ALCF AI FOR SCIENCE TRAINING SERIES

Introduction To Machine Learning

Adapting scientific data, picking the right approach, and testing it well

Logan Ward
Data Science and Learning Division, Argonne National Laboratory

You’ve already done machine learning

	A	B
1	1	3.18079
2	2	4.94926
3	3	7.1364
4	4	9.05278
5	5	11.2544
6	6	13.2715
7	7	14.5924
8	8	17.036
9	9	18.9159
10	10	21.0061
11	11	22.5182
12	12	25.046
13	13	26.6946
14	14	29.1493
15	15	31.0294
16		

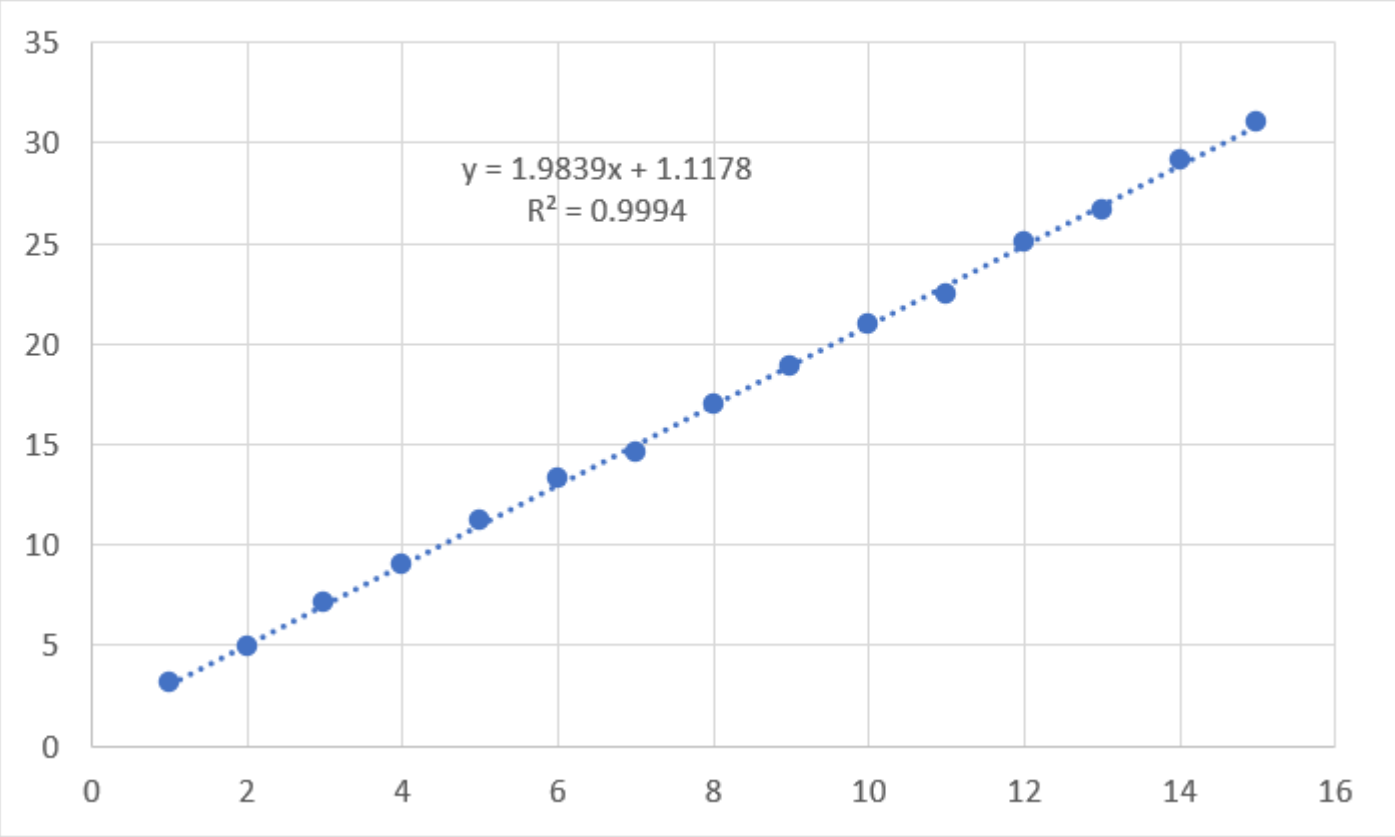
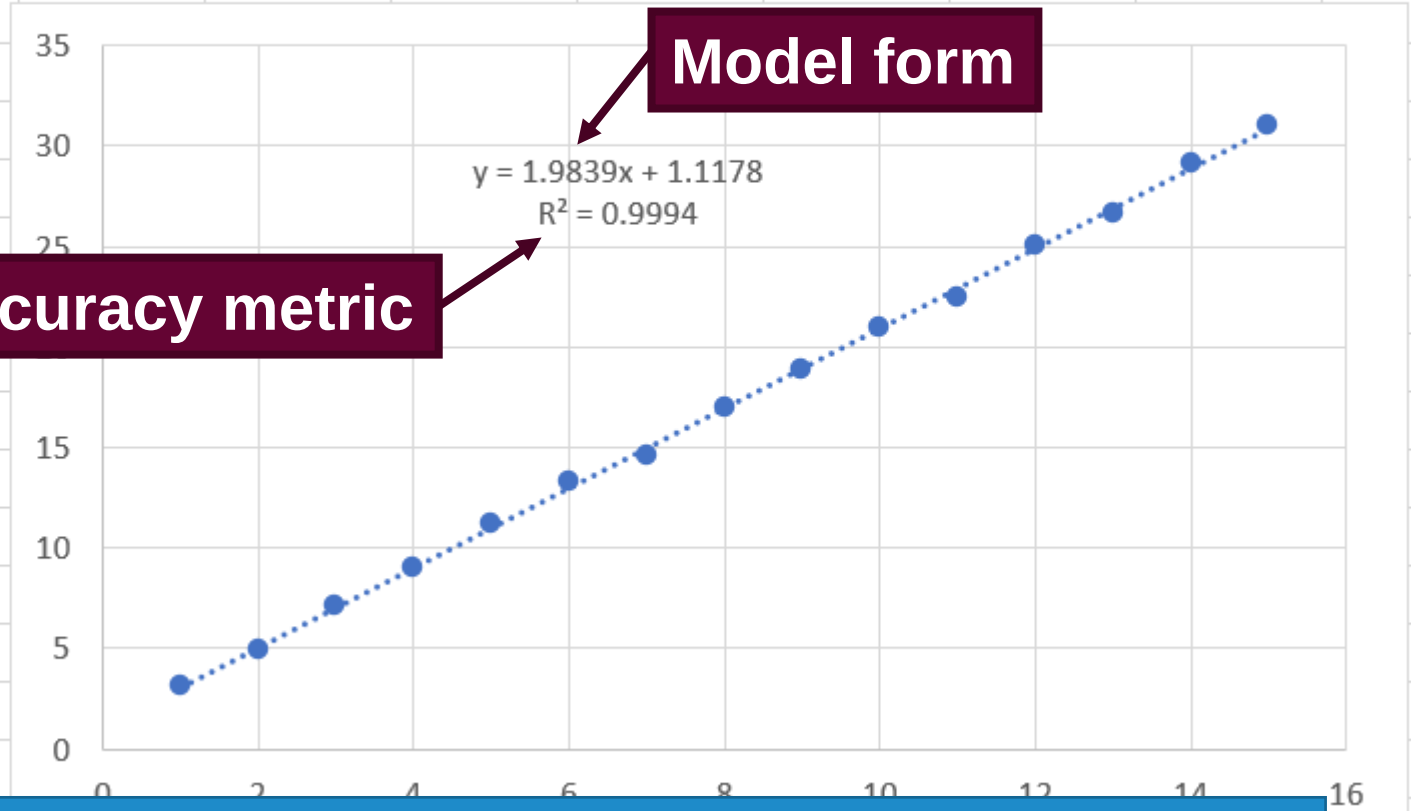


Image from Microsoft Excel

… and already have the main ideas

	A	B
1	1	3.18079
2	2	4.94926
3	3	7.1364
4	4	9.05278
5	5	11.2544
6	6	13.2715
7	7	14.5924
8	8	17.036
9	9	18.9159
10	10	21.0061
11	11	22.5182
12	12	25.046
13	13	26.6946
14	14	29.1493
15	15	31.0294
16		

Training Data



Model form

Accuracy metric

Today's lecture: Fill in the gaps, teach you modern tools

Image from Microsoft Excel

Is it different than conventional modeling? *Somewhat*

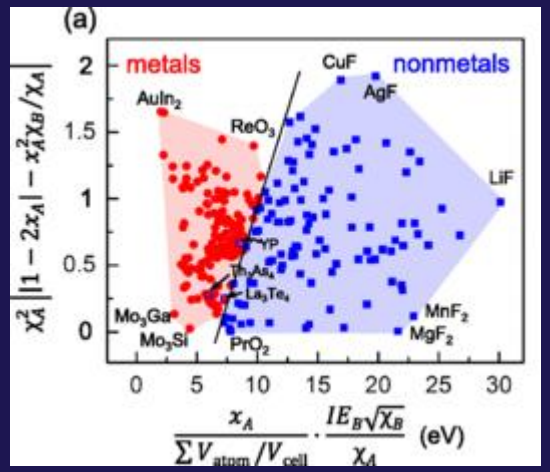
	"Semi-Empirical" Models	Machine Learning
Many Similarities	• Compose **equations** • Fit unknown terms to data • Evaluate to solve problem	• Compose **model forms** • Fit unknown terms to data • Evaluate to solve problem
Difference is in Scale	Tersoff Potential 12 parameters	Gaussian Approximation Potential 300 parameters
Bottom Line	Understand model by understanding ***physics***	Understand model by understanding ***algorithms***

Is it different than conventional empiricism? *Mostly*

$$M_d30 = 497 - 462(x_C + x_N) - 8.1x_{Mn}$$

Engineering is rife with empirical models, fit to data, used to understand physics

Modern AI techniques delegate more creativity to the algorithm, but more interpretation to the human



Ouyang et al. PRM. (2018), 083802

Ok, why do we have an “AI4Sci” tutorial series?

(And 2 hours on “machine learning”)

Key reason: AI requires different skills

1. Understanding algorithms is key for success
2. Complex models must be used with care
3. We still do not understand all uses of AI
4. Tools are not commonly taught or used elsewhere

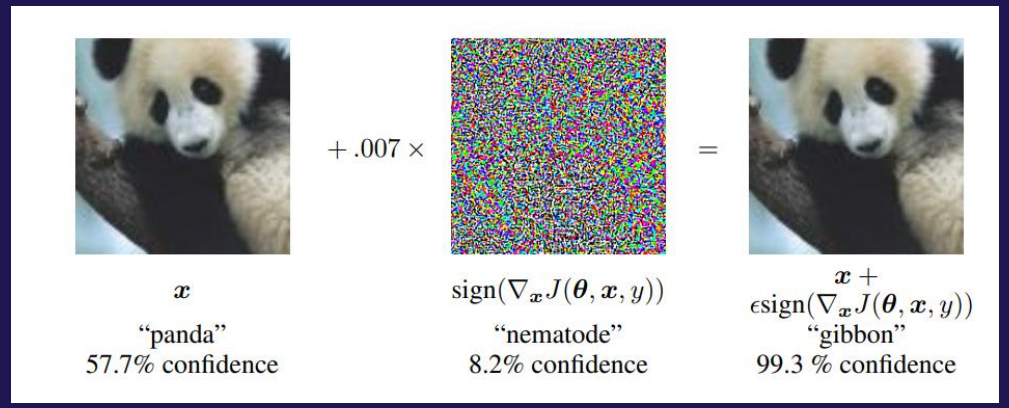


Figure: Goodfellow et al. ICLR (2015)

Today’s Goal: Introduce “machine learning”
1. What are the key algorithms?
2. How do I use them for “science?”

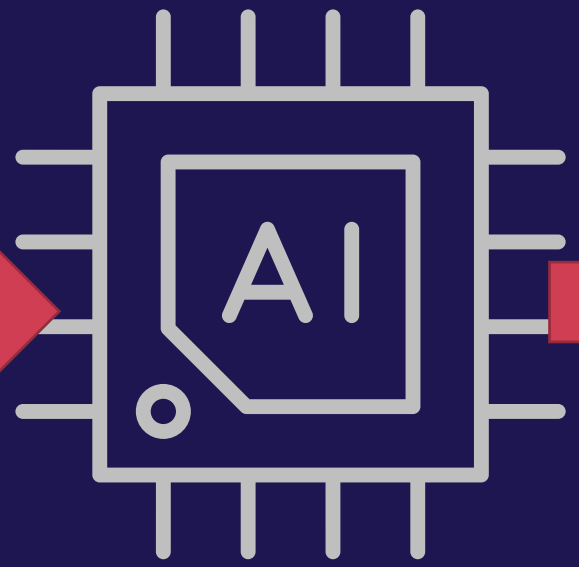
PART 1: THE ALGORITHMS!

Anatomy of a machine learning algorithms

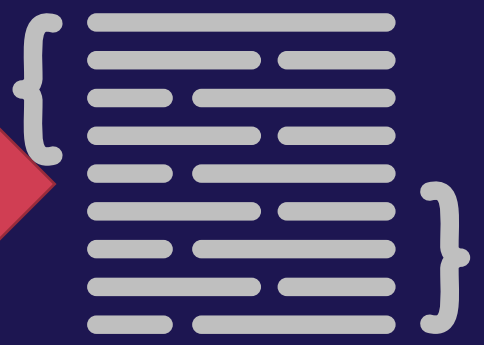
All have roughly the same parts

Training Data

Learning Function



Model/“Software”



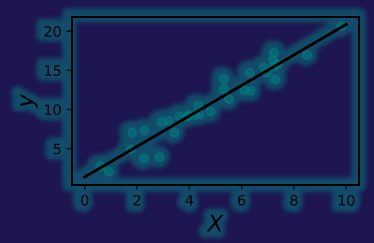
- <u>Entry/Data Point/Instance</u>
  Individual piece of data
- <u>Labels</u>
  measured about a record

- <u>Objective Function</u>
  What marks “better”?
- <u>Optimization Routine</u>
  How do I make it better?

- <u>Architecture</u>
  What form does the model take?
- <u>Weights</u>
  Values of adjustable parameters

Let’s go through a few key examples

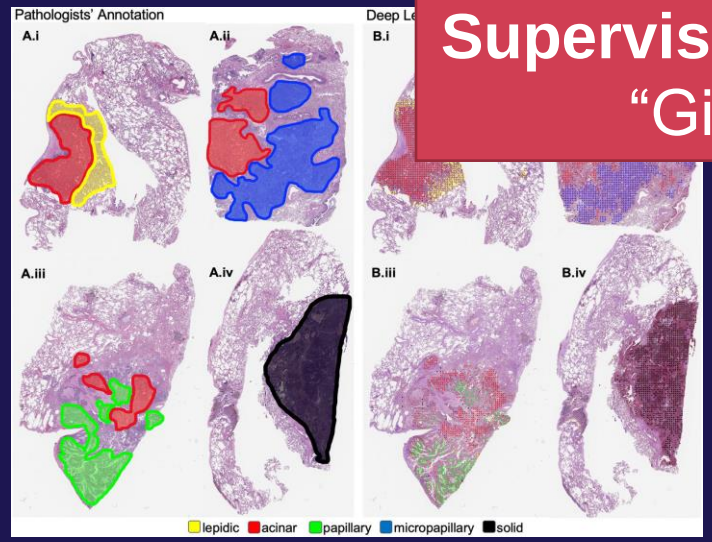
Anatomy of a machine learning algorithms

	Training Data	Learning Function	Model/“Software”
Supervised Learning	Entries: X Labels: y	Find *f* where $y \approx f(X)$ 	Linear models Lookup tables Decision Trees
Clustering	Entries: X	Discover similar subsets with X	k Means Agglomerative Clustering
Dimensionality Reduction	Entries: X	Learn *h* where $X \approx h(X)$	Principle Component Analysis Autoencoders

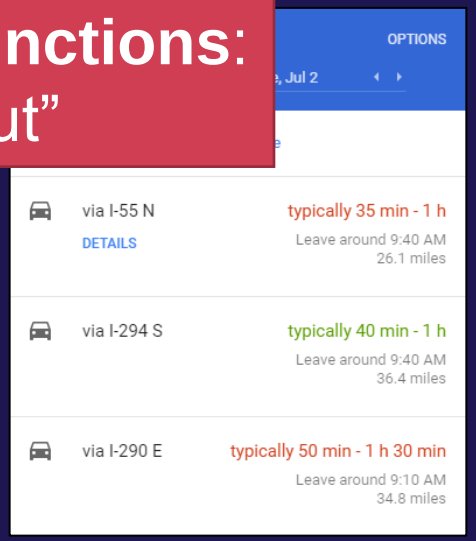
There are many types “learning” you can do. This list is not complete!

Supervised Learning: The ML you've done before

Supervised Learning models functions:
"Given inputs, predict output"

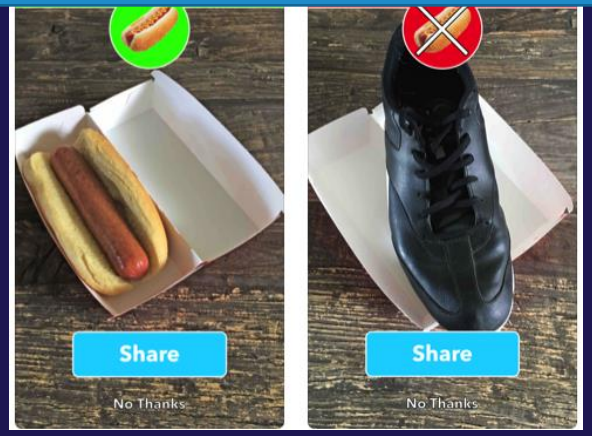


Wei *et al.* Sci Rep. (2019), 3358



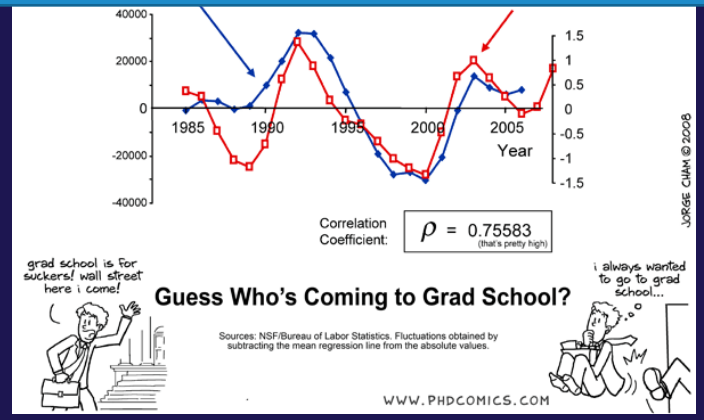
Google Maps

Classification: Outputs are *discrete*



https://apps.apple.com/us/app/not-hotdog/id1212457521

Regression: Outputs are *continuous*



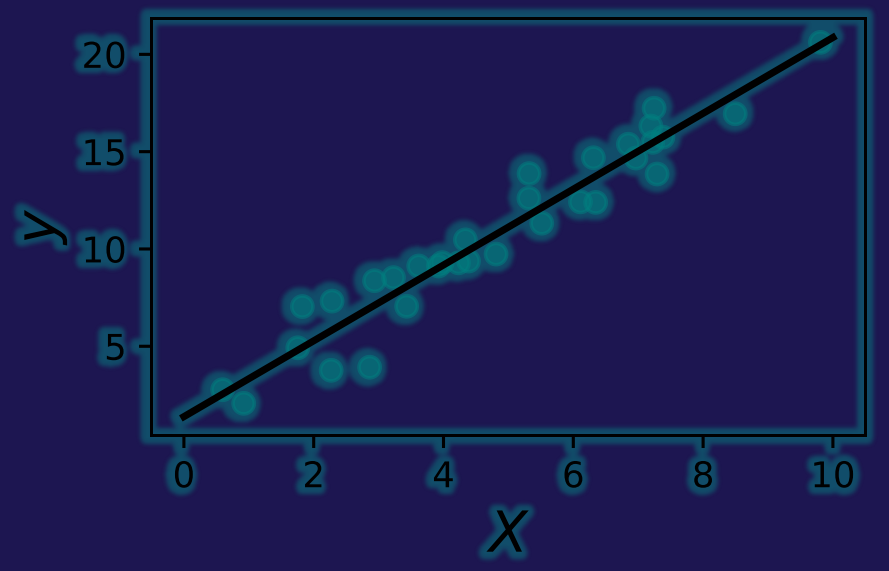
http://phdcomics.com/comics/archive.php?comicid=1078

An old friend: Simple Linear Regression

… but let's make it sound modern

Model Architecture

$$\boldsymbol{f(x; m, b) = mx + b}$$



Training Data: Inputs (x_i) and outputs (y_i)

Goal: Determine m and b that minimize

Loss Function

$$\sum_i (f(x_i; m, b) - y_i)^2$$

Optimizer

by computing

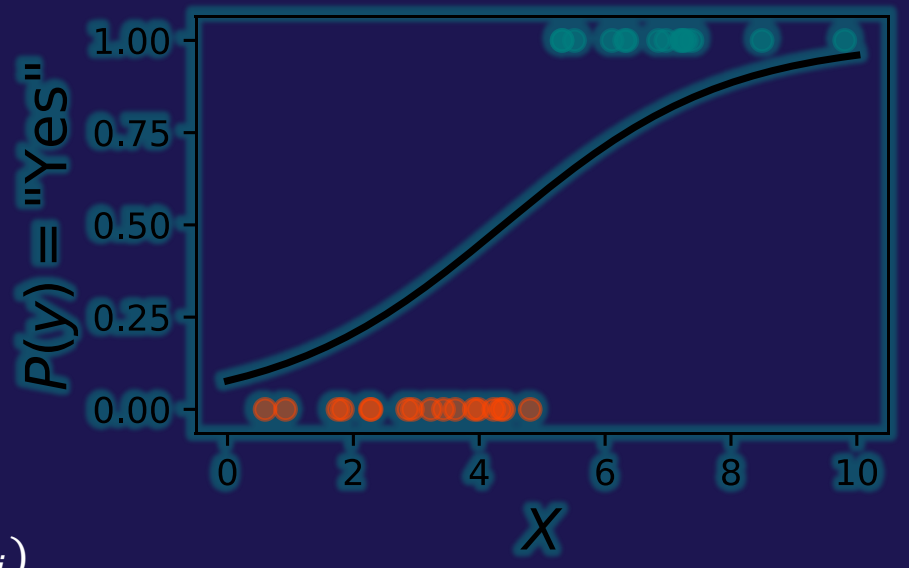
$$m = \mathrm{Cov}[x, y]/Var[x]$$
$$b = \bar{y} - m\bar{\mathrm{x}}$$

Simple Logistic Regression

A version of Linear Regression suitable for classification

Model Architecture

$$f(x; m, b) = \frac{1}{1 + e^{-(mx+b)}}$$



Training Data: Inputs (x_i) and outputs (y_i)

Goal: Determine m and b that minimize

Loss Function

$$L(m, b) \sum_i y_i \ln\big(f(x_i)\big) + (1 - y_i)\ln(1 - f(x_i)$$

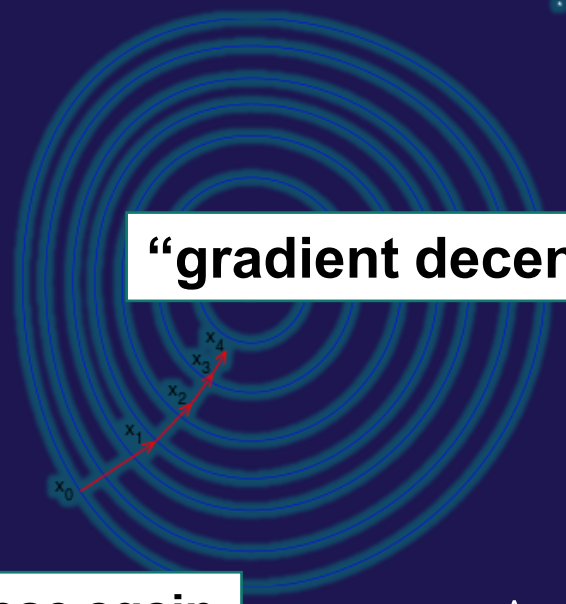
by computing

$$x_0 = (1, 0)$$
$$x_{n+1} = x_n + \gamma \nabla L(m, b)$$

"log loss"*

Optimizer

"gradient decent"*



Architecture + Loss + Optimizer = ML Algorithm
For Regression *and* Classification

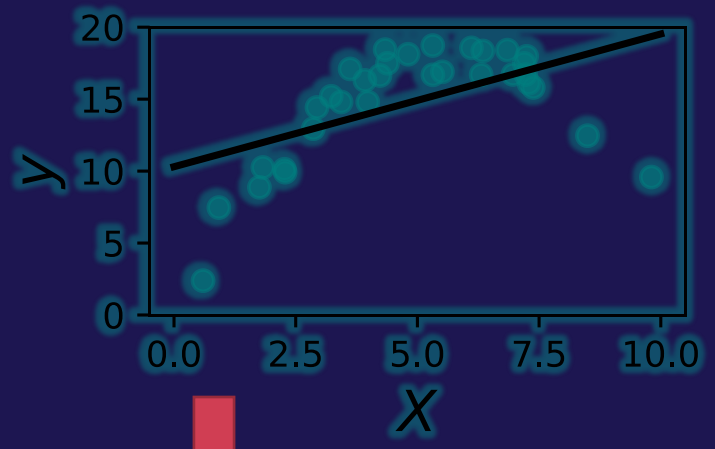
***You will be seeing these again**

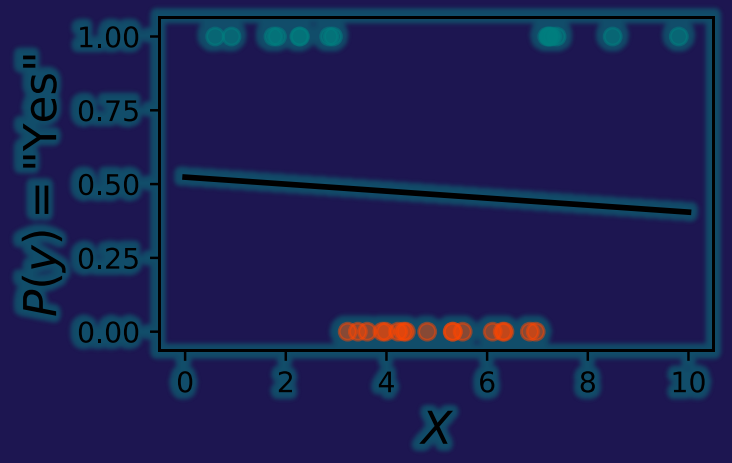
Linear Models Are Not Sufficient

Otherwise, this would be a very short lecture

Why Not? Model complexity is limited

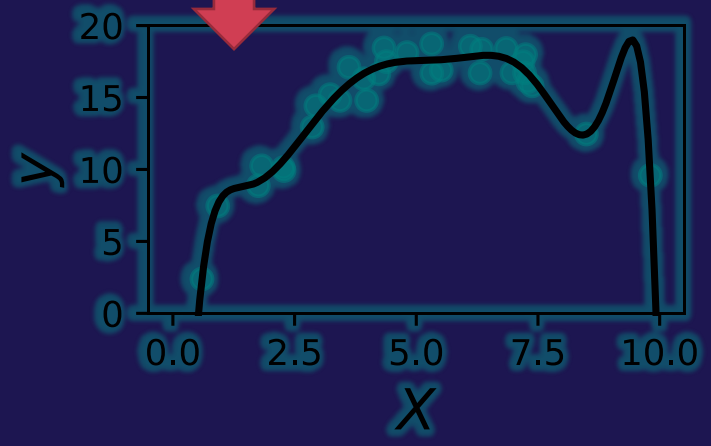
"underfit"*





… and adding complexity comes with risks

"overfit"*



Key Questions for Supervised ML:

1. How to add more complexity?
2. How to know when "overfit"?

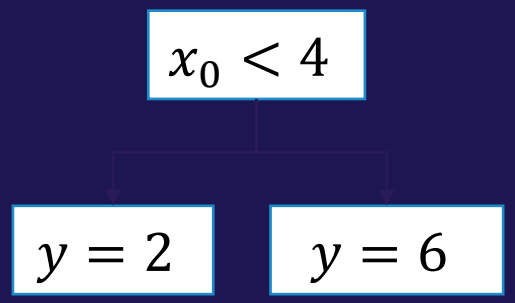
There is a ZOO of ML Algorithms

Variations of Bayes' Theorem

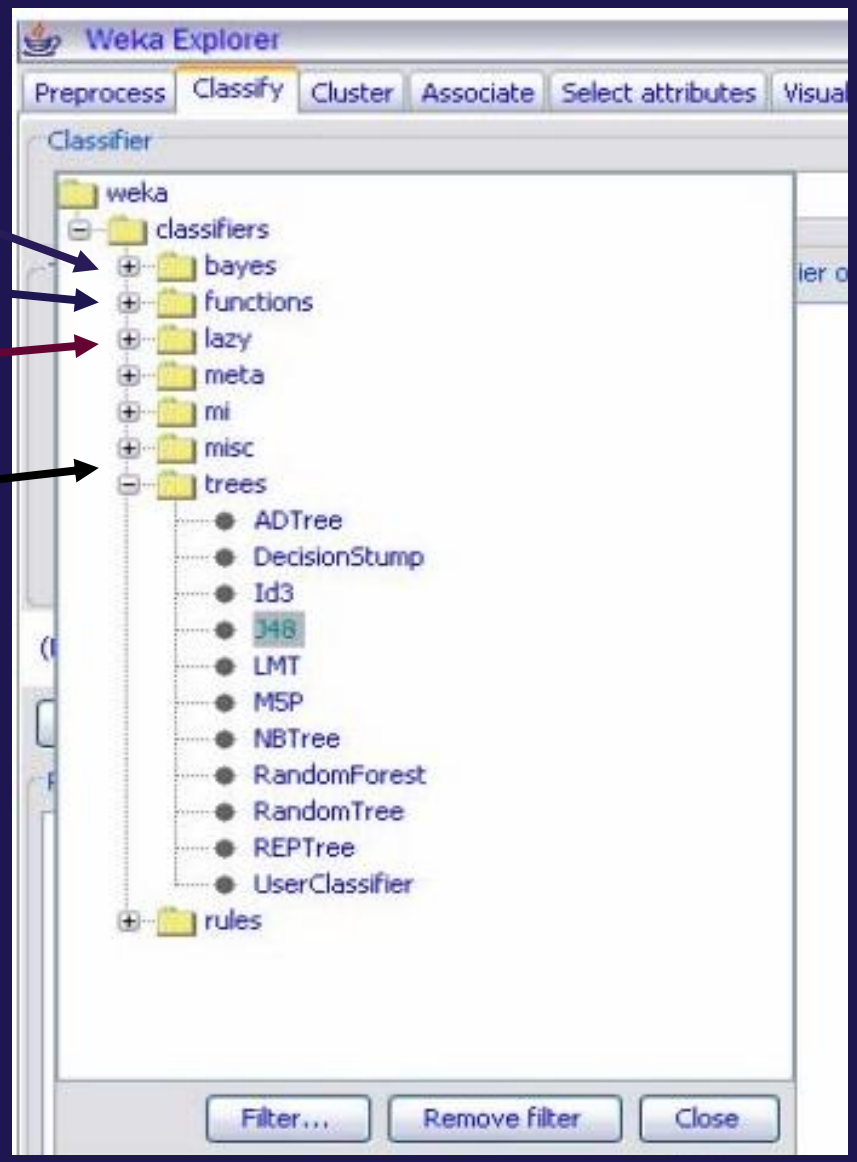
Linear regression, etc.

k-Nearest neighbors

Many kinds of decision trees



Trees vary in…
- whether to process inputs
- ways picking splits
- what are on the "leaves"
- how to prune after training



http://www.site.uottawa.ca/~stan/csi7163/weka_workshop_slides.pdf

There is no "algorithm to rule them all"

Linear Models are quite important

Key Concept: "Penalized" regression methods

$$\|y - \alpha X\|_2 + \lambda\|\alpha\|_p \quad \text{<- "Regularizer"}$$

Subtlety: More than one type of "regularization" loss

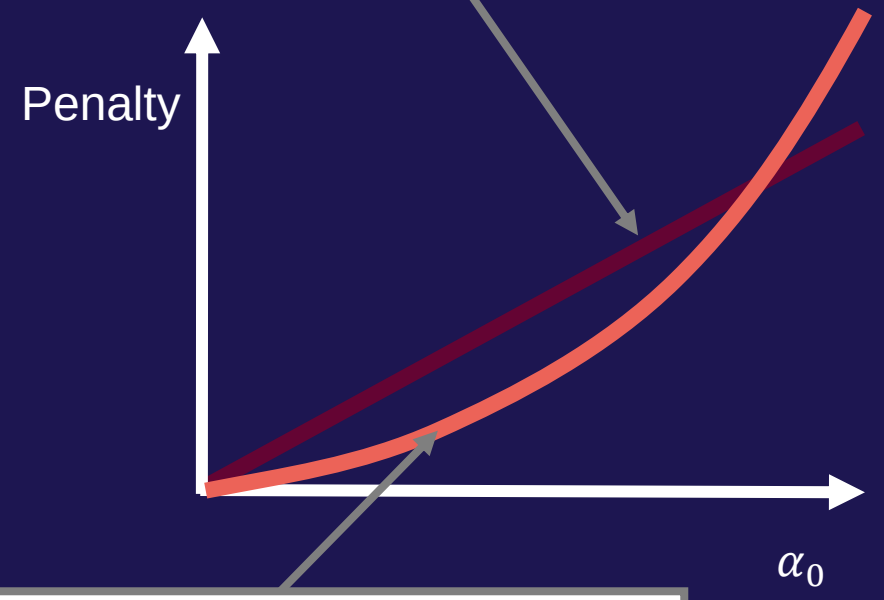
where $\|\alpha\|_p$ is the "L^p norm": $\|\alpha\|_p = (|\alpha_0|^p + \cdots + |\alpha_n|^p)^{\frac{1}{p}}$

Advantages:

1. Fast to train! L_2 has an analytic solution
2. Interpretable, especially for L_1

Disadvantage: Limited complexity

L^1-norm: "sparse linear regression"
- Moderate penalties for small parameters
- Sparse models (many $\alpha = 0$)



L^2-norm: "ridge regression"
- Small penalties for small parameters
- Many parameters (No $\alpha = 0$)

Key algorithm: Decision Trees

Model form: Series of rules

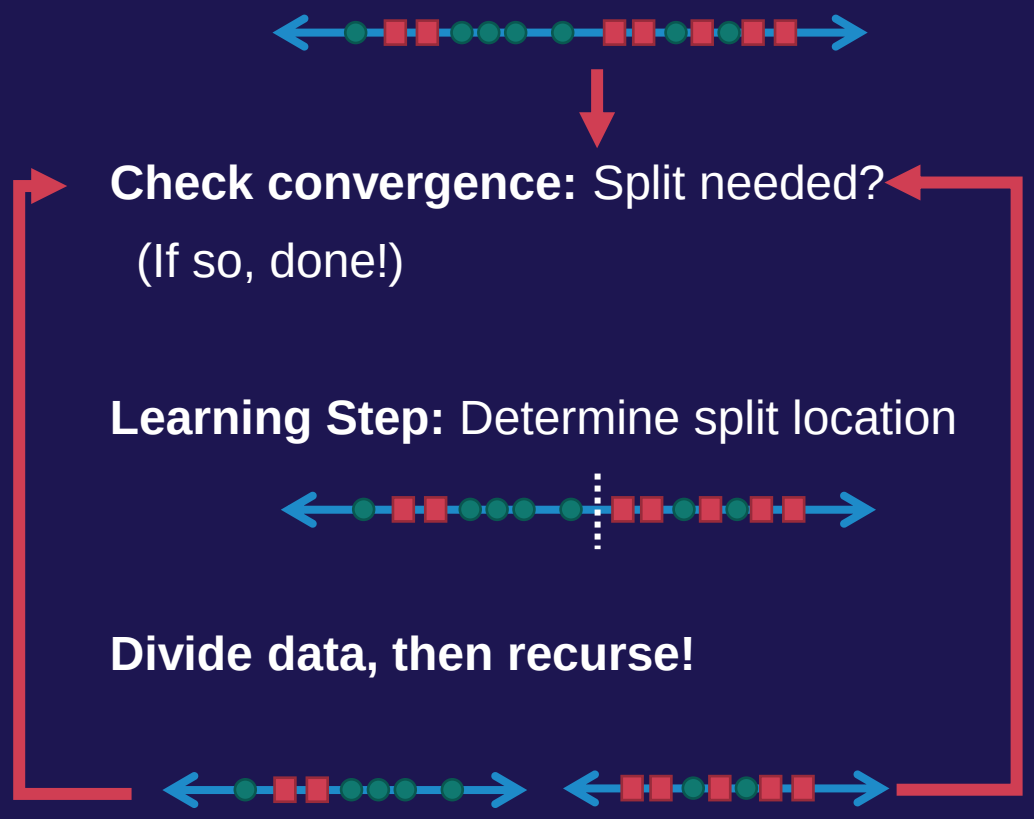
Learning objective: Find rules that maximize performance

Learning algorithm: Recursive rule selection

(Many variations of these components!)

Why might I choose decision trees:

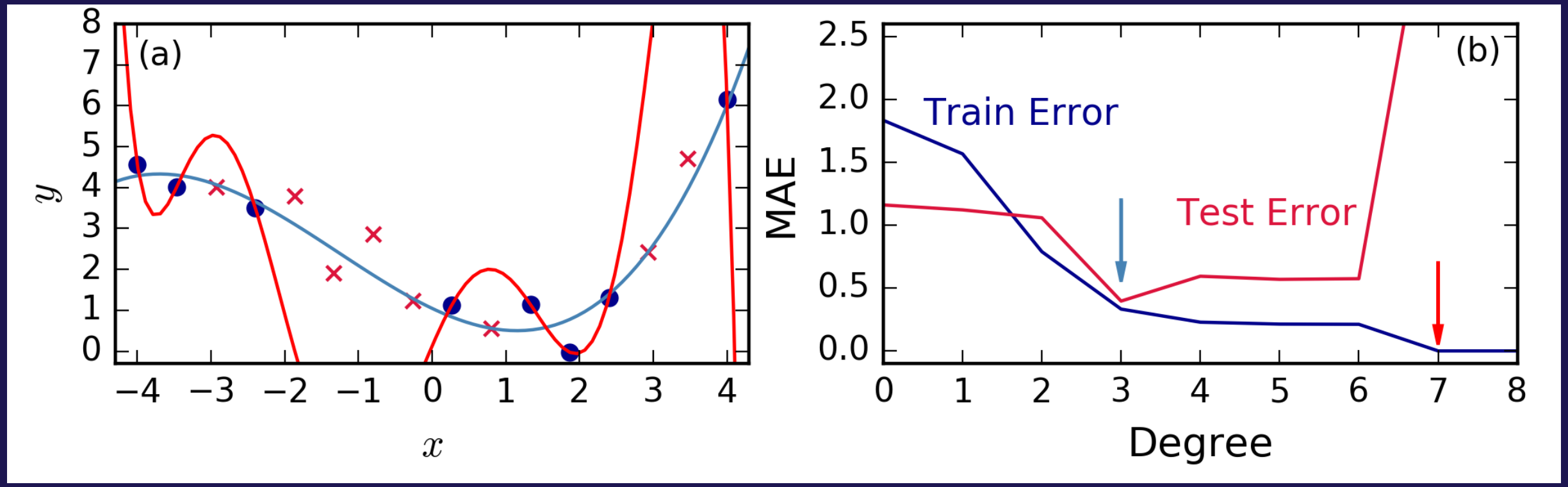
- Fast to train ($O(N\log N)$)
- Mostly interpretable by humans



MODEL SELECTION:
THE KEY TASK IN SUPERVISED ML

A Reprise: Overfitting and Complexity

Training accuracy vs “generalizability”



How do We Adjust COMPLEXITY?

We'll just talk linear models for now, but these are general ideas

LIMIT COMPLEXITY

Reduce degrees of freedom

$$f(x) = a_0 + a_1 x + \cancel{a_2 x^2 + [\ldots]}$$

PENALIZE COMPLEXITY

Add "complexity" to loss function

$$\sum_i (f(x_i) - y_i)^2 + \boldsymbol{\lambda} a a^T$$

Larger coefficients = bigger penalty

Two *MAJOR* Concepts

Don't leave this room without them!

CROSS VALIDATION

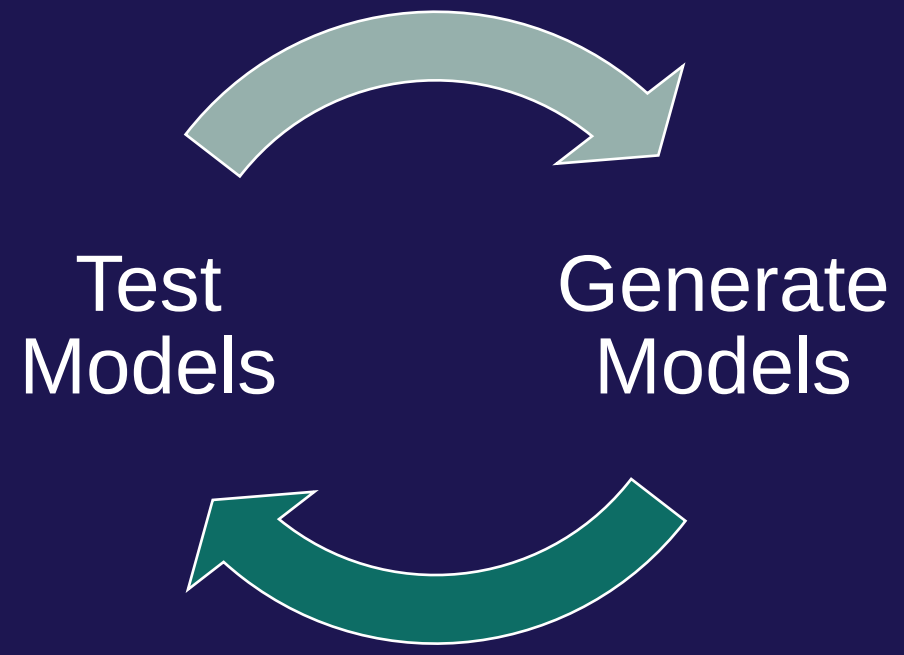
Given available data, test whether model is predictive

Basic Techniques:

- *Shuffle Split*
  1. Pick 10% as test set
  2. Train on remaining 90%
  3. Test model on test set
- *Leave-one-out*
  1. Pick one entry
  2. Train on remaining entries
  3. Test model on held-out entry
  4. Repeat using each entry

HYPERPARAMETER OPTIMIZATION

Adjust model settings to maximize CV performance



Summary for Part 1

What is machine learning? Algorithms that generate software from desired outcomes

What makes kinds of learning different? Desired inputs and outputs, learning algorithms

Example: “Supervised Learning” finds $y = f(x)$

Why are there so many learning algorithms? Each have their own advantages and drawbacks

Example: Kernel methods train time is $O(N^3)$

Why must I optimize hyperparameters? Adjust been “under” and “over” fitting

Example: Number of polynomial terms

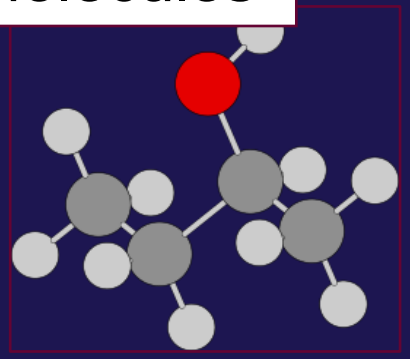
PRACTICAL EXERCISE 1:
BUILDING MODELS WITH SKLEARN!

PART 2: USING IT FOR SCIENCE (MOSTLY MATERIALS SCIENCE, BECAUSE I'M A MATERIALS ENGINEER)

Not all data are vectors

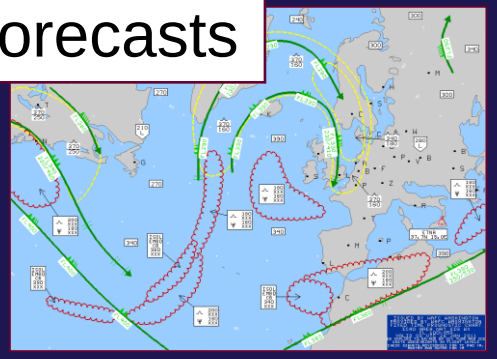
And that's OK!

Molecules

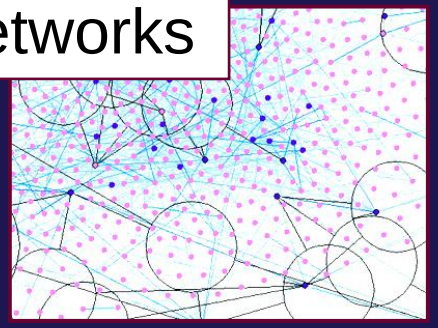


Images

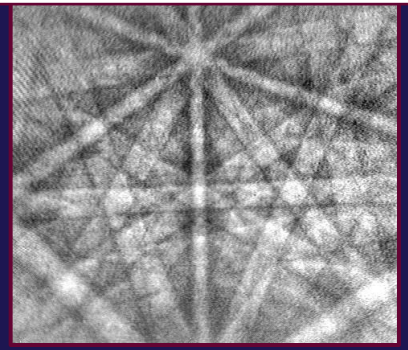
Weather Forecasts



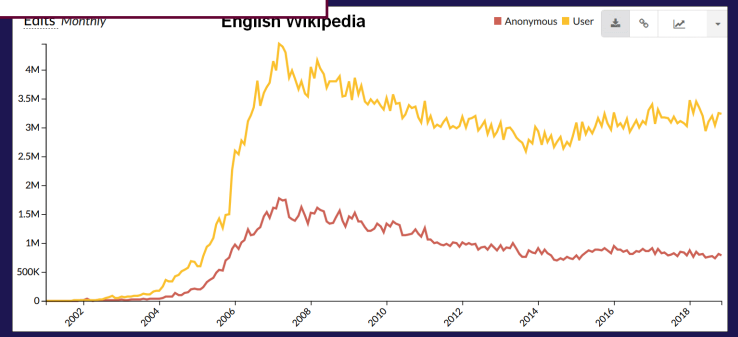
Social Networks



EBSD Patterns



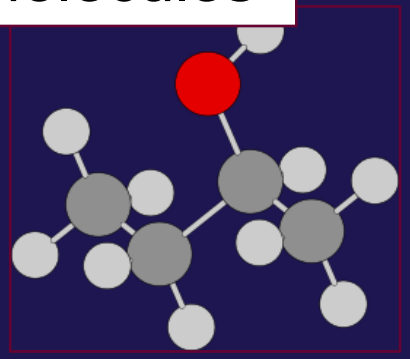
Timeseries



Not all data are vectors

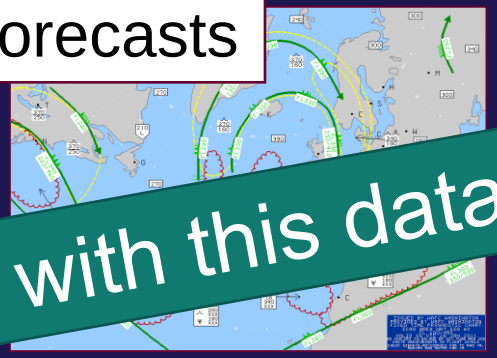
And that's OK!

Molecules



Images

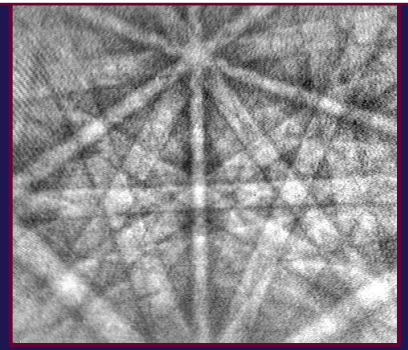
Weather Forecasts



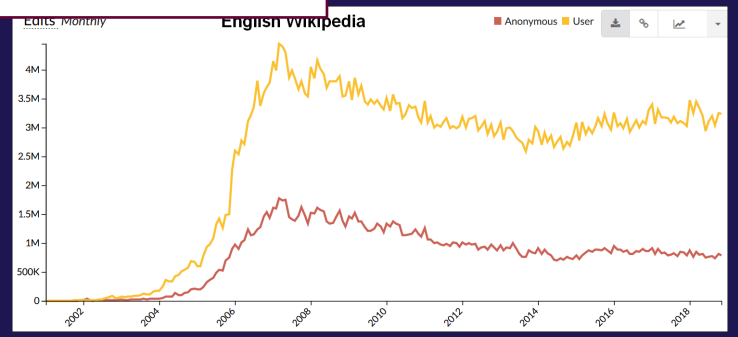
It's still possible to use machine learning with this data

Sc
Netw

EBSD Patterns



Timeseries



What do NaCl crystals look like?

High Electronegativity Difference

Large Band Gap

Minimal Interaction with Visible Light

Clear

NaCl

Representations make knowledge accessible to learning algorithms

Similar Atomic Radii

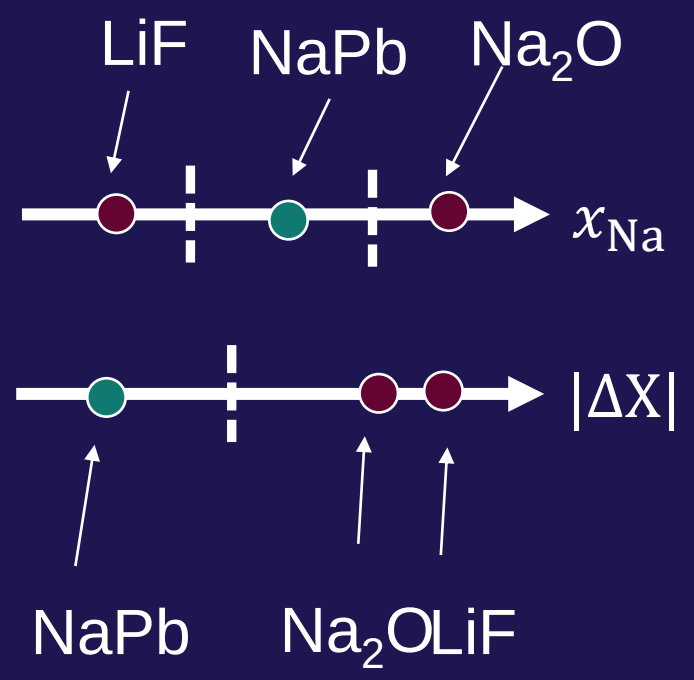
Coordination Number of 6

Cubic Symmetry

Cubic Crystals

How to translate chemistry/physics/… to a computer?

Representation: *Set of quantitative attributes that describe a material, molecule, …*



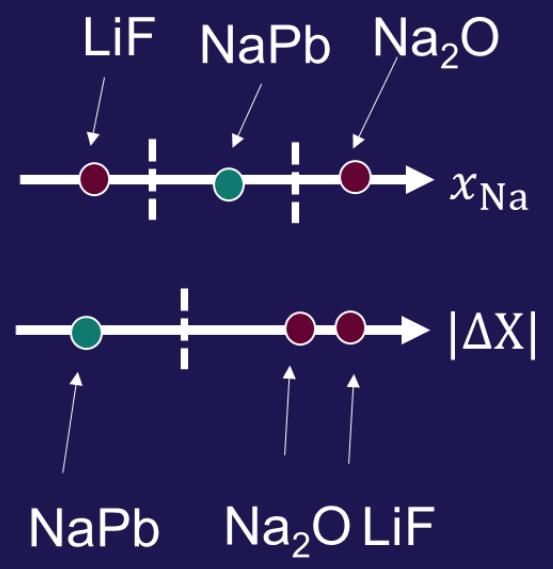
Why do we need them?

- Machine learning tools take <u>tensors</u>

What makes a good one?

- Easy to learn generalizable rules

What makes a good representation?

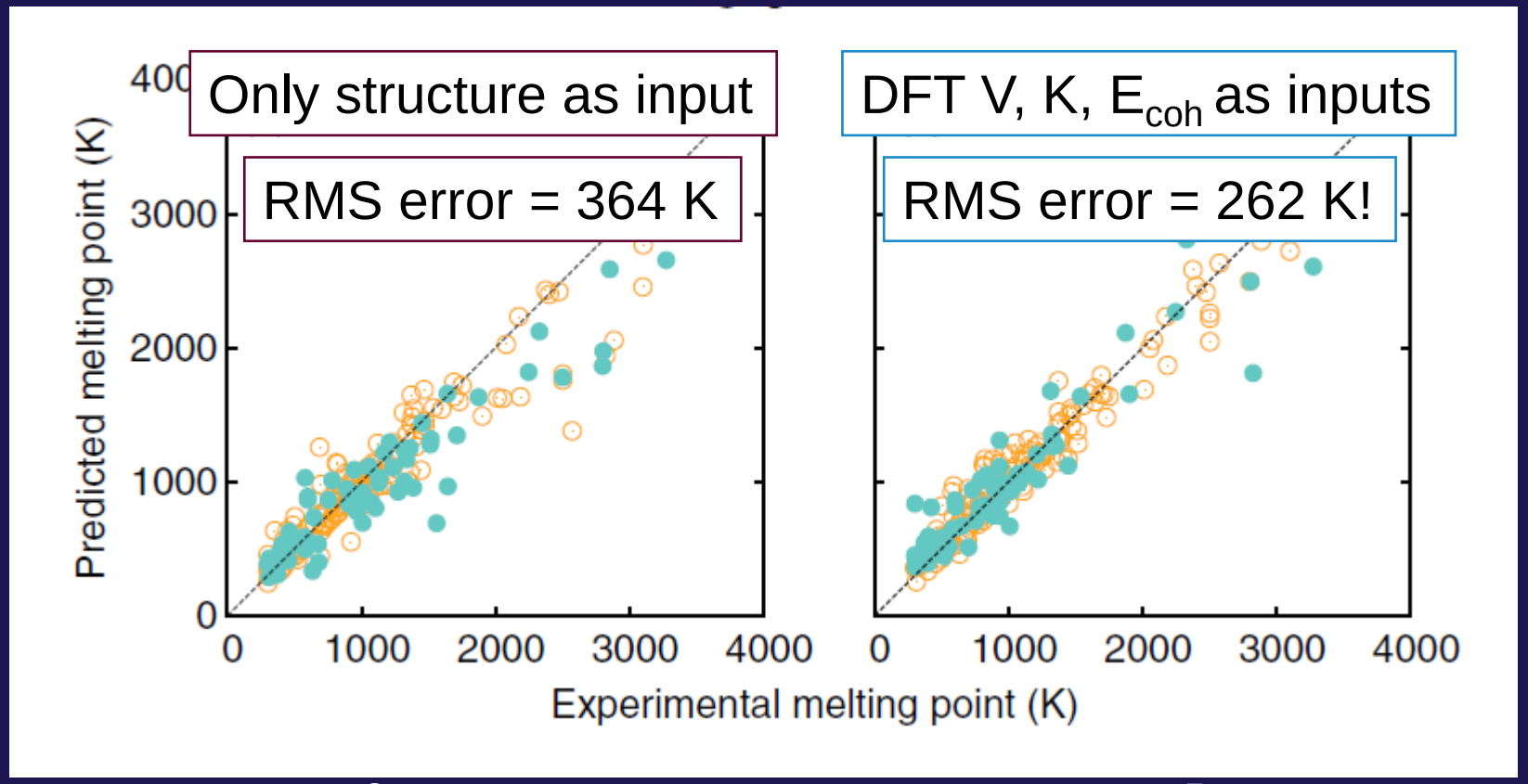


	Fraction of Na	Difference in Electronegativity
Complete	✓ Yes, no overlapping points	✓ Yes, no overlapping
Compact	✓ Yes, single feature!	✓ Yes, single feature!
Descriptive	✗ No, rules are complicated	✓ Yes, matches textbook
Simple	✓ Yes, no compute	✓ Yes, minimal compute

Ref: Faber et al. Int. J. Quantum Chem. (2015)

Representations standards are not clear cut

Seko et al. used DFT-computed properties as inputs to an ML model



Ok for 10^2 compounds. Not OK for 10^5 compounds

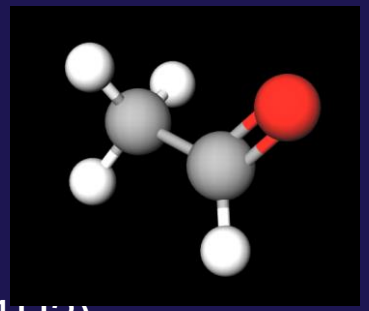
Key Concept: There is not and will not be a "one representation for all uses."

Types of representations: Discriminative vs Descriptive

Discriminative		Descriptive
Make features that capture intuition	**Concept**	Make features that distinguish examples
Element properties ($\lvert\Delta X\rvert$) Interocular distances	**Examples**	Atomic fractions (x_{Na}) Pixel values in images
Learn from little data (potentially) Interpretable models	**Advantages**	Maximum expressivity
Baking basis into the model	**Challenges**	Requires more data (learn features *and* model)
Typical for "conventional" ML		**Typical for "deep learning"**

Molecular Descriptors

Discriminative

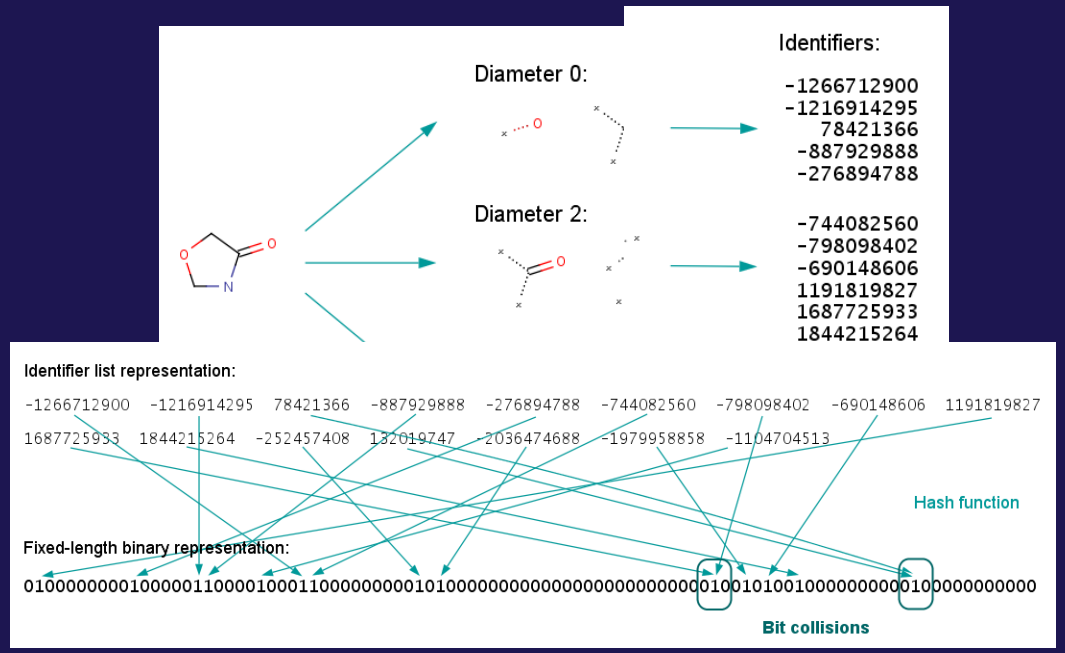


Descriptive

Line-notations for structure:

- SMILES (ex: “CC=O”)
- InChI (ex: InChI=1S/C2H4O/c1-2-3/h2H,1H3)

Fingerprints:



Extremely Well-Studied

Handbook of Molecular Descriptors

Author(s): Prof. Dr. Roberto Todeschini, Dr. Viviana Consonni

First published: 22 September 2000

Print ISBN: 9783527299133 | Online ISBN: 9783527613106 | DOI: 10.1002/9783527613106

Copyright © 2000 WILEY-VCH Verlag GmbH

Book Series: Methods and Principles in Medicinal Chemistry

Free Access

Bibliography (Pages: 524-667)

First Page | PDF | References | Request permissions

Many types of descriptors:

1. Constitutional (ex: “how many Ns?”)
2. Structural (ex: Solvent-Accessible Surface Area)
3. Quantum-chemical (ex: partial charges)

Figures: docs.chemaxon.com

Good Ref: T. Le et al. Chem. Rev. (2012), 2889

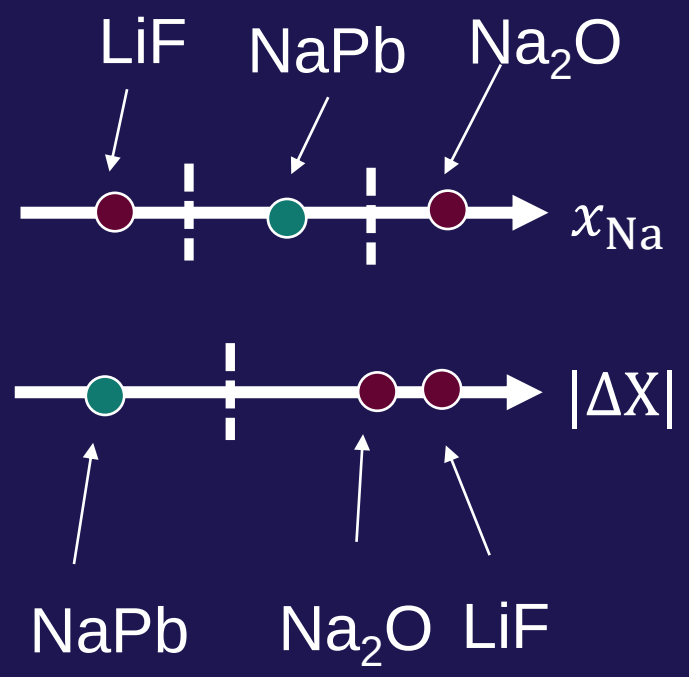
Conclusion: How to translate science to ML?

Key concept: Express your data in a way that captures knowledge

Many ways to achieve "Guiding Principles:"

- **Complete:** Separate different entries
- **Compact:** Minimal complexity
- **Descriptive:** Maximum relevance
- **Simple:** Fast to gather

Work here can yield **significant benefits**,
and is a great way to engage **domain experts**

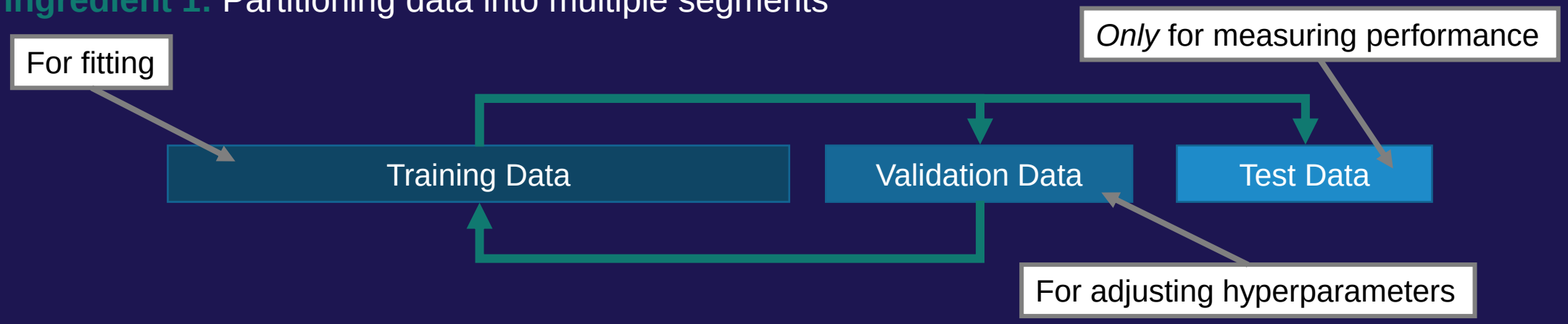


PRACTICAL EXERCISE 2: MACHINE LEARNING WITH CHEMISTRY DATA

PART 2B: VALIDATE LIKE A PRO

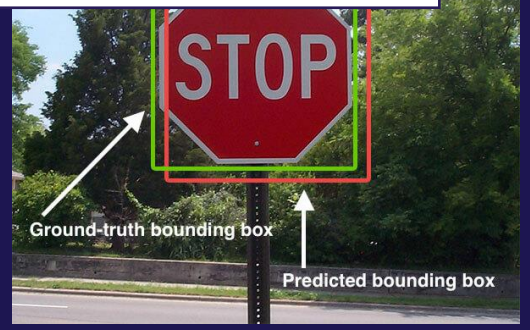
General approach to testing: “cross validation”

Ingredient 1: Partitioning data into multiple segments

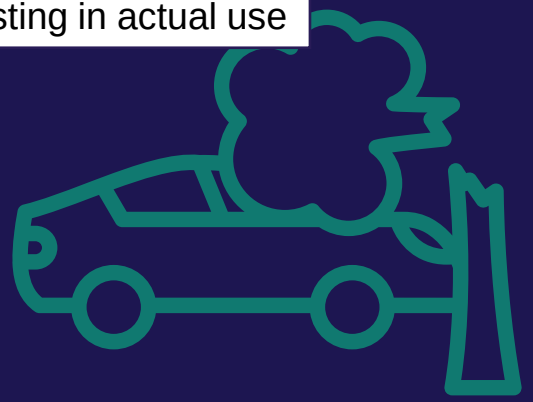


Ingredient 2: Choose how you measure performance

✔ Simple Performance Metrics



💯 Testing in actual use



My advice: Think carefully about how to mimic your application

Image: pyimagesearch

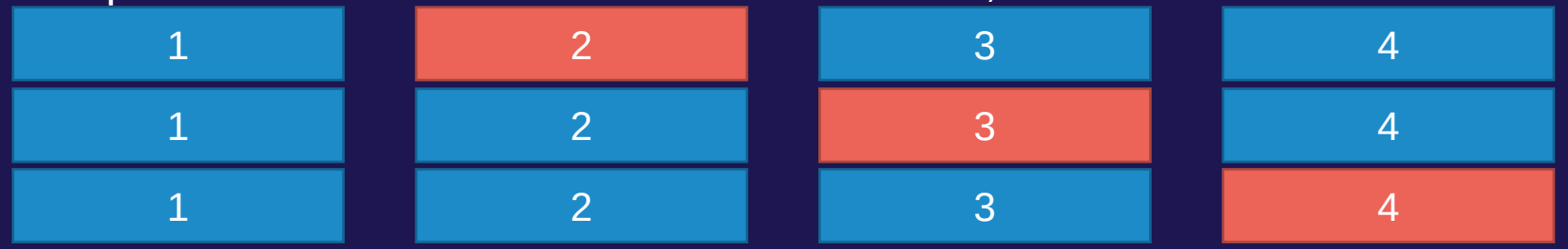
A standard approach: k-fold cross validation

Step 1: Split dataset randomly into *k* equally-sized chunks

Step 2: Hold out 1 chunk as <u>test data</u>, use remainder as <u>train</u> and <u>validation</u> data

Step 3: Measure performance on remaining chunk

Step 4: Repeat from Step 2 with a different choice from the *k* chunks, until all chunks used



Step 5: Report performance as average over all iterations

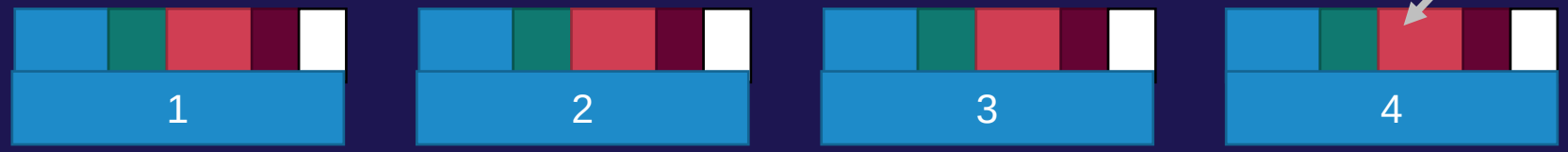
A wrinkle: Data has hidden structure

Common case: Some data are more alike than others

Question: How would the model perform on new types of data?

K-fold validation does not answer this question!

Always predicting within same group!



Simulate predicting outside of groups.

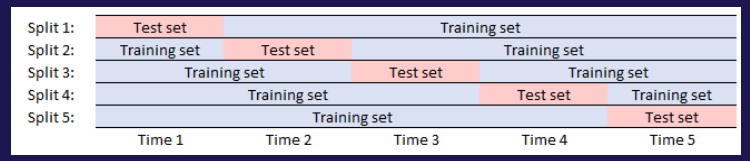
Alternative: Use "hidden" groups as training and test sets

Aside: I avoid using the term "extrapolation." Ask me later…

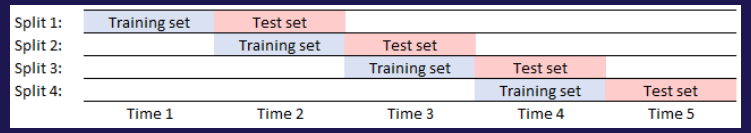
Many examples of “data with hidden structure”

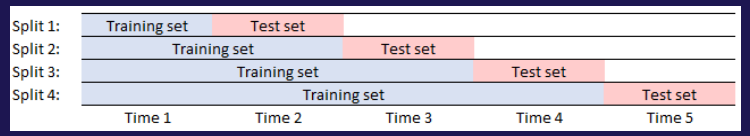
Time Series

✕ Predicting past *and* future



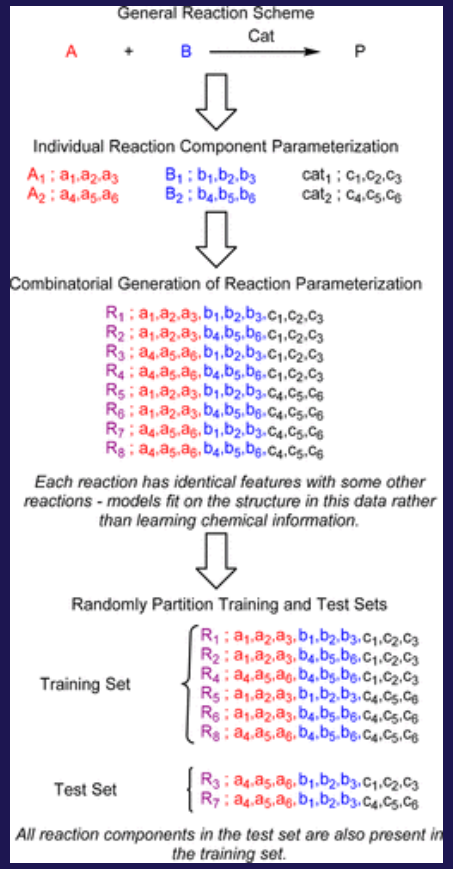
✔ Predict only the future





Source: CrossValidated

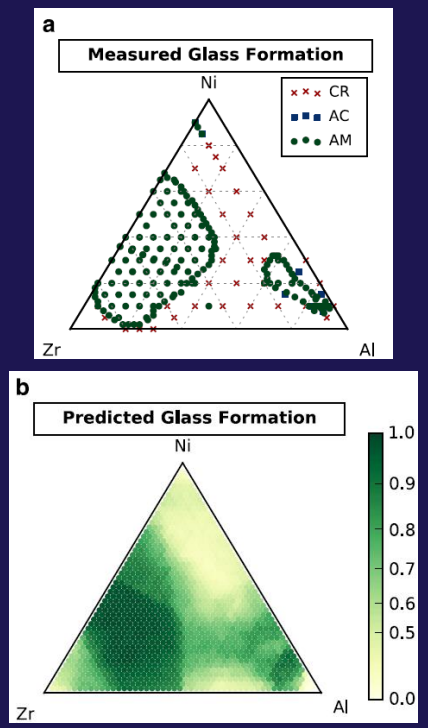
Combinatorial Chemistry



Ref: Zahrt et al. ACS Combi. (2020)

Metallic Alloys

✔ Excluding an alloy system



Ref: Ward et al. npj Comp Mat. (2016)

There are many ways to measure performance

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

Is “No missed diagnoses” your goal?

Or is it “No false convictions”?

Table from Wikipedia

These only scratch the service for classification

Sources: [15][16][17][18][19][20][21][22] view · talk · edit

		Predicted condition			
	Total population $= P + N$	**Positive (PP)**	**Negative (PN)**	Informedness, bookmaker informedness (BM) $= TPR + TNR - 1$	Prevalence threshold (PT) $= \frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**Positive (P)**	**True positive (TP),** **hit**	**False negative (FN),** **type II error, miss,** **underestimation**	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power $= \frac{TP}{P} = 1 - FNR$	False negative rate (FNR), miss rate $= \frac{FN}{P} = 1 - TPR$
	Negative (N)	**False positive (FP),** **type I error, false alarm,** **overestimation**	**True negative (TN),** **correct rejection**	False positive rate (FPR), probability of false alarm, fall-out $= \frac{FP}{N} = 1 - TNR$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{TN}{N} = 1 - FPR$
	Prevalence $= \frac{P}{P + N}$	Positive predictive value (PPV), precision $= \frac{TP}{PP} = 1 - FDR$	False omission rate (FOR) $= \frac{FN}{PN} = 1 - NPV$	Positive likelihood ratio (LR+) $= \frac{TPR}{FPR}$	Negative likelihood ratio (LR−) $= \frac{FNR}{TNR}$
	Accuracy (ACC) $= \frac{TP + TN}{P + N}$	False discovery rate (FDR) $= \frac{FP}{PP} = 1 - PPV$	Negative predictive value (NPV) $= \frac{TN}{PN} = 1 - FOR$	Markedness (MK), deltaP (Δp) $= PPV + NPV - 1$	Diagnostic odds ratio (DOR) $= \frac{LR+}{LR-}$
	Balanced accuracy (BA) $= \frac{TPR + TNR}{2}$	F_1 score $= \frac{2PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$	Fowlkes–Mallows index (FM) $= \sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) $= \sqrt{TPR \times TNR \times PPV \times NPV} - \sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{TP}{TP + FN + FP}$

You should think about what success for your model looks like!

Better regression validation: An example

Great Example: Predicting Density of States (phDOS) of materials (Kong et al., 2021)

Best model for predicting C_v (a derived property)…

ML model	Setting		phDOS prediction				Calculated C_v		Calculated $\bar{\omega}$	
	Scaling	Loss	R^2	MAE	MSE	WD	MAE	MSE	MAE	MSE
Mat2Spec	SumNorm	WD	0.57	0.085	0.026	**21**	**1.32**	**10**	**10.6**	**348**
E3NN	SumNorm	KL	0.48	0.105	0.036	50	4.88	77	41.1	3718
GATGNN	SumNorm	KL	-1.05	0.177	0.057	215	22.4	756	205	51609
Mat2Spec	SumNorm	KL	0.62	**0.078**	**0.023**	24	1.96	11	17.1	625

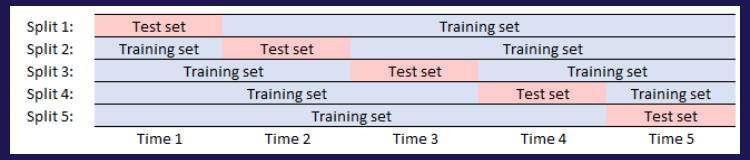
… is not the one that predicts phDOS the best.

Choices of which performance metric to use will affect your model choice

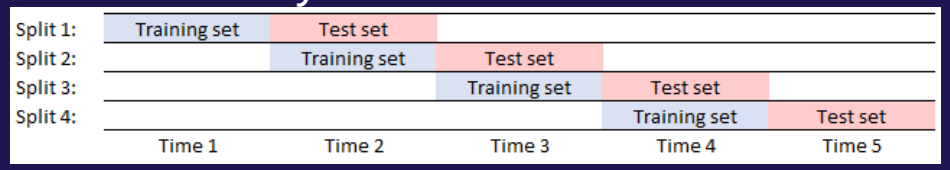
Study your problem, it will give you better models

Ingredient 1: Cross-validation that simulates how you want to use the model

✕ Predicting past *and* future



✔ Predict only the future



Source: CrossValidated

Ingredient 2: Evaluate the model how you actually measure performance

ML model	Setting		phDOS prediction				Calculated C_v		Calculated $\bar{\omega}$	
	Scaling	Loss	R^2	MAE	MSE	WD	MAE	MSE	MAE	MSE
Mat2Spec	SumNorm	WD	0.57	0.085	0.026	**21**	**1.32**	**10**	**10.6**	**348**
E3NN	SumNorm	KL	0.48	0.105	0.036	50	4.88	77	41.1	3718
GATGNN	SumNorm	KL	-1.05	0.177	0.057	215	22.4	756	205	51609
Mat2Spec	SumNorm	KL	0.62	**0.078**	**0.023**	24	1.96	11	17.1	625

Source: Kong et al., 2021

My advice: Think carefully about how to mimic your application

WRAP UP: WHAT DID YOU LEARN?

Ok, why do we have an “AI4Sci” tutorial series?

(And 2 hours on “machine learning”)

Key reason: AI requires different skills

1. Understanding algorithms is key for success
2. Complex models must be used with care
3. We still do not understand all uses of AI
4. Tools are not commonly taught or used elsewhere

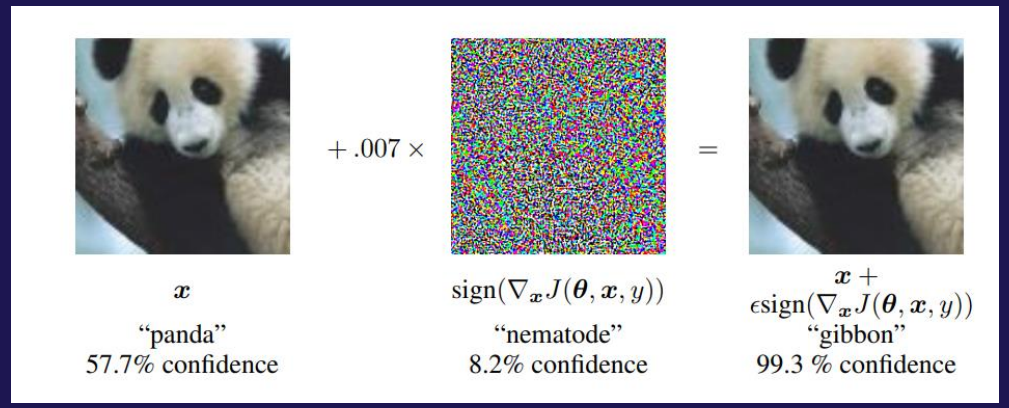


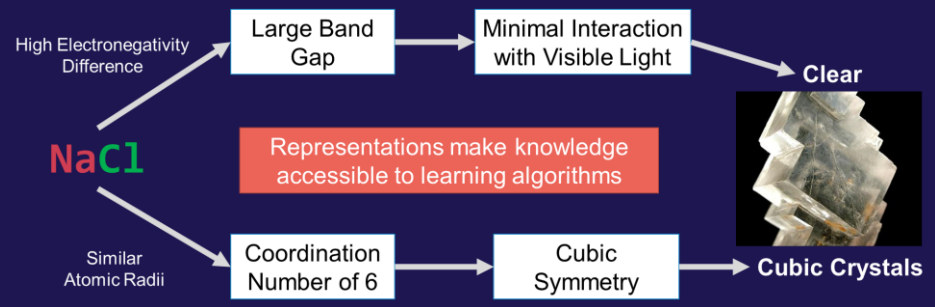
Figure: Goodfellow et al. ICLR (2015)

Today’s Goal: Introduce “machine learning”
1. What are the key algorithms?
2. How do I use them for “science?”

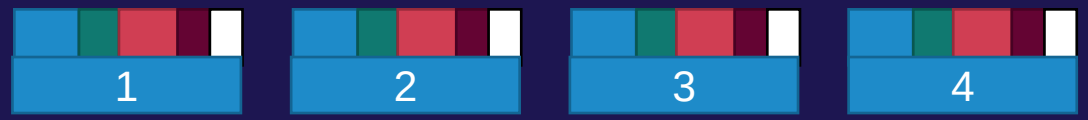
Key takeaways!

1. A view into the depths of how many ML algorithms there are, and knowledge of how to use a few *classic* ones

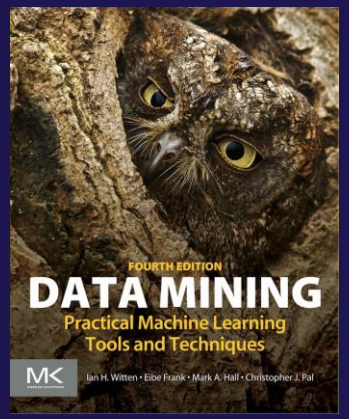
2. Understanding that data representation is one key to success



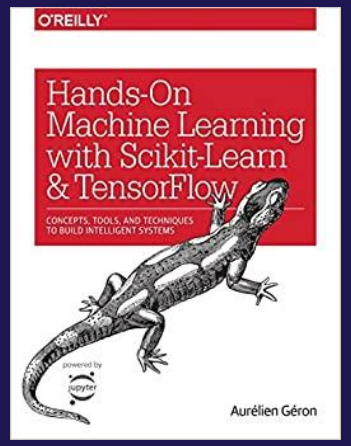
3. … and that knowing how to validate is the other!



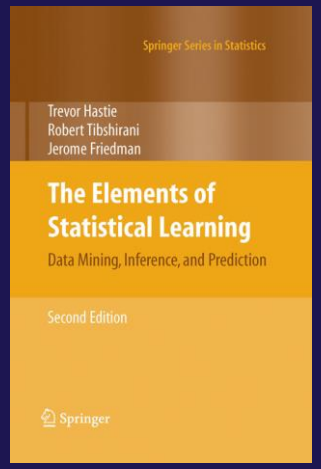
Where to next? I like reading books…



I learned with a previous version of this book,
and liked the level of mathematics vs application.
- It teaches Weka, which is Java based



Recommended by Bethany Lusch (ALCF).
Similar focus on application, but teaches scikit-learn



Deeper dive into the mathematics,
very thorough coverage of key algorithms