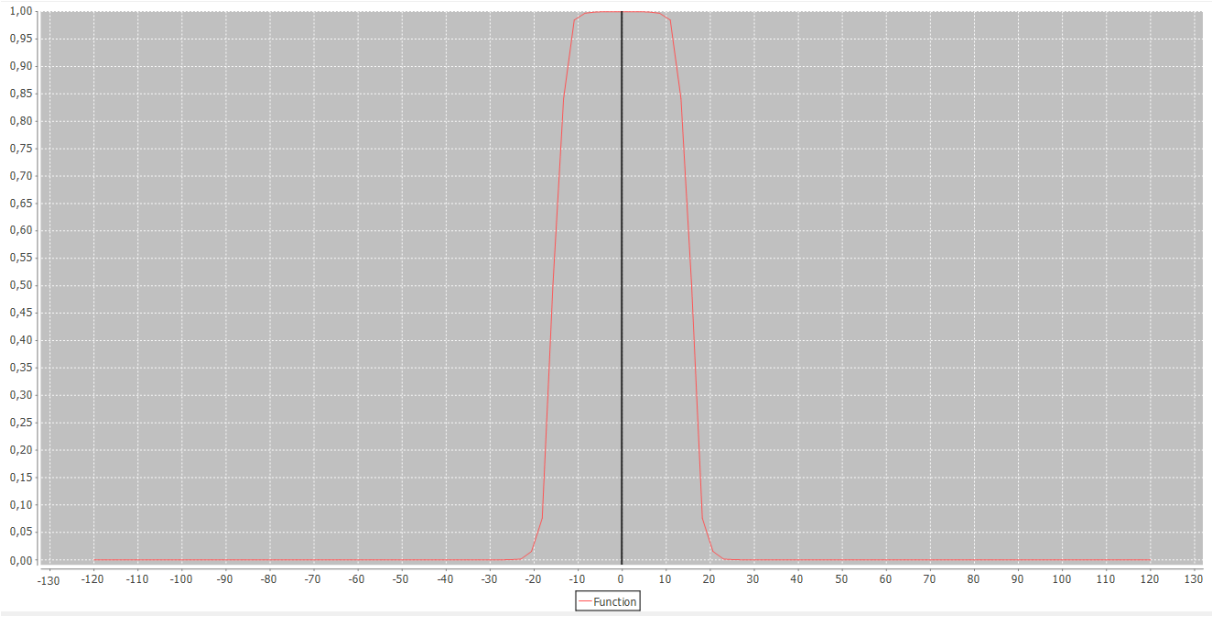
## 2. Case Retrieval

| | high | low | medium |
|---|---|---|---|
| high | 1.0 | 0.0 | 0.5 |
| low | 0.0 | 1.0 | 0.5 |
| medium | 0.5 | 0.5 | 1.0 |

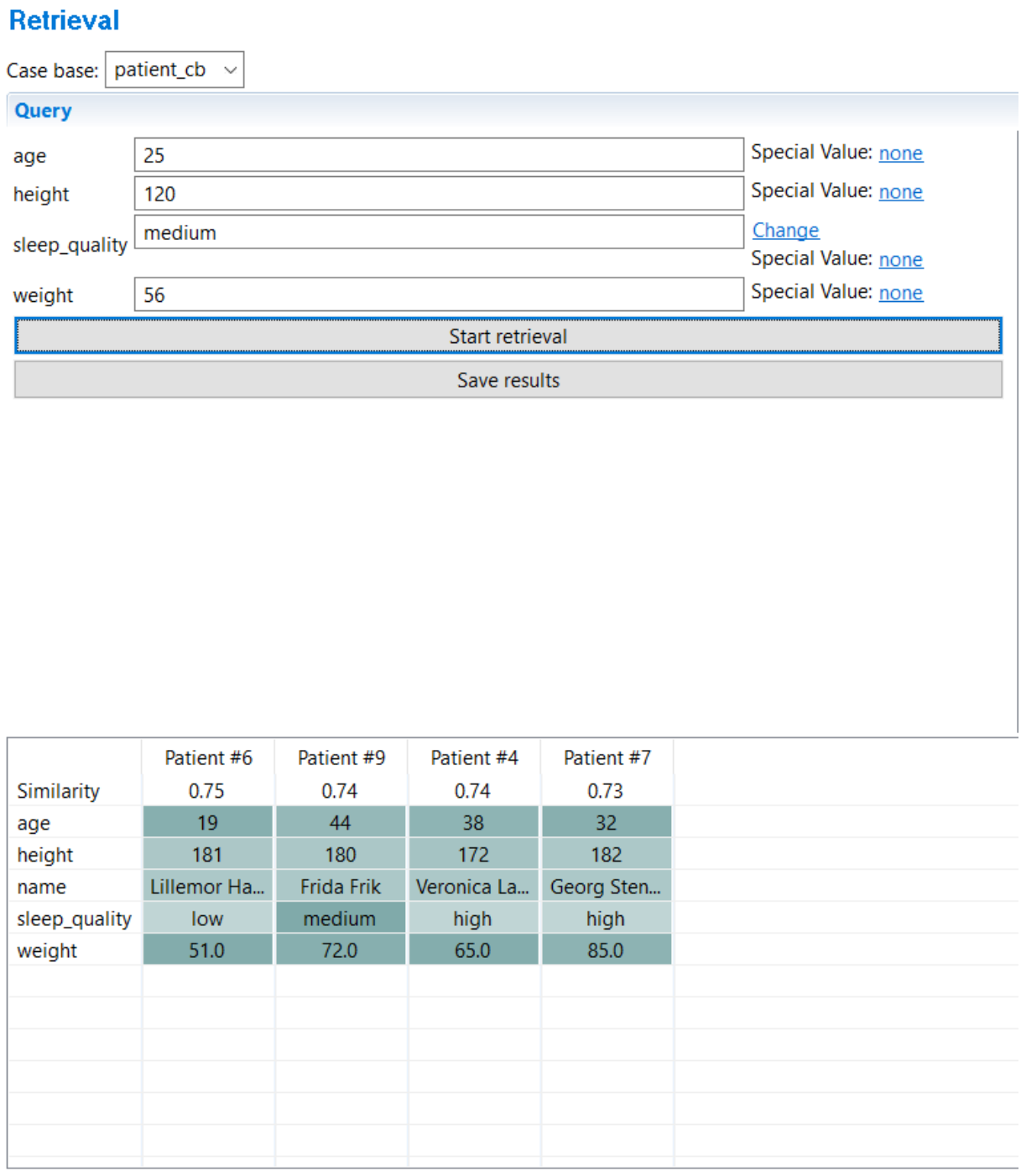
For the similarities functions there are some encoding to be done. I did a symbol-check for the sleep_quality as shown to the left. High shares no similarities with low, but some similarities with medium. So medium is somewhat similar to high and low.



The weight function is a simple polynomial one, where the similarity is 1.0 if the weigh matches exactly. The similarity is regarded lower if the weights in the two cases are far apart, this decrease in similarity is then encoded linearly as a function of the difference in weight. The height function is encoded in the exact same way. The age function is chosen a bit different, where the penalty of being in another group age is harder, but as long as you are within the same group it is pretty similar. Smooth-step at 15, image shown below.



As for the global function, I discriminated the name from being a factor. The weights is as follows; weight – 2, height – 2, age – 3, sleep_quality – 0.6.




| | Patient #6 | Patient #9 | Patient #4 | Patient #7 |
|---|---|---|---|---|
| Similarity | 0.75 | 0.74 | 0.74 | 0.73 |
| age | 19 | 44 | 38 | 32 |
| height | 181 | 180 | 172 | 182 |
| name | Lillemor Ha... | Frida Frik | Veronica La... | Georg Sten... |
| sleep_quality | low | medium | high | high |
| weight | 51.0 | 72.0 | 65.0 | 85.0 |

The image shows one retrieval done. It seems that the system put the weight and age similarities higher than the sleep_quality similarities. This is already mentioned in the weight section, so it was not a surprise. If I penalized harder for the difference of height, the result might have come back very different. This serves as an example of why experts are still needed to make these kinds of method work, because we as humans (and experts) needs to tune the system to perform well.

This system we just created might be used to for instance find out reasons for focus problems, or perhaps diagnose the reason behind fatigue problems.

The retrieve part can be executed just like in this exercise, by adding the information for the patient in question and retrieving the most similar patients in the CB. Then one should have a different KB where diagnosis or procedures are stored, so this would handle the *reuse* part. After each diagnosis or solved problem, one would come back to the myCBR and input the patient in questions details into the system alongside the applied fix.