ICTR Institute for Clinical and Translational Research AT EINSTEIN AND MONTEFIORE
Building Bridges in the Bronx and Beyond

DATA SCIENCE INSTITUTE AT EINSTEIN

DS 101 lecture series: Machine Learning

Melissa J Fazzari, PhD

EINSTEIN Albert Einstein College of Medicine

Montefiore

Outline of today's talk

1. An example
2. Logistic regression
3. Penalized logistic regression
4. Neural networks
5. Tree-based classifiers
6. Random Forest
7. Super Learners

Focus is on classifiers for binary outcomes

What we will not have time to discuss today

1. Comparing classifiers
   - Discrimination
   - Calibration

2. XAI
   - Permutation-based approaches
   - Shap values
   - LIME

3. Cross validation

4. Specific coding

Classification example

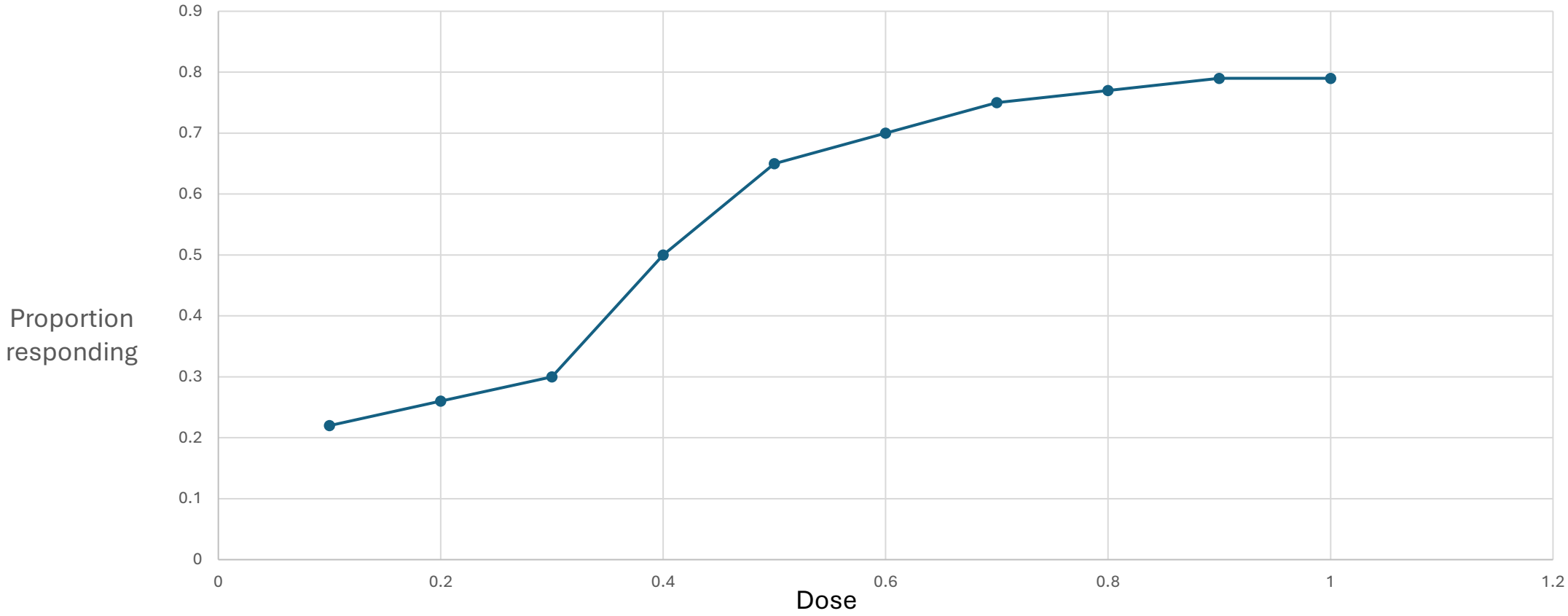
Suppose we want to predict patient response based on the treatment dose received

- **Treatment response is binary** – each patient in the data set either responds or does not respond

- **Treatment dose is continuous** – for simplicity we will assume that dose can be anywhere between 0 (no treatment) and 1 (full treatment).

We treat 100 patients at each dose level (.1,.2,.3,.4,.5,.6,.7,.8,.9,1.0), which gives us a nice sample size to estimate the proportion of responders at each dose level

Predicting treatment response



Probability at the individual level

- Suppose we now want to **estimate the true dose-response function**
- What we observed may be close, but we want some sort of model to describe it
- We want to say **“if your dose is X then the probability of responding to treatment is Z”**

  This will allow us to build a prediction tool for future patients
- In practice, we would also include other predictors such as age, sex, and body weight in the model to get even more precise, tailored predictions

Method 1: Logistic regression (LR)

- Called ***logistic*** regression because we regress the ***log odds*** of the probability of **response (p)** onto a linear function of **dose**

Recall, the odds of response=0.25/(1-0.25) = 0.33

and log odds = ln(0.333) = -1.11

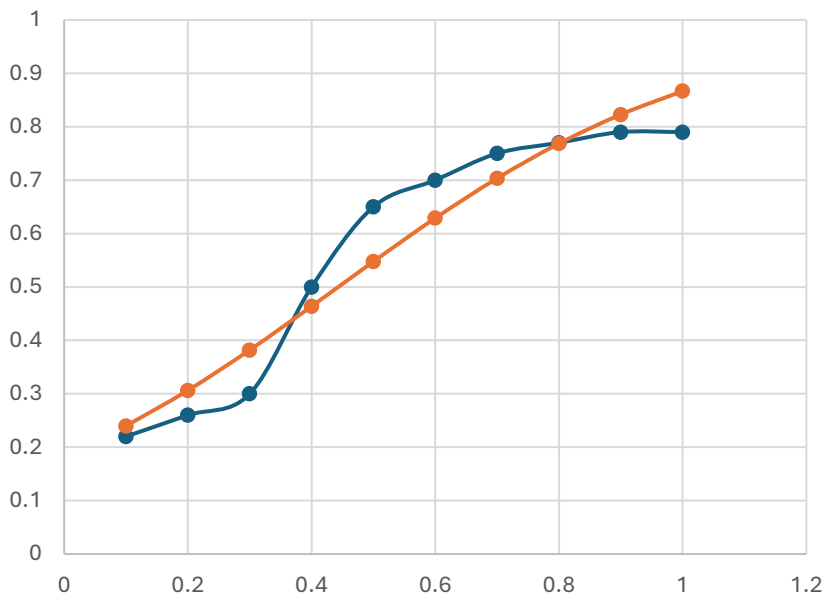
$$\textbf{logit(p)} = \log(p/(1-p)) = \beta_0 + \beta_1 \times \textbf{dose}$$

β_1 controls how much the response probability changes as dose changes

Because we are modeling the log odds of p as a linear function, we are assuming that the relationship between dose and p is nonlinear

Method 1: Logistic regression

Once we fit the logistic regression model, we can *predict* the probability of response at each dose level



- $\widehat{logit(p)} = -1.4897 + 3.3618\ x\ dose$

Intercept slope

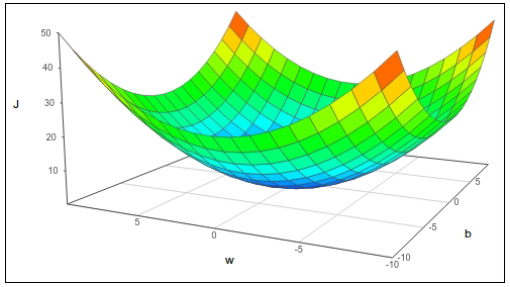
Transform to get predicted probabilities:

- $\hat{p} = exp(\widehat{logit(p)})/(1 + exp(\widehat{logit(p)}))$

When dose=0.1 (smaller dose), predicted response probability=24%

When dose=1 (full treatment), predicted response probability =86%

How do we find the best model?



- In our model, we have two parameters that must be estimated – the intercept and slope
- There are an infinite number of possibilities for this model – so how do we find the best combination for our data?
- Answer: we find the combination that minimizes the ***cost***
- **This cost function rewards highly confident, correct predictions. It penalizes low confidence or incorrect predictions**

How do we find the best LR model?

$$COST_{logistic} = -\frac{1}{n}\sum[ylog(\hat{\boldsymbol{p}}) + (1-y)\log(1-\hat{\boldsymbol{p}})]$$

Patient 12 responded (Y=1), but their predicted response probability assigned by the model was 0.22

$\text{COST}_{\text{patient 12}} = [1\ x\ \log(0.22) + (1-1)\log(0.78)] = 1.51$

Patient 25 did <u>not</u> respond (Y=0), and their predicted response probability assigned by the model was 0.22

$\text{COST}_{\text{patient 25}} = [0\ x\ \log(0.22) + (1-0)\log(0.78)] = 0.248$

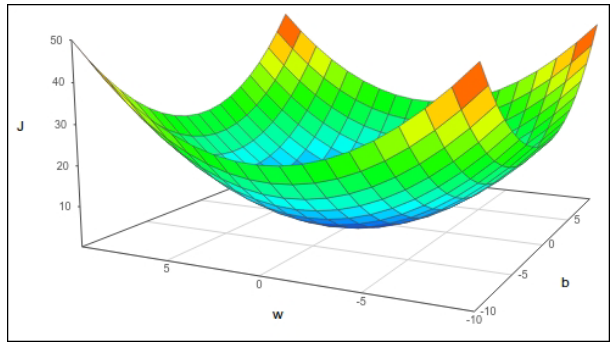
Patient 33 did <u>not</u> respond (Y=0), and their predicted response probability assigned by the model was 0.49

$\text{COST}_{\text{patient 33}} = [0\ x\ \log(0.50) + (1-0)\log(0.49)] = 0.713$

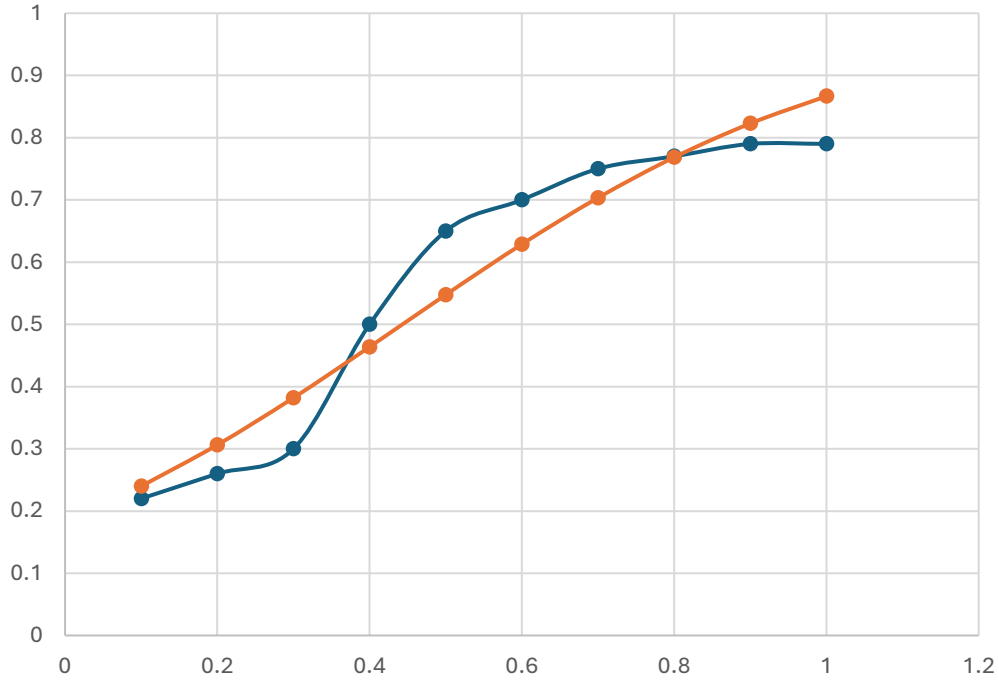
How do we find the best LR model?

$$COST_{logistic} = -\frac{1}{n}\sum[ylog(\hat{\boldsymbol{p}}) + (1-y)\log(1-\hat{\boldsymbol{p}})]$$

Intercept	slope	Total cost
-2.10	2.0	0.799
-1.55	5.31	0.678
-0.94	2.33	0.596
-1.4897	**3.3618**	**0.588**



Logistic regression



Predicted from model
Observed proportion

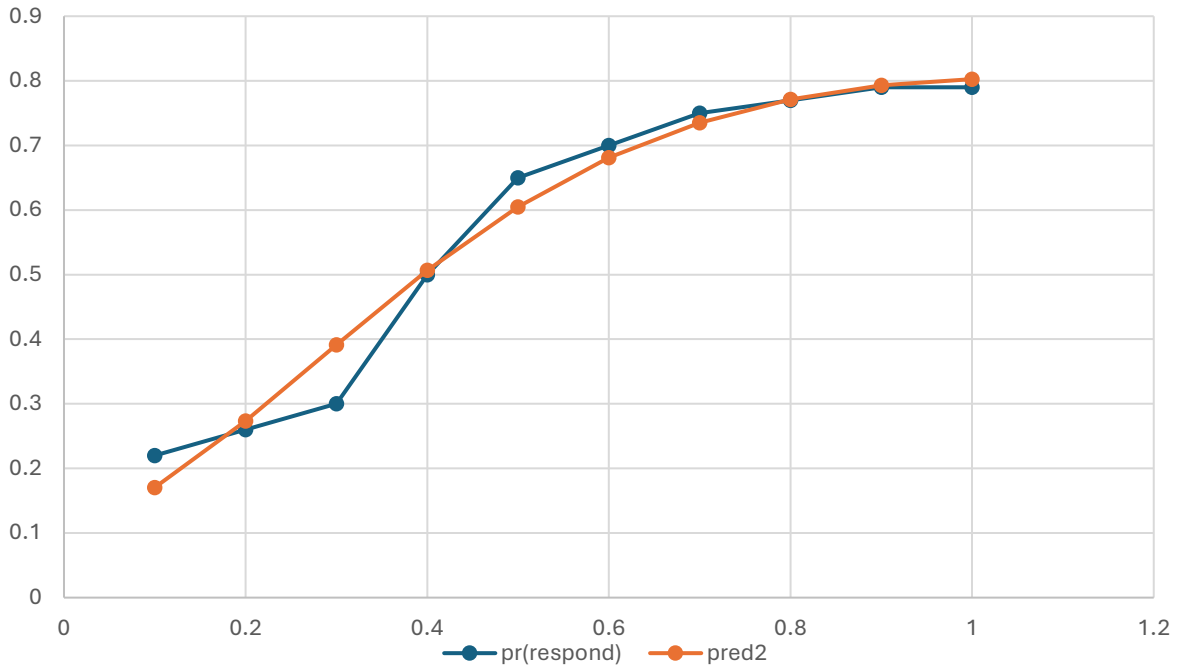
This simple linear function of dose does not seem flexible enough to capture the observed curve.

We can't get the steepness of the middle part of the curve plus the flatness of the extremes…

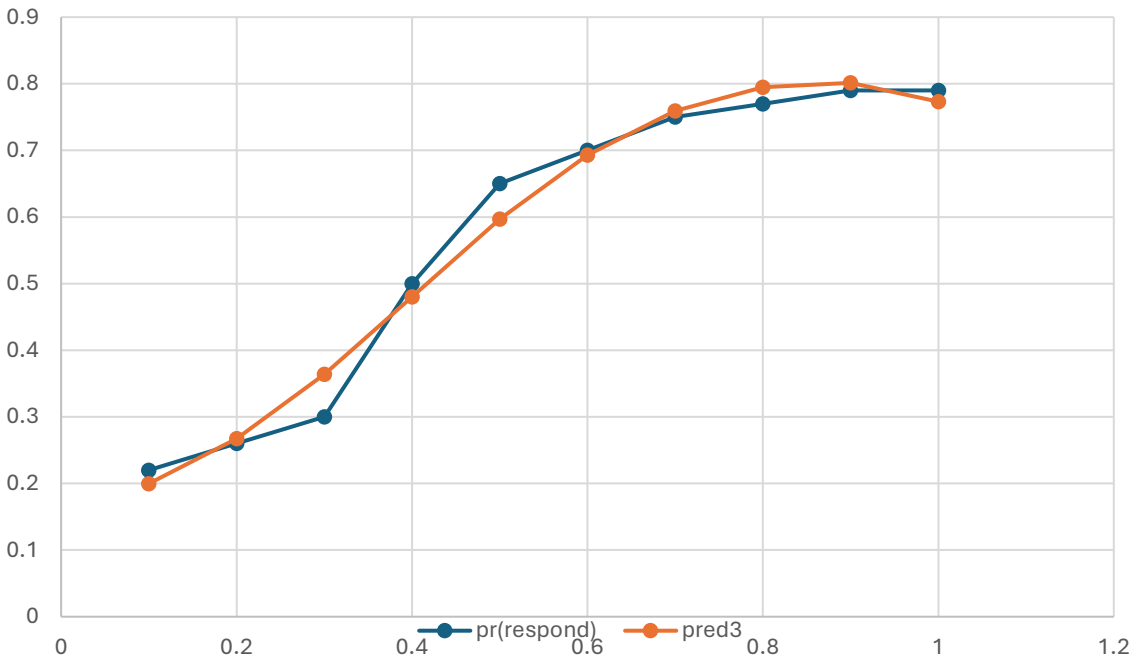
Logistic regression

Now suppose we add some complexity and fit two new models:

$$\widehat{logit}(p) = -2.2546 + 7.0657\ x\ \boldsymbol{dose} - 3.4083\ x\ \boldsymbol{dose^2}$$



$$\widehat{logit}(p) = -1.6573 + 2.0176\ x\ \boldsymbol{dose} - 7.4567\ x\ \boldsymbol{dose^2} - 6.5893\ x\ \boldsymbol{dose^3}$$



Overfitting

- The question we face here – **does the model with dose, dose^2 and dose^3 reflect biological reality** and will thus generalize well to other data sets, **or are we just modeling noise in our data set?**

- Overfitting the data occurs when your model is fitting the nuances of your training dataset, rather than the true function

- An overfit model will fit the original data set very closely, but then not fit any new data sets as well

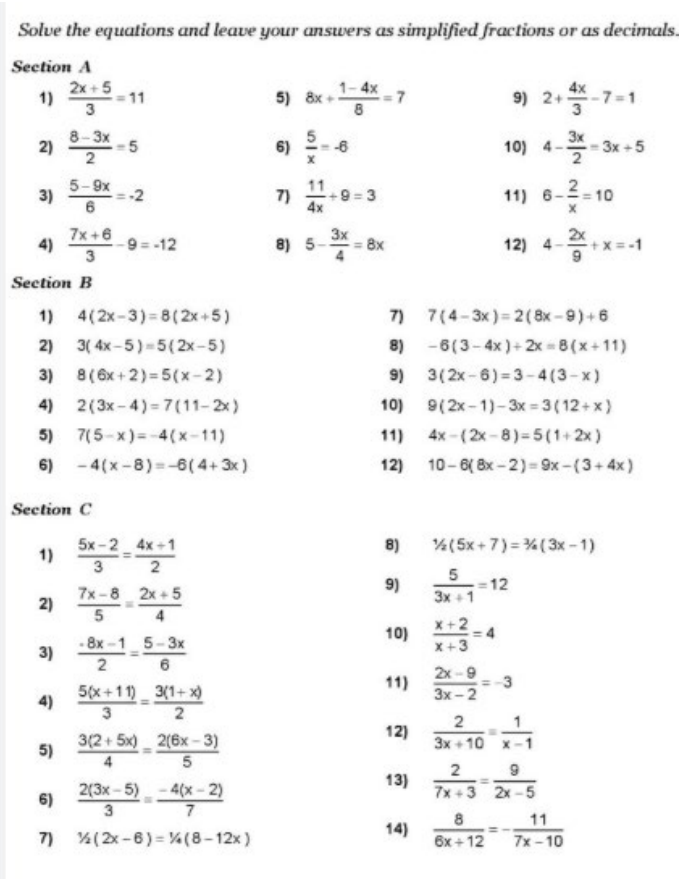
A simple example of overfitting

Suppose 8th grade you had advance access to your math quiz.....

You could study the answers to those exact questions to get 100% correct

But what if your teacher was on to you and swapped the test? How much do you think your very specific knowledge of the original exam will generalize to her new test questions?

You have overfit your studying to one specific exam!



Solve the equations and leave your answers as simplified fractions or as decimals.

Section A

1) $\frac{2x+5}{3}=11$
2) $\frac{8-3x}{2}=5$
3) $\frac{5-9x}{6}=-2$
4) $\frac{7x+6}{3}-9=-12$
5) $8x+\frac{1-4x}{8}=7$
6) $\frac{5}{x}=-6$
7) $\frac{11}{4x}+9=3$
8) $5-\frac{3x}{4}=8x$
9) $2+\frac{4x}{3}-7=1$
10) $4-\frac{3x}{2}=3x+5$
11) $6-\frac{2}{x}=10$
12) $4-\frac{2x}{9}+x=-1$

Section B

1) $4(2x-3)=8(2x+5)$
2) $3(4x-5)=5(2x-5)$
3) $8(6x+2)=5(x-2)$
4) $2(3x-4)=7(11-2x)$
5) $7(5-x)=-4(x-11)$
6) $-4(x-8)=-6(4+3x)$
7) $7(4-3x)=2(8x-9)+6$
8) $-6(3-4x)+2x=8(x+11)$
9) $3(2x-6)=3-4(3-x)$
10) $9(2x-1)-3x=3(12+x)$
11) $4x-(2x-8)=5(1+2x)$
12) $10-6(8x-2)=9x-(3+4x)$

Section C

1) $\frac{5x-2}{3}=\frac{4x+1}{2}$
2) $\frac{7x-8}{5}=\frac{2x+5}{4}$
3) $\frac{-8x-1}{2}=\frac{5-3x}{6}$
4) $\frac{5(x+11)}{3}=\frac{3(1+x)}{2}$
5) $\frac{3(2+5x)}{4}=\frac{2(6x-3)}{5}$
6) $\frac{2(3x-5)}{3}=\frac{-4(x-2)}{7}$
7) $\frac{1}{2}(2x-6)=\frac{1}{4}(8-12x)$
8) $\frac{1}{2}(5x+7)=\frac{3}{4}(3x-1)$
9) $\frac{5}{3x+1}=12$
10) $\frac{x+2}{x+3}=4$
11) $\frac{2x-9}{3x-2}=-3$
12) $\frac{2}{3x+10}=\frac{1}{x-1}$
13) $\frac{2}{7x+3}=\frac{9}{2x-5}$
14) $\frac{8}{6x+12}=-\frac{11}{7x-10}$

Method 2: Penalized logistic regression: LASSO

- A method to control for overfitting

- Minimizes the following cost function:

$$COST_{penalized} = COST_{logistic} + \lambda|\sum\beta|$$

- While this cost function penalizes errors in the same way as a regular logistic regression, it also prevents the slope coefficients in the model from getting too large in magnitude (large coefficients now add to the cost)

- λ reflects how much we want to penalize coefficients vs rely on the original cost to guide model fitting.

Note: λ is selected through computational methods (internal cross validation).

Cost functions

- We have now seen that we can switch from a **regular logistic regression** to a **penalized logistic regression** just by changing the cost function slightly

  Why do we care about how large our coefficients get?

- Because minimizing cost rewards extreme separation, the model can overfit - assigning very large coefficients to achieve nearly perfect predictions on the training data

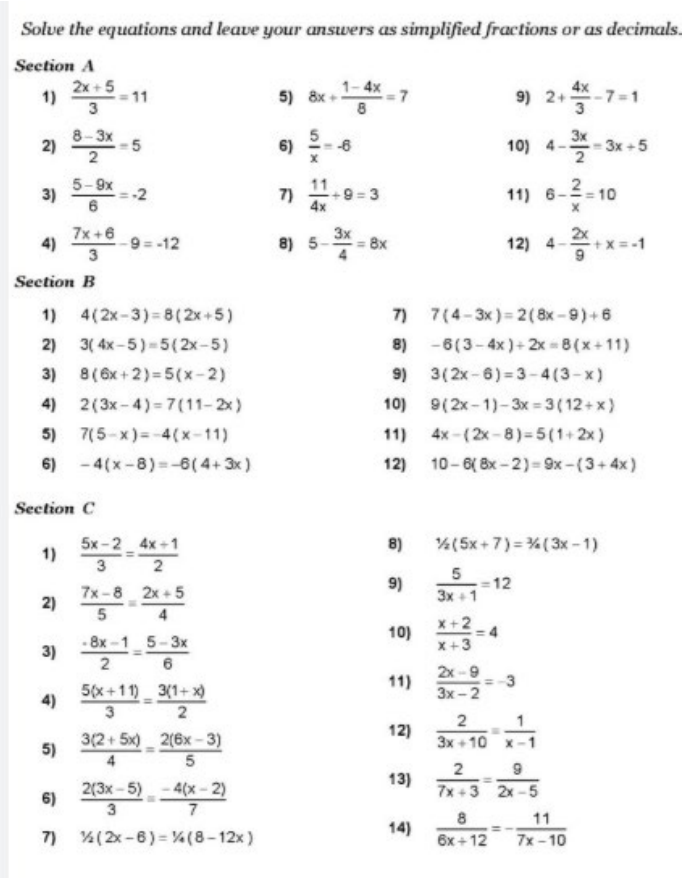
- Penalized LR like LASSO can reduce overfitting by setting some coefficients equal to 0 or shrinking the coefficient so that it is not as influential in generating model predictions

A simple example

Back to your cheating….

- Now suppose we forced 8th grade you (who has the exam) to only use the exam to cram the main concepts being asked. You couldn't copy down specific examples

- This is acting **just like a penalized regression** – *you are focusing more on the concepts that will be generalizable and ignoring exam specifics that probably won't appear on any new tests.*



Solve the equations and leave your answers as simplified fractions or as decimals.

Section A

1) $\frac{2x+5}{3}=11$
2) $\frac{8-3x}{2}=5$
3) $\frac{5-9x}{6}=-2$
4) $\frac{7x+6}{3}-9=-12$
5) $8x+\frac{1-4x}{8}=7$
6) $\frac{5}{x}=-6$
7) $\frac{11}{4x}+9=3$
8) $5-\frac{3x}{4}=8x$
9) $2+\frac{4x}{3}-7=1$
10) $4-\frac{3x}{2}=3x+5$
11) $6-\frac{2}{x}=10$
12) $4-\frac{2x}{9}+x=-1$

Section B

1) $4(2x-3)=8(2x+5)$
2) $3(4x-5)=5(2x-5)$
3) $8(6x+2)=5(x-2)$
4) $2(3x-4)=7(11-2x)$
5) $7(5-x)=-4(x-11)$
6) $-4(x-8)=-6(4+3x)$
7) $7(4-3x)=2(8x-9)+6$
8) $-6(3-4x)+2x=8(x+11)$
9) $3(2x-6)=3-4(3-x)$
10) $9(2x-1)-3x=3(12+x)$
11) $4x-(2x-8)=5(1+2x)$
12) $10-6(8x-2)=9x-(3+4x)$

Section C

1) $\frac{5x-2}{3}=\frac{4x+1}{2}$
2) $\frac{7x-8}{5}=\frac{2x+5}{4}$
3) $\frac{-8x-1}{2}=\frac{5-3x}{6}$
4) $\frac{5(x+11)}{3}=\frac{3(1+x)}{2}$
5) $\frac{3(2+5x)}{4}=\frac{2(6x-3)}{5}$
6) $\frac{2(3x-5)}{3}=\frac{-4(x-2)}{7}$
7) $\frac{1}{2}(2x-6)=\frac{1}{4}(8-12x)$
8) $\frac{1}{2}(5x+7)=\frac{3}{4}(3x-1)$
9) $\frac{5}{3x+1}=12$
10) $\frac{x+2}{x+3}=4$
11) $\frac{2x-9}{3x-2}=-3$
12) $\frac{2}{3x+10}=\frac{1}{x-1}$
13) $\frac{2}{7x+3}=\frac{9}{2x-5}$
14) $\frac{8}{6x+12}=-\frac{11}{7x-10}$

Penalized logistic regression

Original model:

$$\widehat{\boldsymbol{logit}}(\boldsymbol{p}) = -1.6573 + 2.0176\ x\ \boldsymbol{dose} - 7.4567\ x\ \boldsymbol{dose^2} - 6.5893\ x\ \boldsymbol{dose^3}$$

Penalized model:

$$\widehat{\boldsymbol{logit}}(\boldsymbol{p}) = -1.523 + 4.966\ x\ \boldsymbol{dose} - 3.899\ x\ \boldsymbol{dose^2} - 0.051\ x\ \boldsymbol{dose^3}$$

Logistic regression -> neural networks

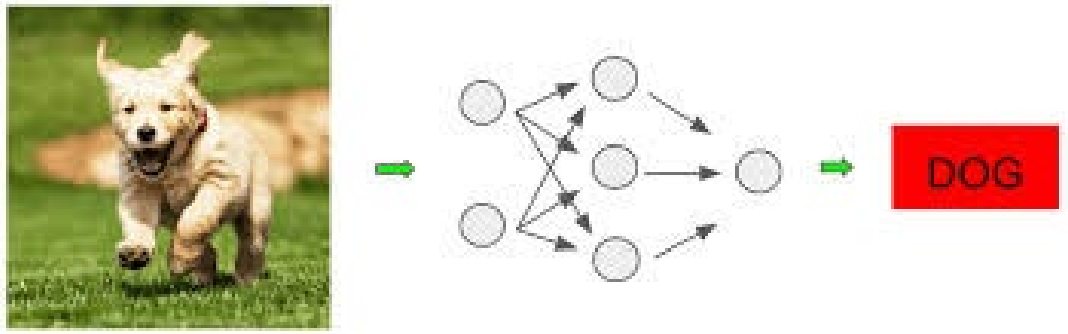
- We already saw that the standard logistic regression model did not fit the data as well as we would like

- We added some complexity through feature engineering, which helped the model predictions get closer to the true proportions of responders observed at each dose level

- However sometimes we do not know what kind of complexity to add. The true function may be a highly complex function of the model inputs

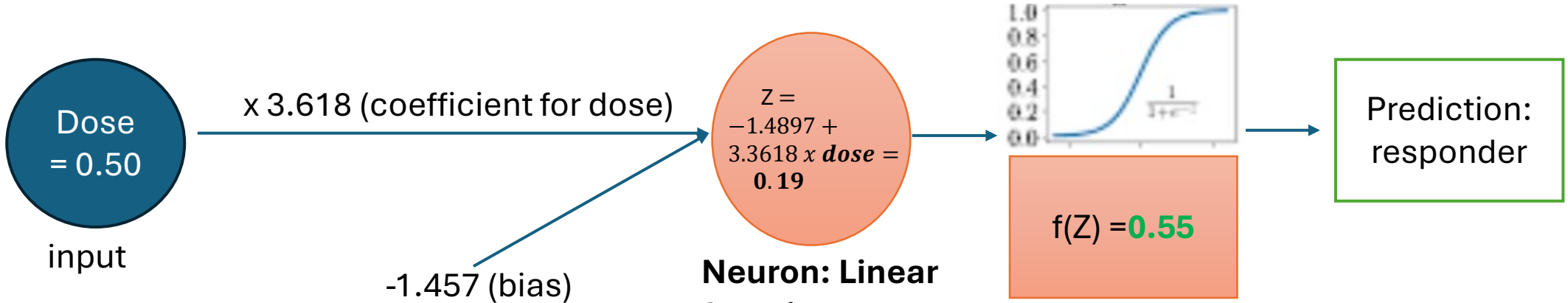
Method 3: Neural Networks - *inspired* by the human brain

- We receive input (say a photo of a dog), certain neurons will fire up and send signals to the second layer of neurons, which then send signals to a third layer of neurons… and then finally output an answer: “a dog”



A Feed-Forward Neural Network (NN) = Logistic Regression

Let's use our dose/response example in a NN framework:



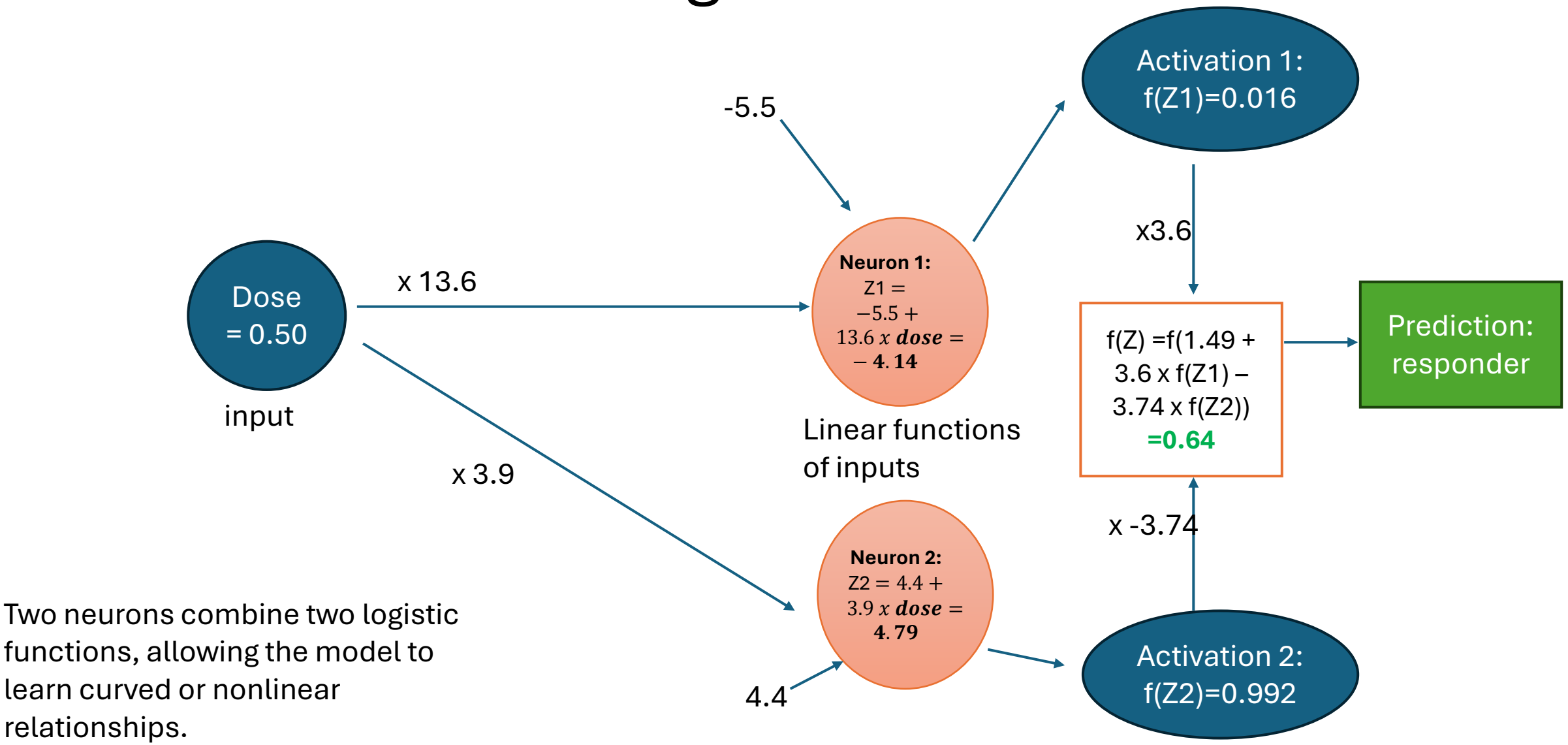
A Feed-Forward Neural Network (NN) = Logistic Regression

- We saw that a single-neuron neural network is mathematically identical to logistic regression.

- So why do we care about neural networks?

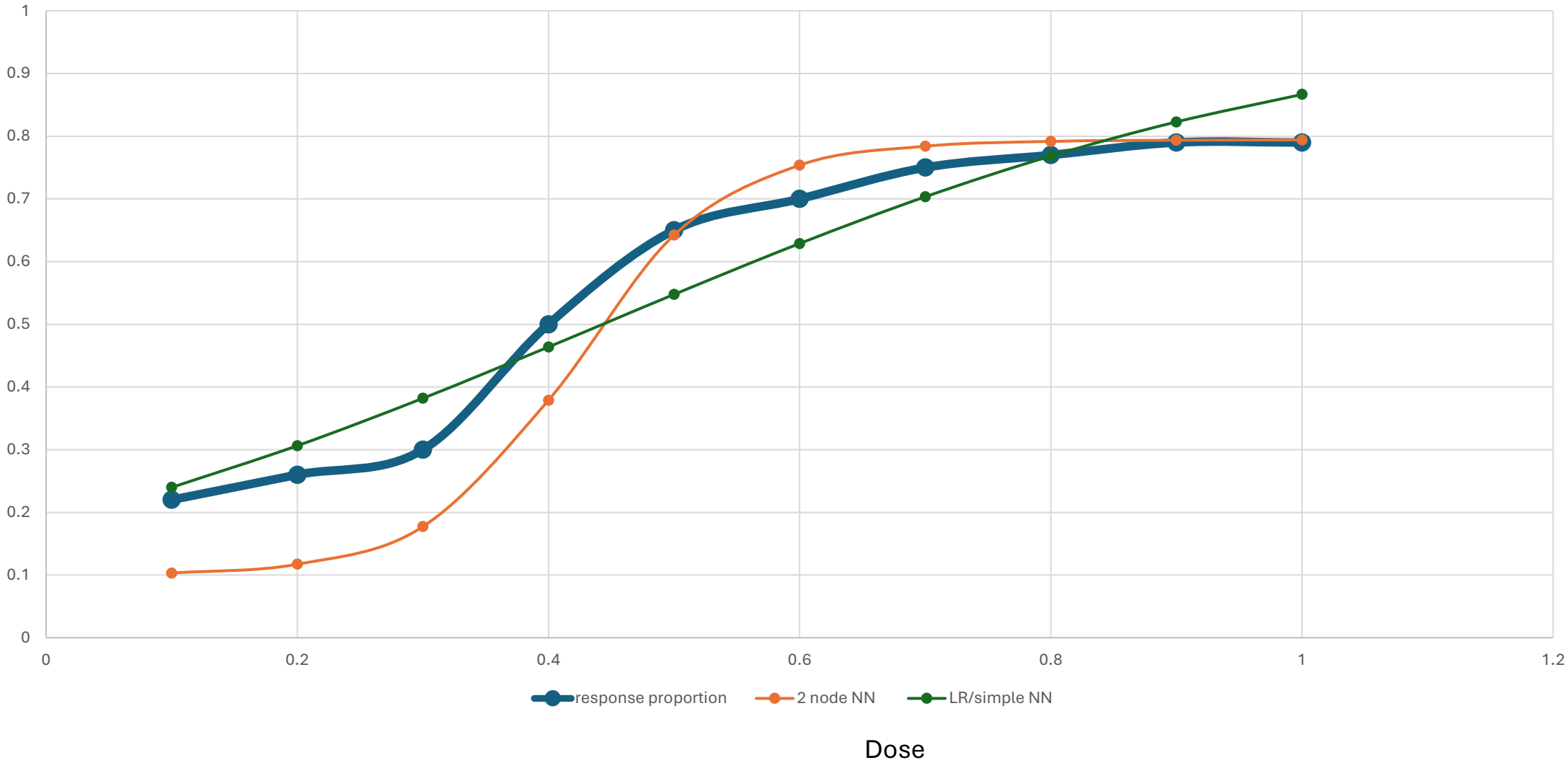
- *Adding more neurons and layers lets the model learn much more complex patterns….*

A 2-node Feed-Forward Neural Network (NN) = A mixture of two logistic curves



Two neurons combine two logistic functions, allowing the model to learn curved or nonlinear relationships.

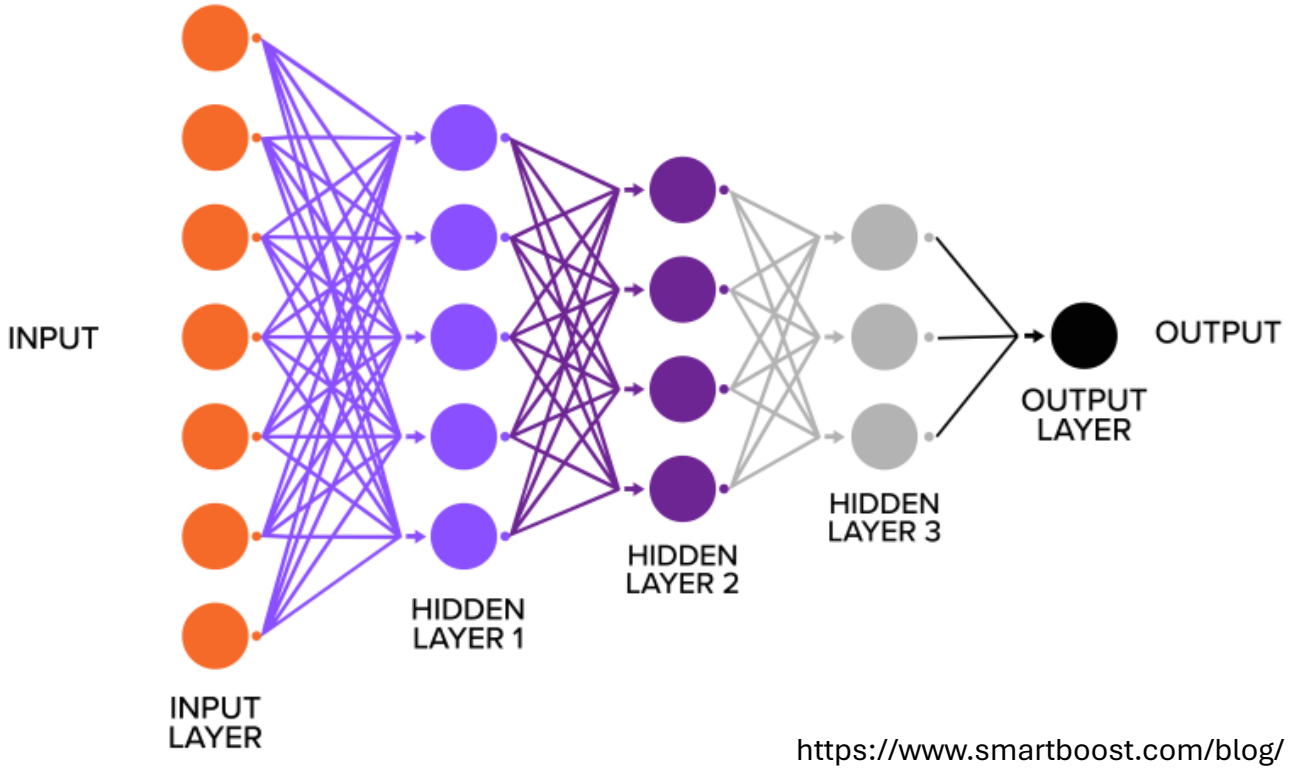
A 2-node Feed-Forward Neural Network (NN)



Deep Learning: Neural Networks

Deep learning = more layers and nodes

Within each layer, models are combined and then passed to the next layer as **new features** to be weighted



https://www.smartboost.com/blog/

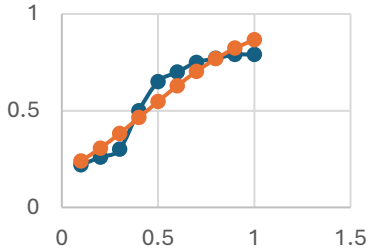
What are the benefits of deep learning?

- **Increased complexity**
- **Improved model flexibility**
- **Feature extraction**

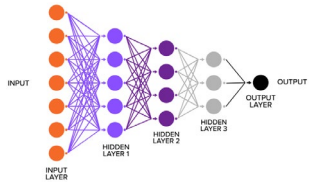
Recap 1

We started with a **single logistic regression then we:**

- Saw how a logistic regression model is estimated through **minimizing a cost function**
- Examined how to **add complexity** to improve model fit



- Learned how adding a **penalty** to the logistic cost function reduces potential overfitting
- Saw that a **feed-forward neural network** with one node/one hidden layer using a logistic activation function is **the same as a simple logistic regression**
- Explored how **neural networks can stack a few logistic regression models** to automatically gain flexibility



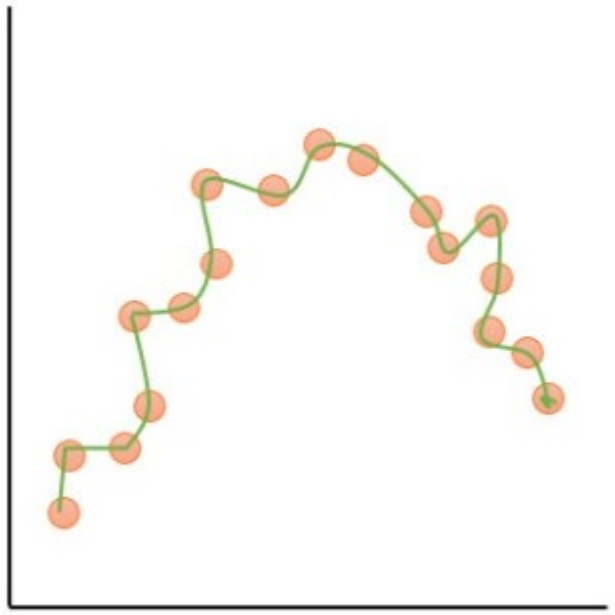
Recap 1

- We also discussed **overfitting** – when you fit a model that is too complex and too close to the data at hand (essentially fitting noise)
- Another related definition: a **high variance** classifier
- High variance classifiers are those classifiers that tend to produce very different predictions when trained on slightly different samples from the same population
- **High variance is the mechanism that leads to overfitting** - the model ends up learning nuances of the training sample rather than the underlying data-generating process (the true underlying function).

Recap 1

Which classifiers have the capacity to be high-variance?

- Logistic regression?
- Penalized logistic regression?
- Neural networks?

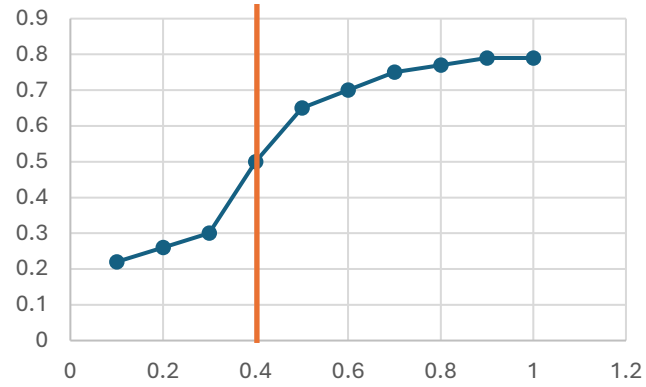


Tree-based classifiers

- **Decision trees** are a type of classifier that aims to separate different outcomes (for example, patients who respond to treatment vs. those who do not respond).

- The algorithm starts with all 1000 observations and then searches for the best feature and splitting value.

Decision Trees: First split



1000 patients
Proportion responding: 57%

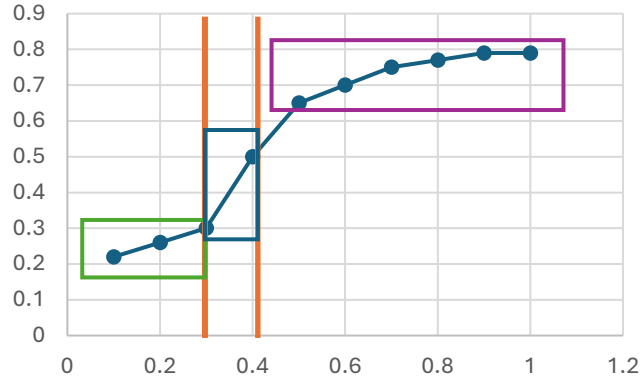
Dose ≤ 0.40

Dose > 0.40

400 patients
Proportion responding: 32%

600 patients
Proportion responding: 74%

Decision Trees: Second split



1000 patients
Proportion
responding: 57%

Dose ≤ 0.40

Dose > 0.40

400 patients
Proportion
responding: 32%

600 patients
Proportion
responding: 74%

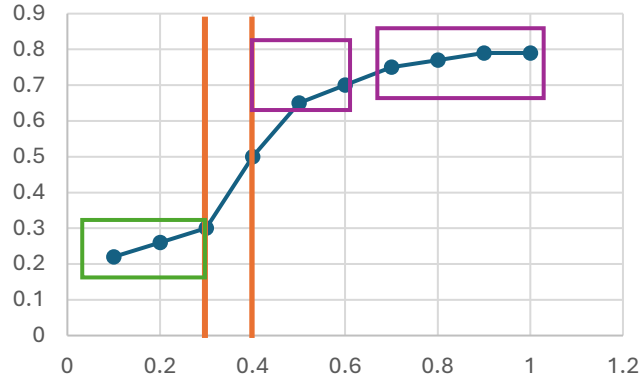
Dose ≤ 0.30

Dose > 0.30

300 patients
Proportion
responding: 26%

100 patients
Proportion
responding: 50%

Decision Trees: Third split



1000 patients
Proportion
responding: 57%

Dose ≤ 0.40

Dose > 0.40

400 patients
Proportion
responding: 32%

600 patients
Proportion
responding: 74%

Dose ≤ 0.30

Dose > 0.30

Dose ≤ 0.60

Dose > 0.60

300 patients
Proportion
responding: 26%

100 patients
Proportion
responding: 50%

200 patients
Proportion
responding: 68%

400 patients
Proportion
responding: 78%

Decision Trees

By breaking up the data into smaller and smaller nodes, decision trees are flexible enough to model curves and other complex relationships like interactions

Decision Trees – how do these splits happen?

- The best split maximizes **purity** - getting responders and non-responders into separate nodes

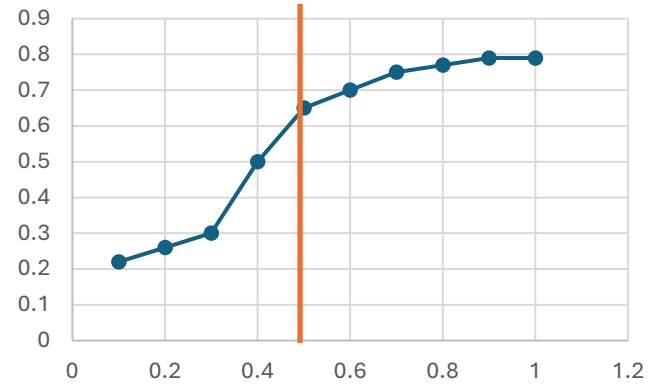
- There are many ways to compute purity, but for all of them the idea is to get as many responders in one node, and as many non-responders in the other node.

- We will focus on one purity measure called the **Gini index**. The Gini index can have a minimum value of 0 (pure) and a maximum value of 0.5 (not pure at all).

- Gini index$_{\text{node k}}$:

  1 - prop(responders in node k)^2 - prop(responders in node k)^2

Decision Trees: First split



Gini =1-(.6)^2 – (.4)^2 =0.48 (close to max impurity)

1000 patients
Proportion
responding: 60%

Dose ≤ 0.40

Dose > 0.40

500 patients
Proportion
responding: 38%

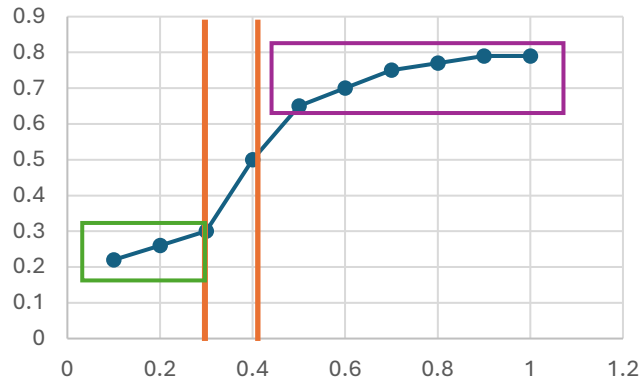
500 patients
Proportion
responding: 76%

Gini = 1-(.62)^2 – (.38)^2 = 0.435

Gini = 1-(.76)^2 – (.24)^2 =0.383

Weighted Gini = 0.5 x .435 + 0.5 x 0.383 = 0.404 – an improvement of 0.08

Decision Trees: Second split



1000 patients
Proportion
responding: 57%

Dose ≤ 0.40

Dose > 0.40

400 patients
Proportion
responding: 32%

600 patients
Proportion
responding: 74%

Dose ≤ 0.30

Dose > 0.30

300 patients
Proportion
responding: 26%

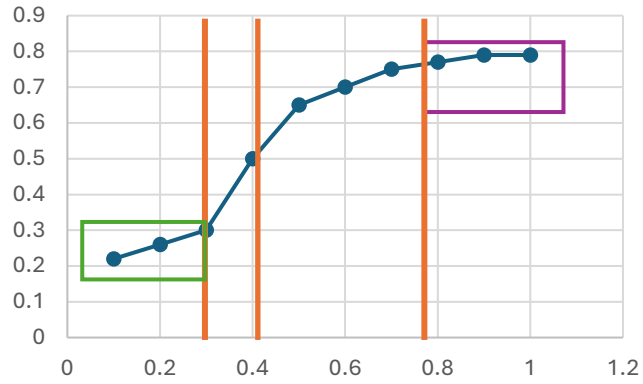
100 patients
Proportion
responding: 50%

Gini = 1-(.26)^2 – (.74)^2 = 0.385

Gini = 1-(.50)^2 – (.50)^2 = 0.50

Weighted Gini = 0.75 x .385 + 0.25 x 0.50 = 0.414 – a reduction of 0.435-0.414=0.021

Decision Trees: Third split



1000 patients
Proportion
responding: 57%

Dose ≤ 0.40

Dose > 0.40

400 patients
Proportion
responding: 32%

600 patients
Proportion
responding: 74%

Dose ≤ 0.30

Dose > 0.30

Dose > 0.70

Dose > 0.70

300 patients
Proportion
responding: 26%

100 patients
Proportion
responding: 50%

300 patients
Proportion
responding: 65%

300 patients
Proportion
responding: 78.3%

Gini = 0.42

Gini = 0.34

Weighted Gini = 0.42 x .5 + 0.34 x 0.50 = 0.379 – a decrease of 0.385-0.380 = 0.005

Some notes on decision trees

- Decision trees may be grown and pruned *or* prevented from growing too large **to avoid overfitting** the training data set.

- Recall, **overfitting means we are modeling the nuances of our training data set** (such as creating another split at dose 0.70). Chances are, this will not hold up as a true difference in response probability in future data sets

- The final tree assigns a predicted outcome (here, response yes or no) based on **the response status for the majority of the observations in that node.**

Some notes on decision trees

- Decision trees have the capacity to be high-variance, highly flexible classifiers, which can adapt closely to the training data (**low bias**) and **easily overfit**, capturing noise as if it were signal.

- High variance means that **small changes in the training set can produce very different trees** (different splits, thresholds, and predicted classes), even if the data come from the same population.

- Contrast this with **high bias/low variance classifiers** - for example, a rule predicting response if dose > 0.50 is high bias. This inflexible decision rule ignores most of the complexity in the data in favor of stable predictions across data sets.

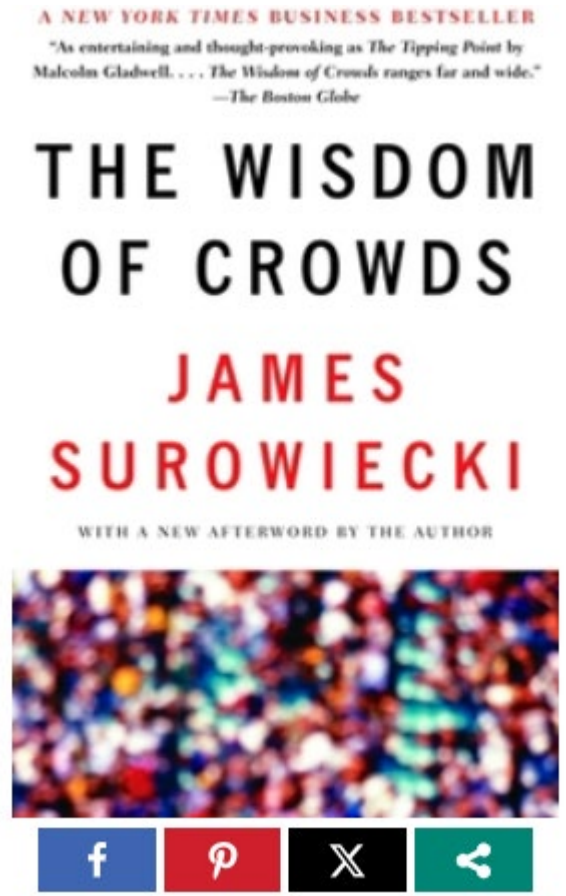
Method 4: Random Forest

- A ***very*** popular off-the-shelf machine learning classifier
- Because fully grown single decision trees tend to be high variance classifiers, the idea is to build an ensemble of many trees (typically 500-1000).
- Each tree sees a slightly different sample of the data and subset of predictors.
- When we average or “vote” across these trees, the result is far more stable and accurate than any one tree.

Principle behind Random Forest: *The Wisdom of Crowds*

- In 1906, British scientist Francis Galton analyzed 787 guesses for the weight of a slaughtered ox at a fair and found the average guess was off by less than 1% of the ox's true weight of 1,198 pounds.

- The collective intelligence of a **diverse group** is often more accurate than any one individual's guess.



Method 4: Random Forest

- Of course, if we repeatedly ran the same classification tree algorithm on our data set, we would end up with 1000 of the same trees…..

- To get new, **diverse** trees, Random Forest does two main things:

    1. It perturbates the sample through bootstrapping
    2. It randomly selects a subset of the inputs at each potential split

Random Forest – bootstrapping part

- Bootstrapping data means that we sample from our training data set with replacement
- If our data set looked like this:

Response	Dose
1	.5
0	.4
0	.1
1	.9

- Then one bootstrapped sample could look like this:

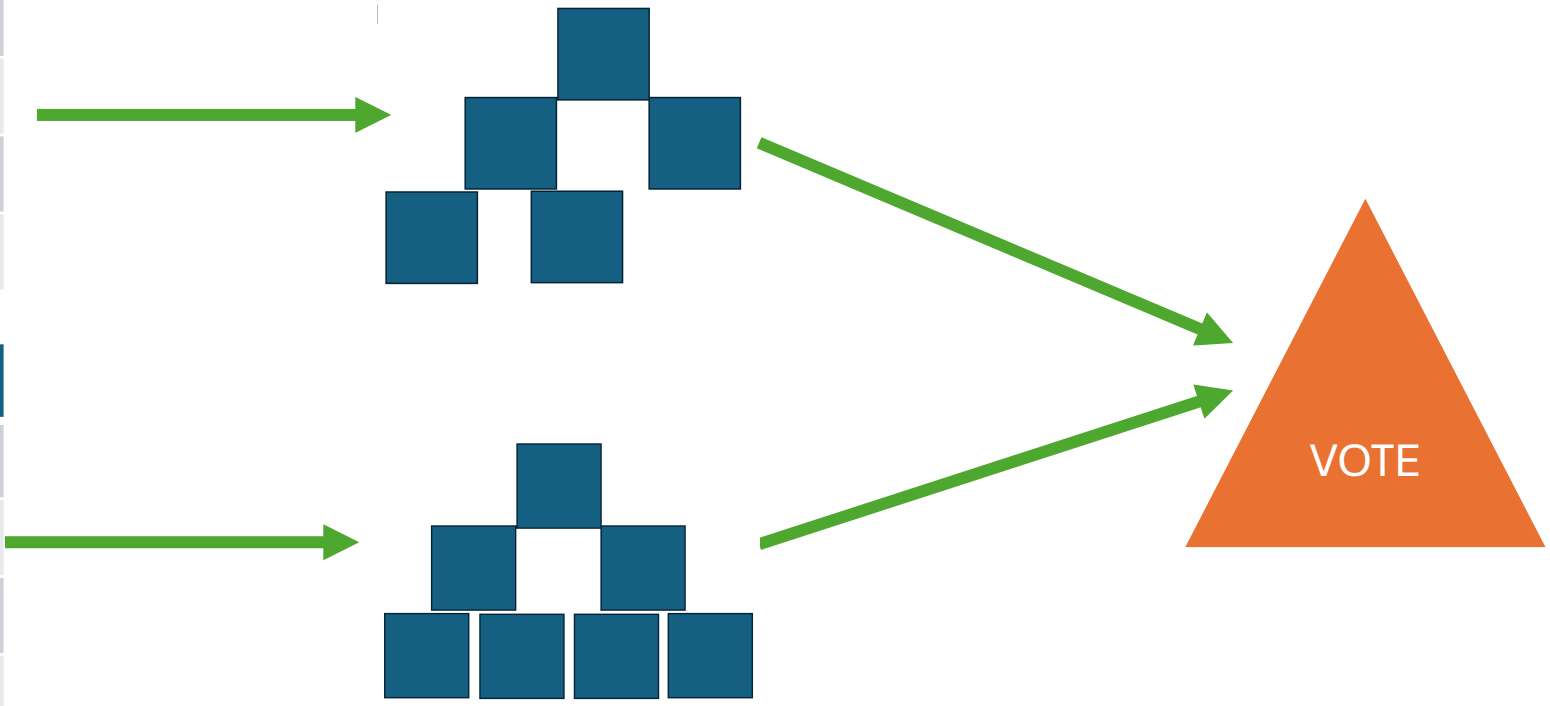
Response	Dose
1	.5
1	.5
0	.1
0	.1

Random Forest

We then fit a fully grown classification tree to each bootstrapped sample:

Response	Dose
1	.5
0	.4
0	.1
1	.9

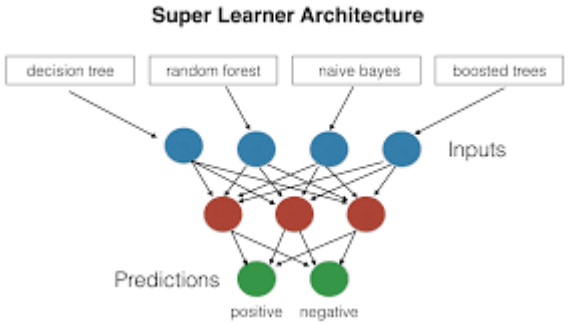
Response	Dose
1	.5
1	.5
0	.1
0	.1



Recap 2

- We started by looking at logistic regression and its variants
- We then showed how to **add flexibility with a neural network**, which can learn complex relationships automatically
- We saw that these ***probabilistic*** methods work by **minimizing a cost function**, a penalty to the model for producing predictions that were far from the truth
- We discussed ***overfitting*** and how this can happen
- We then pivoted to algorithmic tree-based classifiers and used the **Gini index** to get tree nodes that are increasingly “pure”
- Finally, we explored how **Random Forest** combats overfitting by averaging many overfit, large trees into a stable, low-variance ensemble

Super Learner – ensemble learning



- We saw that Random Forest was an ensemble of decision trees
- Using the same idea, we can fit **an ensemble of different types of classifiers**
- The benefit of this is that you can incorporate the strengths of each type of classifier (regression vs. tree-based vs. instance-based vs. domain knowledge based)
- You can also vary hyper-parameters within a class of models – such as decision tree stumps vs fully grown trees

Super Learner: basic idea

- Each learner is trained and their “best” predictions generated
- These predictions are then used as new features to train a second-level model, known as the meta-learner, which learns how to best combine the first-level predictions
- The meta-learner determines the optimal weights for each base learner's prediction to produce the final output. **Models that perform better in validation are given higher weights**
- The Super Learner model's final prediction is a weighted average of the predictions from the base learners, with the weights determined by the meta-learner.

Super Learner

- **Reduces model selection uncertainty:** Since it's difficult to know in advance which model will perform best for a specific problem, the Super Learner method avoids the need to choose just one by combining many possibilities.

- **Improves predictive accuracy:** By creating an optimal weighted average, it leverages the strengths of multiple models and often results in more accurate predictions than any single model could achieve alone.

Machine Learning Goal

- **The goal is <u>not</u> complexity, it's generalization**
- Whatever **patterns we learn we want to hold up in future patient groups**
- Machine Learning can give us much more complexity above standard statistical models, **however the ability to generalize is the most important goal**

BERD House

https://einsteinmed.edu/centers/ictr/biostatistics-epidemiology-research-design-core/berd-house/