

# AI & TESTING: TIPS FROM THE TRENCHES



**Adam Carmi**

*Co-Founder and CTO  
Appliteools*



# AGENDA

- Major approaches to implement AI
- AI @ Applitools
- AI testing tips
- How can you leverage AI for testing?
- Q&A



**“AI”, YOU KEEP  
USING THAT WORD**

**I DON'T THINK IT MEANS  
WHAT YOU THINK IT MEANS**

# WHAT IS AI?



[AI] is **intelligence demonstrated by machines**, in contrast to the natural intelligence displayed by humans and other animals.

WIKIPEDIA

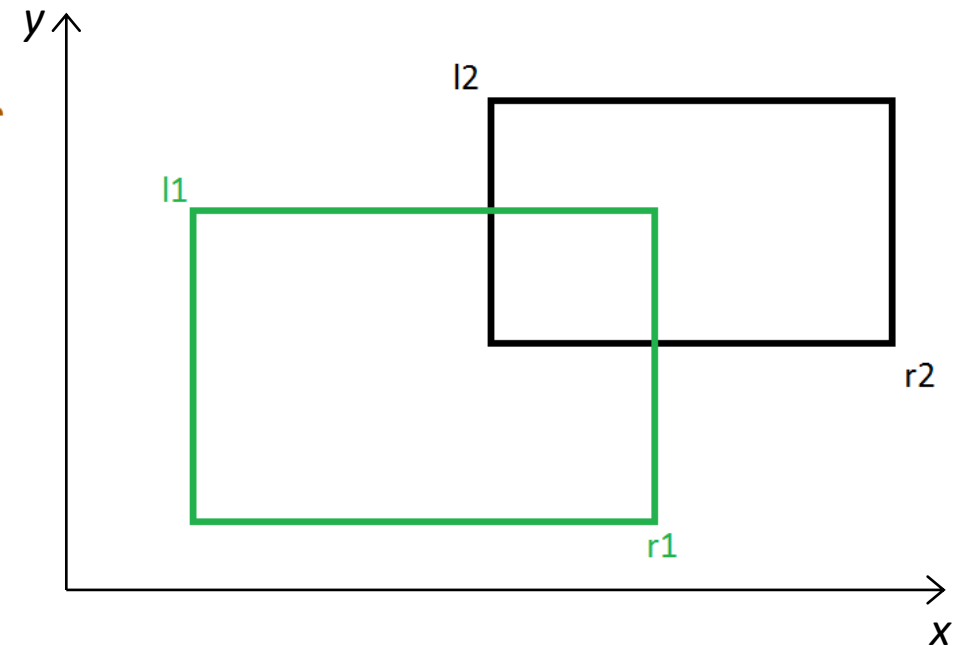
# IS THIS AI?

Are the two rectangles overlapping?

```
// Not if one rectangle is to the right of the other
if (l1.x > r2.x || l2.x > r1.x)
    return false;

// Not if one rectangle is below the other
if (l1.y < r2.y || l2.y < r1.y)
    return false;

return true
```



# HAND-CODED ALGORITHMS

- The best approach providing that
  - You are smart enough to figure out the solution
  - The implementation code is readable and maintainable
- Very little data is needed (to understand the problem)
- Easy to debug
- **Can be white-box / gray-box tested**

# HAND-CODED HEURISTICS?

```
if (feature1 > 0.8)
    return true
else
    return false
```

# HAND-CODED HEURISTICS?

```
if (feature1 > 0.9)
    return true
else
    return false
```

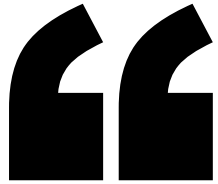


# HAND-CODED HEURISTICS?

```
if (feature1 > 0.8)
  if (feature2 > 0.9 && feature3 < 0.4)
    return false
  else if (feature1 > 0.9 && feature2 < 0.3 &&
    feature3 > 0.8)
    return false
  return true
else
  return false
```



# MACHINE LEARNING



A field of computer science that uses **statistical techniques to give computer systems the ability to "learn"** (e.g., progressively improve performance on a specific task) **with data, without being explicitly programmed.**

WIKIPEDIA

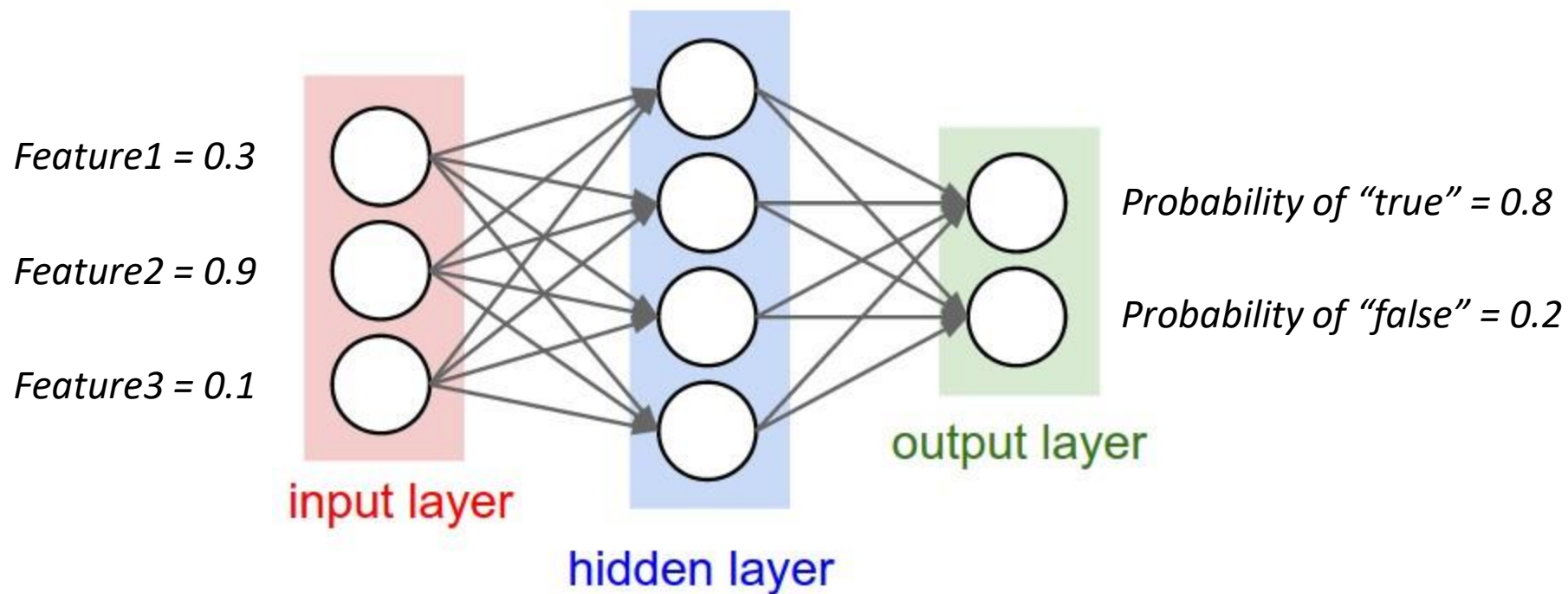
# NO NEED TO HAND CODE THIS!

```
if (feature1 > 0.8)
  if (feature2 > 0.9 && feature3 < 0.4)
    return false
  else if (feature1 > 0.9 && feature2 < 0.3 &&
    feature3 > 0.8)
    return false
  return true
else
  return false
```

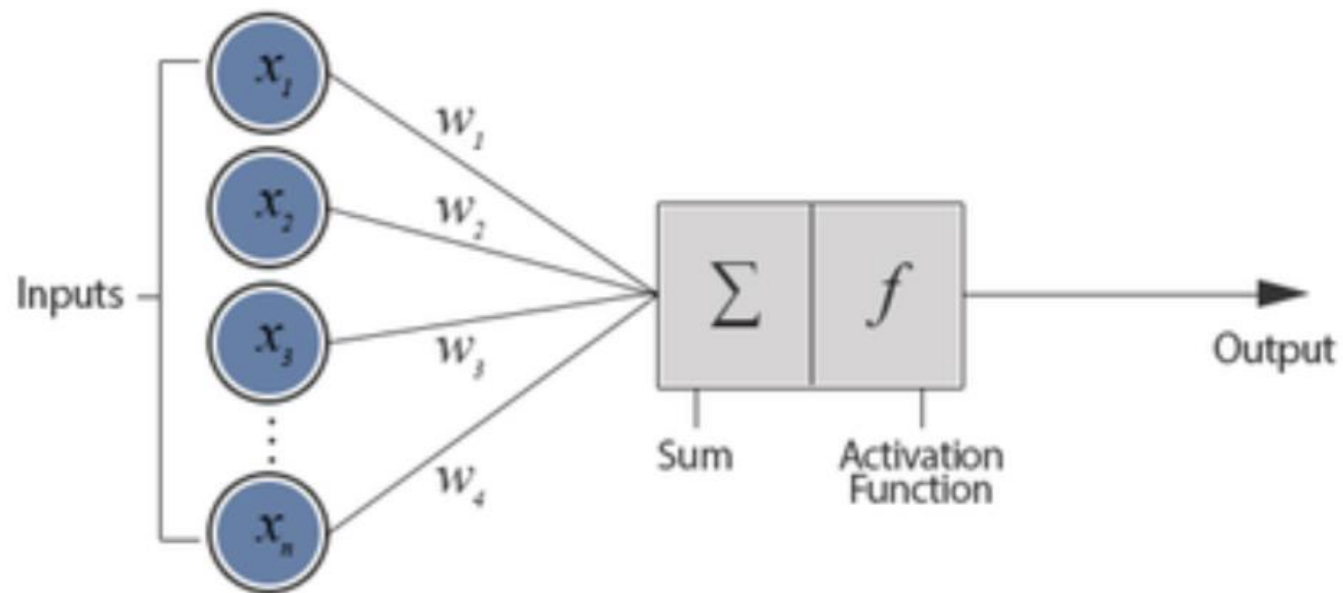
# MACHINE LEARNING

- Multiple tools and approaches
  - Decision trees
  - Support Vector Machines (SVMs)
  - Clustering
  - **Artificial Neural Networks**
  - And many others...

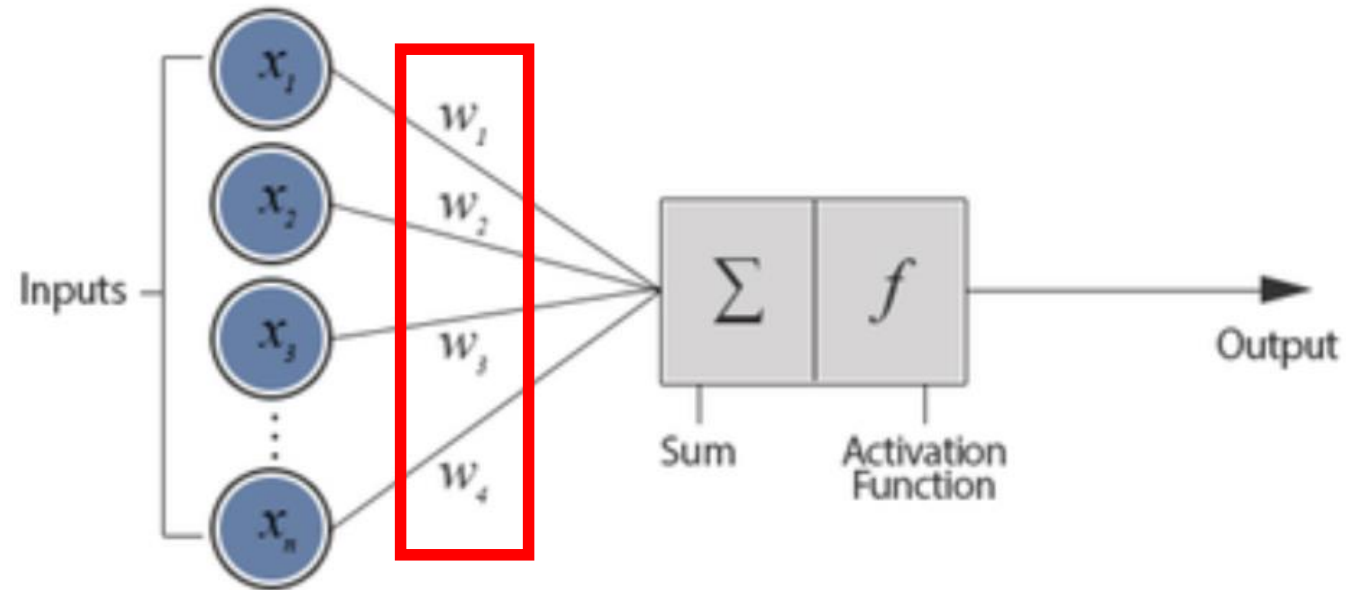
# ARTIFICIAL NEURAL NETWORKS



# ARTIFICIAL NEURON



# TRAINING (== LEARNING)



Find the weights that together produce the most accurate output possible with respect to the training data set



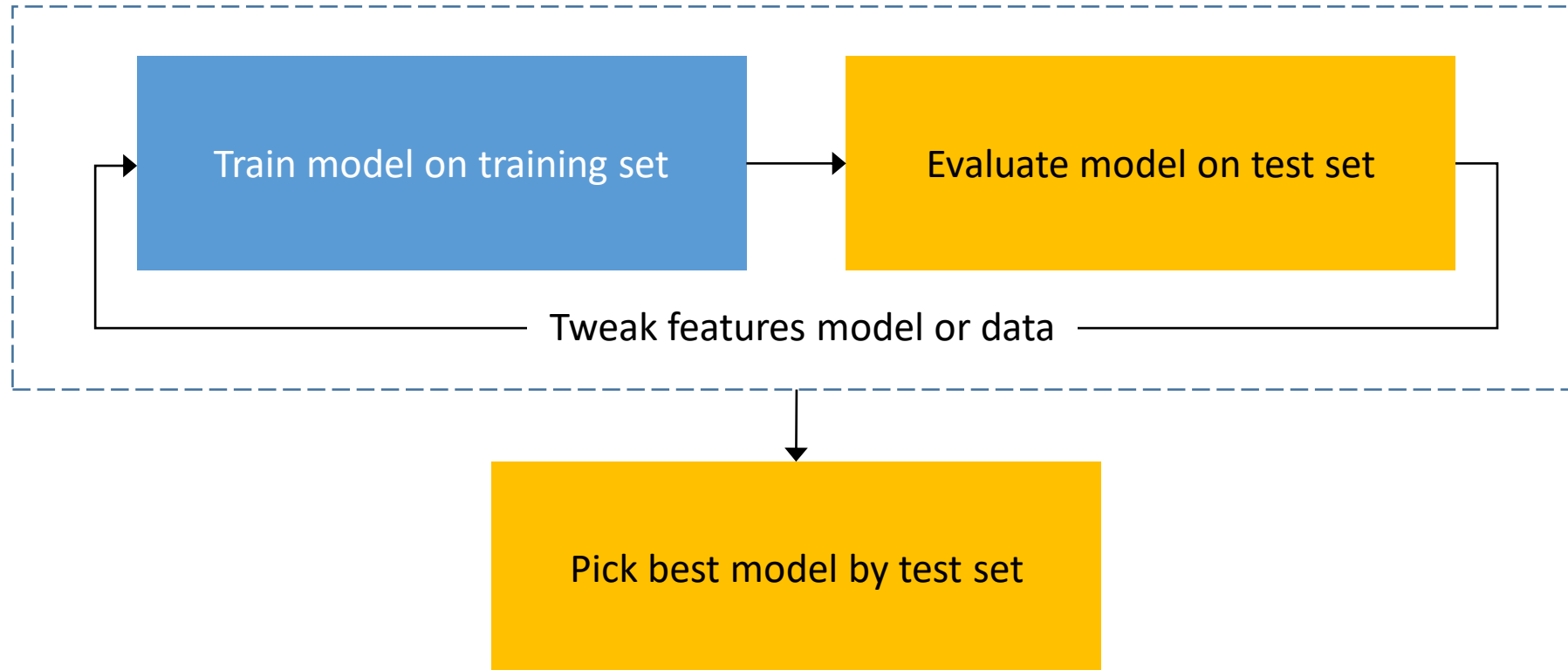
# DATA SETS\*



Features 1	Feature 2	Feature 3	Output
0.4	0.6	0.9	True
0.7	0.6	0.8	False
0.2	0.8	0.1	True

\* Validation data set is left out to keep things simple 😊

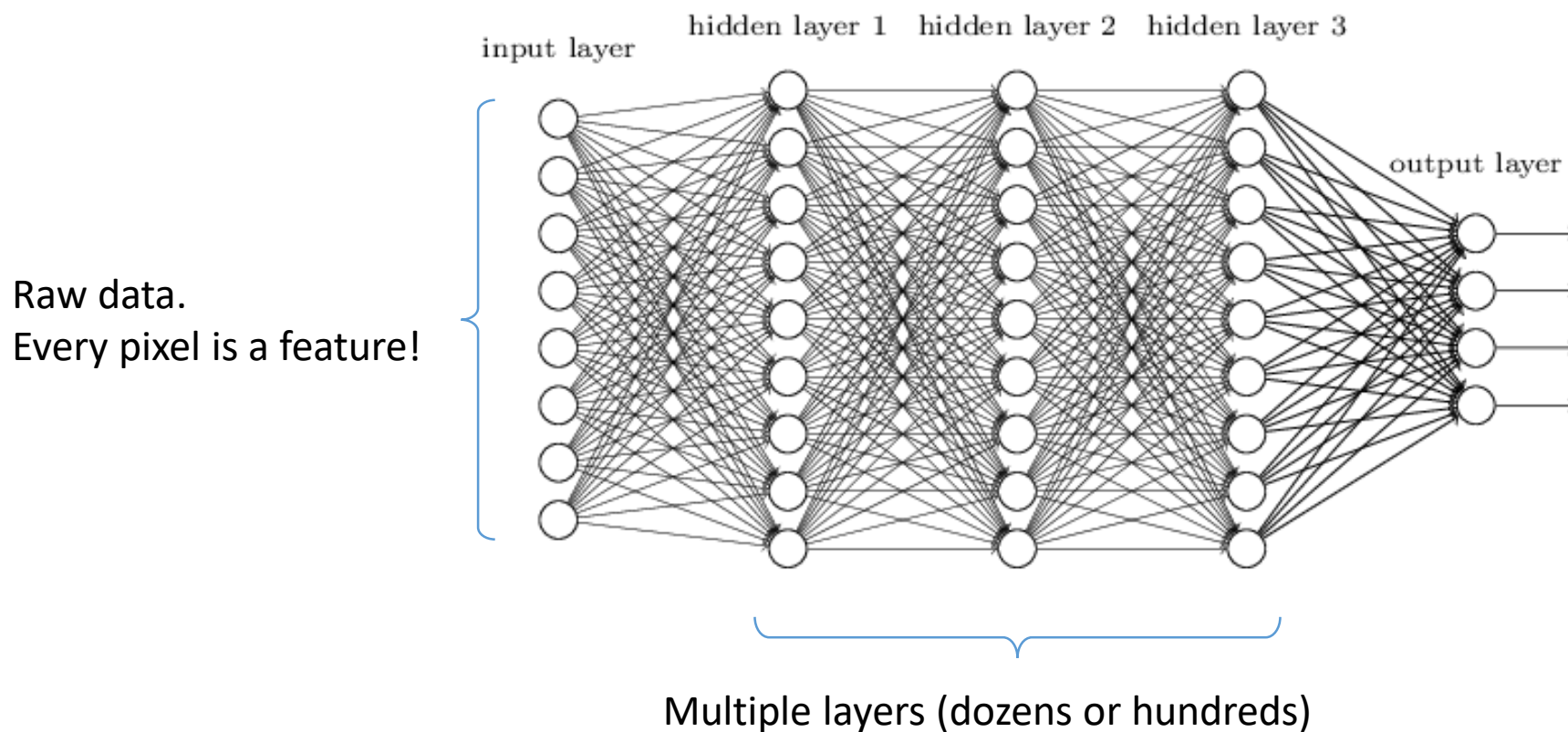
# MODEL DEVELOPMENT WORKFLOW



# MACHINE LEARNING CHALLENGES

- Feature extraction is difficult
  - Requires domain expertise
  - May imply non-trivial algorithms
- What if you don't know which features to select?
- What if you can't get your model to be accurate enough?

# DEEP LEARNING TO THE RESCUE



# DEEP LEARNING

- No need to manually define features. The network **truly learns on its own** from raw data
- Outperforms all other existing approaches
- The reason behind the current AI renaissance



**Andrew Ng** 

@AndrewYNg

Follow



Pretty much anything that a normal person can do in  $<1$  sec, we can now automate with AI.

6:11 PM - 18 Oct 2016 from [Osaka-shi Fukushima, Osaka](#)

541 Retweets 724 Likes



 67

 541

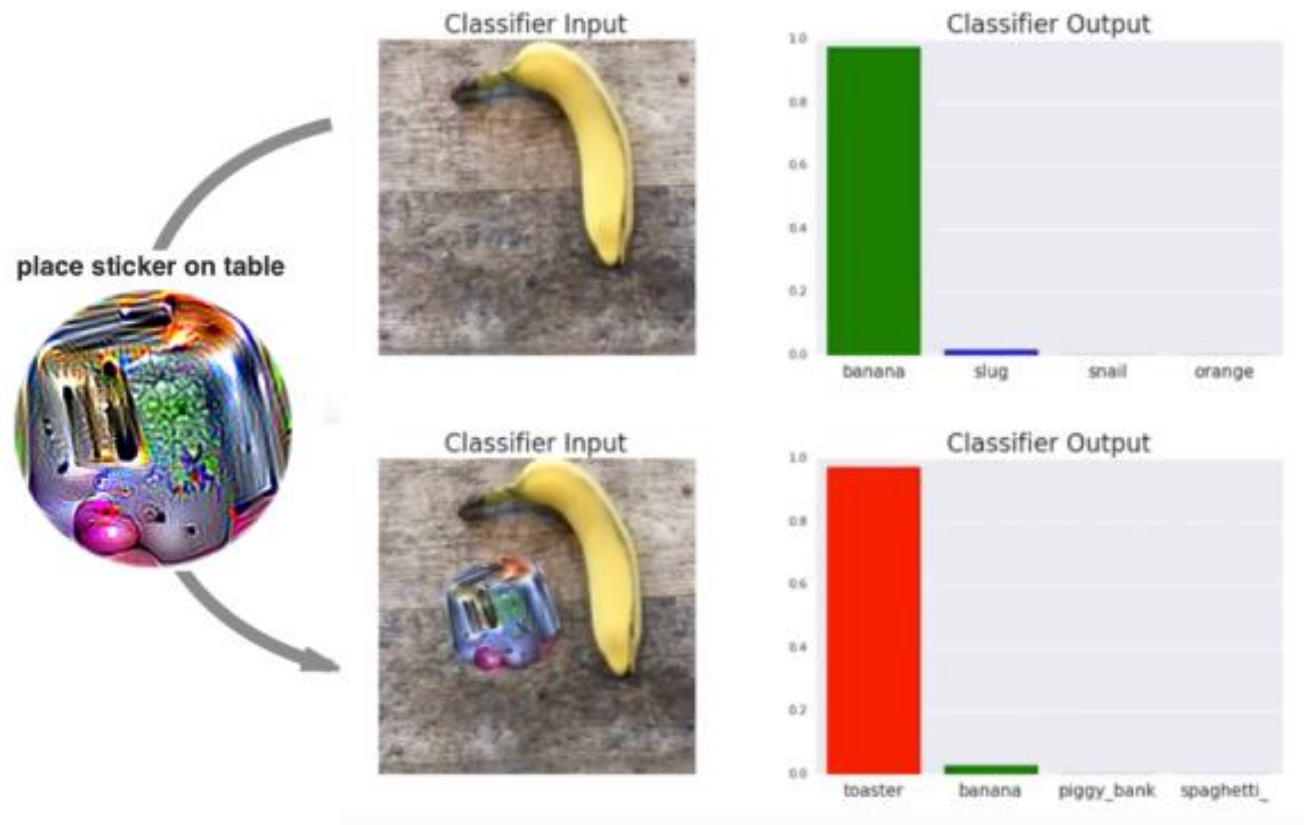
 724



# DEEP LEARNING CHALLENGES

- Obtaining data and labeling is a massive effort
- Experiments take hours or days instead of minutes
- Expensive to train and run
- Increasing accuracy by %1 can be very difficult
- **“Black box” (for testing and development)**
  - We can't really tell what features the network selected

# THE REALITY...





# AI @ APPLITOLS

# UI TESTING VS VISUAL TESTING

# TEST REQUIRED LOGIN DETAILS

## Sign in



or

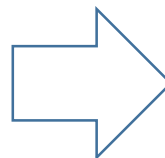
Email

Password

Forgot?

SIGN IN

New to AppliTools? Try for free.



## Sign in



or

Email

Required

Password

Required

Forgot?

SIGN IN


New to AppliTools? Try for free.

# A CYPRESS UI TEST

```
1 describe('Testing our login screen', () => {
2   it('properly warns on missing input', () => {
3     cy.visit('https://applitools.com/users/login');
4
5     cy.get('button[type=submit]')
6       .click();
7
8     cy.get('.validation-error')
9       .eq(0)
10      .should('contain', 'Required');
11
12     cy.get('.validation-error')
13       .eq(1)
14       .should('contain', 'Required');
15
16     cy.get('button[type=submit]')
17       .should('contain', 'Sign in');
18   });
19 });
```

# WHAT IF THE CSS CHANGES?

## Sign in

 SIGN IN WITH GITHUB

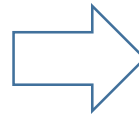
or

Email  
Required


Password [Forgot?](#)  
Required

**SIGN IN**

New to Applitools? Try for free.



## Sign in

 SIGN IN WITH GITHUB

or

Email  
Required

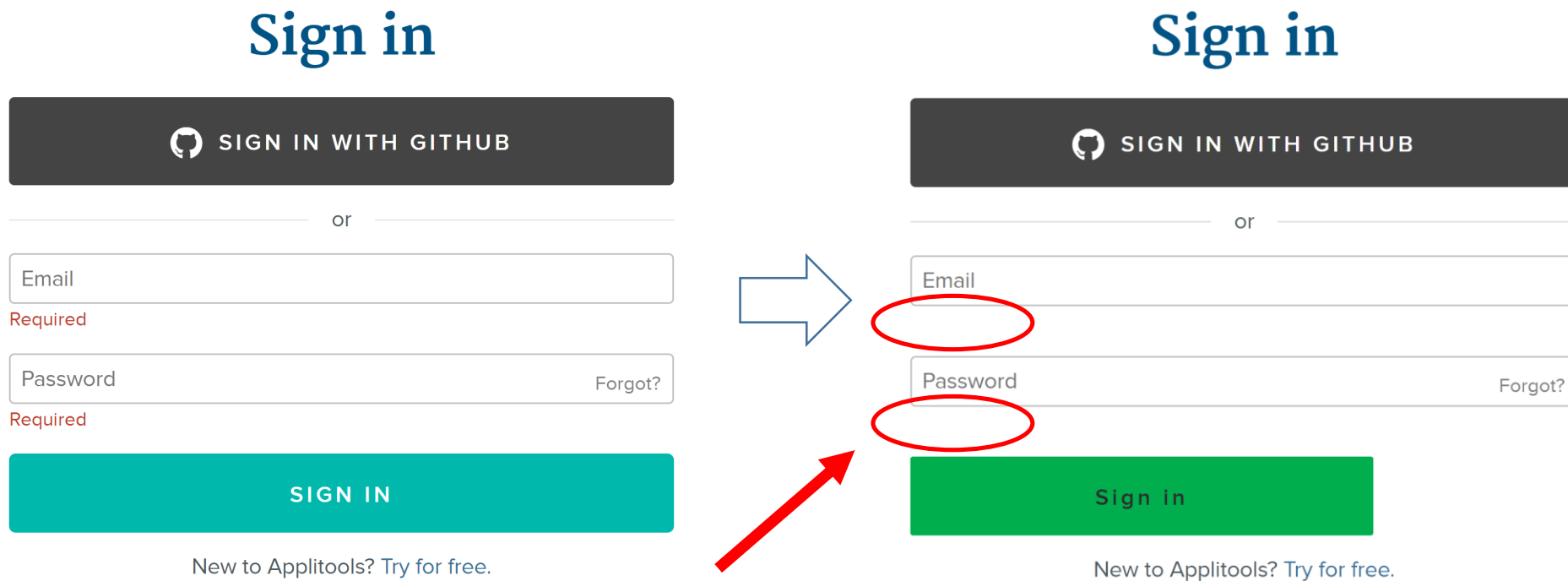
Password [Forgot?](#)  
Required

Sign in

New to Applitools? Try for free.



# WHAT IF THE CSS CHANGES?



# HOW CAN YOU TEST THE UI?

- An average web page has hundreds of elements
- Each element has dozens of visual attributes

The screenshot shows a web browser displaying the AppliTools website. The browser's developer tools are open, showing the DOM tree and the CSS styles for a selected element. The DOM tree shows a `span` element with the text "Our Story". The CSS styles panel shows various border-related properties, such as `border-bottom-color`, `border-bottom-style`, `border-bottom-width`, `border-image-outset`, `border-image-repeat`, `border-image-slice`, `border-image-source`, `border-image-width`, `border-left-color`, `border-left-style`, `border-left-width`, `border-right-color`, `border-right-style`, `border-right-width`, `border-top-color`, `border-top-style`, and `border-top-width`.

# HOW CAN YOU TEST THE UI?

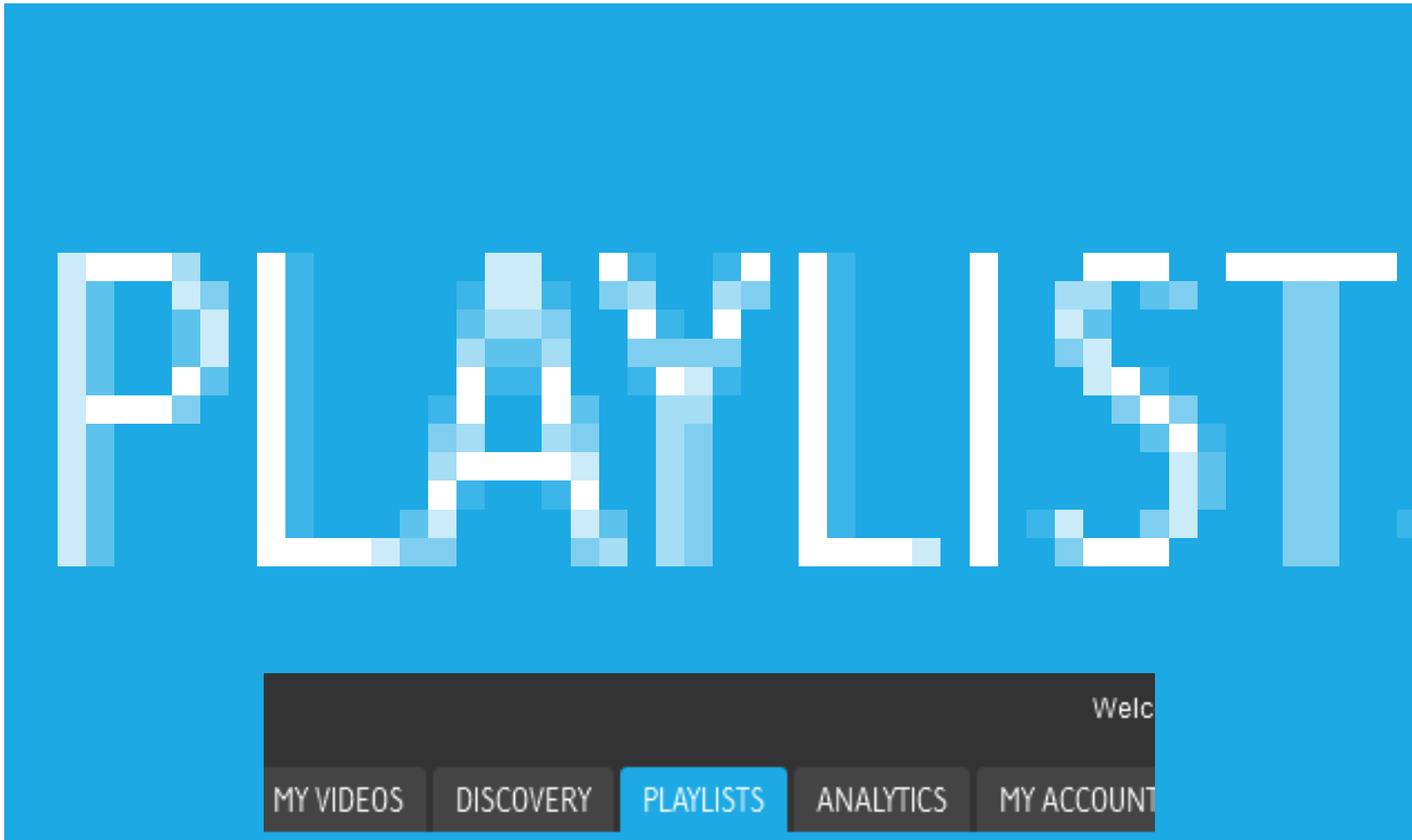
- Different form factors / viewport sizes?
- Different browsers (Blink, Gecko, WebKit, Trident)?
- Canvas, WebGL?
- New browser versions?

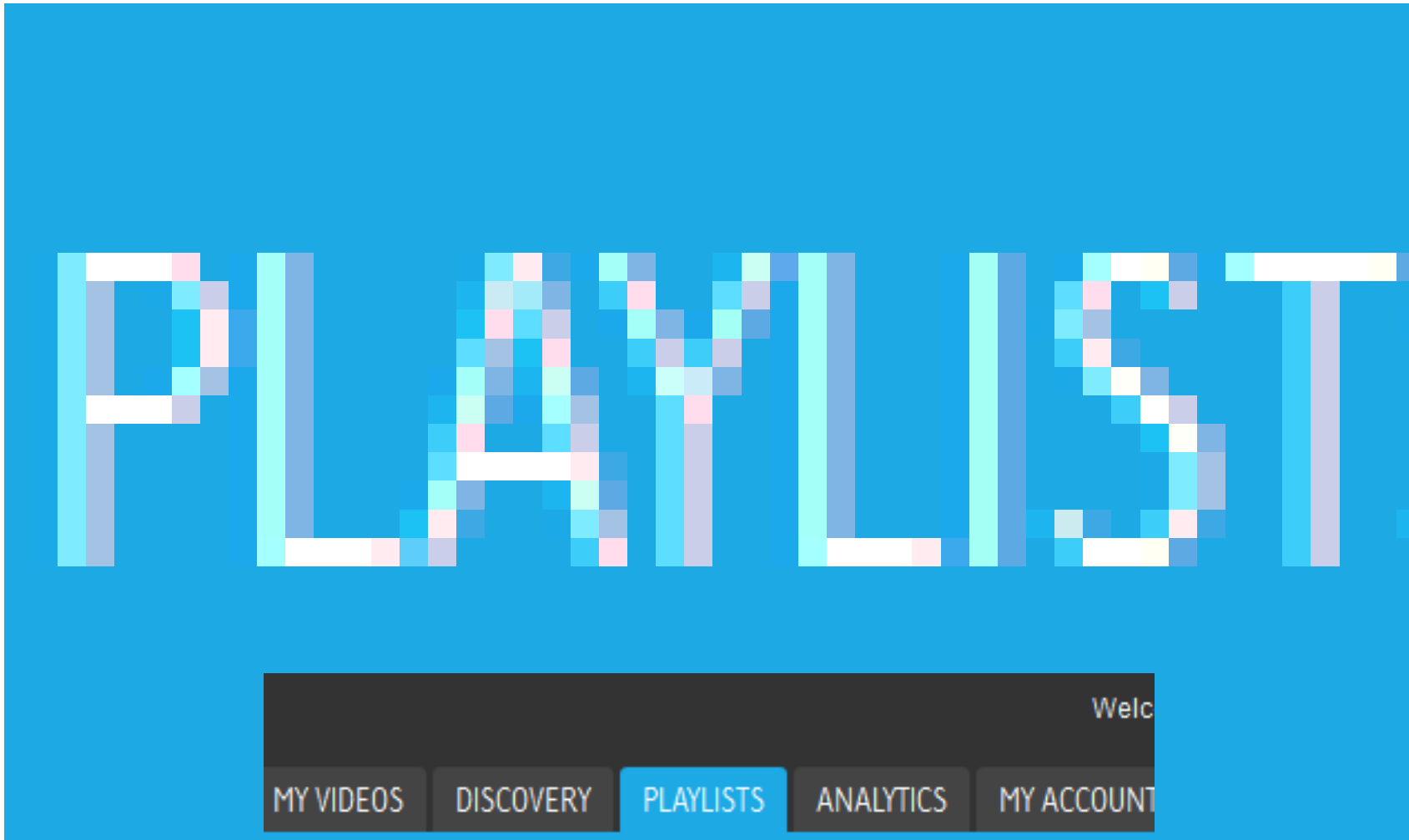


# VISUAL TESTING

# ADDING A VISUAL CHECKPOINT

```
1 describe('Testing our login screen', () => {
2   it('properly warns on missing input', () => {
3     cy.visit('https://applitools.com/users/login');
4
5     cy.get('button[type=submit]')
6       .click();
7
8     cy.get('.validation-error')
9       .eq(0)
10      .should('contain', 'Required');
11
12    cy.get('.validation-error')
13      .eq(1)
14      .should('contain', 'Required');
15
16    cy.get('button[type=submit]')
17      .should('contain', 'Sign in');
18  });
19 });
```





# IGNORE DISPLACEMENT DIFFS

The screenshot displays the AppliTools Eyes interface for a test named "Test: Pink8". The top navigation bar includes a "Test results" dropdown, the test name, and status indicators: "Steps: 1 | Unresolved: 1". On the right, there are icons for notifications, help, and a user profile. Below the navigation bar is a toolbar with sections for "VIEW" (containing icons for grid, zoom, and layers), "HIGHLIGHT DIFFS" (with left and right arrow icons), "ANNOTATIONS" (with an exclamation mark and speech bubble icon), and "AUTO MAINTENANCE" (with a "Scope: Default" dropdown). To the right of the toolbar are icons for thumbs up, thumbs down, and a save icon. The main content area shows a "Checkpoint | 1/1 Pink8" with a "Strict | 1024x768" resolution. A small icon with a not-equal sign ( $\neq$ ) is visible on the left. The central focus is a mobile application screenshot of a pharmaceutical website. A vertical orange bar highlights a displacement difference on the left side of the mobile view. On the right side of the mobile view, a mouse cursor is visible. At the bottom right of the main content area, there are zoom controls (left and right arrows, plus, and minus) and a small teal chat bubble icon.

VIEW: [Grid] [List] [Code] [Layers] [Diff] [Annotations] [Auto Maintenance]

HIGHLIGHT DIFFS: [Left Arrow] [Hand Icon] [Right Arrow]

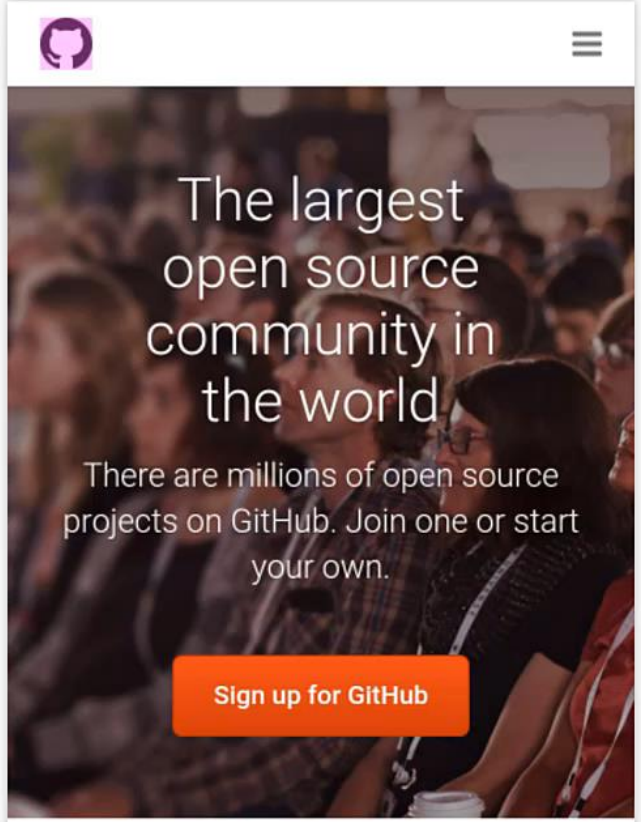
ANNOTATIONS: [Warning Icon] [Comment Icon] [Link Icon]

AUTO MAINTENANCE: Scope: Batch

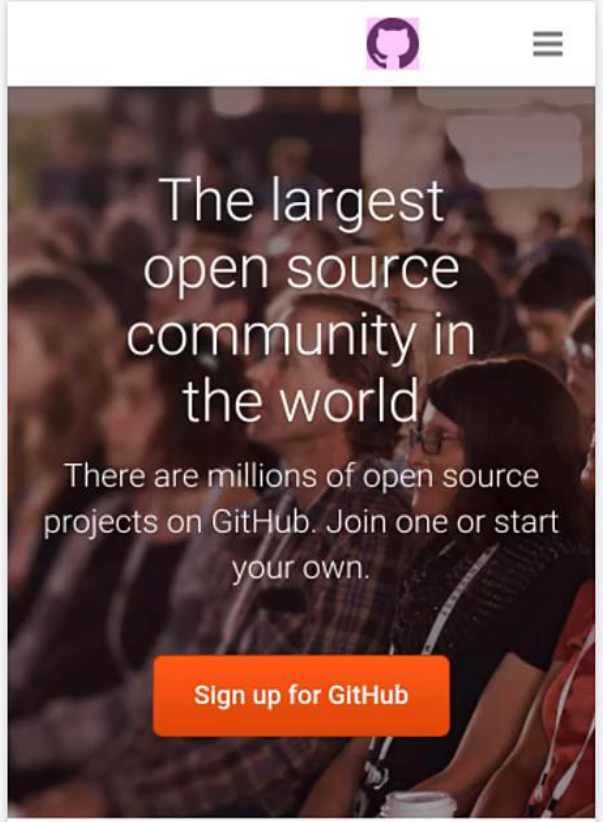
👍 🗨️ 🗑️

**Baseline** | 3/3 Open source  
 Layout | Android 5.1 | Chrome | 384x511 | [Google Nexus 4]

Highlight differences



**Checkpoint** | 3/3 Open source  
 Layout | Android 5.1 | Chrome | 360x511 | [Google Nexus 5]





**99%  
ACCURACY?**



**WE  
EXPECT 100%**







**99.9999%**  
**accuracy**  
**...**  
**and**  
**improving!**



# AI @ APPLITOLS

- A combination of multiple approaches
  - Dozens of sub-problems each solved with the most suitable approach
  - Multiple approaches to the same sub-problem
  - Try multiple decision paths if a result is not conclusive enough

# LAYOUT

## SECTIONS

## COLUMNS

## IMAGES

## TEXT LINES

## PARAGRAPHS

The screenshot shows a Yahoo Finance page with several elements highlighted:

- Top Navigation:** Gold 1,242.90 -0.06%, Crude Oil 94.72 -0.75%, EUR/USD 1.3557 -0.02%
- Left Sidebar:** Finance Home, My Portfolio, Market Data, Company News, Economic News, Personal Finance, Yahoo Originals, CNBC.
- Main Content Area:**
  - Large Article:** A 'Second Wind' for Earnings Could Quiet the Bears' Huffing. Includes sub-headlines: "One Key Slice of the Market Is Crumbling While Stocks Set Record Highs" and "What a Yellen Paper Really Means for Stocks".
  - Grid of Smaller Articles:** Includes "Obamacare: Your Questions Answered", "Why Best Buy May Be the Worst Play at Christmas", "Black Friday Ain't What It Used to Be", and "Here's Where You Can (and Can't) Buy a Home Or Your Salary".
  - Bottom Section:** "Newbie Investors: Listen Up", "Insight: Kim Jong Un, North Korea's master builder", and "Big Insurance Can Now Bypass Obamacare Website Tech Companies Still Waiting".
- Right Sidebar:** Market Data, Currencies, and Yahoo Finance.

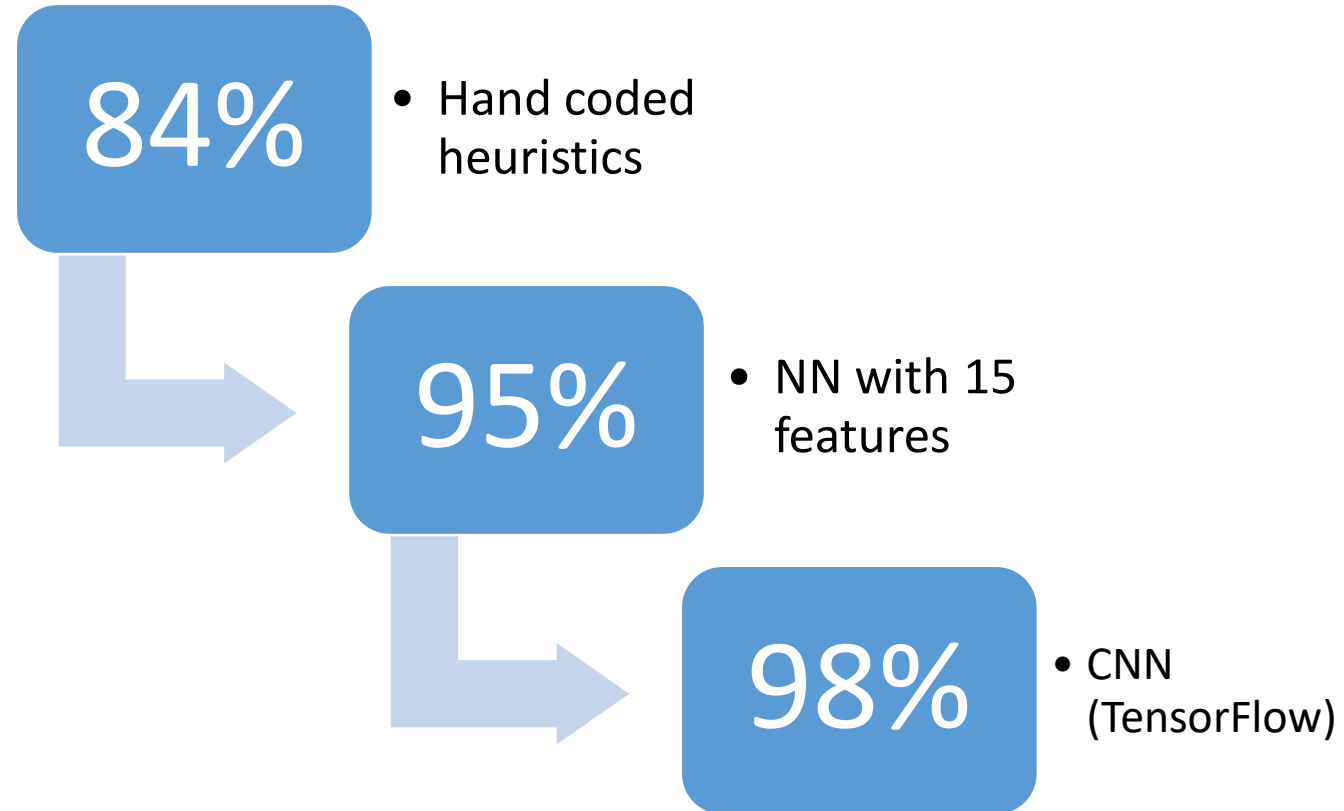
# PROBLEM IS SEGMENT PART OF AN IMAGE OR TEXT?



# WHY IS IT DIFFICULT TO SOLVE?



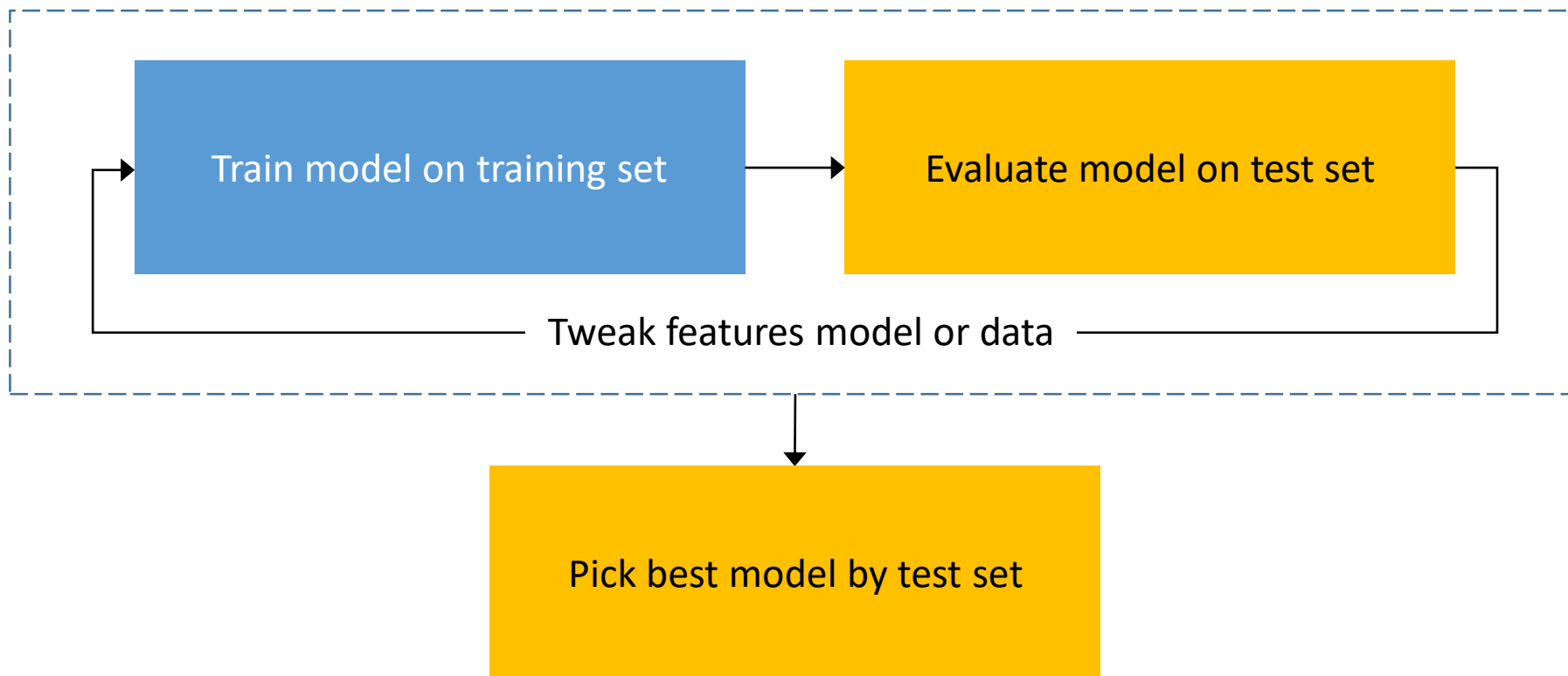
# SOLUTION EVOLUTION



# AI TESTING TIPS

# TIP #1

Test your test set!



# TIP #2

Accuracy is not a sufficient quality measure for the model

Finance Home  
My Portfolio  
Market Data  
Company News  
Economic News

**100% text**



**Gray area**

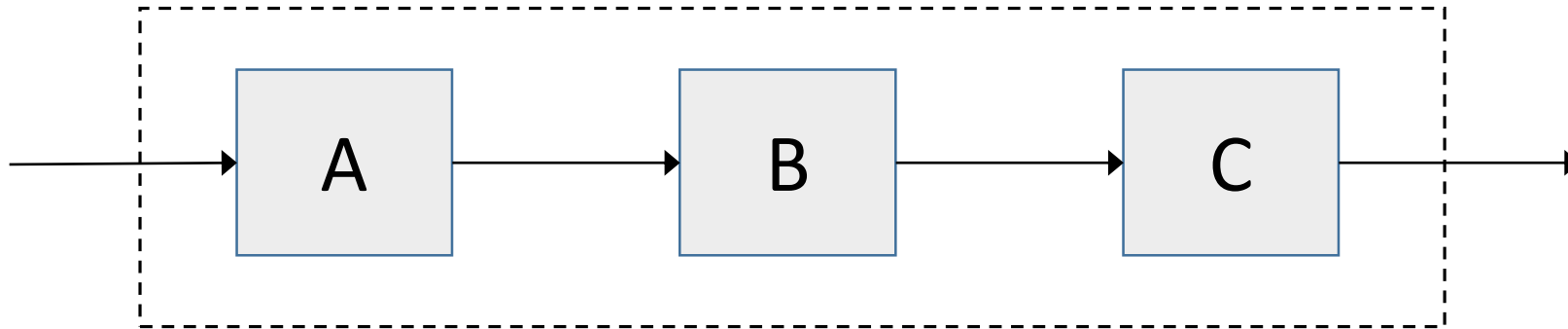


**100% image**



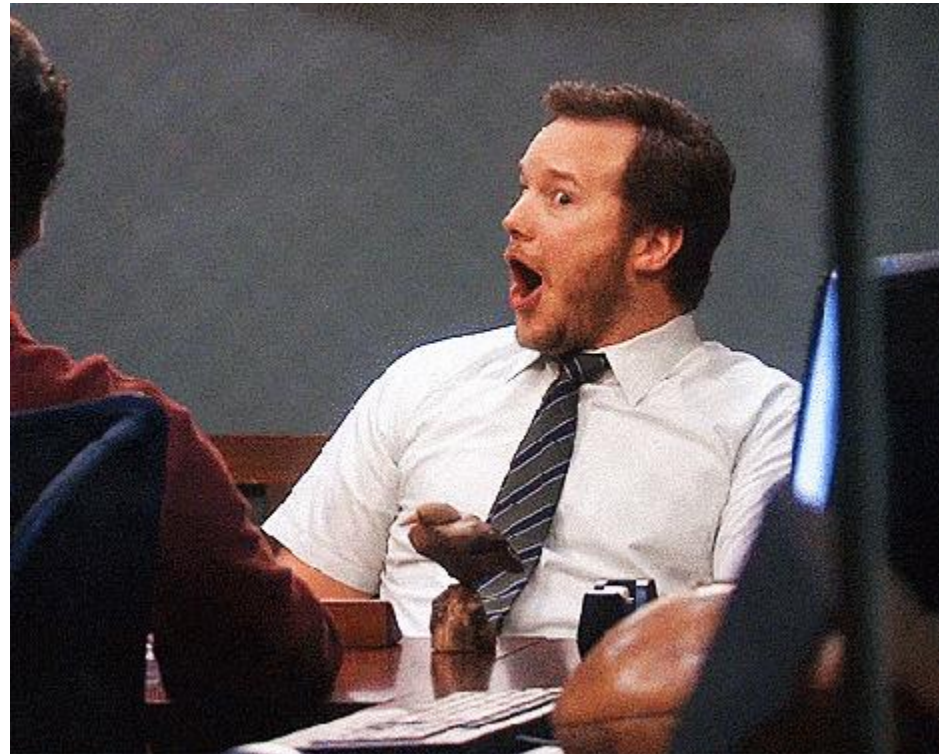
# TIP #3

Model improvements may cause overall degraded results!



# TIP #4

Cover *Open* bugs with regression tests



# TIP #5

Test the time & space performance of your algorithms

# TIP #6

Expect similar results rather than exact

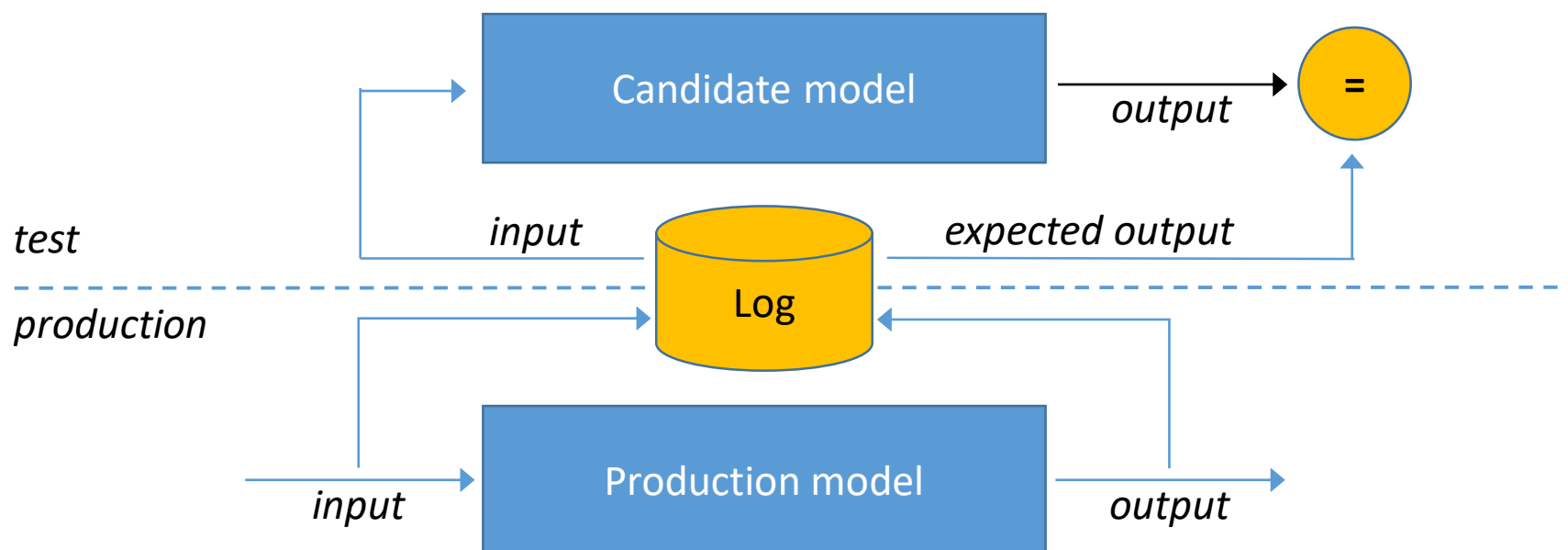
```
Assert.True(GeometryUtils.IsSimilar(textBounds, new Rectangle(130, 100, 120, 20), 2));
```

Is *\*much\** more robust than

```
Assert.AreEqual(textBounds, new Rectangle(130, 100, 120, 20));
```

# TIP #7

Run candidate models in parallel with production



# HOW CAN YOU LEVERAGE AI?

# HOW CAN YOU LEVERAGE AI?

- The easiest wins are possible when
  - You have the (labeled) data
  - You can easily outperform current approaches
  - You can tolerate errors and easily detect them
- Otherwise its going to be **very** difficult
  - Buy a solution

# SELF-ADJUSTING LOCATORS

Project: the-internet\*

Executing ▾

login succeeded\*

	Command	Target	Value
1	<i>open</i>	/	
2	<i>click</i>	linkText=Form Authentication	
3	<i>type</i>	id=username	tomsmith
4	<i>type</i>	id=password	SuperSecretPassword!
5	<i>send keys</i>	id=password	#{KEY_ENTER}
6	<i>assert element present</i>	id=flash	You logged into a secure area!\n

Runs: 1 Failures: 0

Log Reference

Command:  //

Target:

Value:  id

Description:  css:finder

Opens Window:  css

xpath:attributes

xpath:idRelative

xpath:position



# TEST RESULTS DE-DUPLICATION

The screenshot displays the AppliTools Eyes interface. On the left, a sidebar lists the last 30 batch runs with checkboxes and status indicators (Failed or Passed). The main area shows a detailed view for a batch named 'Github', including its start time, duration, and test results. Below this, a grid of six visual comparison panels is shown, each with a '≠' icon and a star icon, indicating failed tests. The panels are labeled with their respective test names and counts.

**Batch: Github**  
Started: 03/29/2016 at 2:13 PM | Duration: 07:45:16 | Tests: 19 | Unresolved: 19  
**Failed** Resolved diffs 0% | 76 unresolved steps

**Last 30 batch runs**  
No filtering

- Layout2**  
06/08/2016 at 5:07 PM  
**Failed**
- Layout2**  
06/08/2016 at 5:00 PM  
**Passed**
- Homepage**  
05/18/2016 at 7:44 PM  
**Failed**
- Homepage**  
05/18/2016 at 7:43 PM  
**Passed**
- Github**  
03/29/2016 at 2:13 PM  
**Failed**
- Wikimedia**  
03/17/2016 at 4:30 PM  
**Failed**

Visual comparison results:

- 1/4 Home
- 2/4 Personal
- 3/4 Open Source
- 4/4 Business
- 1/4 Home
- 2/4 Personal

# AND MORE...

- Run tests based on changed files
  - And run the full suite frequently enough to catch misses
- Organize test results in order of importance
  - Anyway better than random order
- Detect response time anomalies
- Show new errors before common ones

# QUESTIONS?

[Adam Carmi](#) (@carmiadam)

Co-Founder and CTO at [Applitools](#)