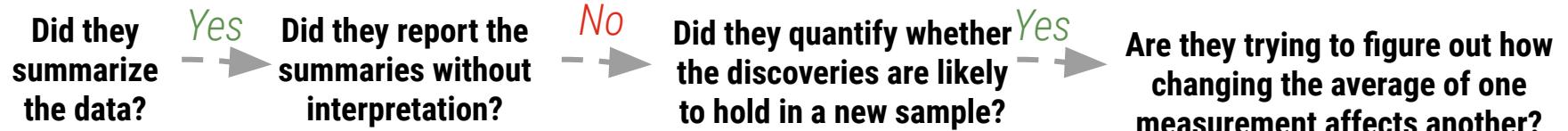


Machine Learning

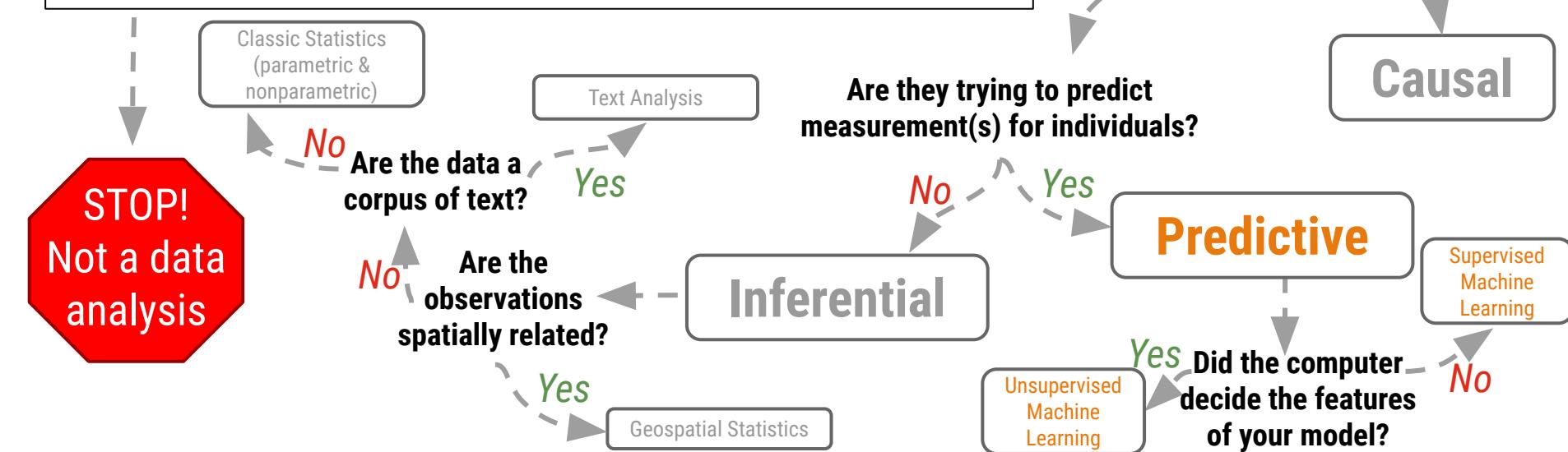
Shannon E. Ellis, Ph.D
UC San Diego

• • •

Department of Cognitive Science
sellis@ucsd.edu



Predictive: apply machine learning techniques to data you have currently to generate a model that will be able to make a prediction on future data

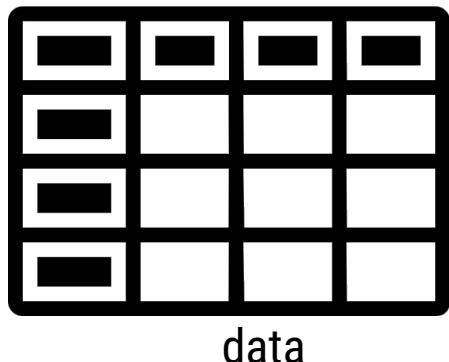


- **Problem:** Detecting whether credit card charges are fraudulent.
- **Data science question:** Can we use the time of the charge, the location of the charge, and the price of the charge to predict whether that charge is fraudulent or not?
- **Type of analysis:** Predictive analysis



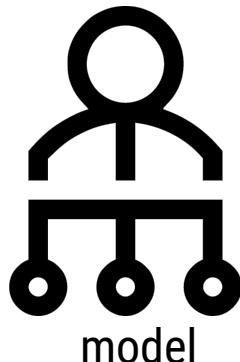
predictive analysis uses data
you have now to make
predictions in the future

machine learning
approaches are used for
predictive analysis!



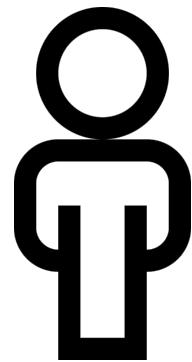
data

train →



model

predict →



What is machine learning?

“Machine learning is the science of getting computers to act without being explicitly programmed”

- Andrew Ng, Stanford, ex-Google, chief scientist at Baidu, Coursera founder, Stanford Adjunct Faculty



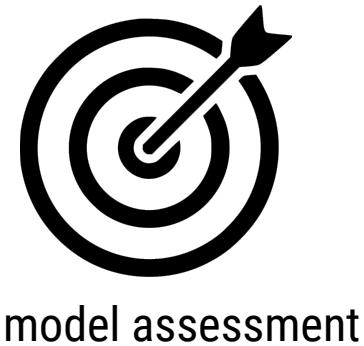
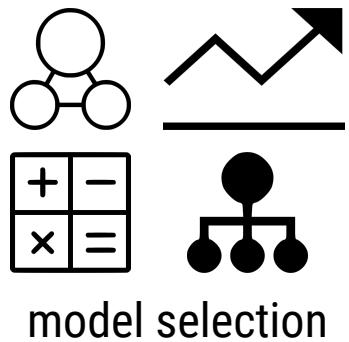
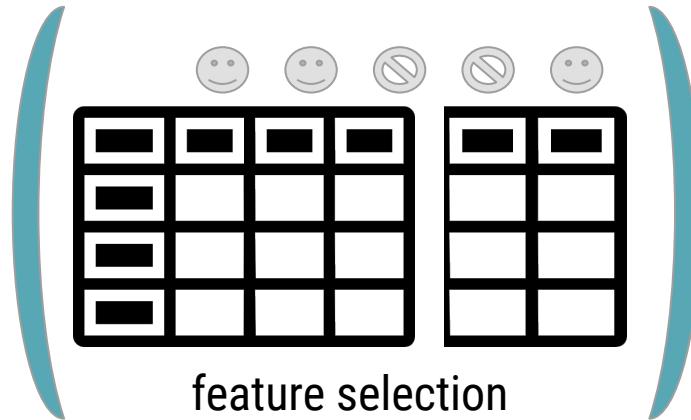
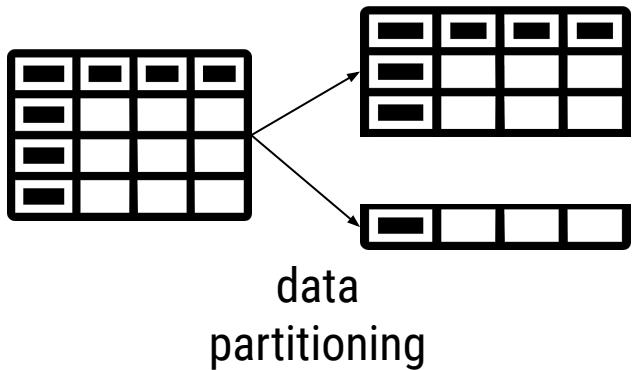
Prediction Questions

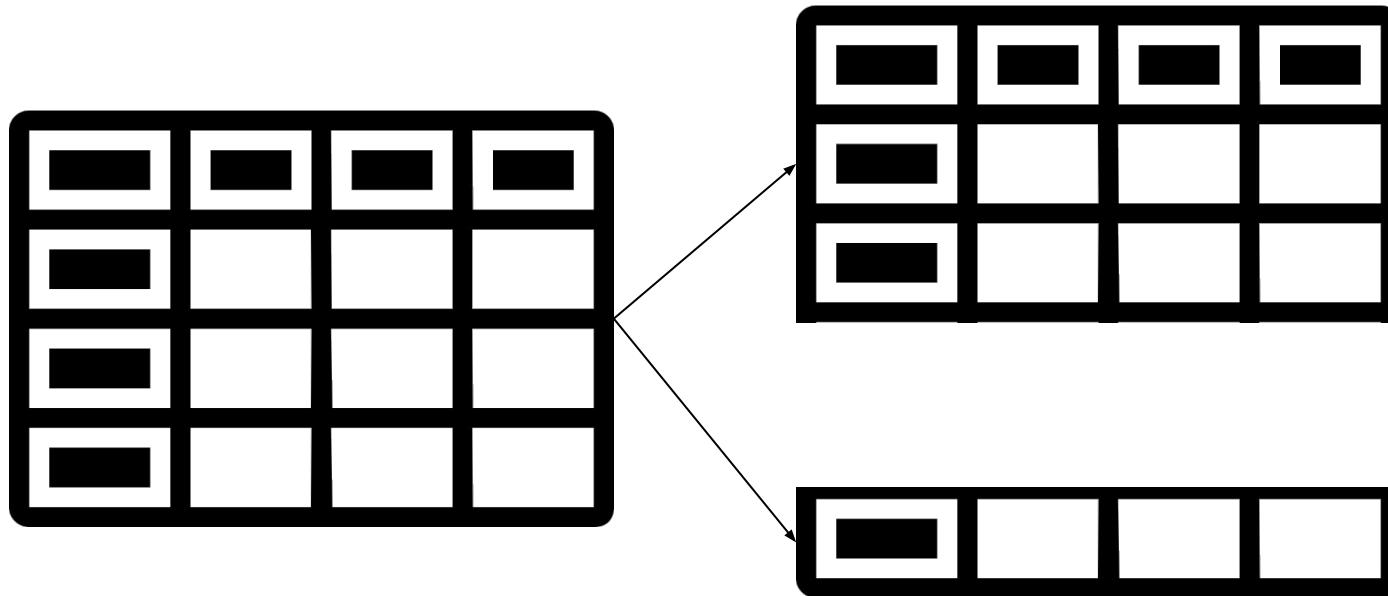
Which of these questions is most appropriate for machine learning?

- A How common is watching Sesame Street in the US?
- B What is the effect of watching Sesame Street on children's brains?
- C What is the relationship between early childhood educational programming and success in elementary school?
- D Can we use information about one's early childhood to predict their success in elementary school?
- E How does Sesame Street cause an increase in educational attainment?

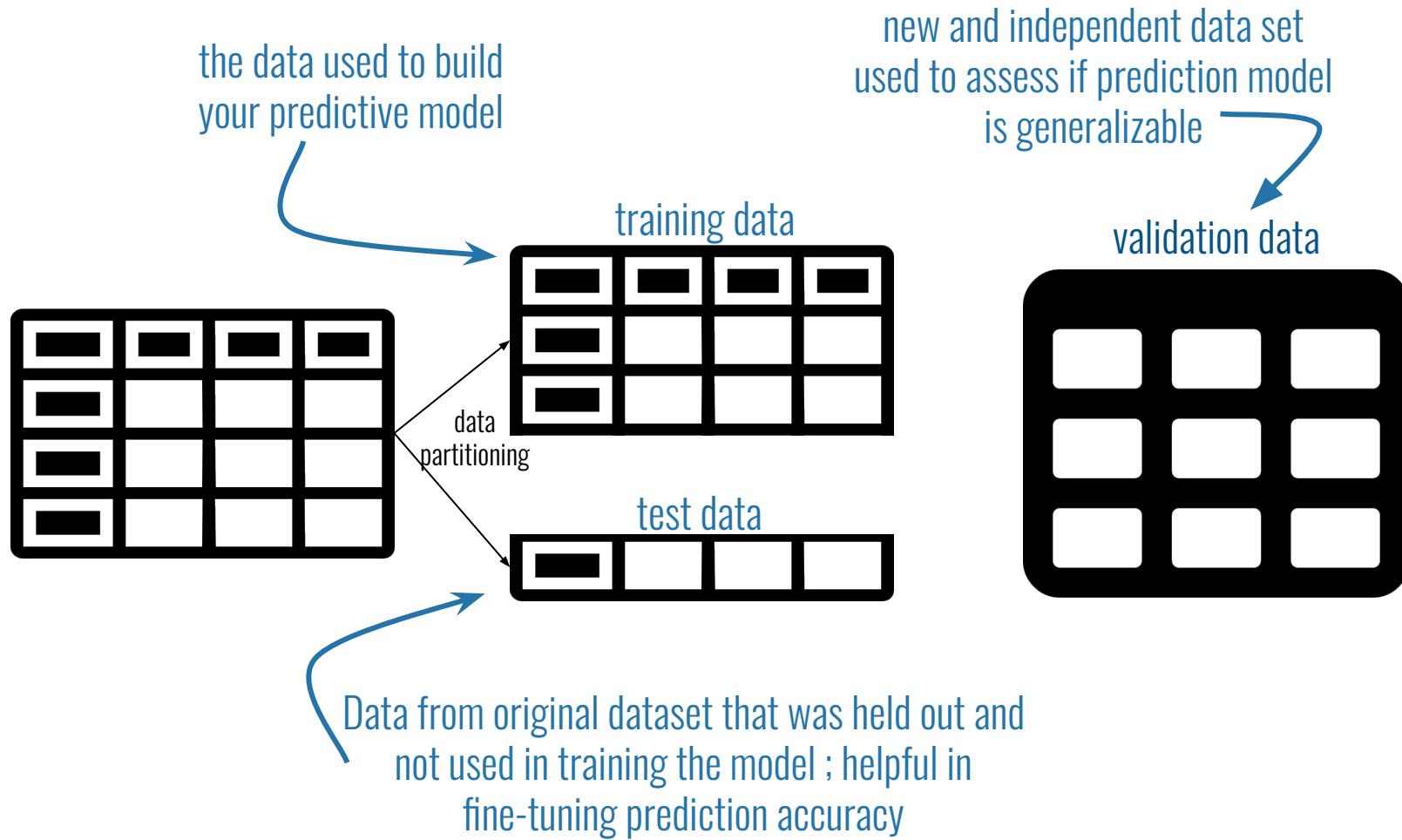
Machine Learning Generalizations

Basic Steps to Prediction





data partitioning

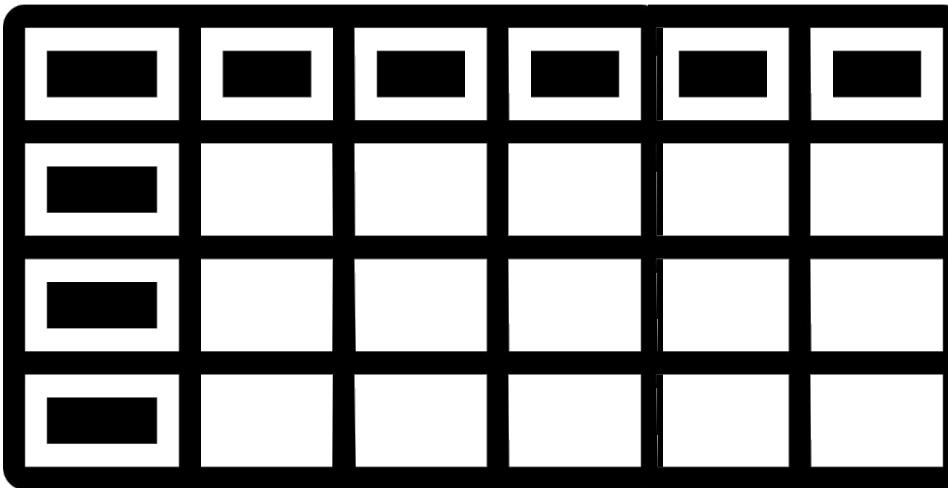




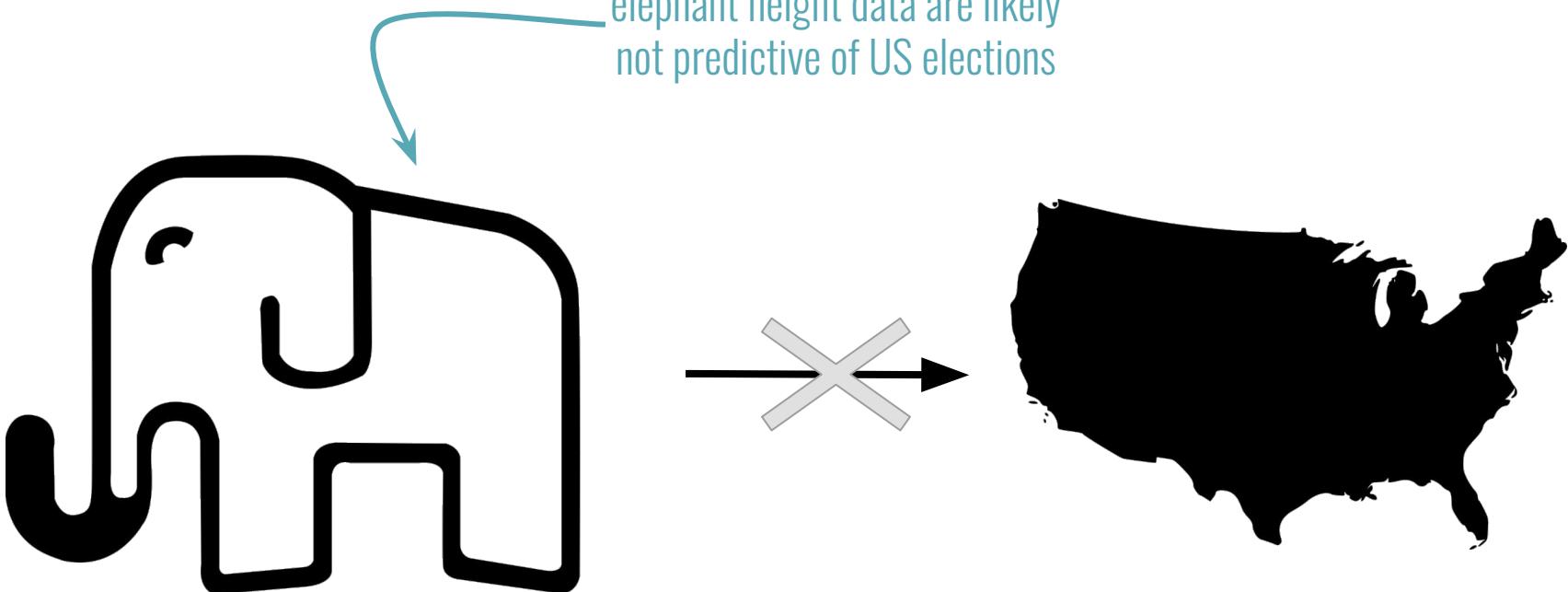
Data Partitioning

What portion of the data are typically used for generating the model?

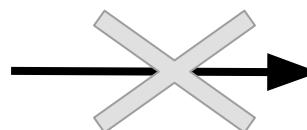
- A
The entire dataset
- B
The training data
- C
The testing data
- D
The validation data

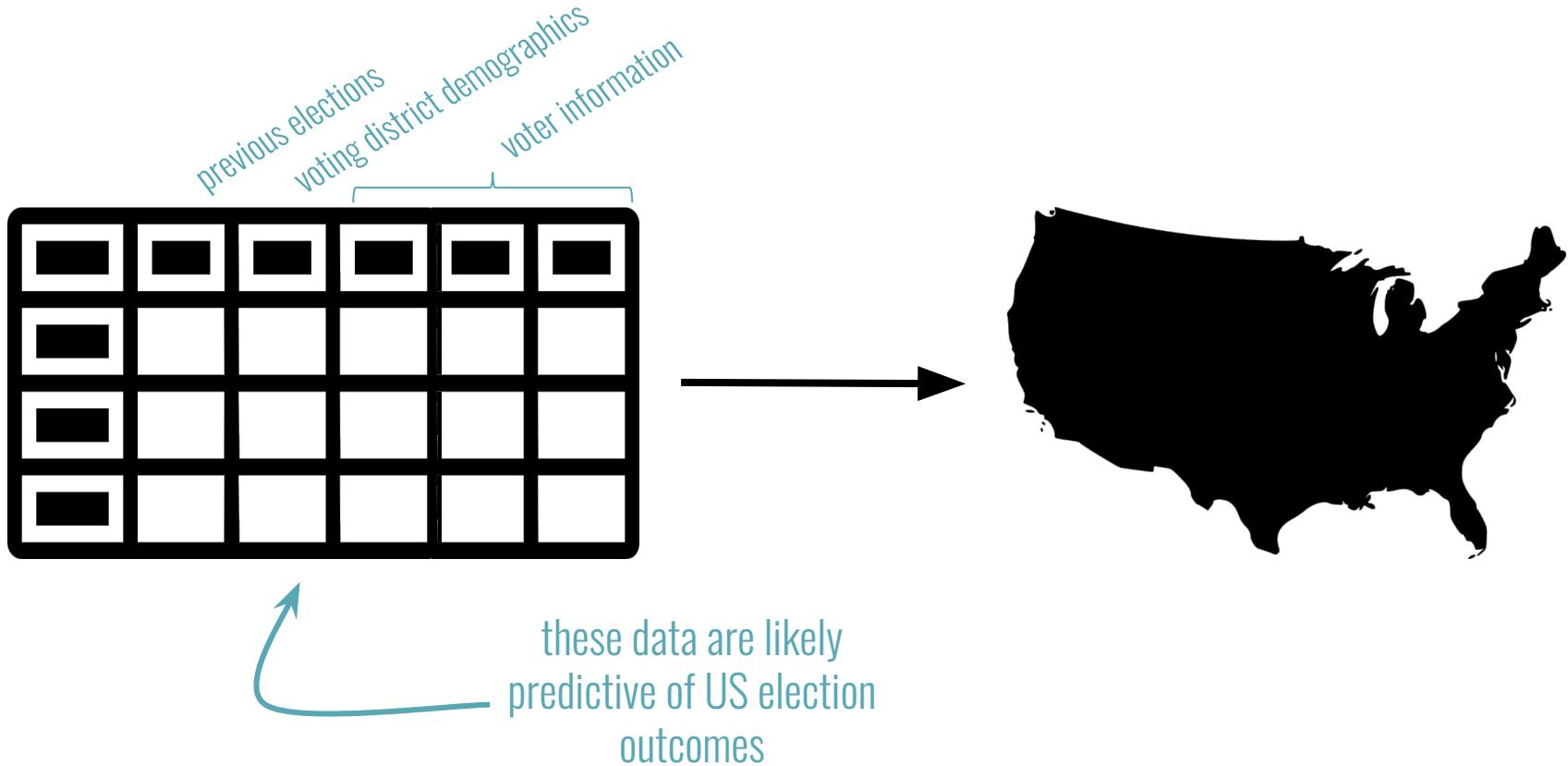


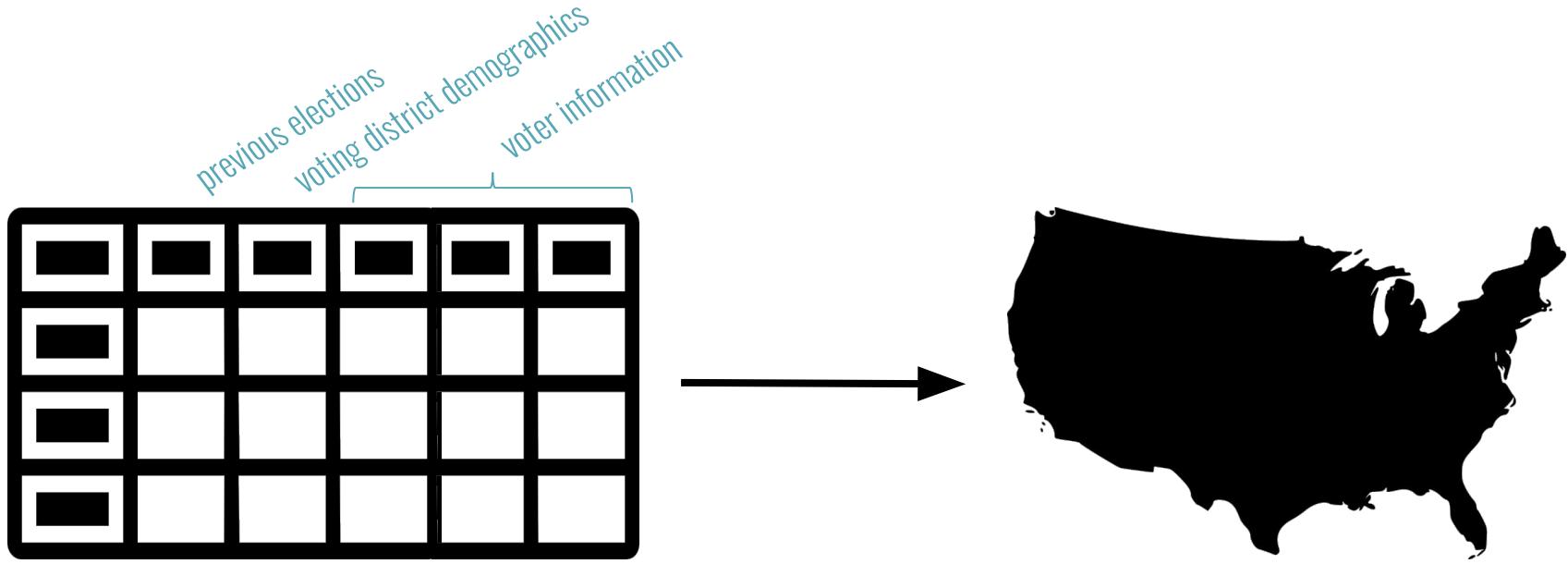
feature selection



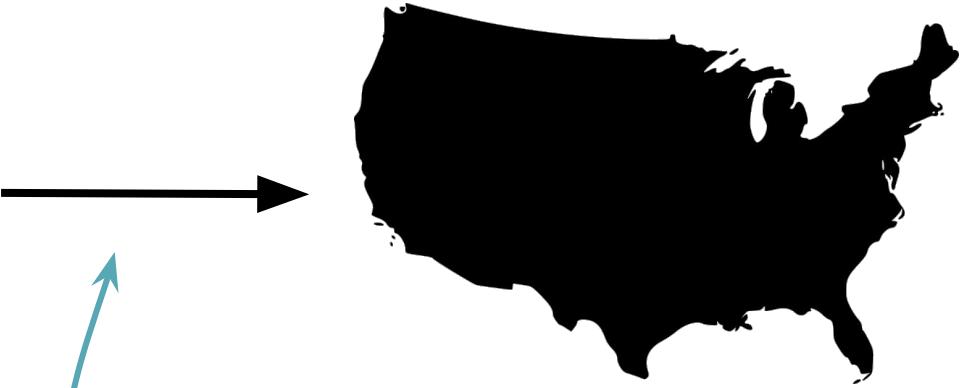
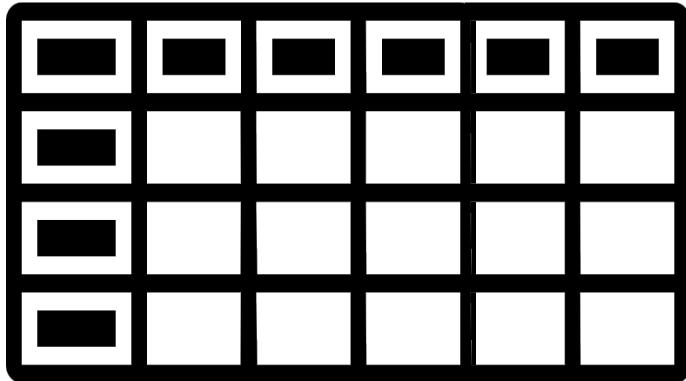
elephant height data are likely
not predictive of US elections







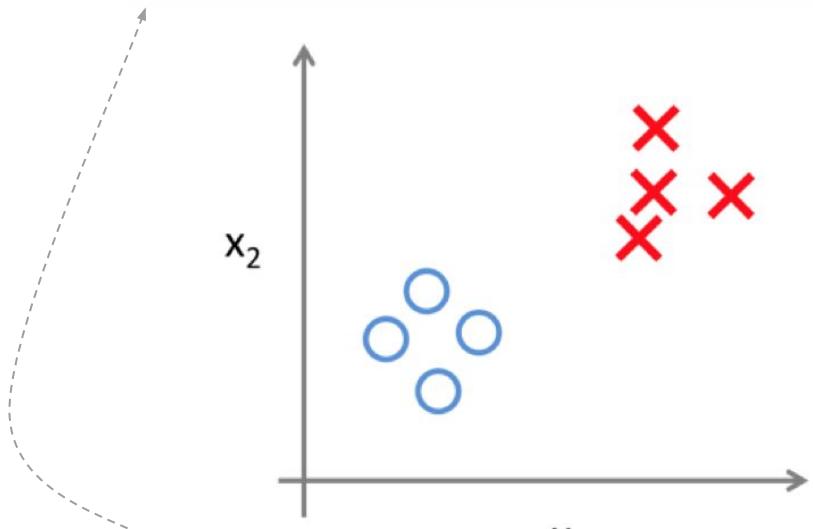
feature selection determines which variables are most predictive and includes them in the model



variables that can be used for accurate prediction exploit the relationship between the variables but do NOT mean that one causes the other

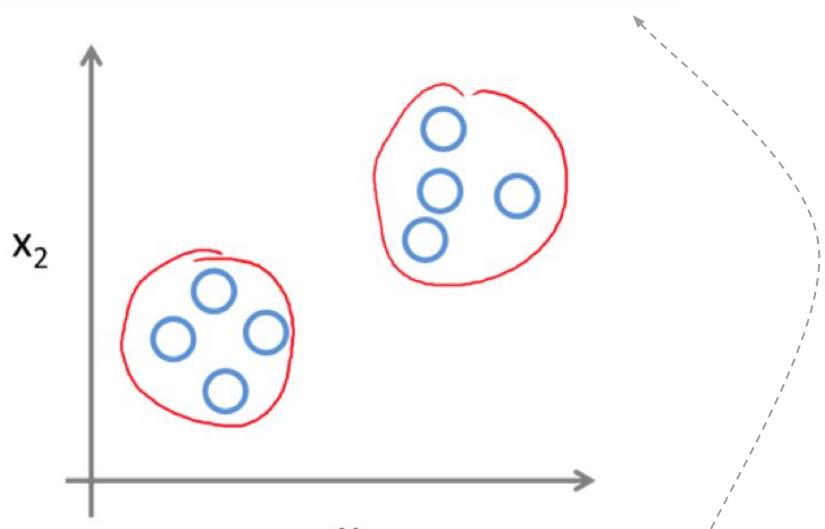
To modes of machine learning

Supervised Learning



You tell the computer how to classify the observations

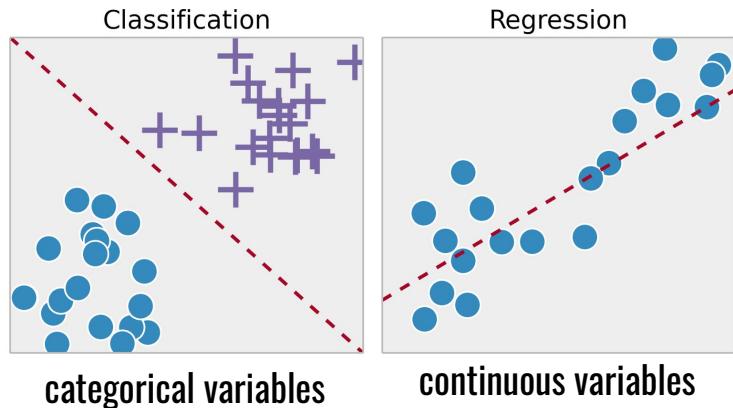
Unsupervised Learning



The computer determines how to classify based on properties within the data

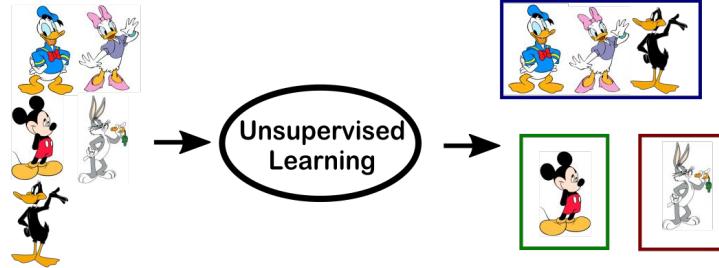
Approaches to machine learning

Supervised Learning



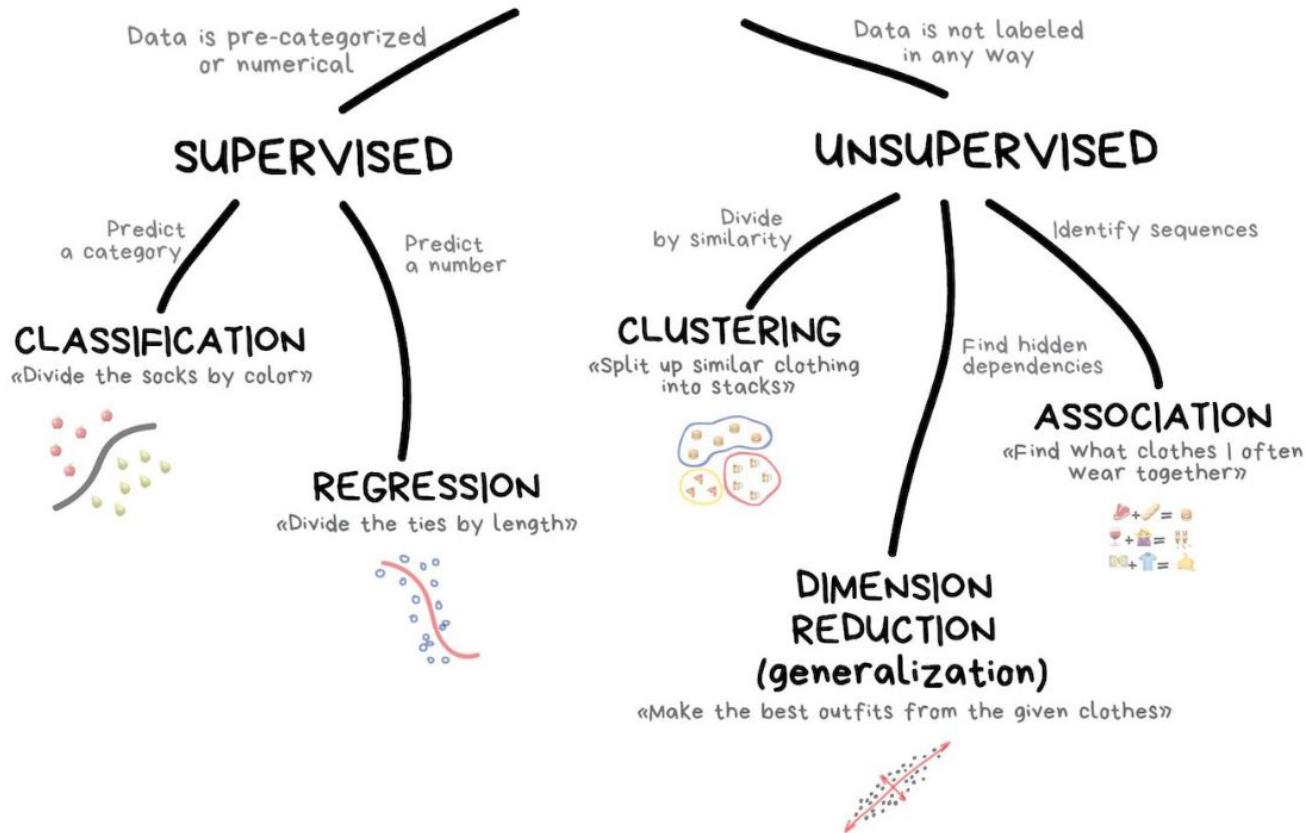
Prediction accuracy
dependent on
training data

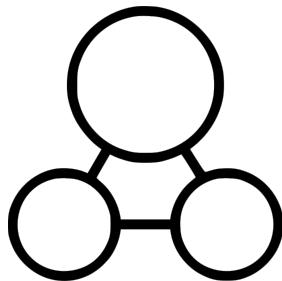
Unsupervised Learning



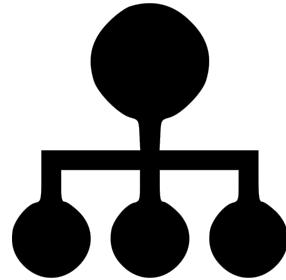
can automatically
identify structure in
data

CLASSICAL MACHINE LEARNING

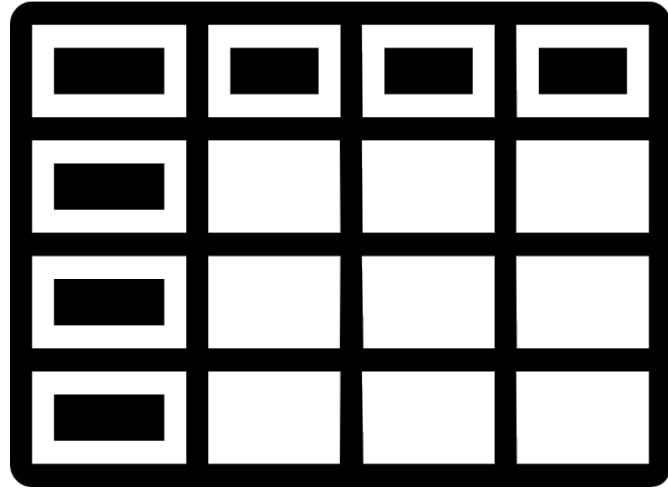




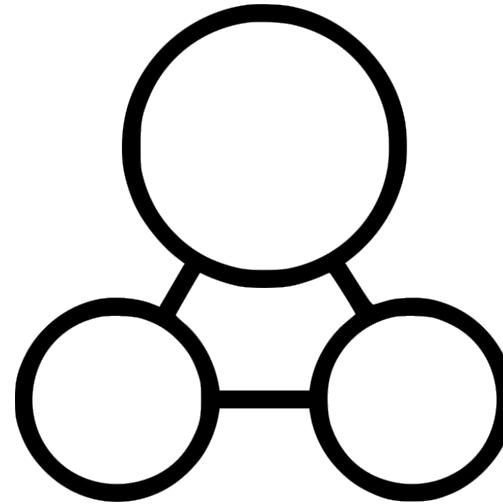
+	-
x	=



model selection



big
datasets



simple
models

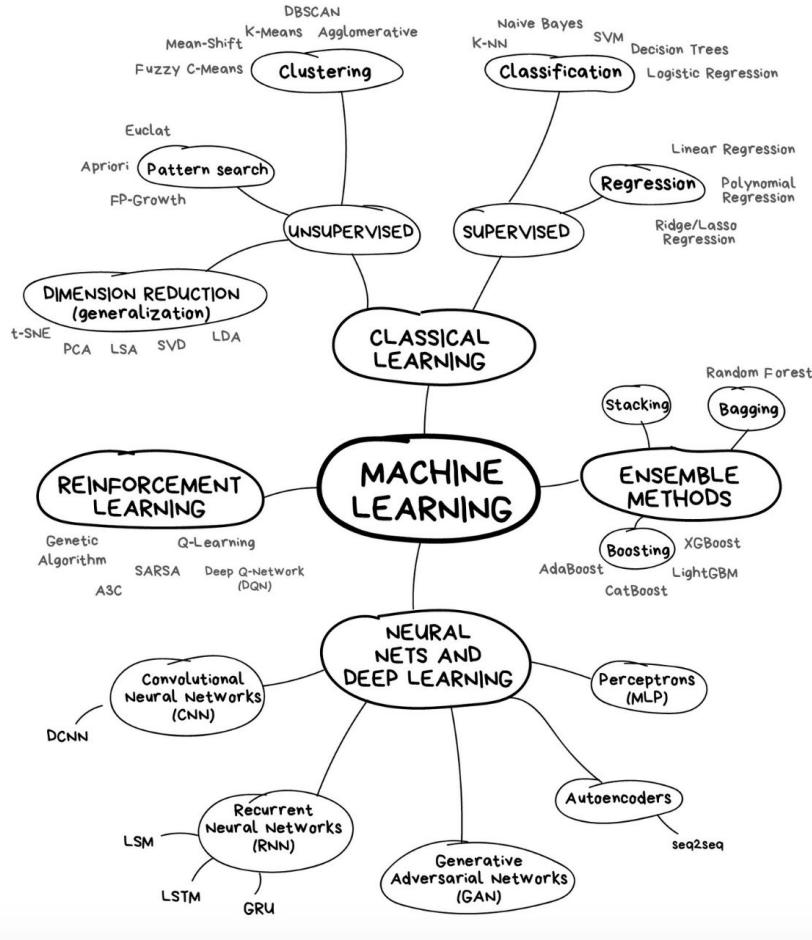
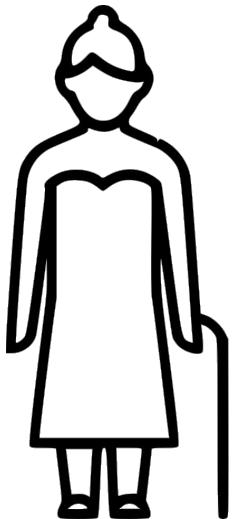


Image source: <https://vas3k.com/blog/machine-learning/>

Supervised Learning



Regression:
predicting continuous
variables
(i.e. Age)

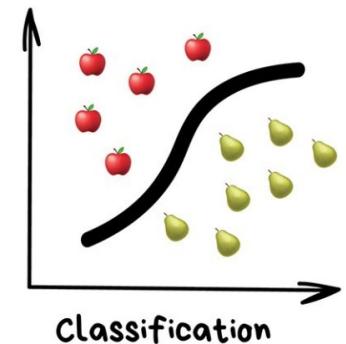
continuous variable prediction



Classification:
predicting categorical
variables
(i.e. education level)

categorical variable prediction

Classification

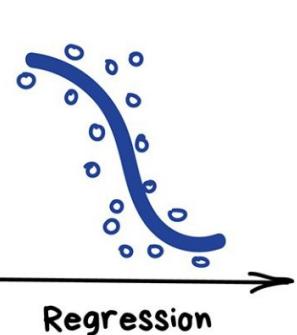


Classification

"Splits objects based at one of the attributes known beforehand. Separate socks by based on color, documents based on language, music by genre"

Today used for:

- Spam filtering
- Language detection
- A search of similar documents
- Sentiment analysis
- Recognition of handwritten characters and numbers
- Fraud detection



Regression

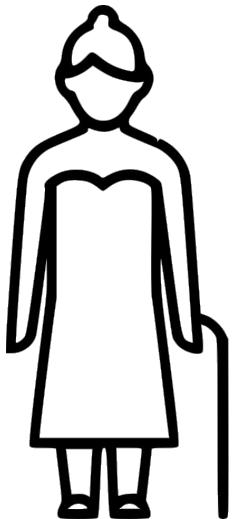
"Draw a line through these dots. Yep, that's the machine learning"

Today this is used for:

- Stock price forecasts
- Demand and sales volume analysis
- Medical diagnosis
- Any number-time correlations

Popular algorithms are Linear and Polynomial regressions.

Popular algorithms: Naive Bayes, Decision Tree, Logistic Regression, K-Nearest Neighbours, Support Vector Machine



Regression:

predicting continuous variables
(i.e. Age)

continuous variable prediction



Classification:

predicting categorical variables
(i.e. education level)

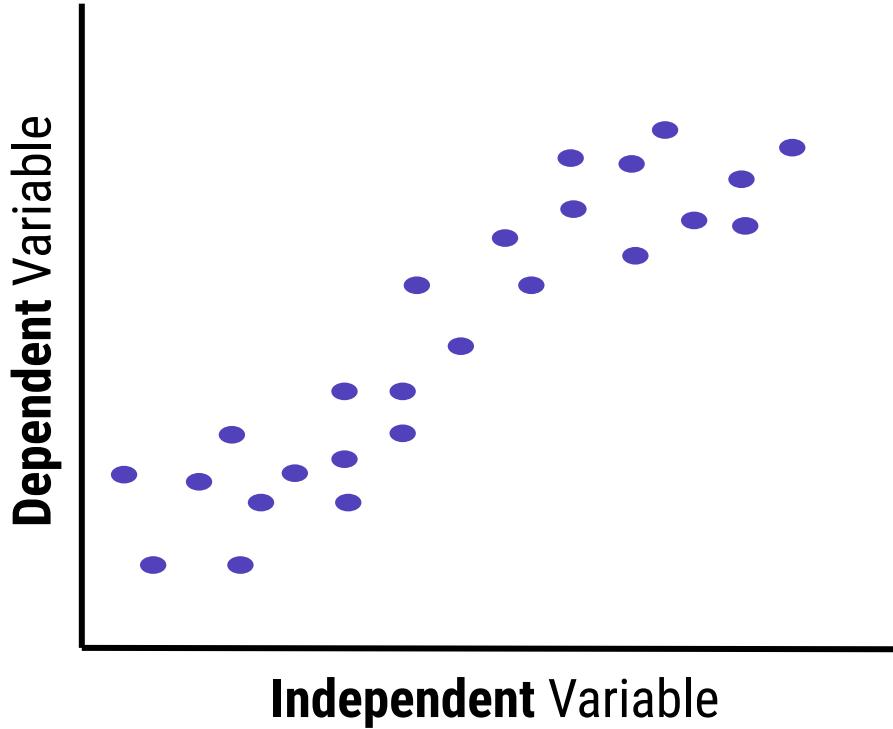
categorical variable prediction

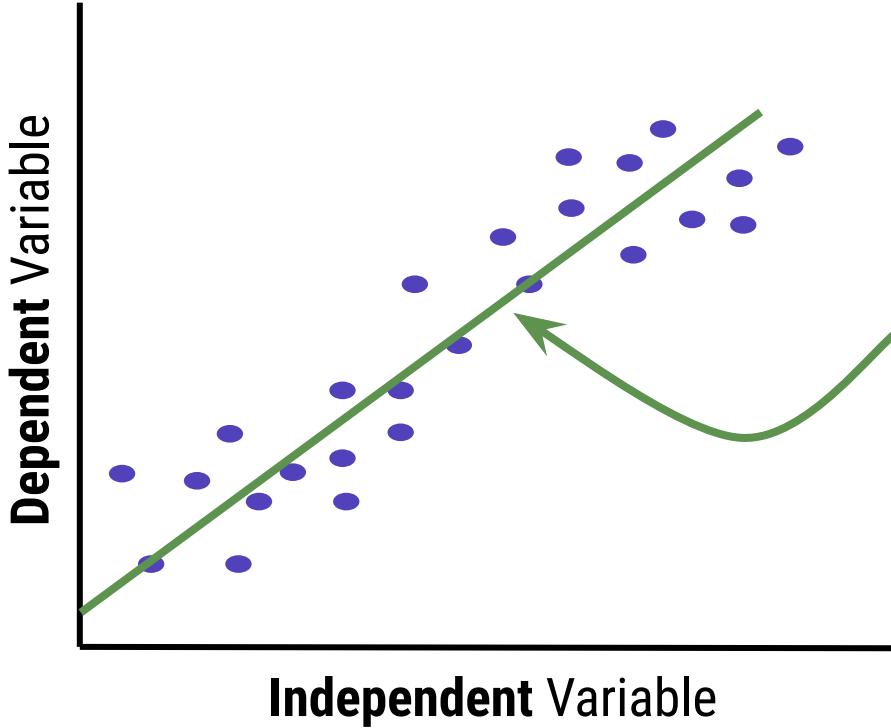
Supervised Learning

regression

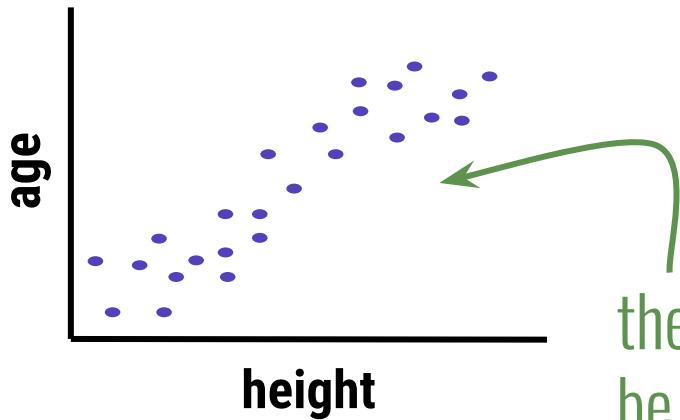
continuous variable prediction



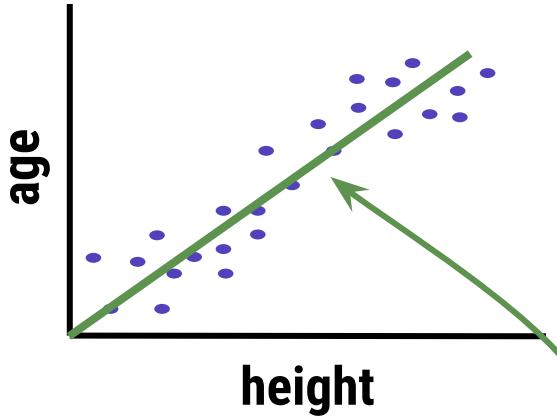
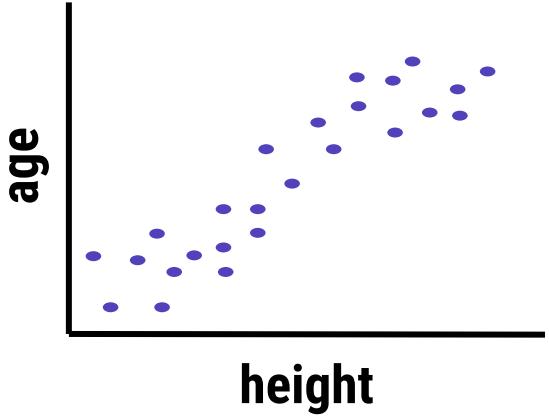




We'll use the linear relationship between variables to generate a **predictive model**



the training data will
be used to build the
predictive model



Supervised Learning

regression

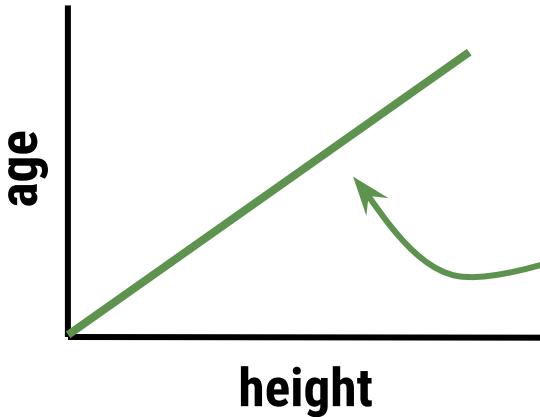
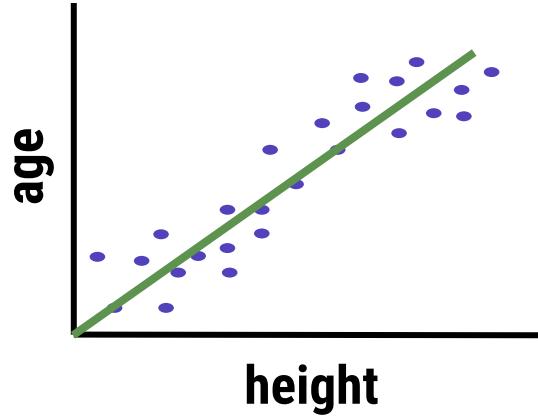
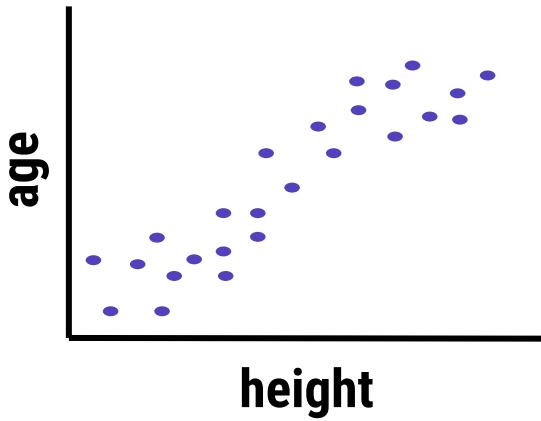
continuous variable prediction



use linear regression to
model the relationship



continuous variable prediction

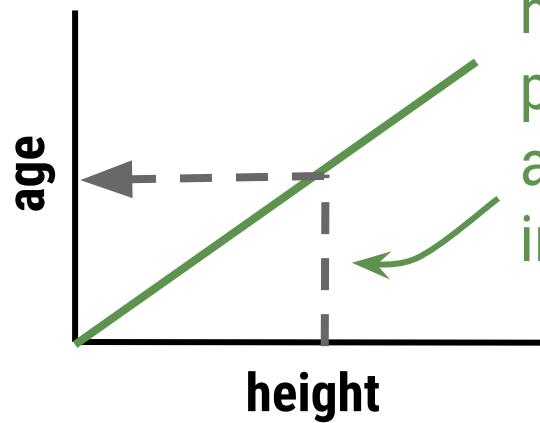
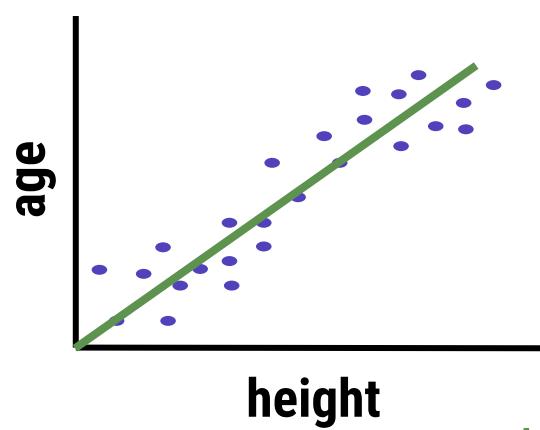
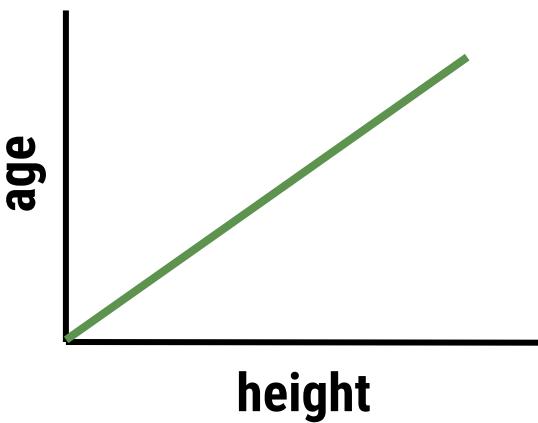
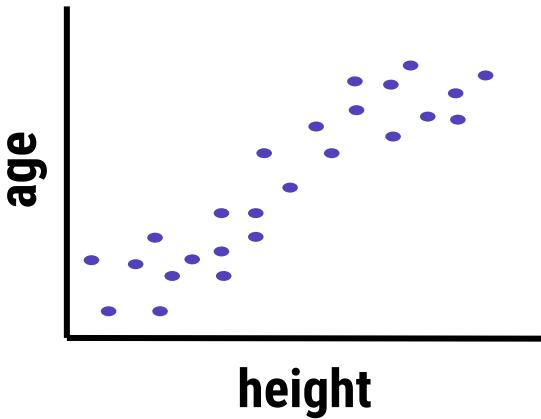


For prediction, the individual values in the training data are *not* important. We only need the model.

Supervised Learning

regression

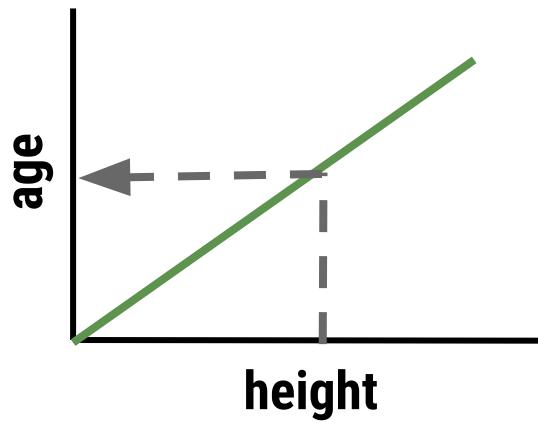
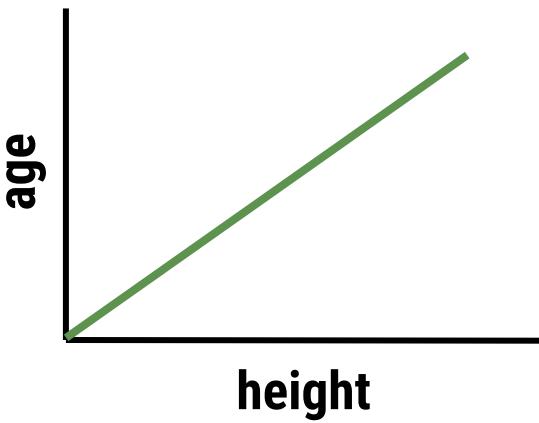
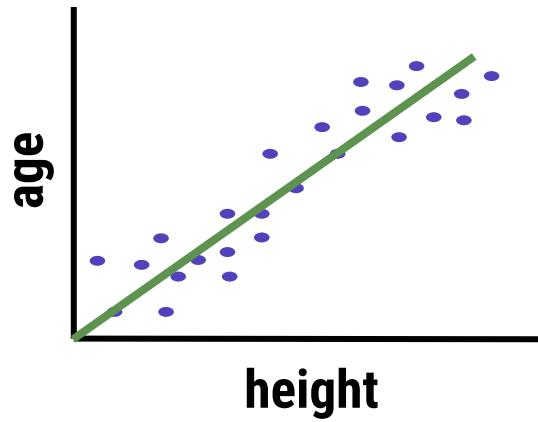
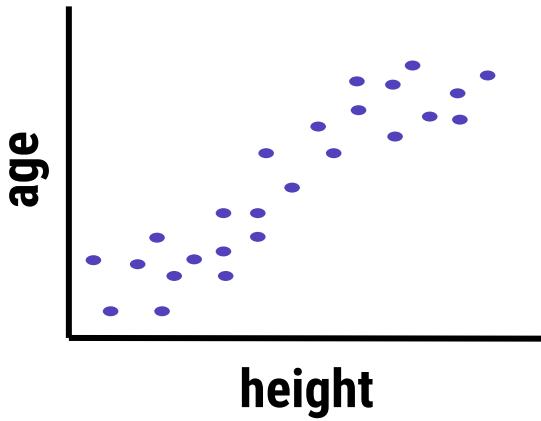
continuous variable prediction

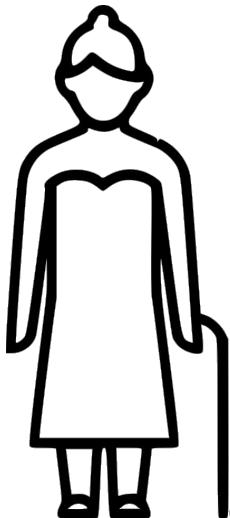


Supervised Learning

regression

continuous variable prediction





Regression:
predicting continuous
variables
(i.e. Age)

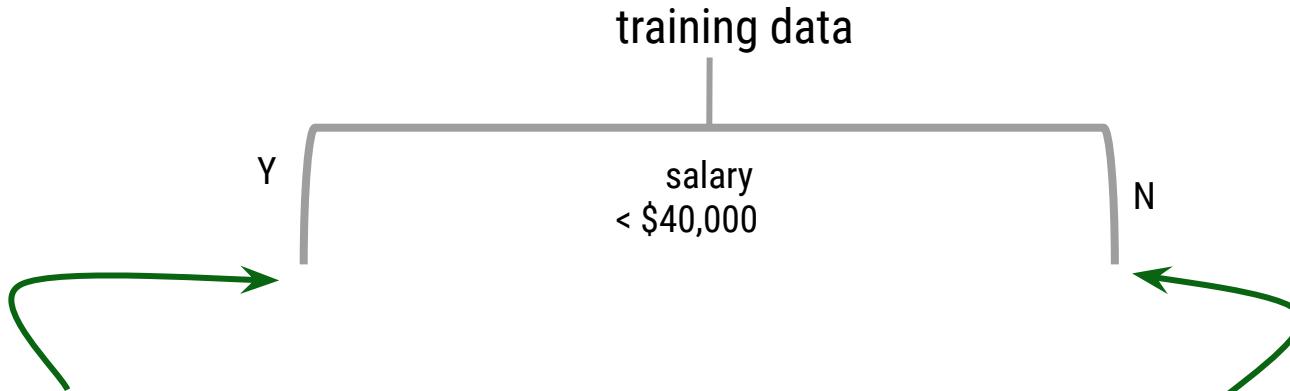


Classification:
predicting categorical
variables
(i.e. education level)

Supervised Learning

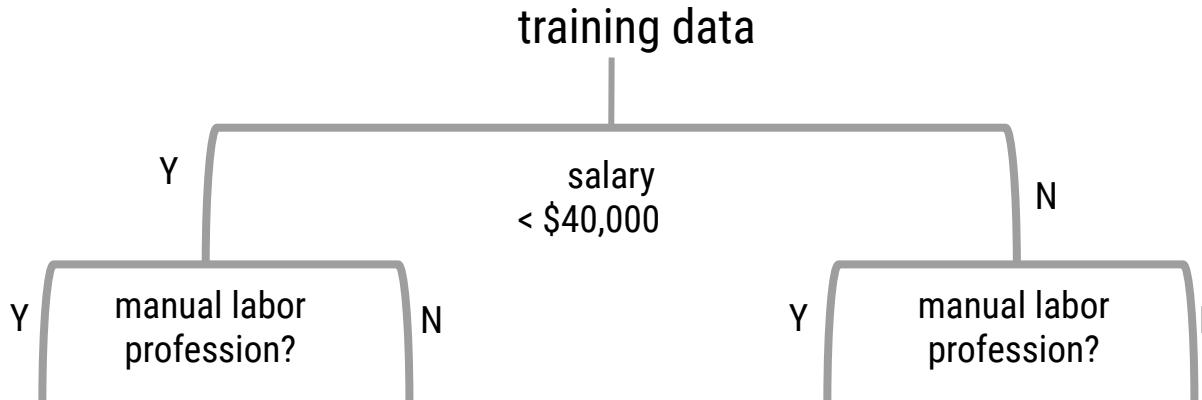
classification 

categorical variable prediction



All the people
who make *less*
than 40K over
here

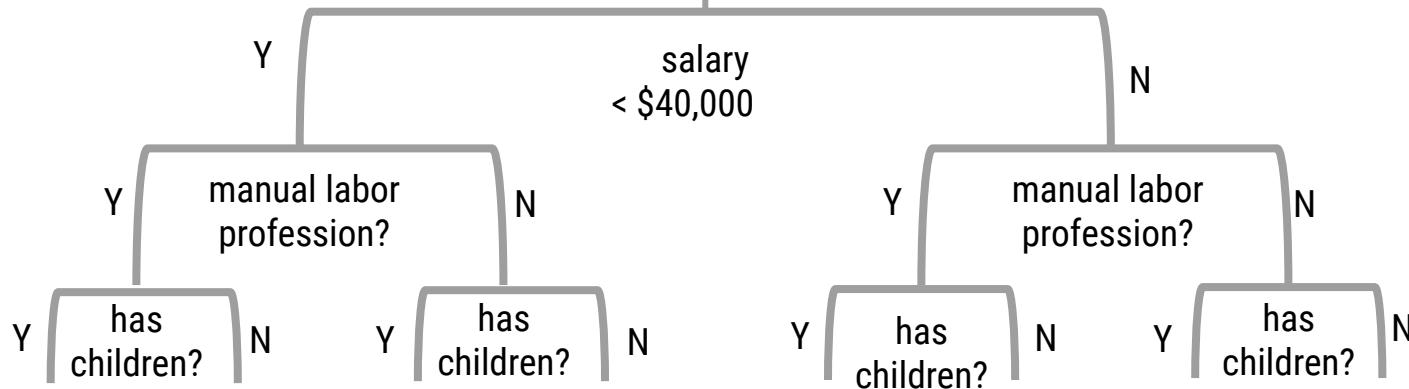
All the people
who make *more*
than 40K over
here



Continue adding *branches* to the **decision tree** where the variables and information in the training data decide which observations goes down which branch

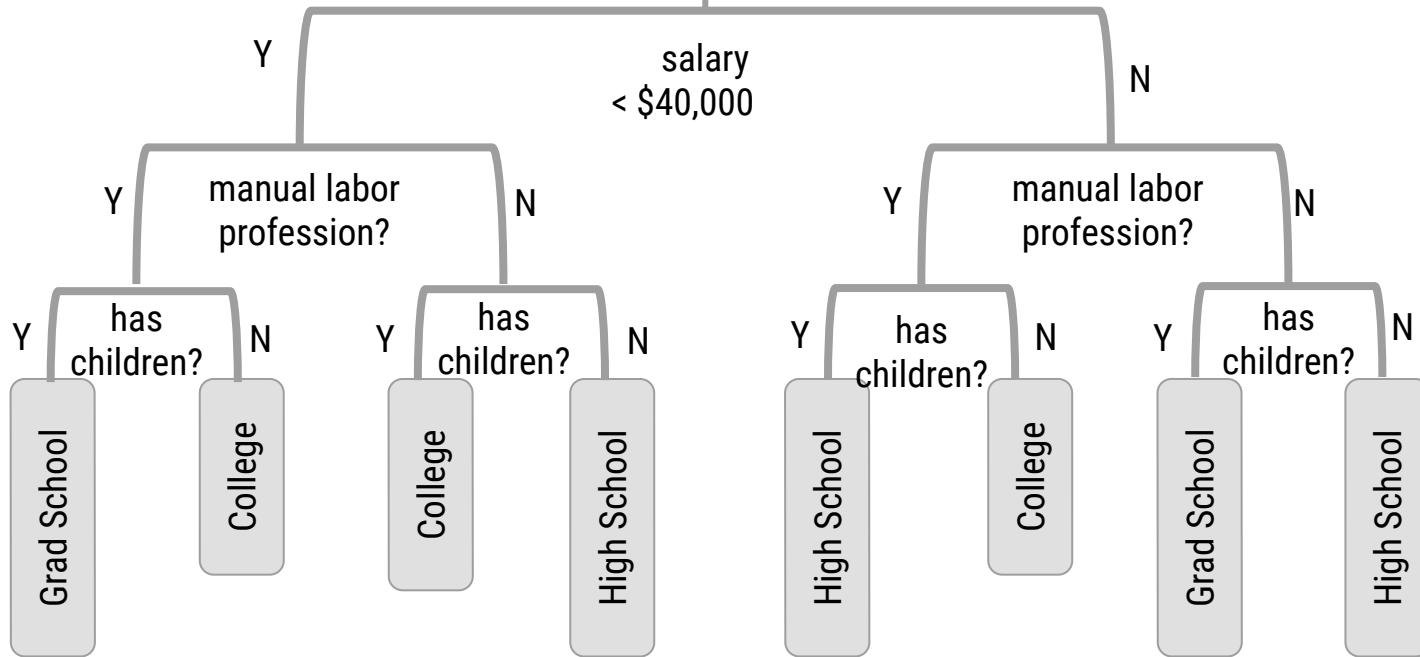


training data





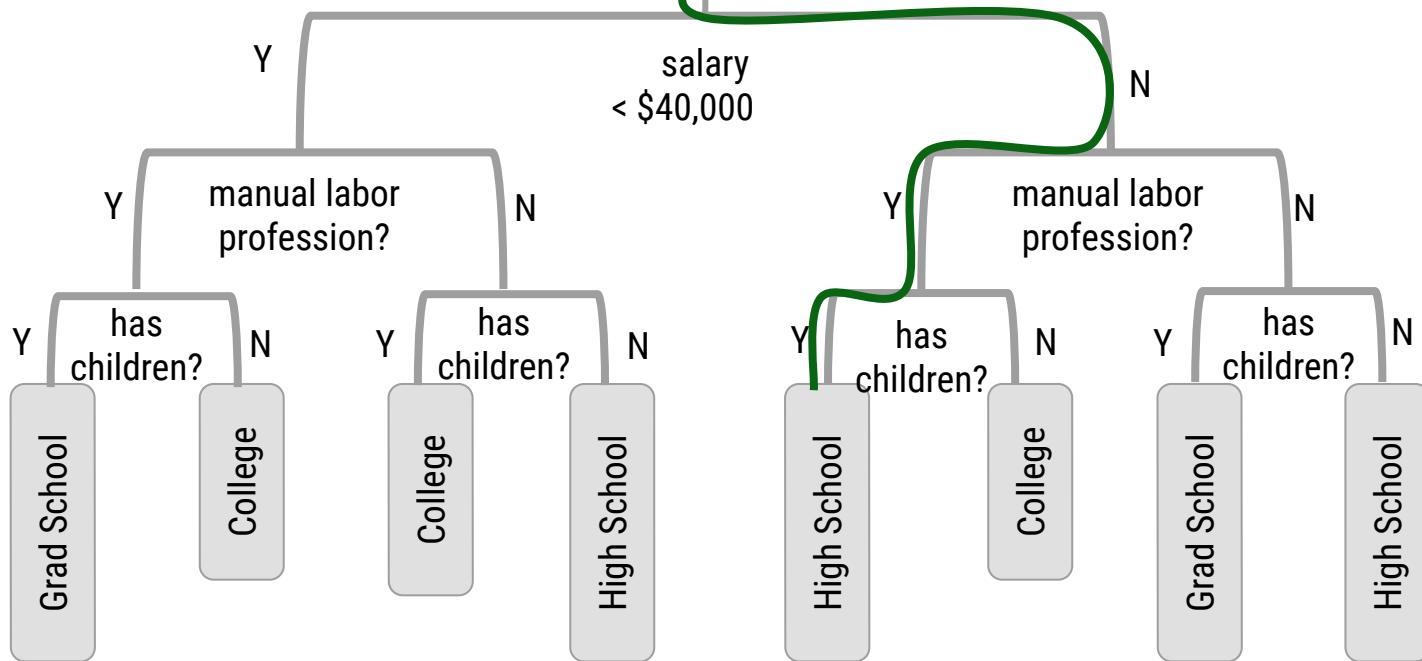
training data



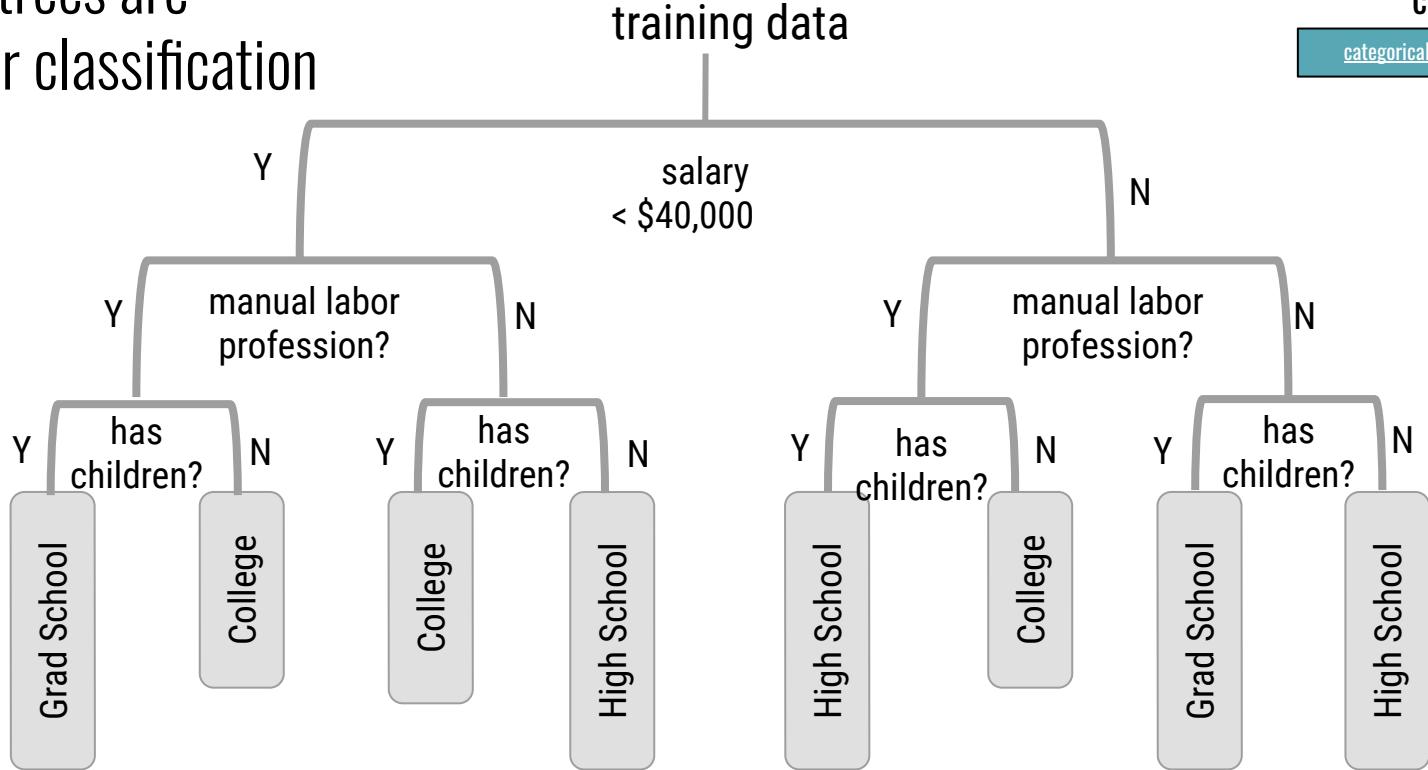
At the end of the tree, labels will be applied to each *leaf* of the tree



training data



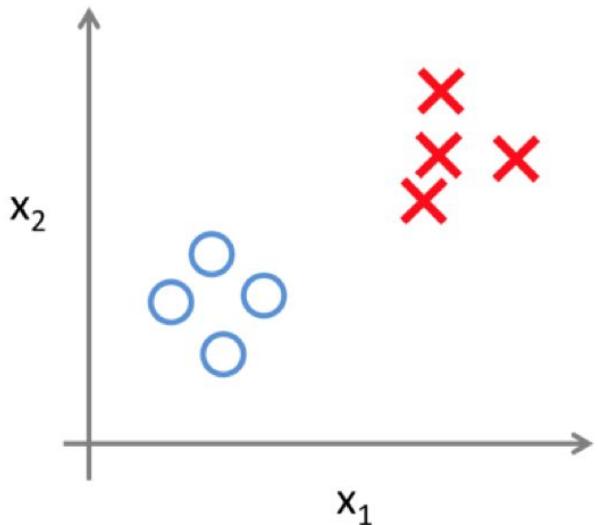
Decision trees are helpful for classification



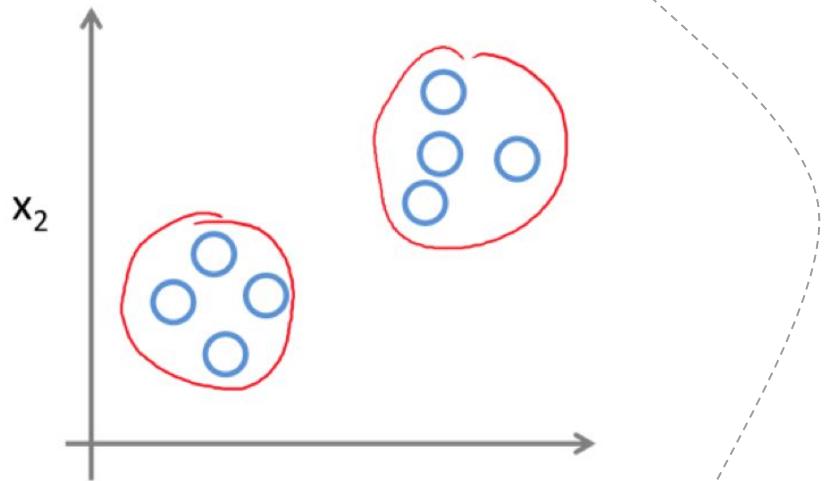
Unsupervised Learning

To modes of machine learning

Supervised Learning



Unsupervised Learning



The computer determines how to classify based on properties within the data

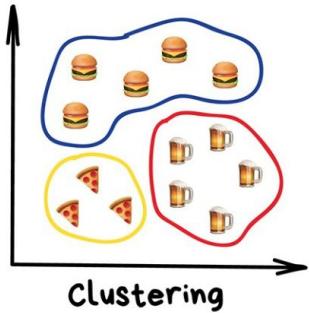
Dimensionality Reduction (Generalization)

Clustering

"Divides objects based on unknown features.
Machine chooses the best way"

Nowadays used:

- For market segmentation (types of customers, loyalty)
- To merge close points on a map
- For image compression
- To analyze and label new data
- To detect abnormal behavior

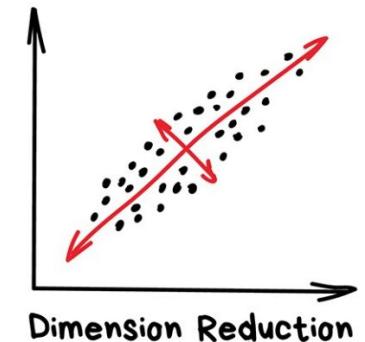


Popular algorithms: [K-means clustering](#), [Mean-Shift](#), [DBSCAN](#)

"Assembles specific features into more high-level ones"

Nowadays is used for:

- Recommender systems (★)
- Beautiful visualizations
- Topic modeling and similar document search
- Fake image analysis
- Risk management



Popular algorithms: [Principal Component Analysis \(PCA\)](#), [Singular Value Decomposition \(SVD\)](#), [Latent Dirichlet allocation \(LDA\)](#), [Latent Semantic Analysis \(LSA, pLSA, GLSA\)](#), [t-SNE](#) (for visualization)

**Features are given
as inputs**

No labels are
provided during
modeling

model identifies patterns
in the input data



**predictions are
output**

PCA, k-means clustering,
t-SNE, neural nets,
self-organizing maps,
(i.e. facial recognition, image
processing, EDA)



model identifies patterns
in the input data
→
Is it cake?



model identifies patterns
in the input data
Is it cake?



1



2



model identifies patterns
in the input data

Is it cake?



1



2

new emoji



→ Is it cake?

No

Yes

Class 1
NOT CAKE

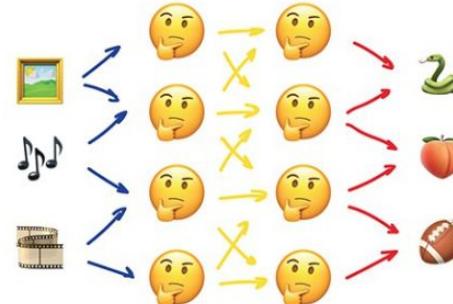
Class 2
CAKE



"We have a thousand-layer network, dozens of video cards, but still no idea where to use it. Let's generate cat pics!"

Used today for:

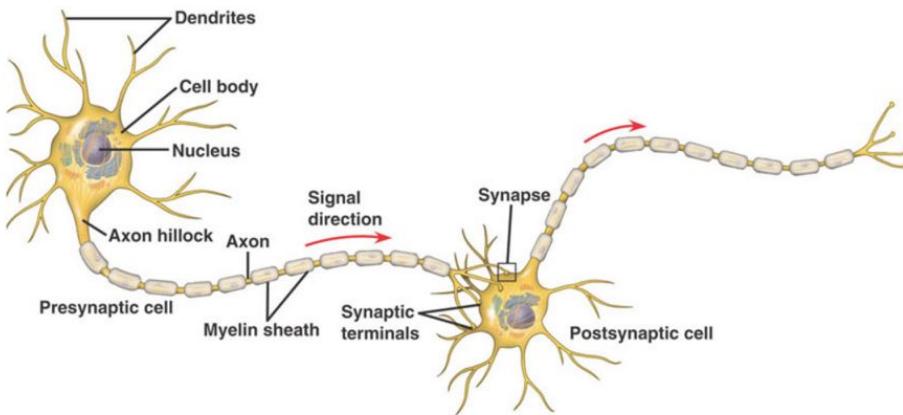
- Replacement of all algorithms above
- Object identification on photos and videos
- Speech recognition and synthesis
- Image processing, style transfer
- Machine translation



Neural Networks

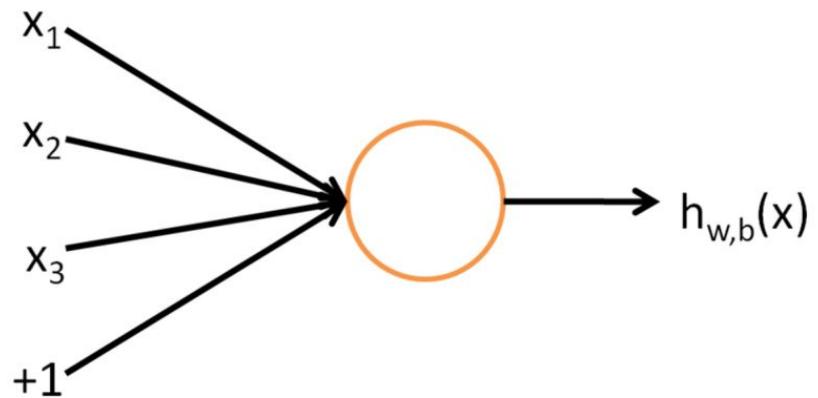
Popular architectures: Perceptron, Convolutional Network (CNN), Recurrent Networks (RNN), Autoencoders

WHAT IS A NEURON?



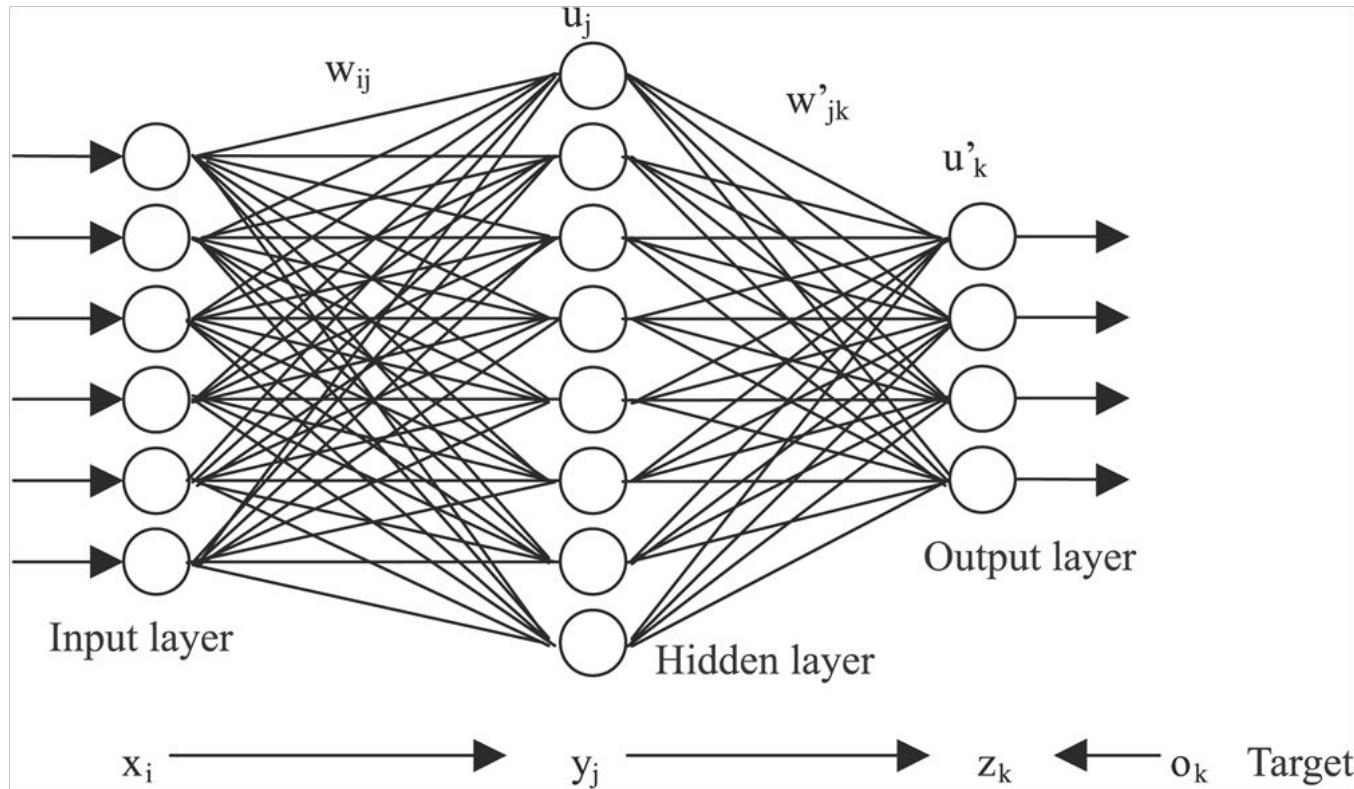
- Receives signal on synapse
- When trigger sends signal on axon

MATHEMATICAL NEURON



- Mathematical abstraction, inspired by biological neuron
- Either on or off based on sum of input

This will likely not be the last time you see this (mostly unhelpful) neural net image



HOW A DEEP NEURAL NETWORK SEES

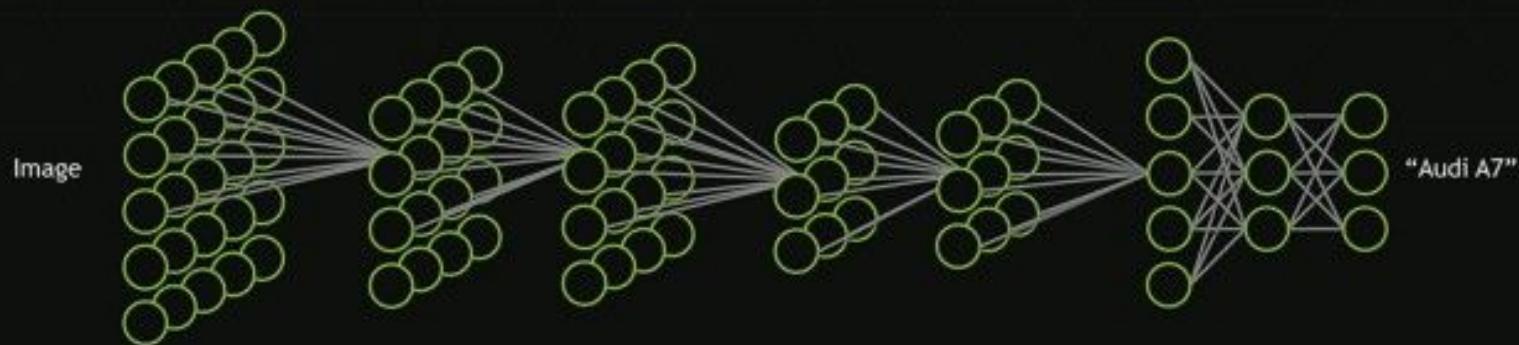
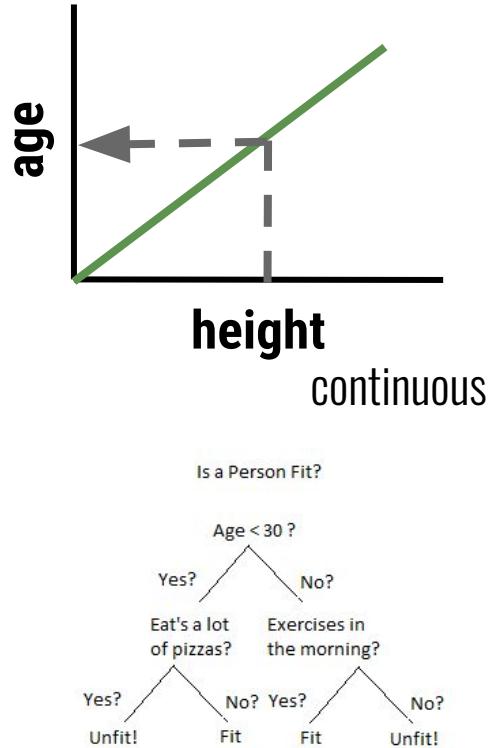


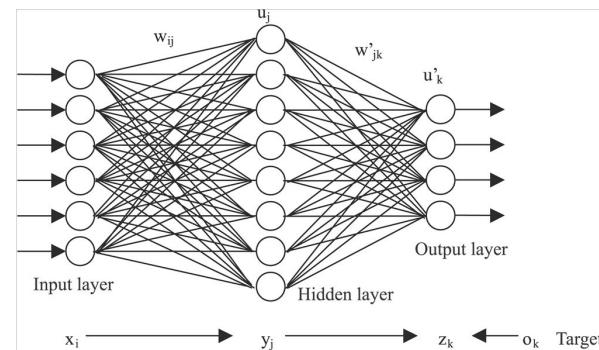
Image source: "Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks" (ICML 2009 & Comm. ACM 2011, Honglak Lee, Roger Grosse, Rajish Ranganath, and Andrew Ng)

Supervised Learning



classification

Unsupervised Learning



dimensionality reduction & clustering

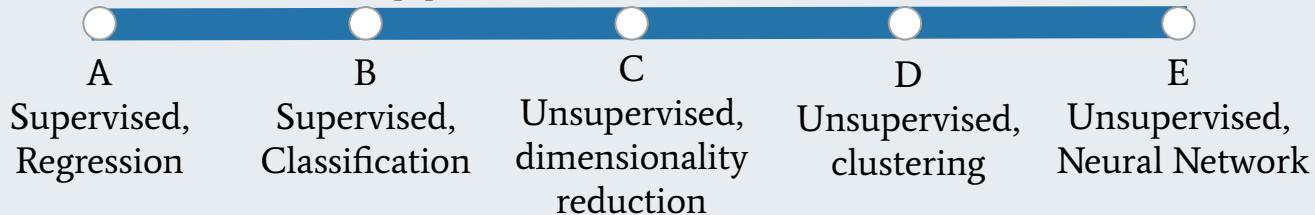




Prediction Approach

You want to predict someone's emotion based on an image.

How would you approach this with machine learning?





model assessment

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (\text{Predicted}_i - \text{Actual}_i)^2}{N}}$$



A few outliers can lead to a big increase in RMSE, even if all the other predictions are pretty good

categorical variable prediction

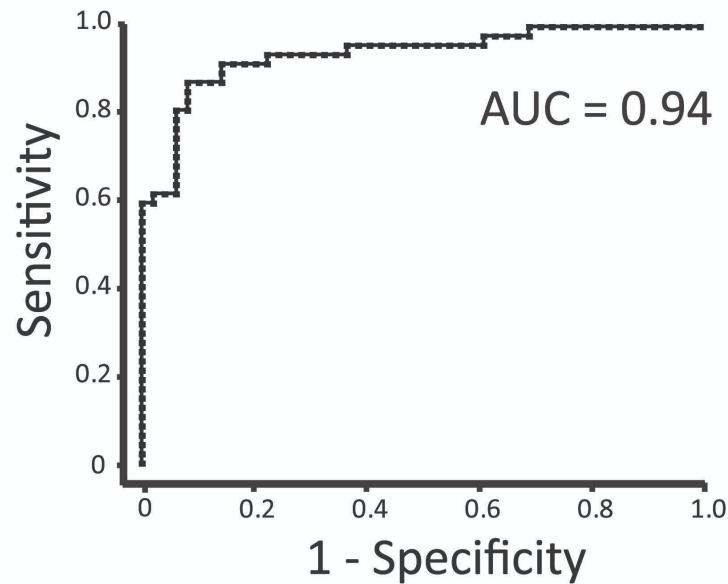
continuous variable prediction

$$\text{Accuracy} = \frac{\# \text{ of samples predicted correctly}}{\# \text{ of samples predicted}} * 100$$

		Actual	
		Positive	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

A 2x2 table is a type of
confusion matrix

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$



$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

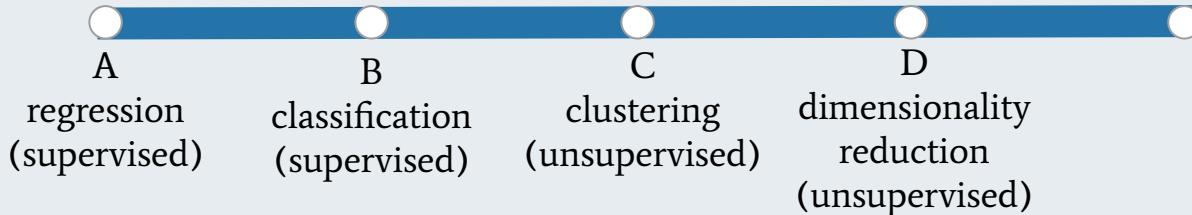
Accuracy	What % were predicted correctly?
Sensitivity	Of those that <i>were positives</i> , what % were predicted to be positive?
Specificity	Of those that were <i>negatives</i> , what % were predicted to be negative?



Prediction Approach

You've been given a dataset with a number of features and have been asked to predict each individual's age.

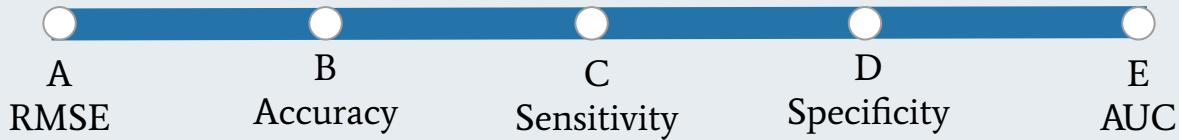
What prediction approach would you use?





Prediction Approach

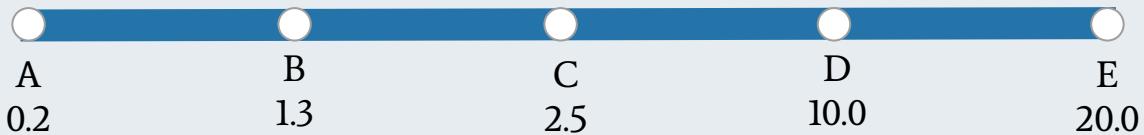
After predicting each person's age, how would you assess your model?





Prediction Approach

Which would be the error value you'd want from your model?



When models are trained
on historical data,
predictions will
perpetuate historical
biases



Predictive Analysis Ethics



Dare Obasanjo

@Carnage4Life

Product leader at Microsoft. My team is responsible for advertiser experience for Bing Ads; mobile apps, web UX, desktop apps & SDKs.



Dare Obasanjo

@Carnage4Life

Follow

Machine learning algorithms are driven more by the training data than math. Give an algorithm biased data then results will be biased. E.g.

- Amazon's resumé referral algo which auto rejected women
- Search ads algo which showed background check ads for "black sounding names"



Ryan Saavedra  @RealSaavedra

Socialist Rep. Alexandria Ocasio-Cortez (D-NY) claims that algorithms, which are driven by math, are racist

8:59 PM - 22 Jan 2019

Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin

8 MIN READ



SAN FRANCISCO (Reuters) - Amazon.com Inc's ([AMZN.O](#)) machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

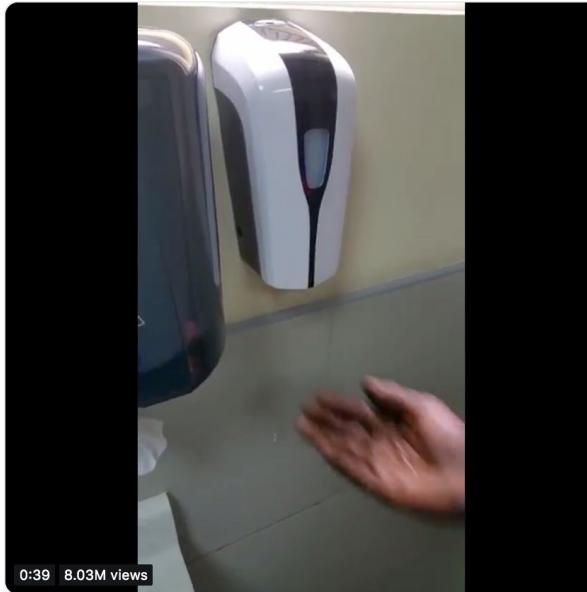




Chukwuemeka Afigbo
@nke_ise

Follow

If you have ever had a problem grasping the importance of diversity in tech and its impact on society, watch this video



5:48 AM - 16 Aug 2017

155,234 Retweets 215,762 Likes



https://twitter.com/nke_ise/status/897756900753891328

What to do about bias...

1. Anticipate and plan for potential biases before model generation. Check for bias after.
2. Have diverse teams.
3. Use machine learning to improve lives rather than for punitive purposes.
4. Revisit your models. Update your algorithms.
5. You are responsible for the models you put out into the world, unintended consequences and all.

Discussed so far...

- data partitioning
- feature selection
- supervised & unsupervised machine learning
 - Continuous variables: regression (supervised) and dimensionality reduction (unsupervised)
 - Categorical variables: classification (supervised; decision trees) or clustering (unsupervised)
- model assessment
 - Continuous: RMSE (& Accuracy)
 - Categorical: Accuracy, Sensitivity, Specificity, AUC
- biased data can & will lead to biased predictions

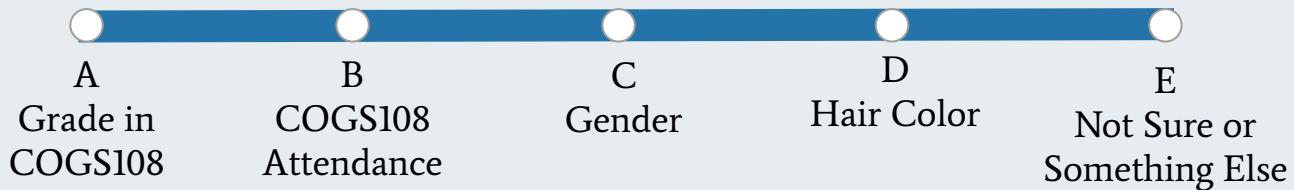
Data Science Question

Based on data I have about you all, can I predict
who in this course will be successful?



Prediction Approach

Which would be the most predictive of your future success?



N = 254

train the model:
N = 178
(70% of the data)

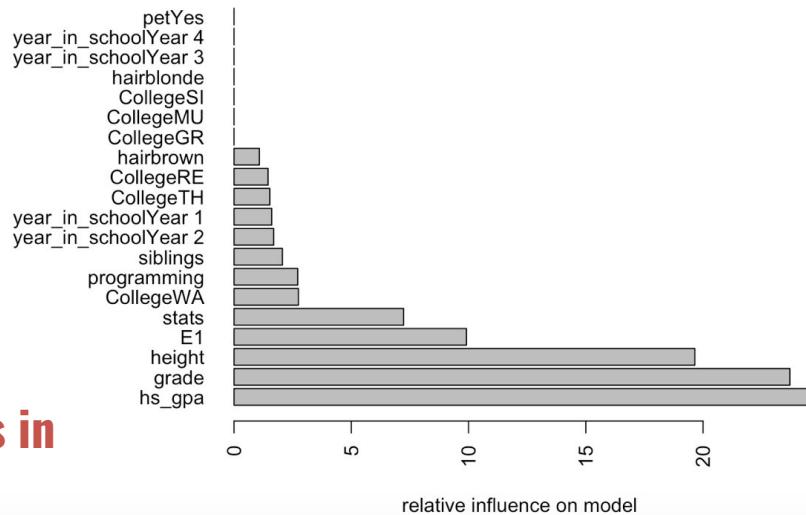
test the model:
N = 76
(30% of the data)

train the model

**predicted success in
test set**

	Accuracy	Sensitivity	Specificity
training set	71.2%	76%	67%
test set	49.1%	40%	60%

Assess Prediction Model



N = 254

train the model:
N = 178
(70% of the data)

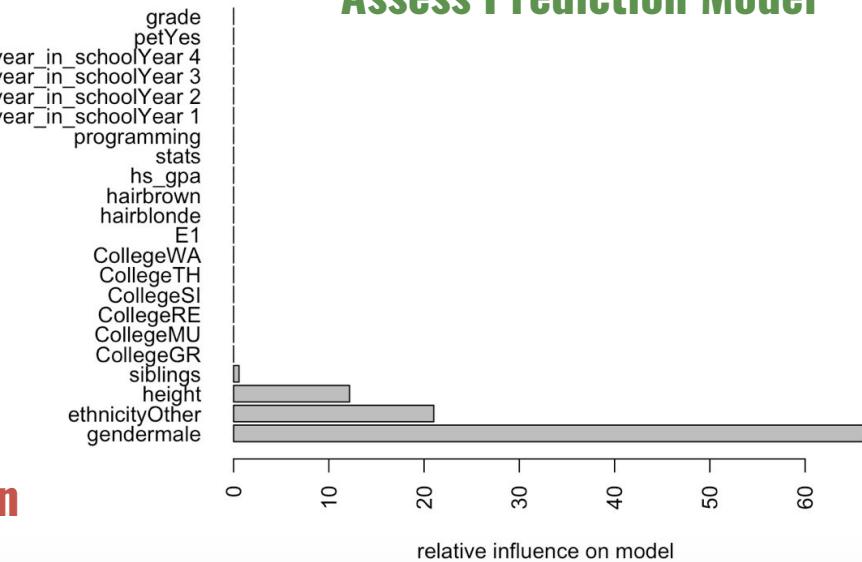
test the model:
N = 76
(30% of the data)

train the model

**predicted success in
test set**

	Accuracy	Sensitivity	Specificity
training set	100%	100%	100%
test set	100%	100%	100%

Assess Prediction Model



What if I were using these data to determine who I should write recommendation letters for?

Or to determine which students I focus my attention on?

Or whose projects I read?

Or who I allow to come to office hours?

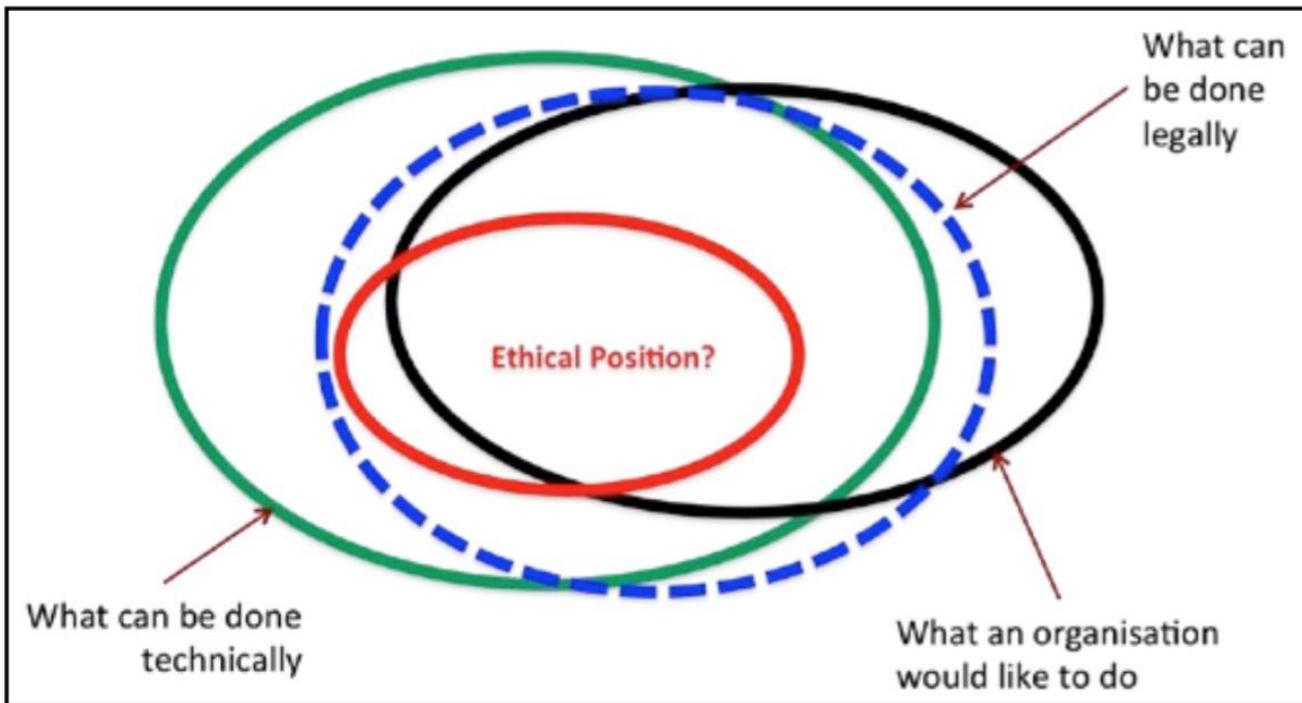
Or who UCSD allows to be data science majors?

What to do about bias...

1. Anticipate and plan for potential biases before model generation. Check for bias after.
2. Have diverse teams.
3. Use machine learning to improve lives rather than for punitive purposes.
4. Revisit your models. Update your algorithms.
5. You are responsible for the models you put out into the world, unintended consequences and all.

Think about whether the models you're building should even be built.

Big Data Ethics



Predictive algorithms should (*at a minimum*) be FAT

Fair: lacking biases which create unfair and discriminatory outcomes

- For whom does this algorithm fail?
- Steps to take:
 1. Verify data about individual is correct
 2. Carry out “sensitivity test”

Accountable/Accurate: answerable to the people subject to them

- Correct data used? Is there a mechanism for appeal?

Transparent: open about how and why particular decisions were made

- Think *carefully* about what transparency is (Handing over source code likely isn't the answer)

A Mulching Proposal

Analysing and Improving an Algorithmic System for Turning the Elderly into High-Nutrient Slurry

Os Keyes

Department of Human Centered Design & Engineering
University of Washington
Seattle, WA, USA
okaneys@uw.edu

Meredith Durbin

Department of Astronomy
University of Washington
Seattle, WA, USA
mdurbin@uw.edu

Jevan Hutson

School of Law
University of Washington
Seattle, WA, USA
jevanh@uw.edu

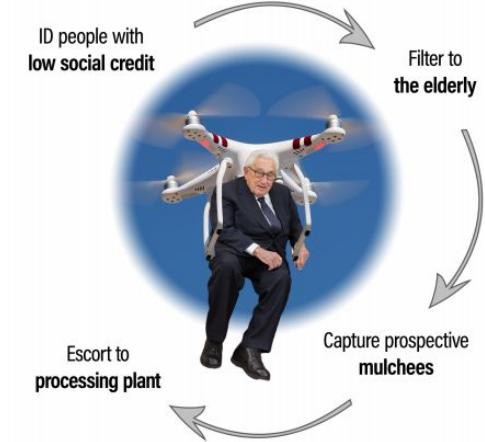
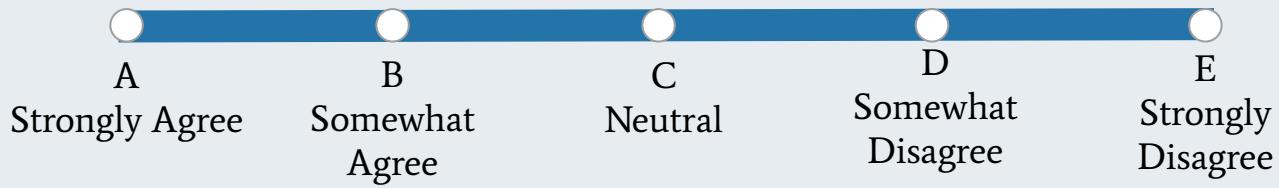


Figure 1: A publicity image for the project, produced by Logan-Nolan Industries



Prediction Thoughts

We should start using this algorithm to mulch up the elderly

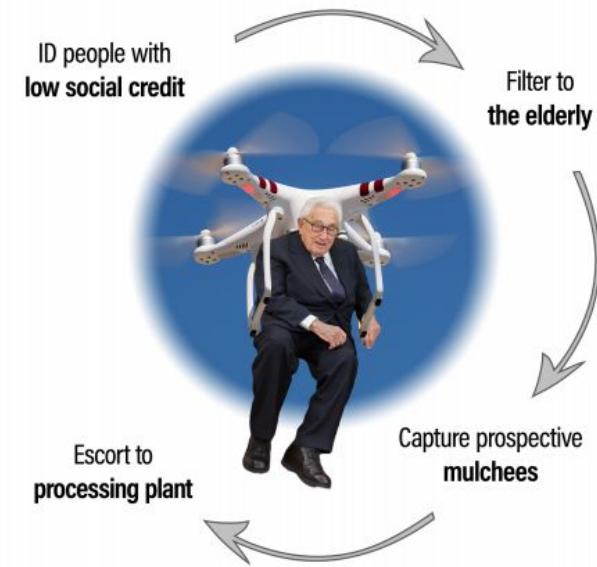


A Mulching Proposal

FAIR - equally considers all elderly individuals

ACCURATE - pre- has mechanism for appeal; post - compensation

TRANSPARENT - website with all features; testable



Logan-Nolan Industries
Helping Humanity Make Ends Meat

Figure 1: A publicity image for the project, produced by Logan-Nolan Industries

Checklists are helpful, but they're not an excuse for thoughtlessness.

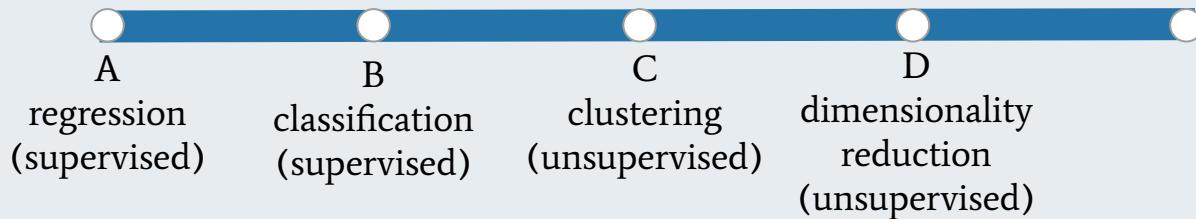
Worked Example I:

Predicting whether a home is in San Francisco, CA or New York, NY



Prediction Approach

What type of machine learning task is “Predicting whether a home is in San Francisco, CA or New York, NY?”



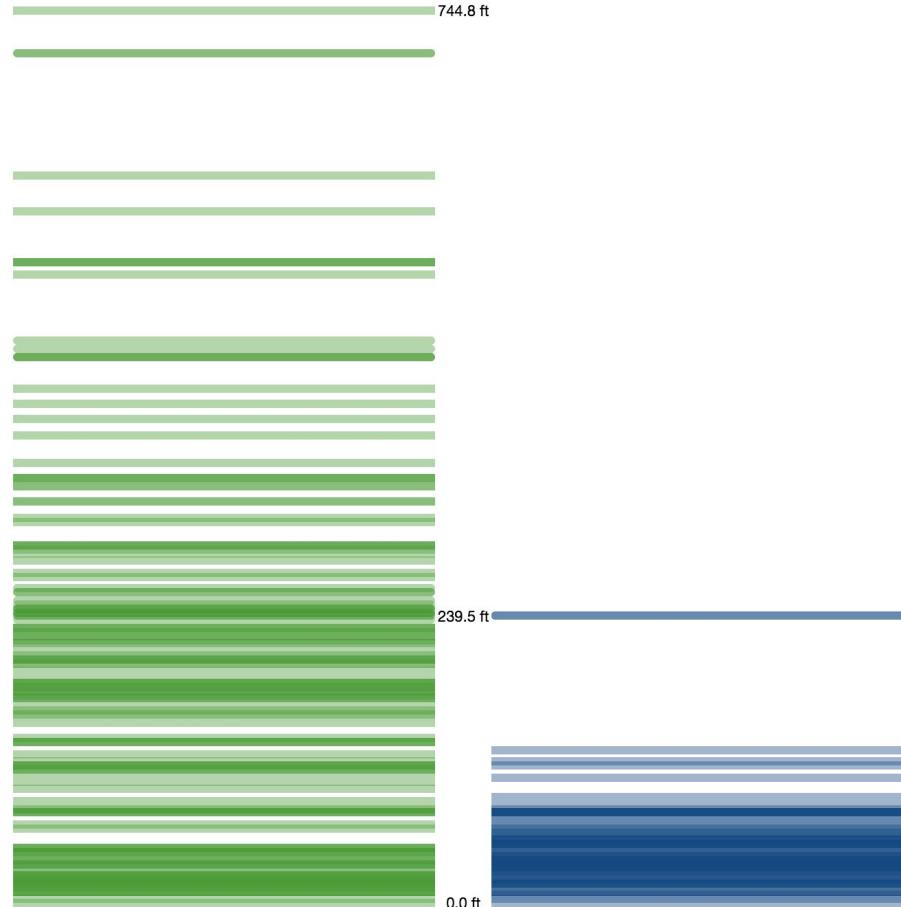
What features distinguish a
house in New York from a
house in San Francisco?

First, some intuition

Let's say you had to determine whether a home is in **San Francisco** or in **New York**. In machine learning terms, categorizing data points is a **classification** task.

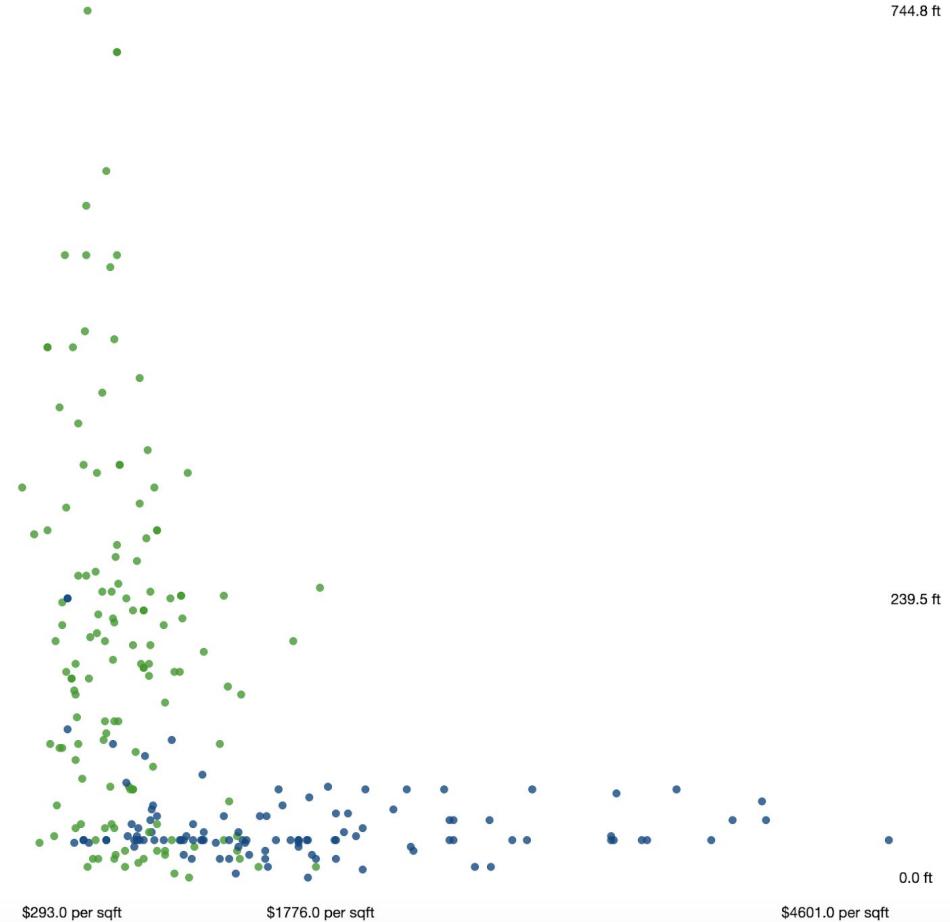
San Fran is hilly ...so elevation may be a helpful feature.

With the data here, homes > ~73m should be classified as San Fran homes



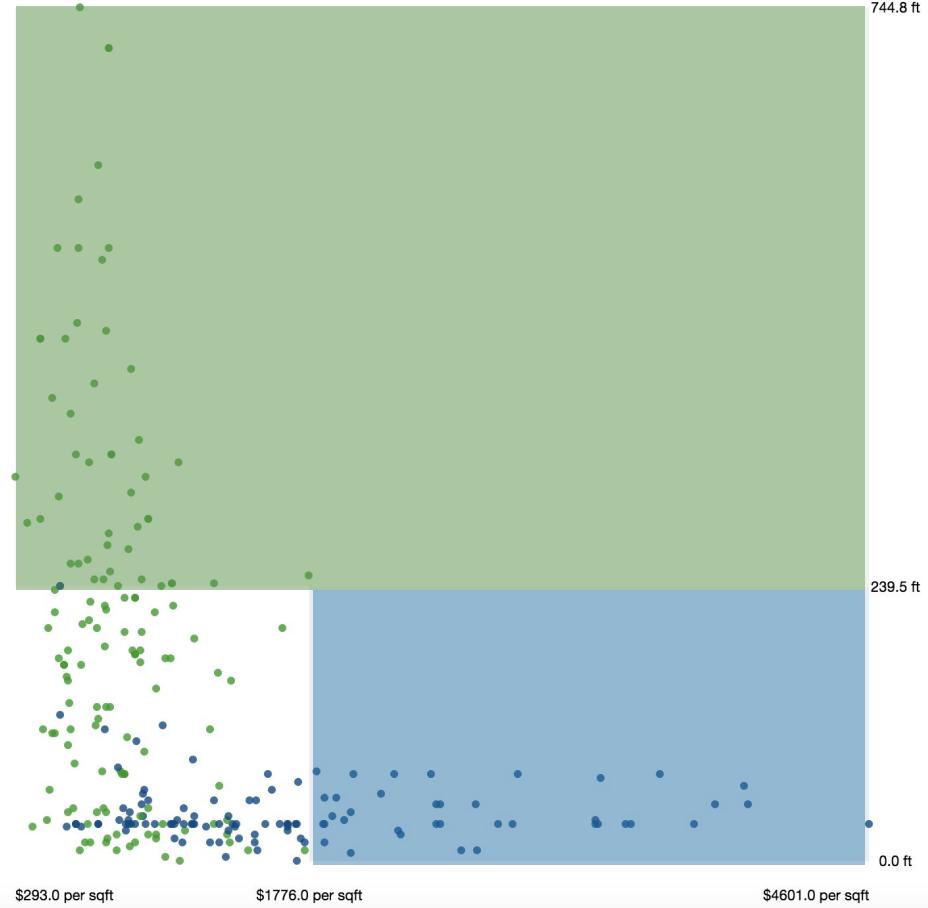
Adding nuance

Elevation isn't a perfect feature for classification, so we can look at its relationship to other features, like *price per square foot*



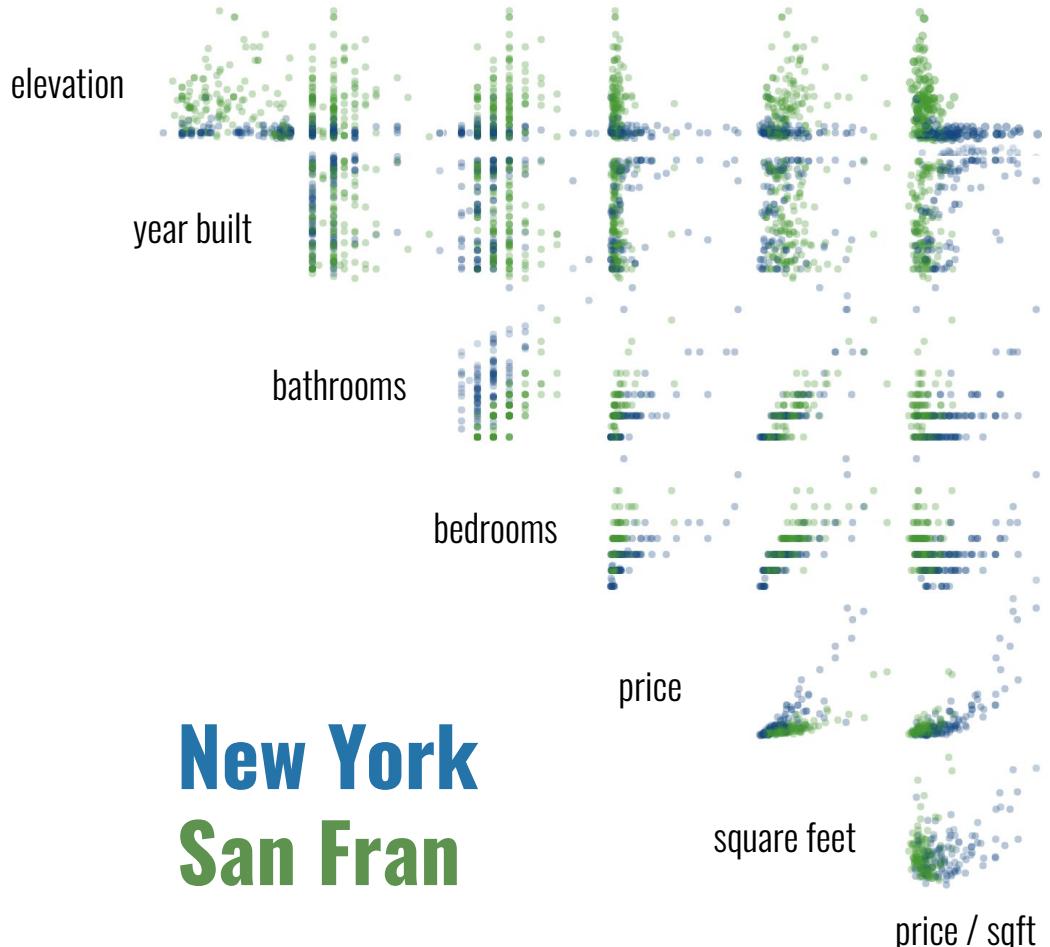
Drawing boundaries

Boundaries can be drawn so that if a house falls in the green box, it's classified as a San Fran home. Blue box, New York. Statistical learning figures out how to best draw these boxes.



Our training set will use 7 different **features**. At the right we see the **scatterplot matrix** of the relationship between these features.

Patterns are clear, but boundaries for delineation are not obvious.



New York
San Fran

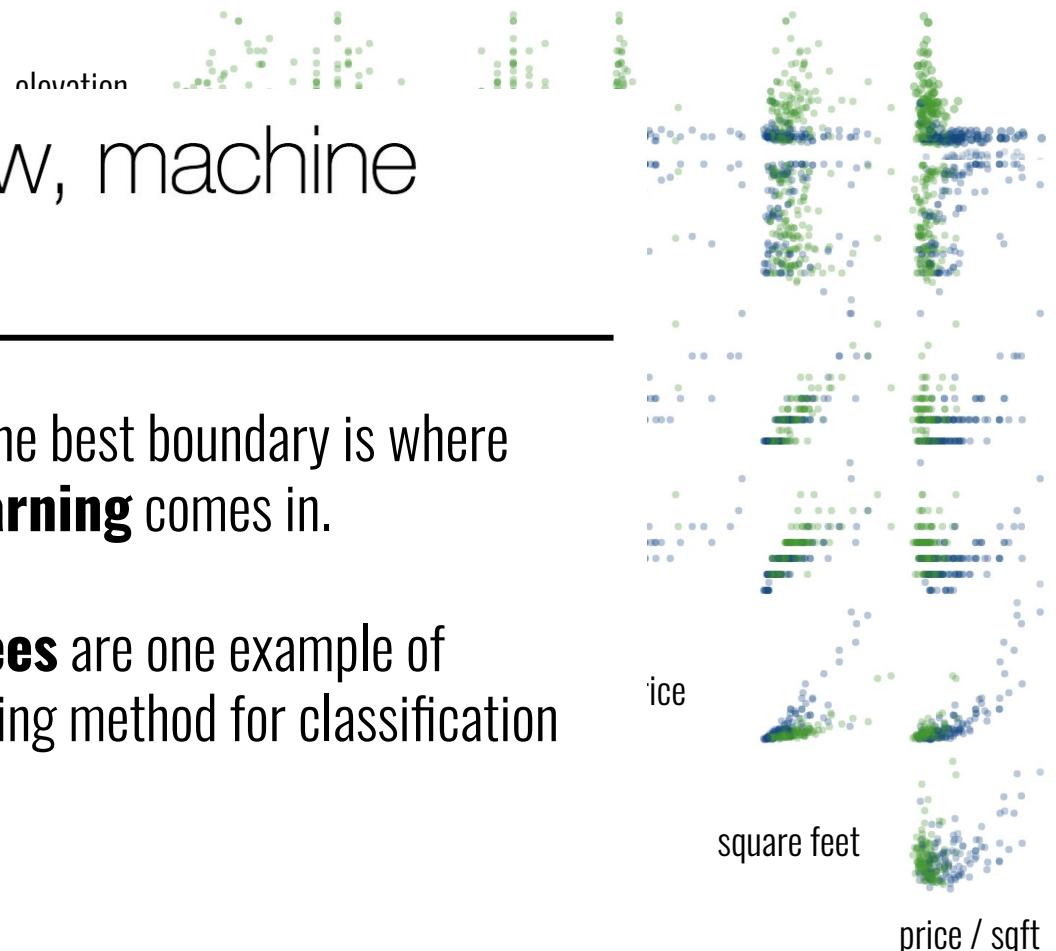
Our training set will use **features**. At the right is a **scatterplot matrix** showing the relationship between

Patterns are clear, but delineation are not obvious.

And now, machine learning

Determining the best boundary is where **machine learning** comes in.

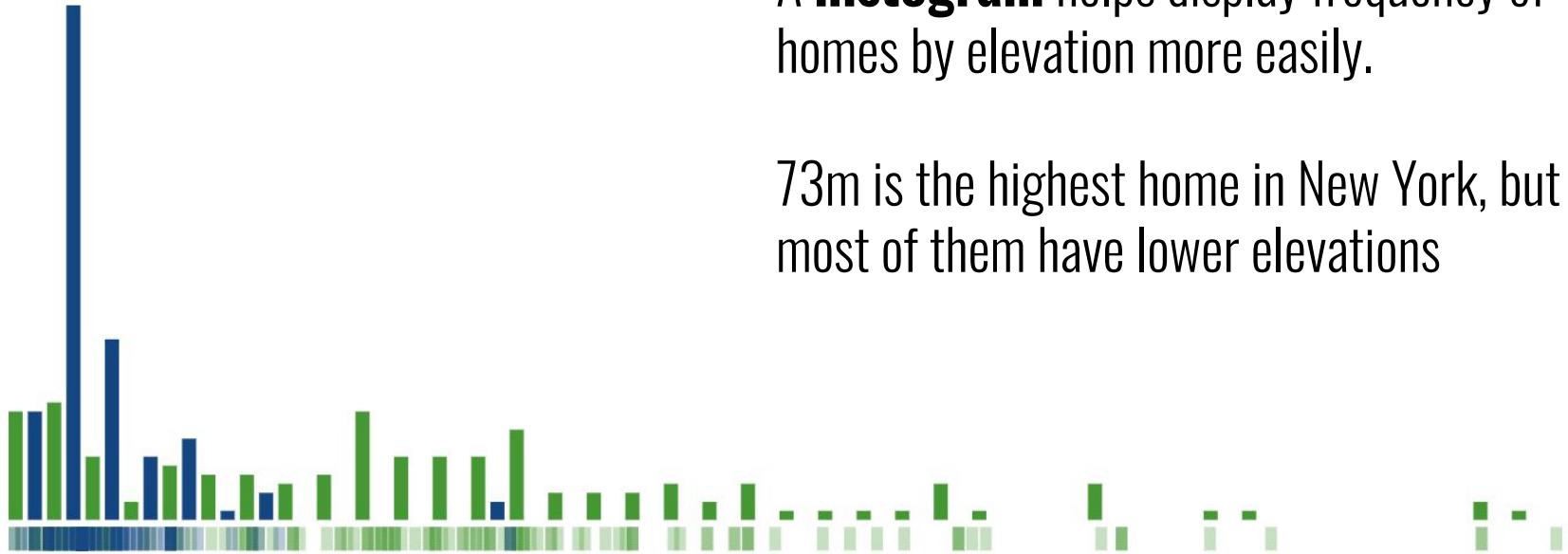
Decision trees are one example of machine learning method for classification tasks.





Finding better boundaries

We guessed ~73m before. Let's improve on that guess...



A **histogram** helps display frequency of homes by elevation more easily.

73m is the highest home in New York, but most of them have lower elevations

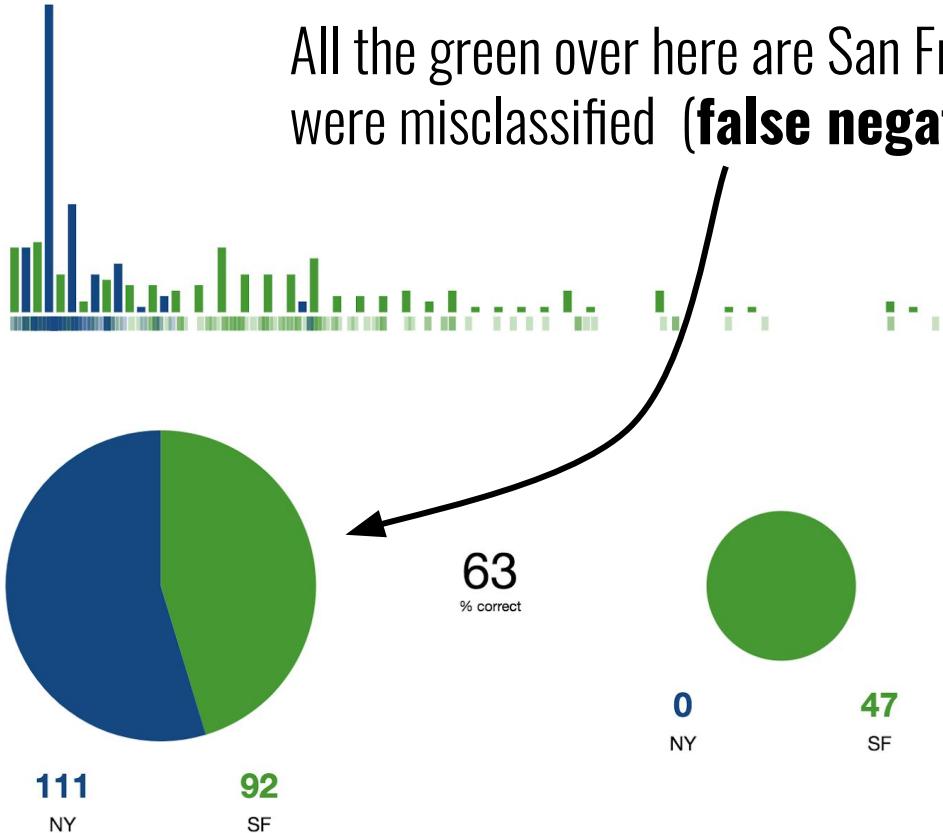
In machine learning, the splits are called **forks** and they split the data into **branches** based on some value.

The value that splits the branches is the **split point**. Homes to the left get categorized differently than those on the right.



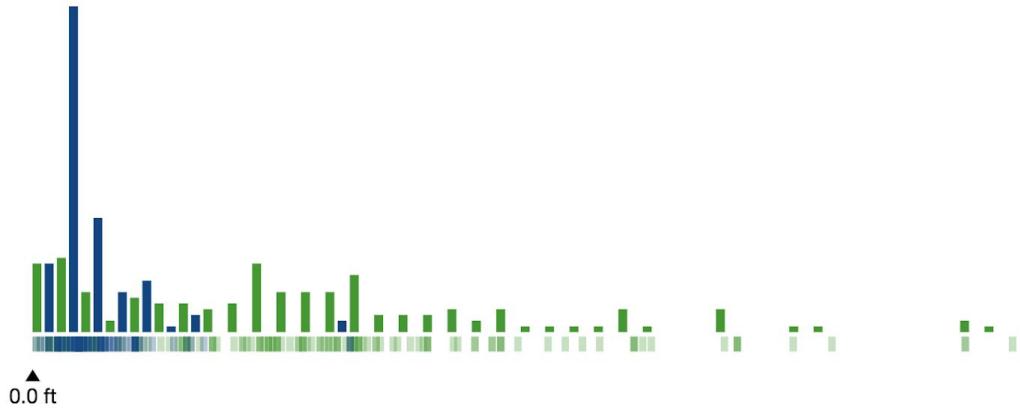
Your first fork

A decision tree uses if-then statements to define patterns in the data.



Tradeoffs

Splitting at ~73m incorrectly classifies some San Francisco homes as New York homes.

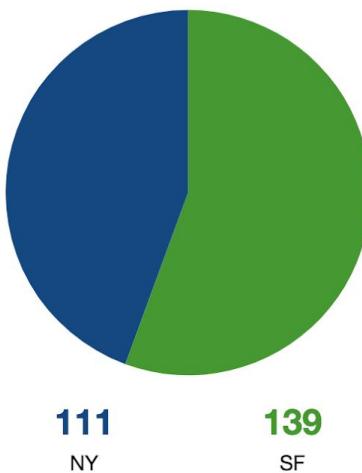


0.0 ft

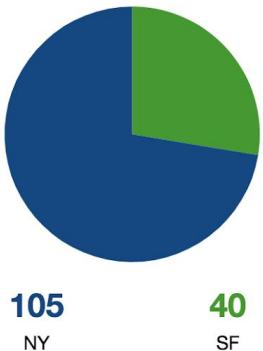
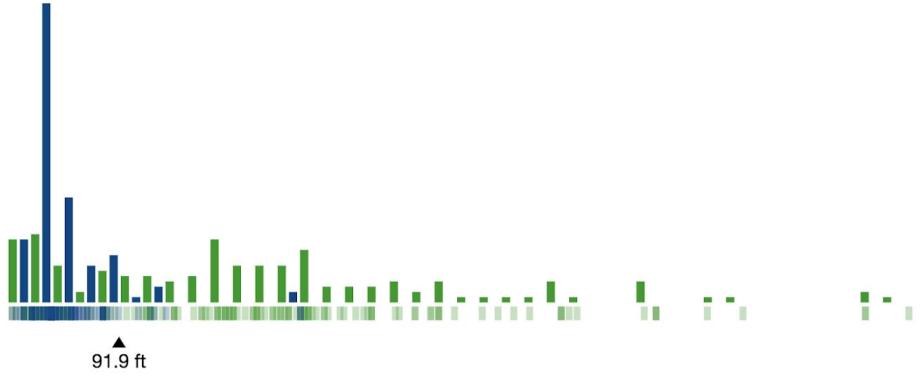
0
NY

0
SF

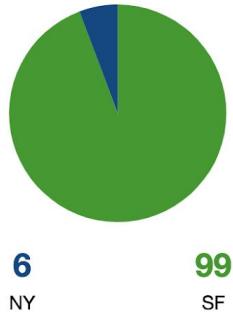
56
% correct



If you split to capture *every* home in San Fran, you'll also get a bunch of New York homes (**false positives**)

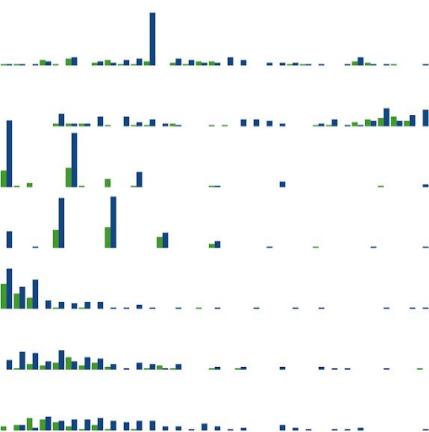
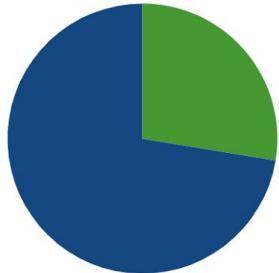


82
% correct

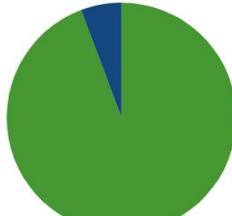


The best split

The best split point aims for branches that are as homogenous (pure) as possible

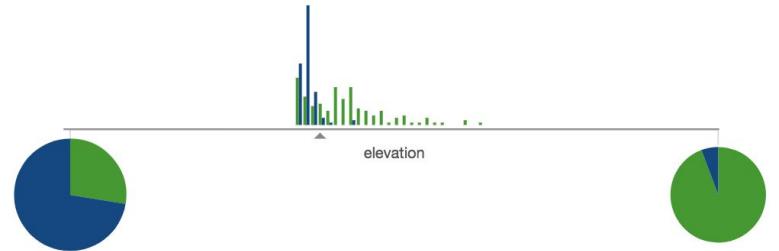


82
% correct



Recursion

Additional split points are determined through repetition (**recursion**)



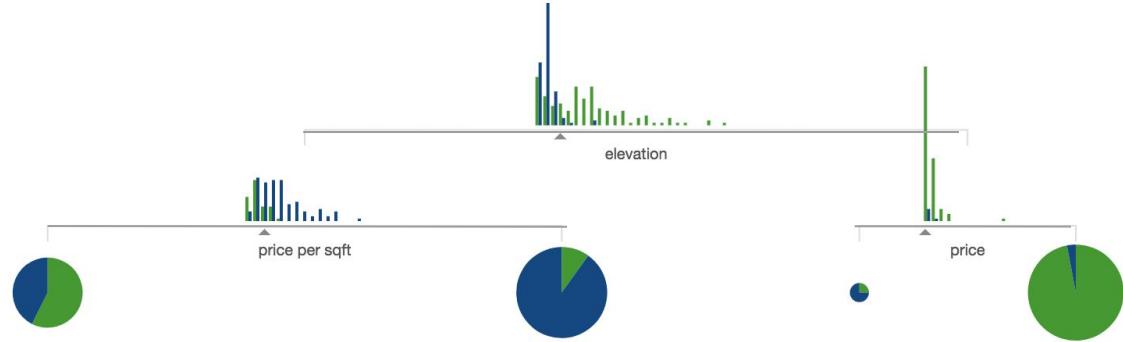
Growing a tree

Additional forks add new
information to improve
prediction accuracy.

Accuracy: 82%

Growing a tree

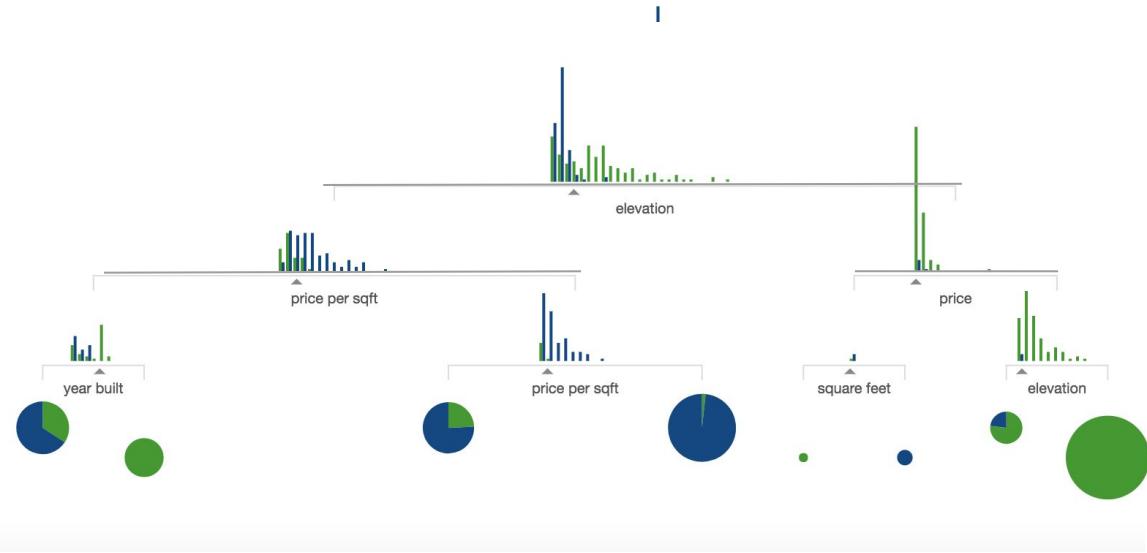
Additional forks add new
information to improve
prediction accuracy.



Accuracy: 86%

Growing a tree

Additional forks add new information to improve
prediction accuracy.



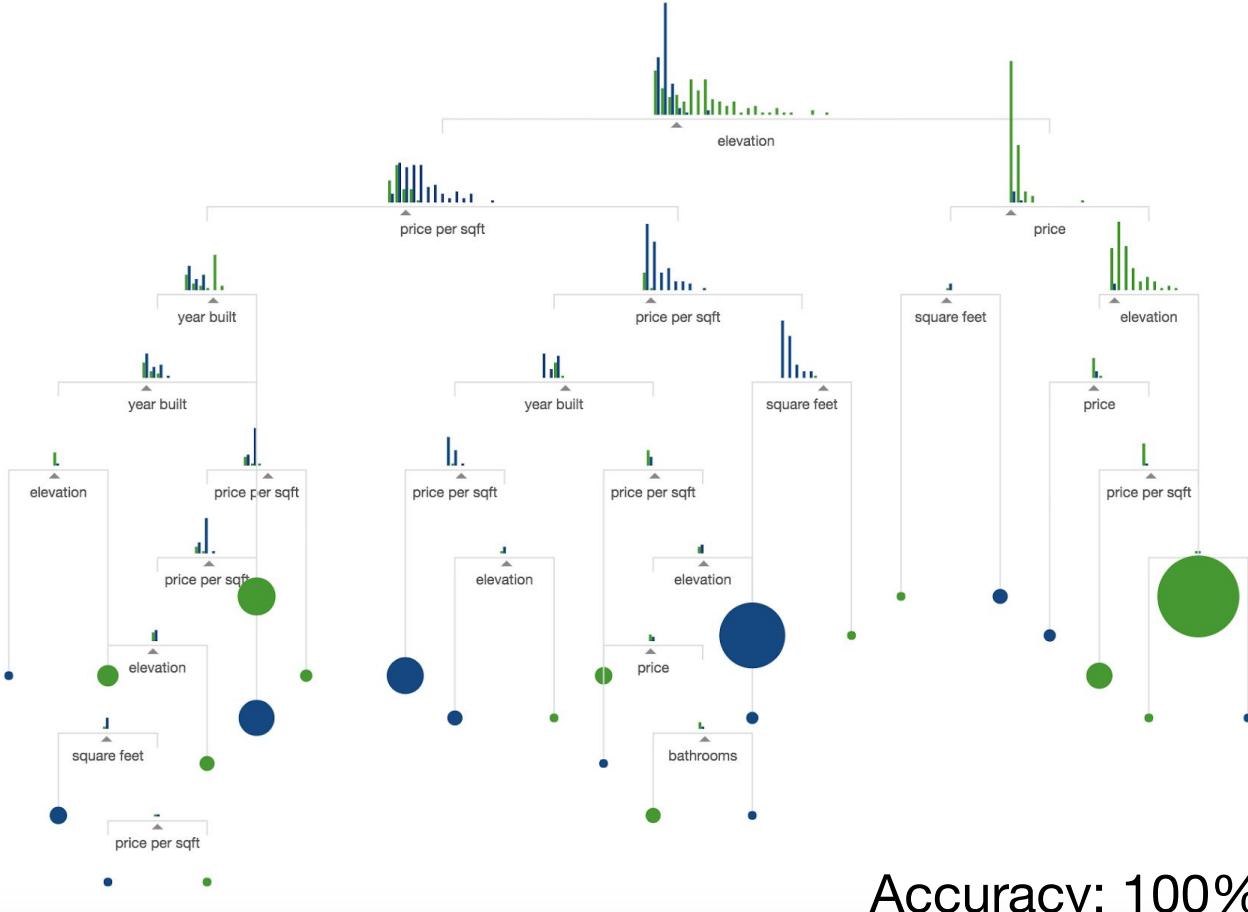
Adding several more layers, we get to

96%.



Accuracy: 96%

It's possible to add branches until your model is **100% accurate.**

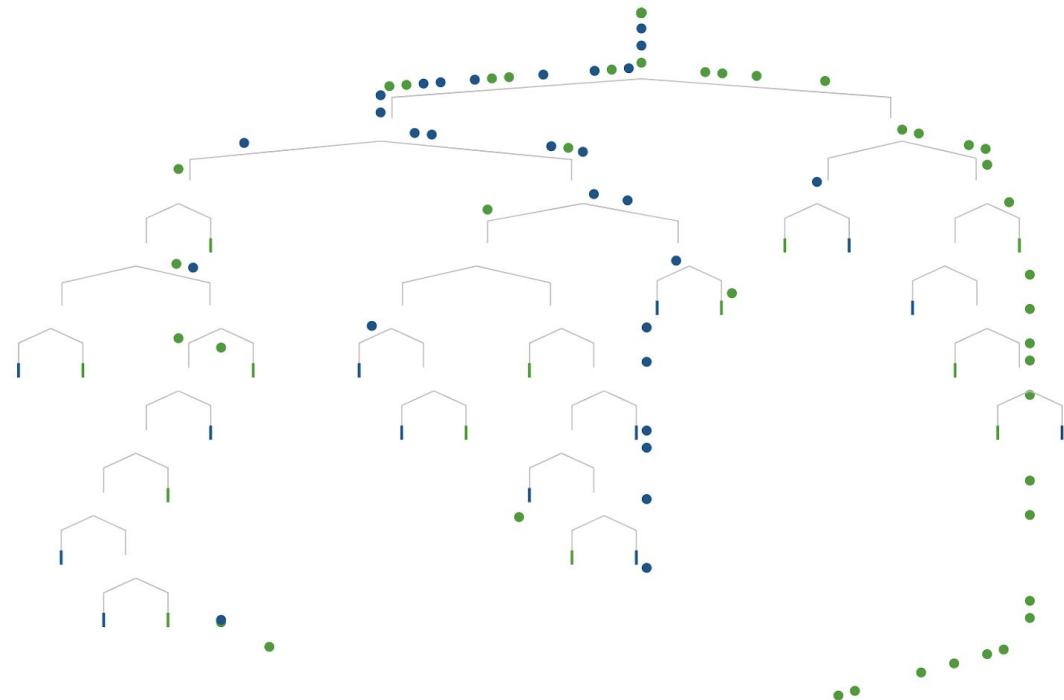


Accuracy: 100%

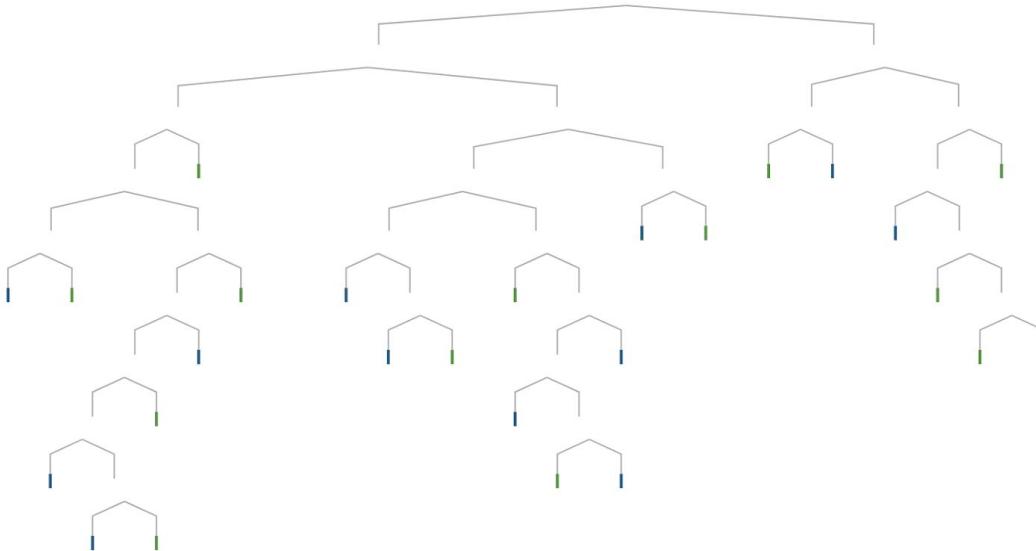
Making predictions

The decision tree **model** can then predict which homes are in which city.

Here, we're using the **training data**.



Because our tree was trained on this data and we grew the tree to 100% accuracy, each house is perfectly sorted



111/111

Training Accuracy
100%

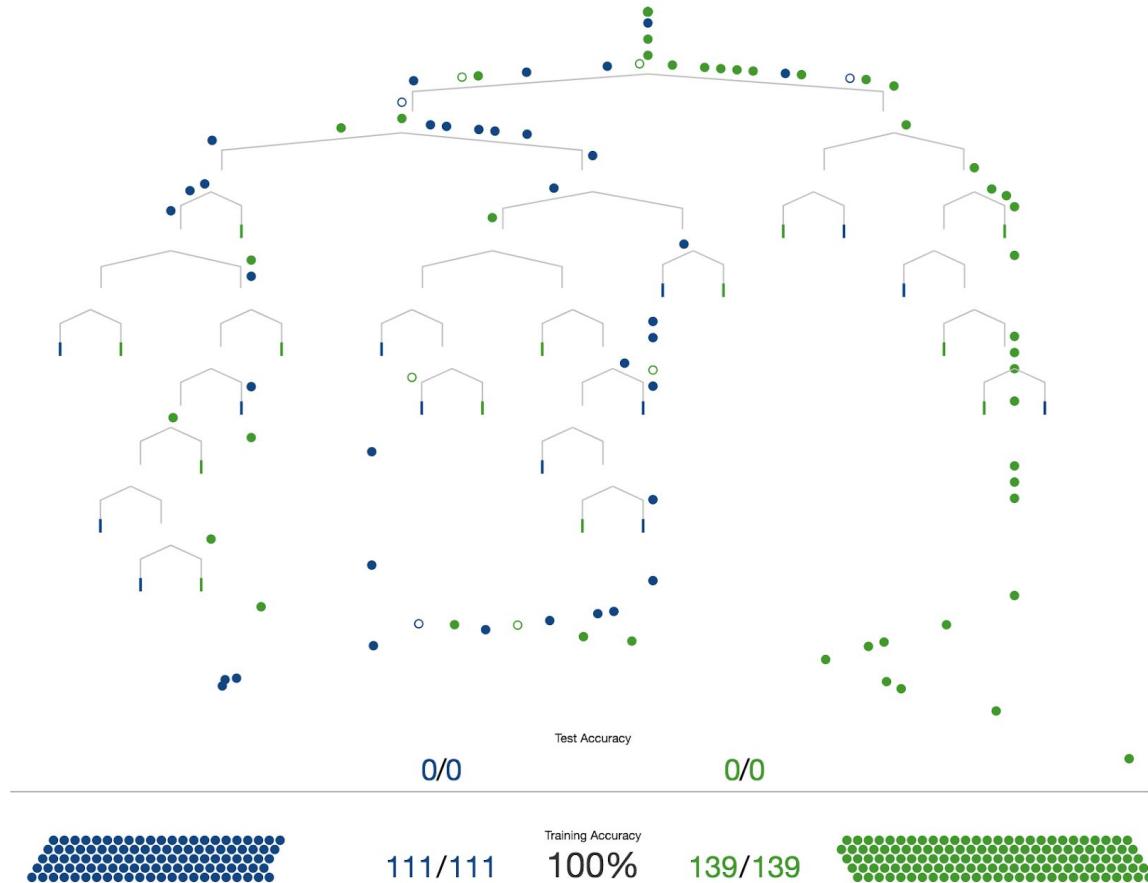
139/139



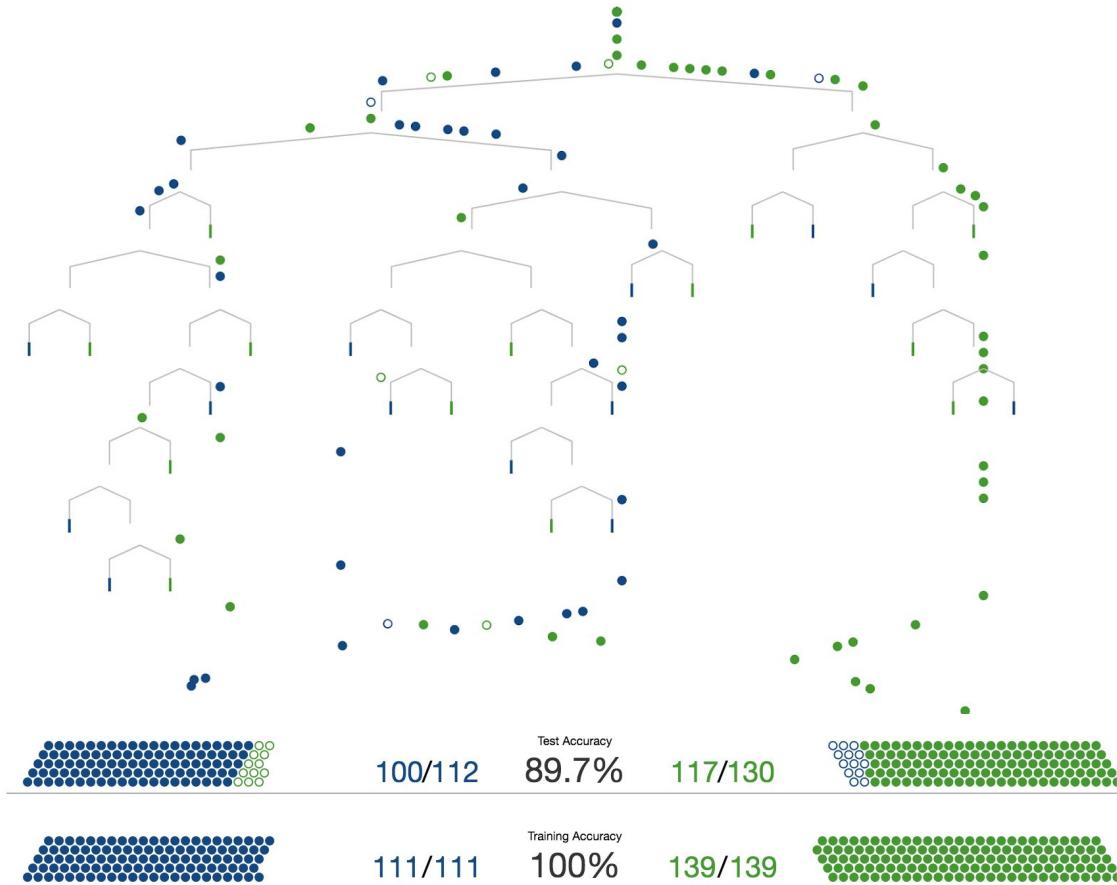
Reality check

But...how does this tree
on data that the model
hasn't seen before?

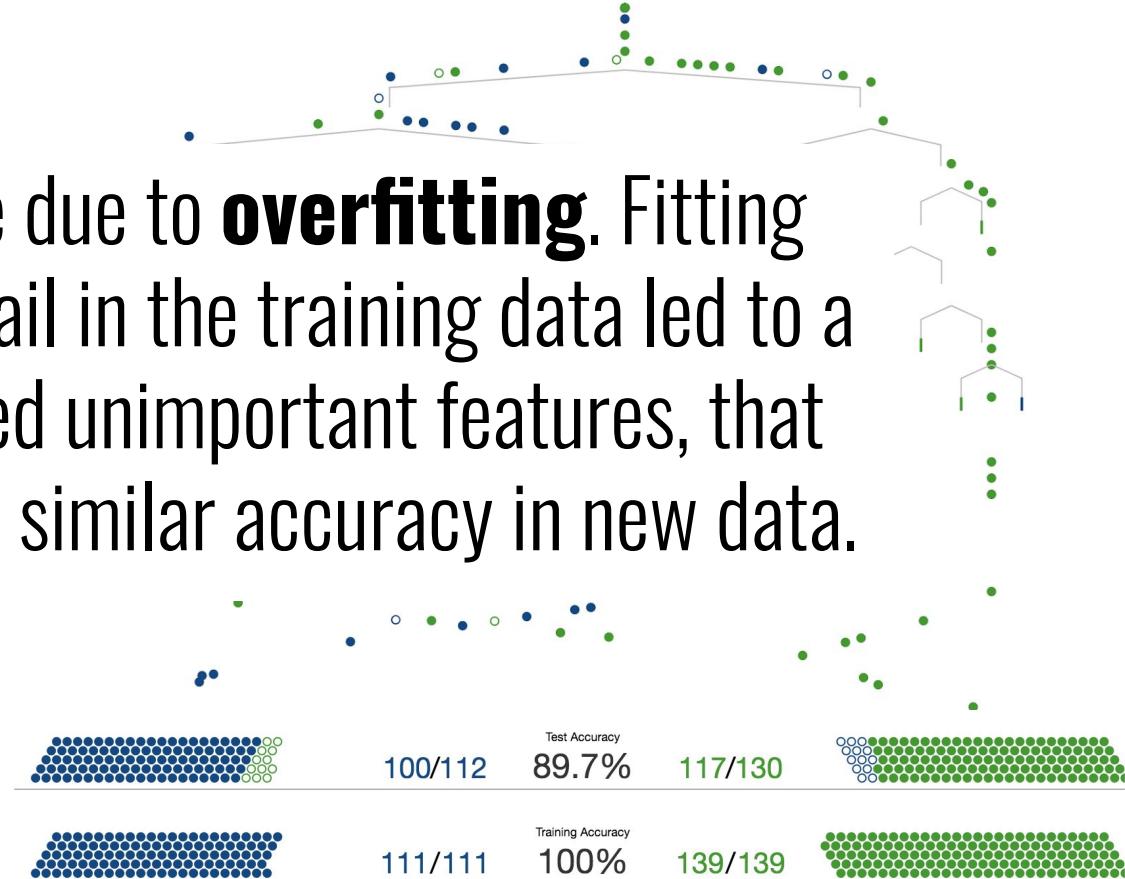
The **test set** then
makes it way through
the decision tree.



Ideally the tree should perform similarly on both known and unknown data



These errors are due to **overfitting**. Fitting every single detail in the training data led to a tree that modeled unimportant features, that did not allow for similar accuracy in new data.

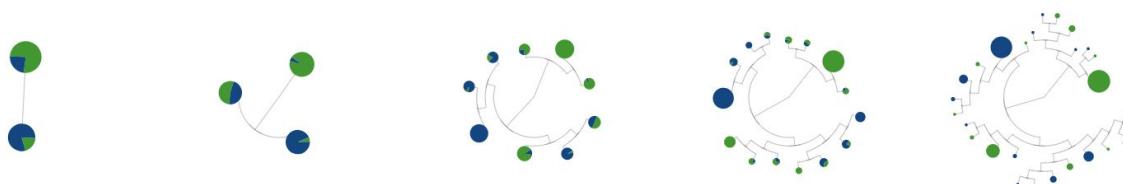
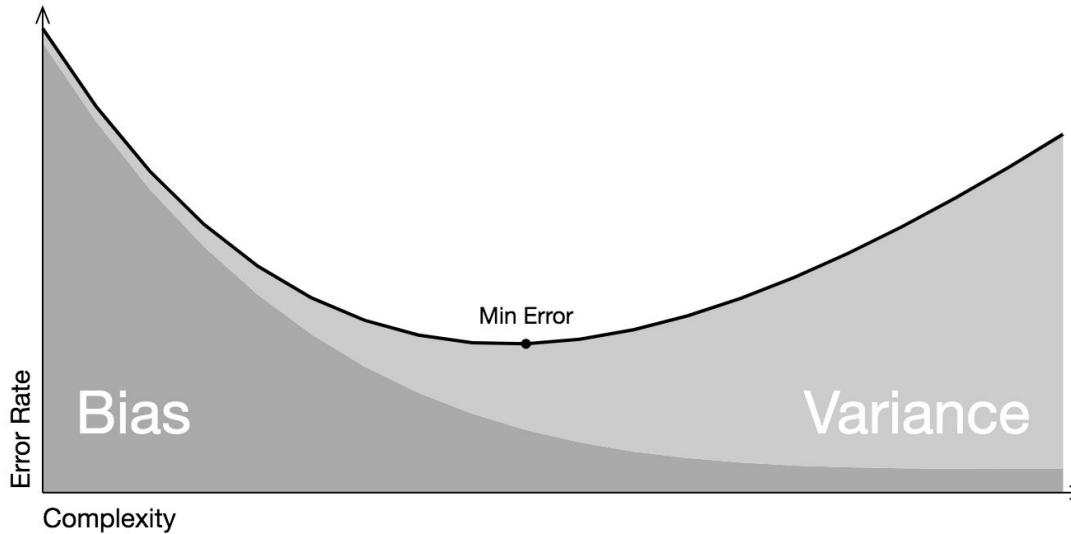


Recap

1. Machine learning identifies patterns using **statistical learning** and computers by unearthing **boundaries** in data sets. You can use it to make predictions.
2. One method for making predictions is called a decision trees, which uses a series of if-then statements to identify boundaries and define patterns in the data.
3. **Overfitting** happens when some boundaries are based on *on distinctions that don't make a difference*. You can see if a model overfits by having test data flow through the model.

So...what can we do
about overfitting?

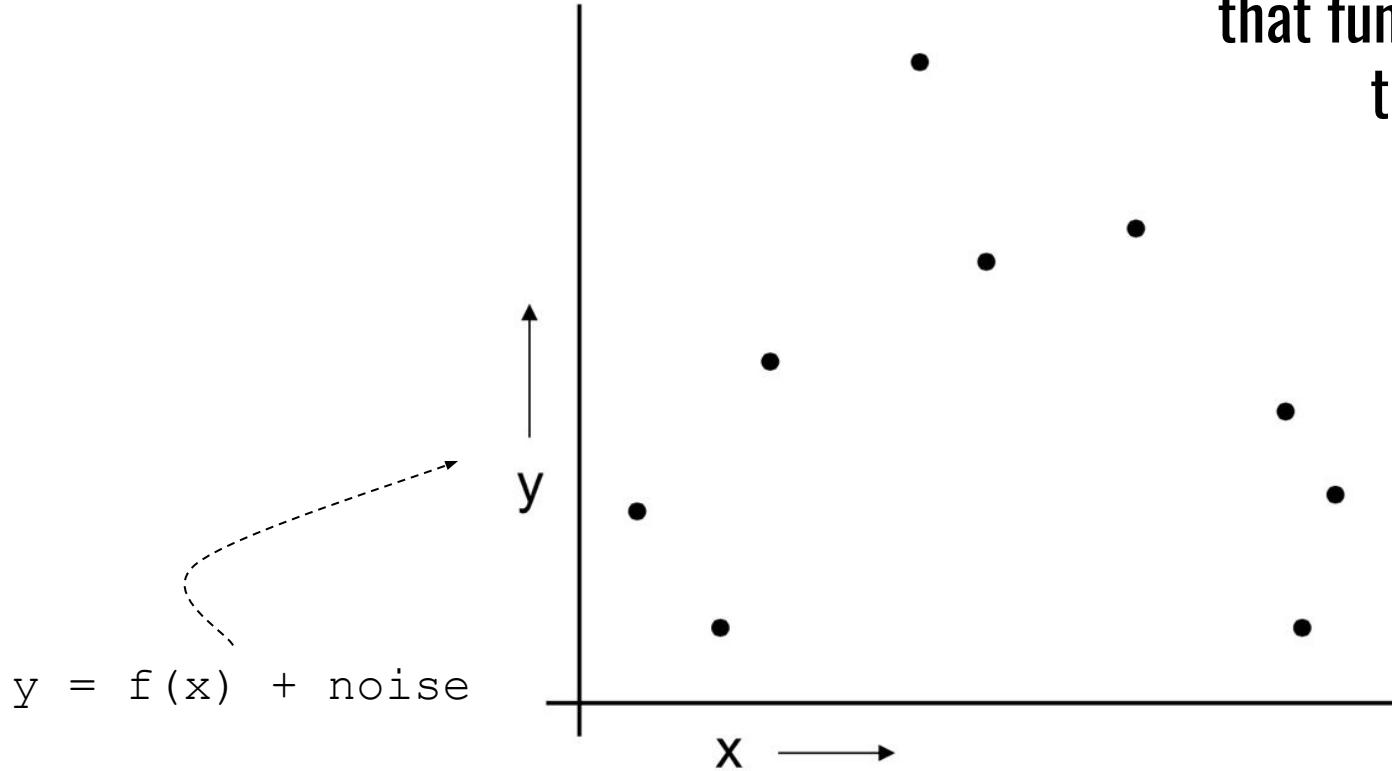
Bias-variance tradeoff



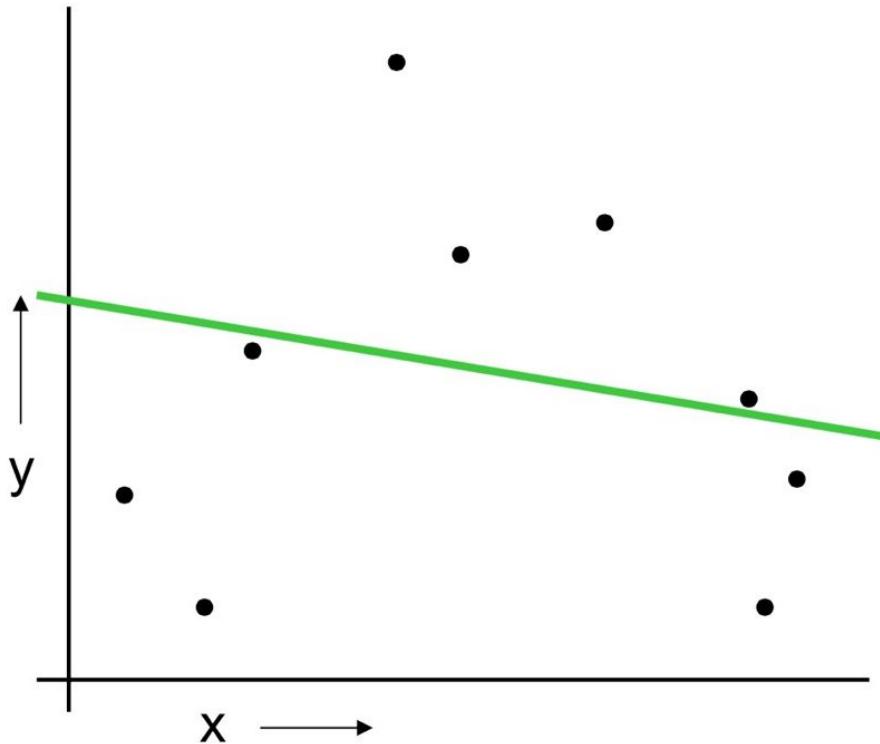
Bias-variance tradeoff

- **High variance** models make mistakes in *inconsistent* ways.
- **Biased models** tend to be overly simple and not reflect reality
- What to do:
 - Consider tuning parameters in the model
 - can avoid overfitting by setting minimum node size threshold (fewer splits; variance decreased)
 - Changing model approach
 - Bagging, boosting, & ensemble methods
 - Re-consider data splitting approach
 - Training + test?
 - LOOCV
 - K-fold CV

Can we determine what
that function (f) *is* using
these data?

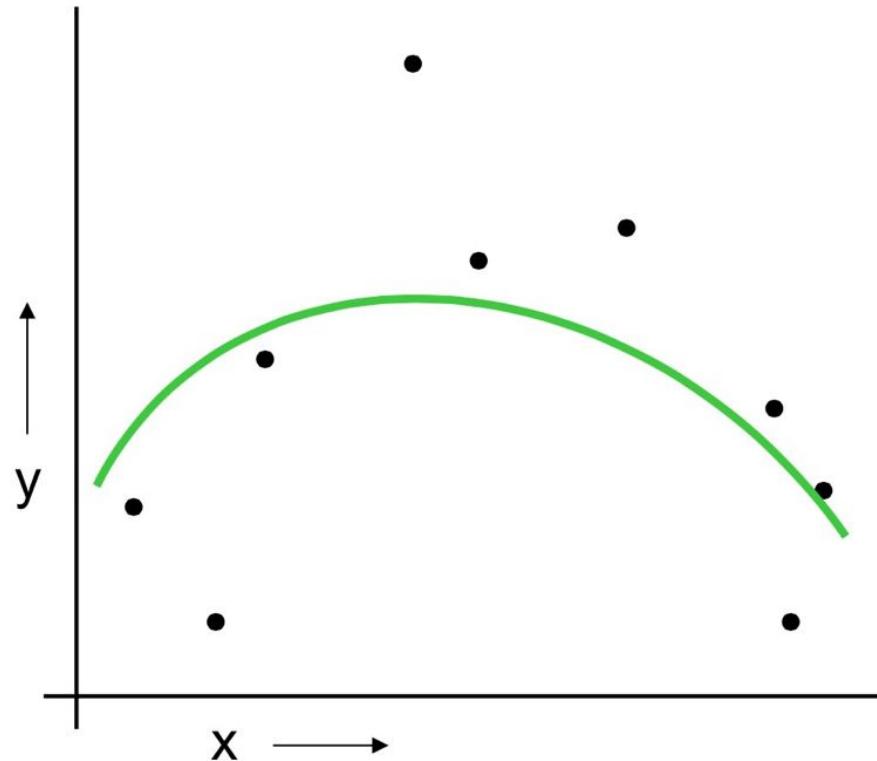


Linear regression



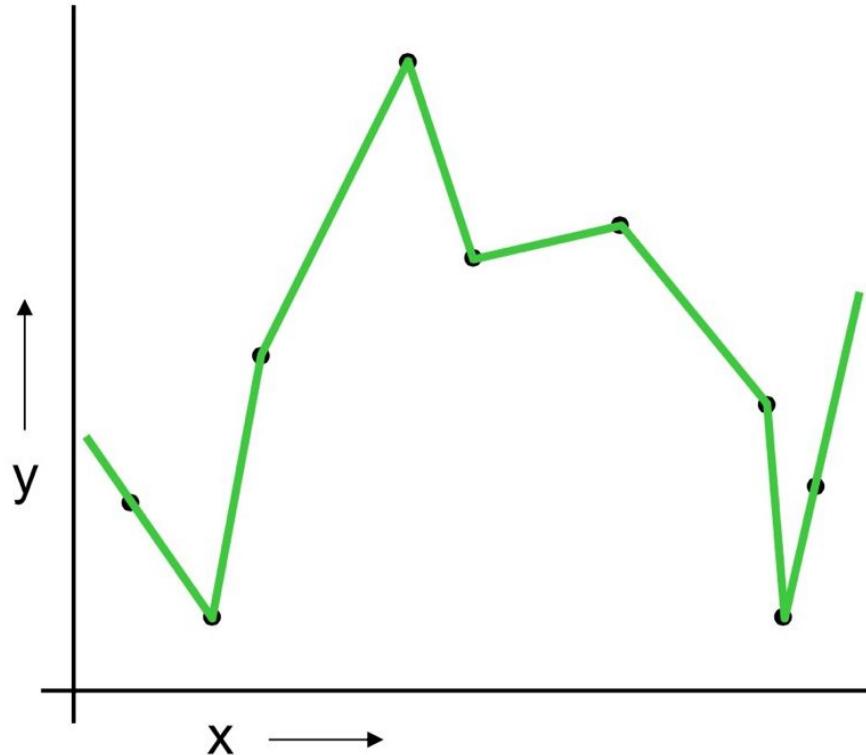
adapted from Brad Voytek

Quadratic regression



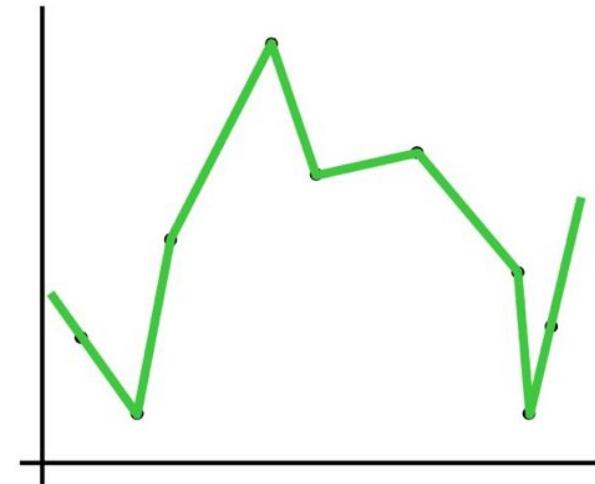
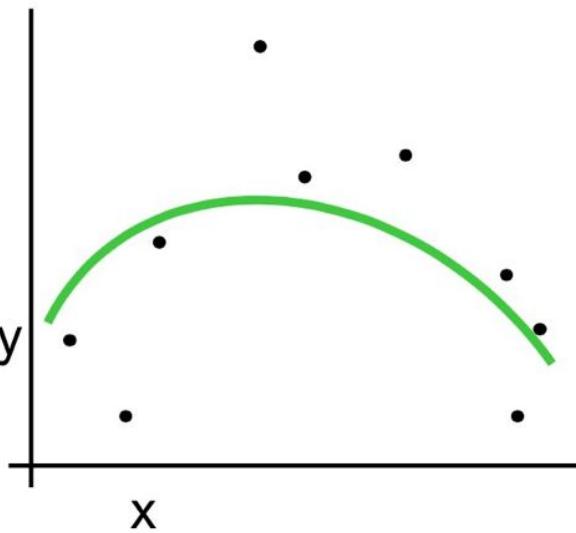
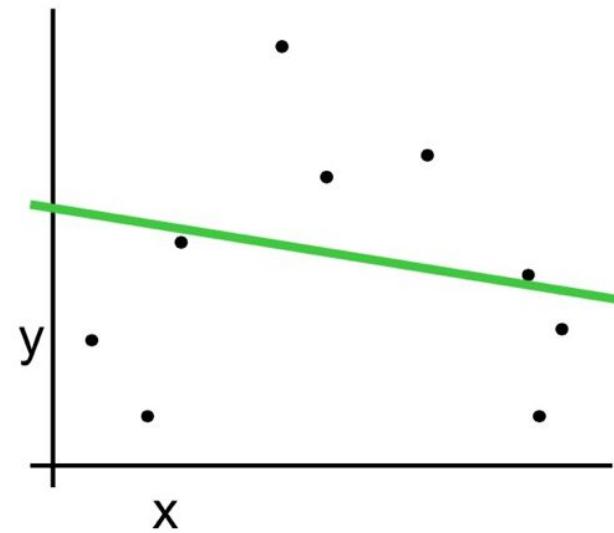
adapted from Brad Voytek

Piecewise linear nonparametric regression



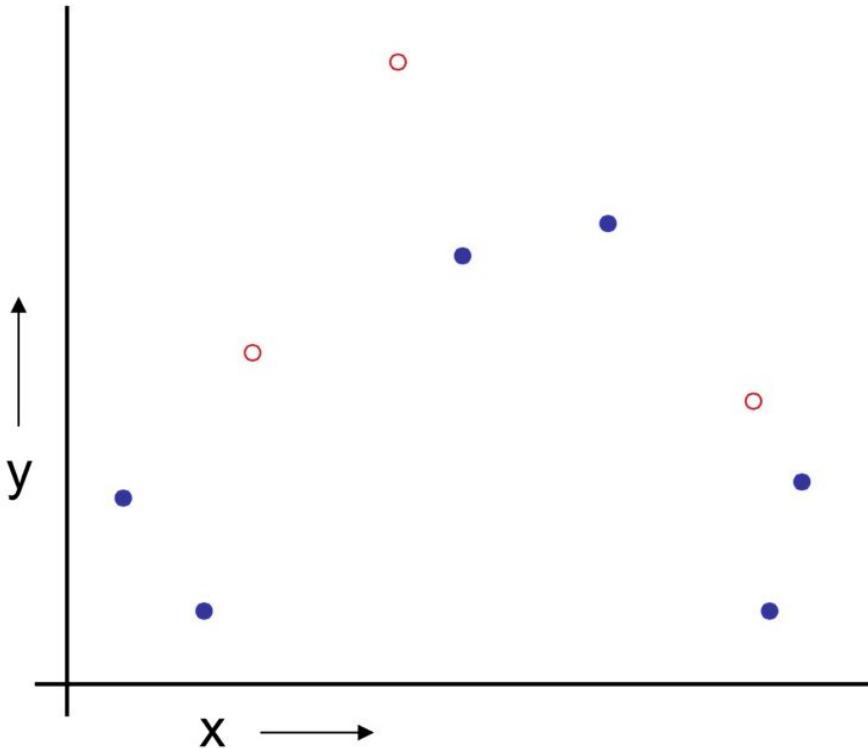
adapted from Brad Voytek

Which to choose?



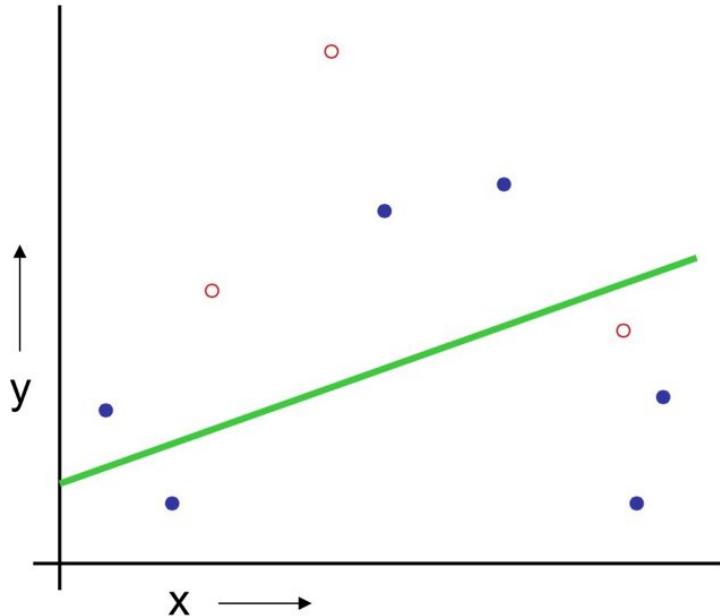
adapted from Brad Voytek

The data partition method



1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**

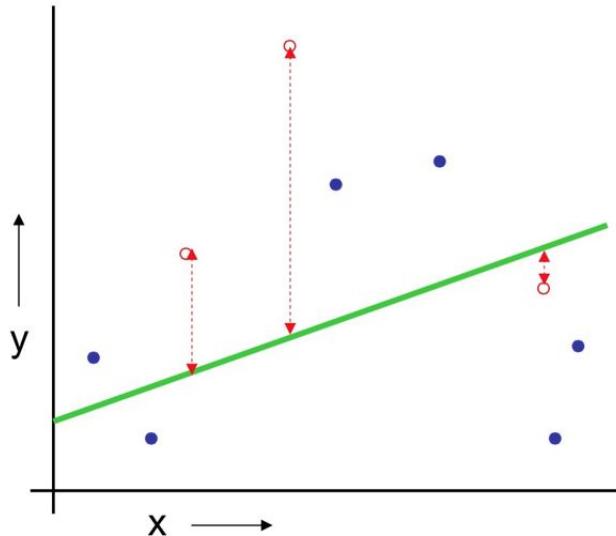
Train the model on your training set



1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**
3. Perform your regression on the training set

(Linear regression example)

Assess future performance using the **test set**



(Linear regression example)

Mean Squared Error = 2.4

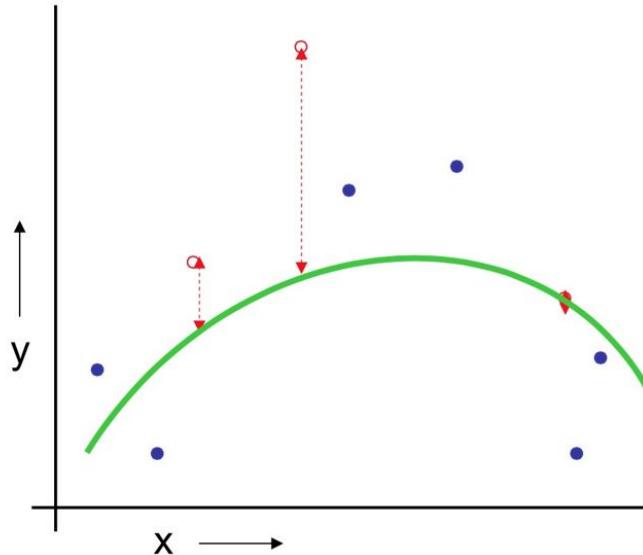
1. Randomly choose
30% of the data to be in a
test set

2. The remainder is a
training set

3. Perform your
regression on the training
set

4. Estimate your future
performance with the test
set

Go through this process for each possible model

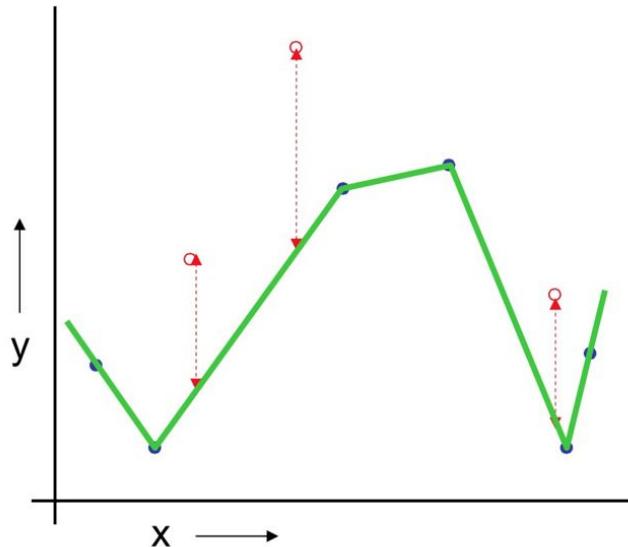


(Quadratic regression example)

Mean Squared Error = 0.9

1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**
3. Perform your regression on the training set
4. Estimate your future performance with the test set

Go through this process for each possible model



(Join the dots example)

Mean Squared Error = 2.2

1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a training set
3. Perform your regression on the training set
4. Estimate your future performance with the test set

Pros and cons of data partitioning

Pros:

- Simple approach
- Can choose model with best test-set score

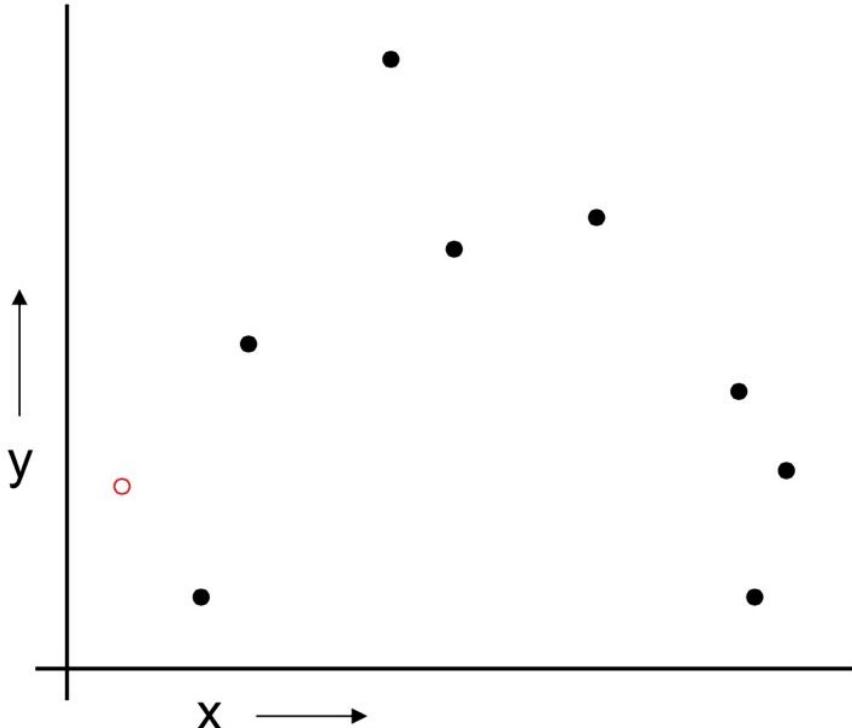
Cons:

- Model fit on 30% less data than you have
- Without a large data set, removing 30% of the data could bias prediction

Leave one out cross validation (LOOCV)

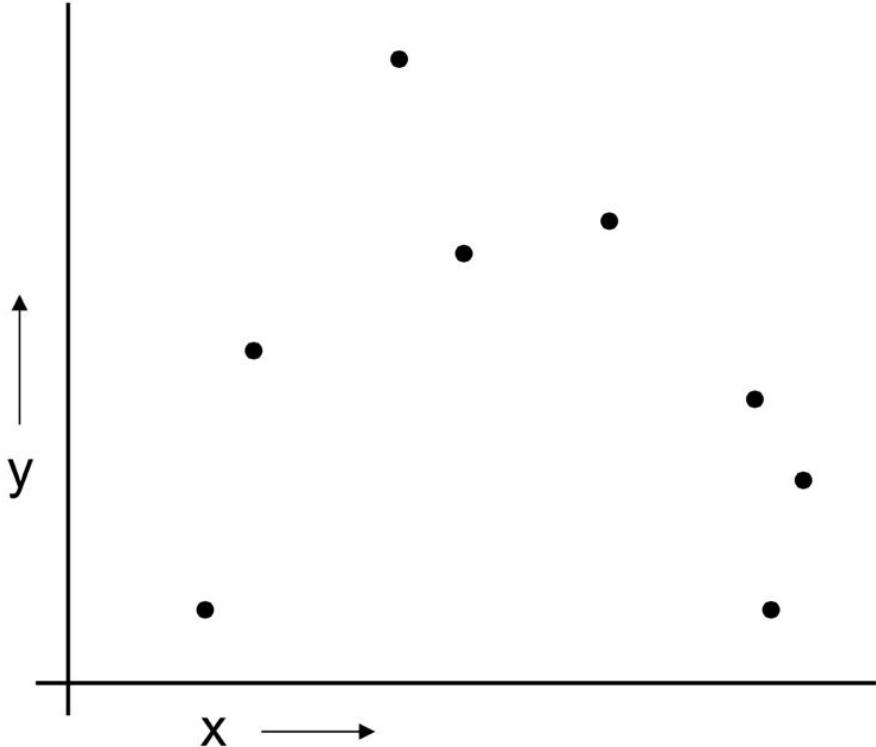
For k=1 to R

1. Let (x_k, y_k) be the k^{th} record



Leave one out cross validation (LOOCV)

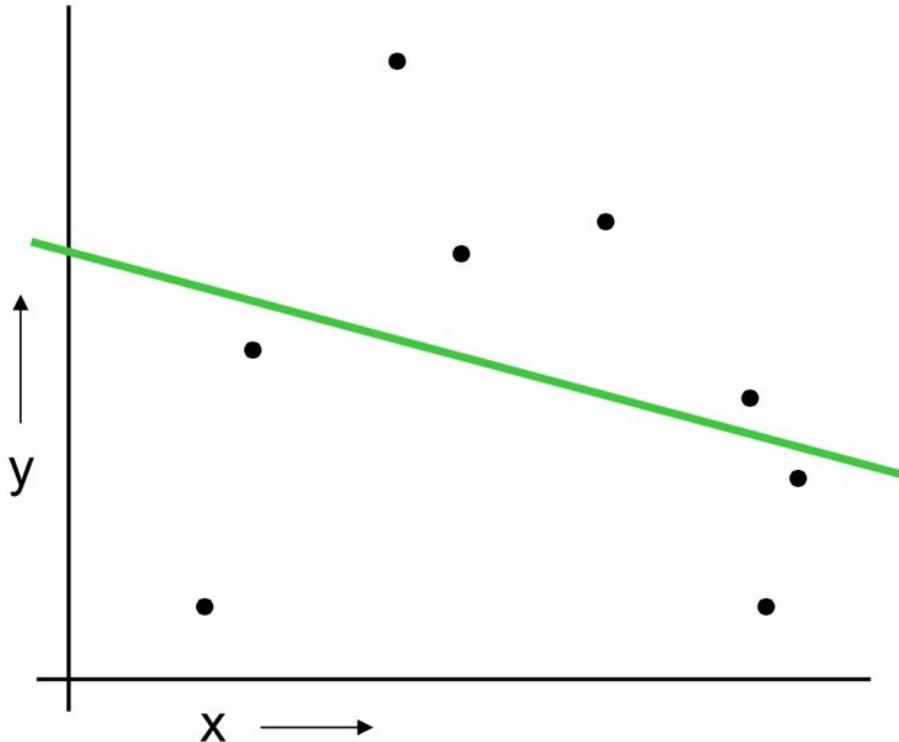
For k=1 to R



1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset

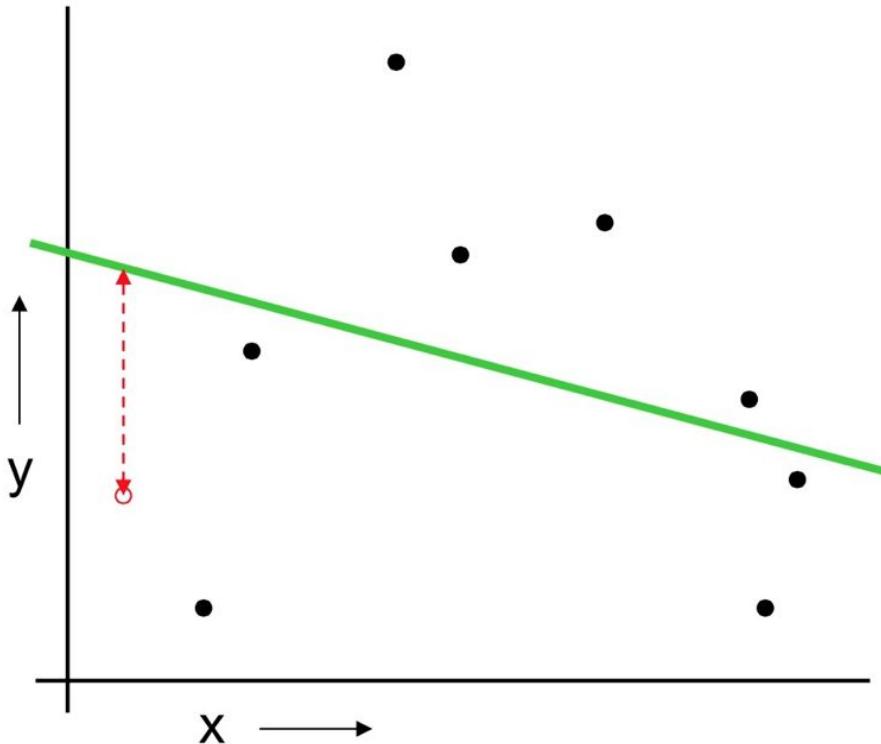
Leave one out cross validation (LOOCV)

For k=1 to R



1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints

Leave one out cross validation (LOOCV)

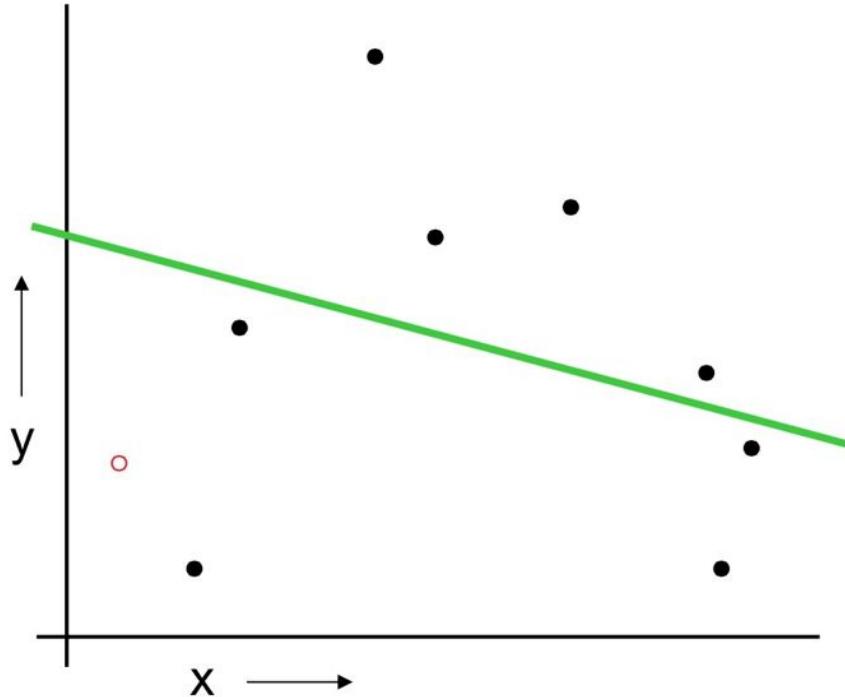


For k=1 to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining R-1 datapoints
4. Note your error (x_k, y_k)

Leave one out cross validation (LOOCV)

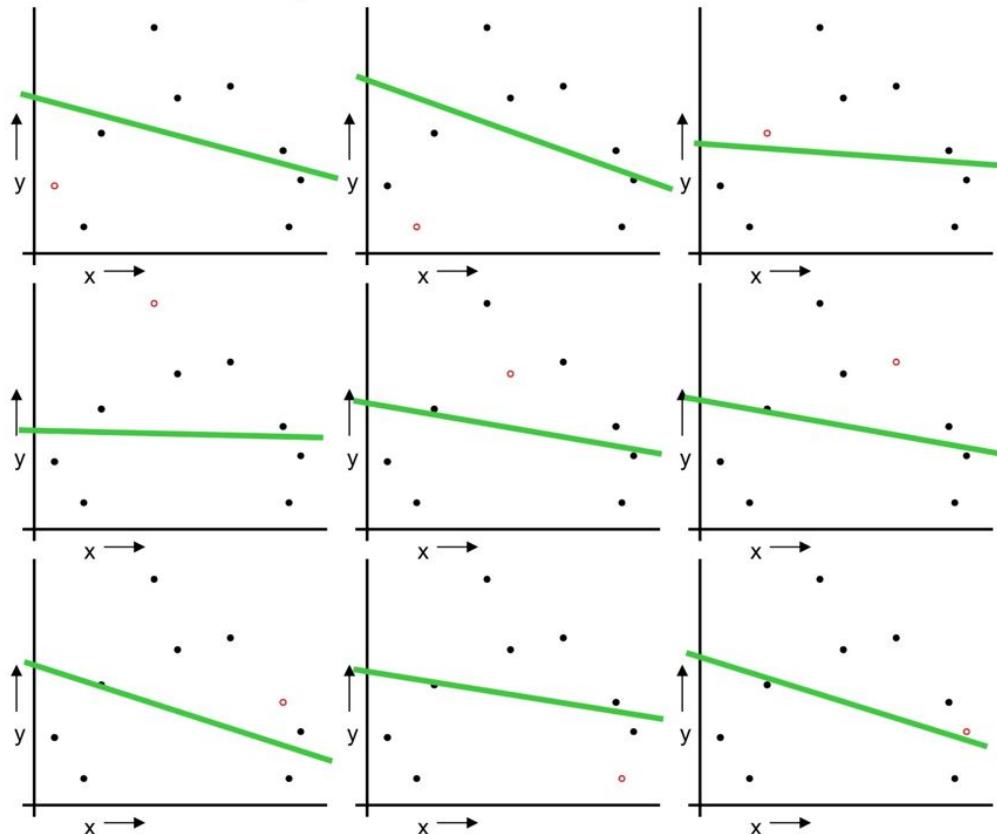
For k=1 to R



1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

When you've done all points,
report the mean error.

Leave one out cross validation (LOOCV)



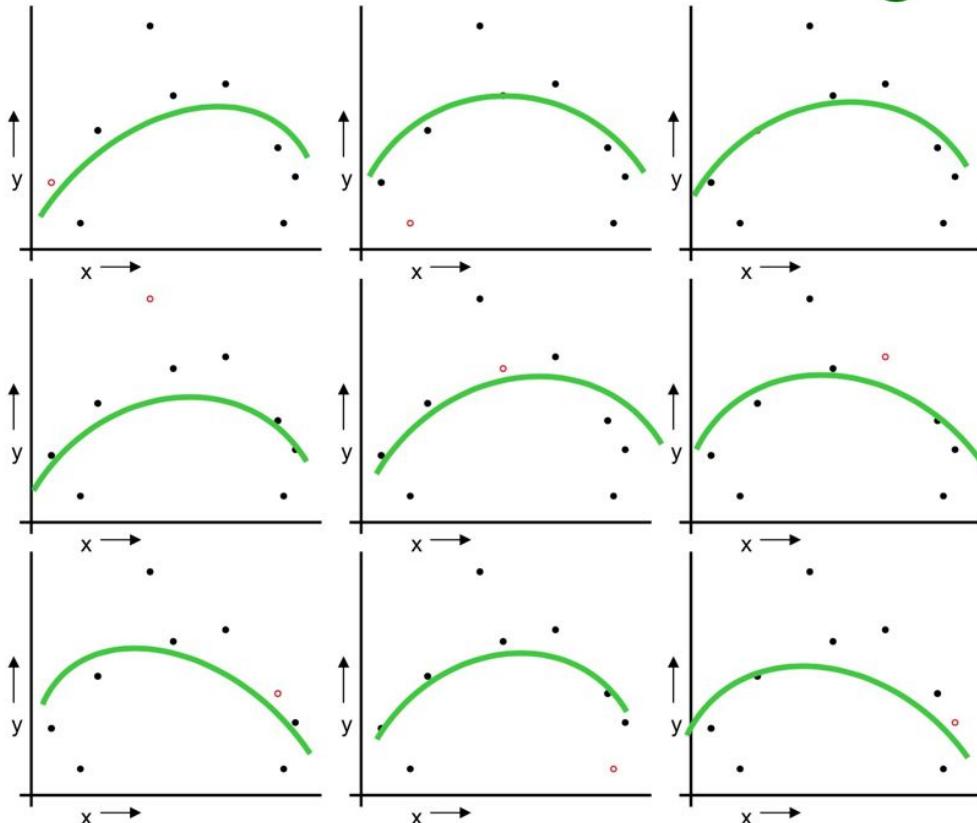
For k=1 to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

When you've done all points, report the mean error.

$$MSE_{\text{LOOCV}} = 2.12$$

Leave one out cross validation (LOOCV)



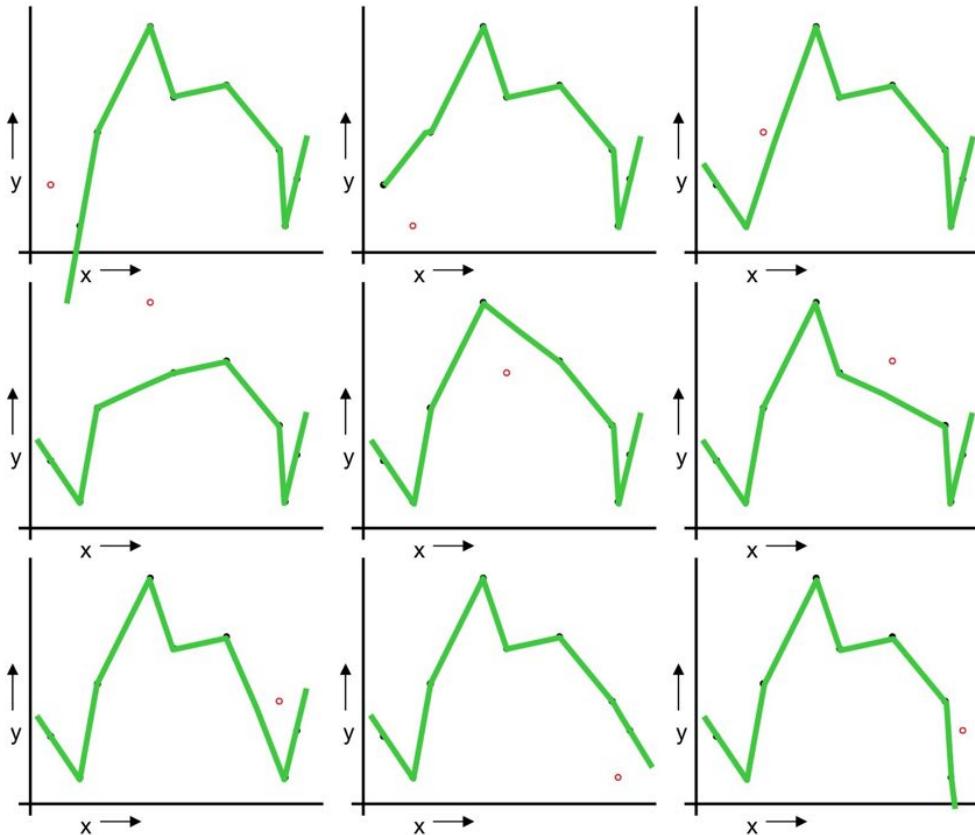
For k=1 to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

When you've done all points, report the mean error.

$$MSE_{LOOCV} = 0.962$$

Leave one out cross validation (LOOCV)



For k=1 to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

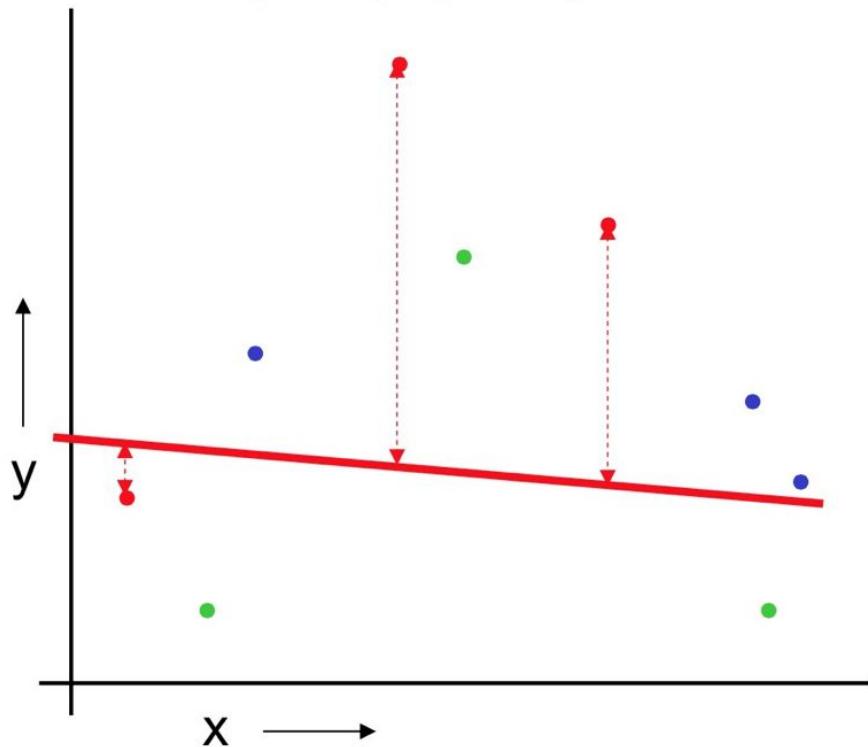
When you've done all points, report the mean error.

$$MSE_{\text{LOOCV}} = 3.33$$

Method Comparison

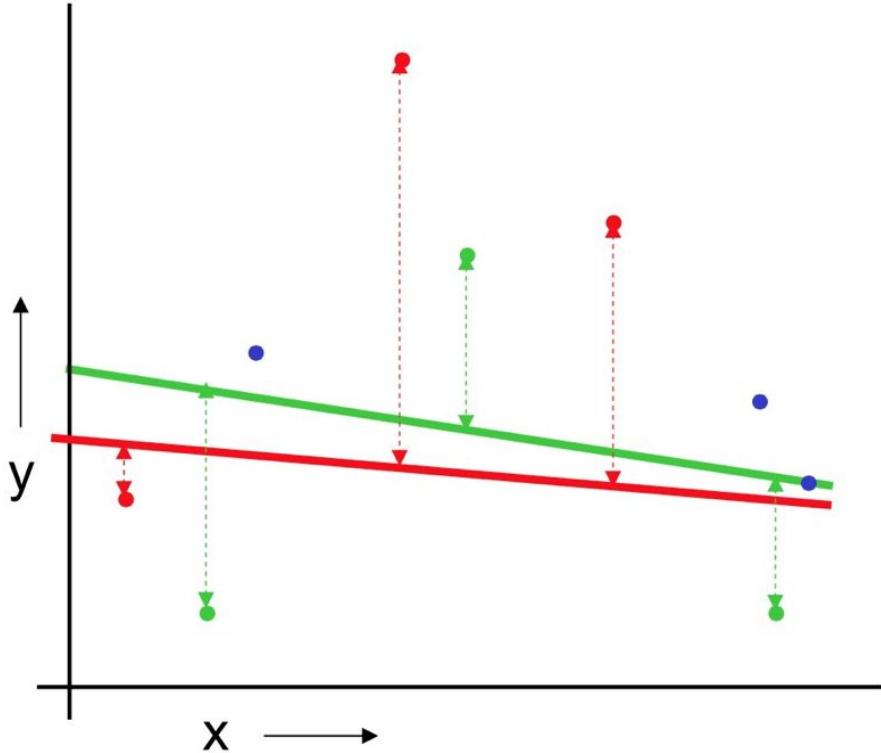
	Cons	Pros
Data partitioning	Variance: unreliable estimate of future performance	Cheap
LOOCV	Computationally expensive; has weird behavior	Uses all your data

k -fold cross validation



For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

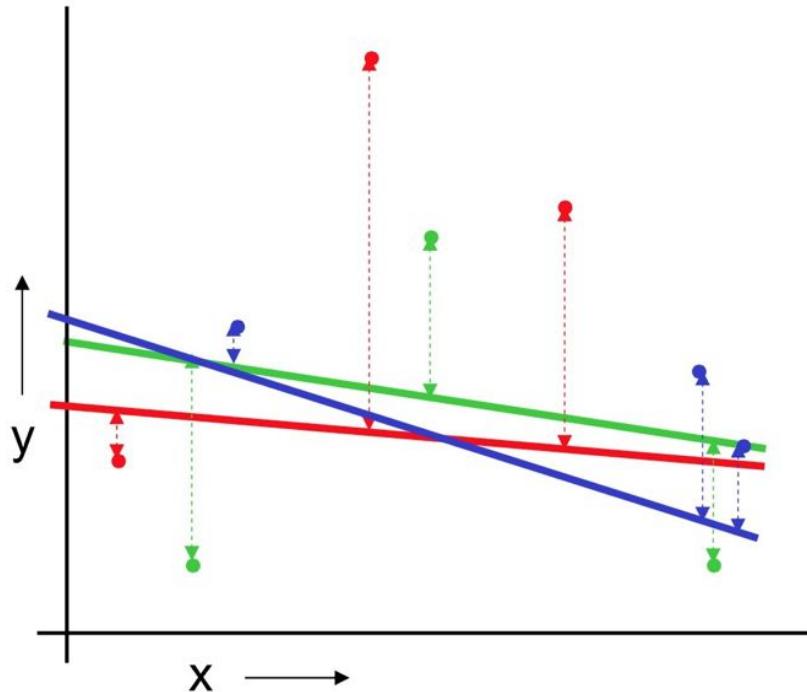
k -fold cross validation



For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

k -fold cross validation

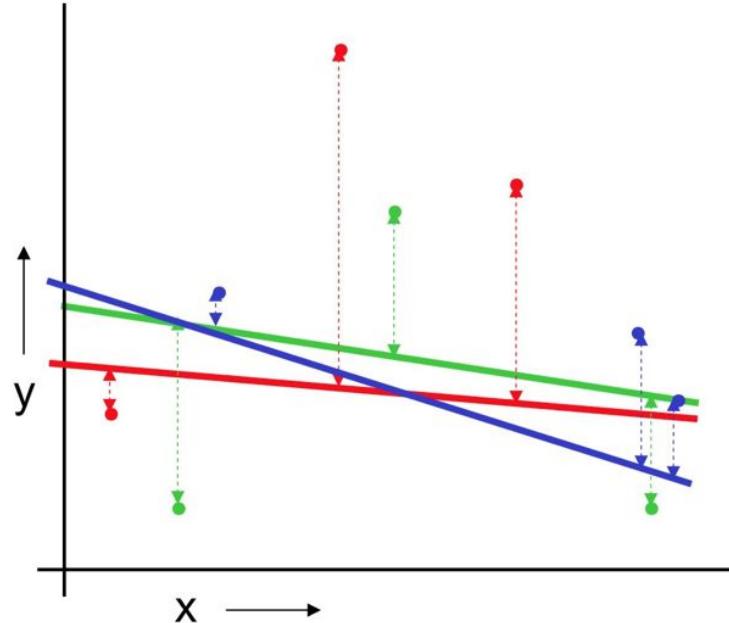


For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

k -fold cross validation



Linear Regression

$$MSE_{3FOLD} = 2.05$$

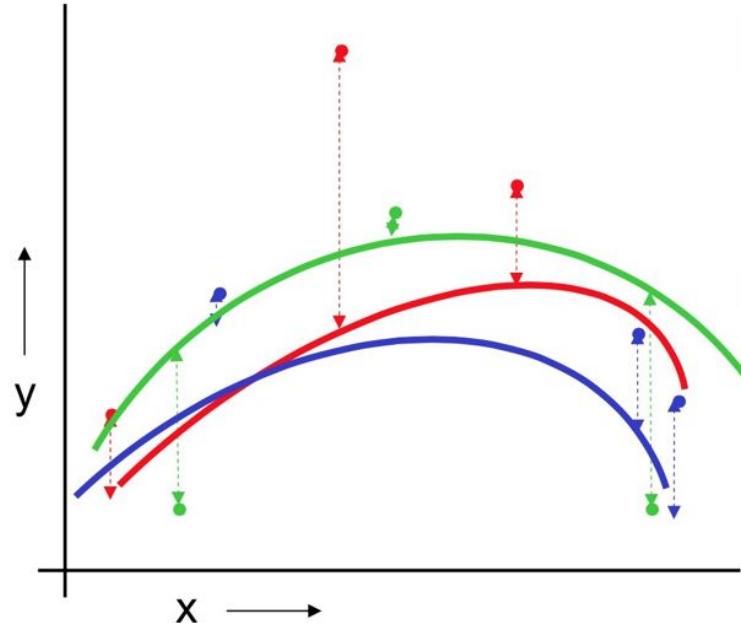
For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error

k -fold cross validation



Quadratic Regression

$$MSE_{3FOLD} = 1.11$$

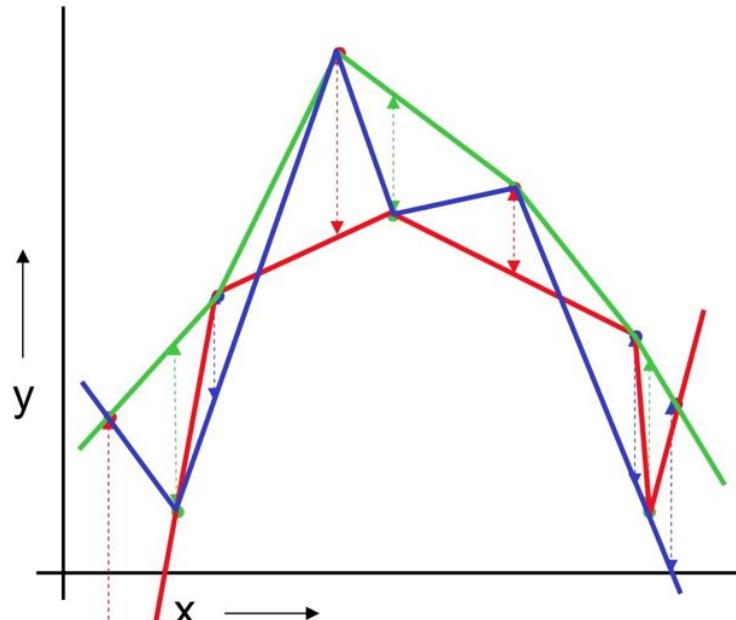
For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error

k -fold cross validation



Joint-the-dots

$$MSE_{3FOLD} = 2.93$$

For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error



Validator

Given the example we just worked, how would
you model these data?

A
linear
regression

Linear Regression
 $MSE_{3FOLD}=2.05$

B
quadratic
regression

Quadratic Regression
 $MSE_{3FOLD}=1.11$

C
pairwise linear
nonparametric
regression

Joint-the-dots
 $MSE_{3FOLD}=2.93$



Validator

Which approach would *you* use to limit overfitting?

