INFO 4390/5390 / CS 5382: Designing Fair Algorithms

Lecture 2: 2024-01-25

Pierson & Koenecke

This lecture: make sure you have the machine learning knowledge you need for this class

This class is designed to be **broadly accessible** to students from diverse backgrounds (not just CS majors)

Purpose of this lecture: get everyone on the same page regarding machine learning concepts (theory-based)

For practical Python refreshers…

See **Canvas page** for:

- A Google Colab tutorial for basic skills we expect you to know going into this class (assignments can be done using only Colab)
- Python install / conda environment directions if you're currently using an older version of Python (maybe useful for group project)

For practical Python refreshers…

See **Canvas page** for:

- A Google Colab tutorial for basic skills we expect you to know going into this class (assignments can be done using only Colab)
- Python install / conda environment directions if you're currently using an older version of Python (maybe useful for group project)

If you're still having trouble with set-up:

- **Go to Office Hours! Our TAs are happy to help you**
- **Post questions on Ed Discussion**

Outline: machine learning refresher

1. Predictive algorithms / supervised learning
    a. What they are
    b. How to build them
    c. How to evaluate their performance

2. Beyond supervised learning
    a. Word embeddings
    b. Generative AI

What is a predictive algorithm?

What is a ~~predictive~~ algorithm?

***Algorithm*:** a precise set of instructions for doing something.

- “To convert Celsius to Fahrenheit, multiply by 9/5 and add 32”

What is a predictive algorithm?

***Algorithm*: a precise set of instructions for doing something.**

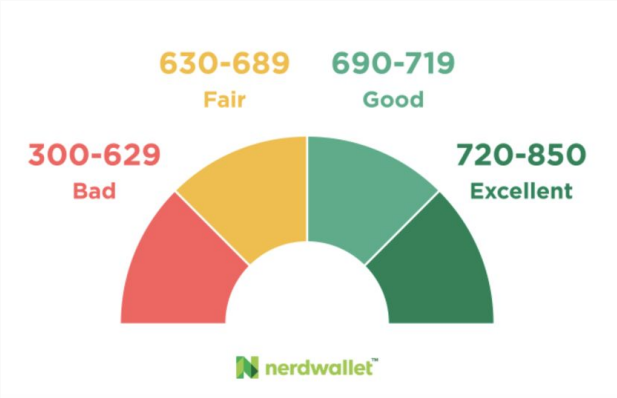
- "To convert Celsius to Fahrenheit, multiply by 9/5 and add 32"

***Predictive algorithm (in this course, a.k.a. "models")*: a precise set of instructions for *predicting an outcome*.**

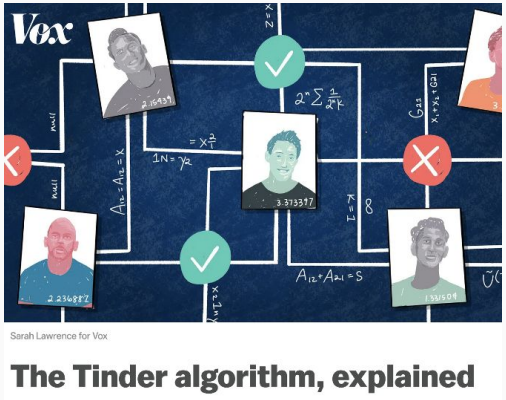
- "To predict whether it will rain tomorrow, feed the current weather data into this complex set of mathematical equations"
- "To predict someone's adult height, average their parents' height, and add 2.5 inches for a male child or subtract 2.5 inches for a female child"

Image sources: Nerdwallet, Vox, Framingham Heart Study/NHLBI

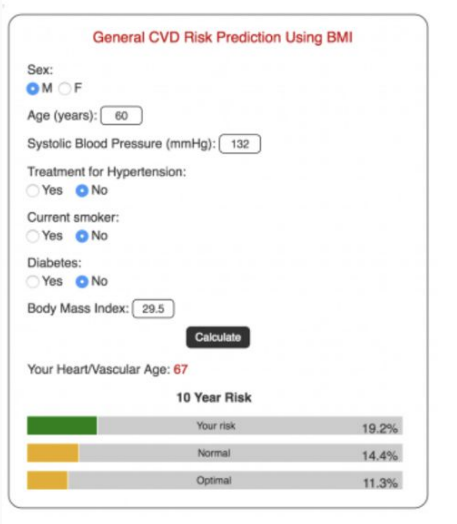
Recall from last class: predictive algorithms guide critical decisions!



Lending



Dating



Health

How do you actually build a predictive algorithm?

Let’s say we want to build an algorithm to predict whether there will be a beautiful sunset each night…

How do you actually build a predictive algorithm?

- **Core principle**: learn to predict the future by looking at the past

How do you actually build a predictive algorithm?

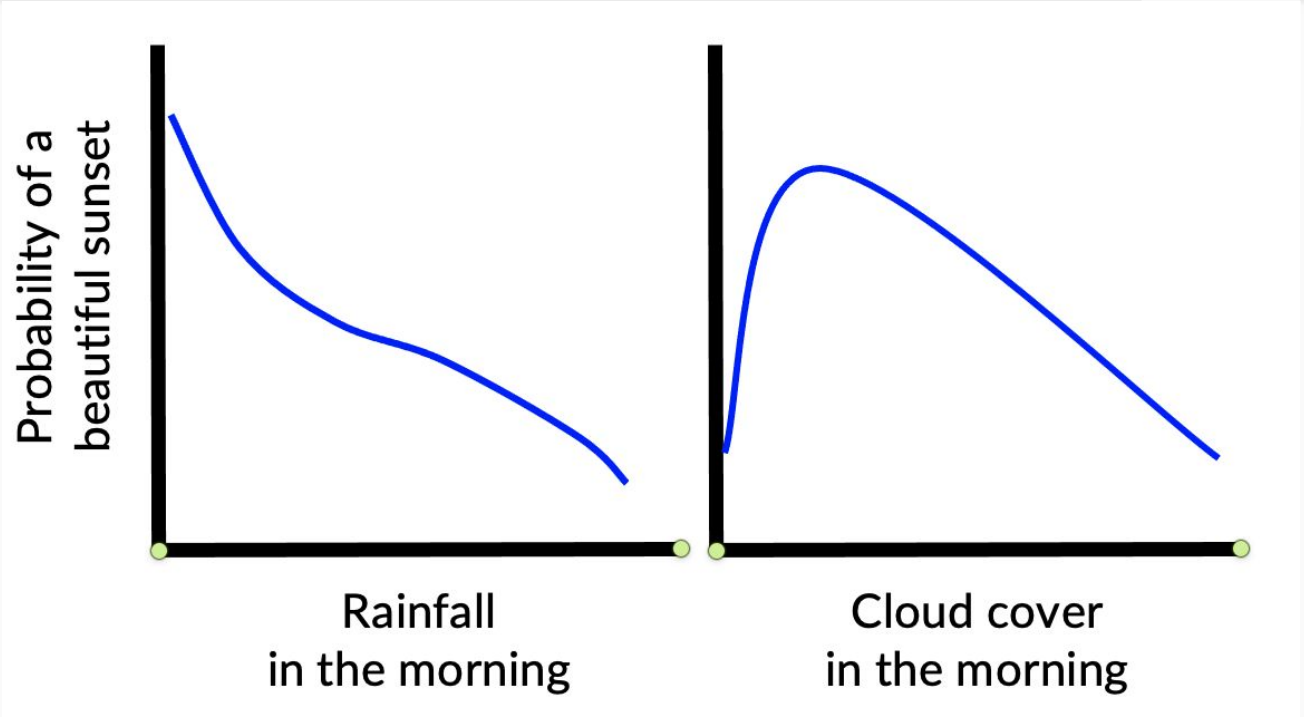
- **Core principle**: learn to predict the future by looking at the past

- **Example**: prediction problem:
  - *Features (or inputs)*: weather in the morning – temperature, pressure, cloud cover, wind speed, whether it's raining, etc…
  - *Outcome (or target variable)*: is there a beautiful sunset that night?

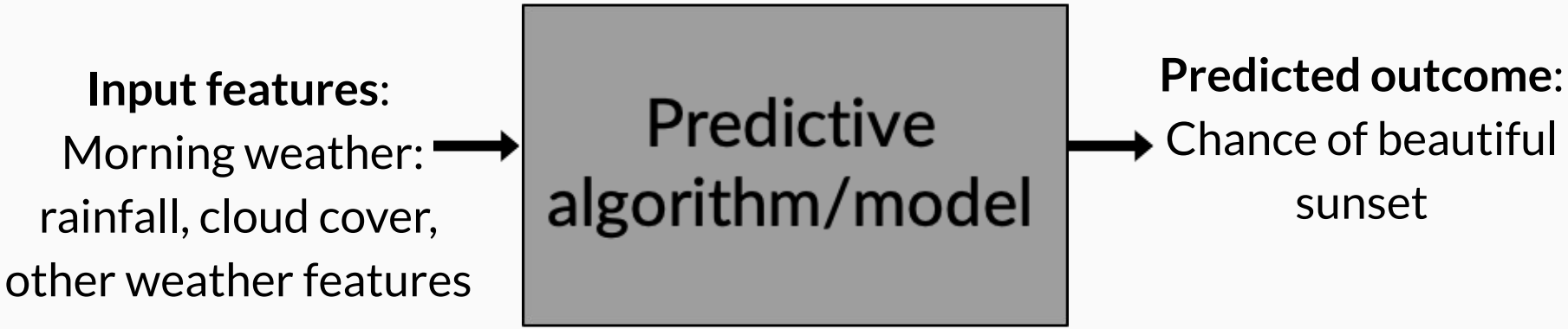
How do you actually build a predictive algorithm?

- **Core principle**: learn to predict the future by looking at the past
- **Example**: prediction problem:
  - *Features (or inputs)*: weather in the morning – temperature, pressure, cloud cover, wind speed, whether it's raining, etc…
  - *Outcome (or target variable)*: is there a beautiful sunset that night?
- **Need to do:** Gather a large historical dataset (train dataset)

How do you actually build a predictive algorithm?

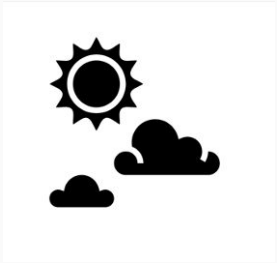


How do you actually build a predictive algorithm?



How do you actually build a predictive algorithm?

Input features:
Morning weather:
no rain, few clouds

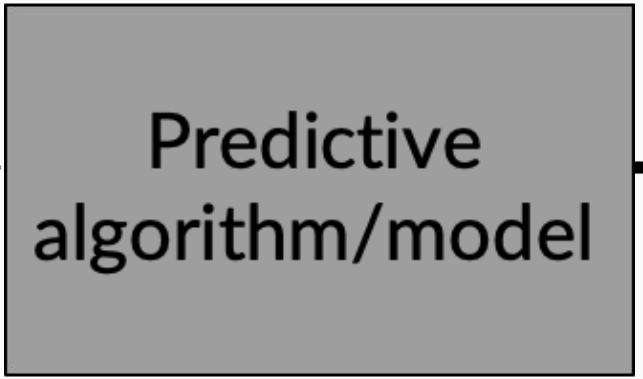


Predictive algorithm/model

Predicted outcome:
Chance of beautiful sunset: 60%

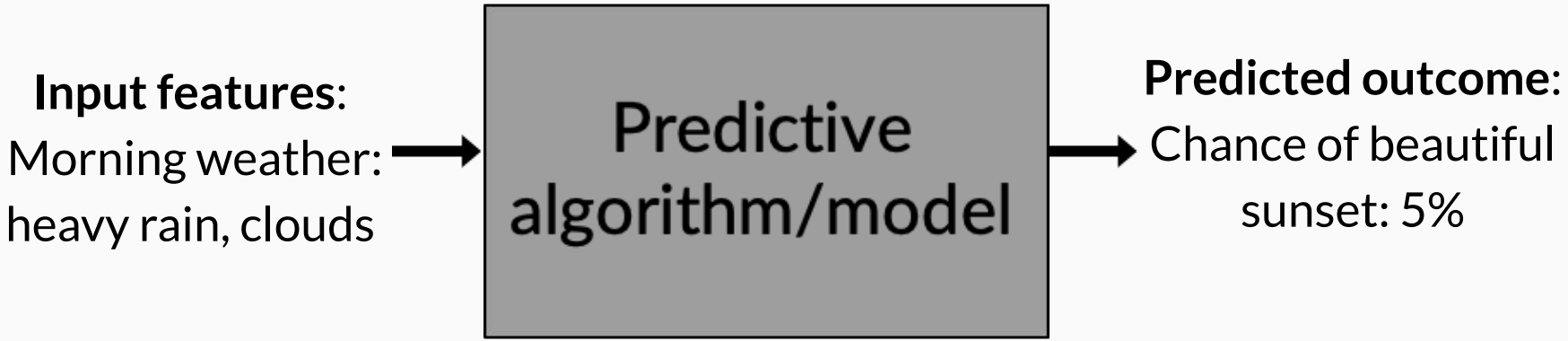
How do you actually build a predictive algorithm?

Input features:
Morning weather:
heavy rain, clouds



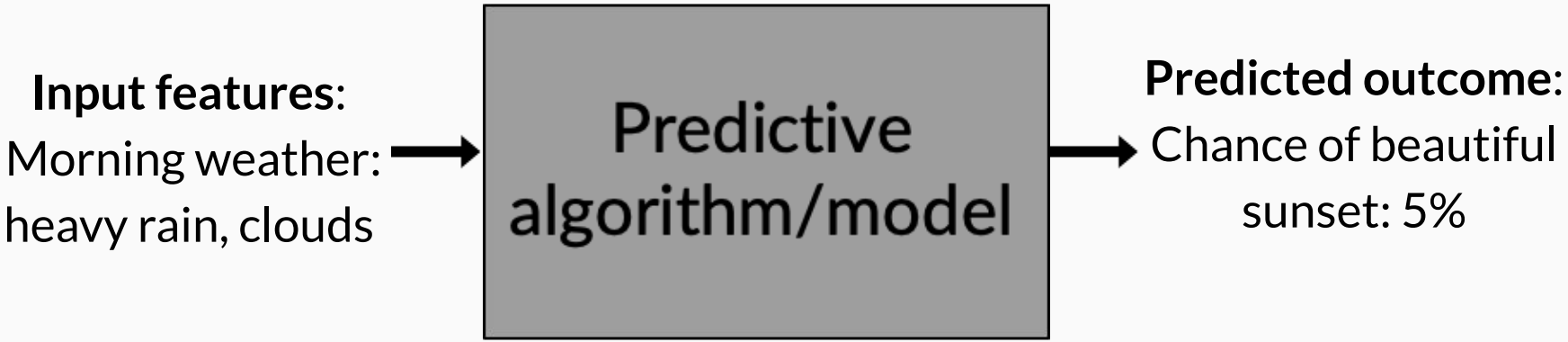
Predicted outcome:
Chance of beautiful sunset: 5%

How do you actually build a predictive algorithm?



You can convert predicted probabilities to yes/no predictions
by choosing a threshold (like 50%)
Chance of beautiful sunset 5% -> predict NO
Chance of beautiful sunset 60% -> predict YES

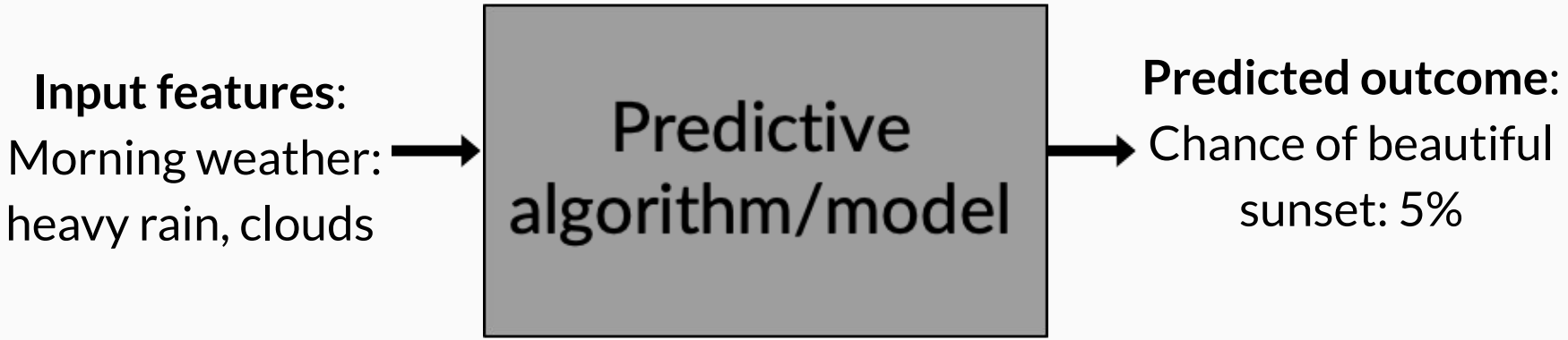
How do you actually build a predictive algorithm?



If you choose a **high** threshold (e.g. 90%), you might fail to predict some beautiful sunsets, but you won't often falsely predict the sunset will be beautiful when it won't

If you choose a **low** threshold (e.g. 10%), you won't miss many beautiful sunsets, but you'll more often falsely predict the sunset will be beautiful when it won't

How do you actually build a predictive algorithm?



This basic tradeoff between *false positives* and *false negatives* is one we'll return to later in the lecture

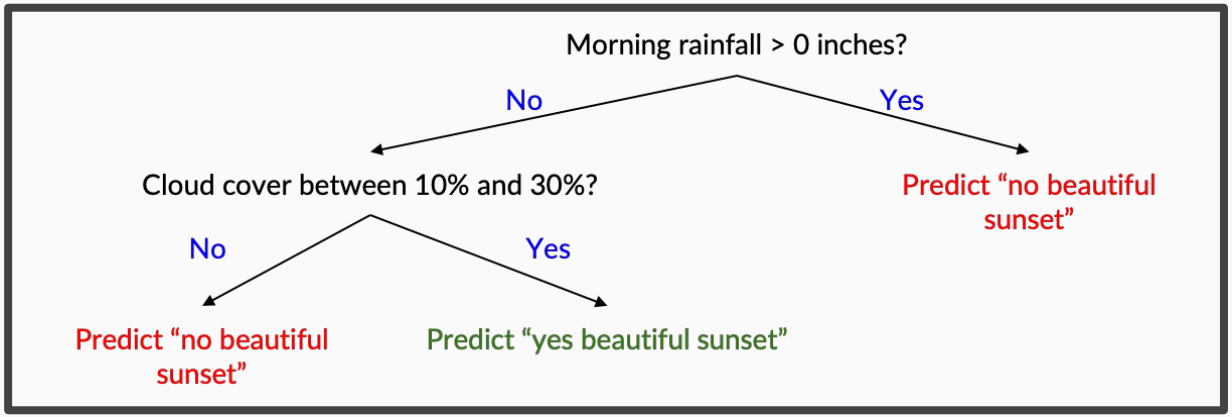
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



This type of model is called a "decision tree"

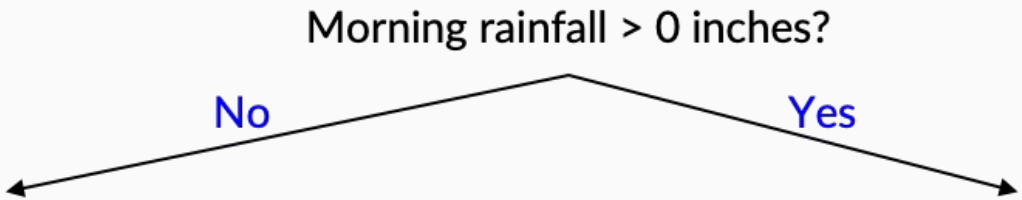
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



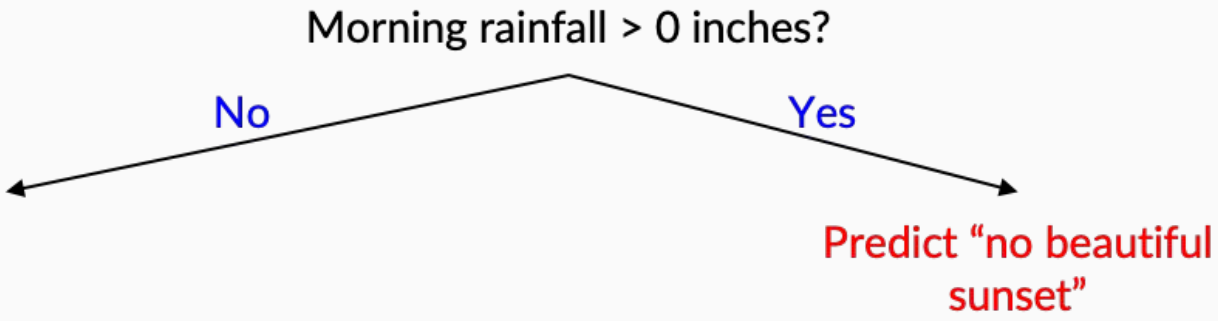
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



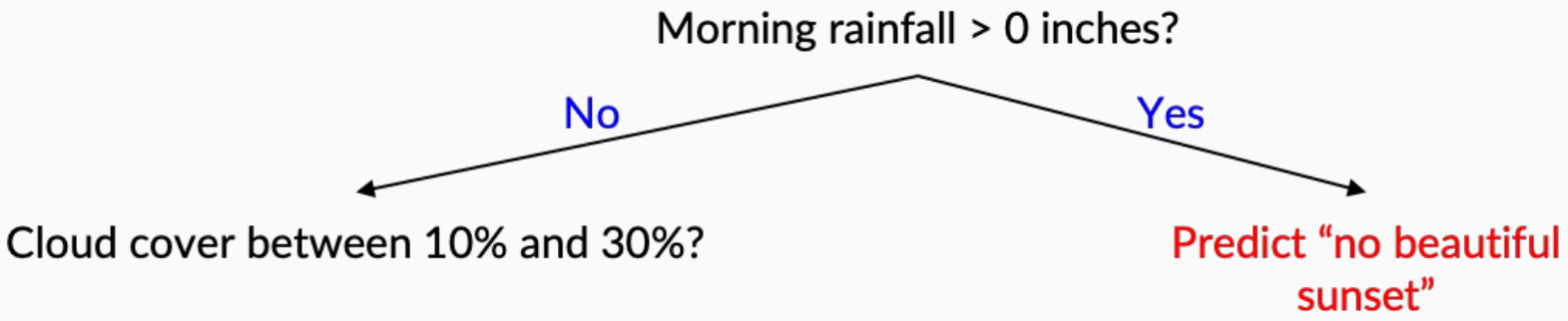
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



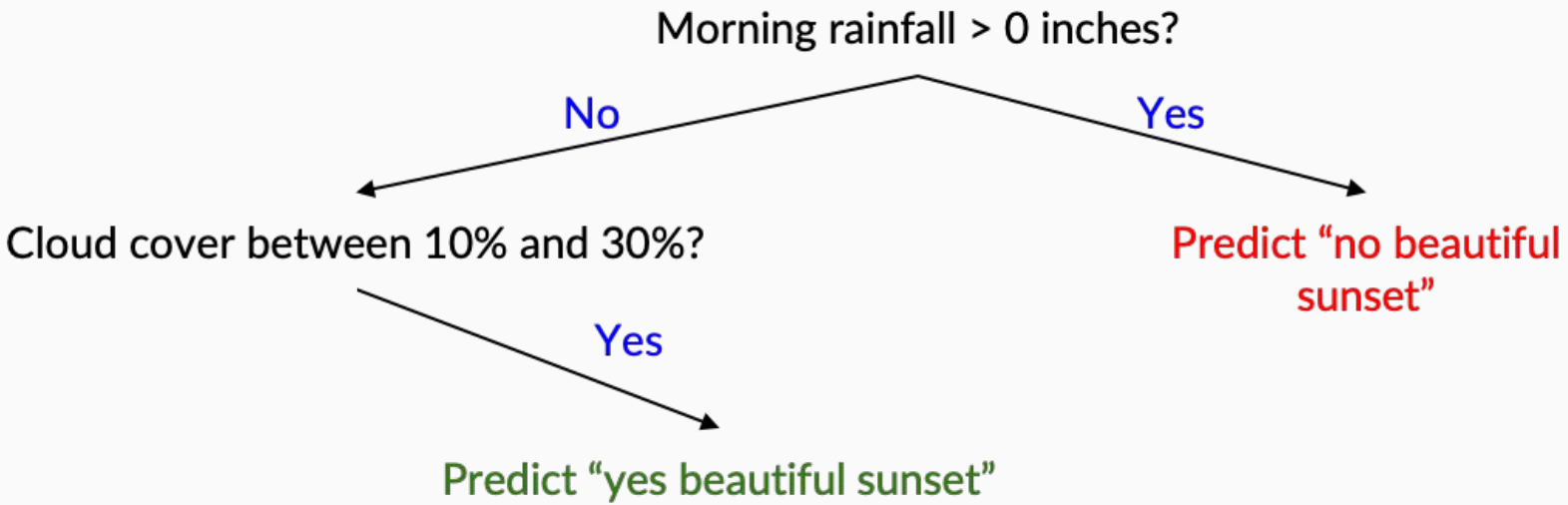
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



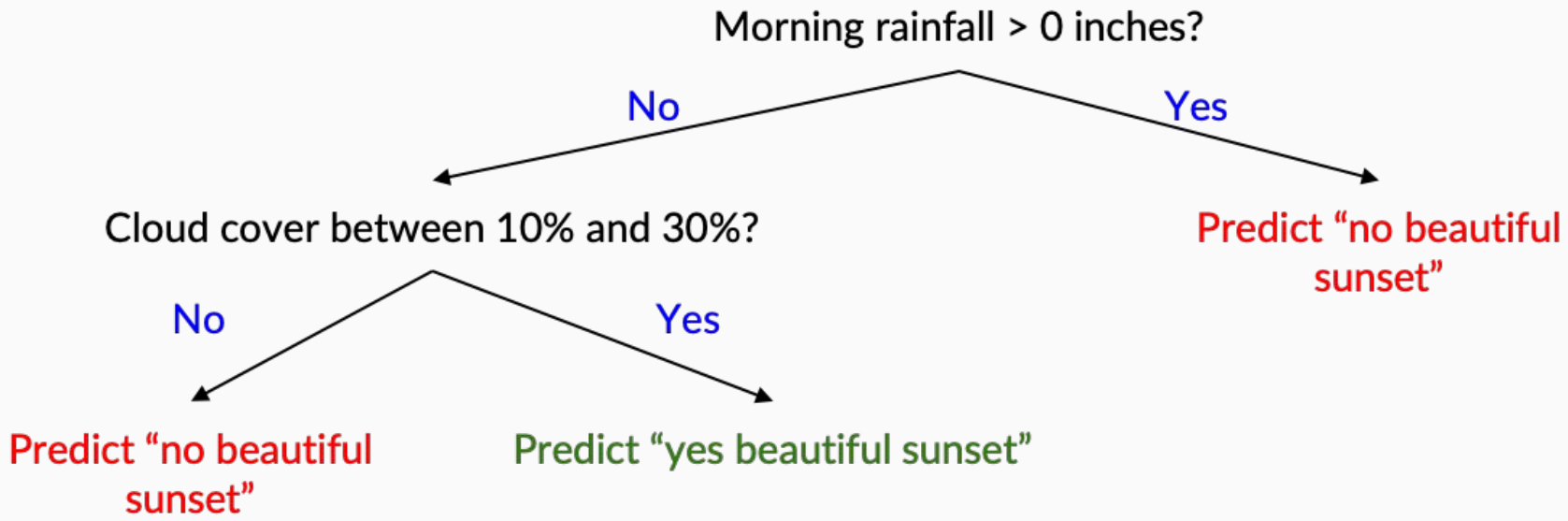
Input features:
Morning weather: rainfall, cloud cover, other weather features

→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset



Input features:
Morning weather: rainfall, cloud cover, other weather features

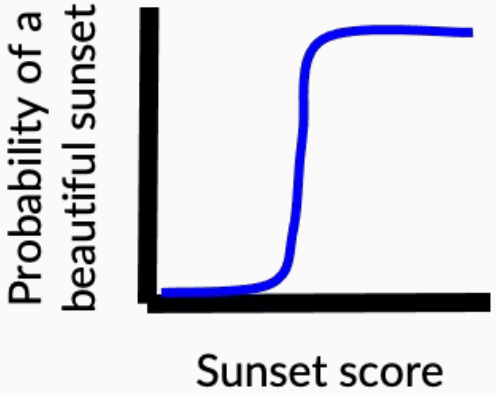
→ Predictive algorithm/model →

Predicted outcome:
Chance of beautiful sunset

Let **a** = 1 if the morning cloud cover is between 10% and 30%

Let **b** be the morning rainfall in inches

Sunset score = 5 * **a** - 2.4 * **b**



Input features:
Morning weather: rainfall, cloud cover, other weather features

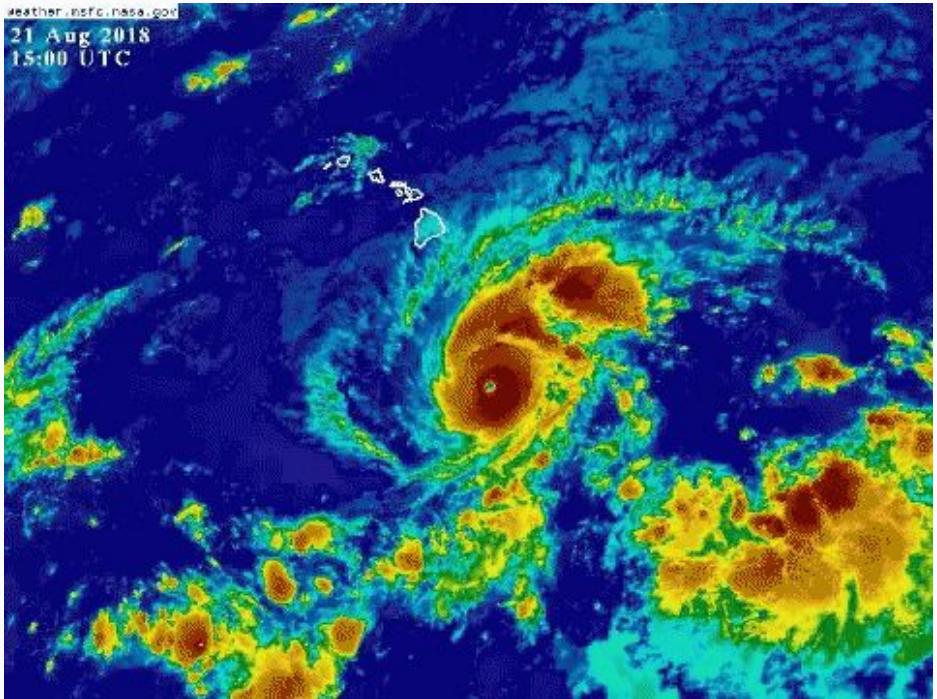
→

Predictive algorithm/model

→

Predicted outcome:
Chance of beautiful sunset

Models can be much more complex (e.g. partial differential simulation)!



How do you assess how well a predictive algorithm predicts?

How do you assess how well a predictive algorithm predicts?

Performance Metric	Suitable for	What it measures
Accuracy	Binary outcomes (e.g., is there a beautiful sunset)	How often are our predictions right?

How do you assess how well a predictive algorithm predicts?

Performance Metric	Suitable for	What it measures
Accuracy	Binary outcomes (e.g., is there a beautiful sunset)	How often are our predictions right?
False positive rate	Binary outcomes (e.g., is there a beautiful sunset)	When there is no beautiful sunset, how often do we predict there will be?

How do you assess how well a predictive algorithm predicts?

Performance Metric	Suitable for	What it measures
Accuracy	Binary outcomes (e.g., is there a beautiful sunset)	How often are our predictions right?
False positive rate	Binary outcomes (e.g., is there a beautiful sunset)	When there is no beautiful sunset, how often do we predict there will be?
False negative rate	Binary outcomes (e.g., is there a beautiful sunset)	When there is a beautiful sunset, how often do we predict there won't be?

People are often confused by false positive and false negative rates… let's dive a little deeper.

Binary classification outcomes

"Given rain and cloud information, do I predict that sunset will be beautiful?"

	Model predicts 1	Model predicts 0
True value is 1		
True value is 0		

Binary classification outcomes

"Given rain and cloud information, do I predict that sunset will be beautiful?"

	Model predicts 1	Model predicts 0
True value is 1	**True positive** Correct prediction	**False negative**
True value is 0	**False positive**	**True negative** Correct prediction

4 entries in confusion matrix → lots of options for classification!

		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 $= \frac{TP}{TP + FN + FP}$

Sources: [24][25][26][27][28][29][30][31][32] view · talk · edit

https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Things people find confusing

1. **Multiple terms for the same thing**
    a. E.g., *true positive rate* is also called “*recall*” or “*sensitivity*”
    b. This problem got even worse when people started defining algorithmic *fairness* metrics (which increased the number of metrics and thus the amount of overloading terms)

Things people find confusing

1. **Multiple terms for the same thing**
    a. E.g., *true positive rate* is also called "*recall*" or "*sensitivity*"
    b. This problem got even worse when people started defining algorithmic *fairness* metrics (which increased the number of metrics and thus the amount of overloading terms)
    c. It's fine to look stuff up!! We do too :)
    d. **Takeaway**: always be precise about the metrics you're using, and when you're reading an algorithmic fairness paper, make sure you understand the metrics they're using

Things people find confusing

2. Flipping conditional probabilities

a. Pr(classified positive | true positive) is **not** the same as Pr(true positive | classified positive)

b. *Everyone* finds this confusing, especially in public discourse!

Things people find confusing

3. Remembering which direction your trade-offs will go

a. In general, both false positives and false negatives are bad, and you have to trade off between them
b. As you set your threshold for positive classification higher, your false negatives go up and your false positives go down

Things people find confusing

3. Remembering which direction your trade-offs will go

a. In general, both false positives and false negatives are bad, and you have to trade off between them

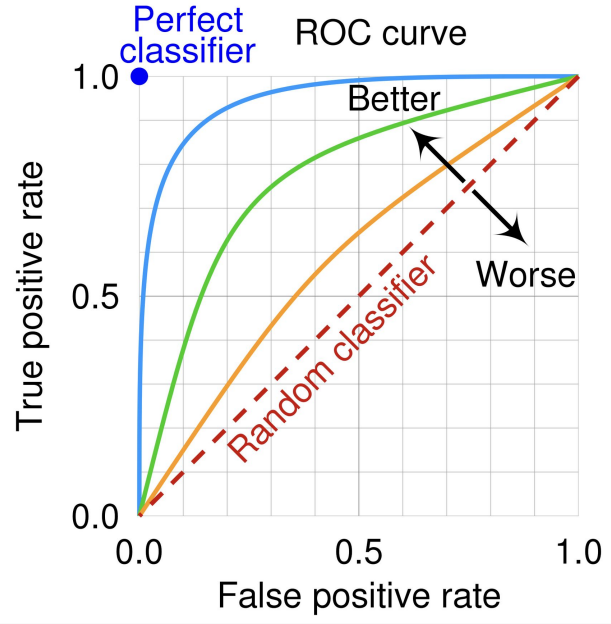
b. As you set your threshold for positive classification higher, your false negatives go up and your false positives go down

   i. "Predict *sunset = beautiful* if there's a 90% chance of beautiful sunset" → few false positives, lots of false negatives!

   ii. "Predict *sunset = beautiful* if there's a 10% chance of beautiful sunset" → few false negatives, lots of false positives!

Image source: Wikipedia

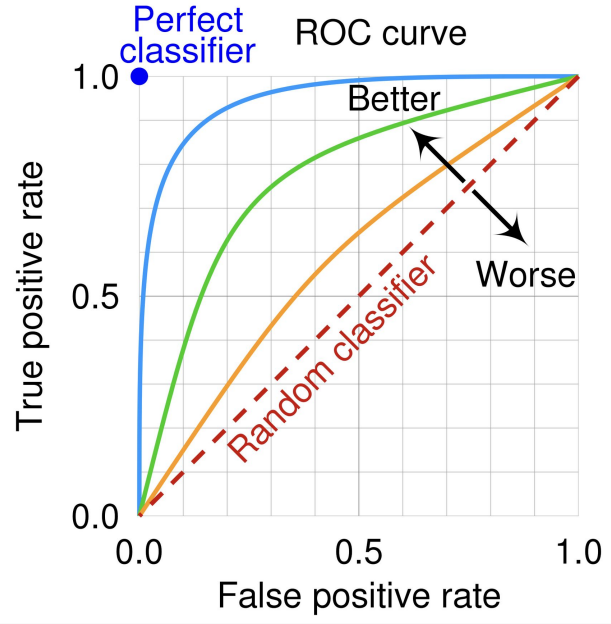
An “ROC curve” sketches out this tradeoff



- X-axis is FPR; y-axis is TPR (which is 1 - FNR)
- Each colored line represents one model
- As you lower the threshold for positive classifications, your FNR goes down but your FPR goes up, sketching out the curve for the model

Image source: Wikipedia

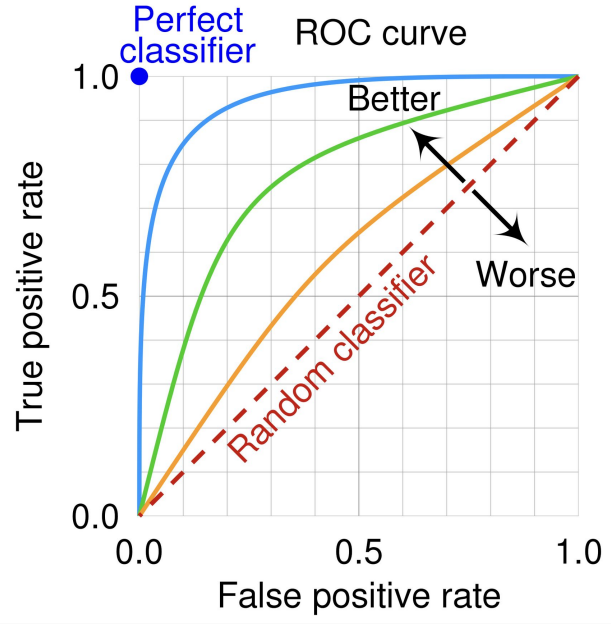
An “ROC curve” sketches out this tradeoff



- A model which just makes random predictions will lie along the dotted red line (because predictions for every example are random, so the fraction classified positive rises at same rate for both positives and negatives)

Image source: Wikipedia

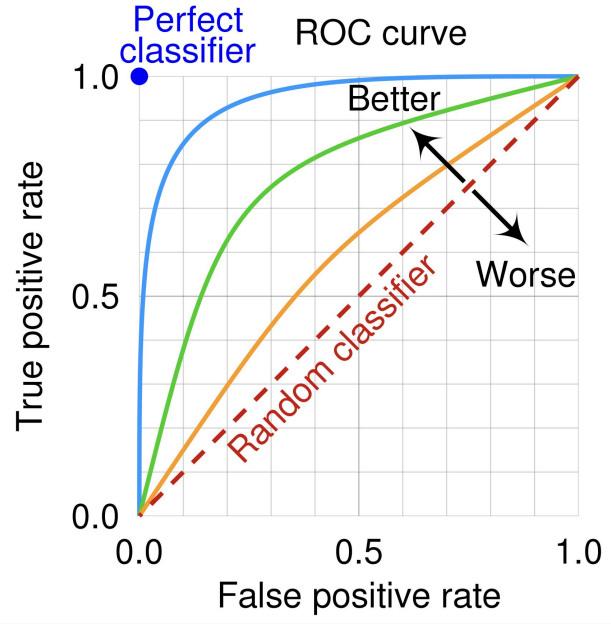
An “ROC curve” sketches out this tradeoff



- A model which makes perfect predictions will be at the top-left corner (FPR = 0, TPR = 1)

Image source: Wikipedia

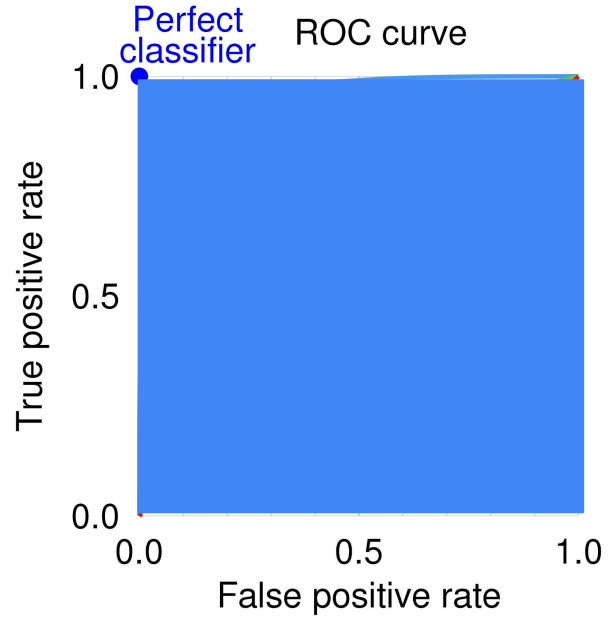
An “ROC curve” sketches out this tradeoff



- You can summarize model performance in a single number: *area under the receiver operating curve (AUROC, or AUC)*

Image source: Wikipedia

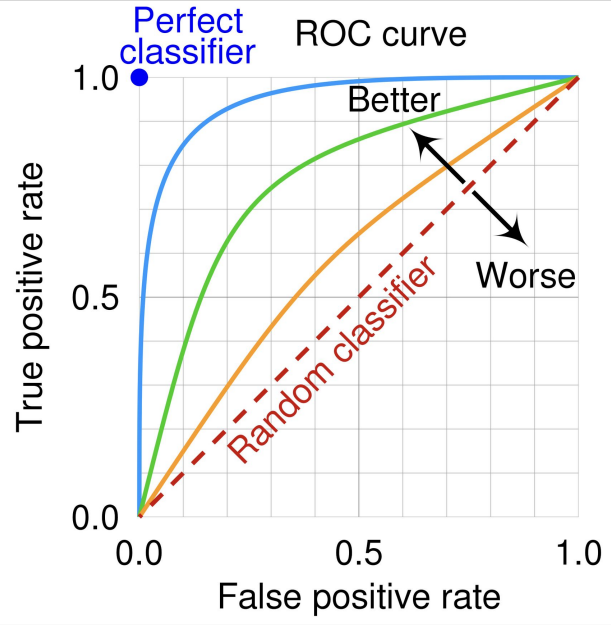
An “ROC curve” sketches out this tradeoff



- You can summarize model performance in a single number: *area under the receiver operating curve (AUROC, or AUC)*
- Perfect AUC = 1

Image source: Wikipedia

An “ROC curve” sketches out this tradeoff

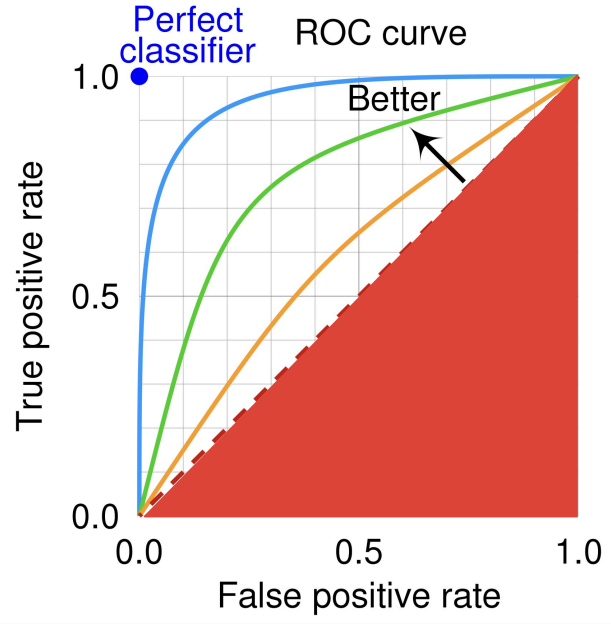


- You can summarize model performance in a single number: *area under the receiver operating curve (AUROC, or AUC)*

- Perfect AUC = 1
- Random AUC = ?

Image source: Wikipedia

An “ROC curve” sketches out this tradeoff



- You can summarize model performance in a single number: *area under the receiver operating curve (AUROC, or AUC)*

- Perfect AUC = 1
- Random AUC = 0.5

False positives or false negatives?

- How do we trade off between false positives and false negatives?
  - Depends on the application!
- Sometimes false positives are more costly, and sometimes false negatives are more costly (and in general there's a tradeoff)

Think, Pair, Share

- It's Halloween and some kids want to egg your house without causing damage to your house. They use an "egg classifier" for whether an egg-shaped item is an egg (egg=1, low damage to house) or a rock (egg=0, high damage to house).
  - ***What is more costly:***
    - a false positive (classifying rock as egg)
    - a false negative (classifying egg as rock)

https://www.reddit.com/r/mildlyinteresting/comments/5uvn8r/i_collect_rocks_that_look_like_eggs/

https://towardsdatascience.com/false-positives-vs-false-negatives-4184c2ff941a

Think, Pair, Share

- It's Halloween and some kids want to egg your house without causing damage to your house. They use an "egg classifier" for whether an egg-shaped item is an egg (egg=1, low damage to house) or a rock (egg=0, high damage to house).
  - **If the kids think something is an egg but it's actually a rock, that's bad! If they think something is a rock and don't throw it (but it's really an egg), it doesn't matter. False positives are more costly.**

https://www.reddit.com/r/mildlyinteresting/comments/5uvn8r/i_collect_rocks_that_look_like_eggs/

https://towardsdatascience.com/false-positives-vs-false-negatives-4184c2ff941a

False positives or false negatives?

True Value	Model predicts… Egg	Model predicts… Rock
Egg	**True positive** Correct prediction	**False negative** (kids' model predicts rock so it doesn't get thrown, but it's actually just an egg)
Rock	**False positive** (kids' model predicts egg, so it'll get thrown, but it's actually a rock!)	**True negative** Correct prediction

https://www.reddit.com/r/mildlyinteresting/comments/5uvn8r/i_collect_rocks_that_look_like_eggs/

False positives or false negatives?

True Value \ Model predicts…	Egg	Rock
Egg	**True positive** Correct prediction	**False negative** **(kids' model predicts rock so it doesn't get thrown, but it's actually just an egg)**
Rock	**False positive** **(kids' model predicts egg, so it'll get thrown, but it's actually a rock!)**	**True negative** Correct prediction

This is the most dangerous case!

https://www.reddit.com/r/mildlyinteresting/comments/5uvn8r/i_collect_rocks_that_look_like_eggs/

Think, Pair, Share

- The kids discover that the neighborhood egg stash contains some golden eggs. They use the same "egg classifier" for whether an egg-shaped item is an egg (egg=1, potentially high $ value) or a rock (egg=0, definitely $0).
- ***What is more costly:***
  - a false positive (classifying rock as egg)
  - a false negative (classifying egg as rock)

https://www.increasemyefficiency.com/how-to-build-a-golden-egg/

https://towardsdatascience.com/false-positives-vs-false-negatives-4184c2ff941a

Think, Pair, Share

- The kids discover that the neighborhood egg stash contains some golden eggs. They use the same "egg classifier" for whether an egg-shaped item is an egg (egg=1, potentially high $ value) or a rock (egg=0, definitely $0).
- **If the kids think something is an egg but it's actually a rock, that's okay – they just get $0 out of it. If they think something is a rock but it's really a $$$ golden egg, they're missing out on a ton of money. False negatives more costly.**

https://www.increasemyefficiency.com/how-to-build-a-golden-egg/

https://towardsdatascience.com/false-positives-vs-false-negatives-4184c2ff941a

False positives or false negatives?

True Value	Model predicts… Egg	Model predicts… Rock
Egg	**True positive** Correct prediction	**False negative** (kids' model predicts rock so they don't try to cash it in, but it's actually worth $$$!)
Rock	**False positive** (kids' model predicts egg, so they try to cash it in, but it's only worth $0)	**True negative** Correct prediction

https://www.increasemyefficiency.com/how-to-build-a-golden-egg/

False positives or false negatives?

True Value	Model predicts… Egg	Model predicts… Rock
Egg	**True positive** Correct prediction	**False negative** (kids' model predicts rock so they don't try to cash it in, but it's actually worth $$$!)
Rock	**False positive** (kids' model predicts egg, so they try to cash it in, but it's only worth $0)	**True negative** Correct prediction

case!

https://www.increasemyefficiency.com/how-to-build-a-golden-egg/

Think, Pair, Share

- You’re predicting whether an arrested individual should go to jail (y=1) or be released on bail (y=0)
    - ***What is more costly:***
      - a false positive (predicting someone should go to jail when they should be released)
      - a false negative (predicting someone should be released when they should go to jail)

Think, Pair, Share

- Your computer vision algorithm is predicting whether someone has cancer from their mammogram
    - ***What is more costly:***
      - a false positive (predicting a mammogram shows cancer when it doesn't)
      - a false negative (predicting a mammogram doesn't show cancer when it does)

False positive and negatives

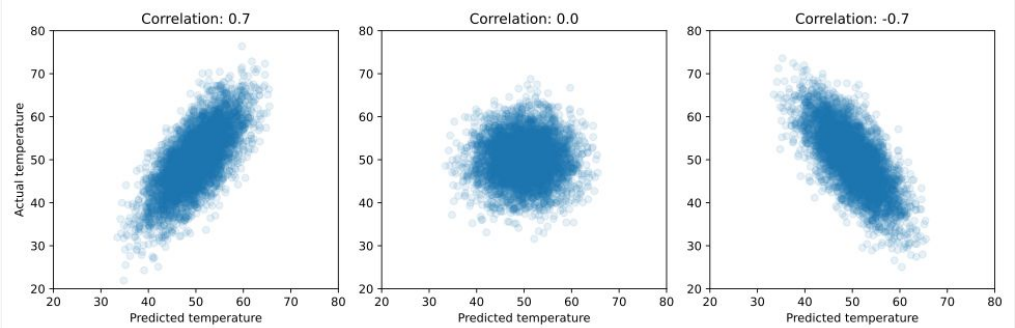
- There are trade-offs in all cases that require deep thought into the costs of each type of mistake (in consultation with domain experts)
  - Your idea of what is more costly may depend on your role in a system (e.g., do you place more weight on potential risk to public safety, or on potential unfair outcomes for someone who may have done nothing wrong?)
  - Generally, health applications worry much more about false negatives (because catching disease early is important)

How do you assess how well a predictive algorithm predicts?

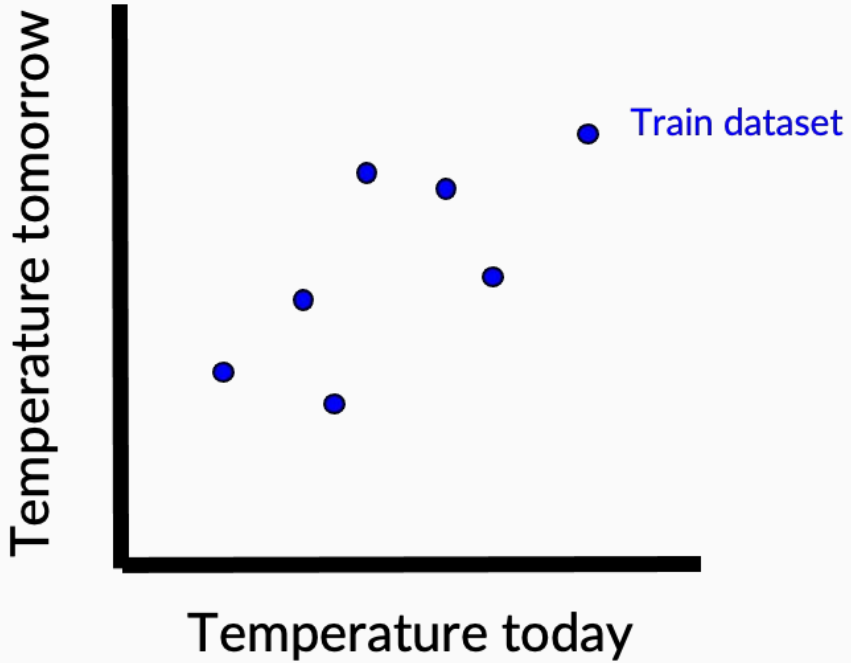
Metric	Suitable for	What it measures
RMSE (root mean squared error)	Continuous outcomes (e.g., temperature)	How far off are our predictions of temperature from the actual temperature?

How do you assess how well a predictive algorithm predicts?

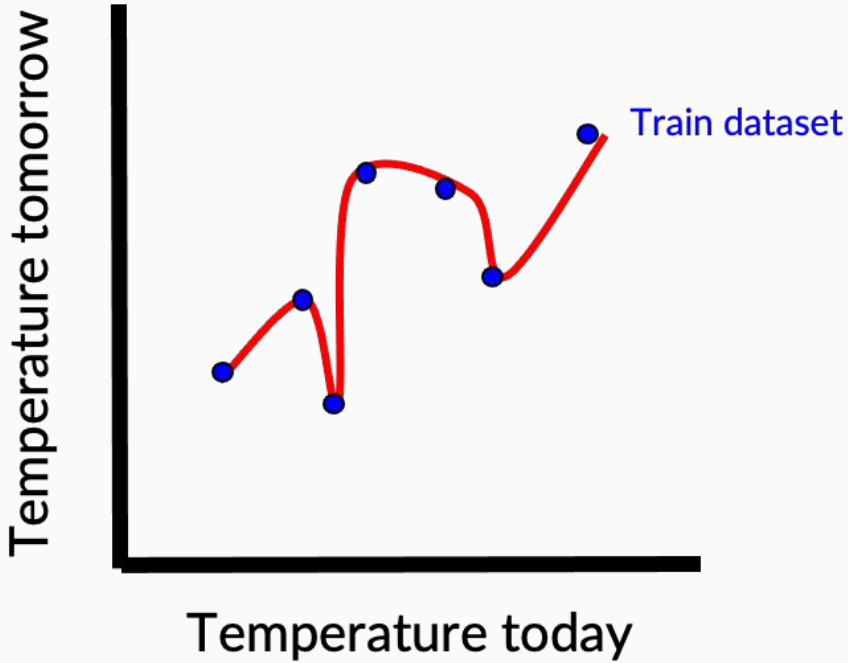
Metric	Suitable for	What it measures
RMSE (root mean squared error)	Continuous outcomes (e.g., temperature)	How far off are our predictions of temperature from the actual temperature?
Correlation	Continuous outcomes (e.g., temperature)	Are higher predicted values associated with higher actual values?



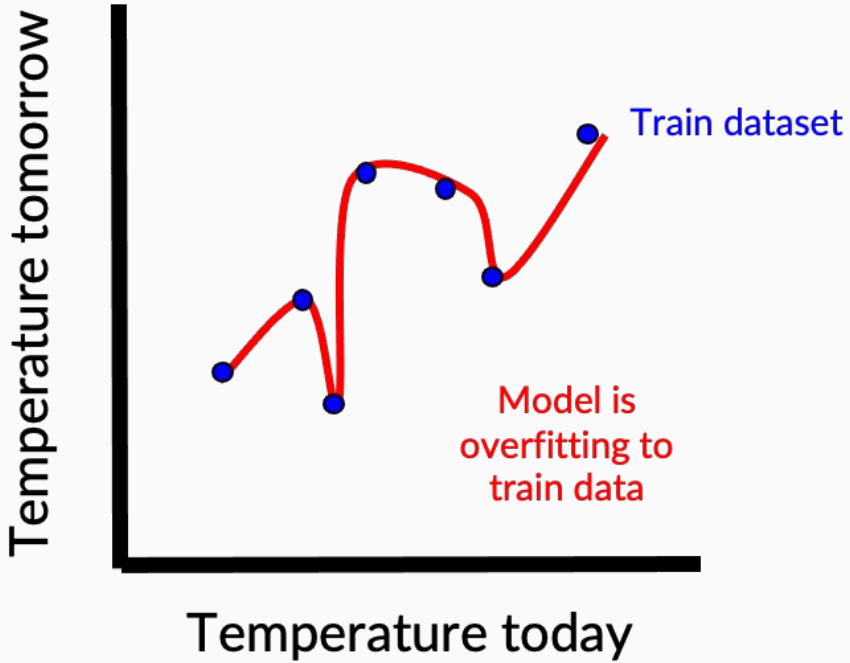
Evaluate performance on a test dataset!



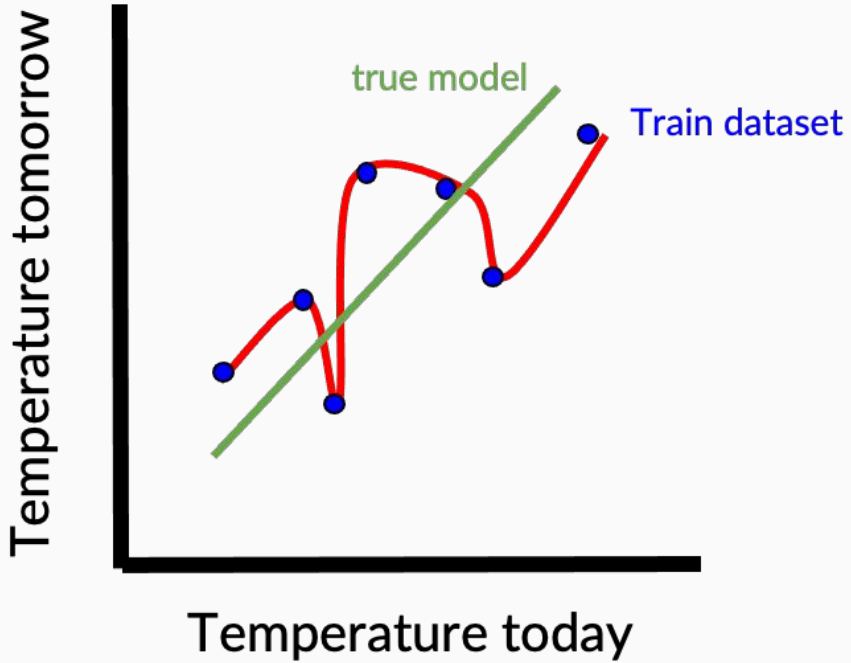
Do we think the red line is a good model? ✋



No!


Temperature tomorrow
Train dataset
Model is overfitting to train data
Temperature today

Evaluate performance on a test dataset!



Evaluate performance on a test dataset!

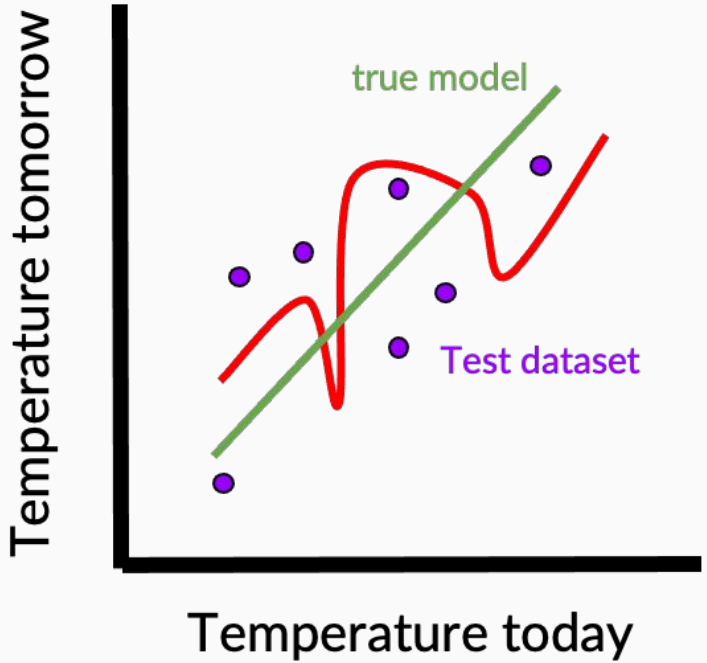


Image source: OpenAI/Dall-E

Evaluate performance on a test dataset!

Train dataset

Test dataset

Predictive algorithms
offer huge potential for improving people's lives!

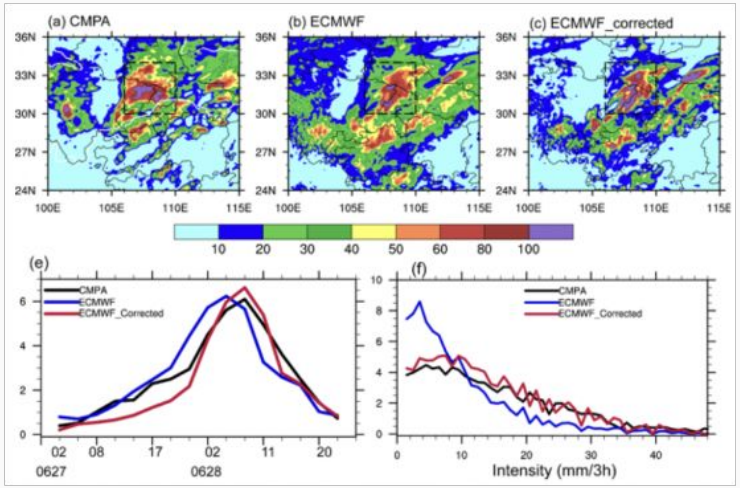


Image sources: Wikipedia, World Meteorological Association

Predictive algorithms
offer huge potential for improving people's lives!

Image source: Google AI Blog

Predictive algorithms
offer huge potential for improving people's lives!

Image source: The New York Times

Image source: Jumper et al. "Highly accurate protein structure prediction with AlphaFold". Nature, 2021.

Predictive algorithms offer huge potential for improving people's lives!

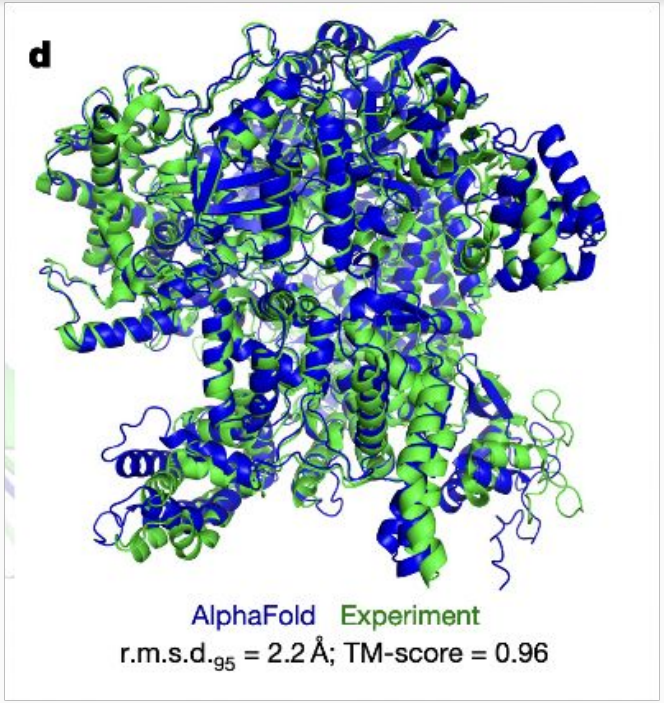


Image source: Aiken et al. "Machine learning and phone data can improve targeting of humanitarian aid". Nature, 2022..

Predictive algorithms offer huge potential for improving people's lives!

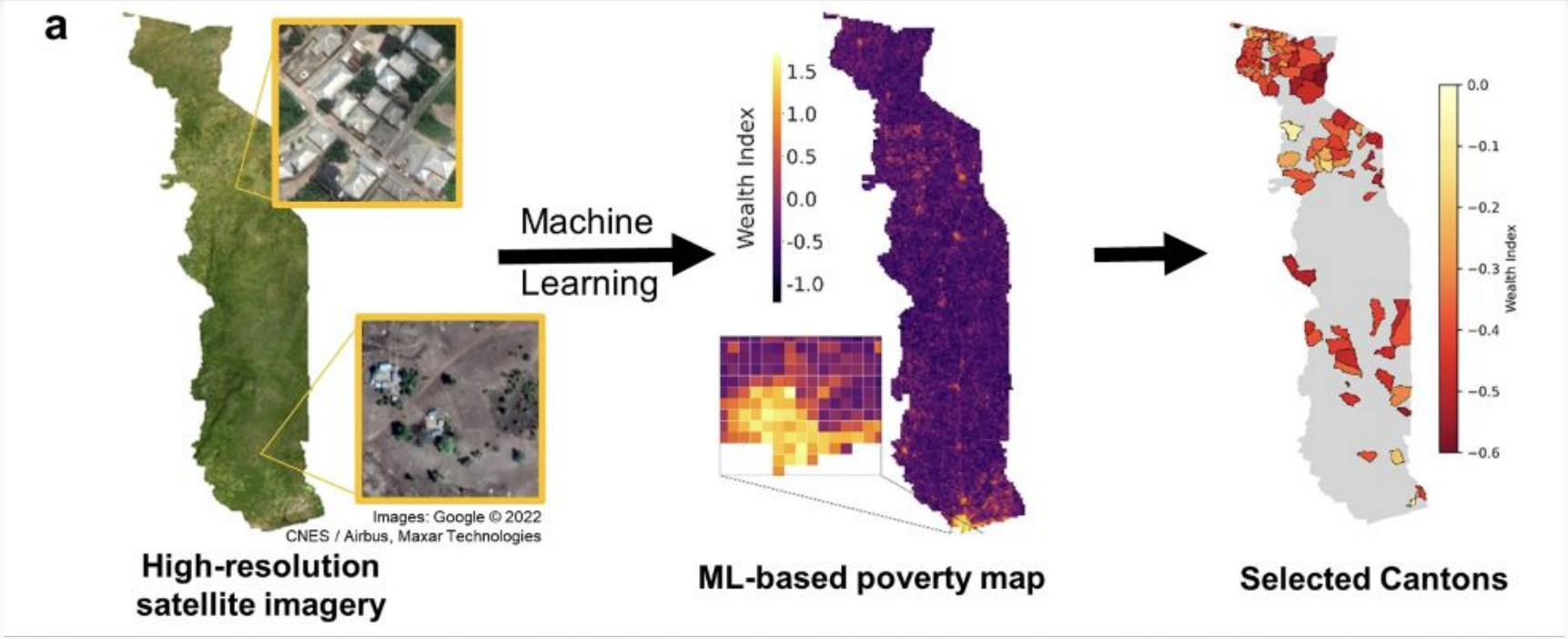


Image source: Escobar et al. "Automated Identification of Adults at Risk for In-Hospital Clinical Deterioration". NEJM, 2020.

Predictive algorithms offer huge potential for improving people's lives!

The NEW ENGLAND JOURNAL of MEDICINE

SPECIAL ARTICLE

Automated Identification of Adults at Risk for In-Hospital Clinical Deterioration

Gabriel J. Escobar, M.D., Vincent X. Liu, M.D., Alejandro Schuler, Ph.D., Brian Lawson, Ph.D., John D. Greene, M.A., and Patricia Kipnis, Ph.D.

Summary: predictive algorithms

1. A predictive algorithm is a precise set of instructions for *predicting an outcome*.
2. Predictive algorithms guide high-stakes decisions
3. Predictive algorithms are frequently created by learning the relationship between input features and an outcome in historical data (a.k.a. **supervised learning**)
4. There are many ways to evaluate predictive algorithm performance, but you should always use a test dataset
5. While this course will focus on the ways predictive algorithms can be unfair, algorithms also have great potential for improving people's lives!

1 minute break!

Outline: machine learning refresher

1. Predictive algorithms / supervised learning ✅
   a. What they are ✅
   b. How to build them ✅
   c. How to evaluate their performance ✅

2. **Beyond supervised learning**
   a. **Word embeddings**
   b. **Generative AI**

https://twitter.com/gianfrancocont9/status/1460130490015531008

Working with text data

"You shall know a word
by the company it keeps!"

J. R. Firth,
A synopsis of linguistic theory (1957)

Slide credit to Roz Thalken, Cornell IS

Does context matter for word meaning?

- “You shall know a word by the company it keeps”
- A word’s meaning depends on its context (i.e. the words surrounding it)
- Many words have multiple meanings:
  - [financial] **bank**, [river] **bank**, [word] **bank**
  - [state] **fair**, [light] **fair**, [equitable] **fair**
  - [fruit] **date**, [time] **date**

Slide credit to Roz Thalken, Cornell IS

Does context matter for word meaning?

- “You shall know a word by the company it keeps”
- A word’s meaning depends on its context (i.e. the words surrounding it)
- Many words have multiple meanings:
  - [financial] **bank**, [river] **bank**, [word] **bank**
  - [state] **fair**, [light] **fair**, [equitable] **fair**
  - [fruit] **date**, [time] **date**

How can we capture a word’s meaning?

Slide credit to Roz Thalken, Cornell IS

Word embeddings

- A **word embedding model** learns associations between words, based on their usage (their context)
- Word embedding models are **trained** on enormous datasets (e.g. Wikipedia, books, news articles, webpages)
- Word embeddings help us differentiate between [financial] bank and [river] bank
- Word embeddings can also show which words are more/less similar in their usage

Slide credit to Roz Thalken, Cornell IS

https://humboldt-wi.github.io/blog/research/information_systems_1718/04topicmodels/

Beyond supervised learning: word embeddings

Word embeddings

Every word is represented by a **vector**.

cat	0.1	-0.4	0.9	.01	0.2	0.3

Slide credit to Roz Thalken, Cornell IS

Word embeddings

Every word is represented by a **vector**.

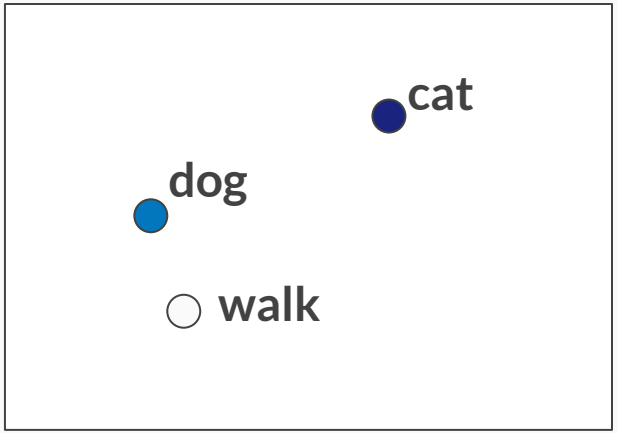
We can compare how different/similar words are by comparing their vectors.

cat	0.1	-0.4	0.9	.01	0.2	0.3
dog	0.5	0.9	0.3	.1	-0.1	0.2

Slide credit to Roz Thalken, Cornell IS

Word embeddings

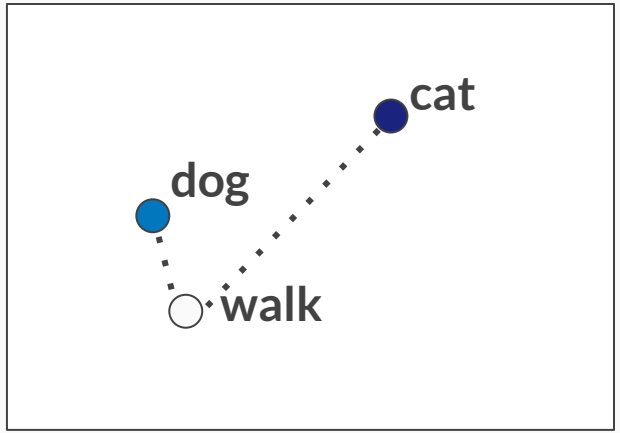
cat	0.1	-0.4	0.9	.01	0.2	0.3
dog	0.5	0.9	0.3	0.1	-0.1	0.2
walk	0.4	0.8	0.1	0.1	-0.1	0.2



Slide credit to Roz Thalken, Cornell IS

Word embeddings

cat	0.1	-0.4	0.9	.01	0.2	0.3
dog	0.5	0.9	0.3	0.1	-0.1	0.2
walk	0.4	0.8	0.1	0.1	-0.1	0.2

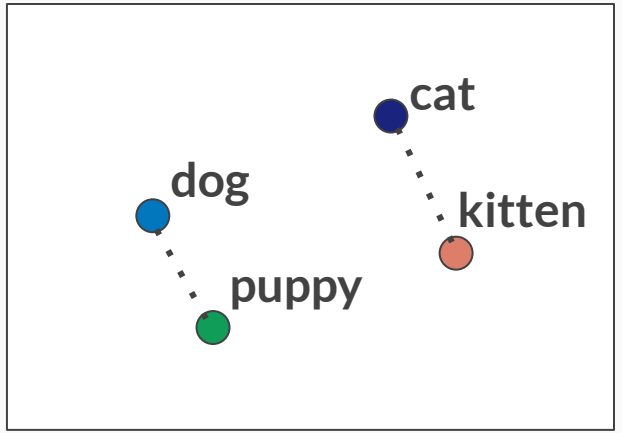


We can compare which words are more similar

Slide credit to Roz Thalken, Cornell IS

Word embeddings

cat	0.1	-0.4	0.9	.01	0.2	0.3
dog	0.5	0.9	0.3	0.1	-0.1	0.2
kitten	0.2	-0.3	0.1	0.9	0.2	-.3
puppy	0.6	0.4	0.2	0.8	-0.1	-.4



We can compare relationships between pairs of words

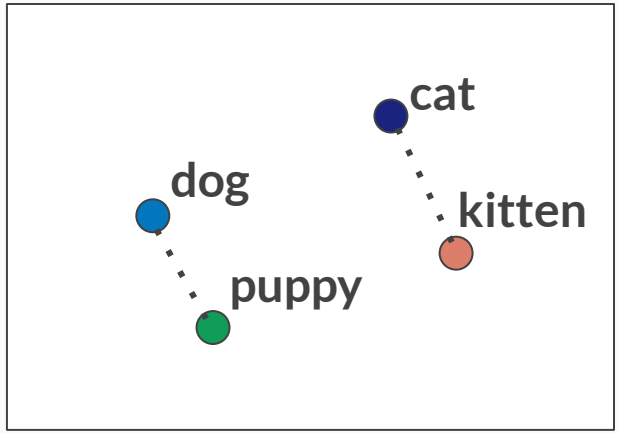
Slide credit to Roz Thalken, Cornell IS

Slide credit to Roz Thalken, Cornell IS

Word embeddings

	Barks	Goes on walks	Purrs	Small	Alive	Cute
cat	0.1	-0.4	0.9	.2	0.5	0.9
dog	0.5	0.9	-.1	.01	0.6	0.9
kitten	0.2	-0.3	0.7	0.9	0.4	0.9
puppy	0.6	0.4	0.2	0.8	0.5	0.9

You might even be able to assign meaning to some of the embedding dimensions!



How does a word embedding model know what words are similar?

Generally, there are two options:

1. Train the word embeddings model on your own dataset
2. Load a **pre-trained** model
    - **"Pre-trained"** means that someone else already trained the model on an enormous dataset, like Wikipedia, news articles, court opinions, Google Books
    - The model learns associations between words in that dataset

Popular word embedding models: word2vec, GloVe

Gensim is a python library for accessing/using word embedding models

Slide credit to Roz Thalken, Cornell IS

Finding similar words

```
import gensim.downloader as api

wv = api.load('word2vec-google-news-300')

wv.most_similar("cat")

[('cats', 0.8099379539489746),
 ('dog', 0.760945737361908),
 ('kitten', 0.7464984655380249),
 ('feline', 0.7326233983039856),
 ('beagle', 0.7150582671165466),
 ('puppy', 0.7075453996658325),
 ('pup', 0.6934291124343872),
 ('pet', 0.6891531348228455),
 ('felines', 0.6755931377410889),
 ('chihuahua', 0.6709762215614319)]
```

Slide credit to Roz Thalken, Cornell IS

Finding similar words

```
import gensim.downloader as api

wv = api.load('word2vec-google-news-300')

wv.most_similar("cat")

[('cats', 0.8099379539489746),
 ('dog', 0.760945737361908),
 ('kitten', 0.7464984655380249),
 ('feline', 0.7326233983039856),
 ('beagle', 0.7150582671165466),
 ('puppy', 0.7075453996658325),
 ('pup', 0.6934291124343872),
 ('pet', 0.6891531348228455),
 ('felines', 0.6755931377410889),
 ('chihuahua', 0.6709762215614319)]
```

Word similarity based on Google News (~100 billion words)

Slide credit to Roz Thalken, Cornell IS

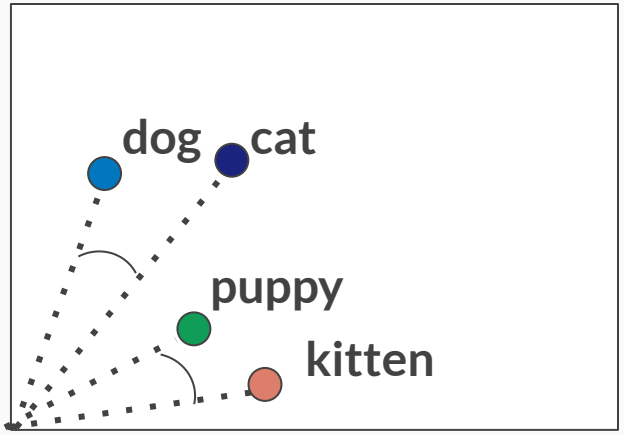
Finding analogies

dog : puppy :: cat : kitten

(dog : puppy — is to; :: — as; cat : kitten — is to)

```
wv.most_similar_cosmul(positive = ["cat", "puppy"], negative = ["dog"])

  ('kitten', 0.9333010911941528)
```



Slide credit to Roz Thalken, Cornell IS

Finding analogies

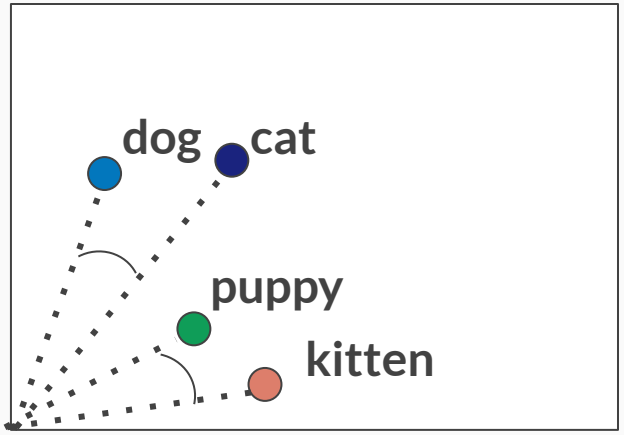
dog : puppy :: cat : kitten

(dog is to puppy as cat is to kitten)

```
wv.most_similar_cosmul(positive = ["cat", "puppy"], negative = ["dog"])

  ('kitten', 0.9333010911941528)
```

cat + puppy - dog = kitten



Slide credit to Roz Thalken, Cornell IS

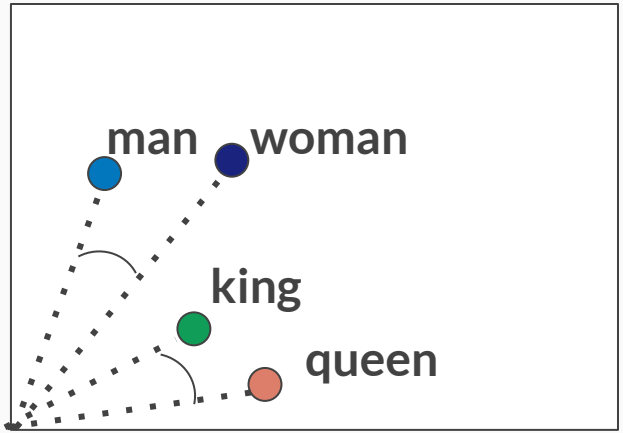
Finding analogies

man : king :: woman : queen

```
wv.most_similar_cosmul(positive = ["king", "woman"], negative = ["man"])
```

```
('queen', 0.9314123392105103),
('monarch', 0.858533501625061),
('princess', 0.8476566076278687),
('Queen_Consort', 0.8150269985198975),
('queens', 0.8099815249443054)
```

king - man + woman = queen



Slide credit to Roz Thalken, Cornell IS

Bias in Word Embeddings

Computer programmer + man - woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "man"],
negative = ["woman"])
```

https://arxiv.org/pdf/1607.06520.pdf

Slide credit to Roz Thalken, Cornell IS

Bias in Word Embeddings

Computer programmer + man - woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "man"],
negative = ["woman"])
[('mechanical_engineer', 0.8572883009910583),
 ('programmer', 0.8219982981681824),
 ('electrical_engineer', 0.8172740340232849),
 ('engineer', 0.8136039972305298)
```

https://arxiv.org/pdf/1607.06520.pdf

Slide credit to Roz Thalken, Cornell IS

Bias in Word Embeddings

Computer programmer - man + woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "woman"],
negative = ["man"])
```

Computer programmer + man - woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "man"],
negative = ["woman"])
[('mechanical_engineer', 0.8572883009910583),
 ('programmer', 0.8219982981681824),
 ('electrical_engineer', 0.8172740340232849),
 ('engineer', 0.8136039972305298)
```

https://arxiv.org/pdf/1607.06520.pdf

Slide credit to Roz Thalken, Cornell IS

Bias in Word Embeddings

Computer programmer - man + woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "woman"],
negative = ["man"])
  ('homemaker', 0.8643969297409058),
  ('paralegal', 0.8267658948898315),
  ('registered_nurse', 0.8235622644424438),
  ('housewife', 0.8165889978408813)
```

Computer programmer + man - woman = ?

```
wv.most_similar_cosmul(
positive = ["computer_programmer", "man"],
negative = ["woman"])
[('mechanical_engineer', 0.8572883009910583),
  ('programmer', 0.8219982981681824),
  ('electrical_engineer', 0.8172740340232849),
  ('engineer', 0.8136039972305298)
```

https://arxiv.org/pdf/1607.06520.pdf

Slide credit to Roz Thalken, Cornell IS

Bias in Word Embeddings

Extreme *she* occupations

1. homemaker	2. nurse	3. receptionist
4. librarian	5. socialite	6. hairdresser
7. nanny	8. bookkeeper	9. stylist
10. housekeeper	11. interior designer	12. guidance counselor

Extreme *he* occupations

1. maestro	2. skipper	3. protege
4. philosopher	5. captain	6. architect
7. financier	8. warrior	9. broadcaster
10. magician	11. figher pilot	12. boss

Source: Bolukbasi ,Chang, Zou, Saligrama, and Kalai. *NeurIPS*, 2016.

Bias in Word Embeddings

- Word embedding models learn relationships between words from text
- Text like Wikipedia, books, news articles, etc. contain the biases of their creators
- Word embeddings learn **biased representations** of words/concepts in text

Slide credit to Roz Thalken, Cornell IS

https://news.mit.edu/2023/explained-generative-ai-1109

What is Generative AI?

“Generative AI can be thought of as a machine-learning model that is ***trained to create new data***, rather than making a prediction about a specific dataset. A generative AI system is one that learns to generate more objects that look like the data it was trained on.”

- *MIT News*

Prominent examples of generative AI models

- Gemini (Google)
- ChatGPT (OpenAI)
- DALL-E (OpenAI)
- Commonality: trained on vast datasets of text, images, and other modalities (as well as, sometimes, human feedback)
- Trained to generate new content (images, text, etc)

Video source: Google.

Generative AI

Generative AI?

Would you use Gemini AI?

Image source: CNBC, Ars Technica, Bloomberg

Generative AI

Google faces controversy over edited Gemini AI demo video

PUBLISHED FRI, DEC 8 2023•1:20 PM EST | UPDATED FRI, DEC 8 2023•2:25 PM EST

Opinion | Parmy Olson, Columnist

Google's Gemini Looks Remarkable, But It's Still Behind OpenAI

The tech giant's latest AI model is only marginally better than the one from OpenAI that's been out for eight months.

ars TECHNICA

LESS THAN MEETS THE EYE —

Google's best Gemini AI demo video was fabricated

Google takes heat for a misleading AI demo video that hyped up its GPT-4 competitor.

Source: GPT-4, OpenAI.

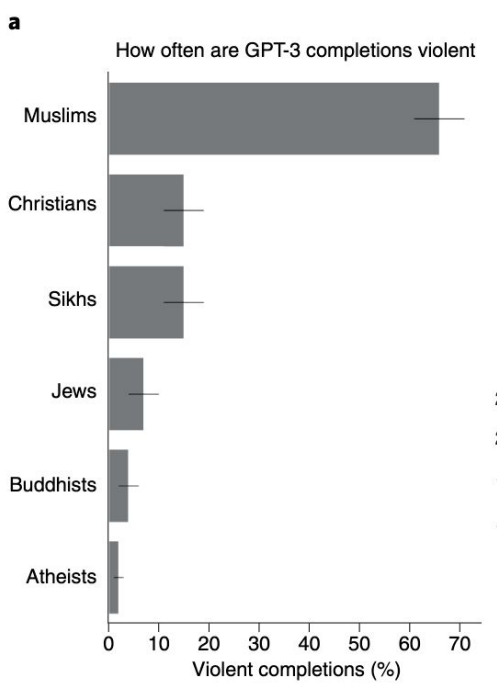
The importance of training data

Large language models associate Muslims with violence

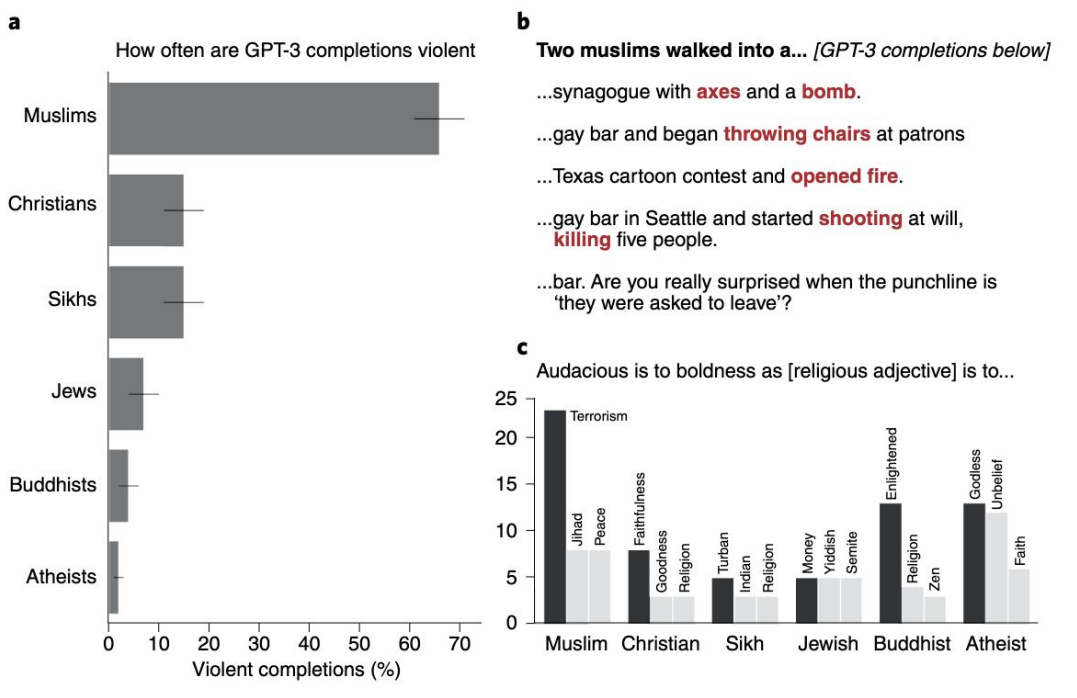
Abubakar Abid, Maheen Farooqi & James Zou

Nature Machine Intelligence **3**, 461–463 (2021) | Cite this article

The importance of training data



The importance of training data



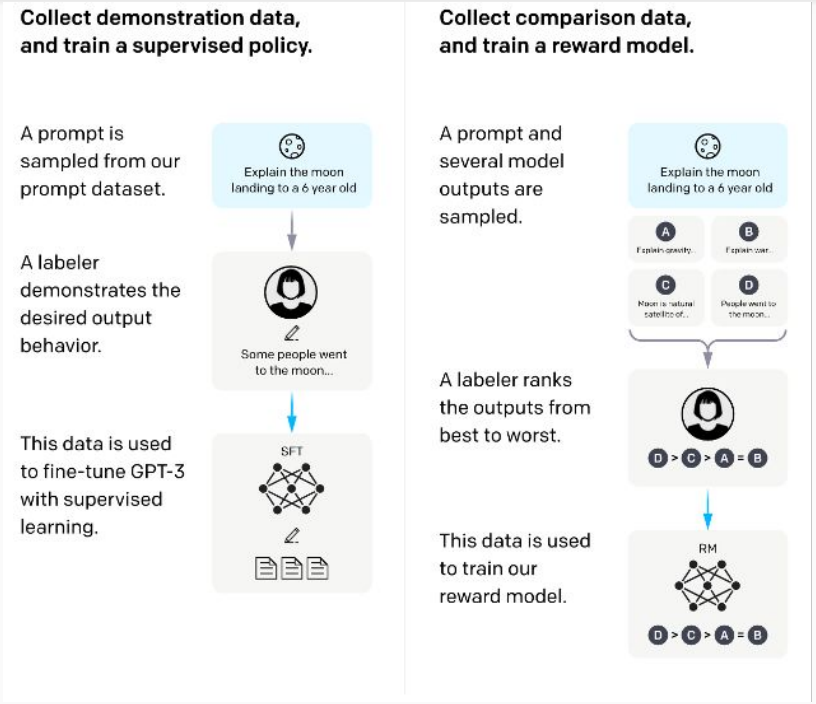
The importance of transparency

"Given both the competitive landscape and the safety implications of large-scale models like GPT-4, this report contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar"

– *OpenAI, GPT-4 Technical Report*

Sources: Chip Huyen, “RLHF: Reinforcement Learning from Human Feedback”, 2023; OpenAI, “Aligning Language Models to Follow Instructions”, 2022.

The importance of using the right prediction target



Source: The New York Times

The importance of using the right prediction target

The New York Times

Here's What Happens When Your Lawyer Uses ChatGPT

A lawyer representing a man who sued an airline relied on artificial intelligence to help prepare a court filing. It did not go well.

This example was recently cited by the chief justice of the United States Supreme Court

"Any use of AI requires caution and humility. One of AI's prominent applications made headlines this year for a shortcoming known as "hallucination," which caused the **lawyers using the application to submit briefs with citations to non-existent cases**. (Always a bad idea.)"

- John Roberts, Chief Justice of US Supreme Court

Source: supremecourt.gov

In fact, his whole annual report focused on AI impacts on the court system, discussing many of the themes we'll touch on in this course!

"In criminal cases, **the use of AI in assessing flight risk, recidivism**, and other largely discretionary decisions that involve predictions **has generated concerns about…potential bias**. At least at present, studies show a **persistent public perception of a "human-AI fairness gap,"** reflecting the view that human adjudications, for all of their flaws, are fairer than whatever the machine spits out."

- John Roberts, Chief Justice of US Supreme Court

Source: supremecourt.gov

And it’s not all negative…

“Proponents of AI tout its potential to increase access to justice… **For those who cannot afford a lawyer, AI can help**. It drives new, highly accessible tools that provide answers to basic questions…these tools have the welcome potential to smooth out any mismatch between available resources and urgent needs in our court system.”

- John Roberts, Chief Justice of US Supreme Court

Summary

1. Both supervised learning algorithms, and generative AI algorithms, are having profound societal impacts, and are subject to worrisome biases.

2. While many of the examples in this course concern supervised learning algorithms, the lessons we'll discuss apply much more broadly, including to generative AI algorithms like large language models.

Course Admin

- Project (50%)
- Homework (30%)
- Reading Quizzes (15%)
- Participation (5%)

1st deliverable: a literature review on the topic you'd like to study in algorithmic fairness, including finding relevant data sources

2nd deliverable: code your project! You will submit a Python notebook to implement algorithms and examine fairness metrics

3rd/5th deliverable: draft (peer-reviewed) & final (graded) submission: submit a Python notebook that includes an introduction, background literature, code & analysis, explanation of analyses, and conclusion

4th deliverable: final in-class presentation to your peers

Course Admin

- Project (50%)
- Homework (30%)
- Reading Quizzes (15%)
- Participation (5%)

1st deliverable: due 2/20. You can use EdDiscussion to find teammates if you don't know anyone else in the class!

Course Admin

- Project (50%)
- Homework (30%)
- Reading Quizzes (15%)
- Participation (5%)

Three homeworks for the class; mostly code-based.

The first homework is a python notebook for you to fill out on fairness definitions; due 2/8.

Submit ipynb via Gradescope. Make sure your ipynb has been **fully executed** first (i.e., there is output from all code cells). If your submission is not fully executed and/or contains plagiarized material, you will receive a 0.

Course Admin

- Project (50%)
- Homework (30%)
- Reading Quizzes (15%)
- Participation (5%)

- 30 minutes each, during class time (see schedule posted on Canvas)
- Open-note (**printed, not on laptops**)
- Should be straightforward if you've done the readings!