What is "Machine Learning"?

Or rather, *why* is it?

Machine learning applications

Can you think of an example? (Write it down)

- Electronic health records to predict which patients will require more care
- Genome sequence data for tissue samples to detect different kinds of cancer
- Text scraped from social media to predict events of social unrest, or track spread of misinformation
- Tech platform user data to target relevant content, or detect policy/regulation violations
- Learning adaptive control of robot prosthesis

etc...

Machine learning, proper

These *application* examples help motivate the value of ML

(Actually, much of the value comes from work specific to the application, like the creation/gathering/processing of the data, and the real world *actions* taken based on the output of ML)

We'll use "ML" to refer to the *theory and general methods*

(Skills like gathering and cleaning data are very useful--and we'll practice them a little--but they're not the main focus of this course)

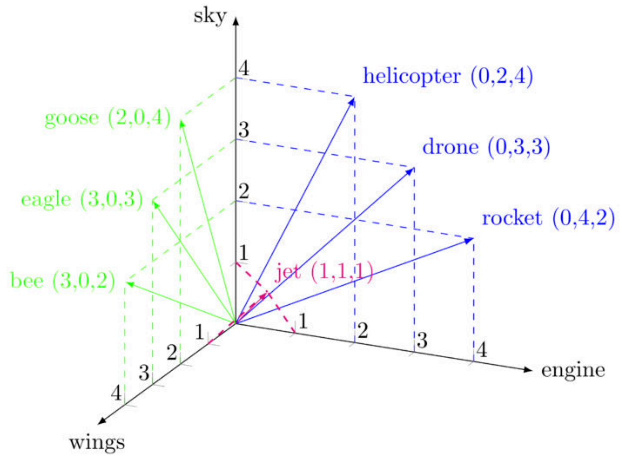
What is "artificial intelligence"?

(Don't tell anyone I said this, but) It's a collection of computational tools that people use to create mathematically structured data out of non-mathematically structured data

e.g. a (possibly randomized) function from $\{\text{some image file type}\} \to \mathbb{R}^d$ for some d.

e.g. word embedding for text data. (Image credit)

We'll usually assume our data is already mathematically structured



Abstraction and notation

Along came some data which someone formats as a collection of p distinct variables

$$X = (X_1, X_2, \ldots, X_p) \in \mathbb{R}^p$$

We assume **each observation is a point in a vector space** (which we also implicitly assumed is finite-dimensional, and that's OK by any practical standard)

Question: is there a Y variable?

Think about your application example (the one you wrote down)

Categories of ML tasks

Supervised learning (most of the term)

Often we focus on one variables, name it Y, and give it the special status of being an "outcome"/"response"

Unsupervised learning (a bit of this)

If there is no obvious choice of an outcome variable, we may just wish to "find structure" in the X variables. Clustering, dimension reduction

Other tasks (probably not these)

Ranking, anomaly detection, network data, embeddings, correspondence, recsys, multi-armed bandit, etc...

Supervised ML sub-categories

If Y is numeric: **regression**

- Concentration levels of a protein (disease status/severity)
- Selling price of a house

If Y is categorical: **classification**

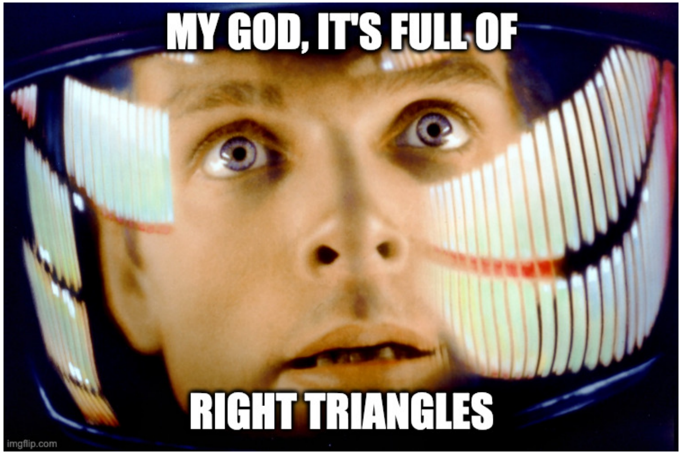
- Should this item be flagged for (human) review? yes/no
- Identify type of cancer: lymphoma, sarcoma, neuroblastoma, etc

Special cases

- Y is binary with rare cases, e.g. anomaly detection
- Y is a time to event, survival analysis
- Multi-class, hierarchical classes, etc

Focus on regression

- Simpler math (orthogonal projection, Euclidean geometry)
- Intuition pump for other cases
- Often underlies other cases



- e.g. binary classification by thresholding a numeric **score**, or ranking (ordinal outcome) / set selection (select items with $\mathrm{top}\text{-}k$ scores)

How to predict Y from X?

- Would be sweet if $\exists f$ such that the graph of the function $y = f(x)$ fit the data perfectly
- **Problem**: what if $(x_1, y_1) = (1, 0)$ and $(x_2, y_2) = (1, 1)$?
- **Problem**: even our most tested and verified physical laws won't fit data *perfectly*

Solution: applied mathematics

For any function f we can always write $\varepsilon \equiv y - f(x)$. Look for an f which makes these "errors" "small" for the observed data

Uncertainty opens the door for probability

- Assume a probability distribution (adequately) models the data/errors

Define a good function as one that minimizes

$$\mathbb{E}[\varepsilon^2] = \mathbb{E}\{[Y - f(X)]^2\}$$

- Assume the data/error is sampled independently

Motivates the **plug-in principle**: compute an estimate $\hat{f}$ of the good function f by solving the corresponding problem on the dataset, i.e.

$$\text{minimize} \sum_{i=1}^{n} \left[y_i - \hat{f}\,(x_i)\right]^2$$

Very useful assumptions!

The *why* of machine learning

"it works"

- Squared error $\rightarrow$ simpler math

(we'll come back to this and consider other loss functions)

- i.i.d. sampling $\rightarrow$ simpler estimation, justifies generalisation

(we'll come back to this too)

Minimizing expected squared-error also gives us...

One of the most powerful ideas in all of statistics

$$\mathbb{E}\{[Y - \hat{f}\,(X)]^2\} = \mathrm{Var}(\hat{f}\,) + \mathrm{Bias}(\hat{f}\,)^2 + \mathrm{const.}$$

the bias-variance trade-off

Are the errors systematic (bias) or not (variance)?

With model complexity:

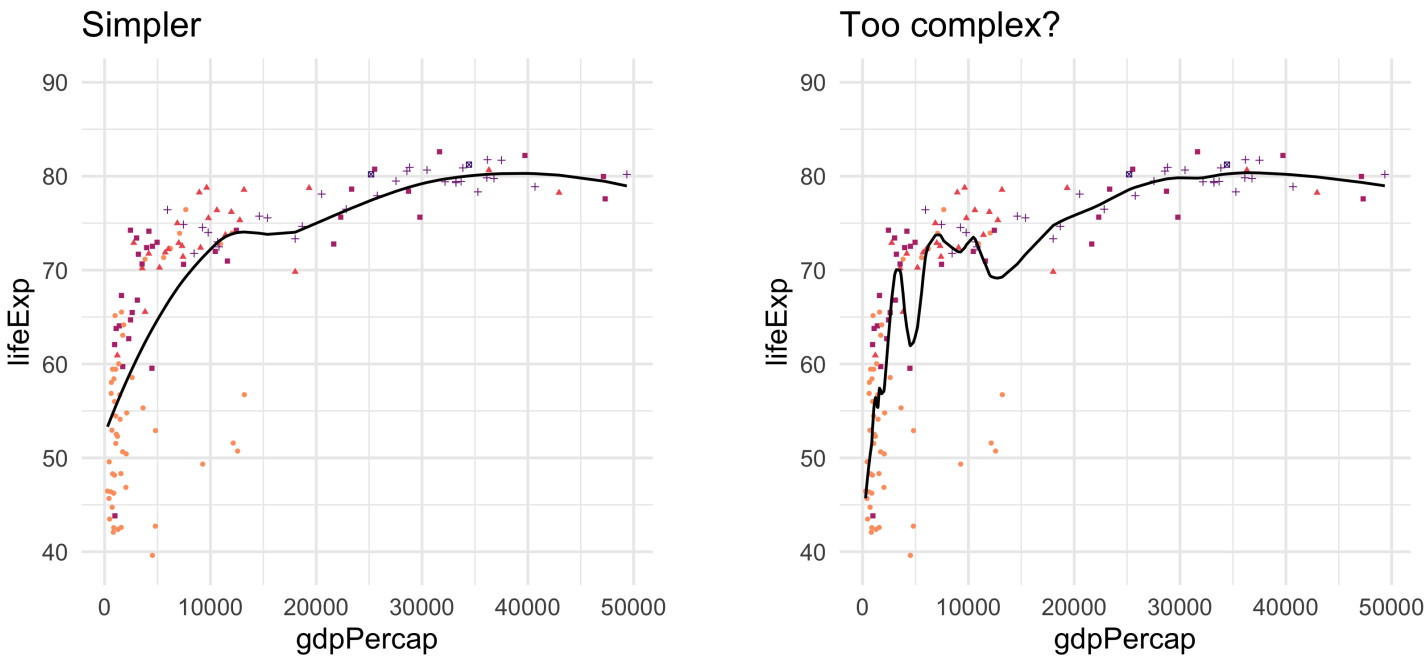
Typically, more complex models have lower bias and higher variance

And typically, there is a "right amount" of complexity

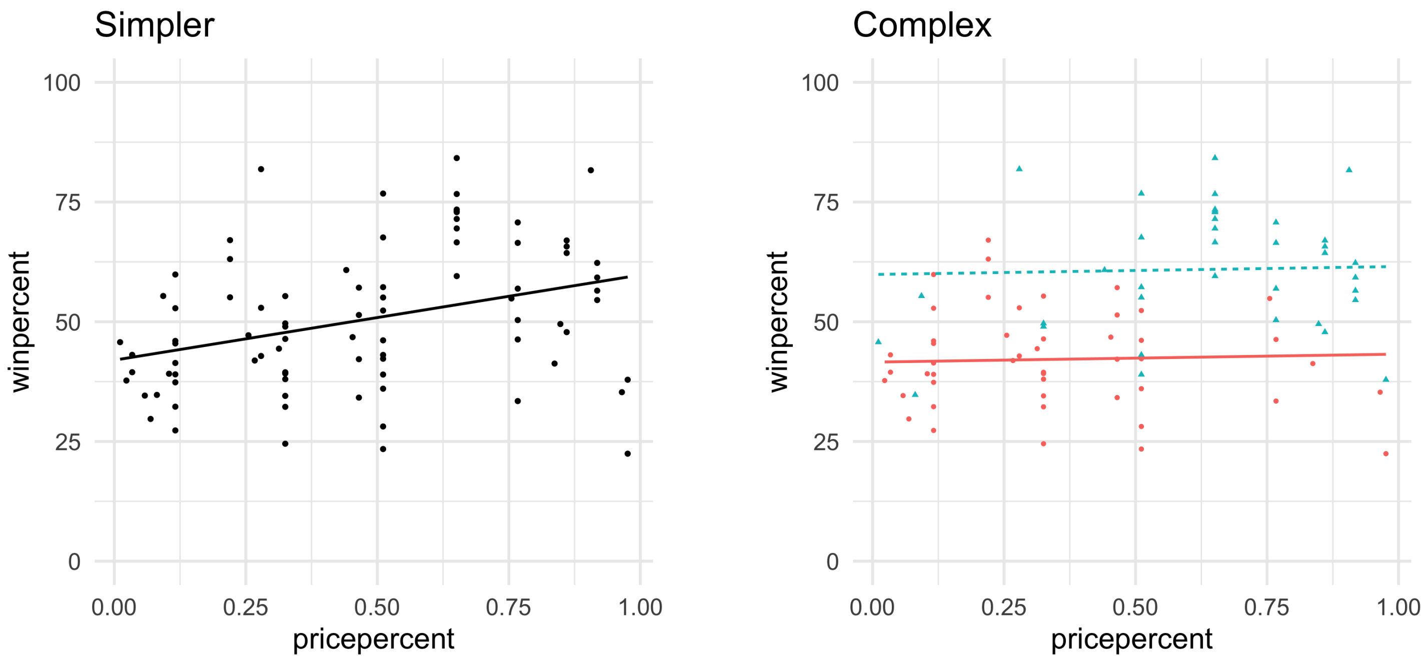
- Too low? Little variance, but overwhelming bias
- Too high? Little bias, but overwhelming variance
- Just Right: [insert happy statistician meme]

statisticians celebrate finding the right model complexity

gapminder example



candy ranking example



Evaluation: mean squared error

`gapminder` models

```
c(mean(residuals(gm_simple)^2),
mean(residuals(gm_complex)^2))
```

```
## [1] 54.47218 41.08507
```

`candy_rankings` models

```
c(mean(residuals(candy_simple)^2),
mean(residuals(candy_complex)^2))
```

```
## [1] 188.4498 127.1098
```

A victory for machine learning!

... or is it? Find out in our first seminar