wipro holmes

wipro

# Testing of AI/ML based systems

The related challenges and effective techniques to overcome them

Artificial Intelligence (AI) is transforming the technology landscape of the digital age. The world is moving towards an increased adoption of AI powered smart applications/systems which will increase exponentially over the next few years. While we see the advancements, the key challenge would be testing of Artificial Intelligence (AI)-Machine Learning (ML) based systems. This is due to lack of any structured or standard literature on the methodology or approach to be adopted while testing. The actions and responses of these systems differ over time, based on input data and thus are less predictable than traditional IT-systems. Also, traditional testing techniques are based on fixed inputs producing certain fixed outputs. Test cases are no longer expected to pass with an accuracy of 100% but are replaced by a set of metrics. These metrics are expected to meet certain levels of acceptability to ensure viable deployment. The challenge here is that with every new version of the ML model, evaluation and acceptance tests need to be carried out to ensure an acceptable level of performance.

Given the changing environment, tests such as unit testing of individual components and end-to-end tests of running systems, though valuable, are not enough to provide evidence that a system is working as intended. Comprehensive live monitoring of system behavior in real time combined with automated response is critical for long-term system reliability. The speed at which these systems need to be tested will be dramatically faster than the current world of Agile/DevOps based continuous testing.

For AI-powered smart applications, significant focus would be on accuracy-based testing of trained models, the data sets used to train and test these models and the testing techniques deployed in the various testing stages (shown in Figure 1).

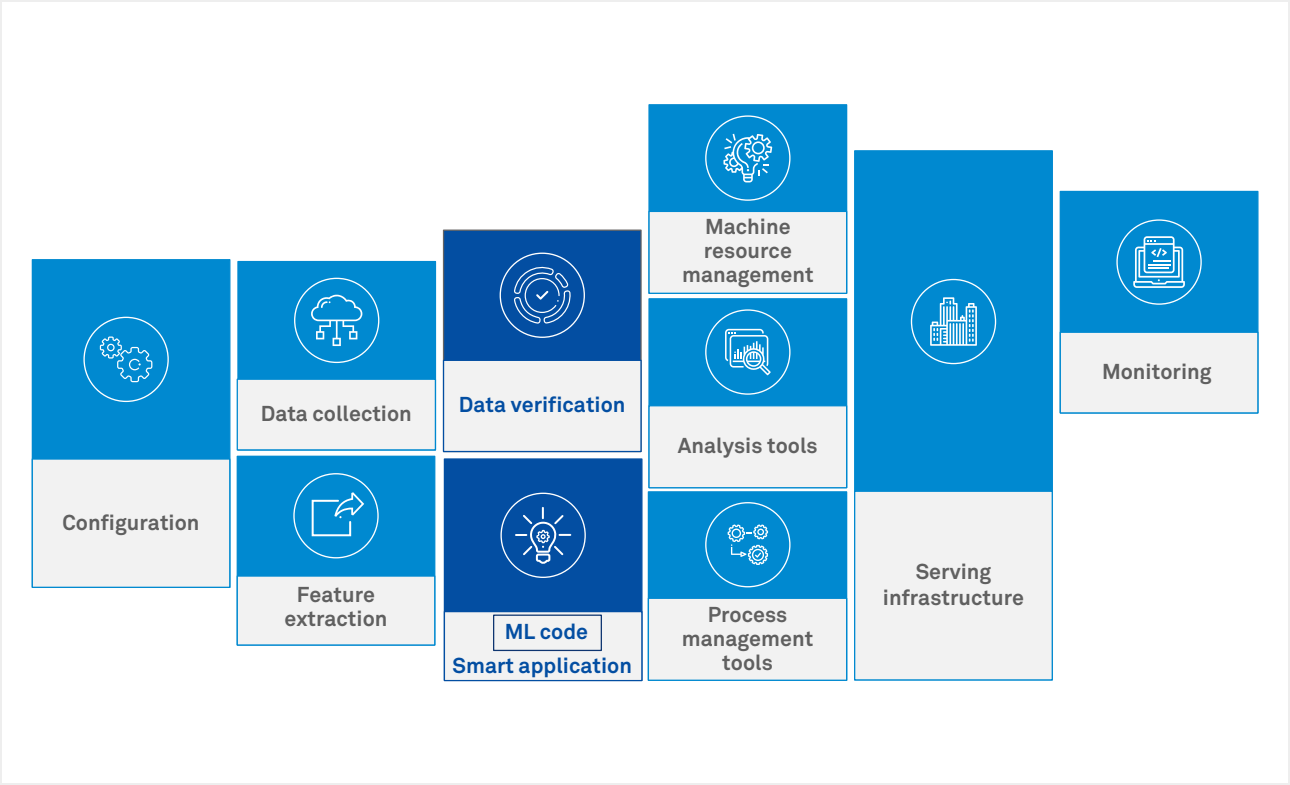


Figure 1: Lifecycle of AI-based smart apps development – build to deploy

Only a small fraction of real-world ML applications/systems is composed of ML code as represented in Figure 1. The surrounding infrastructure required is vast and complex.

We will focus only on two key aspects here which are data verification and the key elements of testing to be deployed for ML code based smart application.

## Data verification: Data set paradigms and eliminating data biases

Data is the new code for AI-based solutions. These solutions need to be tested for every change in input data, to have a smoothly functioning system. This is analogous to the traditional testing approach wherein any changes in the code triggers testing of the revised code. Key factors to be considered when testing AI-based solutions include:

**Developing curated training data sets:** Training data sets that are curated in a semi-automated way include input data and the expected output. Conducting static analysis of data dependencies is important to enable annotation of data sources and features - an important feature for migration and deletion.

**Developing test data sets:** Test data sets are intelligently built to test all possible permutations and combinations to deduce the efficiency of trained models. The model is further refined during training as the number of iterations and data richness increase.

**Developing system validation test suites:** System validation test suites are based on algorithms and test data sets. For example, for a system built to predict patient outcomes based on pathological or diagnostics reports, test scenarios must be built around risk profiling of patients for the concerned disease, patient demography, patient treatment, and other similar test scenarios.

**Reporting of test results:** Must be done in statistical terms as validation of ML-based algorithms produces range based accuracy (confidence scores) rather than expected outcomes. Testers must determine and define the confidence thresholds within a certain range for each outcome.

Building unbiased systems has become essential for modern businesses. Supervised learning techniques, that cover more than 70% of AI use-cases today, have labelled data that is fraught with human judgement and biases. This makes testing the 'bias-free' quotient of the input training data sets, a 'double-edged sword'. If we don't factor the human experience in labelled data, we miss out on experiential knowledge. And if we do, then data biases can creep in.

These can be reduced through Apriori testing of the input labelled data for any hidden patterns, spurious correlations, heteroscedasticity, etc. Let's look at some of the key biases that need to be considered during AI/ML testing

**Data bias:** Often, the data that we use to train the model is extremely skewed. The most common example is sentiment analysis – most of the data sets do not have equal (or enough) number of data points for different types of sentiments. Hence, the resulting model is skewed and "biased" toward the sentiments that have larger data sets.

**Prediction bias:** In a system that is working as intended, the distribution of predicted labels should equal that of the observed labels. While this is not a comprehensive test, it is a surprisingly useful diagnostic step. Changes in metrics such as this are often indicative of an issue that requires attention. For example, this method can help detect cases in which the system's behavior changes suddenly. In such cases, training distributions drawn from historical data are no longer reflective of the current reality.

**Relational bias:** Users are typically limited and biased on how a data pattern or problem set needs to be solved, based on their view of the relational mapping of which solution would have worked for a specific kind of problem set. However, this can skew the solution towards what a user is comfortable with, avoiding complex or less familiar solutions.

While there is a need to resolve data biases, as explained above, we should also look at the problem of under-fitting or over-fitting of the model through training data, which happens much too often, resulting in poor performance of the model. The ability to measure the extent of over-fitting is crucial to ensuring the model has generalized the solution effectively, and that the trained model can be deployed to production.

## Key elements for testing of AI/ML code-based smart applications

The top four elements that we have considered for testing of AI systems and applications are illustrated in Figure 2.

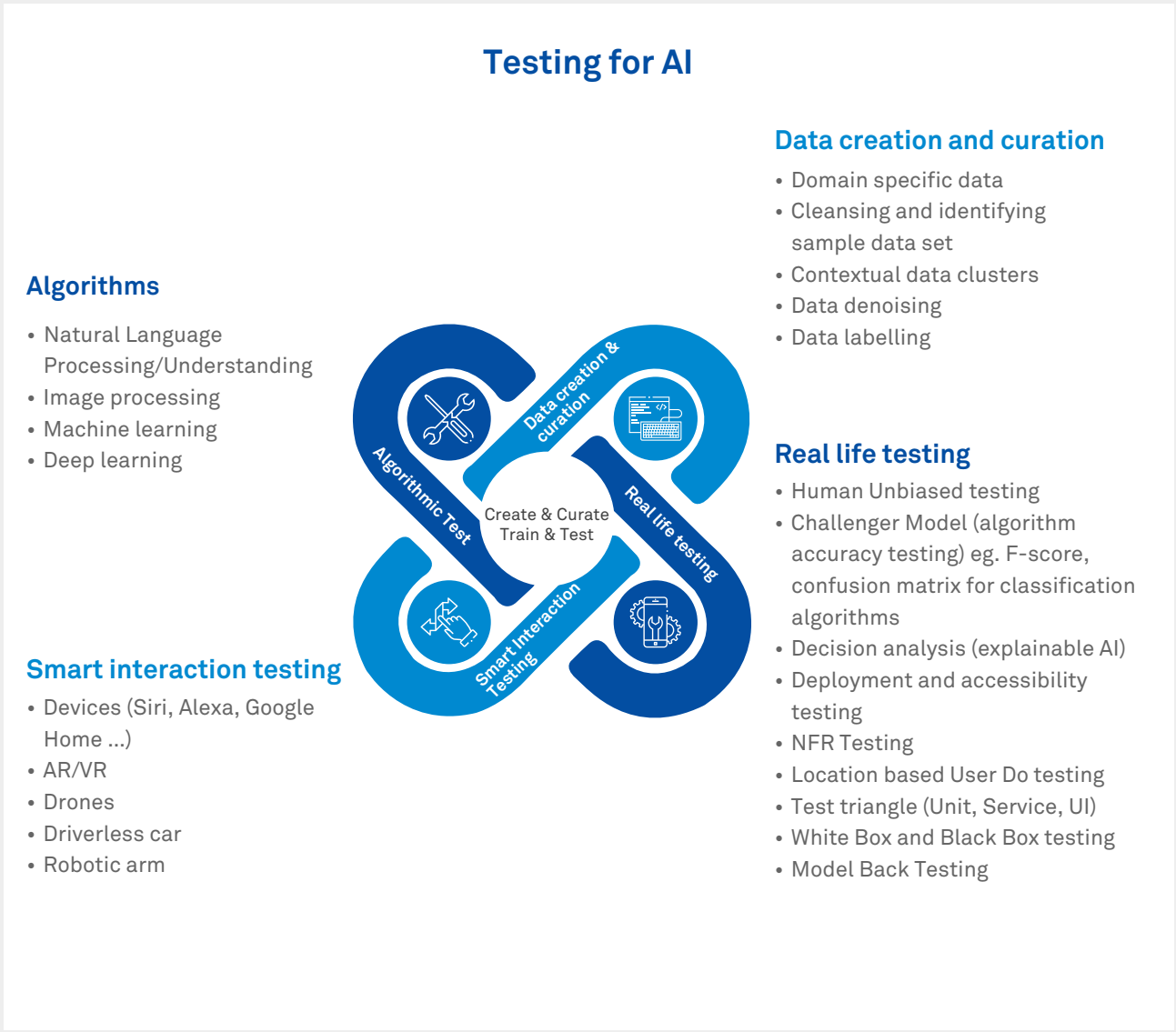


Figure 2: Top four elements for AI testing

While we foresee that Explainable AI (XAI) and Auto-ML techniques will greatly improve testing effectiveness going forward, we will focus on only some of the techniques that will need to be used in real life testing from a model and data set perspective.

### Black Box and White Box testing

Testing ML models involves Black Box and White Box testing much like traditional test methods. Obtaining training data sets that are sufficiently large and comprehensive enough to meet 'ML testing' needs is a major challenge.

During the model development phase, data scientists test the model performance by comparing the model outputs (predicted values) with the actual values. Some of the techniques used to perform Black Box testing on ML models are:

- Model performance testing: Involves testing with test data/new data sets and comparing the model performance in terms of parameters such as precision recall, F-score, confusion matrix (False and True positives, False and True negatives) to that of pre-determined accuracy with which the model is already built and moved into production.
- Metamorphic testing: This attempts to reduce the test oracle problem. Test oracle is the mechanism by which a tester can determine whether a system reacts correctly. It occurs when it is difficult to determine the expected outcomes of selected test cases or to determine whether the actual output is in accordance with the expected outcomes.
- Dual coding/Algorithm ensemble: Multiple models using different algorithms are built and predictions from each of them are compared, given the same input data set. For building a typical model to address classification problems, multiple algorithms like Random Forest or a neural network like LSTM could be used - but the model that produces the most expected outcomes is finally chosen as the default model.
- Coverage Guided Fuzzing: Data fed into the ML models is designed to test all the feature activations. For instance, for a model built with neural networks, testers need test data sets that could result in the activation of each of the neurons/nodes in the neural network.

### Model back testing

Back testing is a predictive model based on historical data. This technique is popular in estimating the performance of past models used in financial sectors, especially for trading, investment, fraud detection or credit risk assessments.

### NFR (Non-Functional Requirements) testing

In addition to performance and security testing, factors such as a representative sample view of things along with the deployment approach also need to be considered while testing ML Models. Questions such as: How do we replace an existing model in production? What is our view of A/B Testing or Challenger Models? – must also be answered.

## Driving successful AI testing outcomes

In conclusion, the number of considerations that must be accounted for, while deploying an AI/ML model into production, are fundamentally different from the traditional software testing frameworks. The periodicity of constantly testing for accuracy of the AI model also alters the 'test once and deploy forever approach' used earlier. As organizations increasingly adopt AI for building their systems and applications, the approach and practices to test such systems will evolve and get refined over the next few years, eventually achieving the maturity and standardization of traditional testing methods.

## About the authors

**Sanjay Nambiar,** Vice President and Head of AI & Automation Ecosystem, Wipro

Sanjay is responsible for building AI partnerships, formulating go to market strategies, and driving service delivery for enterprise clients. With over two decades of experience in client relationship management and business development roles, Sanjay specializes in shaping new offerings and driving business outcomes across multiple verticals, including retail, consumer goods, financial services and telecom.

**Prashanth Davey,** Practice Head - Quality Engineering & Testing Practice, Wipro

Prashanth is the product owner for Intelliassure - a cognitive and intelligent automation platform. His current responsibilities include product development, roadmap creation, customer rollout support, and driving synergies with Wipro's AI practice to incorporate cognitive automation capabilities for other peer platforms. Prashanth has over two decades of experience in testing across verticals such as telecom, mobile, and embedded domains as part of various functions, including delivery, pre-sales and IP creation.

**Wipro Limited**
Doddakannelli, Sarjapur Road,
Bangalore-560 035, India

Tel: +91 (80) 2844 0011
Fax: +91 (80) 2844 0256
**wipro.com**

Wipro Limited (NYSE: WIT, BSE: 507685, NSE: WIPRO) is a leading global information technology, consulting and business process services company. We harness the power of cognitive computing, hyper-automation, robotics, cloud, analytics and emerging technologies to help our clients adapt to the digital world and make them successful. A company recognized globally for its comprehensive portfolio of services, strong commitment to sustainability and good corporate citizenship, we have over 175,000 dedicated employees serving clients across six continents. Together, we discover ideas and connect the dots to build a better and a bold new future.

For more information, please write to us at **info@wipro.com**

IND/BRD/MAR 2019 - FEB 2020