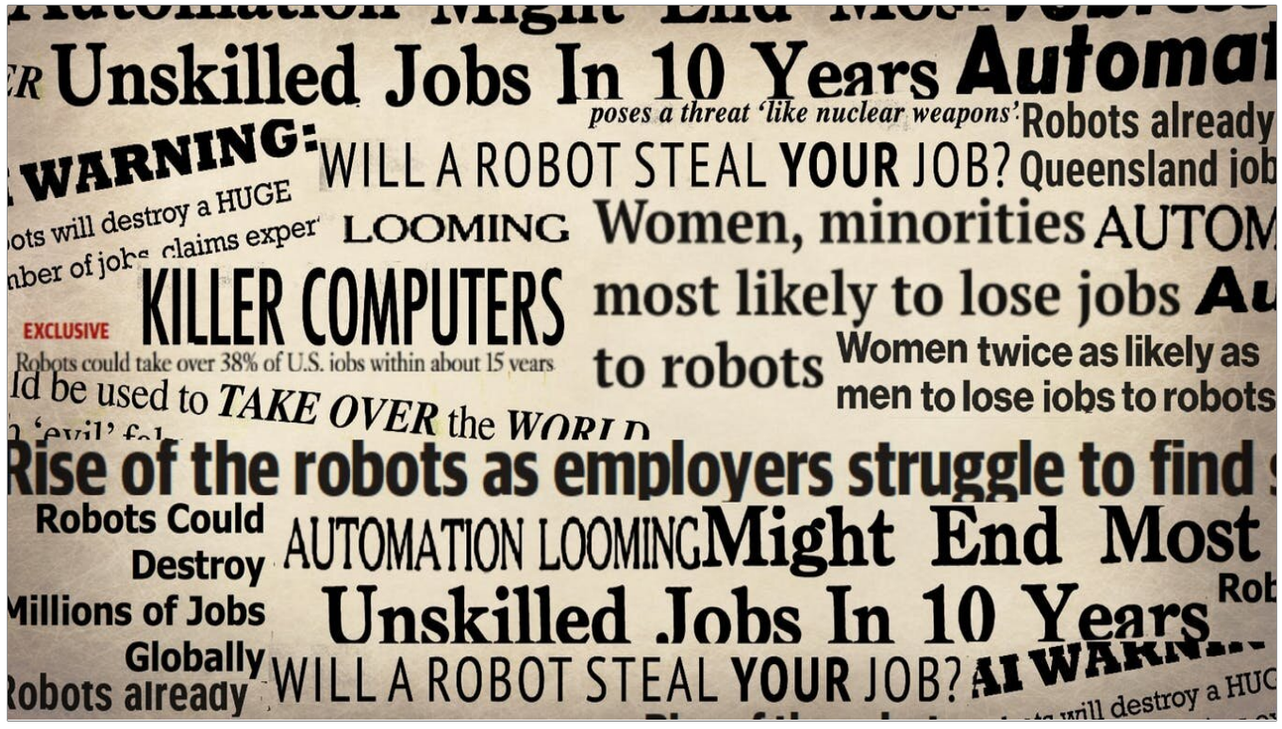
We have all heard the stories in the media about AI. It's either going to save the world and solve all of society's problems, or it's the greatest threat to humankind and it's going to take all our jobs off us.



If you believe the hype then AI will make us more productive, do all the boring jobs for us, do jobs we could never do and even be our best friend. But if you believe the doom-sayers then AI will put us all out of work, turn us in to couch potatoes and then kill us off in a Terminator style apocalypse!

Well, I've got some good news and some bad news. The bad news is It's not going to be the saviour of the world and solve all of society's problems. The good news is that it's not going to destroy the world and that we're still going to need Software Testers to test the AI systems. There is going to be plenty of work for us to do. In fact, one of the main drivers for that demand is legislation.

Legislation

EU AI Act

"The conformity assessment body will conduct an assessment of the AI system, including a review of its technical documentation, testing, and validation."

US

The White House Blueprint for an AI Bill of Rights

US Congress currently considering 2 bills to regulate the use of AI

Organization for Economic Co-operation and Development's (OECD's) 2019 Recommendation on Artificial Intelligence

G7 Hiroshima AI Process

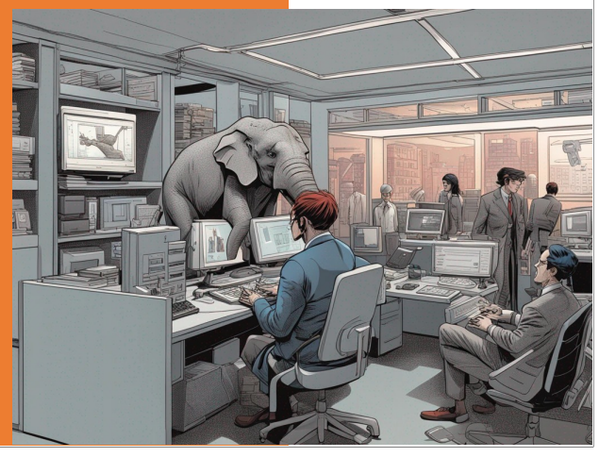
Global initiative to harmonise AI rules and ensure a coordinated approach to AI governance

EU AI Act, US Congress AI legislation, G7 Hiroshima, Australia/India agreement, Malaysia AI regulation Act, Ukraine AI regulation Act (not to mention using extensively in their war against Russia), Russia/China agreement

Add to this the problems that regularly hit news due to over-enthusiastically rushing out AI projects into the wild. And that makes everyone more nervous. And we expect a high level of safety and ethical behaviour from an AI. More so than we do from humans. If a self-driving car crashes just once it is a problem, but humans do it all the time. My favourite is Microsoft's Chatbot Tay which went from "humans are super cool" to full-on Nazi in 16 hours of exposure to Twitter users. Another is the CanadaAir's chatbot that gave out not existent flight advice and even personal advice to customers. Not to mention all the erroneous facial recognition court cases that are going on.

There is going to be plenty of work for us to do.

The Elephant in the AI Room



But there is an elephant in the room when it comes to conversations about AI. At the moment, almost everyone is taking about Generative AI and if AI and Testing are mentioned in the same breath, it's about AI in Test Automation tools.

As a Quality Engineering & Testing professional I need to consider firstly, How should we use AI in testing? and secondly How do we test the AI based systems we implement? As there are plenty of erudite people and tool vendors to tell you about how to use AI to test with, I am going to focus exclusively on the second, but I'm not going to go into the details of how AI works. Again, I will leave that to more erudite folks than myself.

And by the way, there are more forms of AI than Large Language Models. ChatGPT isn't the only fruit.

Poll of who has worked on AI, Data scientists, testers etc

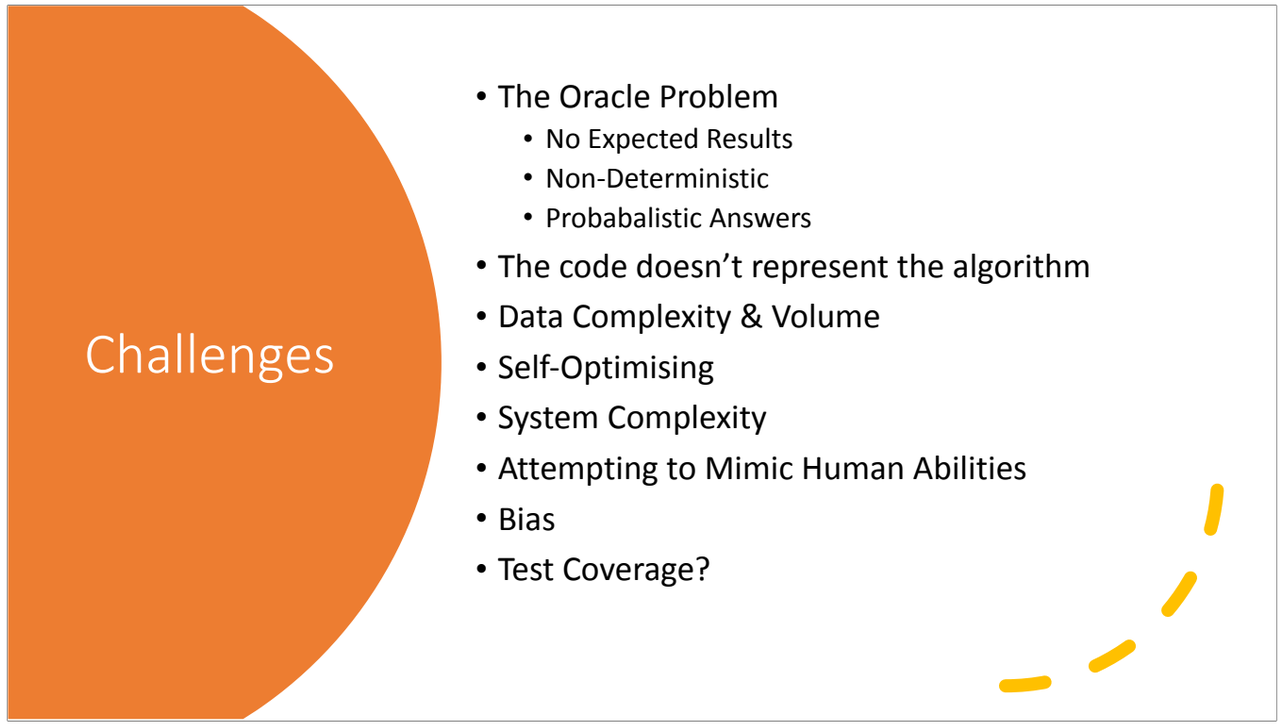
So, How do we actually test AI based systems?

How do we test AI?

I’ve got some more good and bad news.

Good news is that in terms of Test Levels it looks pretty similar to what we all know and love as the traditional test levels with just a couple of new ones.

The Bad News is that we are going to have to change the way we approach testing and learn some new skills. For me that’s good news ‘cos I am a permanently curious, learnaholic, but there are a whole bunch of challenges specific to AI systems, which means that we have to think in a slightly different way.



It is very difficult to say for the output of an AI with absolute certainty what is Right and what is Wrong. This is The Oracle problem. Not having an Oracle to provide a definitive answer. The system deals in probabilities and is non-deterministic, so we can’t have Expected Results to put a tick against, like we are used to.

The code for AI systems isn’t the algorithm. The data, both historic and current, **is**.

The whole point about AI is that it is really good at sifting through massive quantities of data and finding patterns. Often that data isn’t structured, making it complex.

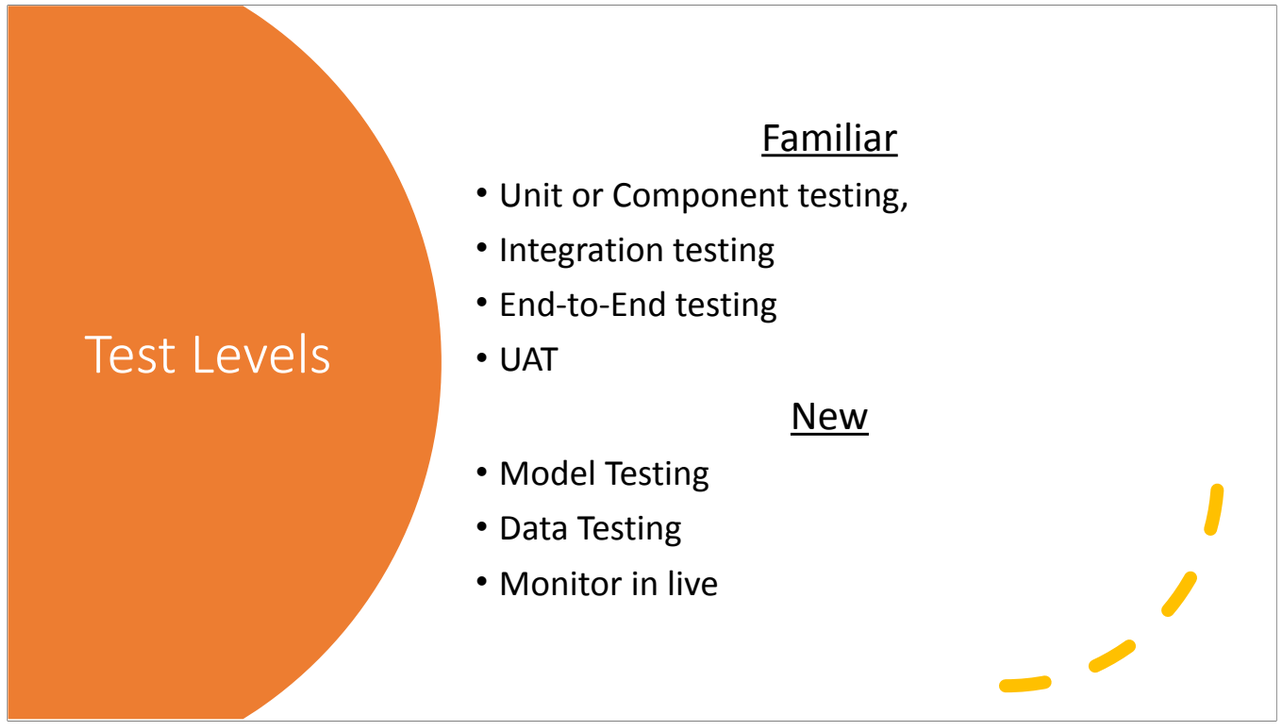
Because the data effects the behaviour, the system is self-optimising. It changes its behaviour as it goes along. Run the same data set through it twice and you won’t necessarily get exactly the same answer.

The situations we try to solve with them, by their nature tend to make them complex.And we are essentially trying to mimic human abilities that are very difficult to precisely define in themselves.

The problems of bias are well known in both humans and AI.

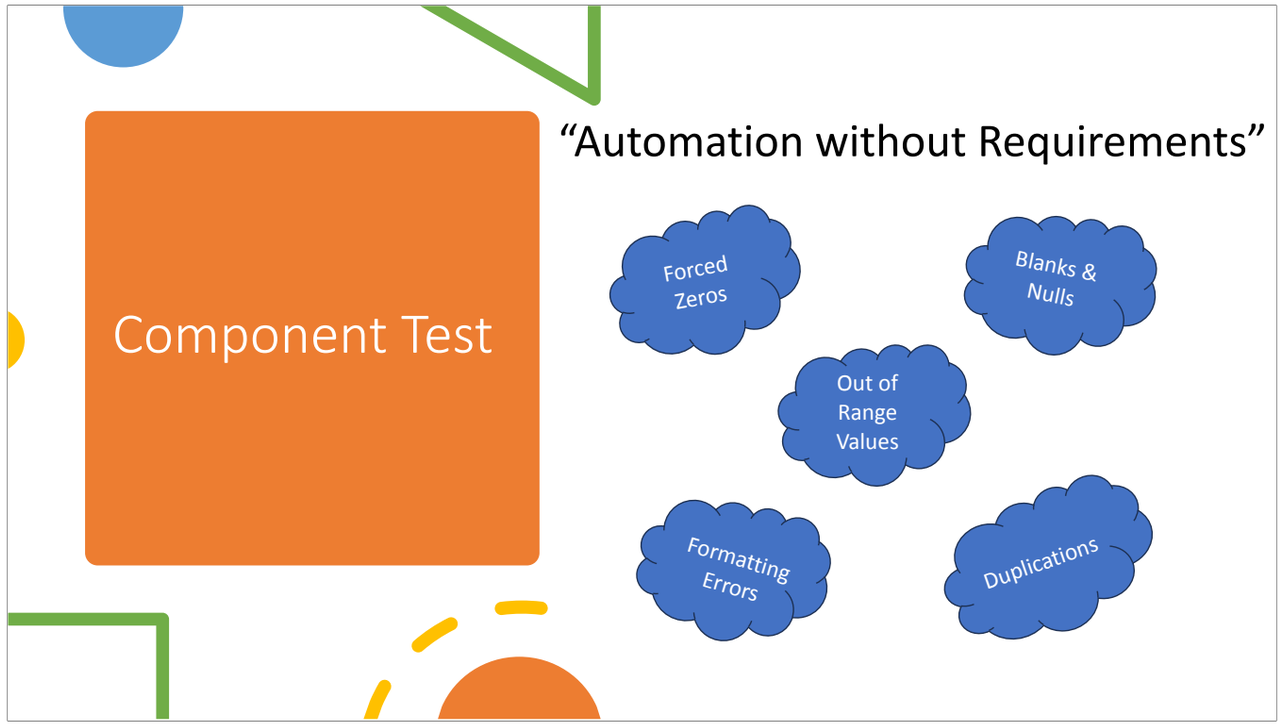
Lastly, how do we actually measure how much testing we have done and whether it is enough? The usual mapping of tests to requirements doesn’t apply in the same way and other traditional coverage metrics are equally inappropriate.

These challenges make our skills as testers even more valuable and as I said the test levels don’t look so different. So, what do they look like?



We have the Unit or Component testing, Integration testing, End-to-End testing and UAT. From that point of view, it looks like a standard simple SDLC. In addition, we also have Model Testing and Data Testing and at the tail-end we have Monitor in live, which many would argue is not Testing, but it is critical to sustained quality and aligns with the DevOps philosophy of shift left/shift right..

So far, so good. Let’s dig into these levels a bit further.



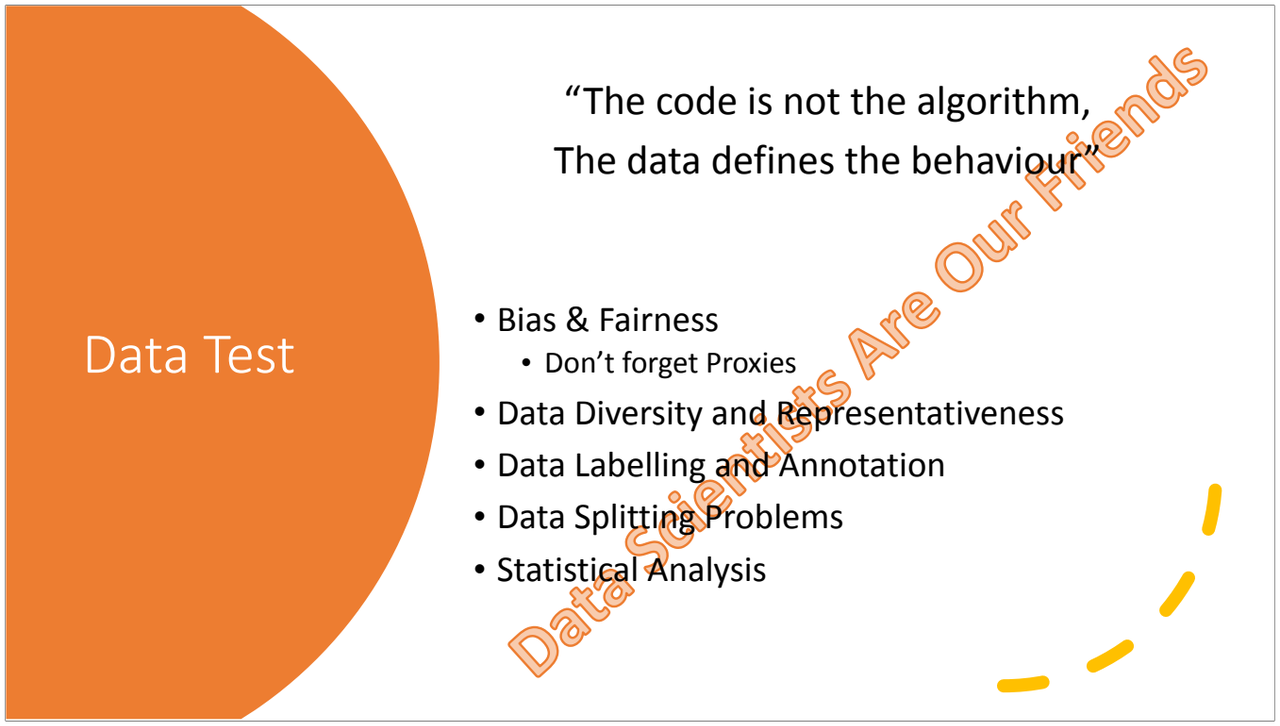
Component test is traditionally Black Box based on the requirement and possibly White Box based on the code itself, but neither of these really make sense for AI, as the code doesn’t represent the behaviour of the model and AI is often referred to as “Automation without Requirements”. We have to take a slightly different view.

The actual AI functionality is usually a library element but they have to be fed data and it is that data pipeline where our focus needs to lie. Does it handle forced zeros, blanks, out of range values, formatting issues and duplications? If it transforms or normalises the data, does it do it correctly?

All areas of testing that I’m sure we’re all familiar with and you can apply the test techniques with which you are familiar.

All of our well known heuristics apply here.

This overlaps with less familiar test level, Data Testing.



The code is not the algorithm, The data defines the behaviour, so we need to be very aware of the impact that the data has. This is an area where we need to learn some more skills and gain an understanding of Data Science. That said we don't want to try to take it away from the Data Scientists. Quite the contrary, we should make friends with them and cozy up to them, so we can apply our critical thinking skills to what they are doing. We can help them test their data better.

But what should we and the Data Scientists be looking for?

Bias and Fairness Assessment: Evaluate the data for biases and potential fairness issues. Check for underrepresented groups or data that may lead to biased AI predictions. But be aware that the real-world data that the system is exposed to in Live may also be subject to biases. Also be aware that there are Proxies for bias, for instance Occupation. Statistically some occupations tend to be more one gender than another e.g. Nursery Nurse. Likewise address areas can be proxies for race, class or income bracket and in some address areas even age. All of which can be considered discriminatory if used for decisions. Techniques like bias audits, fairness metrics, and demographic parity analysis, can help expose this. Tools like Fairlearn and AI Fairness 360 help us to detect and mitigate this.

Data Diversity and Representativeness: Ensure that the data covers a wide range of scenarios, including edge cases and potential outliers. This diversity helps the AI system generalize better to real-world situations. A simple statistical analysis of the distribution of the data should indicate some issues, especially when compared to real-life data.

Data Labelling and Annotation: Labelling and Annotation of the training data is an important part of training an AI. Check Annotation data with correct classifications or values, ensuring consistency in labelling. Too broad or too narrow a granularity of the definition of a classification can cause inaccuracies.

Data Splitting Problems: It is usual with ML systems to provide it with a large amount of training data with a smaller amount of data held back for testing purposes. It is important to check the diversity & representativeness of the testing data. In traditional testing we wouldn't just check the Happy Path

Statistical analysis of the data can used to address these issues, which is the Data Scientists' realm, but it takes a Tester's curiosity and critical thinking to spot the gaps & anomalies, ask questions and find problems.

Data Scientists are our friends!

Integration Test

Familiar but …

Probabilistic!

Chained Probabilities multiply

35% cat, 55% dog, 15% blueberry muffin

The integration level is familiar, but it does have a slight complexity.

AI systems are essentially probabilistic, meaning that they initially output a confidence factor. A probability of the answer being in specific category. E.g. 35% cat, 55% dog and 15% blueberry muffin. If you chain together multiple AI systems you are multiplying the error factors, so making harder to predict correctly and harder to determine where it might go wrong. Not an insurmountable complexity, but again it's where a Tester's curiosity and questioning comes in useful. Systems thinking is also essential.

An example of this chaining is facial recognition. First it has to decide whether it's a face or not, before working out who's face it is. As you can see identifying faces to start with is not always straight forward.

Model Testing

- Statistical in nature
- Large data volumes
- Complex to interpret
- Data Scientists are our friends

Still benefits from a Tester's curiosity, critical thinking, system thinking, and Question Asking!

Model Testing is a new level but shouldn't be too unfamiliar to anyone used to thinking about testing in terms of modelling, but it's not necessarily easy. In this phase, the primary focus is on evaluating the AI model's performance, behaviour, and predictive capabilities, in isolation from the surrounding IT. The way to do this is to do statistical analysis on the processing of large qualities of data. Typically with Machine Learning systems you would have a huge data set that would be split in to something around 80% training data and 20% test data.

There are tools to help with validating the model, such as on the Huggingface website. TensorFlow also has tooling available and there are other open libraries such as PyTorch and SciPy that can provide data analysis functionality.

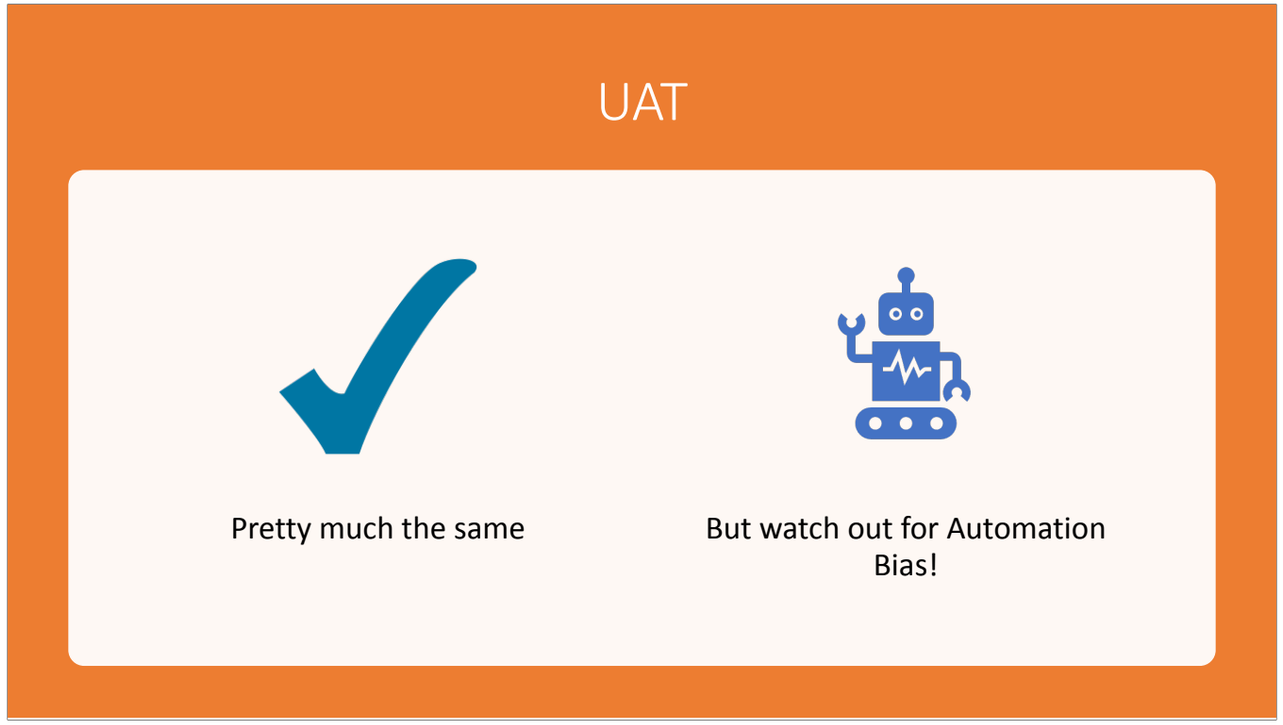
The interpretation of that data and the resulting outputs can be complex, so I would recommend getting friendly with your Data Scientists. They are the experts in statistical analysis, so use them.

That said, the fundamental skills of a tester in asking those awkward questions and revealing the gaps are just as essential here. Data scientist are just as likely to make mistakes and suffer bias as anyone else.

Again this is a very familiar test level, but

Beware of that complexity of probabilities. Just because each of the systems involved in the overall solution have been tested it doesn’t mean that the final result is good. Watch out for multiplying error factors and always be critical of data pipelines.

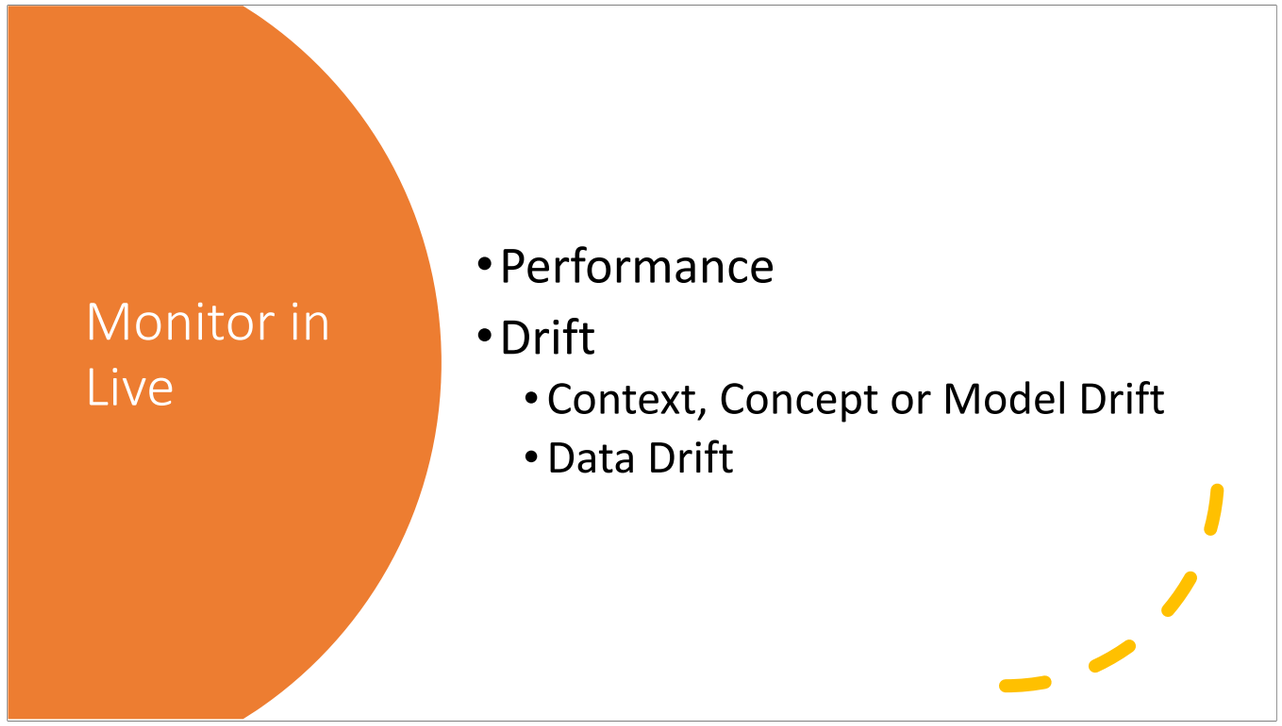
If the system is trying to behave like an expert in a certain field, then make sure you involve an expert in that field to help decide what is a “Right” answer.



UAT is very much the User Acceptance Testing that we all know, but …

Watch out for Automation Bias. Automation bias is the propensity for humans to favour suggestions from automated decision-making systems and to ignore contradictory information made without automation, even if that information is correct. The “Computer says no” syndrome.

One technique to pre-empt what the users will do during UAT is Affordance Modelling, and should be used during the earlier levels of testing.



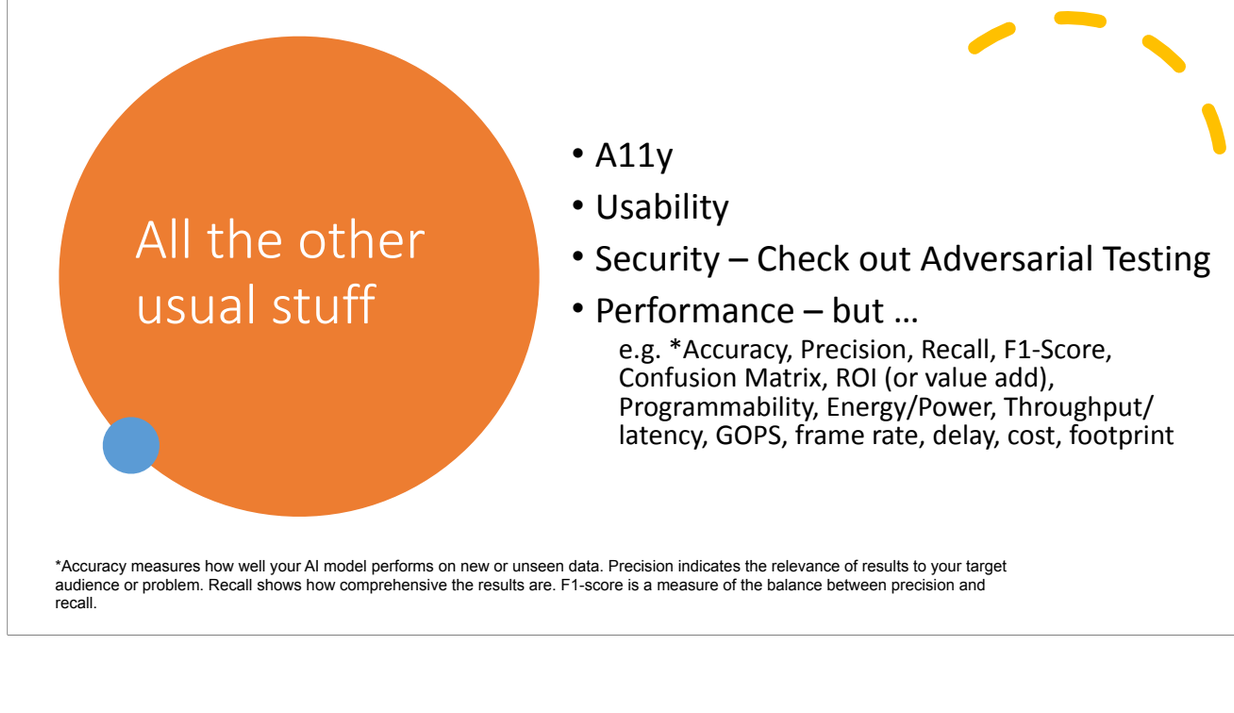
Although Monitoring in Live isn't what a lot of people would consider to be a test level, it is critical to sustaining the quality of the system performance. Bear in mind that the performance of an AI system isn't necessarily how fast it does stuff. More importantly it is how accurate it is. If the accuracy of a cancer detection system decreases that could have catastrophic consequences for the individuals involved.

When a system's performance metrics decrease it is down to Drift. Normally Context, Concept or Model Drift and Data Drift. Concept drift is when the task that the model was designed to perform changes over time. For example, imagine that a machine learning model was trained to detect spam emails based on the content of the email. The spammers over time change their approaches and so the model may become inaccurate.

Data drift is when the distribution of the input data changes over time. An example was a large high street bank that was using AI to advise decisions on loan applications. When Covid struck the income characteristics changed, especially due to the furlough scheme. So the system had to be retrained.

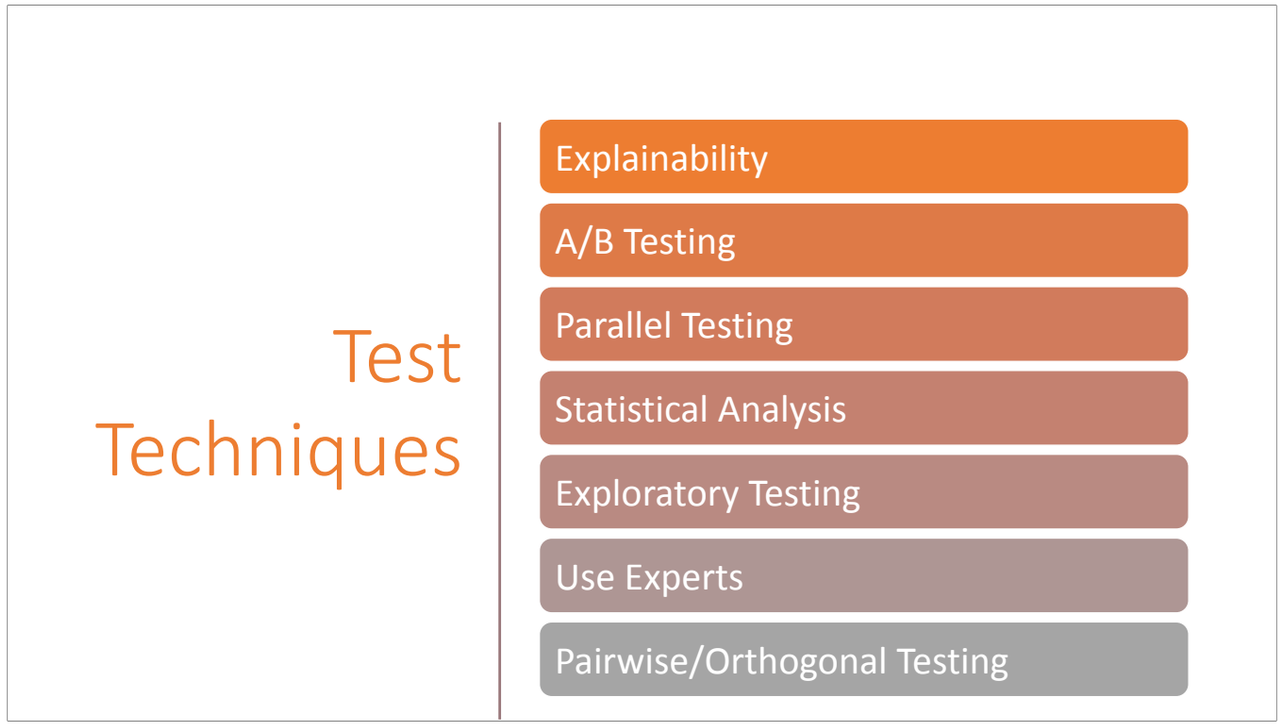
There are tools and techniques for detecting Drift, such as IBM's DetAIL, so it again comes down to the Data Scientist, but we still need to be asking those awkward questions to avoid any complacency. Just 'cos it looks like it's working now, doesn't mean that it will continue to do so.

Observability in the non-AI parts of the system and Explainability are crucial factors to be designed in to the system for working out what happened if & when it does go wrong.



This is of course, all the other usual testing to consider, such as Accessibility, Usability and Security. They are pretty much as you would expect, but Security is an interesting one. Since the data defines the behaviour, a malevolent actor could change the behaviour by manipulating the data. A case in point is Microsoft's Chatbot, Tay. There is an approach called Adversarial testing where you try to do this deliberately to see how the system copes.

And of course, Performance, but remember performance metrics aren't restricted to speed or volume.



In addition to your usual testing techniques here are some that are specifically applicable to AI systems.

The first up is **Explainability**. Decision systems, AI and otherwise, have to be able to explain the parameters that influence the outcome, such as to prove the absence of bias and give the opportunity to change the inputs to change the outcome. An example would be that loan decision system where a loan applicant can legally request an explanation of why they were refused a loan. There are Explainability tools out there that can be embedded in systems and the AI legislation I mentioned earlier requires any AI system to be explainable.

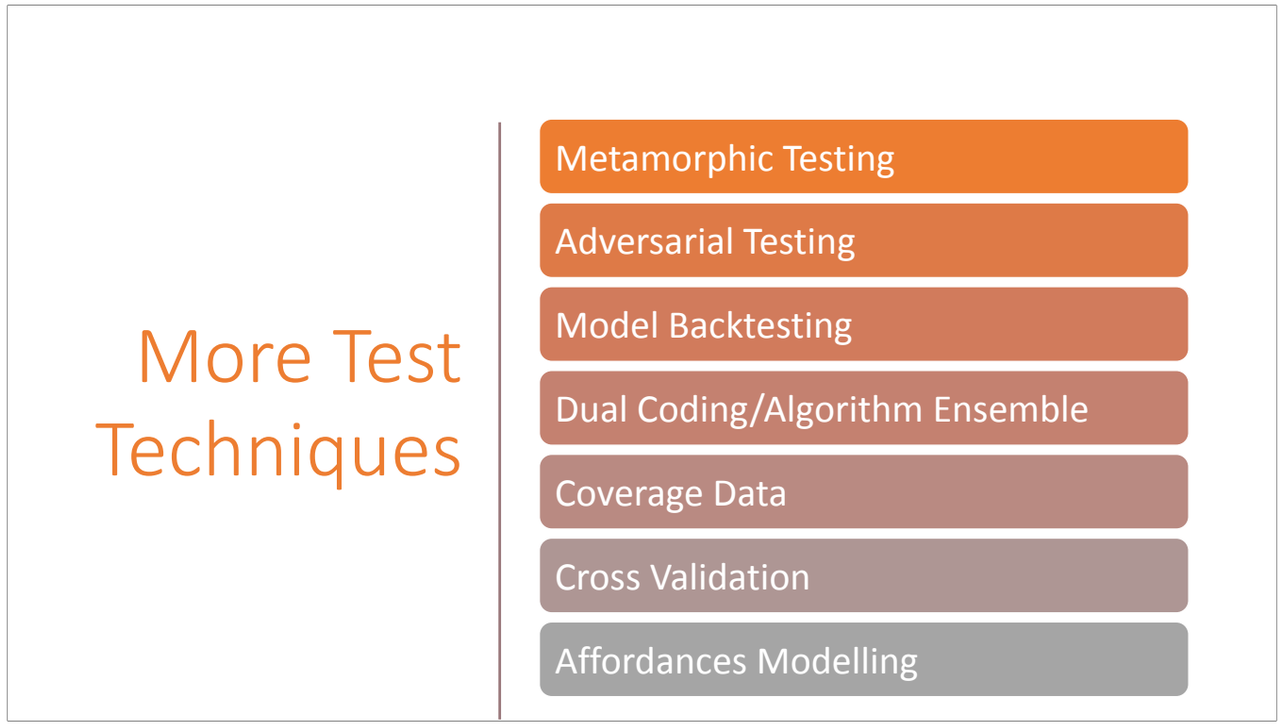
The Oracle Problem means that it's difficult to definitively say whether a new solution is working correctly or whether one model is better than another. One way around this is to run the new system next to the old one, which is Parallel Testing. Or run the alternative versions next to each other and compare, which is A/B testing.

As I've mentioned, **Statistical Analysis** is important to understand and to use. You can even do some of it yourselves on a smaller scale or with tools such as PowerBI, dedicated Python code libraries or even Excel.

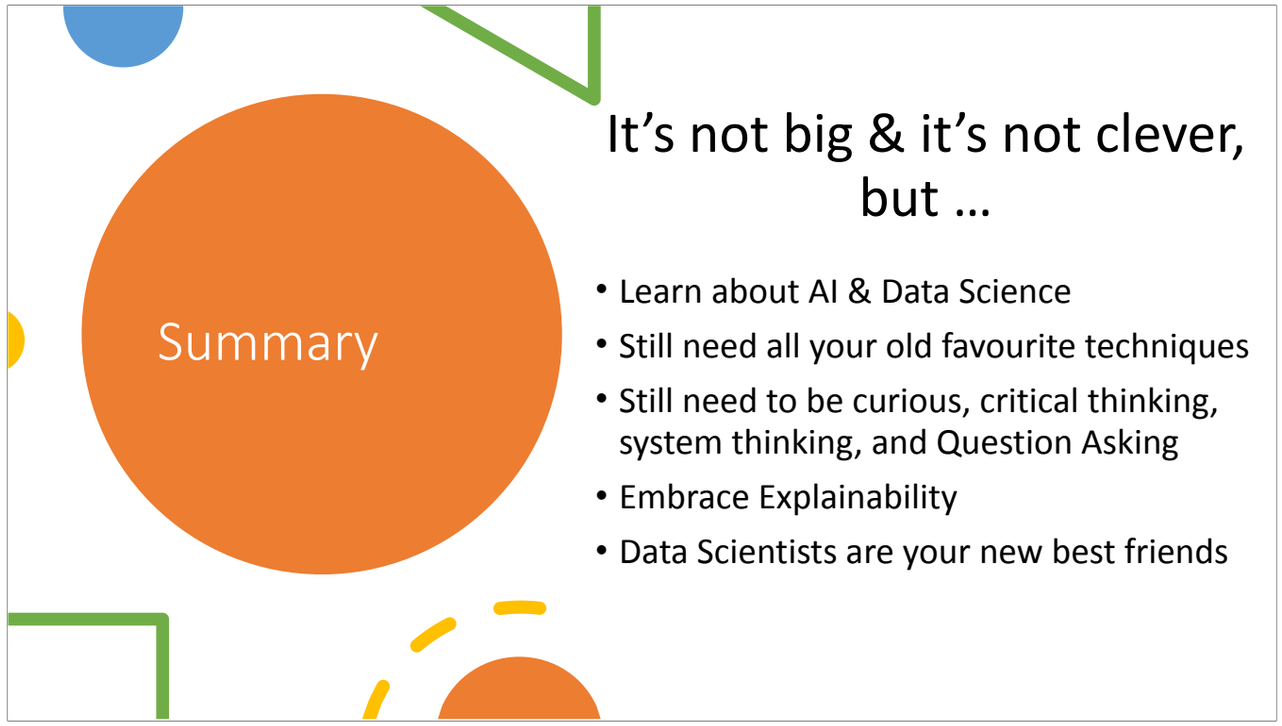
I won't explain **Exploratory Testing** as I'm sure you all know it well, but AI is an ideal arena for Exploration. This is especially true for LLMs - Large Language Models- and Generative AI in general. Fixed, documented Tests don't really make much sense with Generative AI, but taking a theme in the form of an exploratory charter and then exploring it in every dimension you can think of, does. This is very much a case of needing a Human In the Loop to determine the quality of the responses. GenAI's penchant for getting things wrong and hallucinating is quite widely discussed, so the veracity of the information provided also needs to be checked. As for Image generation AI, only a human can judge the quality of the output.

Another way around the Oracle Problem is to use a genuine human oracle. **An expert**.

Lastly **Pairwise, Orthogonal or All-Pairs** testing is a very efficient way of reducing the number of test cases required to test a massive number of permutations. It is a mathematical approach, so there are free tools to help do it as, such as: Hexawise, All Pairs, Testcover, pairwise, Allpairs.
In traditional combinatorial testing it can typically reduce quarter of a million combinations down to less than twenty.



Metamorphic testing: This is a technique where perturbations are introduced into the input data and the change in the output data noted. It's useful because it sidesteps the Oracle problem to a large degree, as you're only interested in the changes in the output, not necessarily the values themselves. It's also possible to automate metamorphic testing to certain amount.
Adversarial Testing: This is providing inputs that try to break the system or trick it in to providing incorrect answers or answers that it's business rules were not suppose to allow it to reveal. Often used in Security testing.
Model Backtesting: A predictive model tested on historical data is known as backtesting. This method is widely used in the financial sector to estimate the performance of previous models, particularly in trading, investment, fraud detection, and credit risk evaluations.
Dual Coding/Algorith Ensemble: Similar to A/B and Parallel testing. Multiple models utilising various algorithms are given the same input data set, and predictions from each one are compared. The model that gives the most expected outcomes is ultimately chosen as the default. It can be used as a development strategy. A large number of models are used with different hyperparameters. It can then be linked with an optimising algorithm, such as a genetic one to select the most optimal result for the input data and tune the hyperparameters.
Coverage Data: Test data sets designed such that it results in the activation of each of the neural network's neurons/nodes. Don't be fooled in to thinking that this gives you full test coverage though.
Cross Validation: Uses different portions of the data to train and test a predictive model iteratively. The overall goal is to try to estimate how the model will perform in practice on unseen data. One of the most popular cross-validation techniques is k-fold cross validation. where the dataset is shuffled and split in to K number of groups. The model is then trained with all the groups except one, which is used to test. The results are kept, the model cleared down and the process repeated until all groups have been used to test. The model should perform similarly each time.
Affordances Modelling: Affordances are essentially the properties of an object that dictates how it will be used. For instance the handle on a mug. When this is moved into the virtual world we are basically looking at how the system will be used by the end user. This provides Use Cases for us to track back to test cases.



So, to summarise, It’s not big and it’s not clever, but it does require a different way of thinking about systems.

So at least learn the basics of AI and Data Science.

You will still need to use all the old favourite test techniques but apply your new understanding to them. The fundamental testing skills of curiosity, critical thinking, systems thing and above all Question Asking are now as ever, crucial to effective testing. Use your newfound knowledge to ask the awkward questions that testers are good at.

Explaininability tooling will be invaluable if they’ve included it early enough.

And remember, Data Scientists are your new best friends.

There are other more academic approaches to Testing AI out there and it is worthwhile researching and staying up to date, but this should give you a solid, on-the-ground approach to refine for your own situations. And also remember not all AI systems are Machine Learning or Large Language Models. There are plenty of other models out there as well.

Useful Resources

Websites

- HuggingFace
- Fairlearn
- AI Fairness 360
- What-If-Tool
- DetAIL

Books & Papers

- Artificial Intelligence and Software Testing (BCS)
- AI Engineering : 11 Foundational Practices (Carnegie Mellon University)
- Testing in the digital age (Sogeti)

Thank You

Questions?

Email Bryan.Jones.QA@gmail.com
Twitter @Bryan_QA_Jones
LinkedIn bryan-jones-mbcs-96953
Podcast Quality Blether