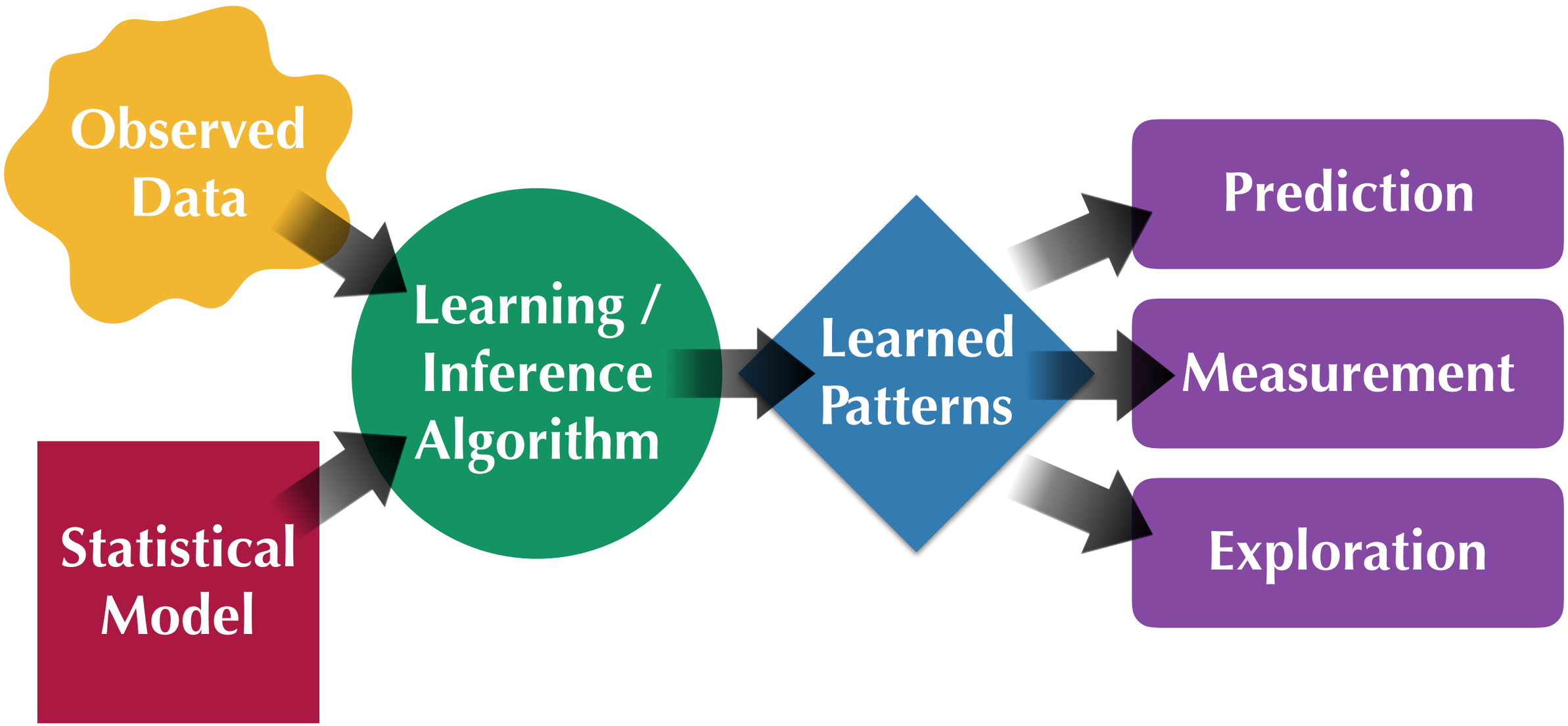
Introduction to **Machine Learning Methods**: What you Need to Know to **Conduct** and **Interpret** Research with ML

Allison J.B. Chaney
ajbc.io/MLintro

DUKE FUQUA SCHOOL OF BUSINESS

What is Machine Learning?



ajbc.io/MLintro

How do I want to use ML?

Prediction

Does person *A* belong to segment *B*?

What will revenue be if we change *X*?

Measurement

What products are perceived as *Y*?

How many people care about idea *M*?

Exploration

How many communities in network *N*?

What themes exist in reviews for *P*?

Outline

Part 1: Overview of Machine Learning

- Survey of model types
- Algorithms
- Software

Outline

Part 2: Case Studies (Methods and Challenges)

- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Part 1:
Overview of Machine Learning

Types of ML Models

Supervised vs. Unsupervised

Types of ML Models

Supervised vs. Unsupervised

Types of ML Models

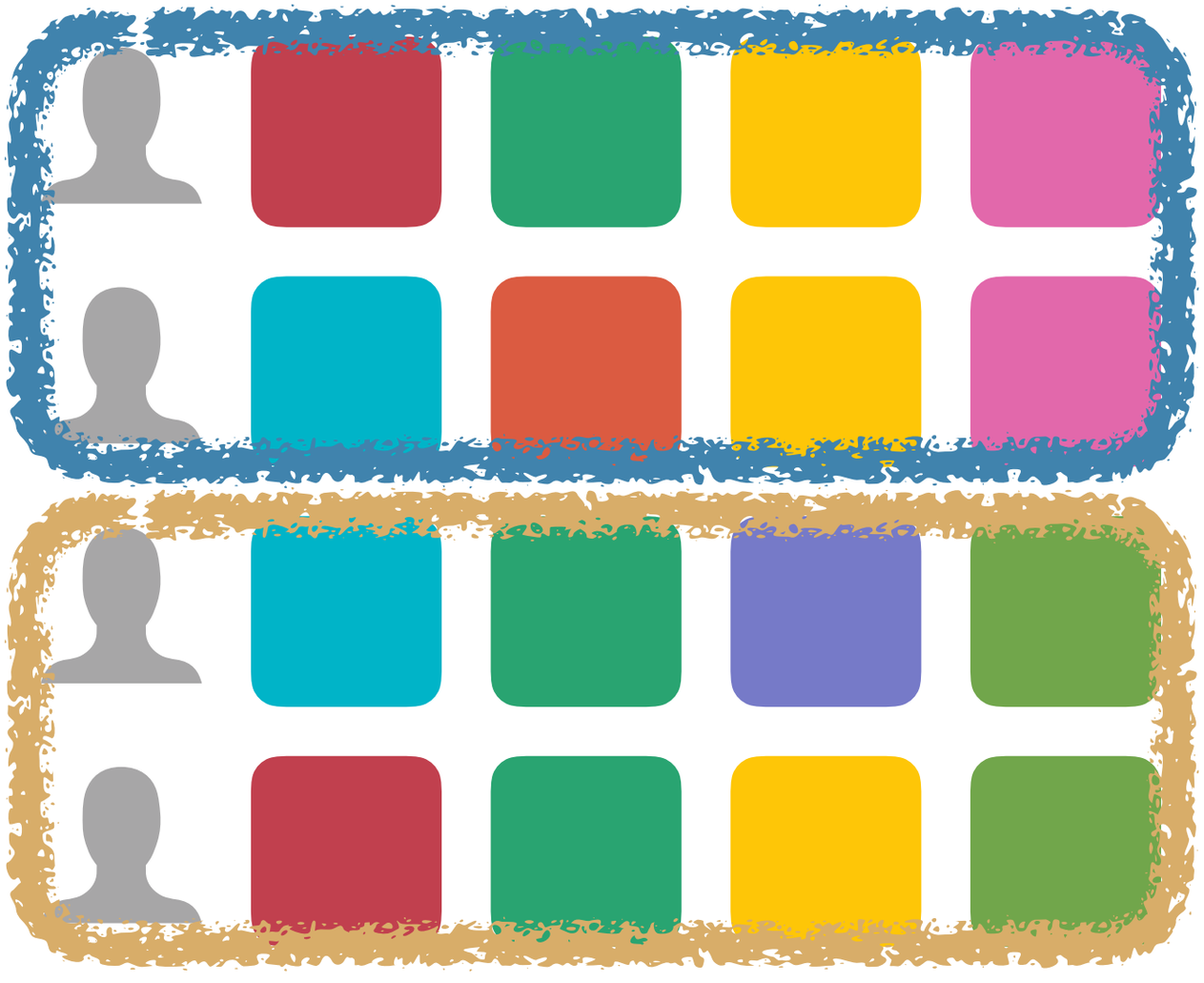
Supervised vs. Unsupervised

Types of ML Models

Supervised vs. Unsupervised

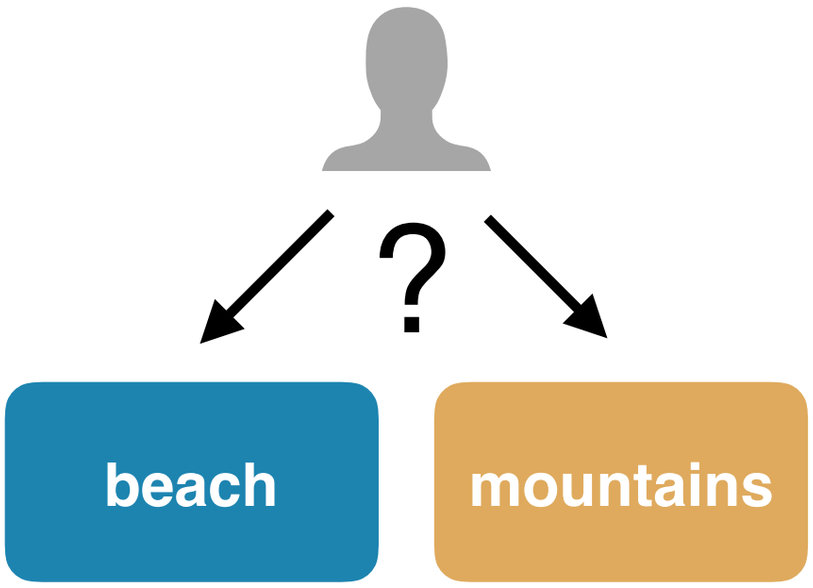
Types of ML Models

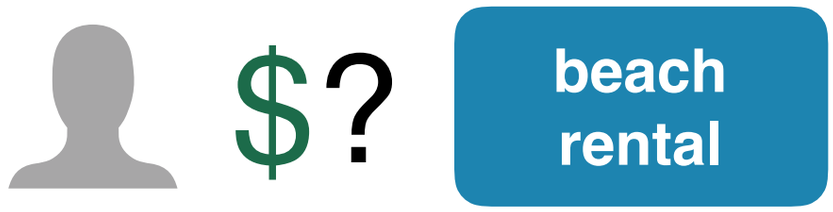
Supervised vs. **Unsupervised**



Types of ML Models

Discrete vs. Continuous





Types of ML Models

One Useful Grouping

	discrete	continuous
supervised	**classification**	**regression**
unsupervised	**clustering**	**dimensionality reduction**

Algorithms

	discrete	continuous
supervised	**classification**	**regression**
unsupervised	**clustering**	**dimensionality reduction**

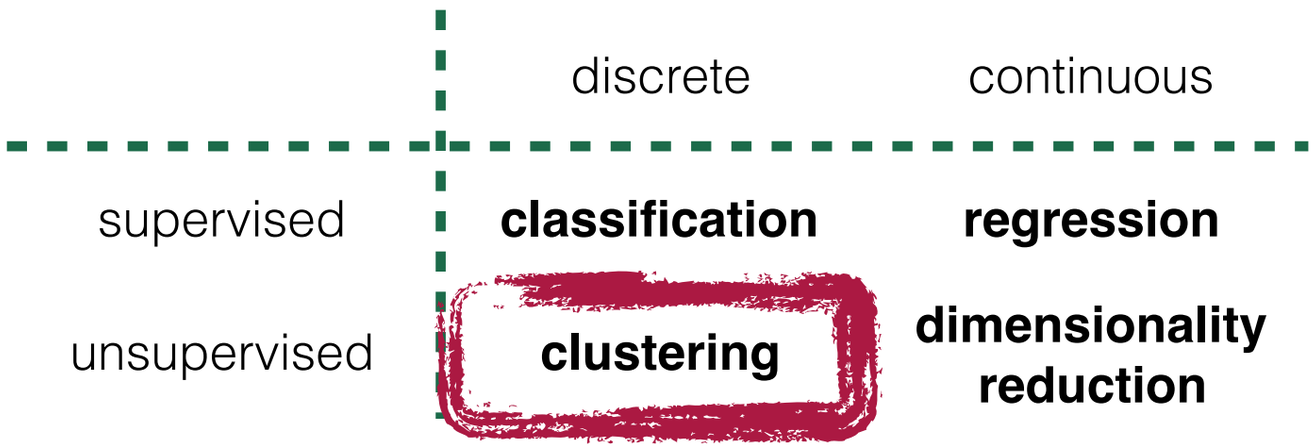
- Neural Networks / Deep Learning
- Decision Trees / Random Forests
- Boosting (ensemble method)
- Support Vector Machines
- ...

Algorithms

	discrete	continuous
supervised	**classification**	**regression**
unsupervised	**clustering**	**dimensionality reduction**

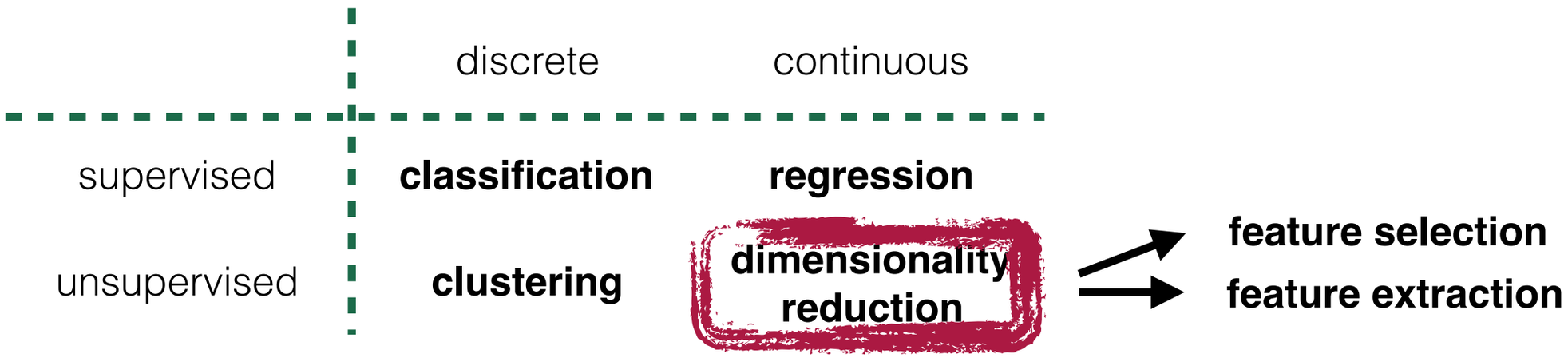
- Least squares
    - Regularization: Ridge, LASSO, ElasticNet
- Neural Networks & Support Vector Machines (again!)
- …

Algorithms



- *k*-means & fuzzy *k*-means (centroid-based)
- Expectation–Maximization (EM) using Gaussian Mixture Models (GMM) (distribution-based)
- DBSCAN (density-based)
- ...

Algorithms

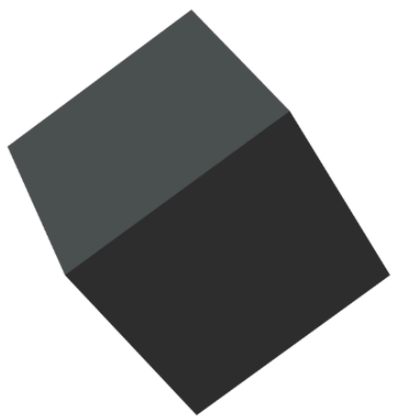


- Principal component analysis (PCA)
- Non-negative matrix factorization (NMF)
- Linear discriminant analysis (LDA)
- Autoencoder (neural network variant)
- ...

Software

VOWPAL WABBIT

Edward



https://scikit-learn.org/stable/
https://www.tensorflow.org/
http://edwardlib.org/
http://hunch.net/~vw/
https://cran.r-project.org/web/views/MachineLearning.html
...

Part 2:
Case Studies

Outline

Part 2: Case Studies (Methods and Challenges)

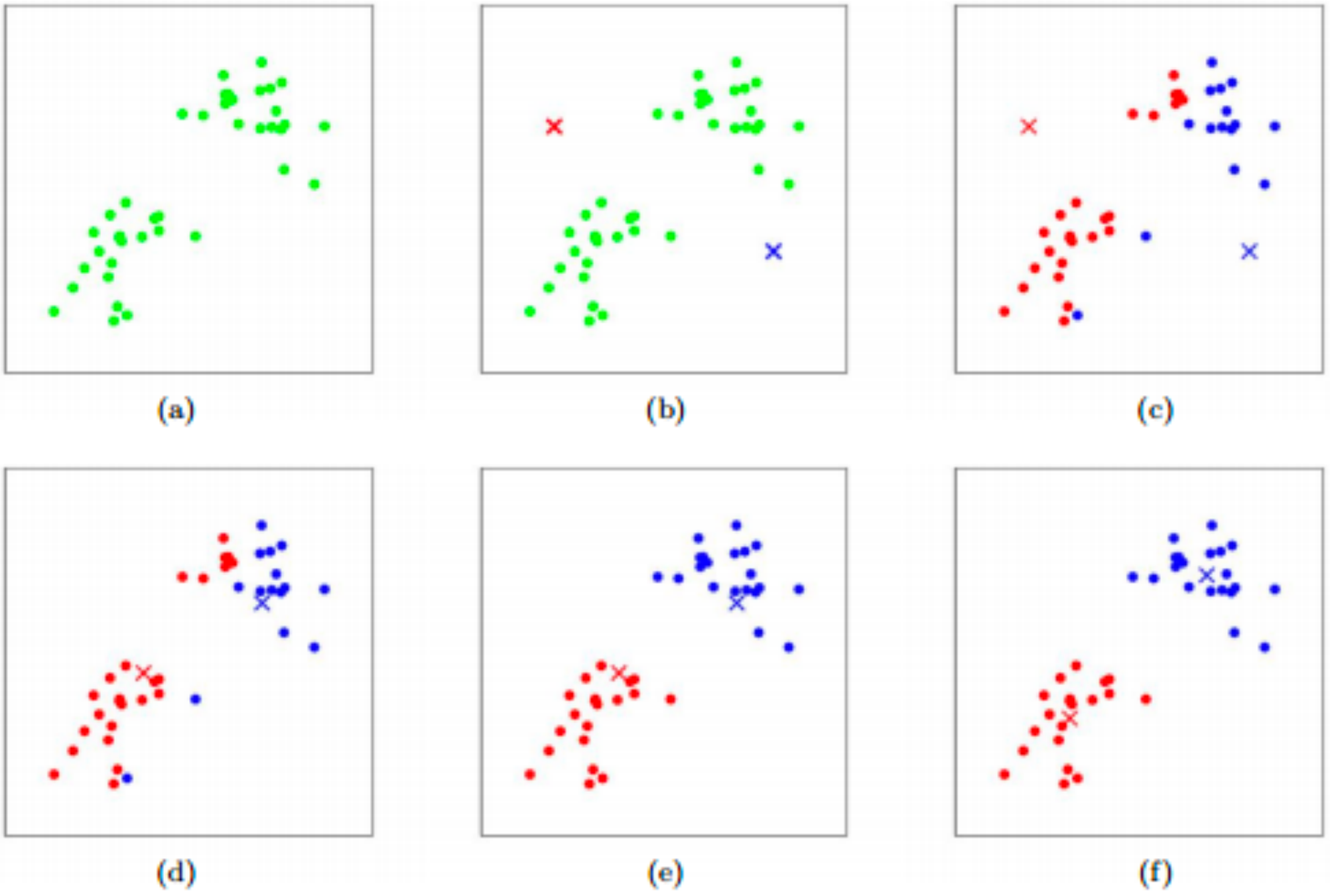
- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Outline

Part 2: Case Studies (Methods and Challenges)

- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

K-means clustering



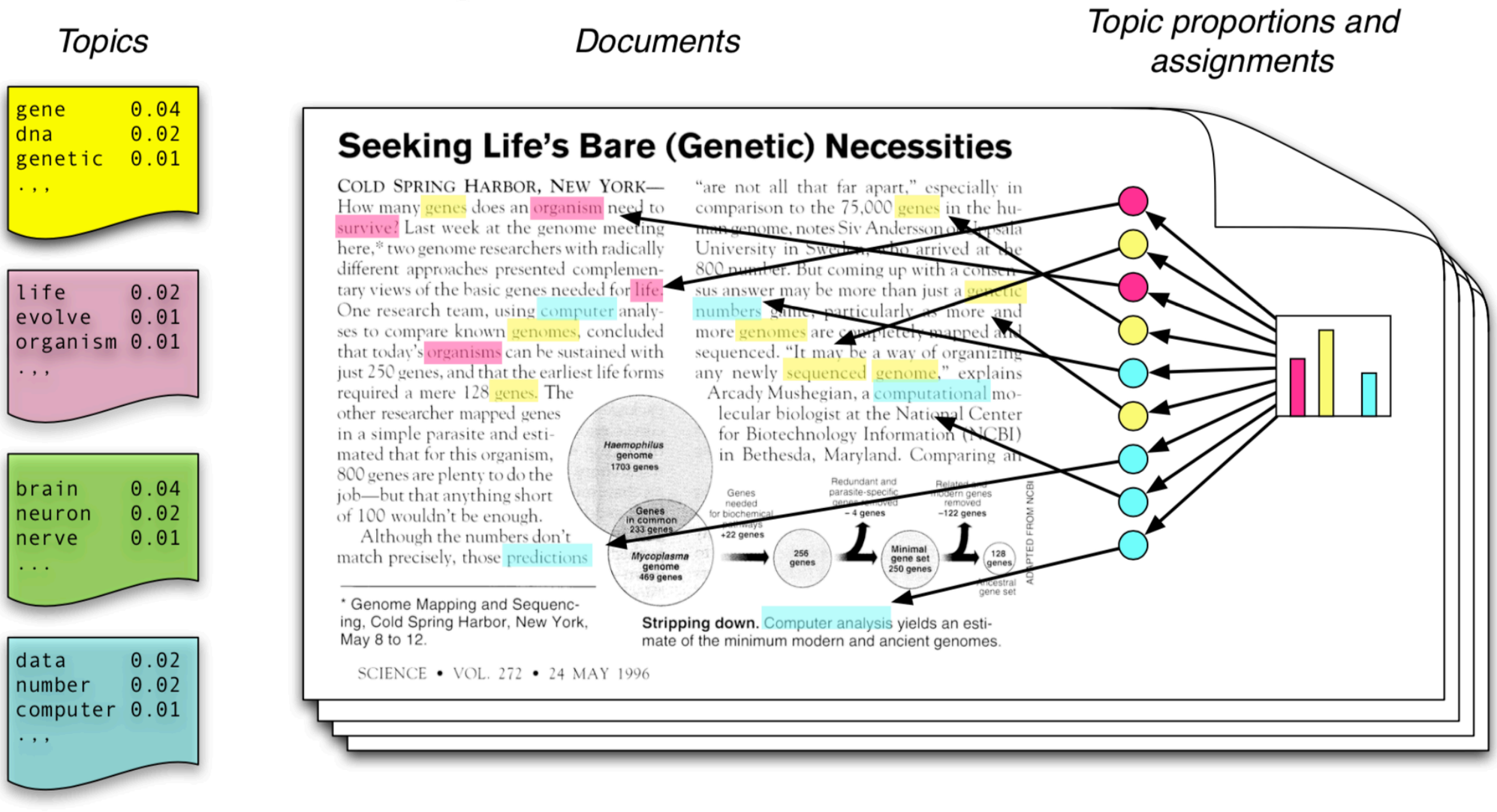
Let's head over to a Jupyter notebook...

Outline

Part 2: Case Studies (Methods and **Challenges)**

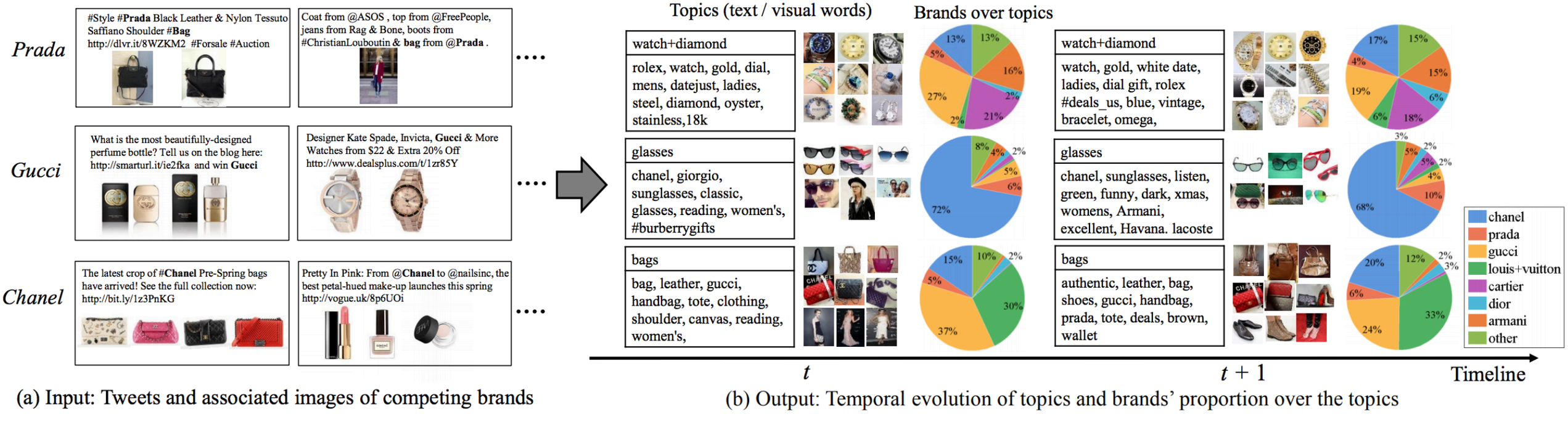
- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Topic Models



- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of those topics

Topic Models



(a) Input: Tweets and associated images of competing brands

(b) Output: Temporal evolution of topics and brands' proportion over the topics

Zhang, Kim, & Xing, 2015. Dynamic Topic Modeling for Monitoring Market Competition from Online Text and Image Data.

Challenge: Data Processing

"Ozymandias," Shelley's famous poem, reveals the impermanence of human achievement. The poem describes a crumbling statue, a "colossal wreck" in the form of a long-lost king. The reader of the poem is thrice-removed from Ozymandias, as the speaker relates a story he heard from a traveller who encountered the statue in the desert. A plate beneath the statue reads "Look on my works, ye Mighty, and despair!" Though Ozymandias presumably means that other mighty kings should despair at their inability to match his strength, the statement ironically evokes despair in the readers of the poem by reminding them of the impermanence of human works.

The traveller describes the shattered statue, abandoned to sink in the desert. He begins building the image of the statue by emphasizing its size, referring to it as "colossal" and "vast." Early in the poem, this description serves to create a sense of the grandness of the statue and the story, but later it will create the sense that even incredible achievements will be lost to time. While the statue's face still conveys something of Ozymandias's nature, it, too, ultimately reinforces the impermanence of human works. By describing the sculptor's skill ("its sculptor well that passion read"), the speaker begins to build the "despair" central to the poem. Neither the might of a king (Ozymandias) nor the skill of an artist (the sculptor) allows the monument to survive the test of time.

The poem separates the reader from Ozymandias: it does not describe the king himself, but the speaker hearing a traveller tell of a statue he saw in the desert. This separation is central to the sense of impermanence in the poem. If the poem exposed the reader to Ozymandias's mightiness, it might lend a sense of meaning to Ozymandias's works. Instead, the poem reveals the ephemeral nature of power and artistry by separating the reader from both the king and his monument. Even though Ozymandias was seemingly powerful enough to build the statue, the speaker only hears of him through happenstance. If the speaker had never met the traveler, the traveler had never found the statue, or Ozymandias had never commissioned the statue, the speaker might have never heard of Ozymandias, let alone experienced a sense of his might. This discovery of Ozymandias by chance, coupled with the separation of the speaker from the king, create the sense of loss around Ozymandias's works.

Beneath the statue, on the pedestal, a placard reads "My name is Ozymandias, king of kings:/ Look on my works, ye Mighty, and despair!" When dictating this placard, Ozymandias surely intended to proclaim his might to anyone drawing near the statue. The phrase "king of kings" demonstrates that he was very powerful, perhaps more akin to an emperor than the prince of a nation-state. While the command to "despair!" once implored his subjects and enemies to dread his power, it now implores the reader to despair at fleeting nature of humanity. Through decay, time inverts this statement to imply that no matter how powerful you are, or how great your works, you will eventually fade into obscurity.

A sense of the impermanence of human achievement permeates this poem. The poem's focus on vastness helps evoke a sense Ozymandias' might, heightening the reader's "despair" at the statue's "decay." By distancing the reader from Ozymandias's power through layers of storytellers (the sculptor, the traveller, and the speaker), and the ironic statement engraved on the statue's pedestal, the poem reveals time's dominance over all human works, including

Tokenizing raw text into terms

For clean text, this is pretty easy:

```
doc = doc.lower() # make everything lower case
doc = re.sub(r'-',' ', doc) # turn hyphens into spaces
doc = re.sub(r'[^a-z ]','', doc) # get rid of all punctuation
doc = re.sub(r' +',' ', doc) # turn multiple spaces into only one
words = doc.split() # split by spaces
```

Tuples and n-grams

“gold watch”

- gold, and watch
- (gold, watch)

- “The restaurant has an exclusive yet laid back feel.”
  - restaurant, exclusive, yet, laid, back, feel
  - (restaurant, exclusive, yet), (laid, back, feel)

Hyphens are tricky

- Some people include and some don't. It really depends on your corpus.

- What would you want to do for each of these?

logisitic-normal	mother-in-law	post-Aristotelian	obser-vations
x-ray	pre-eminent	pre-1900	hel-met
mean-field	dis-abled	50-year-old	… go---I will…
non-sequitur	re-cover	20-30 people	two-thirds
camera-ready	co-op	two- or threefold	ex-wife
mid-July	wind-proof	semi-invalid	mayor-elect

External options

- If you want to just let someone else do the leg work:
  - Stanford Tokenizer
  - Apache Open NLP
  - NLTK (python; interface to Stanford + others)
- IBM article The Art of Tokenization compares some of these options (and is generally a good resource)

Messier raw text

- Beautiful Soup for XML

```
from bs4 import BeautifulSoup

# open the document
xml = open(docfilename, 'r')
soup = BeautifulSoup(xml)
xml.close()

# find all the text
fulltext = soup.find("block", {"class":"full_text"})
paras = fulltext.findAll("p")
doc = ' '.join([p.contents[0] for p in paras])
```

Messier raw text

- Misspellings
  - easiest option is to ignore them
  - fix them using a spelling corrector; this might introduce different kinds of problems
- FTFY for Unicode encoding issues

Messier raw text

- Named entities
  - Many permutations of the same name: “Joe Smith,” “J. Smith,” “Joseph Smith,” “Joseph F. Smith”, just “Smith.”
- fuzzywuzzy to help find fuzzy matches
- Stanford Named Entity Recognizer (available via NLTK)

```
>>> from nltk.tag import StanfordNERTagger
>>> st = StanfordNERTagger('english.all.3class.distsim.crf.ser.gz')
>>> st.tag('Rami Eid is studying at Stony Brook University in NY'.split())
[('Rami', 'PERSON'), ('Eid', 'PERSON'), ('is', 'O'), ('studying', 'O'),
 ('at', 'O'), ('Stony', 'ORGANIZATION'), ('Brook', 'ORGANIZATION'),
 ('University', 'ORGANIZATION'), ('in', 'O'), ('NY', 'LOCATION')]
```

Curating a vocabulary

- exclude short words (generally < ~3 characters)
- minimum # of documents a word must be in
- maximum % of documents a word can be in
- general vocab curation script

Curating a vocabulary

- TF-IDF
  - top N words (e.g., 1000), by TF-IDF
  - pick a threshold manually

$$\text{tfidf}(w) = (\text{total \# of times } w \text{ occurs}) \times \log \frac{\text{total \# of documents}}{\text{\# of docs in which } w \text{ occurs}}$$

Curating a vocabulary

- External stop / common words lists
  - Ranks NL (multiple languages)
  - Word frequency (need to set a threshold)

- Can exclude by part-of-speech (e.g., no adverbs)
  - the NLTK POS tagger is good for this

```
>>> from nltk.tag import pos_tag
>>> from nltk.tokenize import word_tokenize
>>> pos_tag(word_tokenize("John's big idea isn't all that bad."))
[('John', 'NNP'), ("'s", 'POS'), ('big', 'JJ'), ('idea', 'NN'), ('is',
'VBZ'), ("n't", 'RB'), ('all', 'DT'), ('that', 'DT'), ('bad', 'JJ'),
('.', '.')]
```

Curating a vocabulary

- Stemming
  - via NLTK
  - faster but less interpretable
- Lemmatization
  - code snippet (uses NLTK/WordNet)
  - slower, more interpretable

original	stem	lemma
arguing	argu	argue
taller	tall	tall
better	bet	good
provision	provide	provide
cement	cem	cement
maximum	maxim	maximum

General Data Processing

- Think carefully about thresholding data; how will that impact your results? Be prepared to justify your choices.

- Test your data for properties of interest
  - E.g., in networks: density, centrality, or assortativity

  - Ask yourself: what properties should the data have? Are the properties suitable for both a given ML model and the task more broadly?

  - We can try this on different cuts of the data to understand it (e.g., only documents containing certain words)

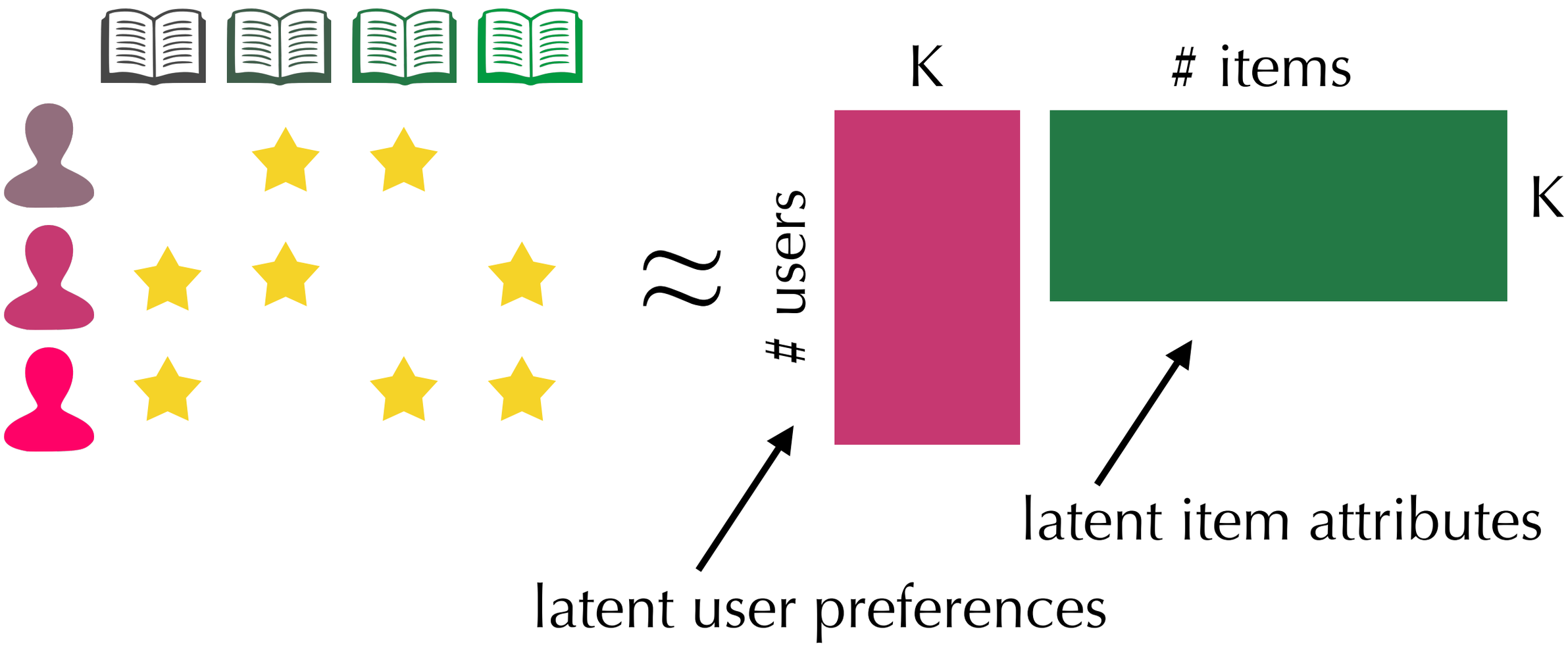
Back to Jupyter…

Outline

Part 2: Case Studies (Methods and **Challenges)**

- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Matrix Factorization



Evaluation Metrics

Task: Predict what users want to read

- Rating Accuracy
  - RMSE
  - MAE
- Rank-based
  - Precision @ N
  - Recall @ N
  - NDCG
  - Reciprocal Rank
  - …

Evaluation Metrics

What metrics matches the real-world setting the best?

Posterior predictive checks can also help us evaluate our models (& help validate our model assumptions)

PPCs compare observed data to a reference distribution based on the posterior (conditioned on the observed data)

More on PPCs: https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/ppc.pdf

Bonus Challenge!

How do we treat unobserved cells?

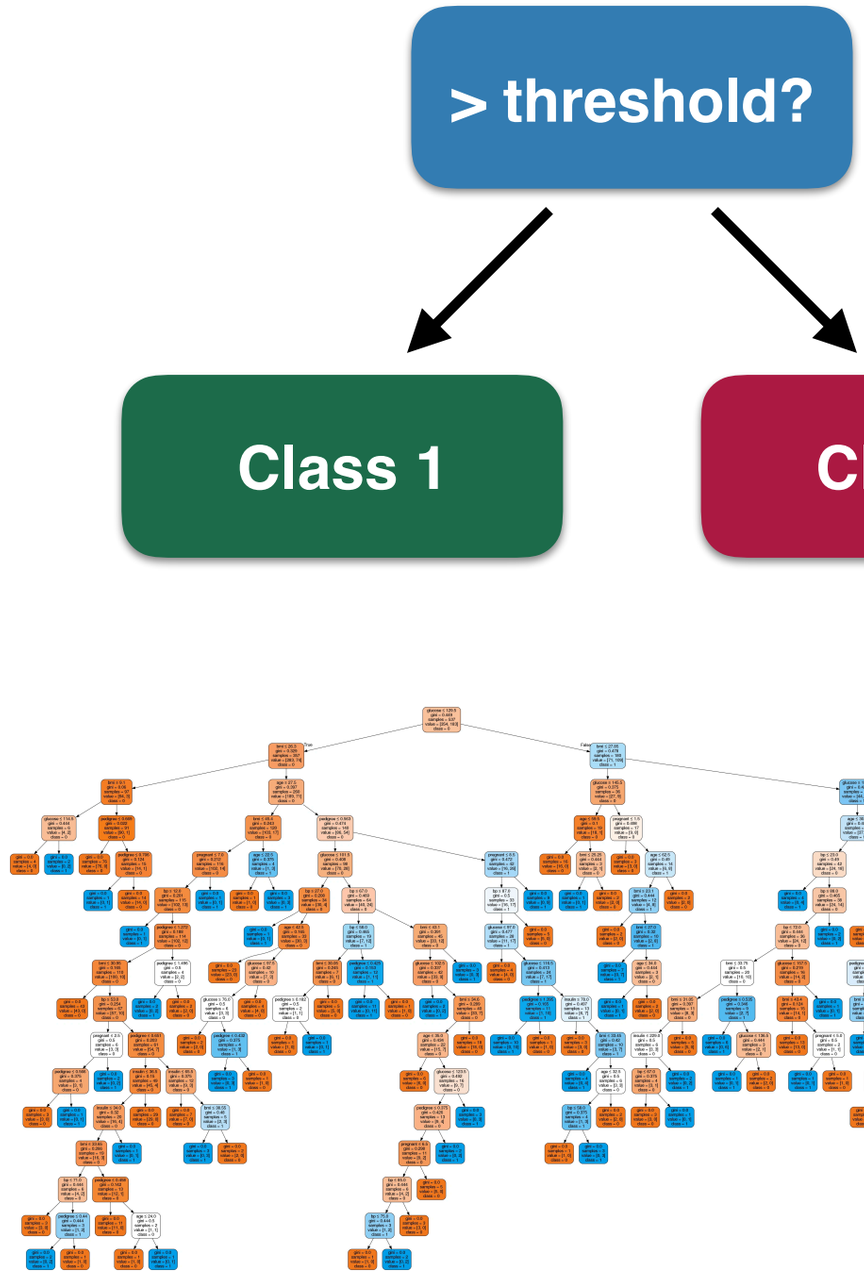
- Ignore them
- treat all as 0's
- subsample 0's
- subsample 0's and give them low weight
- impute values
- model missingness explicitly
- …

Outline

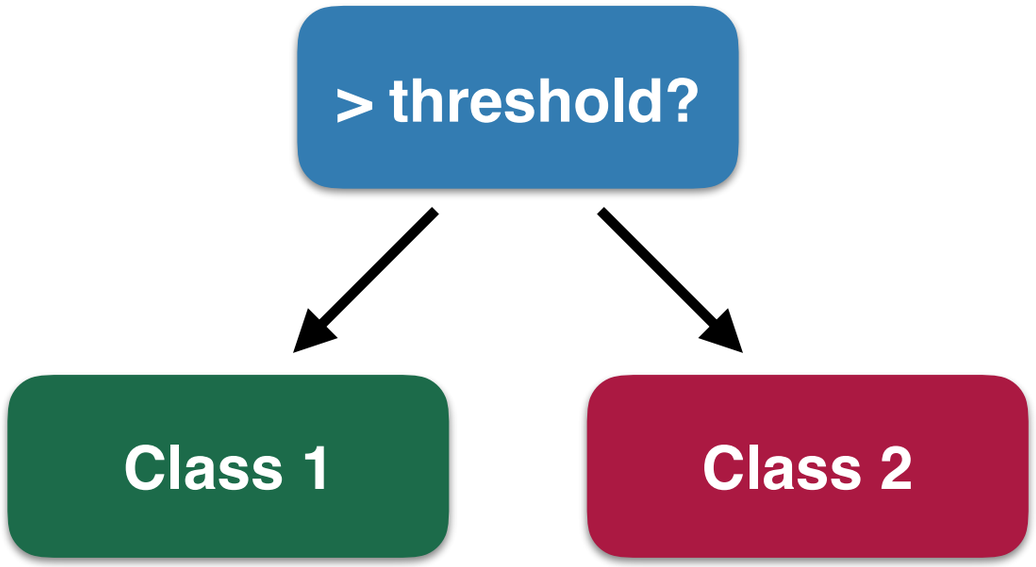
Part 2: Case Studies (Methods and **Challenges)**

- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Decision Trees



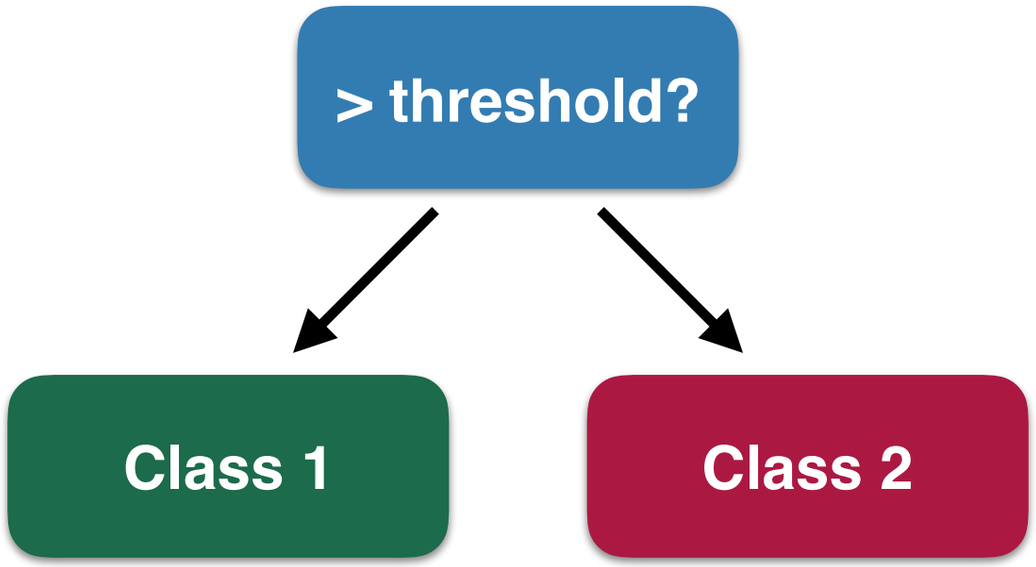
Decision Trees



Foundation for some popular ensemble methods, such as:

- Random Forests
- Boosting

Decision Trees



For both ensemble methods and complex trees,
overfitting is a concern!

Overfitting

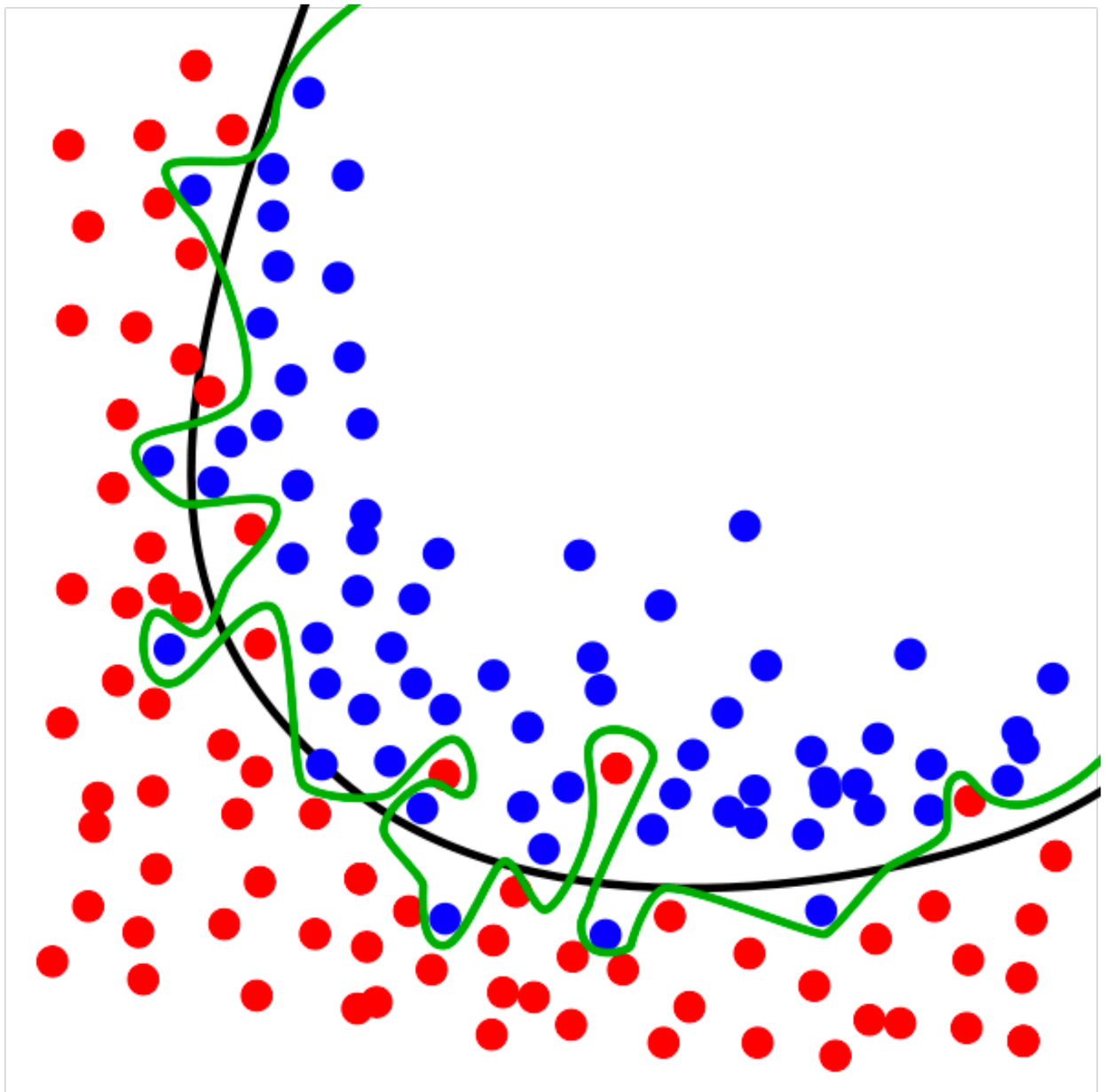
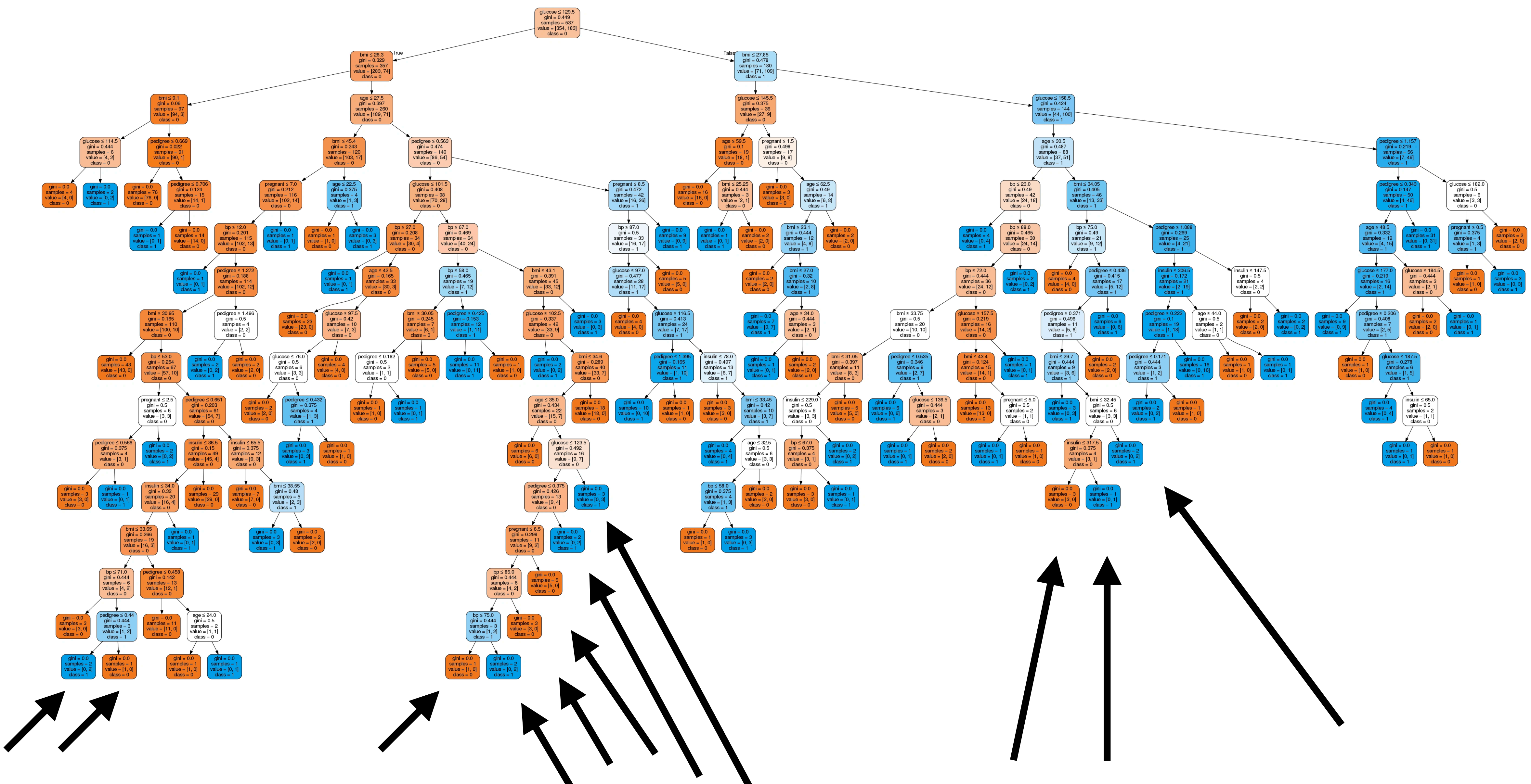


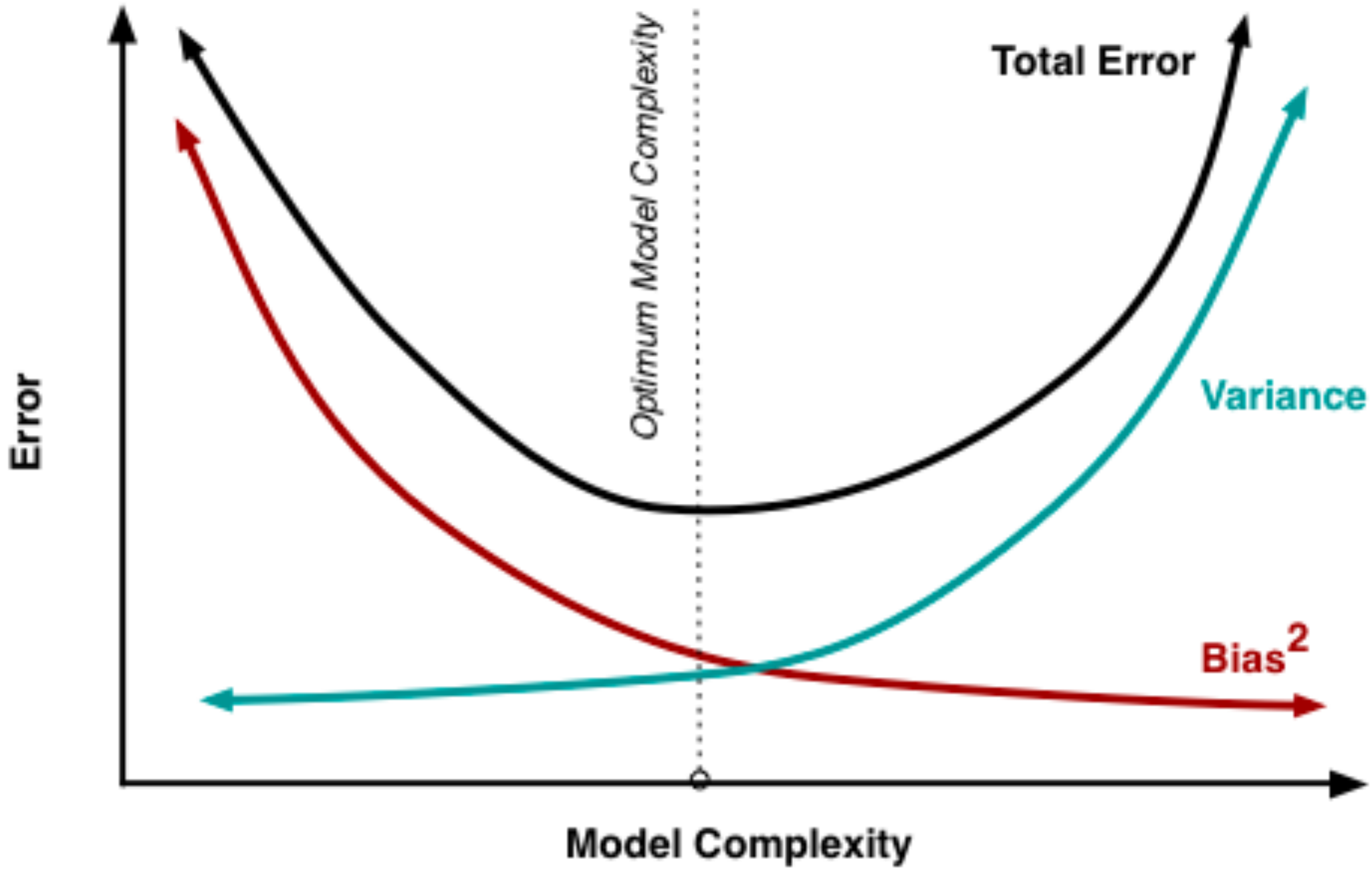
Image source: https://en.wikipedia.org/wiki/Overfitting

Overfitting



E.g., one terminal node for each observation

Solution: Model Selection



Model Selection

Model Selection

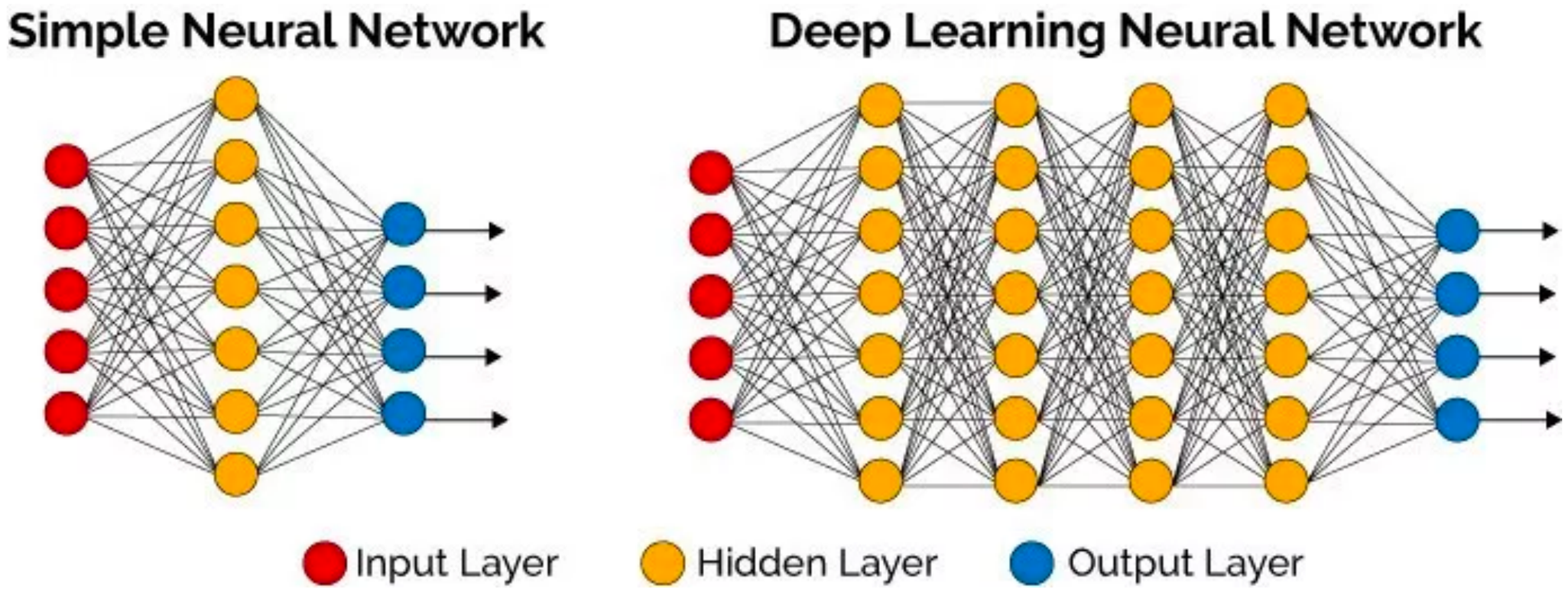
Model Selection

Outline

Part 2: Case Studies (Methods and **Challenges)**

- **K-means Clustering** and **Choosing K**
- **Topic Models** and **Data Processing & Exchangeability**
- **Matrix Factorization** and **Evaluation Metrics**
- **Decision Trees & Ensemble Methods** and **Overfitting & Model Selection**
- **Deep Learning** and **Learning Rates**

Deep Learning



Tutorial on Deep Learning:

https://towardsdatascience.com/deep-learning-for-beginners-practical-guide-with-python-and-keras-d295bfca4487

Deep Learning

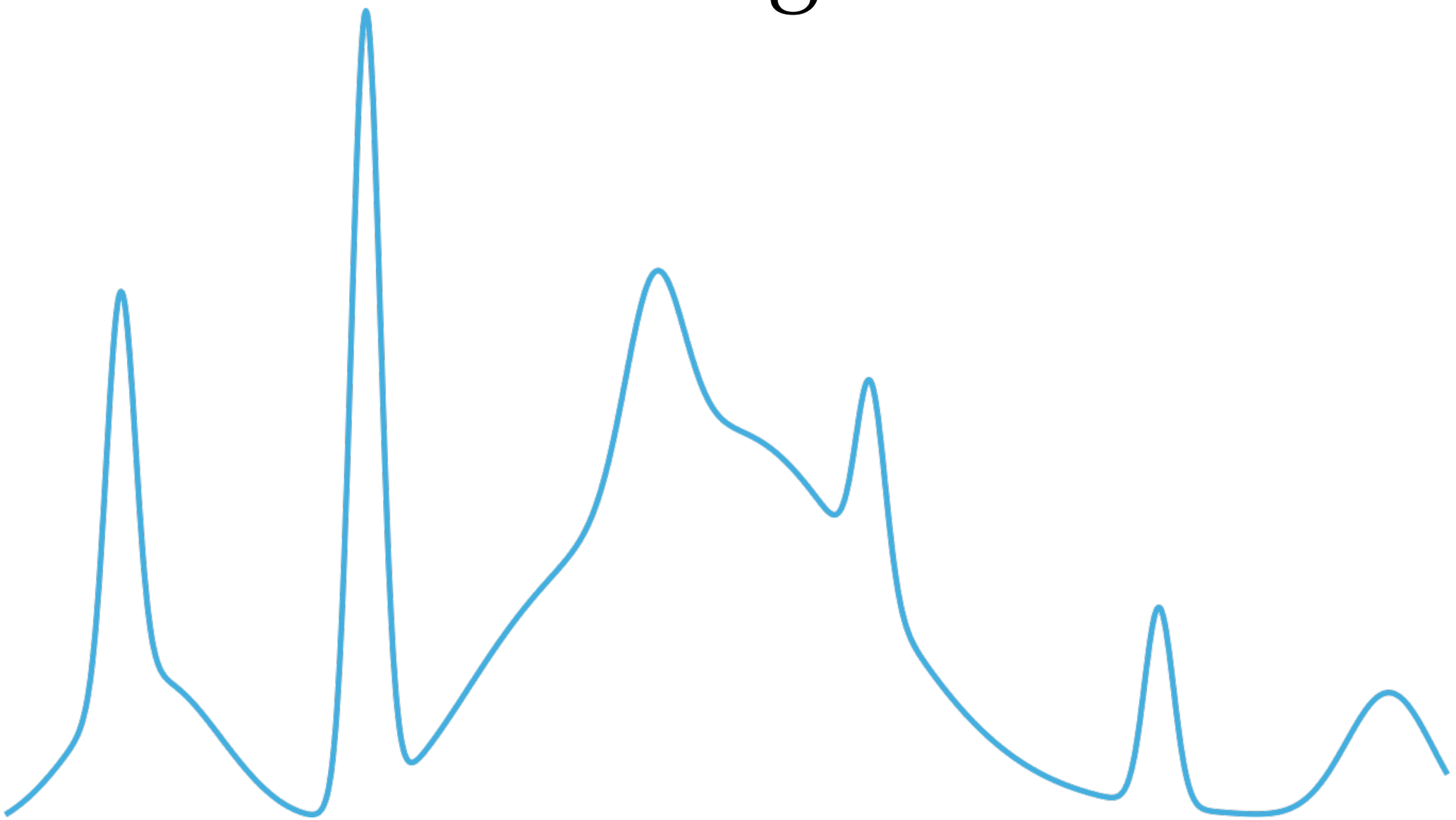
Learn / train with optimization algorithms, such as

- Stochastic Gradient Descent
- Adagrad
- Adam
- AdaDelta
- RMSProp

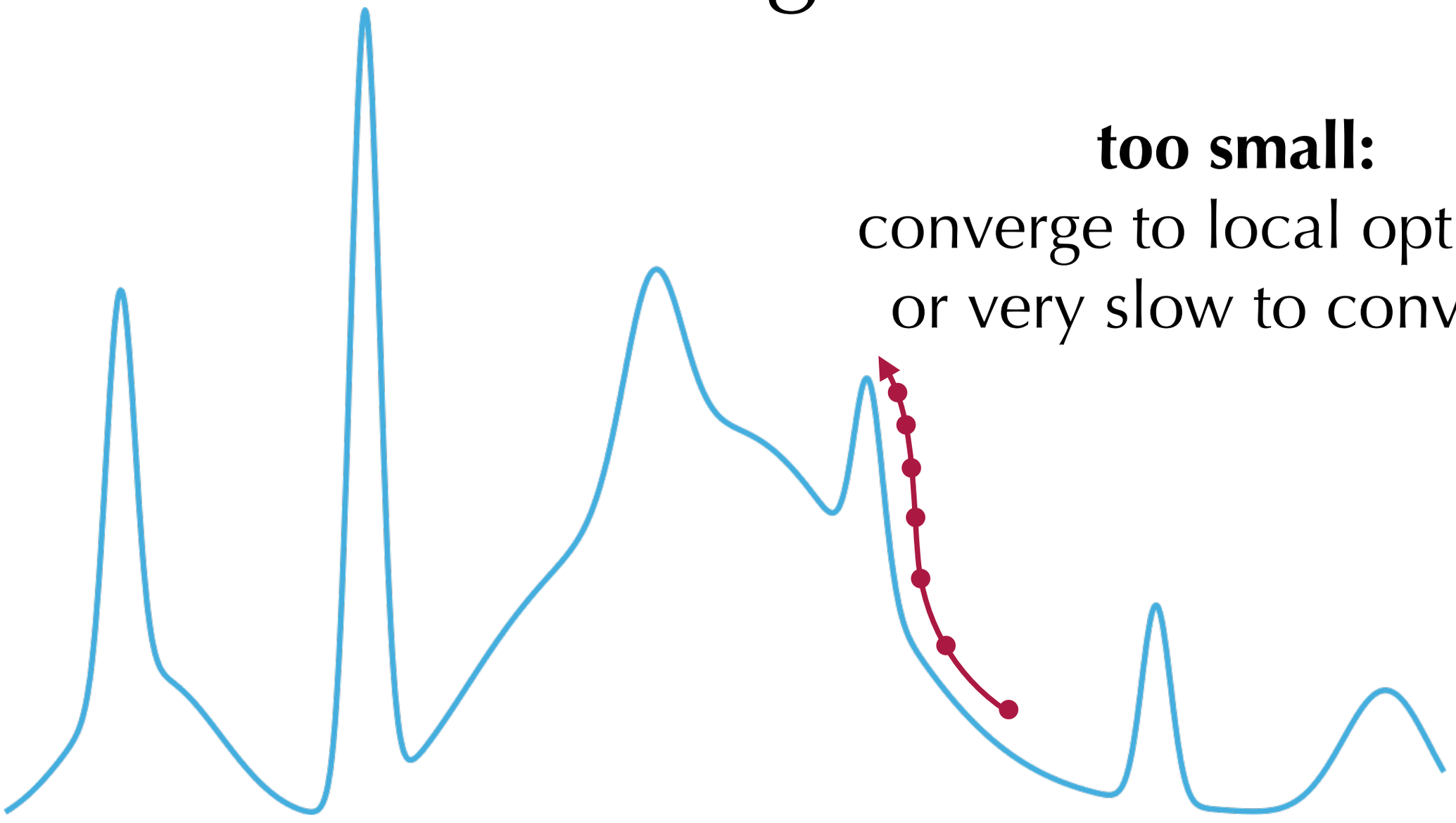
} **adaptive learning rates** & **different rates for different model parameters**

More here: https://rishy.github.io/ml/2017/01/05/how-to-train-your-dnn/

Learning rates

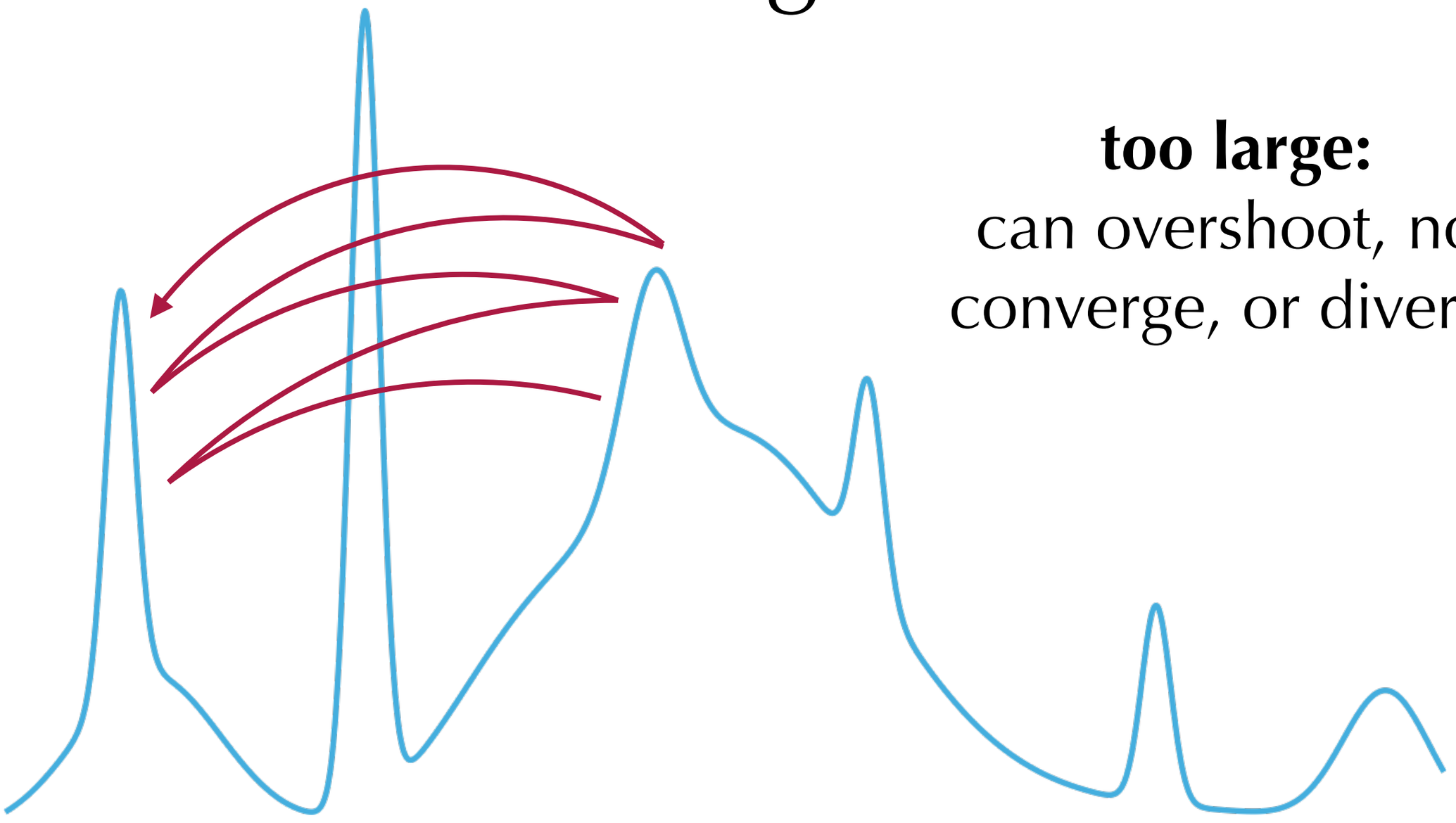


Learning rates



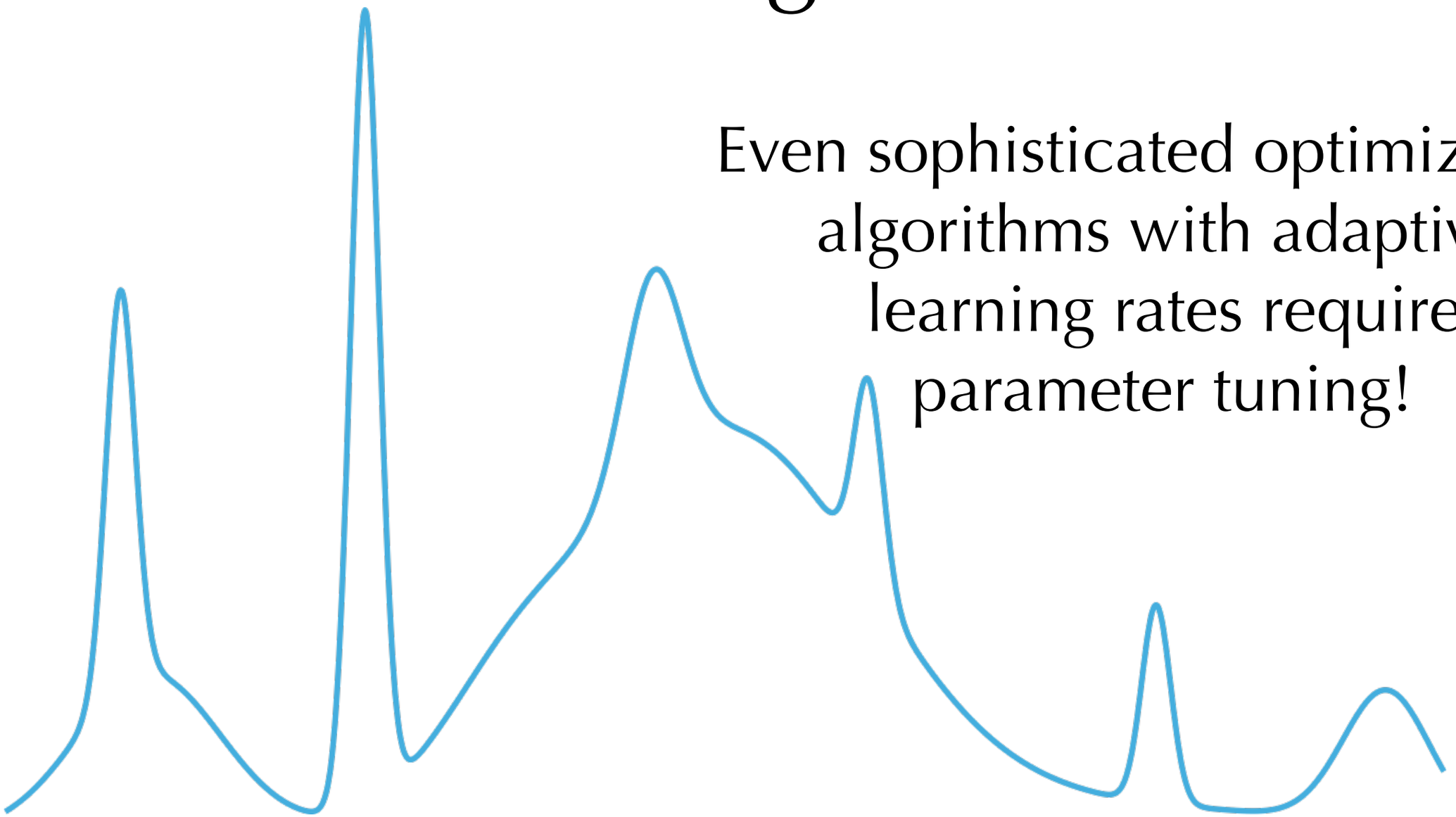
too small:
converge to local optimum
or very slow to converge

Learning rates



too large:
can overshoot, not converge, or diverge

Learning rates



Even sophisticated optimization algorithms with adaptive learning rates require parameter tuning!

Take Care!

Define the problem before choosing an ML method.

Be critical of ML model assumptions. Are a method's assumptions acceptable in the context of your work?

Ask yourself: What is the source of your data? What biases might be created by the data-generating process or your curation of the data?

Beware of tuning parameters, train/tune/test splitting, evaluation metrics, and overfitting.

What are you comparing against? Will a simple method do better?

Take Care!

If our goal is **prediction**, we need:

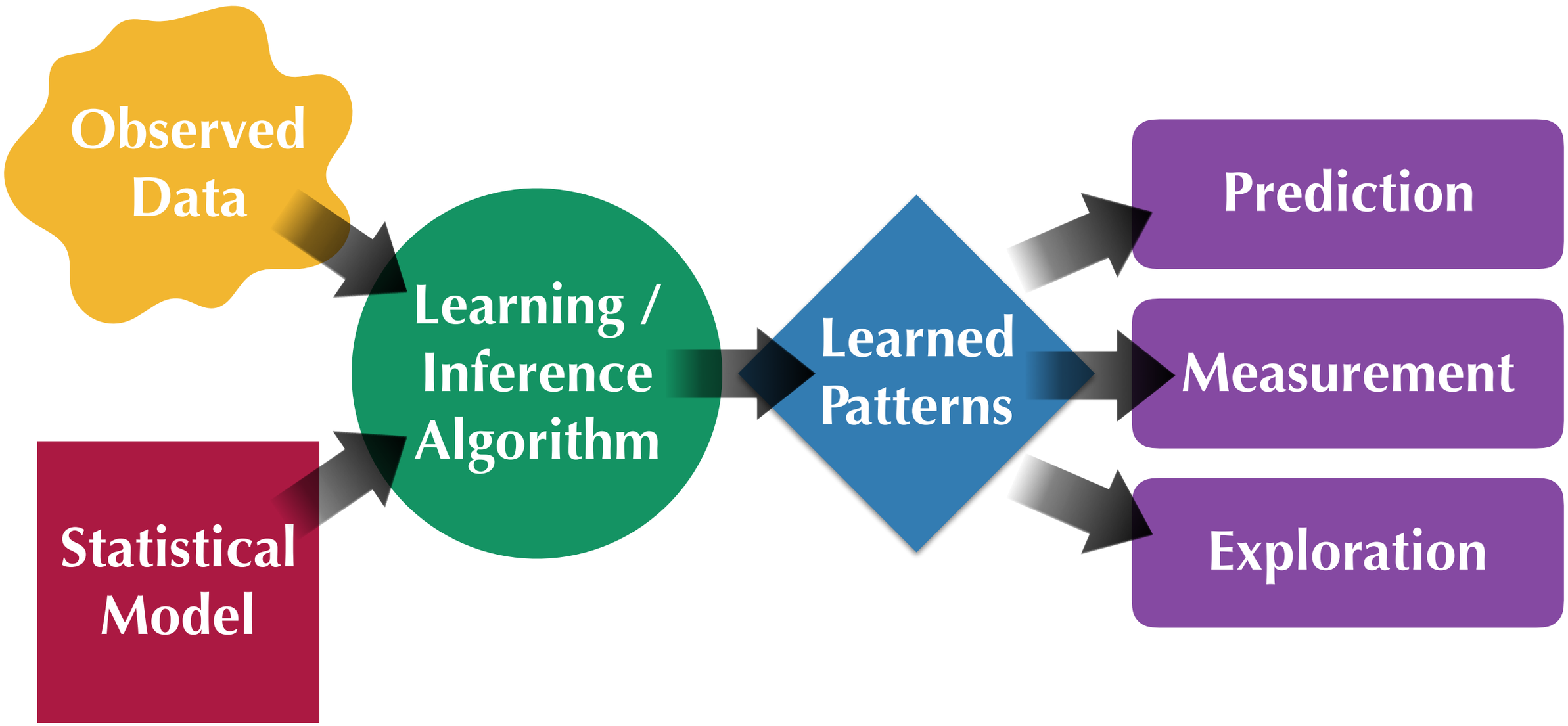
- valid modeling assumptions
- ensure that the model makes good predictions on **new data** (e.g., not used for training/tuning), using **metrics that match real-world use**

If our goal is **measurement** or **explanation** we need:

- valid modeling assumptions
- ensure that the inferred latent variables are **valid operationalizations** of the **theoretical constructs** we wish to measure

Exploration is a preliminary step for both of these where we ask:
What does the data tell me that I didn't already know?

Conducting and Interpreting Research with ML



ajbc.io/MLintro