# Testing Artificial Intelligence


**Gerard Numan**

**Abstract** In AI, the algorithm is not coded but produced by a combination of training data, labelling (concepts) and the neural network. This is the essence of machine learning. The algorithm is not directly insightful and cannot be bug-fixed directly: it is "black box development".

AI systems are used in contexts with diverse data and usage. Choice in training data and labels brings risks in bias and transparency with possible high impact on real people. Testing AI focusses on these risks. An AI tester needs moral, social and worldly intelligence and awareness to bring out the users, their expectations and translate these in test cases that can be run repetitively and automated. AI testing includes setting up metrics that translate test results in a meaningful and quantifiable evaluation of the system in order for developers to optimize the system.

**Keywords** Software testing · Software quality · Artificial intelligence · Machine learning · Test automation

## 1 Introduction

The future is AI. It has entered our everyday lives and is being used by major companies all around the world. Adaptability of AI seems endless. And yet many doubts and concerns exist. For example, in the case of self-driving cars: liability in case of accidents, wobbly object recognition and complex interaction with unpredictable human traffic participants are blocking widespread acceptance.

Some possible scary effects of AI have already manifested themselves. AI algorithms can create or enlarge bias. Like in the case of the ethnic cleansing in Myanmar, where tens of thousands of Rohingya were killed and one million fled. Already existing ethnic tension was supported by the Facebook algorithm, which

G. Numan
Polteq Test Services B.V., Amersfoort, The Netherlands



S. Goericke (ed.), *The Future of Software Quality Assurance*,
https://doi.org/10.1007/978-3-030-29509-7_10

strengthened prejudiced opinions because it was optimised to reward click-success. Negative information showed up in search results increasingly.

Every software developer and customer of AI struggles with these doubts and risks. What is a bug in case of AI and how to fix it? How to be certain that the system does the right thing with a great variety of input and users? How to get the right level of confidence? Are the results fair to all concerned? Are current developments, opinions and values reflected in the algorithm?

What are the biggest risks with AI and how to deal with them from a testing point of view?

## 2 An Introduction to AI for Testers

This chapter is a short introduction to AI and an analysis of aspects relevant to testing.

### *2.1 AI Is Black Box Development*

In AI, the algorithm, the behaviour of the system in terms of criteria, decisions and actions, are *not* explicitly engraved in the code. In non-AI development the code directly expresses the algorithm. In AI the algorithm is the product of training data, parameterisation, labels and choice of the neural network. But the algorithm cannot be found in the code. The code, the neural network, is just a, be it very essential, part of a system which produces the algorithm by training. This is the essence of machine learning.

### *2.2 Machine Learning and Neural Networks*

There is a strong analogy between machine learning and human learning. Take for example a child who learns to use a concept for the first time. The child has been told that the hairy creature it cuddles is a “cat”. Now the child sets its own neural network to work. The concept of the cat is compared to objects which aren’t cats, such as “daddy”. The neural works is finding ways to configure itself in such a way that had it seen the cat, it would classify it as a cat and not as daddy. It does so by finding differences, criteria, such as fur, whiskers, four legs, etc. But we do not know exactly what these criteria are. They might also be “hunting mice”, “purring”, or “being white”. We cannot find the concept of a cat and it’s criteria inside the brain, nor can we correct it directly in the brain.

A neural network consists of many blocks of code (“nodes”) which are arranged in layers. Each layer of nodes is connected to its top and bottom layers. The nodes

are not programmed upfront to perform specific tasks, they are empty. The nodes are merely small calculators, processing parts they have been presented by top layers, returning a calculated result. When the neural network is presented with an example in training it will systematically configure itself so the different layers and nodes will process parts and aspects of the input so the end result of all nodes will give the result that is given to the network (the label). Given two pictures, of a cat and of daddy, it will try different configuration in order to find the configuration that would determine one example to be a cat and the other as daddy. It would seek out the differences so it's configuration would come up with the right classification next time.

In this way the neural network creates models of the labels: these reflect differences between a cat and daddy the neural network has identified based on the training data.

## *2.3 Algorithm = Data + Code + Labels*

So what the system produces is an algorithm that consists of models derived from examples so it can classify and recognise input and assign these to labels. The algorithm is the product of the neural network but based strongly upon the training data (the examples) and the goals (the labels). So the algorithm is NOT the code, but the code + training data + labels. Because the algorithm cannot be identified it can also not be fixed directly. Brain surgery will not fix the child's flaws in recognising a cat.

## *2.4 Fuzzy Logics and Mathematics*

Although all the system does is calculating, produce numbers, these numbers will not produce a Boolean result: for example: "this is daddy" or: "this is a cat". The result will be the summation of all calculated numbers from the nodes and layers, each giving a number which expresses the extent to which criteria have been met as per each given label. This will hardly ever be 1 on a scale of 0–1. Next to that: it will also produce the extent to which the example scores on the other labels. So a new picture presented to the system could score "cat-ness" as 0.87 and "daddy-ness" as 0.13. The conclusion would be that the example is a cat, but it's not 100% a cat, nor is it 0% daddy.

So the end product of AI is a calculation, a probability and never a 100% certainty.

### 2.5 Development and Correction

Development of a neural network consists of developing a neural network itself, but most developers take a neural network off the shelf. Next they need to configure the neural network so it can receive the input at hand and configure labels, so examples are linked to these.

Finally the layers of the neural network can be parameterised: the calculated results can be weighted so certain results will have more impact on the end result than others. These are the main tweaking instruments developers have. If the system is not performing satisfactorily the parameters can be tweaked. This is not a focussed bug fix, correcting one case of faulty decision.

Parametrisation will influence the outcome, but each tweak will have impact on the overall performance. In AI there is massive "regression": unwanted and unexpected impact on parts of the system that are not intended to be changed.

Training data and labels are also likely candidates for influencing the system. In certain issues with AI, such as underfitting, expanding the training data will very likely improve the system. Underfitting means the algorithm has a too simplistic view of reality, for example when a cat is only classified as a furry creature. Adding more examples of a cat to the training data, showing more variety of species, races and behaviour, could help the system distinguish a cat from other creatures better.

### 2.6 Overall Version Evaluation and Metrics

When bug fixes cannot be focussed and each tweak has massive regression, massive regression testing is necessary. The question "did we fix this bug?" becomes a minor issue. We want to know the overall behaviour each time we change something. We want to know what the overall performance of the system is compared to other versions. In that overall evaluation we need to take into account the output of AI: calculated results which are not either true or false. Each result is a grade on a scale. So the end results should be thoroughly compared, weighed and amalgamated so we can decide if a version as a whole is better than another and we should use it or not. The result will be metrics calculating the value of output based on expectations and their relative importance.

## 3 Risks in AI

We'll discuss the most important risks here. These risks are typical of AI and could have serious impact on the quality of AI, it's customers, users, people and even the world. These risks should be considered before starting testing, giving clues to where to put emphasis as a tester. When analysing test results the risks should be

considered as a cause-effect analysis of unwanted outcome. This could give clues for optimising the system. For example: under-fitted systems most likely need more diverse training data, over-fitted systems streamlining of labels.

## *3.1 Bias*

The main risks with AI are types of "bias". In human intelligence we would call this prejudice, reductionism or indecisiveness. Because of limits in training data and concepts, we see things too simple (reductionism) or only from one point of view (prejudice). A high granularity in concepts could mean that the system can't generalise enough, making the outcome is useless (indecisiveness).

Significant types of possible bias in AI are discussed next.

### 3.1.1 Selection Bias

If the training data selection misses important elements from the real world, this could lead to selection bias. Compared to the real results, the polls for the last European elections predicted much higher wins for the Eurosceptic parties in the Netherlands than they did in the real election. The polls did not filter on whether people were really going to vote. Eurosceptics proved more likely not to vote than other voters.

### 3.1.2 Confirmation Bias

Eagerness to verify an hypothesis heavily believed or invested in can lead to selecting or over-weighing data confirming the thesis over possible falsifications. Scientists, politicians and product developers could be susceptible to this kind of bias, even with the best of intentions. A medical aid organisation exaggerated a possible food crises by showing rising death numbers but not numbers of death unrelated to famine and the overall population number in order to raise more funds.

### 3.1.3 Under-fitting

Training data lacking diversity causes under-fitting. The learning process will not be capable to determine critical discriminating criteria. Software that was trained to recognise wolves from dogs, identified a husky as a wolf because it had not learned that dogs can also be seen in snow. What would happen if we only get drugs-related news messages in the Netherlands?

### 3.1.4 Over-fitting

Over-fitting occurs when the labelling is too diverse and too manifold for the purpose of the AI system. If we want to see patterns and groupings, a high granularity of labels compromises the outcome, making it unusable because of its indecisiveness.

### 3.1.5 Outliers

Outliers are extreme examples that have too much influence on the algorithm. If the first cat your 1-year-old child sees is a Sphynx, a naked race, this will have a major impact on his concept of a cat and will take multiple examples of normal cats to correct.

### 3.1.6 Confounding Variables

Pattern recognition and analysis often requires combining data, especially when causal relations are looked out for. Confounding variables occur when different data patterns are associated for data analysis purposes that have no real causal relation. It has often been believed that drinking red wine could evoke a migraine attack, because drinking red wine and migraines reportedly occur sequentially. New research has shown that a migraine attack is preluded by changes in appetite, such as a craving for red wine. Drinking red wine is a side effect and not a cause of migraine!

## *3.2 Over-confidence in AI*

AI can perform some types of mental activities on a scale and with a velocity and precision that is unachievable by humans. The algorithm of AI is not directly accessible or adjustable. From this the intuition can be easily obtained that AI cannot be judged by human standards and is superior. Intellectual laziness and comfort can be an important motivation too for uncritically trusting AI. Who questions the results of Google search?

A possible consequence of over-confidence is the transfer of autonomy to an instance outside of our individual or collective consciousness. AI does not need to achieve self-consciousness to be able to do this, as sci-fi teaches us. It takes over-confidence or laziness.

### *3.3 Under-confidence in AI*

The other side of this is under-confidence. A rational debate on whether to use AI can be blurred by uncertainty, irrational fear or bias in the media (or sci-fi movies). Accidents with self-driving cars get more headlines than ordinary accidents. People are afraid to become obsolete or that a malicious ghost in the machine might arise.

### *3.4 Traceability*

With non-AI-systems the algorithm is the code. This is not the case with AI-systems so we don't know the exact criteria by which the AI-system takes decisions. Next to that it's hard to oversee the total population of training data and therefore get a good understanding of how the AI system will behave. So when the outcome is evidently incorrect, it is hard to pinpoint the cause and correct it. Is it the training data, the parameters, the neural network or the labelling? Lack in traceability fuels over-confidence and under-confidence (as was shown above) and causes uncertainty in liability (was it the software, the data, the labelling or the context that did it?) and lack of maintainability (what to correct?).

## 4 Testing AI

The key to mitigation of the AI risks is transparency. In bias we need insight into the representativeness of training data and labelling, but most of all we need insight into how important expectations and consequences for all parties involved are reflected in the results.

Building the right amount of confidence and traceability needs transparency too. Transparency will not be achieved by illuminating the code. Even if this were possible, by showing a heat-map of the code indicating which part of the neural network is active when a particular part of an object is analysed or a calculation in a layer is produced, means close to nothing. Looking inside a brain will never show a thought or decision. It could show which part is activated but all mental processes always involve multiple brain parts to be involved and most of all experience from the past.

AI systems are black boxes, so we should test them like we do in black box testing: from the outside, developing test cases that are modelled on real-life input. From there expectations on the output are determined. Sounds traditional and well known, doesn't it?

The basic logic of testing AI might be familiar, the specific tasks and elements are very different.

Traditionally requirements and specifications are determined upfront and testers receive them ready to be used at the start. In AI, requirements and specifications are too diverse and dynamic to expect them to be determined at the start completely and once and for all. Product owners and business consultants should deliver requirements, but testers need to take initiative to get the requirements in the form, granularity and actuality that they need.

The challenges with testing AI and their accessory measures from start to finish are discussed next.

## *4.1 Review of the Neural Network, Training Data and Labelling*

Static testing can detect flaws or risky areas early.

The choice for the neural network or its setup can be assessed: is it fit for purpose? What are the alternatives? For this review a broad knowledge is required of all possible neural networks and their specific qualities and shortcomings.

The training data and labels can be reviewed and assessed for risk sensitivity:

1. Does the data reflect real-life data sources, users, perspectives, values well enough? Could there be relevant data sources that have been overlooked? Findings might indicate selection bias, confirmation bias or under-fitting.
2. Are data sources and data types equally divided? How many representatives do various types, groups have compared to one another? Findings might indicate under-fitting, selection bias, confirmation bias or outliers.
3. Are the labels a fair representation of real-life groups or types of data? Do the labels match real-life situations or patterns that the system should analyse? Findings might indicate over-fitting, under-fitting or confounding variables.
4. Is the data current enough? What is the desired refresh rate and is this matched? Are there events in the real world that are not reflected well enough in the data?

## *4.2 Identifying Users*

The owner of the system is not the only valuable perspective! AI-systems like search systems are an important part of the world of its users but also of those that are "labelled" by it. The quality of an AI-system can have moral, social and political dimensions and implications so these need to be taken into account.

The users of AI are often diverse and hard to know. They are not a fixed set of trained users, all gathered in a room and manageable in their behaviour and expectations. They could be the whole world, like in the case of a search engine: an American tourist visiting Amsterdam or an experienced art lover in the field at hand have very different needs and expectations when searching for "Girl with pearl" in

the search engine of a museum. The tourist wants to know if a particular picture is for display, the art lover also wants background information and sketches.

Next to that: as the world changes, the users and their expectations could change overnight. Think of what the fire in the Notre Dame did to what users might expect when searching for "Notre Dame" or "fire in Paris". AI recognising viruses in DNA sequences should take into consideration possible mutations that occur constantly.

So testing AI starts with identifying the users or the perspectives from which output from the system will be used. This means studying data analytics on the usage of the system, interviewing process owners or interviewing real users.

## *4.3 Profiling Users*

Identifying users or groups of data is one, determining what they want, expect, need, are afraid of or will behave like, is another. What the tester needs is profiles of the users and perspectives: what is their typical background, what do they want, what turns them off or upsets them and what do they expect?

A technique to create profiles is "Persona". Key to this technique is to not think of an entire group of users but to pick one from this group and make her or him as concrete as possible. The benefit of Persona is that it makes the user come alive. It's a technique to take the perspective of a user from the inside out. For example: the Persona for American tourists could be Joe, a plumber, living in Chicago, white, aged 45, married, two children. He is not well read but loves colourful and well-crafted paintings. His hobbies are fishing and refurbishing old audio equipment. He is turned off by profound theories but likes the human side and background of things (Fig. 1).

## *4.4 Creating Test Cases*

This part is probably where most of the work is for the tester. Per user profile, input and expected output is determined. Good profiles will provide a good basis but will probably need extra information coming from research and interviews.

Identifying test cases will never be complete nor definitive: you can't test everything, also not in AI. The world and the users change so this needs to be reflected in the requirements. It starts with the most important cases; it will grow constantly and needs permanent maintenance.

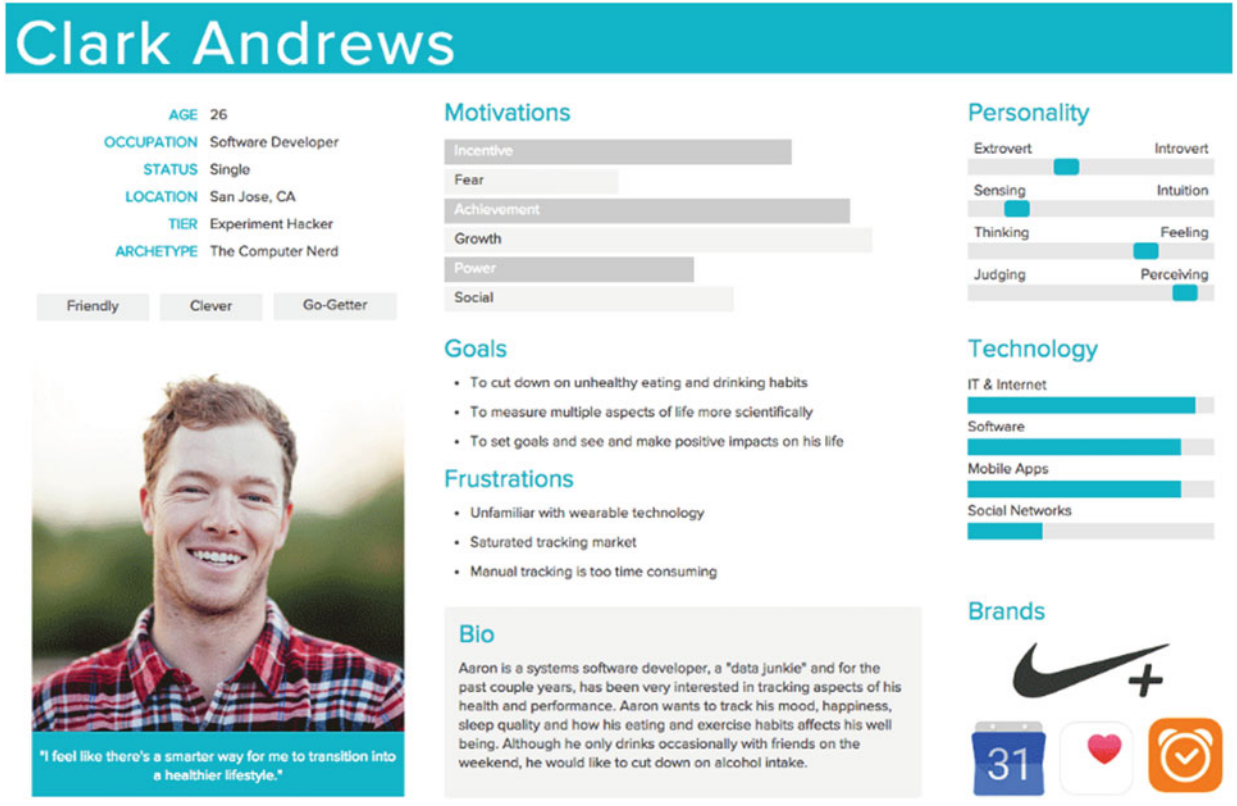



**Fig. 1** Profiling users

## 4.5 Test Data

What test data to use and whether it can be created, found or manipulated depends on the context and the availability of data from production. Data creation or manipulation (like in case of image recognition) is hard to do and sometimes useless or even counter-productive. Using tools to manipulate or create images brings in an extra variable which might create bias of its own! How representative of real-world pictures is test data? If the algorithm identifies aspects in created data that can only be found in test data, the value of the tests is compromised.

AI testers create a test data set from real-life data and strictly separate these from training data. As the AI system is dynamic, the world it is used in is dynamic, test data will have to be refreshed regularly.

## 4.6 Metrics

The output of AI is not Boolean: they are calculated results on all possible outcomes (labels). To determine the performance of the system, it is not enough to determine which label has the highest score. Metrics will be necessary.

Take, for example, image recognition: we want to know if a picture of a cat will be recognised as a cat. In practice this means that the label "cat" will get a higher score than "dog". If the score on cat is 0.43 and dog gets 0.41, the cat wins. But the small difference between the scores might indicate fault probability.

In a search engine we want to know if the top result is the top 1 expectation of the user, but if the top 1 result is number 2 on the list, that sounds wrong, but is still better than if it were number 3. We want to know if all relevant results are in the top 10 (this is called precision) or that there are no offensive results in the top 10.

Depending on the context we need metrics to process the output from the AI system into an evaluation of its performance. Testers need the skills to determine relevant metrics and incorporate them in the tests.

## 4.7 Weighing and Contracts

The overall evaluation of the AI system also has to incorporate relative importance. Some results are more important than others as is with any testing. Think of results with high moral impact like racial bias. As part of designing test cases their weight for the overall evaluation should be determined based on risks and importance to users. Testers need sensitivity for these kinds of risks, being able to identify them, translating them into test cases and metrics. They will need understanding of the context of the usage of the system and the psychology of the users. AI testers need empathy and world awareness.

In the movie *Robocop* officer Murphy had a "prime directive" programmed into his system: if he would try to arrest a managing director of his home company, his system would shut down. AI systems could have prime directives too, or unacceptable results, like offensive language, porn sites or driving into a pedestrian. We call these "contracts": possible unwanted results that should be flagged in the test results as blocking issues or at least be given a high weight.

Required contracts have to be part of the test set. Possible negative side effects of existing contracts should be part of the test set too.

## *4.8 Test Automation*

AI testing needs substantial automation. The amount of test cases request it and tests need to be run repetitively with every new version. When the AI system is trained constantly, testing is necessary, as in the case of search engines where there are feedback loops from real data. But even when the AI system is not trained constantly and versions of the system are stable, a changing context demands constant training. Even when the system does not change, the world will.

Test automation consists of a test framework where the test cases will be run on the AI system and the output from the AI system will be processed. Below a basic setup of such a test framework is shown.

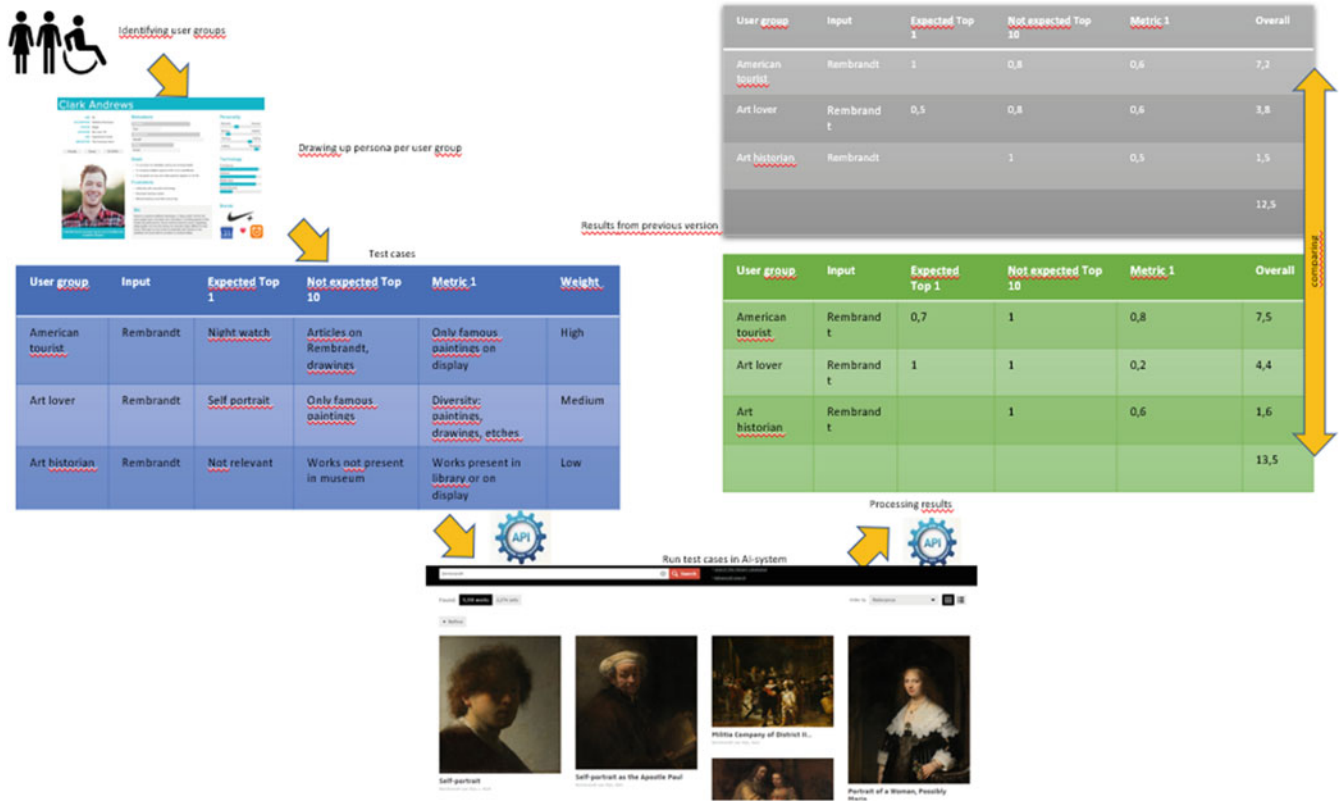
## *4.9 Overall Evaluation and Input for Optimising*

The product of testing is not just a list of bugs to be fixed. Bugs cannot be fixed directly without severe regression, as stated above. The AI-system has to be evaluated as a whole since with the many test cases and regression, no version will be perfect. Programmers want to know which version to take, if a new version is better than a previous one. Therefore the test results should be amalgamated into a total result: a quantitated score. For programmers to get guidance into what to tweak (training data, labelling, parametrisation) they need areas that need improvement. This is as close that we can get to bug fixing. We need metrics, weighing and contracts to achieve a meaningful overall score and clues for optimisation. Low scoring test cases should be analysed as to their causes: is it over-fitting, under-fitting or any of the other risk areas?

## *4.10 Example of AI Test Framework (Fig. 2)*

From left up to bottom and then right up:

1. Identifying user groups
2. Creating persona per user group




| User group | Input | Expected Top 1 | Not expected Top 10 | Metric 1 | Weight |
|---|---|---|---|---|---|
| American tourist | Rembrandt | Night watch | Articles on Rembrandt, drawings | Only famous paintings on display | High |
| Art lover | Rembrandt | Self portrait | Only famous paintings | Diversity: paintings, drawings, etches | Medium |
| Art historian | Rembrandt | Not relevant | Works not present in museum | Works present in library or on display | Low |

| User group | Input | Expected Top 1 | Not expected Top 10 | Metric 1 | Overall |
|---|---|---|---|---|---|
| American tourist | Rembrandt | 0,5 | 0,8 | 0,6 | 7,2 |
| Art lover | Rembrandt | 0,5 | 0,8 | 0,6 | 3,8 |
| Art historian | Rembrandt | | 1 | 0,5 | 1,5 |
| | | | | | 12,5 |

| User group | Input | Expected Top 1 | Not expected Top 10 | Metric 1 | Overall |
|---|---|---|---|---|---|
| American tourist | Rembrandt | 0,7 | 1 | 0,8 | 7,5 |
| Art lover | Rembrandt | 1 | 1 | 0,2 | 4,4 |
| Art historian | Rembrandt | | 1 | 0,6 | 1,6 |
| | | | | | 13,5 |

**Fig. 2** Example of AI test framework

3. Writing testcases: per user group input with expected top result, not expected, metrics and weight
4. Running test cases in AI system (search engine)
5. Processing results
6. Creating test results per test case with overall weighing
7. Comparing results with results from previous version

## 5 Conclusions

The world of AI is very dynamic: the algorithm does not equal the code but is a result of training data and labelling. Training data will be updated constantly as the world changes. The output of AI is not Boolean but calculated results on all labels which could all be relevant.

Despite low transparency and risks in Bias, AI is being used for decision making and is an important part of people's worlds. Testers must play a role in creating transparency, by identifying user groups and their specific expectations and needs and showing how the system reflects these. For this purpose an automated test framework is needed to compare the many versions of the AI system, monitor quality in production constantly and give guidance to optimisation.

An AI tester needs skills in data science but most of all moral and social sensitivity!