Consistency is a Skill

Evaluating umpire performance with machine learning

Who I am

I tweet about random nonsense: @PeterKBonney

Some of my code is on github: github.com/pbonney

I have a (non-baseball) company: @Vendorful

I occasionally write stuff: The Hardball Times

This photo has no relevance but it's pretty cute

Machine learning and umpire consistency, in 12 minutes or less:

Background and goal

How to think about machine learning

Modeling strike zones, with and without ML

Evaluating ML classification models

Umpire consistency is a skill

You too can use machine learning

@PeterKBonney
github.com/pbonney

Background and goal

How should you evaluate an umpire?

Tom Tango's idea: Model each umpire's *individual* strike zone with goal of finding who calls the zone most consistently with *themselves*[1]:

$$UCS_{pitch} = (2p - 1) * (c - p)$$

Examples:

$$p = 0.9, c = 1 \rightarrow UCS = 0.08$$

$$p = 0.9, c = 0 \rightarrow UCS = -0.72$$

To implement UCS, we need a model of *p*. Enter machine learning...

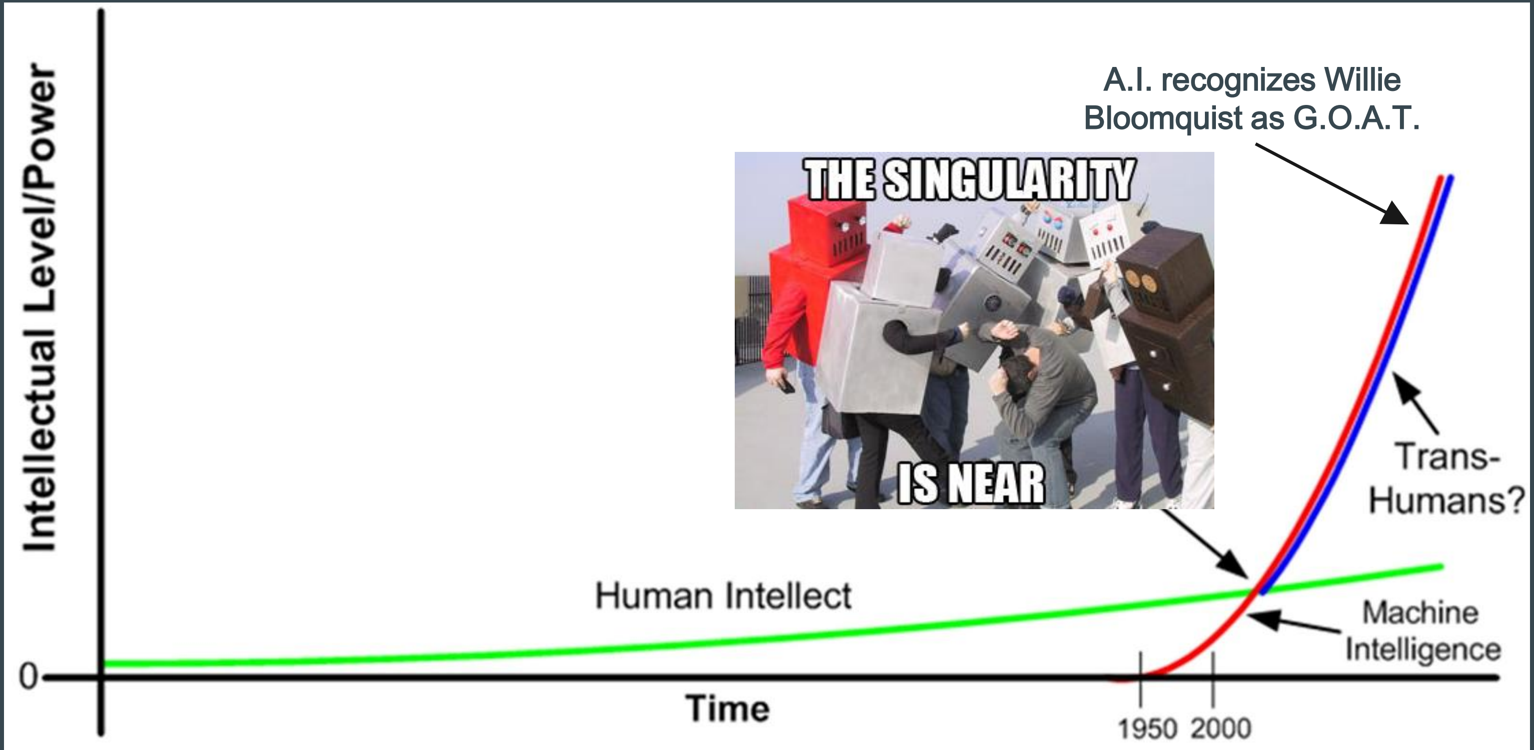
[1] http://tangotiger.com/index.php/site/article/evaluatiing-the-effectiveness-of-an-umpire-effectively

@PeterKBonney
github.com/pbonney

How to think about machine learning

@PeterKBonney
github.com/pbonney

Obviously we must start by talking about the Singularity



But seriously, machine learning is like regression

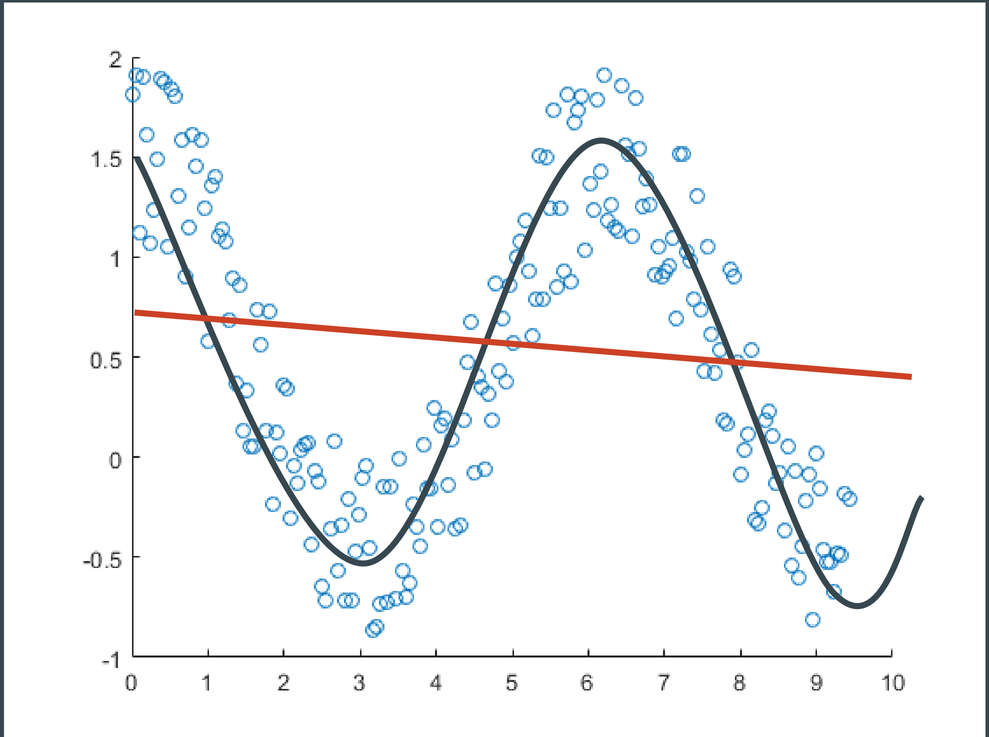
Regression:

$< x_1 , x_2 , x_3 , \ldots > \;+\; < y > \longrightarrow (math) \longrightarrow f(x_1 , x_2 , x_3 , \ldots) \sim y$

Machine learning:

$< x_1 , x_2 , x_3 , \ldots > \;+\; < y > \longrightarrow (math) \longrightarrow f(x_1 , x_2 , x_3 , \ldots) \sim y$

ML's wheelhouse: complicated, unconstrained data



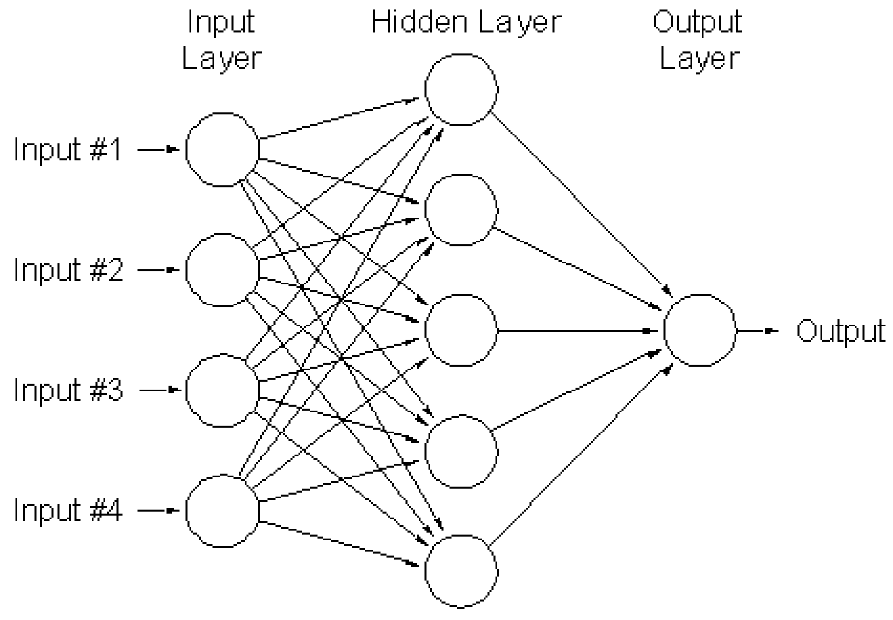
Data:
"Hi everyone, I'm a wave!"

ML model:
"Why yes you are... mmm, curvy..."

Regression:
"Nah, pretty sure you're a line."

Neural networks are the David Ortiz of ML models

(Because they've been around forever and are still useful, not because they can't run and don't play D.)



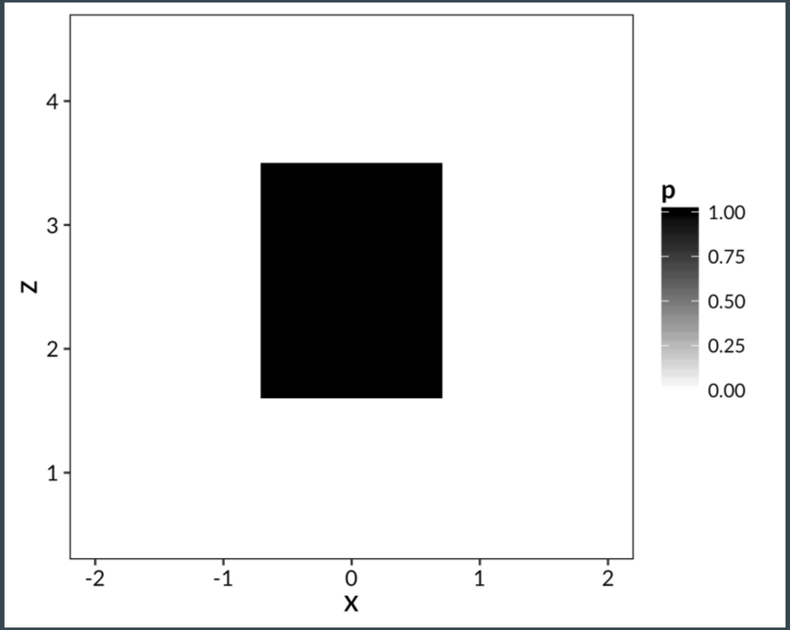
All the ML models in this presentation are neural network models

Modeling strike zones with and without ML

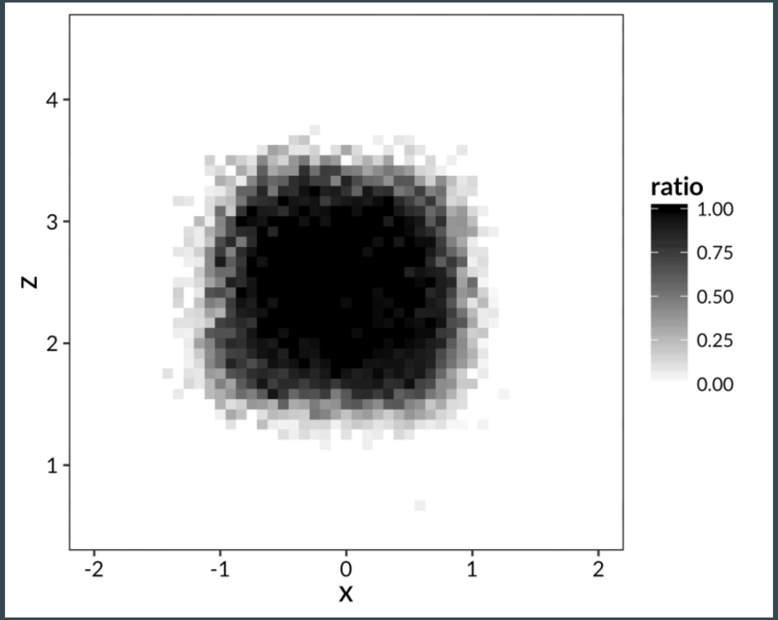
@PeterKBonney
github.com/pbonney

How do you model an umpire's called strike zone?

Draw a box?

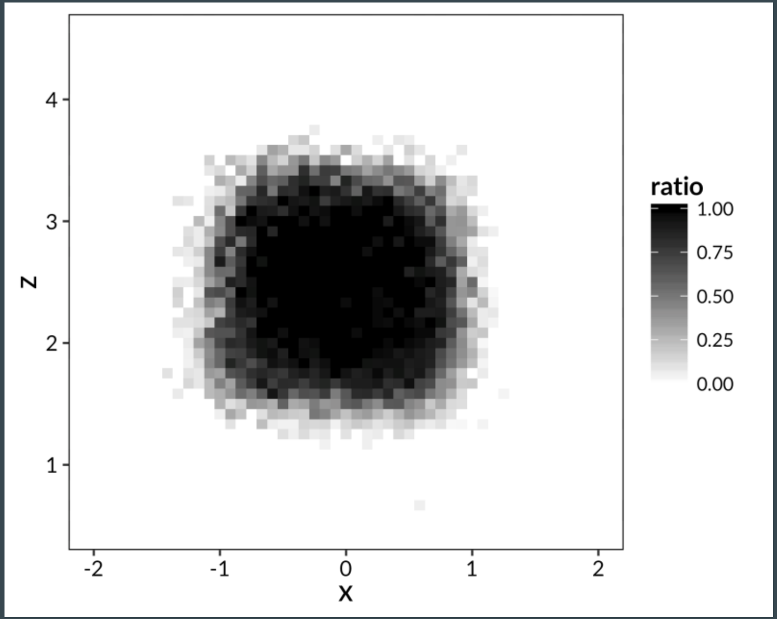


Divide into small buckets and compute historical frequency?

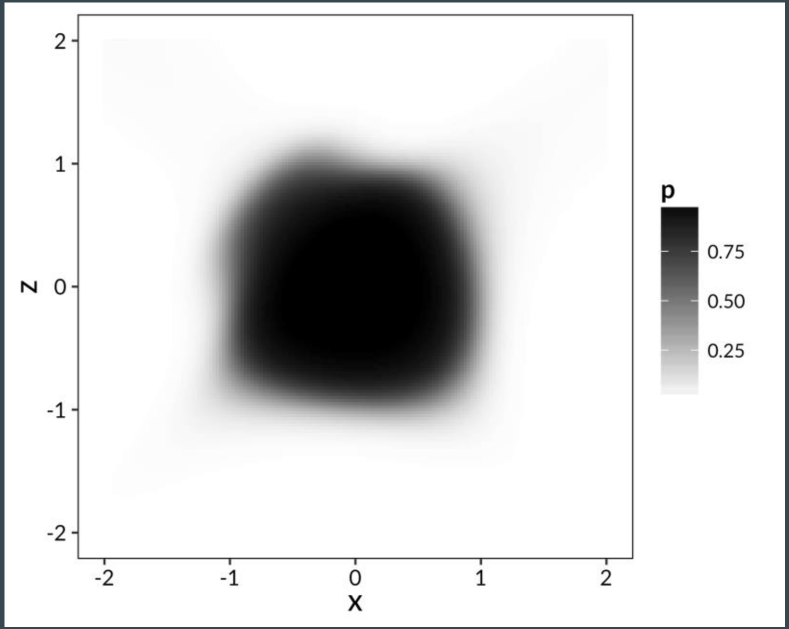


@PeterKBonney
github.com/pbonney

Actual called pitches



NN model of umpire behavior



Or… teach a neural network to mimic the mind of an umpire!

@PeterKBonney
github.com/pbonney

Evaluating ML classification models

Receiver Operating Characteristic (ROC)

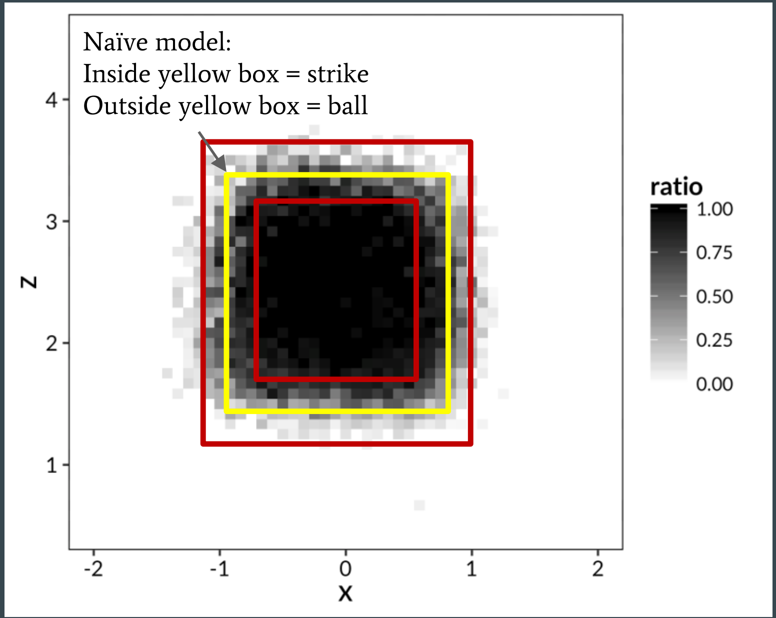
A common measure of binary classification systems. Plot “true positive” rate vs. “false positive” rate at various discrimination thresholds then compare them.

TL;DR: Draw a graph, larger area under curve == better classification

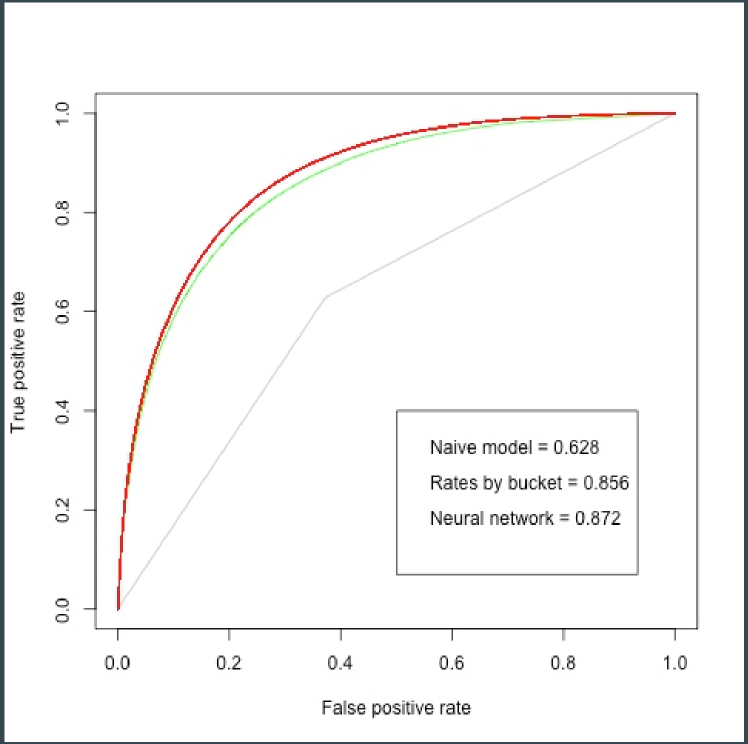
(Name comes from use in WWII evaluating the performance of radar operators)

How does our simple ML umpire model perform?

Recall this plot of ball/strike frequencies:



Heart of zone and way out of zone are too easy, so let's just look at area between the red boxes.



@PeterKBonney
github.com/pbonney

Umpire consistency is a skill

Now that we have a good modeling methodology, recall:

$$UCS_{pitch} = (2p - 1) * (c - p)$$

Examples:

$$p = 0.9, c = 1 \rightarrow UCS = 0.08$$

$$p = 0.9, c = 0 \rightarrow UCS = -0.72$$

According to UCS, strike zone consistency is a skill

Split-half reliability:

- Divide pitches into odd/even samples of size N, calculate UCS for each
- Compute correlation between odd/even samples at N=100, N=200, etc.
- Find N where correlation between odd/even samples 0.5

I found: N ~ 940

So after c. 6 games UCS looks like equal parts luck and skill.

(Which means it IS a skill!)

Read more about UCS in the 2016 Hardball Times Annual

@PeterKBonney
github.com/pbonney

You too can use machine learning

How to train a neural network model in one easy step

With R ‘`neuralnet`’ package:

```
m <- neuralnet(c~px+pz,data=df,hidden=h,linear.output=FALSE)
```

Where:

- `c` is the call (1 for strike, 0 for ball)
- `px` and `pz` are the PITCHf/x horizontal and vertical locations
- `df` is a data frame of training data (with columns `c`, `px` and `pz`)
- `h` is the number of “hidden” neurons

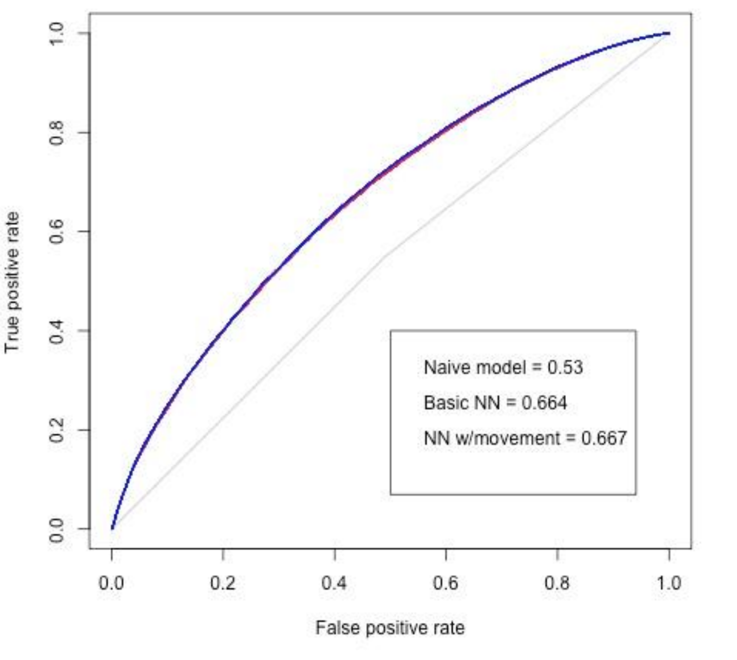
ML can do more… let's add other variables to the model

ML is *particularly* good at incorporating variables that don't easily fit in simple models.

For example... what happens if we add horizontal and vertical pitch movement to the inputs?

We'll also focus on the *most* borderline pitches: in locations historically called as strikes 25-75% of the time.

Umpires aren't strongly influenced by pitch movement!



It has never been easier to give machine learning a try

Machine learning as a service:

- Cloud ML products from Google, Amazon, Microsoft (Azure)

Run it at home for free:

- ML packages for R and Python

Use with caution: don't overfit, don't mix training and evaluation data, don't treat models cruelly and p*ss off our future robot overlords.

Feel free to reach out if you want a hand getting started, but note that I'm a dabbler, not a data scientist or computer scientist or any other kind of scientist.

100%

of this presentation is now done.

Thank you.

@PeterKBonney
github.com/pbonney