Bil 470 / YAP 470

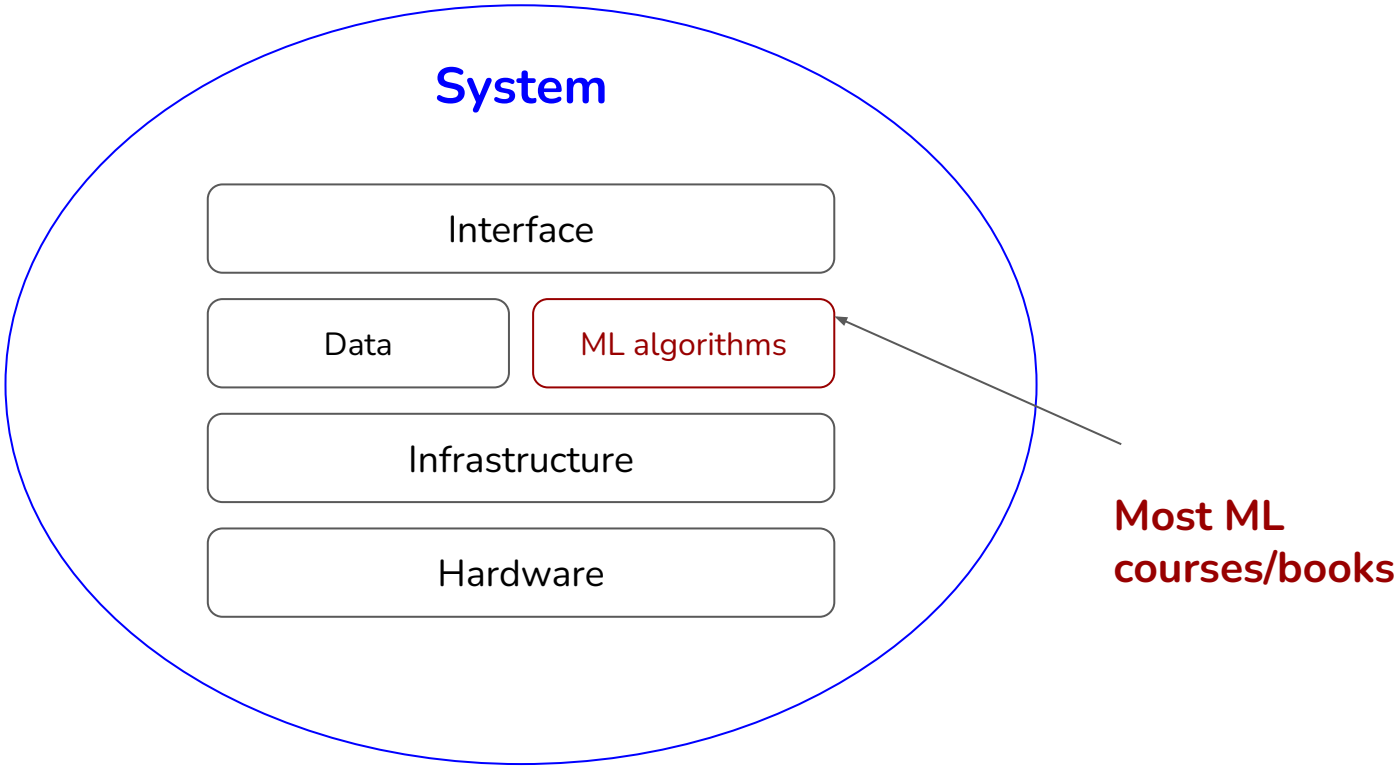
Introduction to Machine Learning (Yapay Öğrenme)

Batuhan Bardak

Machine Learning System Design

Date: 21.11.2022

Understanding ML production



Slide Credit: cs329s, stanford

ML in research vs. in production

	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting
Fairness	Good to have (sadly)	Important
Interpretability	Good to have	Important

Slide Credit: cs329s, stanford

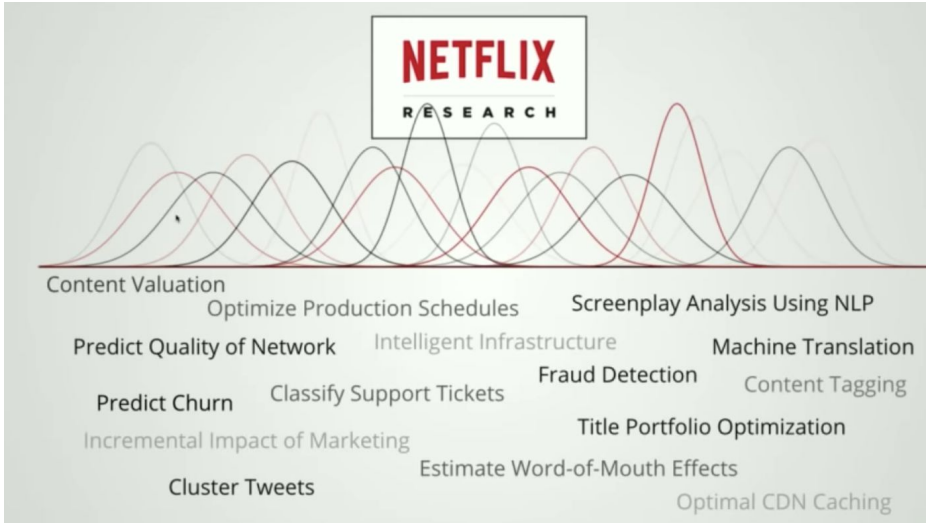
Myth #1: Deploying is hard

Deploying is easy. Deploying reliably is hard

Slide Credit: cs329s, stanford

Myth #2: You only deploy one or two ML models at a time

Booking.com: 150+ models, Uber: thousands



Slide Credit: cs329s, stanford

Myth #3: You won’t need to update your models as much

Slide Credit: cs329s, stanford

DevOps: Pace of software delivery is accelerating

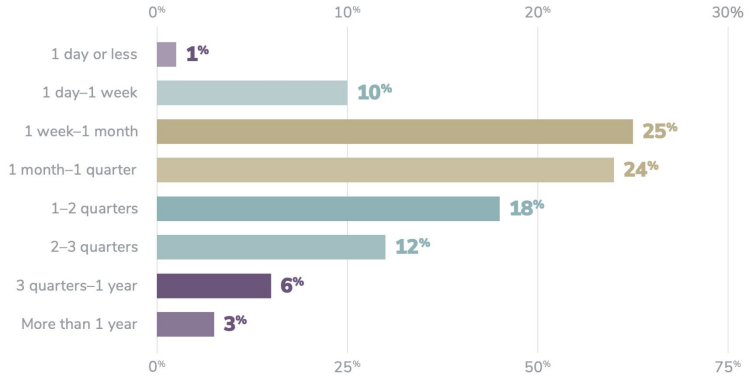
- Elite performers deploy **973x** more frequently with **6570x** faster lead time to deploy (Google DevOps Report, 2021)
- DevOps standard (2015)
  - Etsy deployed 50 times/day
  - Netflix 1000s times/day
  - AWS every 11.7 seconds

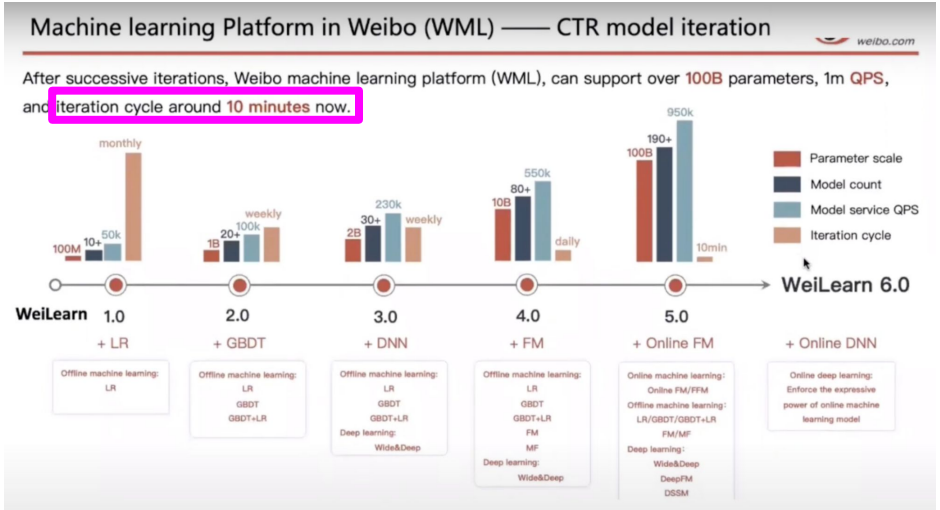
Slide Credit: cs329s, stanford

DevOps to MLOps: Slow vs. Fast

We'll learn how to do minute-iteration cycle!

Only 11% of organizations can put a model into production within a week, and 64% take a month or longer





Slide Credit: cs329s, stanford

Accelerating ML Delivery

ML + DevOps = 🚀

Slide Credit: cs329s, stanford

Myth #4: ML can magically transform your business overnight

Magically: possible
Overnight: no

Slide Credit: cs329s, stanford

ML engineering is more engineering than ML

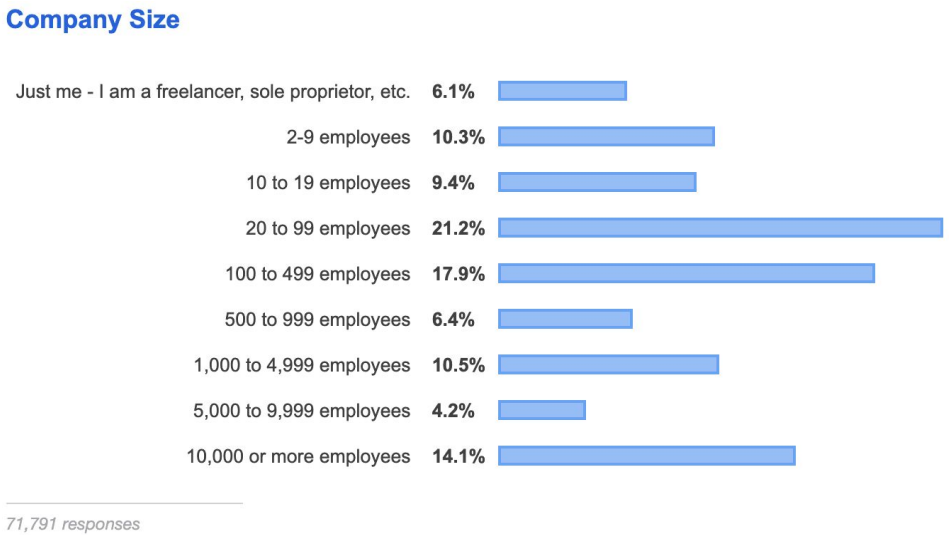
MLEs might spend most of their time:

- wrangling data
- understanding data
- setting up infrastructure
- deploying models

instead of training ML models

Slide Credit: cs329s, stanford

Myth #5: Most ML engineers don’t need to worry about scale



Slide Credit: cs329s, stanford

ML and Data Systems Fundamentals

1. ML in production: expectation
    a. Collect data
    b. Train model
    c. Deploy model
    d. 

Slide Credit: cs329s, stanford

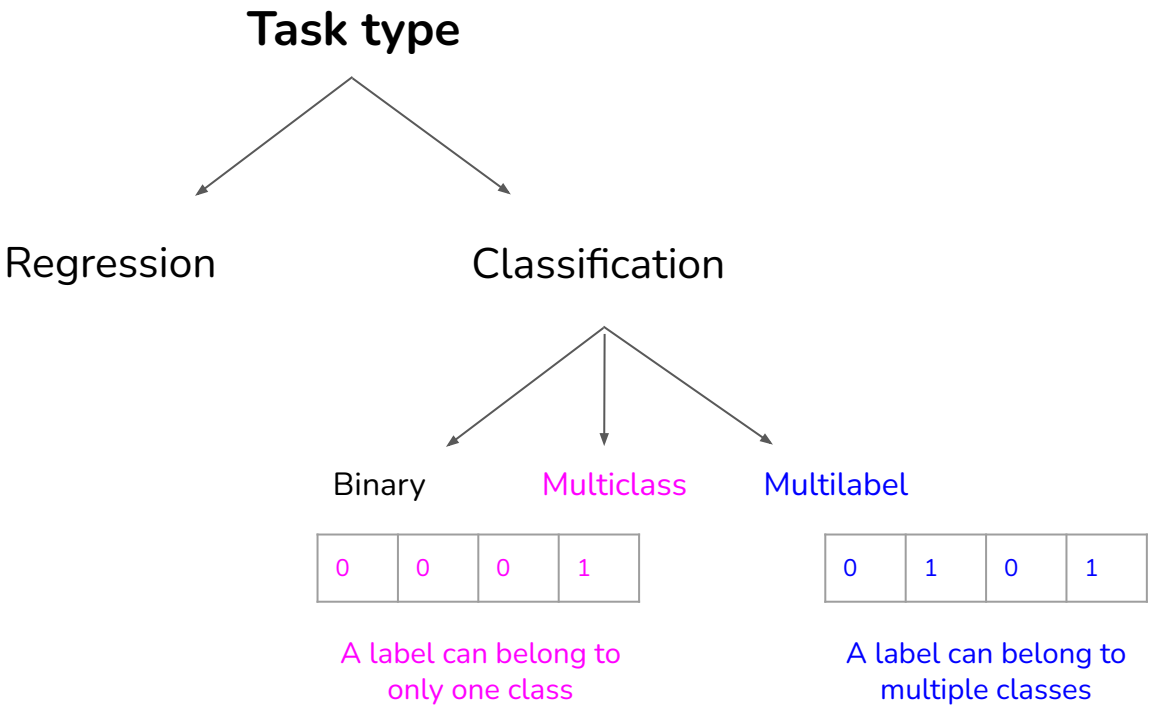
ML in production: reality

Step 15 and 17 are essential

1. Choose a metric to optimize
2. Collect data
3. Train model
4. Realize many labels are wrong -> relabel data
5. Train model
6. Model performs poorly on one class -> collect more data for that class
7. Train model
8. Model performs poorly on most recent data -> collect more recent data
9. Train model
10. Deploy model
11. Dream about $$$
12. Wake up at 2am to complaints that model biases against one group -> revert to older version
13. Get more data, train more, do more testing
14. Deploy model
15. Pray
16. Model performs well but revenue decreasing
17. Cry
18. Choose a different metric
19. Start over

Slide Credit: cs329s, stanford

Multiclass vs. multilabel



Slide Credit: cs329s, stanford

ML <-> business: mapping

- Baselines
  - Existing solutions, simple solutions, human experts, competitors solutions, etc.
- Usefulness threshold
  - Self-driving needs human-level performance. Predictive texting doesn't.
- False negatives vs. false positives
  - Covid screening: no false negative (patients with covid shouldn't be classified as no covid)
  - Fingerprint unlocking: no false positive (unauthorized people shouldn't be given access)
- Interpretability
  - Does it need to be interpretable? If yes, to whom?
- Confidence measurement (how confident it is about a prediction)
  - Does it need confidence measurement?
  - Is there a confidence threshold? What to do with predictions below that threshold—discard it, loop in humans, or ask for more information from users?

Slide Credit: cs329s, stanford

Four phases of ML adoption

Slide Credit: cs329s, stanford

Phase 1: Before ML

"If you think that machine learning will give you a 100% boost, then a heuristic will get you 50% of the way there."

Martin Zinkevich, Google

Slide Credit: cs329s, stanford

Phase 2: Simplest ML models

Start with a simple model that allows visibility into its working to:

- validate hypothesis
- validate pipeline

Slide Credit: cs329s, stanford

Phase 3: Optimizing simple models

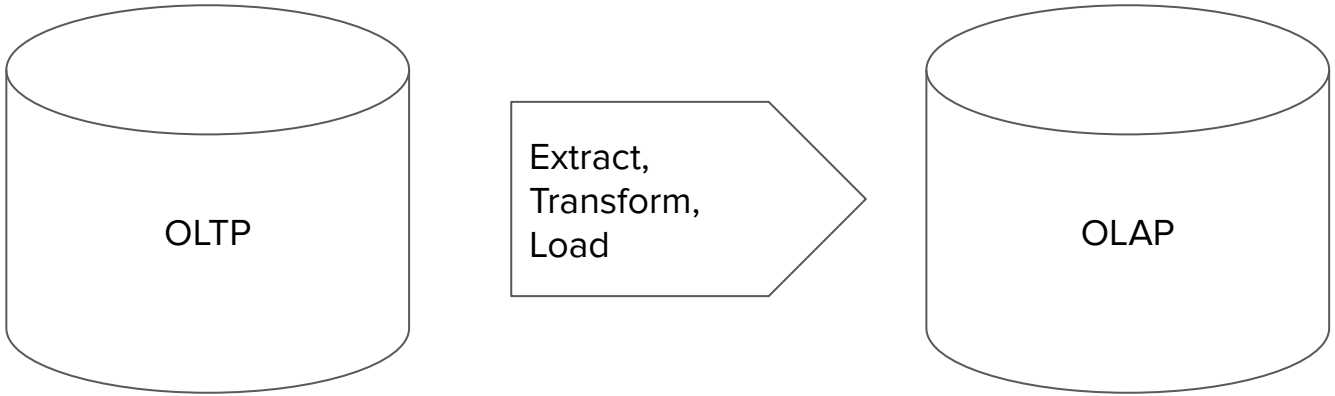
- Different objective functions
- Feature engineering
- More data
- Ensembling

Slide Credit: cs329s, stanford

Phase 4: Complex ML models

Slide Credit: cs329s, stanford

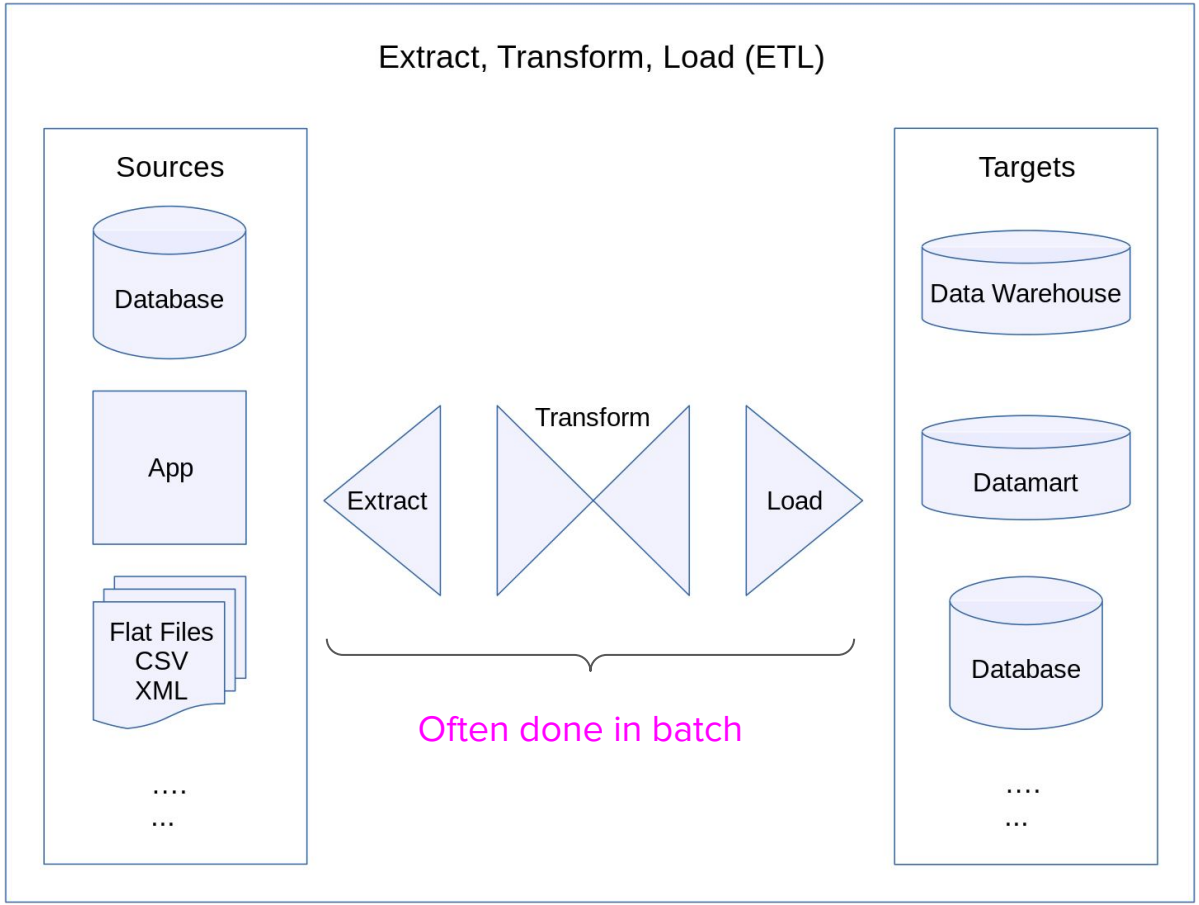
ETL (Extract, Transform, Load)



Transform: the meaty part

- cleaning, validating, transposing, deriving values, joining from multiple sources, deduplicating, splitting, aggregating, etc.

Slide Credit: cs329s, stanford



Slide Credit: cs329s, stanford

https://commons.wikimedia.org/wiki/File:Extract,_Transform,_Load_Data_Flow_Diagram.svg

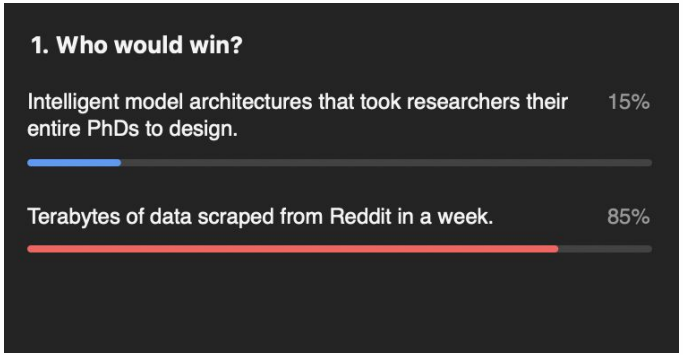
Training Data

1. Data
2. Labeling
3. Sampling
4. Class imbalance

Slide Credit: cs329s, stanford

WHO WOULD WIN?

Intelligent model architectures that took researchers their entire PhDs to design



Terabytes of data scraped from Reddit in a week

From last year

Slide Credit: cs329s, stanford

⚠️ Data: full of potential for biases ⚠️

- sampling/selection biases
- under/over-representation of subgroups
- human biases embedded in historical data
- labeling biases
- ...

Algorithmic biases not covered (yet)!

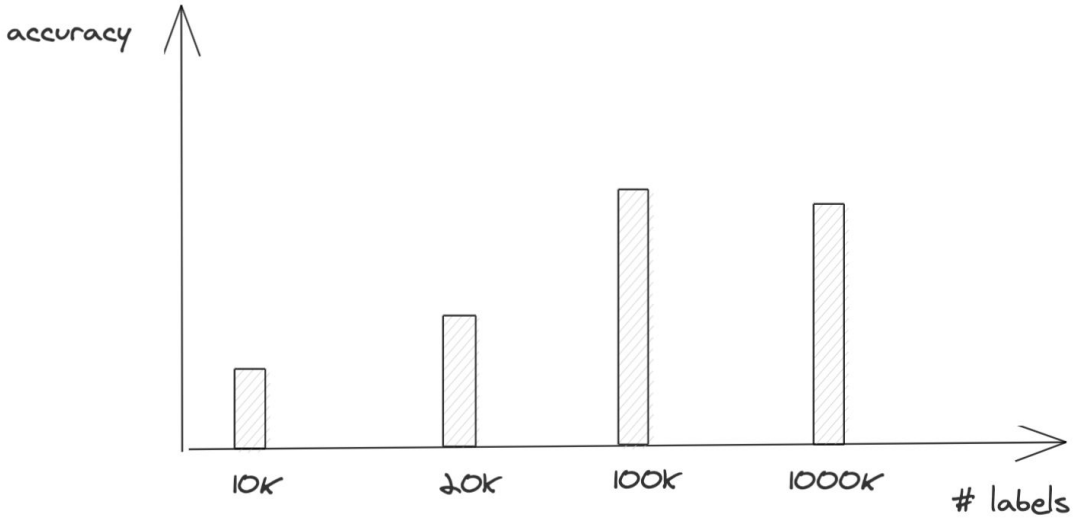
Slide Credit: cs329s, stanford

Labeling Data

1. Hand-labeling
2. Programmatic labeling
3. Weak supervision, semi supervision, active learning, transfer learning

Slide Credit: cs329s, stanford

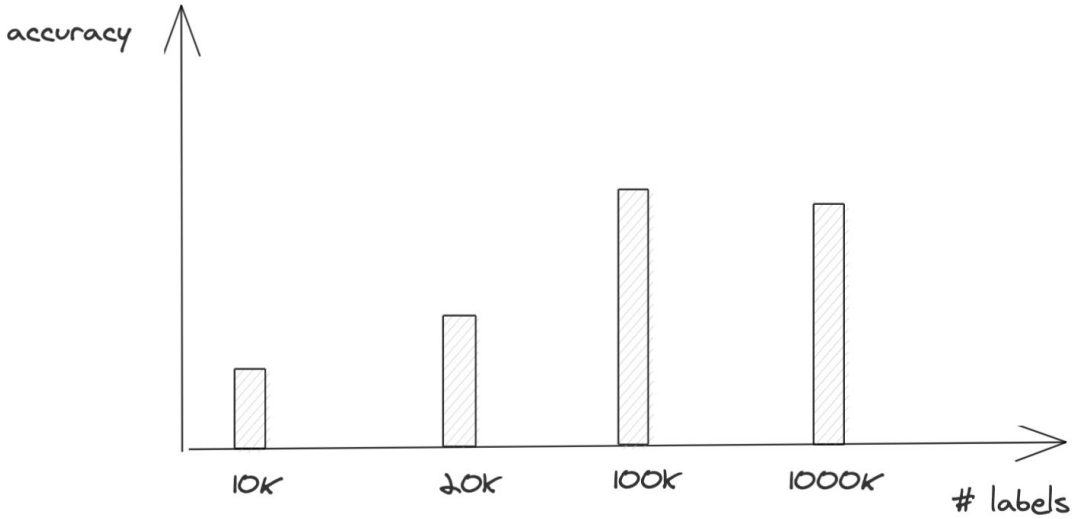
⚠️ More data isn't always better ⚠️



Why is the model getting worse?

Slide Credit: cs329s, stanford

⚠️ Label sources with varying accuracy ⚠️



- 100K labels: internally labeled, high accuracy
- 1M labels: crowdsourced, noisy

Slide Credit: cs329s, stanford

Label multiplicity: example

Task: label all entities in the following sentence:

Darth Sidious, known simply as the Emperor, was a Dark Lord of the Sith
who reigned over the galaxy as Galactic Emperor of the First Galactic Empire.

Annotator	# entities	Annotation
1	3	[**Darth Sidious**], known simply as the Emperor, was a [**Dark Lord of the Sith**] who reigned over the galaxy as [**Galactic Emperor of the First Galactic Empire**]
2	6	[**Darth Sidious**], known simply as the [**Emperor]**, was a **[Dark Lord**] of the [**Sith**] who reigned over the galaxy as [**Galactic Emperor**] of the [**First Galactic Empire**].
3	4	[**Darth Sidious**], known simply as the [**Emperor]**, was a **[Dark Lord of the Sith**] who reigned over the galaxy as [**Galactic Emperor of the First Galactic Empire**].

Slide Credit: cs329s, stanford

Label multiplicity: solution

- Clear problem definition
  - Pick the entity that comprises the longest substring

Annotator	# entities	Annotation
1	3	[**Darth Sidious**], known simply as the Emperor, was a [**Dark Lord of the Sith**] who reigned over the galaxy as [**Galactic Emperor of the First Galactic Empire**]
2	6	[**Darth Sidious**], known simply as the [**Emperor**], was a [**Dark Lord**] of the [**Sith**] who reigned over the galaxy as [**Galactic Emperor**] of the [**First Galactic Empire**].
3	4	[**Darth Sidious**], known simply as the [**Emperor**], was a [**Dark Lord of the Sith**] who reigned over the galaxy as [**Galactic Emperor of the First Galactic Empire**].

Slide Credit: cs329s, stanford

Label multiplicity: solution

- Clear problem definition
- Annotation training
- Data lineage: track where data/labels come from
- Learning methods with noisy labels
  - Learning with Noisy Labels (Natarajan et al., 2013)
  - Loss factorization, weakly supervised learning and label noise robustness (Patrini et al., 2016)
  - Cost-Sensitive Learning with Noisy Labels (Natarajan et al., 2018)
  - Confident Learning: Estimating Uncertainty in Dataset Labels (Northcutt et al., 2019)

Slide Credit: cs329s, stanford

Programmatic labeling

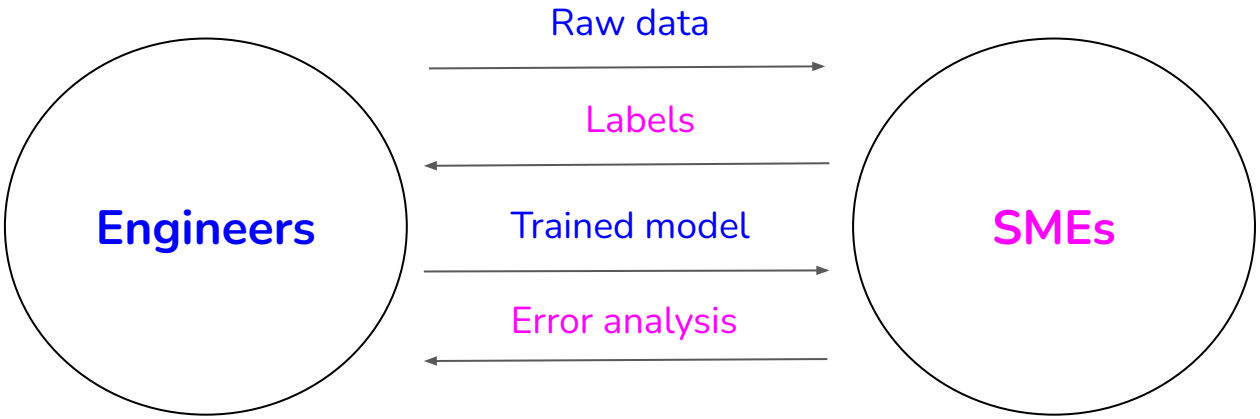
Slide Credit: cs329s, stanford

Hand labeling data is ...

- **Expensive:** Esp. when **subject matter expertise** required
- **Non-private:** Need to ship data to human annotators
- **Slow:** Time required scales linearly with # labels needed
- **Non-adaptive:** Every change requires re-labeling the dataset

Slide Credit: cs329s, stanford

SME as labeling functions




```
def function:
   if X:
      do Y
```

```
def labeling_function(note):
   if "pneumonia" in note:
      return "EMERGENT"
```

Labeling functions (LFs): Encode SME heuristics as function and use them to label training data *programmatically*

snorkel

Slide Credit: cs329s, stanford

LFs: can express many different types of heuristics

Pattern Matching	If a phrase like “send money” is in email
Boolean Search	If unknown_sender AND foreign_source
DB Lookup	If sender is in our Blacklist.db
Heuristics	If SpellChecker finds 3+ spelling errors
Legacy System	If LegacySystem votes spam
Third Party Model	If BERT labels an entity “diet”
Crowd Labels	If Worker #23 votes spam

Slide Credit: cs329s, stanford

LFs: powerful but noisy

```
def LF_contains_money(x):
    if "money" in x.body.text:
        return "SPAM"
```

```
def LF_from_grandma(x):
    if x.sender.name is "Grandma":
        return "HAM"
```

```
def LF_contains_money(x):
    if "free money" in x.body.text:
        return "SPAM"
```

From: **Grandma**

"Dear handsome grandson,
Since you can't be home for Thanksgiving dinner this year, I'm sending you some **money** so you could enjoy a nice meal …"

??

"You have been pre-approved for free **cash** …"

??

- **Noisy**: Unknown, inaccurate
- **Overlapping**: LFs may be correlated
- **Conflicting**: different LFs give different labels
- **Narrow**: Don't generalize well

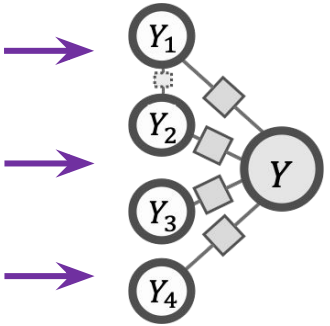
Slide Credit: cs329s, stanford

LF labels are combined to generate ground truths

```
def LF_contains_money(x):
    if "money" in x.body.text:
        return "SPAM"
```

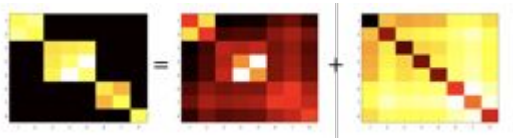
```
def LF_from_grandma(x):
    if x.sender.name is "Grandma":
        return "HAM"
```

```
def LF_contains_money(x):
    if "free money" in x.body.text:
        return "SPAM"
```



[Intuition]

Look at agreements & disagreements



$$(\Sigma^{-1})_O = \Sigma_O^{-1} + zz^T$$

Provably consistent matrix completion-style algorithm over inverse covariance

[Ratner et. al. NeurIPS'16;
Bach et. al. ICML'17;
Ratner et. al. AAAI'19;
Varma et. al. ICML'19l;
Sala et. al. NeurIPS'19;
Fu et. al. ICML'20]

Slide Credit: cs329s, stanford

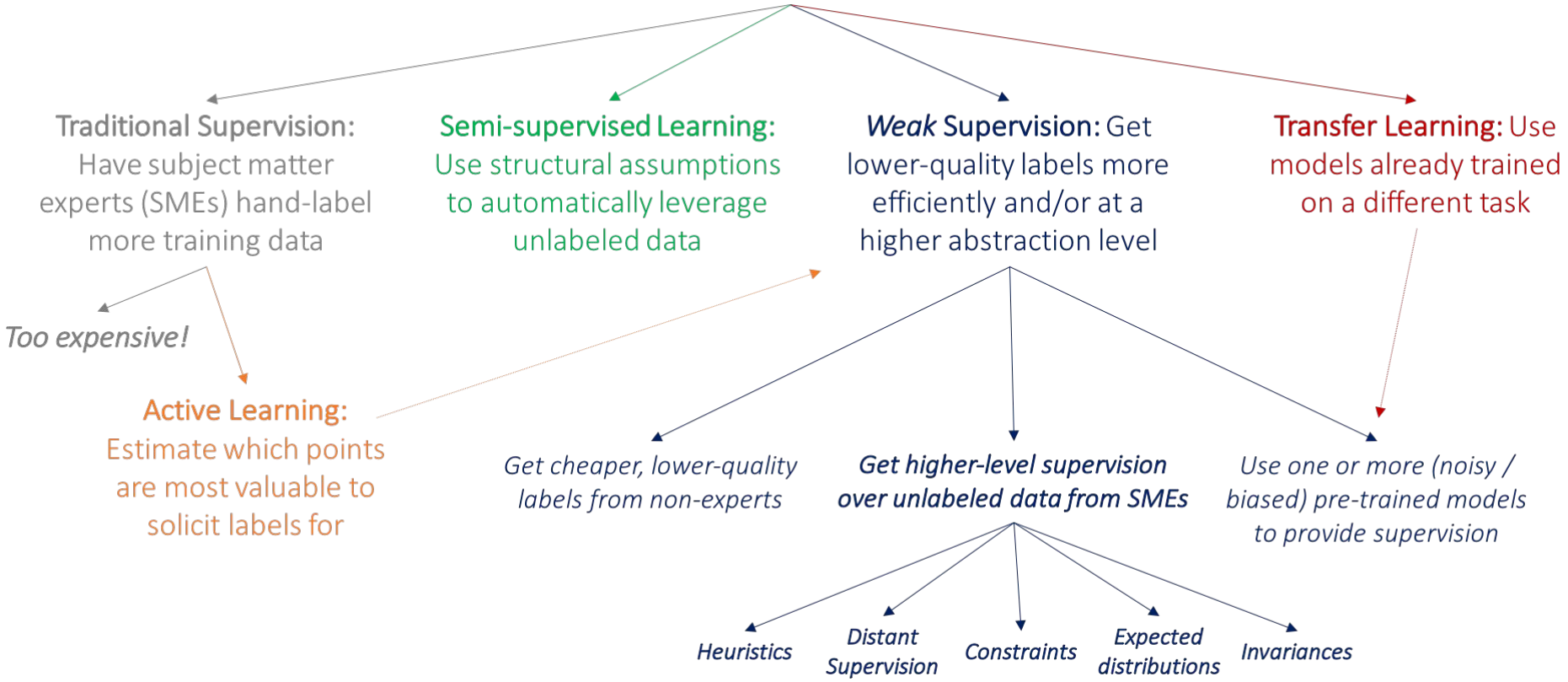
Hand labeling	Programmatic labeling
Expensive: esp. when subject matter expertise required	**Cost saving**: Expertise can be versioned, shared, reused across organization
Non-private: Need to ship data to human annotators	**Privacy**: Create LFs using a cleared data subsample then apply LFs to other data without looking at individual samples.
Slow: Time required scales linearly with # labels needed	**Fast**: Easily scale 1K -> 1M samples
Non-adaptive: Every change requires re-labeling the dataset	**Adaptive**: When changes happen, just reapply LFs!

Slide Credit: cs329s, stanford

Weak supervision,
semi-supervision,
active learning,
transfer learning

Slide Credit: cs329s, stanford

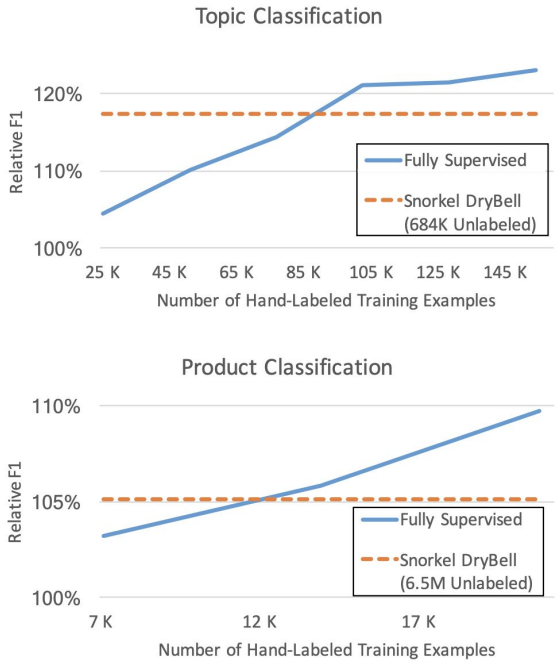
How to get more labeled training data?



Slide Credit: cs329s, stanford

Weak supervision

- Leverage noisy, imprecise sources to create labels
  - e.g. if “money” is in an email it’s probably spam



Slide Credit: cs329s, stanford

Harnessing Organizational Knowledge for Machine Learning (Ratner et al., Google AI Blog 2019)

Semi-supervision

- Use structural assumptions to leverage a large amount of unlabeled data together with a small amount of labeled data
  - Hashtags in the same profile/tweet are probably of similar topics

Slide Credit: cs329s, stanford

Semi-supervision

- Use structural assumptions to leverage a large amount of unlabeled data together with a small amount of labeled data
- Might require complex algorithms like clustering to discover similarity

Slide Credit: cs329s, stanford

Semi-supervision: self-training

1. Train model on a small set of labeled data
2. Use this model to generate predictions for unlabeled data
3. Use predictions with high raw probabilities as labels
4. Repeat step 1 with new labeled data

Slide Credit: cs329s, stanford

Semi-supervision: perturbation-based methods

Assumption: small perturbation wouldn't change a sample's label

- Add white noises to images
- Add small values to word embeddings

Also a data augmentation method!

Slide Credit: cs329s, stanford

Transfer learning

Apply model trained for one task to another task

1. Fine-tuning

Slide Credit: cs329s, stanford

Language Models are Few-Shot Learners (OpenAI 2020)

Transfer learning: fine-tuning

- fine-tuning only some layers
- fine-tuning the entire model

Slide Credit: cs329s, stanford

Language Models are Few-Shot Learners (OpenAI 2020)

Active learning

- **Goal**: Increase the efficiency of labels
- Label samples that are estimated to be most valuable to the model according to some metrics

Slide Credit: cs329s, stanford

Active Learning Literature Survey (Burr Settles, 2010)

Active learning metrics

- Uncertainty measurement
  - e.g. label samples with lowest raw probability for the predicted class

Slide Credit: cs329s, stanford

Active Learning Literature Survey (Burr Settles, 2010)

Active learning metrics

- Uncertainty measurement
- Candidate models' disagreement
  - Have several candidate models (e.g. models with different hypeparams)
  - Each model makes its own prediction
  - Label samples with most disagreement

Slide Credit: cs329s, stanford

Active Learning Literature Survey (Burr Settles, 2010)

Active learning

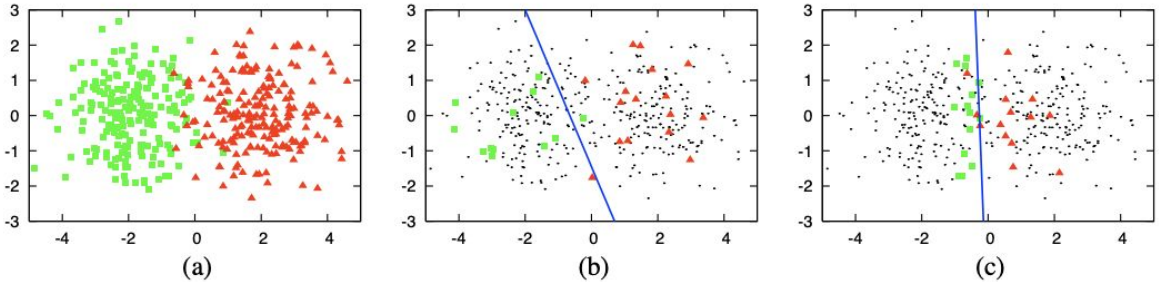


Figure 2: An illustrative example of pool-based active learning. (a) A toy data set of 400 instances, evenly sampled from two class Gaussians. The instances are represented as points in a 2D feature space. (b) A logistic regression model trained with 30 labeled instances randomly drawn from the problem domain. The line represents the decision boundary of the classifier (70% accuracy). (c) A logistic regression model trained with 30 actively queried instances using uncertainty sampling (90%).

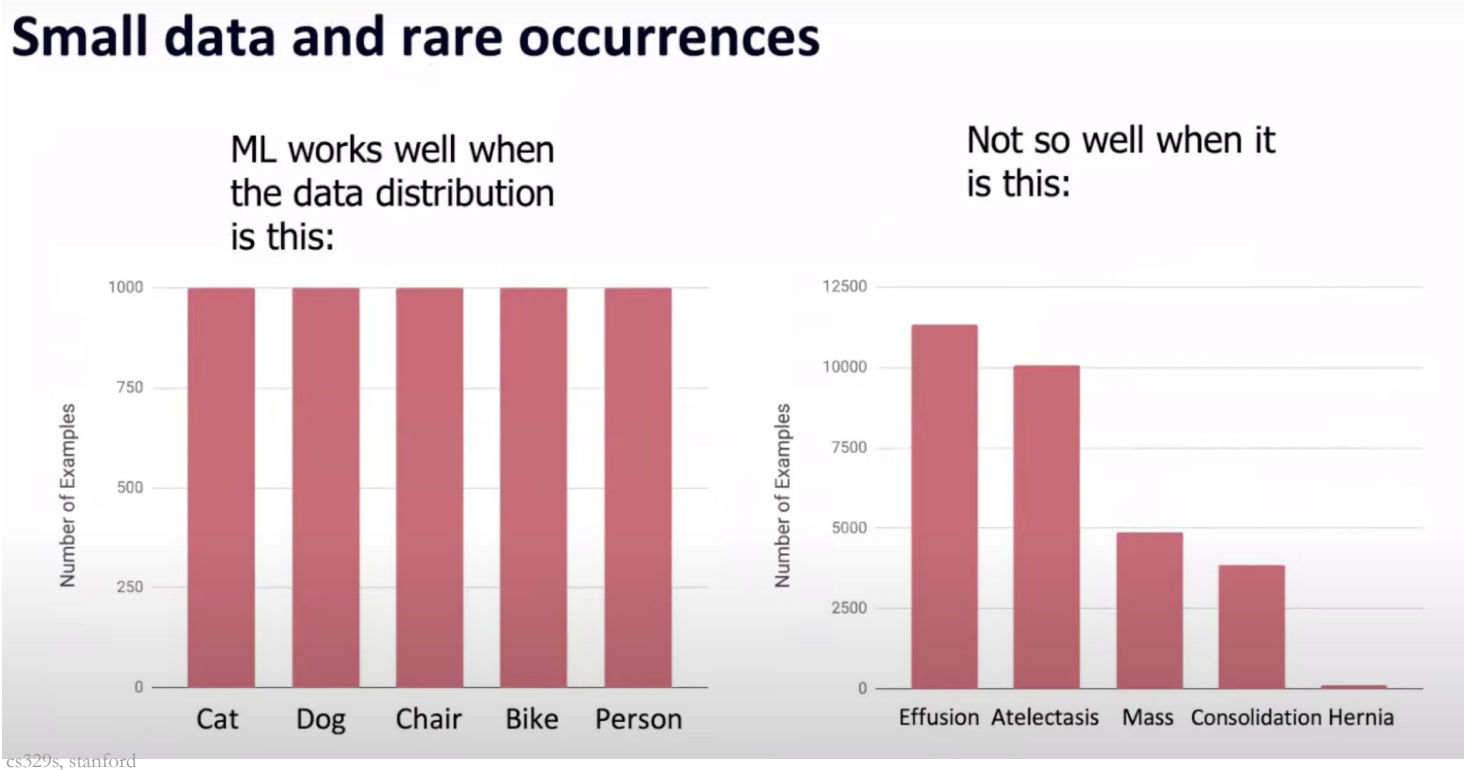
Slide Credit: cs329s, stanford

Active Learning Literature Survey (Burr Settles, 2010)

Method	How	Ground truths required?
Weak supervision	Leverages (often noisy) heuristics to generate labels	No, but a small number of labels is useful to guide the development of heuristics
Semi-supervision	Leverages structural assumptions to generate labels	Yes. A small number of initial labels as seeds to generate more labels
Transfer learning	Leverages models pretrained on another task for your new task	No for zero-shot learning Yes for fine-tuning, though # GTs required is often much less than # GTs required if training from scratch.
Active learning	Labels data samples that are most useful to your model	Yes

Slide Credit: cs329s, stanford

Class imbalance



Slide Credit: cs329s, stanford

Andrew Ng: Bridging AI's Proof-of-Concept to Production Gap (2020)

Why is class imbalance hard?

- Not enough signal to learn about rare classes

Slide Credit: cs329s, stanford

Why is class imbalance hard?

- Not enough signal to learn about rare classes
- Statistically, predicting majority label has higher chance of being right
  - If a majority class accounts 99% of data, always predicting it gives 99% accuracy

Slide Credit: cs329s, stanford

Why is class imbalance hard?

- Not enough signal to learn about rare classes
- Statistically, predicting majority label has higher chance of being right
- Asymmetric cost of errors: different cost of wrong predictions

Slide Credit: cs329s, stanford

Class imbalance is the norm

- Fraud detection
- Spam detection
- Disease screening
- Churn prediction
- Resume screening
  - E.g. 2% of resumes pass screening
- Object detection
  - Most bounding boxes don't contain any object

People are more interested in unusual/potentially catastrophic events

Slide Credit: cs329s, stanford

Image from PyImageSearch

How to deal with class imbalance

1. Choose the right metrics
2. Data-level methods
3. Algorithm-level methods

Slide Credit: cs329s, stanford

1. Choose the right metrics

Model A vs. Model B confusion matrices

Zoom poll:
Which model would you choose?

Model A	Actual CANCER	Actual NORMAL
Predicted CANCER	10	10
Predicted NORMAL	90	890

Model B	Actual CANCER	Actual NORMAL
Predicted CANCER	90	90
Predicted NORMAL	10	810

Slide Credit: cs329s, stanford

Choose the right metrics

Model A vs. Model B confusion matrices

Model B has a better chance of telling if you have cancer

Model A	Actual CANCER	Actual NORMAL
Predicted CANCER	10	10
Predicted NORMAL	90	890

Model B	Actual CANCER	Actual NORMAL
Predicted CANCER	90	90
Predicted NORMAL	10	810

Both have the same accuracy: 90%

Slide Credit: cs329s, stanford

Symmetric metrics vs. asymmetric metrics

Symmetric metrics	Asymmetric metrics
Treat all classes the same	Measures a model's performance w.r.t to a class
Accuracy	F1, recall, precision, ROC

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

$$F_1\text{−score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}$$

- TP: True positives
- TN: True negatives
- FP: False positives
- FN: False negatives

Slide Credit: cs329s, stanford

Class imbalance: asymmetric metrics

- Your model's performance w.r.t to a class

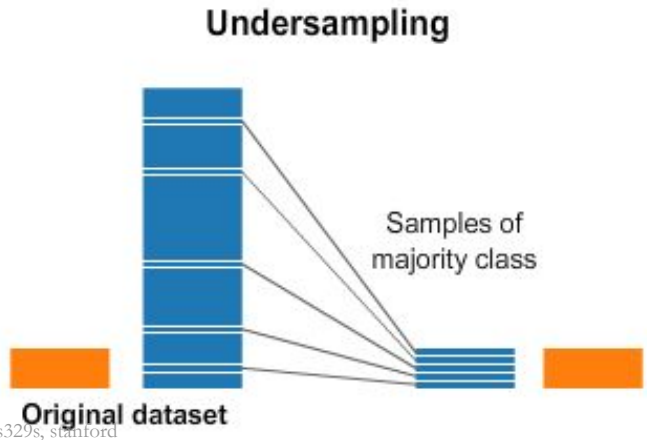
	CANCER (1)	NORMAL (0)	Accuracy	Precision	Recall	F1
Model A	10/100	890/900	0.9	0.5	0.1	0.17
Model B	90/100	810/900	0.9	0.5	0.9	0.64

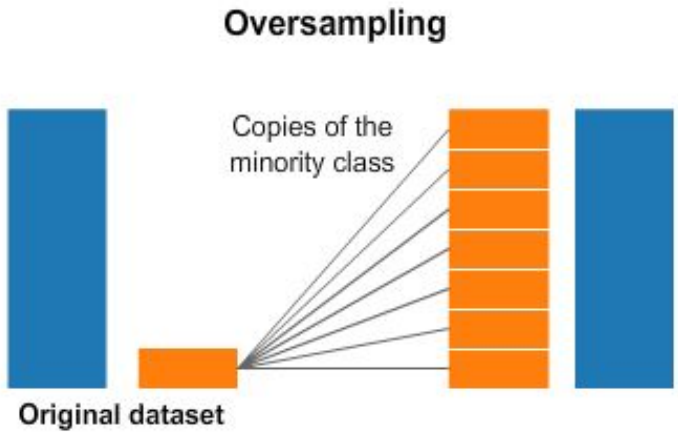
⚠️ F1 score for CANCER as 1 is different from F1 score for NORMAL as 1 ⚠️

Slide Credit: cs329s, stanford

2. Data-level methods: Resampling

Undersampling	Oversampling
Remove samples from the majority class	Add more examples to the minority class



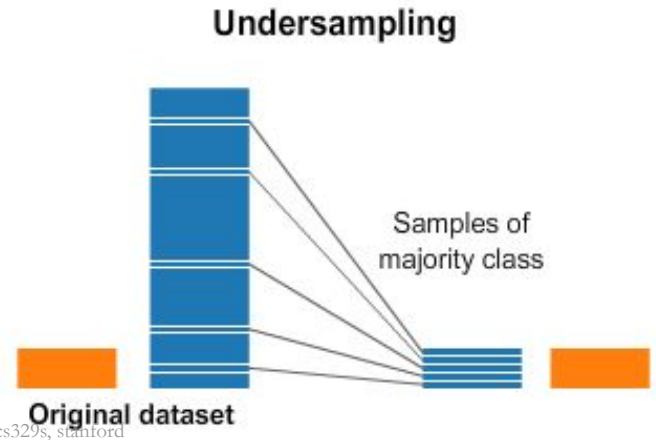


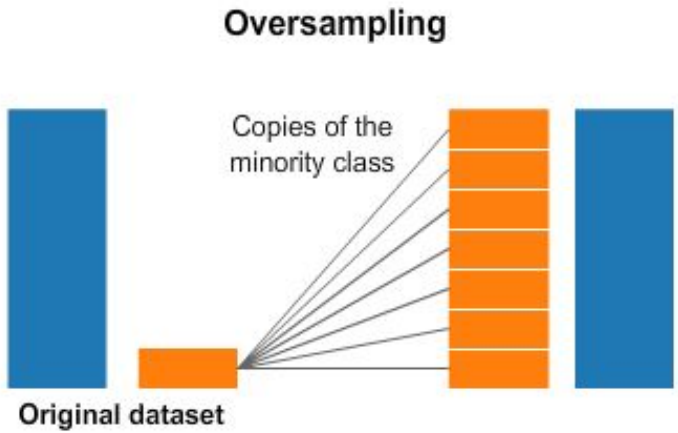
Slide Credit: cs329s, stanford

https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets#t1

2. Data-level methods: Resampling

Undersampling	**Oversampling**
Remove samples from the majority class	Add more examples to the minority class
Can cause overfitting	Can cause loss of information



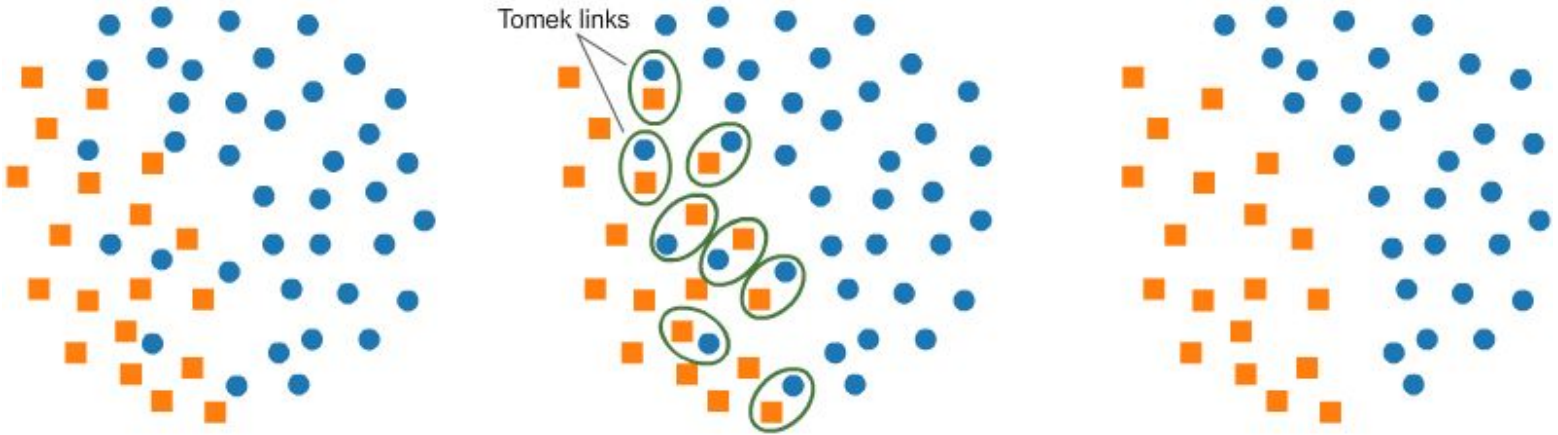


Slide Credit: cs329s, stanford

https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets#t1

Undersampling: Tomek Links

- Find pairs of close samples of opposite classes
- Remove the sample of majority class in each pair
  - Pros: Make decision boundary more clear
  - Cons: Make model less robust

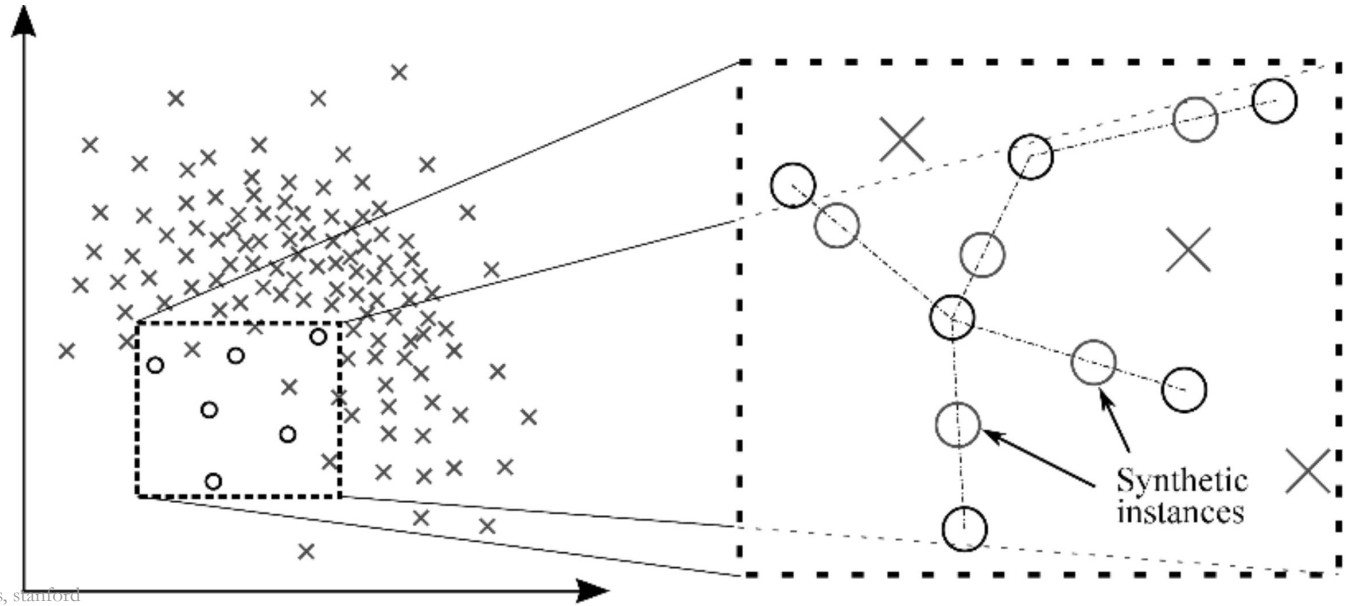


Slide Credit: cs329s, stanford

Image from https://www.kaggle.com/rafjaa/resampling-strategies-for-imbalanced-datasets

Oversampling: SMOTE

- Synthesize samples of minority class as convex (~linear) combinations of existing points and their nearest neighbors of same class.



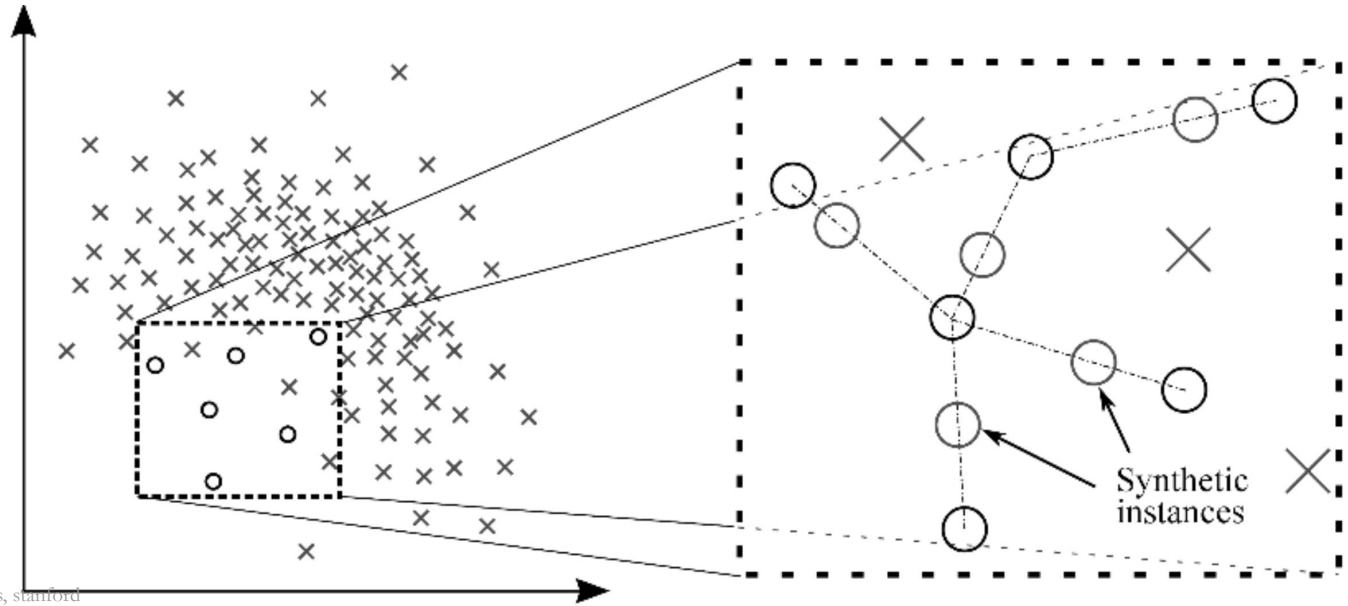
Slide Credit: cs329s, stanford

Image from Analytics Vidhya

Oversampling: SMOTE

Both SMOTE and Tomek links only work on low-dimensional data!

- Synthesize samples of minority class as convex (~linear) combinations of existing points and their nearest neighbors of same class.



Slide Credit: cs329s, stanford

Image from Analytics Vidhya

3. Algorithm-level methods

- Naive loss: all samples contribute equally to the loss
- Idea: training samples we care about should contribute more to the loss

$$L(X;\ \theta) = \sum_{x} L(x\ ;\theta)$$

Slide Credit: cs329s, stanford

3. Algorithm-level methods

- Cost-sensitive learning
- Class-balanced loss
- Focal loss

Slide Credit: cs329s, stanford

Cost-sensitive learning

- C_{ij}: the cost if class i is classified as class j

	Actual NEGATIVE	Actual POSITIVE
Predicted NEGATIVE	$C(0,\ 0) = C_{00}$	$C(1,\ 0) = C_{10}$
Predicted POSITIVE	$C(0,\ 1) = C_{01}$	$C(1,\ 1) = C_{11}$

- The loss caused by instance x of class i will become the weighted average of all possible classifications of instance x.

$$L(x\ ;\theta) = \sum_{j} C_{ij} P(j \mid x;\ \theta)$$

Slide Credit: cs329s, stanford

Class-balance loss

- Give more weight to rare classes

Non-weighted loss

$$L(X;\ \theta)\ =\ \sum_i L(x_i;\theta)$$

Weighted loss

$$L(X;\ \theta)\ =\ \sum_i W_{y_i} L(x_i;\theta)$$

$$W_c\ =\ \frac{N}{number\ of\ samples\ of\ class\ C}$$

```
model.fit(features, labels, epochs=10, batch_size=32, class_weight={“fraud”: 0.9, “normal”: 0.1})
```

Slide Credit: cs329s, stanford

Focal loss

- Give more weight to difficult samples:
  - downweighs well-classified samples

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

$$\mathrm{CE}(p_t) = -\log(p_t)$$

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$$

Slide Credit: cs329s, stanford

Focal Loss for Dense Object Detection (Lin et al., 2017)

1. Data augmentation

"Data augmentation is the new feature engineering"

- Josh Wills, prev Director of Data Engineering @ Slack

Slide Credit: cs329s, stanford

Data augmentation: goals

- Improve model's performance overall or on certain classes
- Generalize better
- Enforce certain behaviors

Slide Credit: cs329s, stanford

Data augmentation

1. Simple label-preserving transformation
2. Perturbation
3. Data synthesis

Slide Credit: cs329s, stanford

Label-preserving: Computer Vision

Random cropping, flipping, erasing, etc.

Image from [An Efficient Multi-Scale Focusing Attention Network for Person Re-Identification](#) (Huang et al., 2021)

Label-preserving: NLP

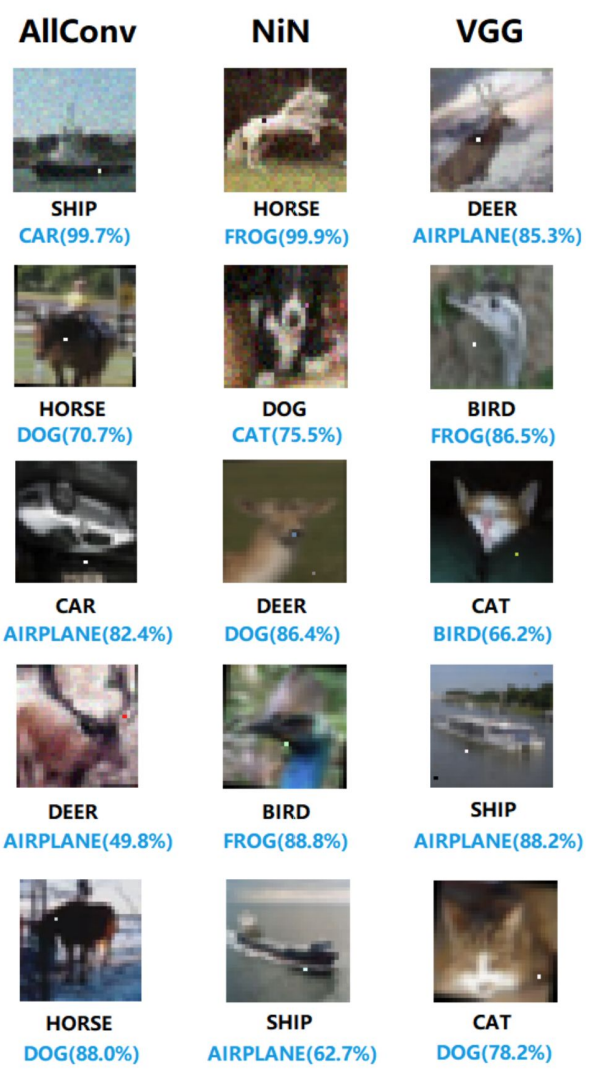
Original sentences	I'm so happy to see you.
Generated sentences	I'm so **glad** to see you. I'm so happy to see **y'all**. I'm **very** happy to see you.

Slide Credit: cs329s, stanford

Perturbation: neural networks can be sensitive to noise

- 67.97% Kaggle CIFAR-10 test images
- 16.04% ImageNet test images

can be misclassified by changing just one pixel (Su et al., 2017)



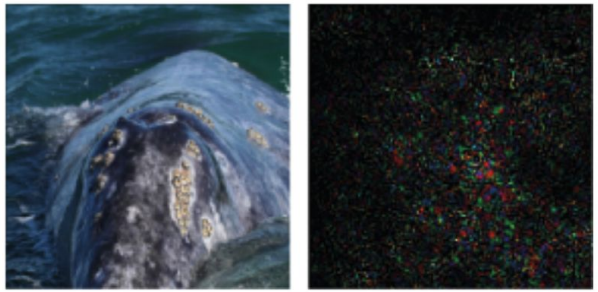
Slide Credit: cs329s, stanford

Perturbation: Computer Vision

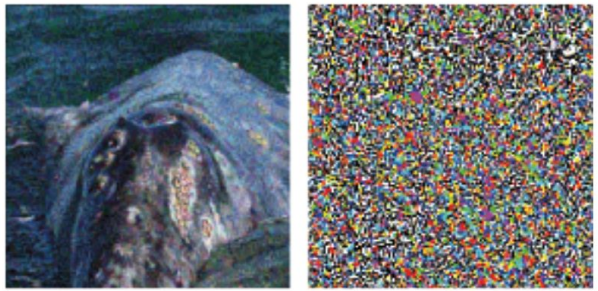
- Random noise
- Search strategy
  - DeepFool (Moosavi-Dezfooli et al., 2016): find the minimal noise injection needed to cause a misclassification with high confidence.

Whale

Turtle
noise by DeepFool



Turtle
noise by fast gradient sign



Slide Credit: cs329s, stanford

Perturbation: NLP

- Random replacement
  - e.g. BERT (10% * 15% = 1.5%)

- 80% of the time: Replace the word with the `[MASK]` token, e.g., `my dog is hairy` → `my dog is [MASK]`
- 10% of the time: Replace the word with a random word, e.g., `my dog is hairy` → `my dog is apple`
- 10% of the time: Keep the word unchanged, e.g., `my dog is hairy` → `my dog is hairy`. The purpose of this is to bias the representation towards the actual observed word.

Slide Credit: cs329s, stanford

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al., 2018)

Data synthesis: NLP

- Template-based
  - Very common in conversational AI
- Language model-based

Template	Find me a [CUISINE] restaurant within [NUMBER] miles of [LOCATION].
Generated queries	• Find me a **Vietnamese** restaurant within **2** miles of **my office**. • Find me a **Thai** restaurant within **5** miles of **my home.** • Find me a **Mexican** restaurant within **3** miles of **Google headquarters**.

Slide Credit: cs329s, stanford

Data Synthesis: Computer Vision

- Mixup
  - Create convex combination of samples of different classes
    - Labels: cat [3], dog [4]
    - Mixup: 30% dog, 70% cat [0.3 * 3 + 0.7 * 4 = 3.7]

Slide Credit: cs329s, stanford

mixup: Beyond Empirical Risk Minimization (Zhang et al., 2017)

Data Synthesis: Computer Vision

- Mixup
  - Incentivize models to learn linear relationships
  - Improves generalization on speech and tabular data
  - Can be used to stabilize the training of GANs

Slide Credit: cs329s, stanford

https://forums.fast.ai/t/mixup-data-augmentation/22764

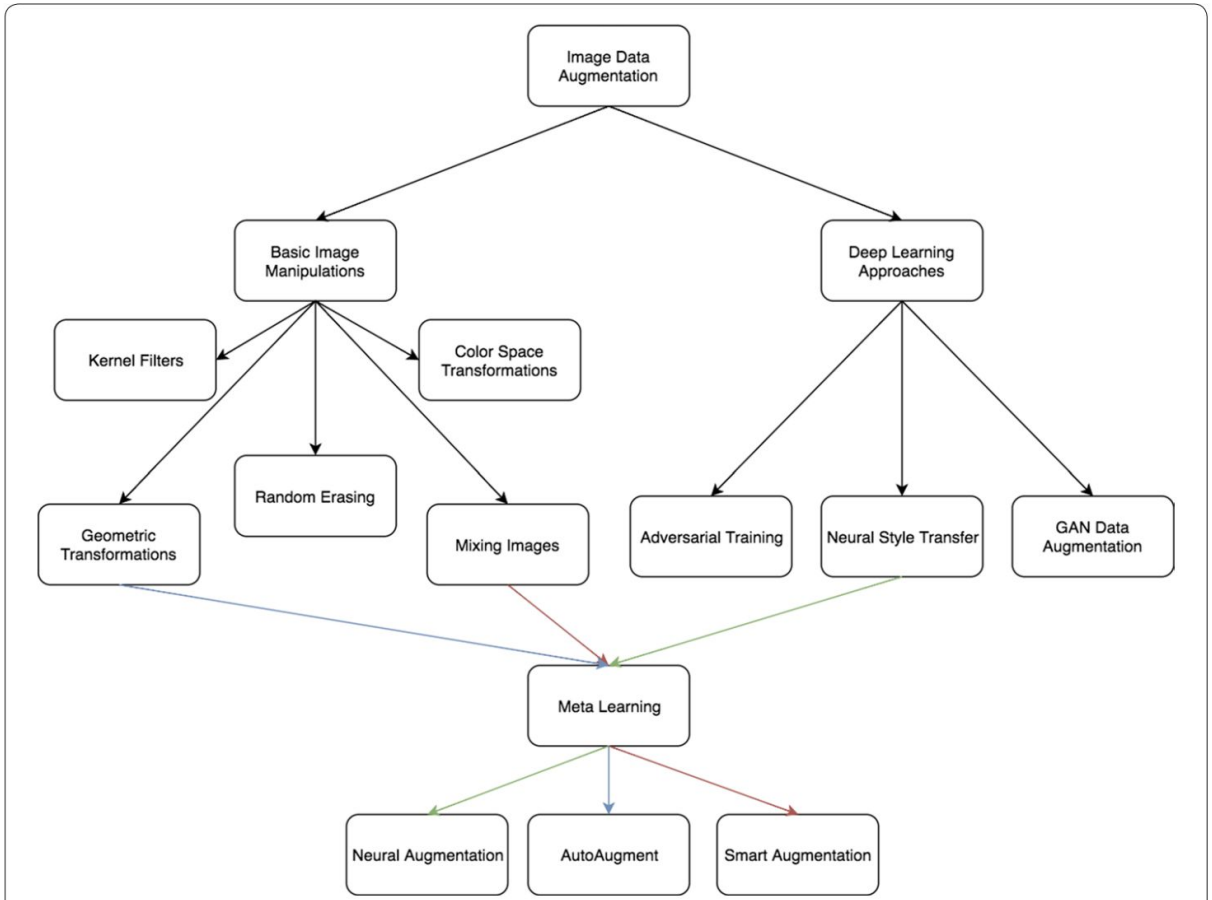


Fig. 2 A taxonomy of image data augmentations covered; the colored lines in the figure depict which data augmentation method the corresponding meta-learning scheme uses, for example, meta-learning using Neural Style Transfer is covered in neural augmentation [36]

Slide Credit: cs329s, stanford

A survey on Image Data Augmentation for Deep Learning (Connor Shorten & Taghi M. Khoshgoftaar, 2019)

Data leakage

- Some form of the label “leaks” into the features
- This same information is not available during inference

Slide Credit: cs329s, stanford

Data leakage: example 1

- Problem: detect lung cancer from CT scans
- Data: collected from hospital A
- Performs well on test data from hospital A
- Performs poorly on test data from hospital B

Patient ID	Date	Doctor note	Medical record	Scanner type	CT scan

Slide Credit: cs329s, stanford

Data leakage: example 1

- Problem: detect lung cancer from CT scans
- Data: collected from hospital A
- Performs well on test data from hospital A
- Performs poorly on test data from hospital B

Patient ID	Date	Doctor note	Medical record	Scanner type	CT scan

At hospital A, when doctors suspect that a patient has lung cancer, they send that patient to a higher-quality scanner

Slide Credit: cs329s, stanford

Data leakage: example 2

- Problem: predicting how many views an article will get
- Data: historical data on the site
- Where might data leakage come from?

Article ID	Date	Title	Article	Author	Language	Translations

Slide Credit: cs329s, stanford

Data leakage: example 2

- Problem: predicting how many views an article will get
- Data: historical data on the site

Not leakage because author popularity also available during inference

Article ID	Date	Title	Article	Author	Language	Translations

The site only translate articles that are already gaining attention

Slide Credit: cs329s, stanford

Partition: shuffle then split

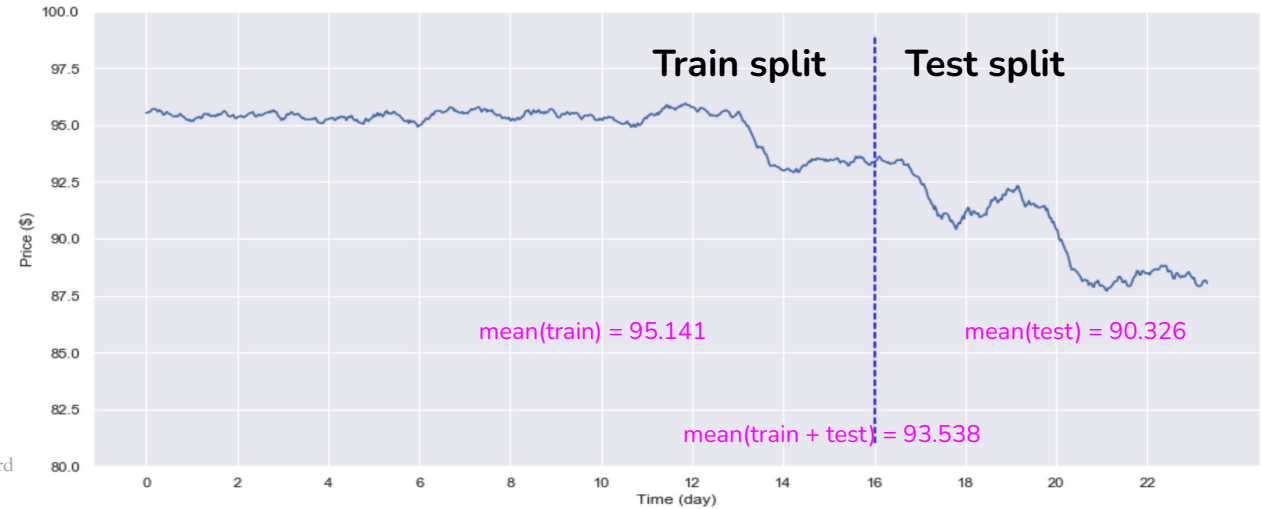
	Week 1	Week 2	Week 3	Week 4	Week 5
Test split	X11	X21	X31	X41	X51
Valid split	X12	X22	X32	X42	X52
Train split	X13	X23	X33	X43	X53
	X14	X24	X34	X44	X54
	...	...	...	...	...

Aim for similar distributions of labels across splits
e.g. each split has 90% NEGATIVE, 10% POSITIVE

Slide Credit: cs329s, stanford

2. Data processing before splitting

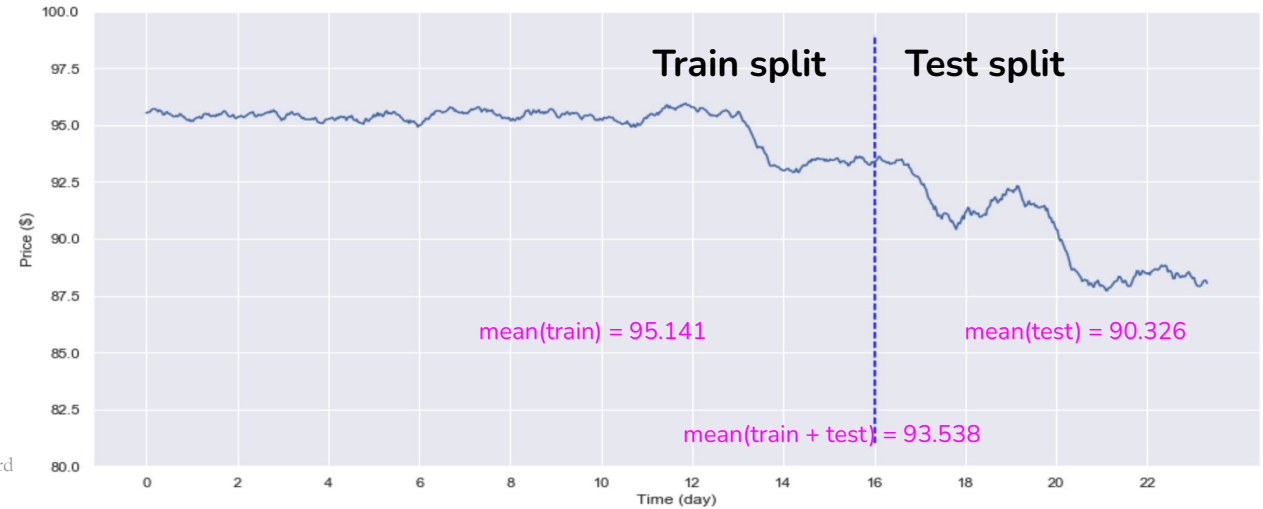
- Use the whole dataset (including valid/test) to generate global statistics/info
  - mean, variance, min, max, n-gram count, vocabulary, etc.
- Statistics are then used to process test data
  - scale, fill in missing values, etc.



Slide Credit: cs329s, stanford

2. Data processing before splitting

- Use the whole dataset (including valid/test) to generate global statistics/info
  - mean, variance, min, max, n-gram count, vocabulary, etc.
- Statistics are then used to process test data
  - scale, fill in missing values, etc.



Slide Credit: cs329s, stanford

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
    a. Test set includes data from the train set

Slide Credit: cs329s, stanford

3. Poor handling of data duplication before splitting

- Test set includes data from the train set
- Solution:
  - Deduplicate data before splitting
  - Oversample after splitting

Slide Credit: cs329s, stanford

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. Group leakage
    a. A group of examples have strongly correlated labels but are divided into different splits
    b. Example: CT scans of the same patient a week apart
    c. Solution: Understand your data and keep track of its metadata

Slide Credit: cs329s, stanford

Causes of data leakage

1. Splitting time-correlated data randomly instead of by time
2. Data processing before splitting
3. Poor handling of data duplication before splitting
4. Group leakage
5. Leakage from data generation & collection process
    a. Example: doctors send high-risk patients to a better scanner
    b. Solution: Data normalization + subject matter expertise

Slide Credit: cs329s, stanford

How to detect leakage?

1. Measure correlation of a feature with labels
    a. A feature alone might not cause leakage, but 2 features together might

Slide Credit: cs329s, stanford

How to detect leakage?

1. Measure correlation of a feature with labels
2. Feature ablation study
    a. If removing a feature causes the model performance to decrease significantly, figure out why.

Slide Credit: cs329s, stanford

How to detect leakage?

1. Measure correlation of a feature with labels
2. Feature ablation study
3. Monitor model performance as more features are added
    a. Sudden increase: either a very good feature or leakage!

Slide Credit: cs329s, stanford

Evaluating a feature

1. Feature importance
2. Feature generalization

Slide Credit: cs329s, stanford

Measuring a feature's importance

How much the model performance deteriorates
if a feature or a set of features containing that feature
is removed from the model?

Slide Credit: cs329s, stanford

Measuring a feature's importance

- XGBoost

get_score(*fmap='', importance_type='weight'*)

Get feature importance of each feature. For tree model Importance type can be defined as:

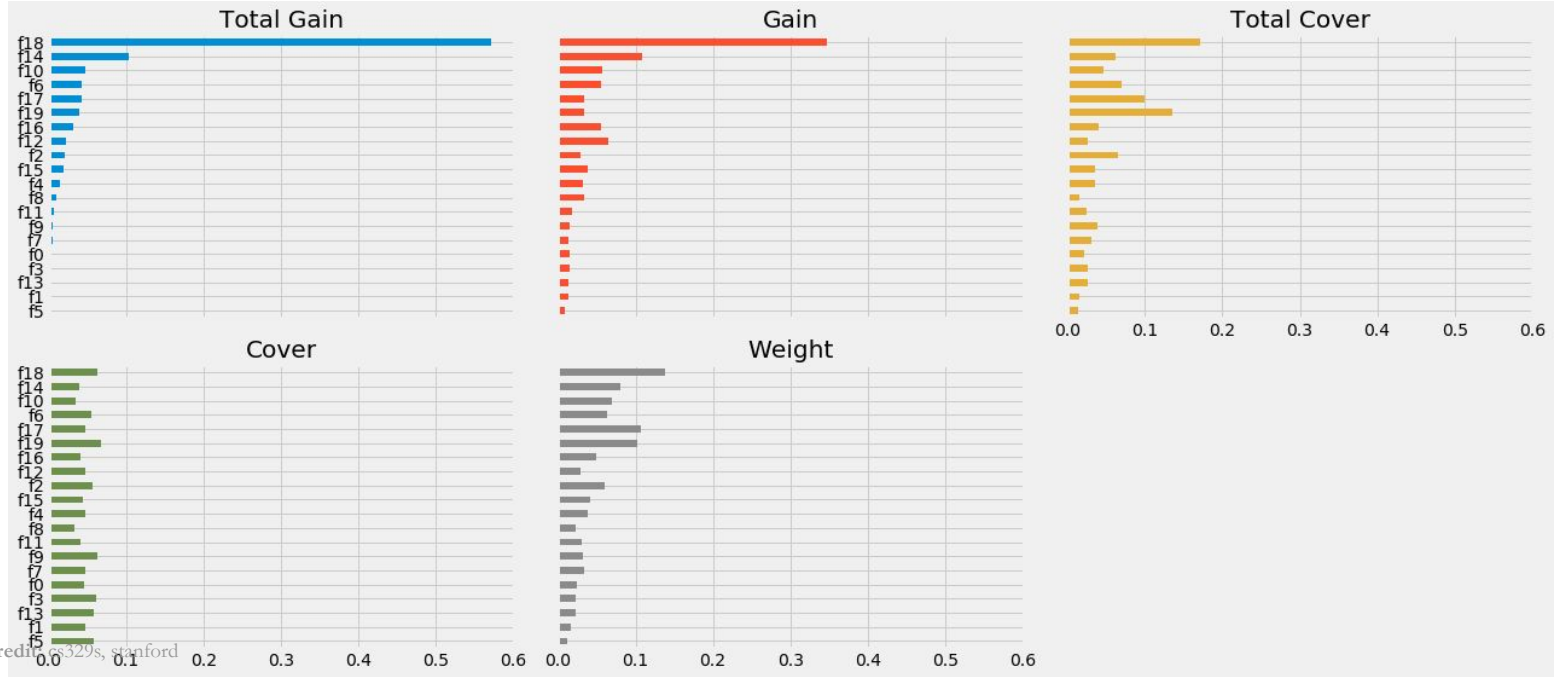
- 'weight': the number of times a feature is used to split the data across all trees.
- 'gain': the average gain across all splits the feature is used in.
- 'cover': the average coverage across all splits the feature is used in.
- 'total_gain': the total gain across all splits the feature is used in.
- 'total_cover': the total coverage across all splits the feature is used in.

Slide Credit: cs329s, stanford

https://xgboost.readthedocs.io/en/latest/python/python_api.html#xgboost.Booster.get_score

Measuring a feature's importance

- XGBoost

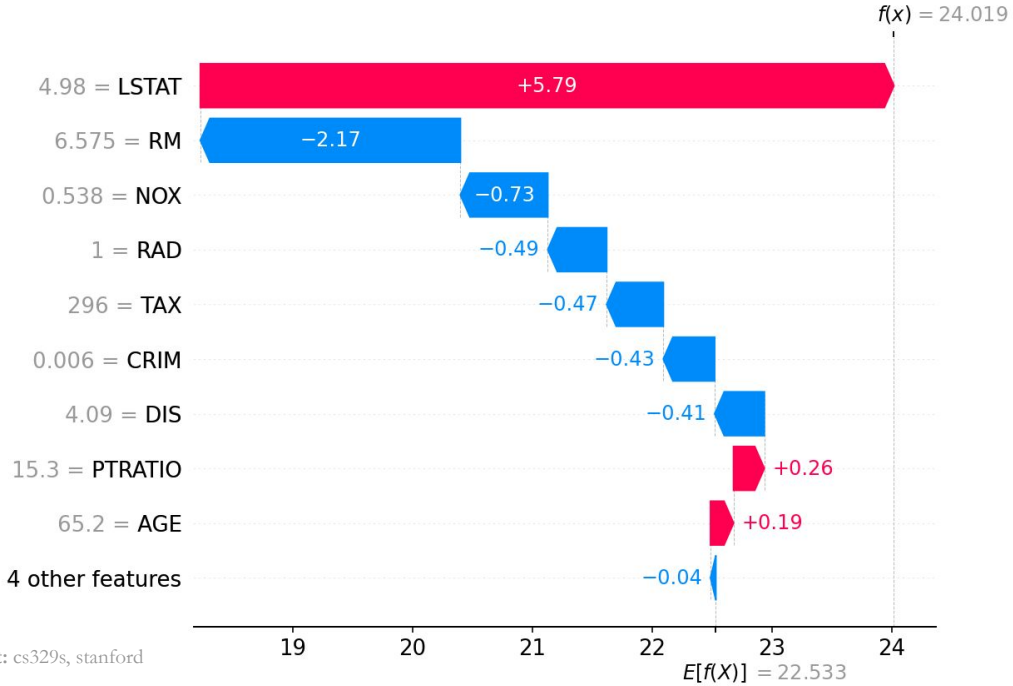


Slide Credit: cs329s, stanford

[The Multiple faces of 'Feature importance' in XGBoost](#) (Amjad Abu-Rmileh, 2019)

SHAP: SHapley Additive exPlanations

- Measuring a feature's contribution to a single prediction

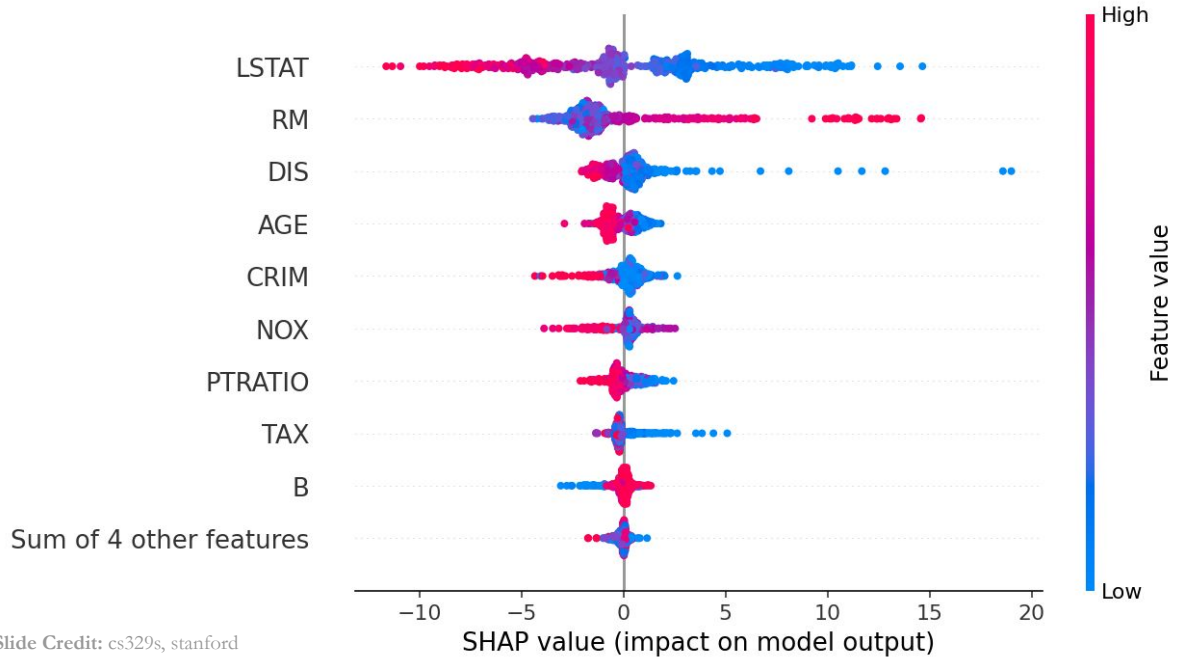


Slide Credit: cs329s, stanford

Image by Scott Lundberg from https://github.com/slundberg/shap

SHAP: SHapley Additive exPlanations

- Measuring a feature's contribution to the entire model

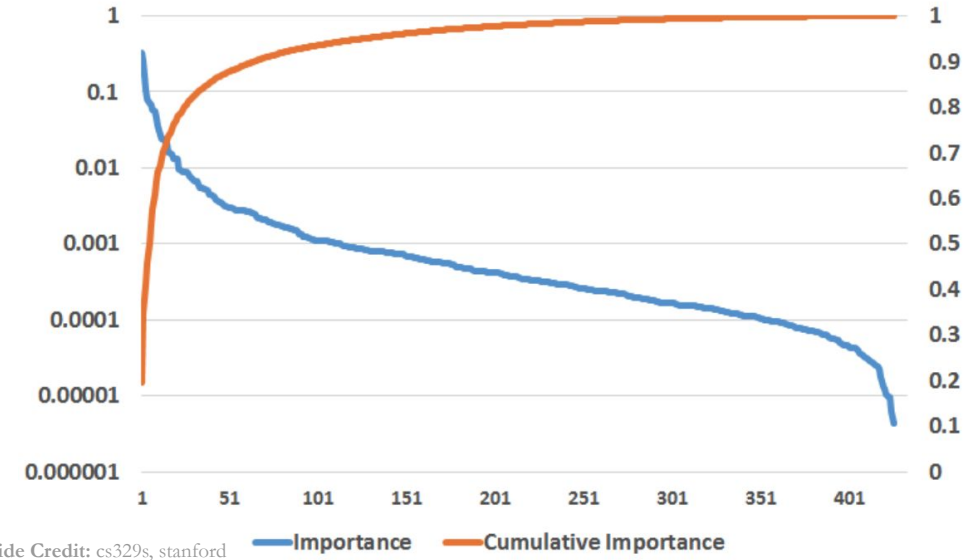


Slide Credit: cs329s, stanford

Image by Scott Lundberg from https://github.com/slundberg/shap

Measuring feature importance @ Facebook

- Top 10 features: 50% total feature importance
- Bottom 300 features: <1% total feature importance



Slide Credit: cs329s, stanford

Practical Lessons from Predicting Clicks on Ads at Facebook (He et al., 2014)

https://github.com/interpretml/interpret

```
ebm = ExplainableBoostingClassifier()
```

Slide Credit: cs329s, stanford

Feature engineering: the more the better?

- Adding more features tends to improve model performance

How can having too many features be bad?

Slide Credit: cs329s, stanford

Slide Credit: cs329s, stanford

Too many features can be bad …

- Training:
  - Overfitting
  - More features, more opportunity for data leakage

Slide Credit: cs329s, stanford

Too many features can be bad ...

- Training:
  - Overfitting
  - More features, more opportunity for data leakage
- Inference
  - Increase inference latency with online prediction
  - Might cause increased memory usage -> more expensive instance required

Slide Credit: cs329s, stanford

Too many features can be bad ...

- Training:
  - Overfitting
  - More features, more opportunity for data leakage
- Inference
  - Increase inference latency with online prediction
  - Might cause increased memory usage -> more expensive instance required
- Stale features become technical debts
  - E.g. if zip codes are no longer allowed for predictions, all features that use zip codes will need to be updated

Slide Credit: cs329s, stanford

Too many features can be bad ...

- Training:
  - Overfitting
  - More features, more opportunity for data leakage
- Inference
  - Increase inference latency with online prediction
  - Might cause increased memory usage -> more expensive instance required
- Stale features become technical debts

Solution:

- Clean up stale / ineffective features
- Store features in case you want to reuse them
  - Feature management

Slide Credit: cs329s, stanford

9 best practices for feature engineering

1. Split data by time instead of doing it randomly.
2. If you oversample your data, do it after splitting.
3. Use statistics/info from the train split, instead of the entire data, for feature engineering: scaling, normalizing, handling missing values, creating n-gram count, item encoding, etc.
4. Understand how your data is generated, collected, and processed. Involve domain experts if necessary.
5. Keep track of data lineage.
6. Understand feature importance to your model.
7. Measure correlation between features and labels.
8. Use features that generalize well.
9. Remove stale features from your models.

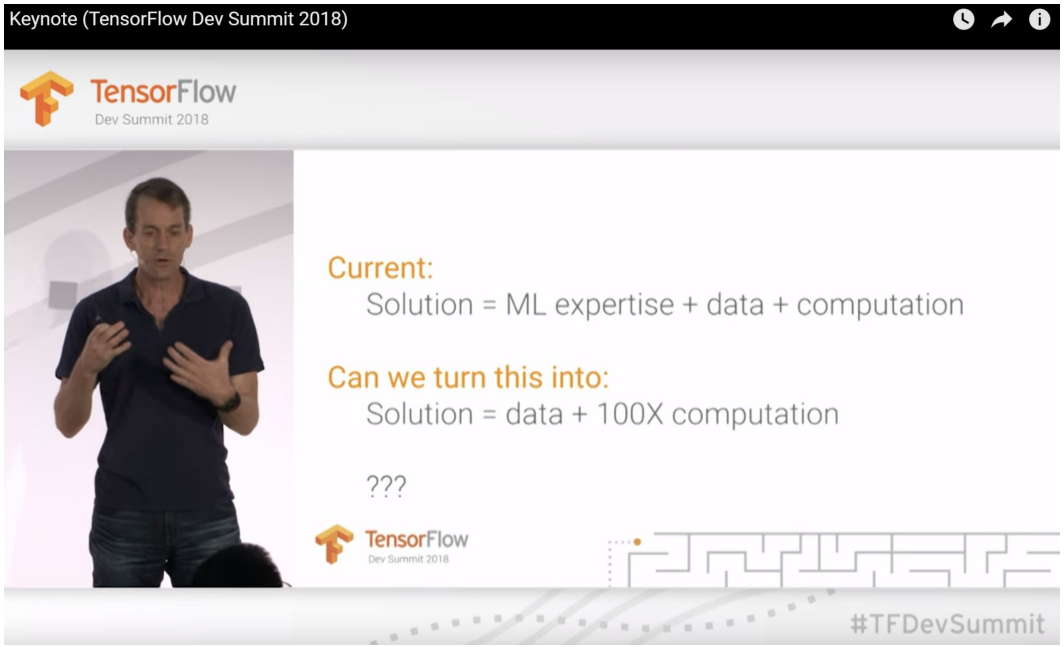
Slide Credit: cs329s, stanford

AutoML

Slide Credit: cs329s, stanford

AutoML

- A good ML researcher is someone who will automate themselves out of job
- Google: what if we replace ML experts with 100x compute?



Slide Credit: cs329s, stanford

AutoML

- Soft AutoML:
  - hyperparameter tuning
- Hard AutoML
  - neural architecture search
  - learned optimizer

More computationally expensive

Slide Credit: cs329s, stanford

Soft AutoML: Hyperparameter tuning

- Weaker models with well-tuned hyperparameters can outperform fancier models
  - On the State of the Art of Evaluation in Neural Language Models (Melis et al. 2018)

Slide Credit: cs329s, stanford

Soft AutoML: Hyperparameter tuning

- Many hyperparameters to tune

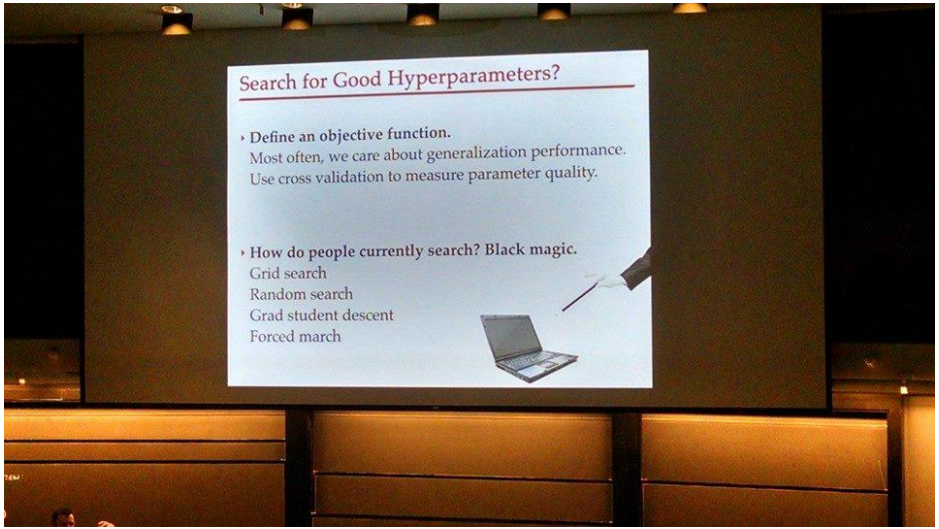
```
model_type = "bert"

def __init__(
    self,
    vocab_size=30522,
    hidden_size=768,
    num_hidden_layers=12,
    num_attention_heads=12,
    intermediate_size=3072,
    hidden_act="gelu",
    hidden_dropout_prob=0.1,
    attention_probs_dropout_prob=0.1,
    max_position_embeddings=512,
    type_vocab_size=2,
    initializer_range=0.02,
    layer_norm_eps=1e-12,
    pad_token_id=0,
    position_embedding_type="absolute",
    use_cache=True,
    classifier_dropout=None,
    **kwargs
):
```

Slide Credit: cs329s, stanford

https://github.com/huggingface/transformers/blob/master/src/transformers/models/bert/configuration_bert.py

Soft AutoML: Hyperparameter tuning

- Graduate Student Descent (GSD)
  - A graduate student fiddles around with the hyperparameters until the model works



Slide Credit: cs329s, stanford

https://twitter.com/GuyZys/status/592847074170896384/photo/1

Soft AutoML: Hyperparameter tuning

- Hyperparam tuning has become a standard part of ML workflows
- Built-in with frameworks
  - TensorFlow: Keras Turner
  - scikit-learn: auto-sklearn
  - Ray Tune
- Popular algos:
  - Random search
  - Grid search
  - Bayesian optimization

Slide Credit: cs329s, stanford

NAS: Neural architecture search

- **Search space**
  - Set of operations
    - e.g. convolution, fully-connected, pooling
  - How operations can be connected

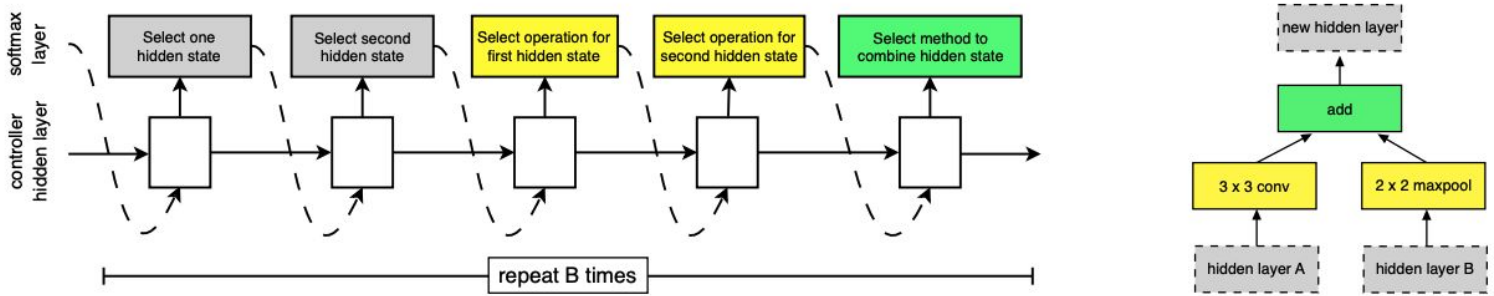


Figure 3. Controller model architecture for recursively constructing one block of a convolutional cell. Each block requires selecting 5 discrete parameters, each of which corresponds to the output of a softmax layer. Example constructed block shown on right. A convolutional cell contains B blocks, hence the controller contains $5B$ softmax layers for predicting the architecture of a convolutional cell. In our experiments, the number of blocks B is 5.

Slide Credit: cs329s, stanford

Learning Transferable Architectures for Scalable Image Recognition (Zoph et al., 2017)

NAS: Neural architecture search

- Search space
- **Performance estimation strategy**
  - How to evaluate **many** candidate architectures?
  - Ideal: should be done without having to re-construct or re-train them from scratch.

Slide Credit: cs329s, stanford

NAS: Neural architecture search

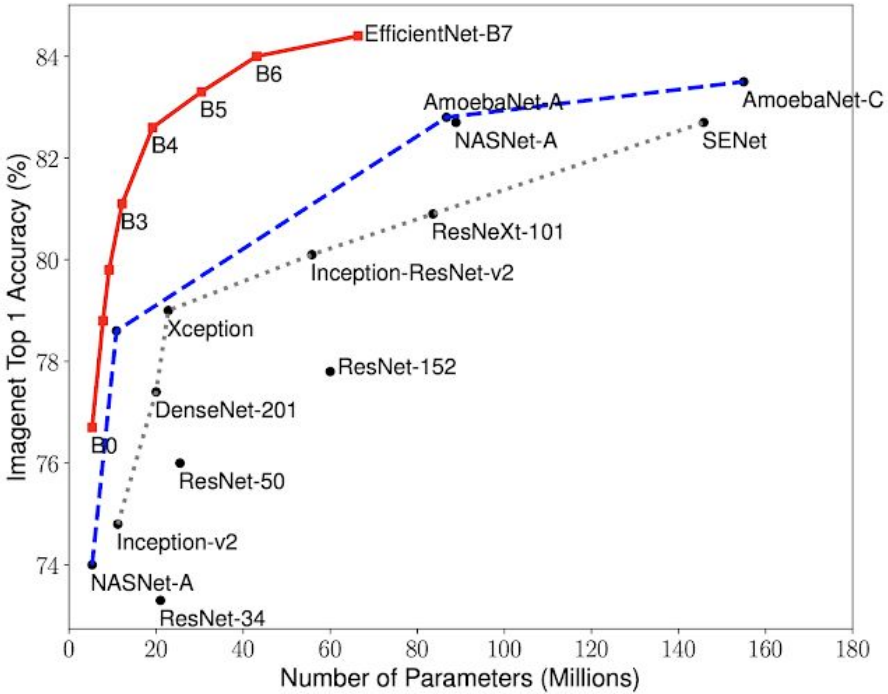
- Search space
- Performance estimation strategy
- **Search strategy**
  - Random
  - Reinforcement learning
    - reward the choices that improve performance estimation
  - Evolution
    - mutate an architecture
    - choose the best-performing offsprings
    - so on

Slide Credit: cs329s, stanford

NAS: Neural architecture search

- **Search space**
- **Performance estimation strategy**
- **Search strategy**

Very successful



Slide Credit: cs329s, stanford

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks (Tan et Le., 2019)

Evaluation methods

1. Perturbation Tests
2. Invariance Tests
3. Directional Expectation Tests
4. Model Calibration
5. Confidence Measurement
6. Slice-based Evaluation

Slide Credit: cs329s, stanford

Perturbation tests

- Problem: users input might contain noise, making it different from test data
  - Examples:
    - Speech recognition: background noise
    - Object detection: different lighting
    - Text inputs: typos, intentional misspelling (e.g. looooooooong)
  - Model does well on test set, but fails in production

Slide Credit: cs329s, stanford

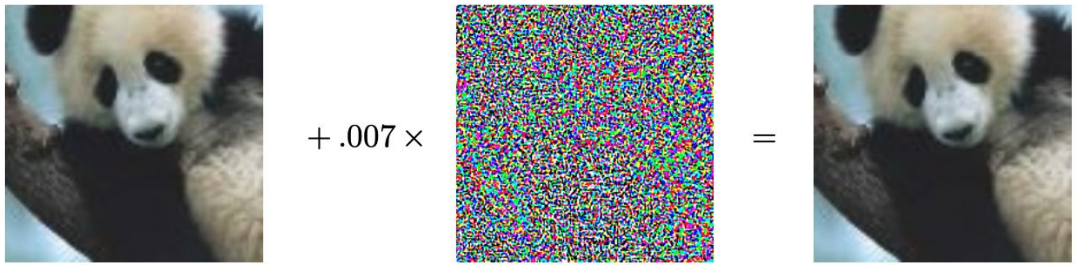
Perturbation tests

- Motivation: users input might contain noise, making it different from test data
- Idea: randomly add small noise to test data to see how much outputs change

Slide Credit: cs329s, stanford

Perturbation tests

- Motivation: users input might contain noise, making it different from test data
- Idea: randomly add small noise to test data to see how much outputs change
- The more sensitive the model is to noise:
  - The harder it is to maintain
  - The more vulnerable the model is to adversarial attacks



Slide Credit: cs329s, stanford

Perturbation tests

- Motivation: users input might contain noise, making it different from test data
- Idea: randomly add small noise to test data to see how much outputs change

If small changes cause model's performance to fluctuate, you might want to make model more robust:

- Add noise to training data
- Add more training data
- Choose another model

Slide Credit: cs329s, stanford

Invariance tests

- Motivation: some input changes shouldn’t lead to changes in outputs
  - Changing race/gender info shouldn’t change predicted approval outcome
  - Changing name shouldn’t affect resume screening results

> The Berkeley study found that both face-to-face and online lenders rejected a total of 1.3 million creditworthy black and Latino applicants between 2008 and 2015. Researchers said they believe the applicants "would have been accepted had the applicant not been in these minority groups." That's because when they used the income and credit scores of the rejected applications but deleted the race identifiers, the mortgage application was accepted.

Slide Credit: cs329s, stanford

Disparity in home lending costs minorities millions, researchers find (CBS News, 2019)

Invariance tests

- Motivation: some input changes shouldn't lead to changes in outputs
- Idea: keep certain features the same, but randomly change values of sensitive features

If changing sensitive features can change model's outputs, there might be biases!

Slide Credit: cs329s, stanford

Directional expectation tests

- Motivation: some changes to inputs should cause predictable changes in outputs
  - E.g. when predicting housing prices:
    - Increasing lot size shouldn't decrease the predicted price
    - Decreasing square footage shouldn't increase the predicted price

Slide Credit: cs329s, stanford

Directional expectation tests

- Motivation: some changes to inputs should cause predictable changes in outputs
- Idea: keep most features the same, but change certain features to see if outputs change predictably

If increasing lot size consistently reduces the predicted price, you might want to investigate why!

Slide Credit: cs329s, stanford

Model calibration

"One of the most important tests of a forecast — I would argue that it is the single most important one — is called calibration."

Nate Silver, **The Signal and the Noise**

Slide Credit: cs329s, stanford

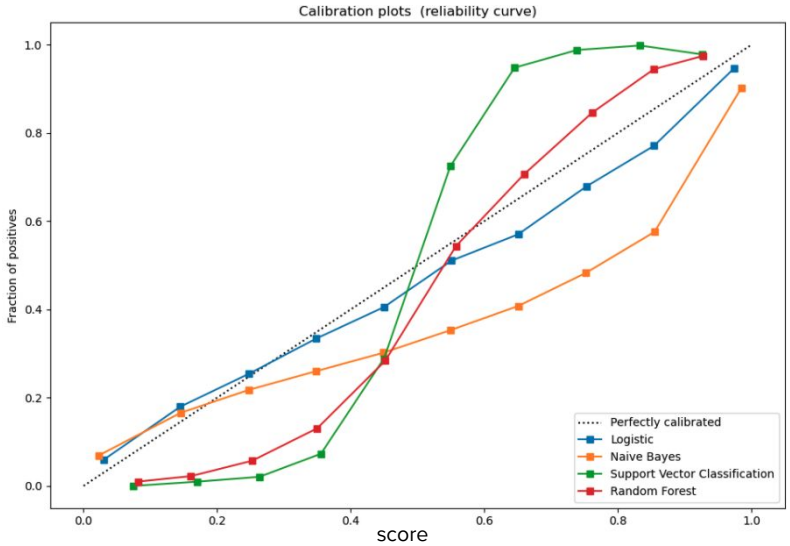
Model calibration

- If you predict team A wins in A vs. B match with 60% probability:
  - In 100 A vs. B match, A should win 60% of the time!

Slide Credit: cs329s, stanford

case

Among all samples predicted POSITIVE with propa 80%, .
80% of them should be POSITIVE



Need to ensure the **top class is correct on average**

Slide Credit: cs329s, stanford

Image from [Probability calibration](#) (sklearn)

Confidence measurement

- Usefulness threshold for each individual prediction
- Uncertain predictions can cause annoyance & catastrophic consequences

Slide Credit: cs329s, stanford

Confidence measurement

- How to measure the confidence level of each prediction?
- What to do with predictions below the confidence threshold?
  - Skip
  - Ask for more information
  - Loop in humans

Slide Credit: cs329s, stanford

Slice-based evaluation

Different performance on different slices

- Classes
  - Might perform worse on minority classes
- Subgroups
  - Gender
  - Location
  - Time of using the app
  - etc.

Slide Credit: cs329s, stanford

Same performance on different slices with different cost

- User churn prediction
  - Paying users are more critical
- Predicting adverse drug reactions
  - Patients with underlying conditions are more critical

⚠️ Focusing on improving only overall metrics might hurt performance on subgroups ⚠️

Slide Credit: cs329s, stanford

Slice-based evaluation: example

- Majority group: 90%
- Minority group: 10%

Zoom poll: Which model would you go with?

	Majority accuracy	**Minority accuracy**
Model A	98%	80%
Model B	95%	95%

Slide Credit: cs329s, stanford

Slice-based evaluation: example

- Majority group: 90%
- Minority group: 10%

Coarse-grained evaluation can hide:
- model biases
- potential for improvement

	Majority accuracy	**Minority accuracy**	**Overall accuracy**
Model A	98%	80%	96.2%
Model B	95%	95%	95%

Slide Credit: cs329s, stanford

Simpson’s paradox

- Models A and B to predict whether a customer will buy your product
- A performs better than B overall
- B performs better than A on both female & male customers

Slide Credit: cs329s, stanford

Simpson’s paradox

	Treatment 1	Treatment 2
Group A	**93% (81/87)**	87% (234/270)
Group B	**73% (192/263)**	69% (55/80)
Overall	78% (273/350)	**83% (289/350)**

Slide Credit: cs329s, stanford

Numbers from a kidney stone treatment study. (Charig et al., 1986)

Simpson's paradox: Berkeley graduate admission '73

	All		Men		Women	
	Applicants	Admitted	Applicants	Admitted	Applicants	Admitted
Total	12,763	41%	8442	**44%**	4321	35%

Bias against women in the process, or is there?

Slide Credit: cs329s, stanford

Sex Bias in Graduate Admissions: Data from Berkeley (Bickel et al., 1975)

Simpson's paradox: Berkeley graduate admission '73

Department	All		Men		Women	
	Applicants	Admitted	Applicants	Admitted	Applicants	Admitted
A	933	64%	***825***	62%	108	**82%**
B	585	63%	***560***	63%	25	**68%**
C	918	35%	325	**37%**	***593***	34%
D	792	34%	417	33%	375	**35%**
E	584	25%	191	**28%**	***393***	24%
F	714	6%	373	6%	341	**7%**

⚠️ Aggregation can conceal and contradict actual situation ⚠️

Slide Credit: cs329s, stanford

Sex Bias in Graduate Admissions: Data from Berkeley (Bickel et al., 1975)

Slice-based evaluation

- Evaluate your model on different slices
  - E.g. when working with website traffic data, slice data among:
    - gender
    - mobile vs. desktop
    - browser
    - location
- Check for consistency over time
  - E.g. evaluate your model on data slices from each day

Slide Credit: cs329s, stanford

Slice-based evaluation

- Improve model's performance both overall and on critical data
- Help avoid biases
- Even when you don't think slices matter, slicing can:
  - give you confidence on your model (to convince your boss)
  - might reveal non-ML problems

Slide Credit: cs329s, stanford

How to identify slices?

- Heuristics
  - Might require subject matter expertise
- Error analysis
  - Patterns among misclassified samples
- Slice finder
  - Exhaustive/beam search
  - Clustering
  - Decision tree

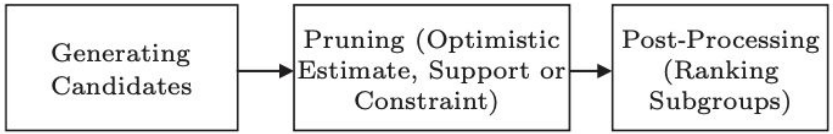


Fig.1. Methodology for subgroup discovery.

Slide Credit: cs329s, stanford

Slice finder: Automated data slicing for model validation (Chung et al., 2019)

Subgroup Discovery Algorithms: A Survey and Empirical Evaluation (Sumyea Helal, 2016)

How to identify slices?

Will go into details next lecture!

- Heuristics
  - Might require subject matter expertise
- Error analysis
  - Patterns among misclassified samples
- Slice finder
  - Exhaustive/beam search
  - Clustering
  - Decision tree

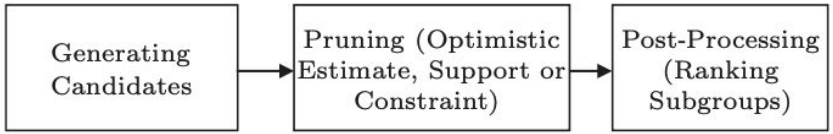


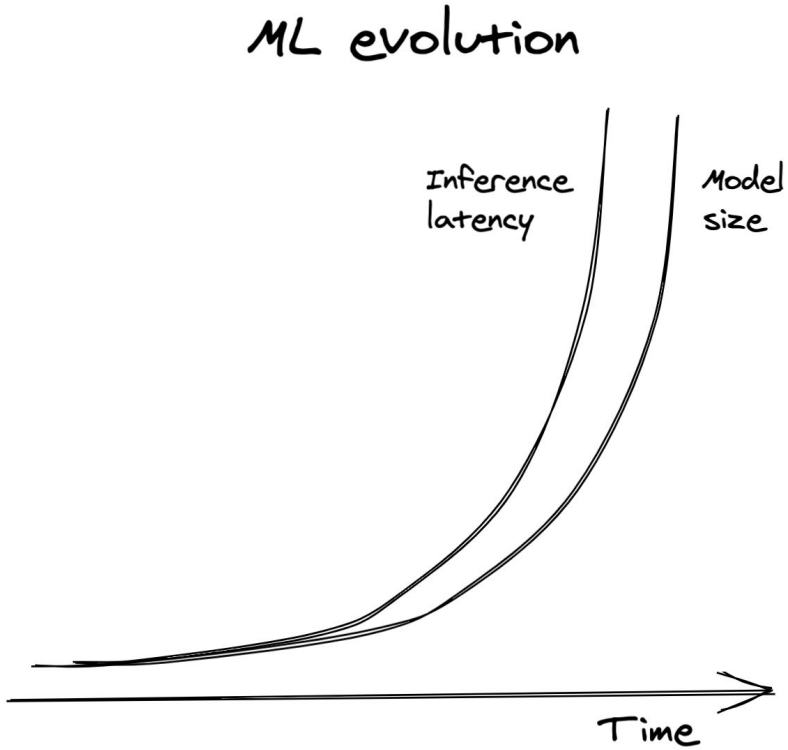
Fig.1. Methodology for subgroup discovery.

Slide Credit: cs329s, stanford

Slice finder: Automated data slicing for model validation (Chung et al., 2019)

Subgroup Discovery Algorithms: A Survey and Empirical Evaluation (Sumyea Helal, 2016)

Model Compression



Slide Credit: cs329s, stanford

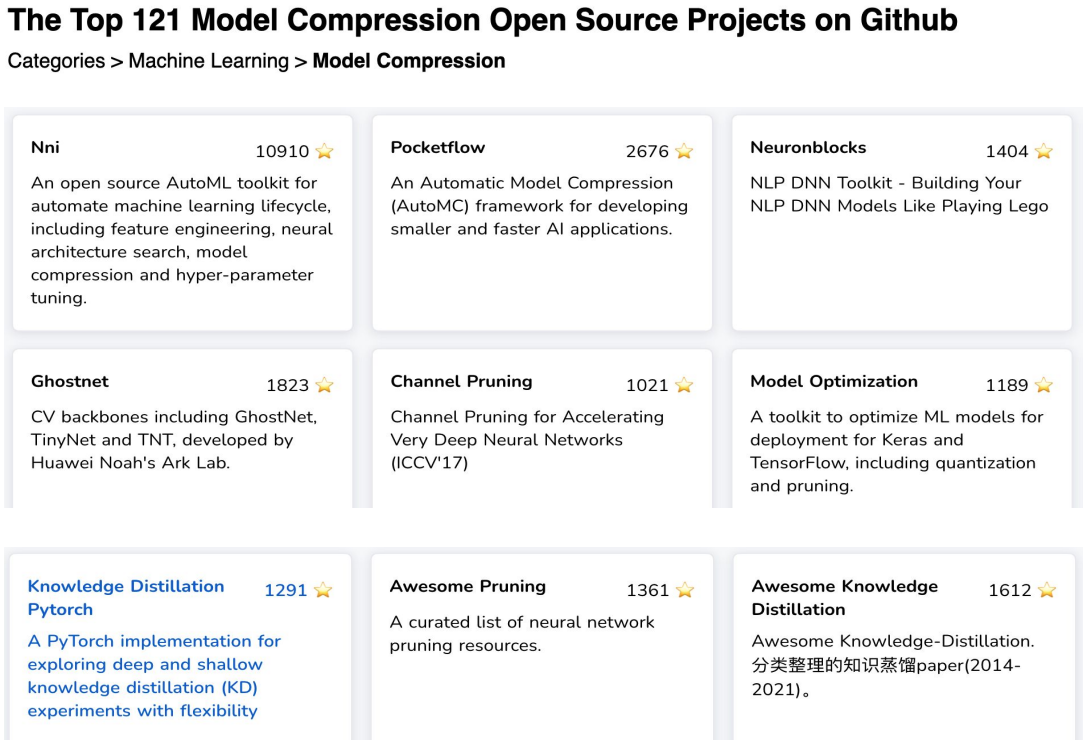
Bigger, better, slower

Model compression

1. Quantization
2. Knowledge distillation
3. Pruning
4. Low-ranked factorization

Slide Credit: cs329s, stanford

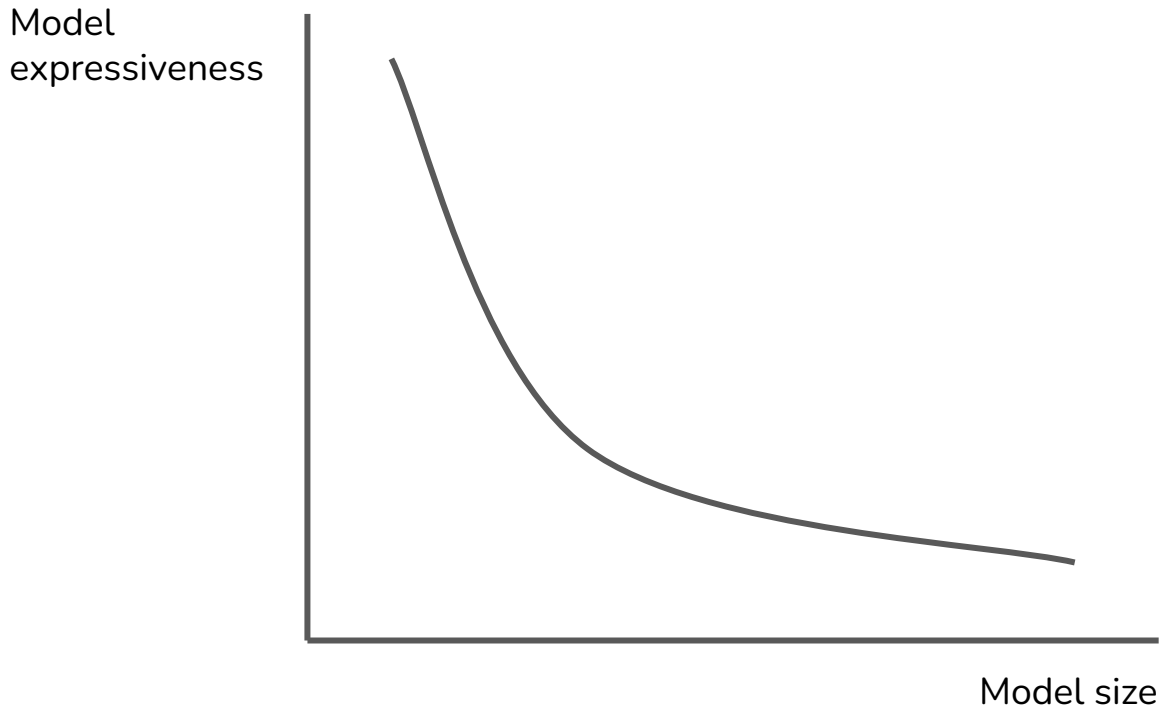
Model compression: active research/development



Slide Credit: cs329s, stanford

https://opensourcelibs.com/libs/model-compression | https://awesomeopensource.com/projects/model-compression

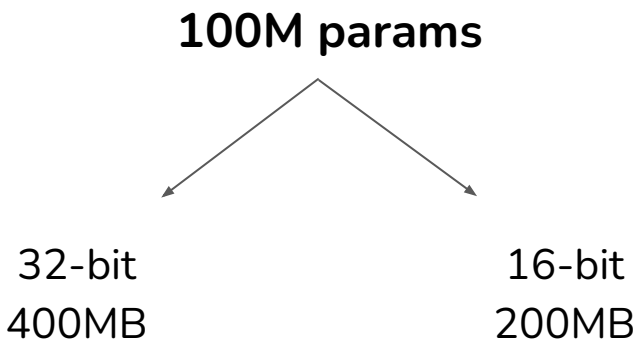
No free lunch!



Slide Credit: cs329s, stanford

Model compression: quantization

- Reduces the size of a model by using fewer bits to represent parameter values
  - E.g. half-precision (16-bit) or integer (8-bit) instead of full-precision (32-bit)



Slide Credit: cs329s, stanford

Model compression: quantization

- Reduces the size of a model by using fewer bits to represent parameter values
  - E.g. half-precision (16-bit) or integer (8-bit) instead of full-precision (32-bit)
  - 1-bit representation: BinaryConnect, Xnor-Net

Exclusive: Apple acquires Xnor.ai, edge AI spin-out from Paul Allen's AI2, for price in $200M range

BY ALAN BOYLE, TAYLOR SOPER & TODD BISHOP on January 15, 2020 at 10:44 am

Slide Credit: cs329s, stanford

Model compression: quantization

Pros	
1. Reduce memory footprint 2. Increase computation speed a. Bigger batch size b. Computation on 16 bits is faster than on 32 bits	

BFloat16: The secret to high performance on Cloud TPUs

Slide Credit: cs329s, stanford

Model compression: quantization

Pros	Cons
1. Reduce memory footprint 2. Increase computation speed a. Bigger batch size b. Computation on 16 bits is faster than on 32 bits	1. Smaller range of values 2. Values rounded to 0 Need efficient rounding/scaling techniques

Slide Credit: cs329s, stanford

Model compression: quantization

- Post-training quantization

```
torch.quantization.convert(model, inplace=True)
```

- Quantization-aware training

Slide Credit: cs329s, stanford

Model compression: knowledge distillation

- Train a small model (“student”) to mimic the results of a larger model (“teacher”)
  - Teacher & student can be trained at the same time.
  - E.g. DistillBERT, reduces size of BERT by 40%, and increases inference speed by 60%, while retaining 97% language understanding.

Slide Credit: cs329s, stanford

DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter (Sanh et al., 2019)

Model compression: knowledge distillation

- Train a small model (“student”) to mimic the results of a larger model (“teacher”)
- Pros:
  - Fast to train student network if teacher is pre-trained.
  - Teacher and student can be completely different architectures.
- Cons:
  - If teacher is not pre-trained, may need more data & time to first train teacher.
  - Sensitive to applications and model architectures.

Slide Credit: cs329s, stanford

Model compression: pruning

- Originally used for decision trees to remove uncritical sections
- Neural networks: reducing over-parameterization

Slide Credit: cs329s, stanford

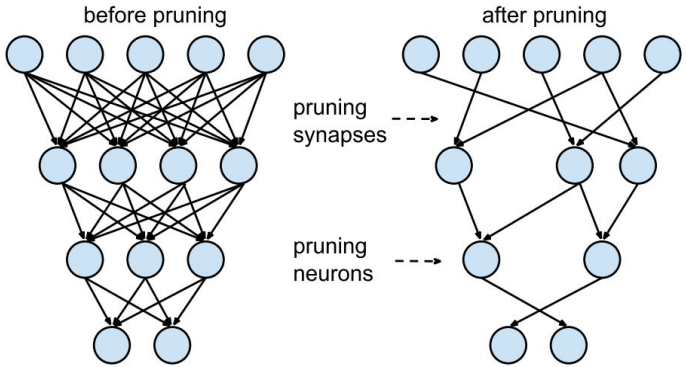
Model compression: pruning methods

1. Remove nodes
    a. Changing architectures & reducing number of params

Slide Credit: cs329s, stanford

Model compression: pruning methods

1. Remove nodes
2. Find least useful params & set to 0
   a. Number of params remains the same
   b. Reducing number of non-zero params

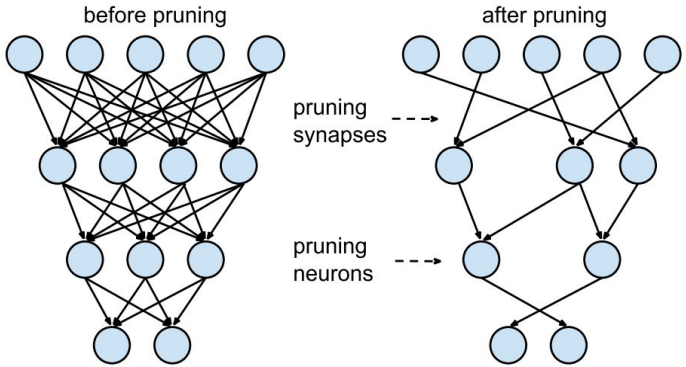


Slide Credit: cs329s, stanford

Learning both Weights and Connections for Efficient Neural Networks (Han et al., 2015)

Model compression: pruning methods

1. Remove nodes
2. Find least useful params ??? & set to 0
   a. Number of params remains the same
   b. Reducing number of non-zero params



Slide Credit: cs329s, stanford

Learning both Weights and Connections for Efficient Neural Networks (Han et al., 2015)

Model compression: pruning methods

1. Remove nodes
2. Find least useful params & set to 0
   a. Number of params remains the same
   b. Reducing number of non-zero params

Makes models more sparse
- lower memory footprint
- increased inference speed

Slide Credit: cs329s, stanford

The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks (Frankle et al., ICLR 2019)

Model compression: pruning methods

1. Remove nodes
2. Find least useful params & set to 0
   a. Number of params remains the same
   b. Reducing number of non-zero params

Can be used for architecture search

Slide Credit: cs329s, stanford

Data distribution shifts

- Source distribution: data the model is trained on
- Target distribution: data the model runs inference on

Slide Credit: cs329s, stanford

Supervised learning: P(X, Y)

1. $P(X, Y) = P(Y|X)P(X)$
2. $P(X, Y) = P(X|Y)P(Y)$

Slide Credit: cs329s, stanford

Types of data distribution shifts

Type	Meaning	Decomposition
Covariate shift	• P(X) changes • P(Y|X) remains the same	P(X, Y) = P(Y|X)P(X)
Label shift	• P(Y) changes • P(X|Y) remains the same	P(X, Y) = P(X|Y)P(Y)
Concept drift	• P(X) remains the same • P(Y|X) changes	P(X, Y) = P(Y|X)P(X)

Slide Credit: cs329s, stanford

Covariate shift

- P(X) changes
- P(Y|X) remains the same

- Statistics: a covariate is an independent variable that can influence the outcome of a given statistical trial.
- Supervised ML: input features are covariates

Slide Credit: cs329s, stanford

Covariate shift

- P(X) changes
- P(Y|X) remains the same

- Statistics: a covariate is an independent variable that can influence the outcome of a given statistical trial.
- Supervised ML: input features are covariates
- Input distribution changes, but for *a given input*, output is the same

Slide Credit: cs329s, stanford

Covariate shift: example

- P(X) changes
- P(Y|X) remains the same

- Predicts P(cancer | patient)
- P(age > 40): training > production
- P(cancer | age > 40): training = production

Slide Credit: cs329s, stanford

Covariate shift: causes (training)

- Data collection
  - E.g. women >40 are encouraged by doctors to get checkups
  - Closely related to sampling biases
- Training techniques
  - E.g. oversampling of rare classes
- Learning process
  - E.g. active learning

- P(X) changes
- P(Y|X) remains the same

- Predicts P(cancer | patient)
- P(age > 40):
  - training > production
- P(cancer | age > 40):
  - training = production

Slide Credit: cs329s, stanford

Covariate shift: causes (prod)

- P(X) changes
- P(Y|X) remains the same

Changes in environments

- Ex 1: P(convert to paid user | free user)
  - New marketing campaign attracting users from with higher income
    - P(high income) increases
    - P(convert to paid user | high level) remains the same

Slide Credit: cs329s, stanford

Covariate shift: causes (prod)

- P(X) changes
- P(Y|X) remains the same

Changes in environments

- Ex 2: P(Covid | coughing sound)
  - Training data from clinics, production data from phone recordings
    - P(coughing sound) changes
    - P(Covid | coughing sound) remains the same

Slide Credit: cs329s, stanford

Covariate shift

- Research: if knowing in advance how the production data will differ from training data, use importance weighting
- Production: unlikely to know how a distribution will change in advance

Slide Credit: cs329s, stanford

Label shift

- P(Y) changes
- P(X|Y) remains the same

- Output distribution changes but for *a given output*, input distribution stays the same.

Slide Credit: cs329s, stanford

Label shift & covariate shift

- Predicts P(cancer | patient)
- P(age > 40): training > production
- P(cancer | age > 40): training = production
- P(cancer): training > production
- P(age > 40 | cancer): training = prediction

- P(X) changes
- P(Y|X) remains the same

- P(Y) changes
- P(X|Y) remains the same

P(X) change often leads to P(Y) change, so covariate shift often means label shift

Slide Credit: cs329s, stanford

Label shift & covariate shift

- Predicts P(cancer | patient)
- New preventive drug: reducing P(cancer | patient) for all patients
- P(age > 40): training > production
- P(cancer | age > 40): training > production
- P(cancer): training > production
- P(age > 40 | cancer): training = prediction

- P(X) changes
- ~~P(Y|X) remains the same~~

- P(Y) changes
- P(X|Y) remains the same

Not all label shifts are covariate shifts!

Slide Credit: cs329s, stanford

Concept Drift

- P(X) remains the same
- P(Y|X) changes

- Same input, expecting different output
- P(houses in SF) remains the same
- Covid causes people to leave SF, housing prices drop
  - P($5M | houses in SF)
    - Pre-covid: high
    - During-covid: low

Slide Credit: cs329s, stanford

Concept Drift

- P(X) remains the same
- P(Y|X) changes

- Concept drifts can be cyclic & seasonal
  - Ride sharing demands high during rush hours, low otherwise
  - Flight ticket prices high during holidays, low otherwise

Slide Credit: cs329s, stanford

General data changes

- Feature change
  - A feature is added/removed/updated

Slide Credit: cs329s, stanford

General data changes

- Feature change
  - A feature is added/removed/updated
- Label schema change
  - Original: *{“POSITIVE”: 0, “NEGATIVE”: 1}*
  - New: *{“POSITIVE”: 0, “NEGATIVE”: 1, “NEUTRAL”: 2}*

Slide Credit: cs329s, stanford

Detecting data distribution shifts

How to determine that two distributions are different?

Slide Credit: cs329s, stanford

Detecting data distribution shifts

How to determine that two distributions are different?

1. Compare statistics: mean, median, variance, quantiles, skewness, kurtosis, ...
    - Compute these stats during training and compare these stats in production

Slide Credit: cs329s, stanford

Detecting data distribution shifts

How to determine that two distributions are different?

1. Compare statistics: mean, median, variance, quantiles, skewness, kurtosis, ...
    - Not universal: only useful for distributions where these statistics are meaningful

Slide Credit: cs329s, stanford

Detecting data distribution shifts

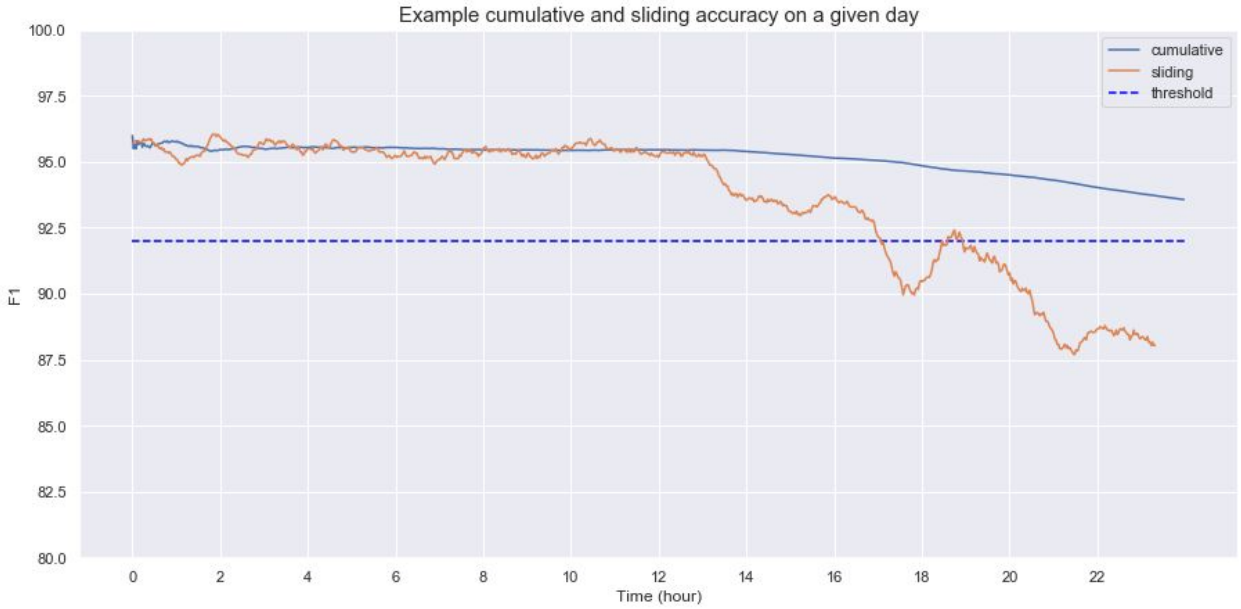
How to determine that two distributions are different?

1. Compare statistics: mean, median, variance, quantiles, skewness, kurtosis, ...
    - Not universal: only useful for distributions where these statistics are meaningful
    - Inconclusive: if statistics differ, distributions differ. If statistics are the same, distributions can still differ.

Slide Credit: cs329s, stanford

Cumulative vs. sliding metrics

- Sliding: reset at each new time window



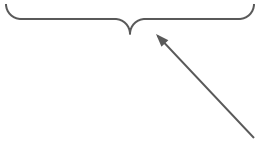
Slide Credit: cs329s, stanford

This image is based on an example from MadeWithML (Goku Mohandas).

Detecting data distribution shifts

How to determine that two distributions are different?

1. Compare statistics: mean, median, variance, quantiles, skewness, kurtosis, ...
2. Two-sample hypothesis test
   - Determine whether the difference between two populations is statistically significant
   - If yes, likely from two distinct distributions



E.g.
1. Data from yesterday
2. Data from today

Slide Credit: cs329s, stanford

Two-sample test: KS test (Kolmogorov–Smirnov)

- Pros
  - Doesn’t require any parameters of the underlying distribution
  - Doesn’t make assumptions about distribution
- Cons
  - Only works with one-dimensional data

- Useful for prediction & label distributions
- Not so useful for features

Slide Credit: cs329s, stanford

Two-sample test

Drift Detection

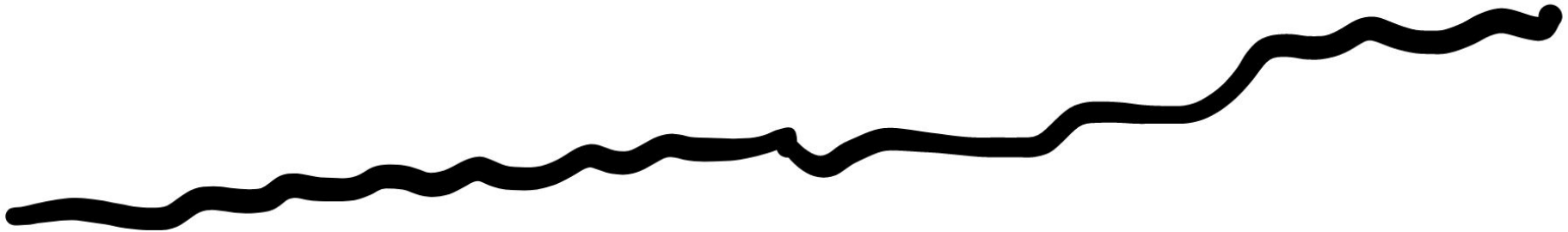
Detector	Tabular	Image	Time Series	Text	Categorical Features	Online	Feature Level
Kolmogorov-Smirnov	✓	✓		✓	✓		✓
Maximum Mean Discrepancy	✓	✓		✓	✓	✓	
Learned Kernel MMD	✓	✓		✓	✓		
Least-Squares Density Difference	✓	✓		✓	✓	✓	
Chi-Squared	✓				✓		✓
Mixed-type tabular data	✓				✓		✓
Classifier	✓	✓	✓	✓	✓		
Spot-the-diff	✓	✓	✓	✓	✓		✓
Classifier Uncertainty	✓	✓	✓	✓	✓		
Regressor Uncertainty	✓	✓	✓	✓	✓		

alibi-detect (OS)

Most tests work better on low-dim data, so dim reduction is recommended beforehand!

Not all shifts are equal

- Sudden shifts vs. gradual shifts
  - Sudden shifts are easier to detect than gradual shifts



Slide Credit: cs329s, stanford

Not all shifts are equal

- Sudden shifts vs. gradual shifts
- Spatial shifts vs. temporal shifts

- New device (e.g. mobile vs. desktop)
- New users (e.g. new country)

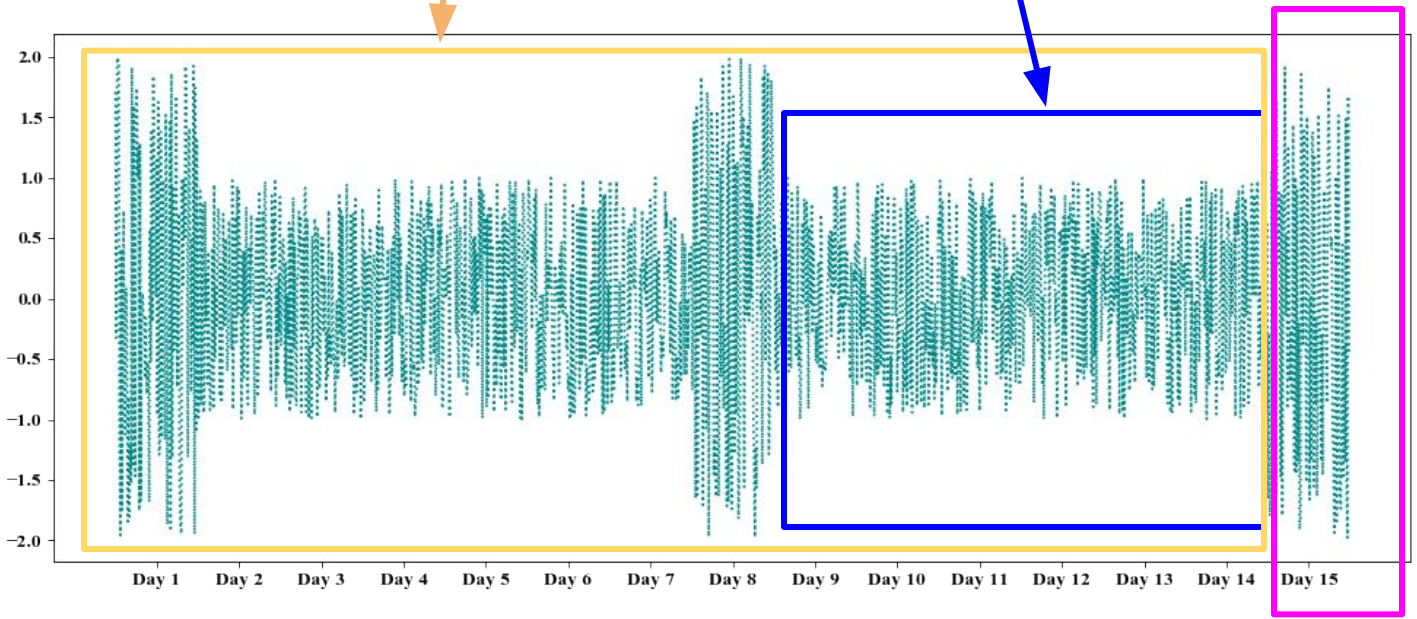
E.g. same users, same device, but behaviors change over time

Slide Credit: cs329s, stanford

Temporal shifts: time window scale matters

Source distribution: unlikely a shift

Source distribution: likely a shift

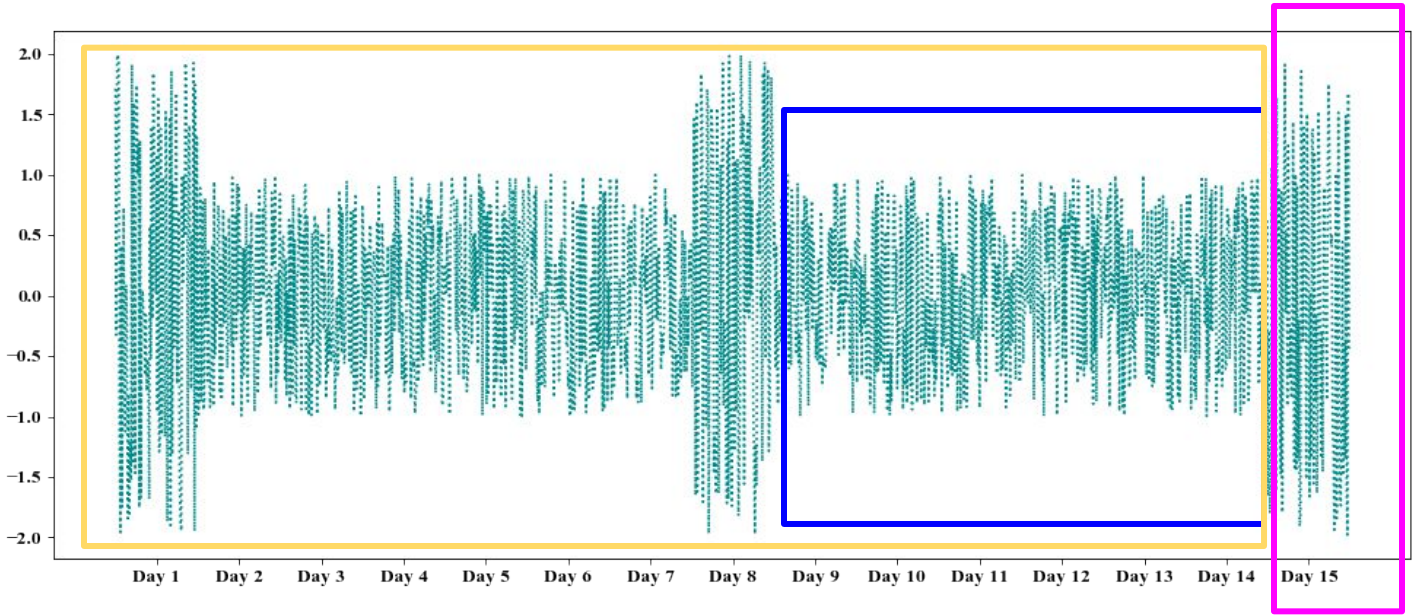


Target distribution

Slide Credit: cs329s, stanford

Temporal shifts: time window scale matters

Difficulty is compounded
by seasonal variation



Slide Credit: cs329s, stanford

Temporal shifts: time window scale matters

- Too short window: false alarms of shifts
- Too long window: takes long to detect shifts

- Granularity level: hourly, daily

Slide Credit: cs329s, stanford

Temporal shifts: time window scale matters

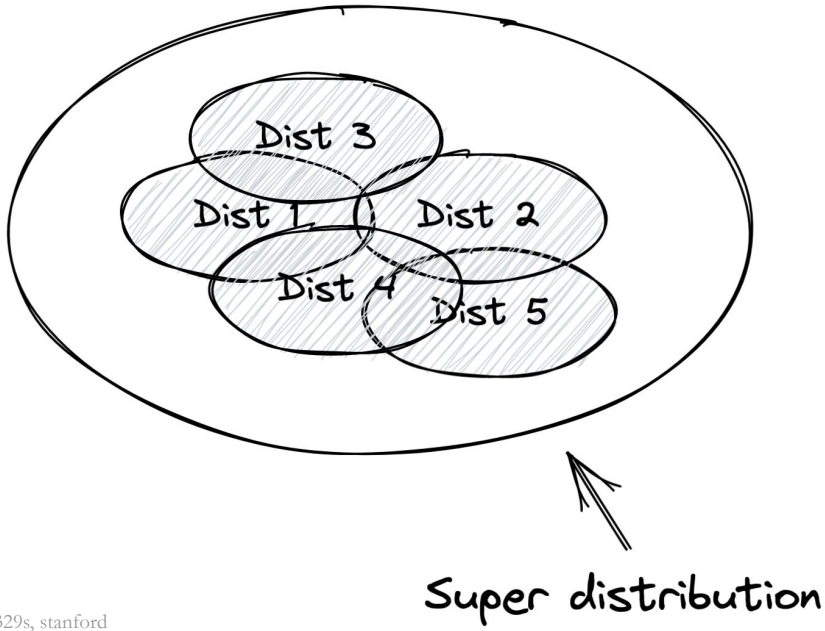
- Too short window: false alarms of shifts
- Too long window: takes long to detect shifts

- Granularity level: hourly, daily
- Merge shorter time scale windows -> larger time scale window
- RCA: automatically analyze various window sizes

Slide Credit: cs329s, stanford

Addressing data distribution shifts

1. Train model using a massive dataset



Slide Credit: cs329s, stanford

Addressing data distribution shifts

1. Train model using a massive dataset
2. Retrain model with new data from new distribution
    - Mode
        - Train from scratch
        - Fine-tune

Slide Credit: cs329s, stanford

Addressing data distribution shifts

1. Train model using a massive dataset
2. Retrain model with new data from new distribution
    - Mode
    - Data
        - Use data from when data started to shift
        - Use data from the last X days/weeks/months
        - Use data form the last fine-tuning point

Need to figure out not just when to retrain models, but also how and what data

Slide Credit: cs329s, stanford

Monitoring & Observability

Monitoring vs. observability

- Monitoring: tracking, measuring, and logging different metrics that can help us determine when something goes wrong
- Observability: setting up our system in a way that gives us visibility into our system to investigate what went wrong

Slide Credit: cs329s, stanford

Monitoring vs. observability

- Monitoring: tracking, measuring, and logging different metrics that can help us determine when something goes wrong
- Observability: setting up our system in a way that gives us visibility into our system to investigate what went wrong

Instrumentation

- adding timers to your functions
- counting NaNs in your features
- logging unusual events e.g. very long inputs
- ...

Slide Credit: cs329s, stanford

Monitoring vs. observability

- Monitoring: tracking, measuring, and logging different metrics that can help us determine when something goes wrong
- Observability: setting up our system in a way that gives us visibility into our system to investigate what went wrong

Observability is part of monitoring

Slide Credit: cs329s, stanford

Monitoring is all about metrics

- Operational metrics
- ML-specific metrics

Slide Credit: cs329s, stanford

Operational metrics

- Latency
- Throughput
- Requests / minute/hour/day
- % requests that return with a 2XX code
- CPU/GPU utilization
- Memory utilization
- **Availability**
- etc.

Slide Credit: cs329s, stanford

Operational metrics

- Latency
- Throughput
- Requests / minute/hour/day
- % requests that return with a 2XX code
- CPU/GPU utilization
- Memory utilization
- **Availability**
- etc.

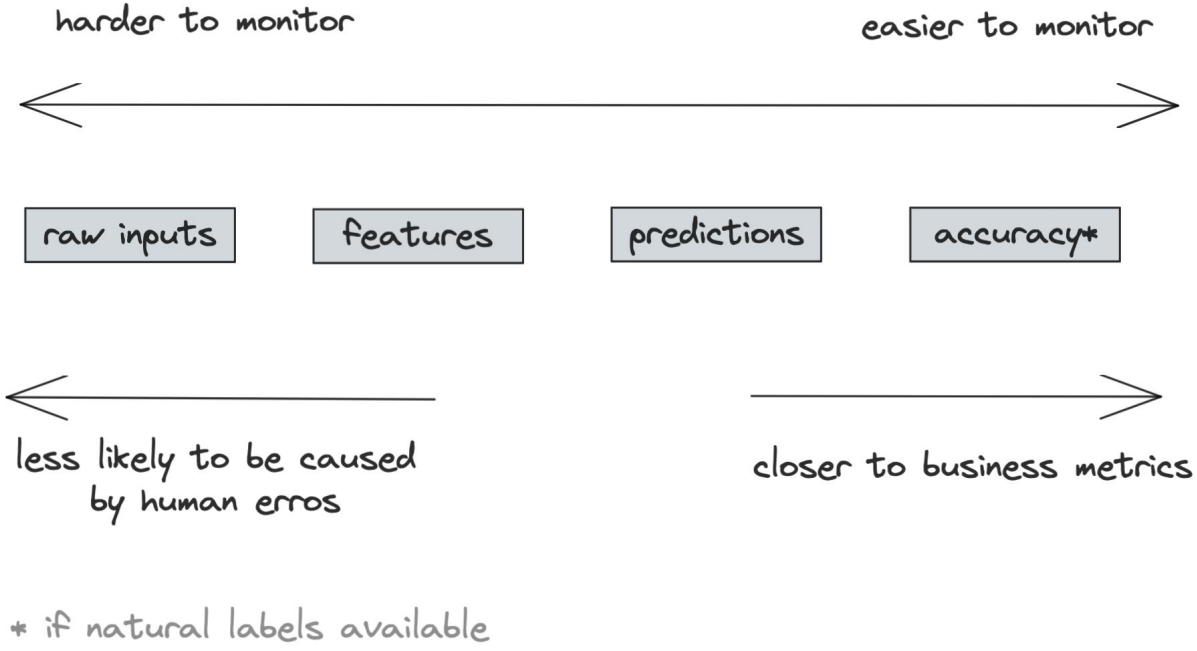
SLA example

- Up means:
  - median latency <200ms
  - 99th percentile <2s
- 99.99% uptime (four-nines)

SLA for ML?

Slide Credit: cs329s, stanford

ML metrics: what to monitor



Slide Credit: cs329s, stanford

Monitoring #1: accuracy-related metrics

- Most direct way to monitor a model's performance
  - Can only do as fast as when feedback is available

Slide Credit: cs329s, stanford

Monitoring #1: accuracy-related metrics

- Most direct way to monitor a model's performance
- Collect as much feedback as possible
- Example: YouTube video recommendations
  - Click through rate
  - Duration watched
  - Completion rate
  - Take rate

Slide Credit: cs329s, stanford

Monitoring #2: predictions

- Predictions are low-dim: easy to visualize, compute stats, and do two-sample tests
- Changes in prediction dist. generally mean changes in input dist.

Slide Credit: cs329s, stanford

Monitoring #2: predictions

- Predictions are low-dim: easy to visualize, compute stats, and do two-sample tests
- Changes in prediction dist. generally mean changes in input dist.
- Monitor odd things in predictions
    - E.g. if predictions are all False in the last 10 mins

Slide Credit: cs329s, stanford

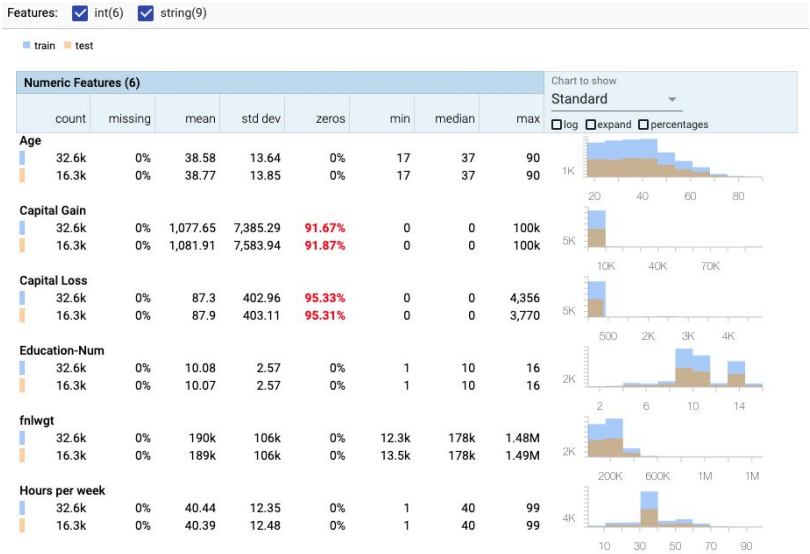
Monitoring #3: features

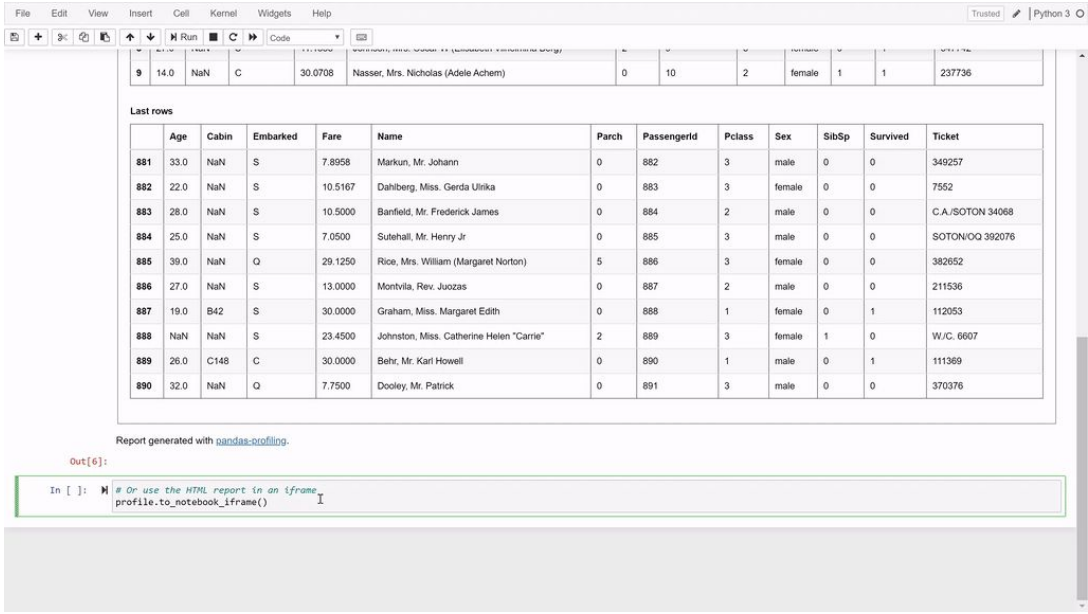
- Most monitoring tools focus on monitoring features
- Feature schema expectations
  - Generated from the source distribution
  - If violated in production, possibly something is wrong
- Example expectations
  - Common sense: e.g. “the” is most common word in English
  - min, max, or median values of a feature are in [a, b]
  - All values of a feature satisfy a regex
  - Categorical data belongs to a predefined set
  - FEATURE_1 > FEATURE_B

Slide Credit: cs329s, stanford

Generate expectations with profiling & visualization

- Examining data & collecting:
  - statistics
  - informative summaries
- [pandas-profiling]
- [facets]





Monitoring #3: features

- Feature schema expectations

Table shape

- `expect_column_to_exist`
- `expect_table_columns_to_match_ordered_list`
- `expect_table_columns_to_match_set`
- `expect_table_row_count_to_be_between`
- `expect_table_row_count_to_equal`
- `expect_table_row_count_to_equal_other_table`

Missing values, unique values, and types

- `expect_column_values_to_be_unique`
- `expect_column_values_to_not_be_null`
- `expect_column_values_to_be_null`
- `expect_column_values_to_be_of_type`
- `expect_column_values_to_be_in_type_list`

```
expect_column_values_to_be_between(
    column="room_temp",
    min_value=60,
    max_value=75,
    mostly=.95
)
```


"Values in this column should be between 60 and 75, at least 95% of the time."

"Warning: more than 5% of values fell outside the specified range of 60 to 75."

Slide Credit: cs329s, stanford

GitHub - great-expectations/great_expectations

Monitoring #3: features schema with pydantic

```python
from pydantic import BaseModel, ValidationError, validator


class UserModel(BaseModel):
    name: str
    username: str
    password1: str
    password2: str

    @validator('name')
    def name_must_contain_space(cls, v):
        if ' ' not in v:
            raise ValueError('must contain a space')
        return v.title()

    @validator('password2')
    def passwords_match(cls, v, values, **kwargs):
        if 'password1' in values and v != values['password1']:
            raise ValueError('passwords do not match')
        return v

    @validator('username')
    def username_alphanumeric(cls, v):
        assert v.isalnum(), 'must be alphanumeric'
        return v
```

```python
user = UserModel(
    name='samuel colvin',
    username='scolvin',
    password1='zxcvbn',
    password2='zxcvbn',
)
print(user)
#> name='Samuel Colvin' username='scolvin' password1='zxcvbn' password2='zxcvbn'

try:
    UserModel(
        name='samuel',
        username='scolvin',
        password1='zxcvbn',
        password2='zxcvbn2',
    )
except ValidationError as e:
    print(e)
    """
    2 validation errors for UserModel
    name
      must contain a space (type=value_error)
    password2
      passwords do not match (type=value_error)
    """
```


https://pydantic-docs.helpmanual.io/usage/validators/

Monitoring #3: features schema with TFX

```
# Generate training stats & schema
train_stats = tfdv.generate_statistics_from_dataframe(df)
schema = tfdv.infer_schema(statistics=train_stats)
```

```
schema
```

```
feature {
  name: "1"
  type: FLOAT
  presence {
    min_fraction: 1.0
    min_count: 1
  }
  shape {
    dim {
      size: 1
    }
  }
}
```

```
# Generate serving stats
serving_stats = tfdv.generate_statistics_from_dataframe(serving_df)
# Domain knowledge required
tfdv.get_feature(schema, "diabetesMed").skew_comparator.infinity_norm.threshold = 0.03
# Compare serving stats to training stats to detect skew
skew_anomalies = tfdv.validate_statistics(
    statistics=train_stats,
    schema=schema,
    serving_statistics=serving_stats)
```

Feature name	Anomaly short description	Anomaly long description
'payer_code'	High Linfty distance between current and previous	The Linfty distance between current and previous is 0.0342144 (up to six significant digits), above the threshold 0.03. The feature value with maximum difference is: MC
'diabetesMed'	High Linfty distance between training and serving	The Linfty distance between training and serving is 0.0325464 (up to six significant digits), above the threshold 0.03. The feature value with maximum difference is: No

Slide Credit: cs329s, stanford

How To Evaluate MLOps Tools (Hamel Husain, CS 329S Lecture 9, 2022)

Monitoring toolbox: alerts

- 3 components
  - Alert policy: condition for alert
  - Notification channels
  - Description
- Alert fatigue
  - How to send only meaningful alerts?

```
## Recommender model accuracy below 90%

${timestamp}: This alert originated from the service ${service-name}
```

Slide Credit: cs329s, stanford

Monitoring -> Continual Learning

- Monitoring is passive
  - Wait for a shift to happen to detect it
- Continual learning is active
  - Update your models to address shifts

Slide Credit: cs329s, stanford

Model’s performance degrades in production

- Data distribution shifts
  - Sudden
  - Cyclic
  - Gradual

Slide Credit: cs329s, stanford

From Monitoring to Continual Learning

- Monitoring: detect changing data distributions
- Continual learning: continually adapt models to changing data distributions

Slide Credit: cs329s, stanford

Continual learning

- Set up infrastructure such that models can continuously learn from new data in production
- Stateful training

Slide Credit: cs329s, stanford

Continual learning: use cases

- Rare events
  - Christmas/Black Friday/Prime Day shopping
  - Total Landscaping
- Continuous cold start (in-session adaptation)
  - New users
  - New devices
  - Users not logged in
  - Users rarely logged in

Slide Credit: cs329s, stanford

Continual learning is especially good for

- Natural labels: e.g. user click -> good prediction
- Short feedback loops
- Examples:
  - RecSys
  - Ranking
  - Ads CTR prediction
  - eDiscovery

Slide Credit: cs329s, stanford

How often to retrain your model is just a knob to turn

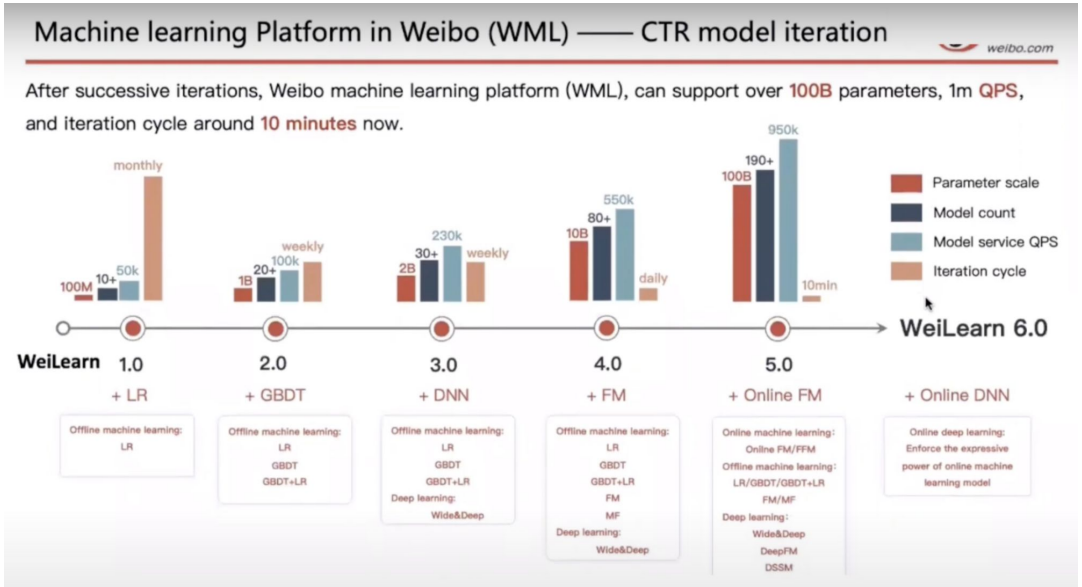
How frequently should you update your models?

- Very few companies actually update models with each incoming sample
  - Catastrophic forgetting
  - Can get unnecessarily expensive*
- Update models with micro-batches

Slide Credit: cs329s, stanford

Iteration cycle: minutes

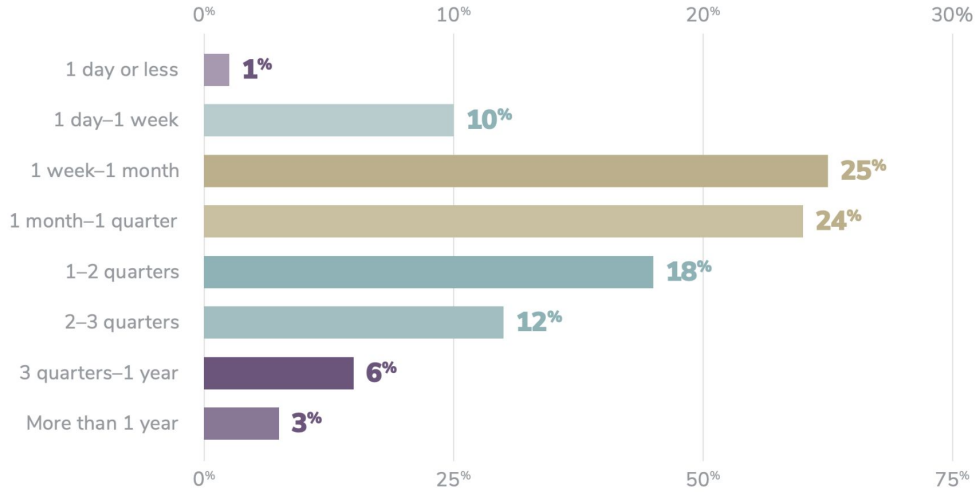
- Alibaba: Singles Day sale
- Weibo
- Tiktok
- SheIN



Slide Credit: cs329s, stanford

Iteration cycle: US

Only 11% of organizations can put a model into production within a week, and 64% take a month or longer



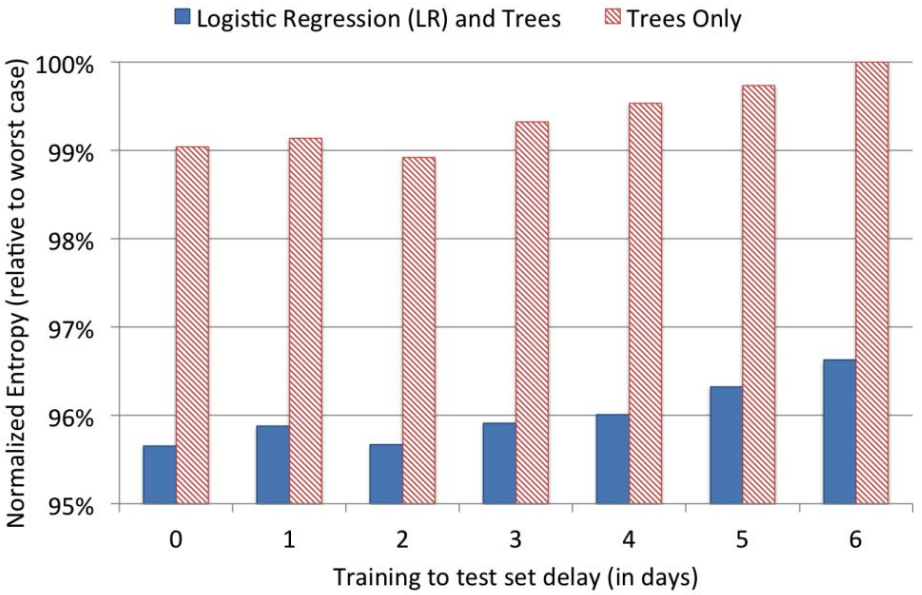
Slide Credit: cs329s, stanford

Algorithmia's State of Enterprise Machine Learning (2021)

Quantify the value of data freshness

1. How much model's performance changes if switch from retraining monthly to weekly to daily to hourly?
   a. FB: CTR loss can be reduced ~1%

      going from training weekly to daily



Slide Credit: cs329s, stanford

[Practical Lessons from Predicting Clicks on Ads at Facebook](#) (2014)

Quantify the value of data freshness

1. How much model's performance changes if switch from retraining monthly to weekly to daily to hourly?
2. How would retention change if you can do in-session adaptation?

Slide Credit: cs329s, stanford

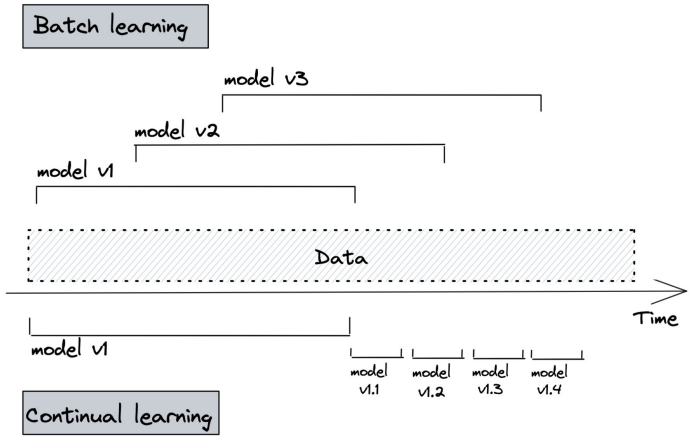
Quantify the value of data freshness

1. How much model's performance changes if switch from retraining monthly to weekly to daily to hourly?
2. How would retention change if you can do in-session adaptation?
3. Model iteration vs. data iteration

Slide Credit: cs329s, stanford

Quantify cloud bill savings

- Train model incrementally each day on fresh data
- Faster convergence → less compute needed



GRUBHUB™ Going from monthly training to daily training gives

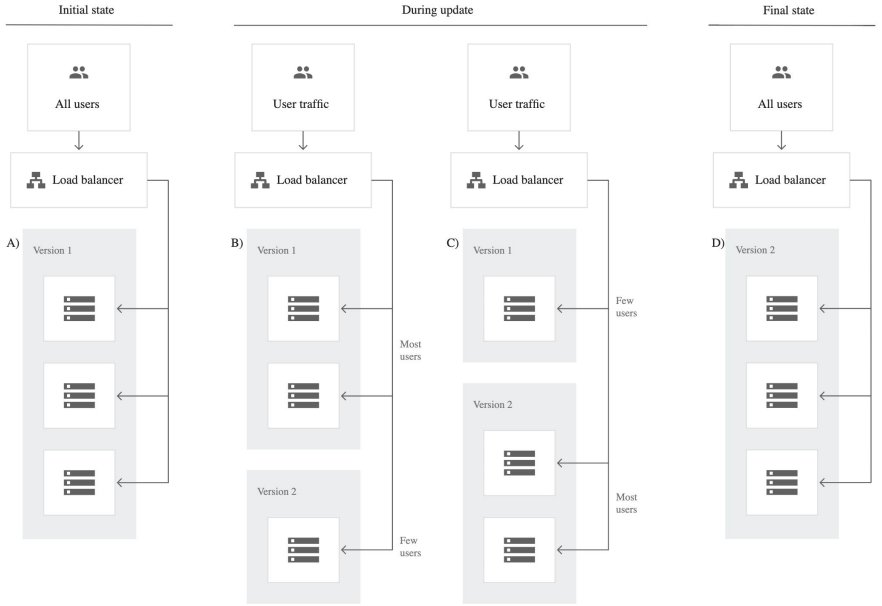
45x cost savings and **+20% metrics increase**

Online Learning for Recommendations at Grubhub (Alex Egg, 2021)

Test in Production

Canary testing

- New model alongside existing system
- Some traffic is routed to new model
- Slowly increase the traffic to new model
  - E.g. roll out to Vietnam first, then Asia, then rest of the world

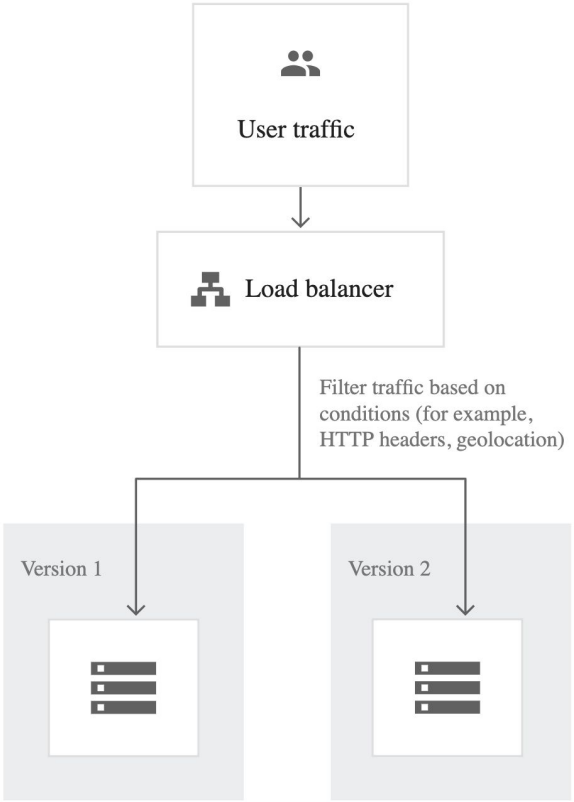


Slide Credit: cs329s, stanford

Application deployment and testing strategies (Google)

A/B testing

- New model alongside existing system
- A percentage of traffic is routed to new model based on routing rules
- Control target audience & monitor any statistically significant differences in user behavior
- Can have more than 2 versions

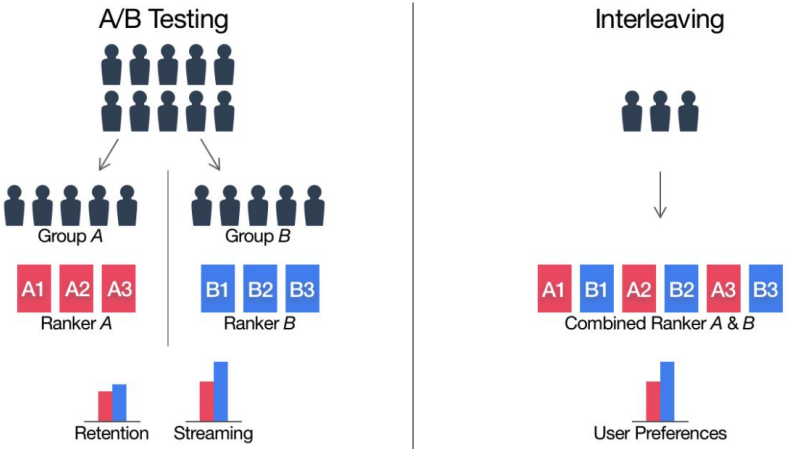


Slide Credit: cs329s, stanford

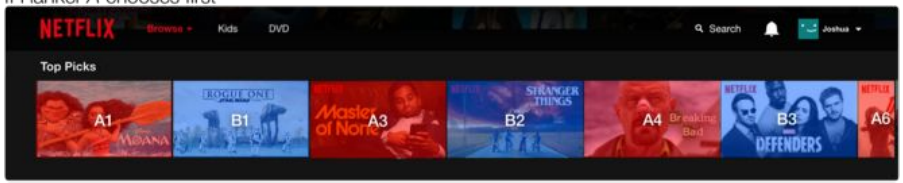
Application deployment and testing strategies (Google)

Interleaved experiments

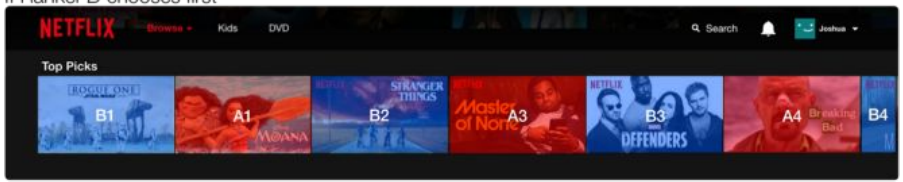
- Especially useful for ranking/recsys
- Take recommendations from both model A & B
- Mix them together and show them to users
- See which recommendations are clicked on



If Ranker *A* chooses first

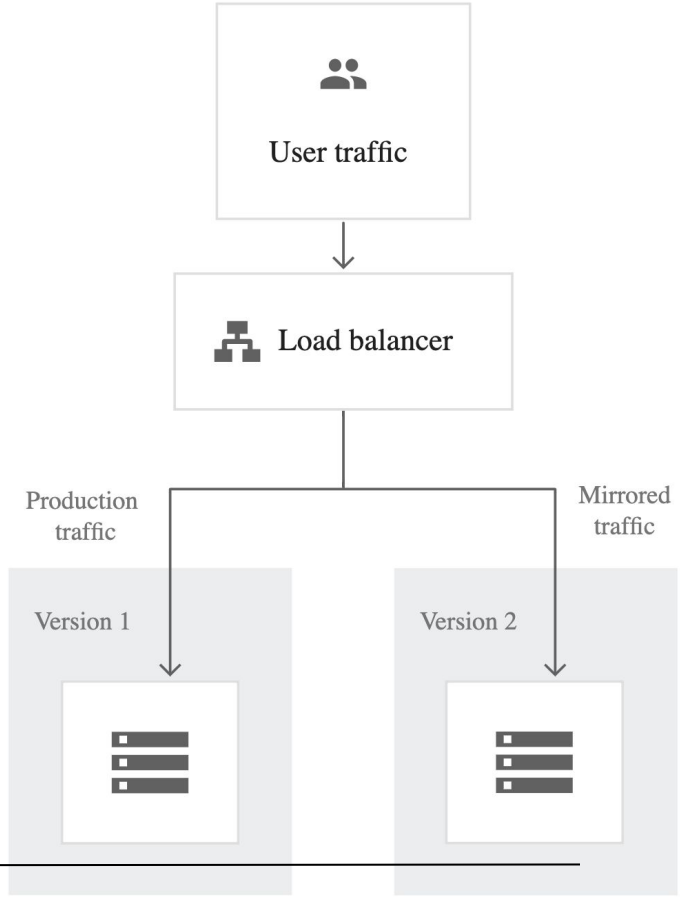


If Ranker *B* chooses first



Shadow testing

- New model in parallel with existing system
- New model's predictions are logged, but not show to users
- Switch to new model when results are satisfactory



Slide Credit: cs329s, stanford

[Application deployment and testing strategies](#) (Google)

“REPRODUCIBILITY IS LIKE BRUSHING YOUR TEETH. ONCE YOU LEARN IT, IT BECOMES A HABIT.”

What does it mean to be reproducible?

For all **models** and **algorithms** presented, check if you include:

- ❑ A clear description of the mathematical setting, algorithm, and/or model.
- ❑ A clear explanation of any assumptions.
- ❑ An analysis of the complexity (time, space, sample size) of any algorithm.

For any **theoretical claim**, check if you include:

- ❑ A clear statement of the claim.
- ❑ A complete proof of the claim.

For all **datasets** used, check if you include:

- ❑ The relevant statistics, such as number of examples.
- ❑ The details of train / validation / test splits.
- ❑ An explanation of any data that were excluded, and all pre-processing step.
- ❑ A link to a downloadable version of the dataset or simulation environment.
- ❑ For new data collected, a complete description of the data collection process, such as instructions to annotators and methods for quality control.

Slide Credit: cs329s, stanford

Source: Checklist by Joelle Pineau – Neurips Reproducibility Chair

For all shared **code** related to this work, check if you include:

- ❑ Specification of dependencies.
- ❑ Training code.
- ❑ Evaluation code.
- ❑ (Pre-)trained model(s).
- ❑ README file includes table of results accompanied by precise command to run to produce those results.

For all reported **experimental results**, check if you include:

- ❑ The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results.
- ❑ The exact number of training and evaluation runs.
- ❑ A clear definition of the specific measure or statistics used to report results.
- ❑ A description of results with central tendency (e.g. mean) & variation (e.g. error bars).
- ❑ The average runtime for each result, or estimated energy cost.
- ❑ A description of the computing infrastructure used.

Slide Credit: cs329s, stanford

Source: Checklist by Joelle Pineau – Neurips Reproducibility Chair

Making ML workflows reproducible, one step at a time

Before W&B

```
Layer (type)                 Output Shape              Param #     Connected to
==================================================================================================
conv2d_1_input (InputLayer)  (None, 299, 299, 3)  0
__________________________________________________________________________________________________
lambda_1 (Lambda)            (None, 299, 299, 3)  0           conv2d_1_input[0][0]
__________________________________________________________________________________________________
lambda_2 (Lambda)            (None, 299, 299, 3)  0           conv2d_1_input[0][0]
__________________________________________________________________________________________________
sequential_1 (Sequential)    (None, 10)            910922      lambda_1[0][0]
                                                               lambda_2[0][0]
__________________________________________________________________________________________________
activation_7 (Concatenate)   (None, 10)            0           sequential_1[1][0]
                                                               sequential_1[2][0]
==================================================================================================
Total params: 910,922
Trainable params: 910,922
Non-trainable params: 0
__________________________________________________________________________________________________
None
Found 49999 images belonging to 10 classes.
Found 8000 images belonging to 10 classes.
Epoch 1/50
39/39 [==============================] - 270s 7s/step - loss: 2.3153 - acc: 0.1178 - val_loss: 2.2428 - val_acc: 0.1589
Epoch 2/50
39/39 [==============================] - 263s 7s/step - loss: 2.2588 - acc: 0.1538 - val_loss: 2.1890 - val_acc: 0.2174
Epoch 3/50
39/39 [==============================] - 262s 7s/step - loss: 2.2060 - acc: 0.1827 - val_loss: 2.1767 - val_acc: 0.2096
Epoch 4/50
39/39 [==============================] - 263s 7s/step - loss: 2.1640 - acc: 0.2107 - val_loss: 2.1133 - val_acc: 0.2357
Epoch 5/50
39/39 [==============================] - 261s 7s/step - loss: 2.0827 - acc: 0.2432 - val_loss: 2.1499 - val_acc: 0.2344
Epoch 6/50
39/39 [==============================] - 262s 7s/step - loss: 2.0810 - acc: 0.2508 - val_loss: 2.0289 - val_acc: 0.2604
Epoch 7/50
39/39 [==============================] - 262s 7s/step - loss: 2.0575 - acc: 0.2602 - val_loss: 1.9912 - val_acc: 0.2865
Epoch 8/50
39/39 [==============================] - 261s 7s/step - loss: 2.0355 - acc: 0.2734 - val_loss: 2.0319 - val_acc: 0.2409
Epoch 9/50
39/39 [==============================] - 263s 7s/step - loss: 2.0254 - acc: 0.2829 - val_loss: 1.9364 - val_acc: 0.3307
Epoch 10/50
39/39 [==============================] - 262s 7s/step - loss: 2.0056 - acc: 0.2804 - val_loss: 1.9934 - val_acc: 0.2773
Epoch 11/50
39/39 [==============================] - 256s 7s/step - loss: 1.9678 - acc: 0.3048 - val_loss: 1.9048 - val_acc: 0.3352
Epoch 12/50
39/39 [==============================] - 260s 7s/step - loss: 1.9634 - acc: 0.3109 - val_loss: 1.8758 - val_acc: 0.3568
```

Table 3: Detection results on **PASCAL VOC 2007 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07+12": union set of VOC 2007 trainval and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2000. †: this number was reported in [2]; using the repository provided by this paper, this result is higher (68.1).

method	# proposals	data	mAP (%)
SS	2000	07	66.9†
SS	2000	07+12	70.0
RPN+VGG, unshared	300	07	68.5
RPN+VGG, shared	300	07	69.9
RPN+VGG, shared	300	07+12	**73.2**
RPN+VGG, shared	300	COCO+07+12	**78.8**

Table 4: Detection results on **PASCAL VOC 2012 test set**. The detector is Fast R-CNN and VGG-16. Training data: "07": VOC 2007 trainval, "07++12": union set of VOC 2007 trainval+test and VOC 2012 trainval. For RPN, the train-time proposals for Fast R-CNN are 2000. †: http://host.robots.ox.ac.uk:8080/anonymous/HZJTQA.html. ‡: http://host.robots.ox.ac.uk:8080/anonymous/YNPLXB.html. §: http://host.robots.ox.ac.uk:8080/anonymous/XEDH10.html.

method	# proposals	data	mAP (%)
SS	2000	12	65.7
SS	2000	07++12	68.4
RPN+VGG, shared†	300	12	67.0
RPN+VGG, shared‡	300	07++12	**70.4**
RPN+VGG, shared§	300	COCO+07++12	**75.9**

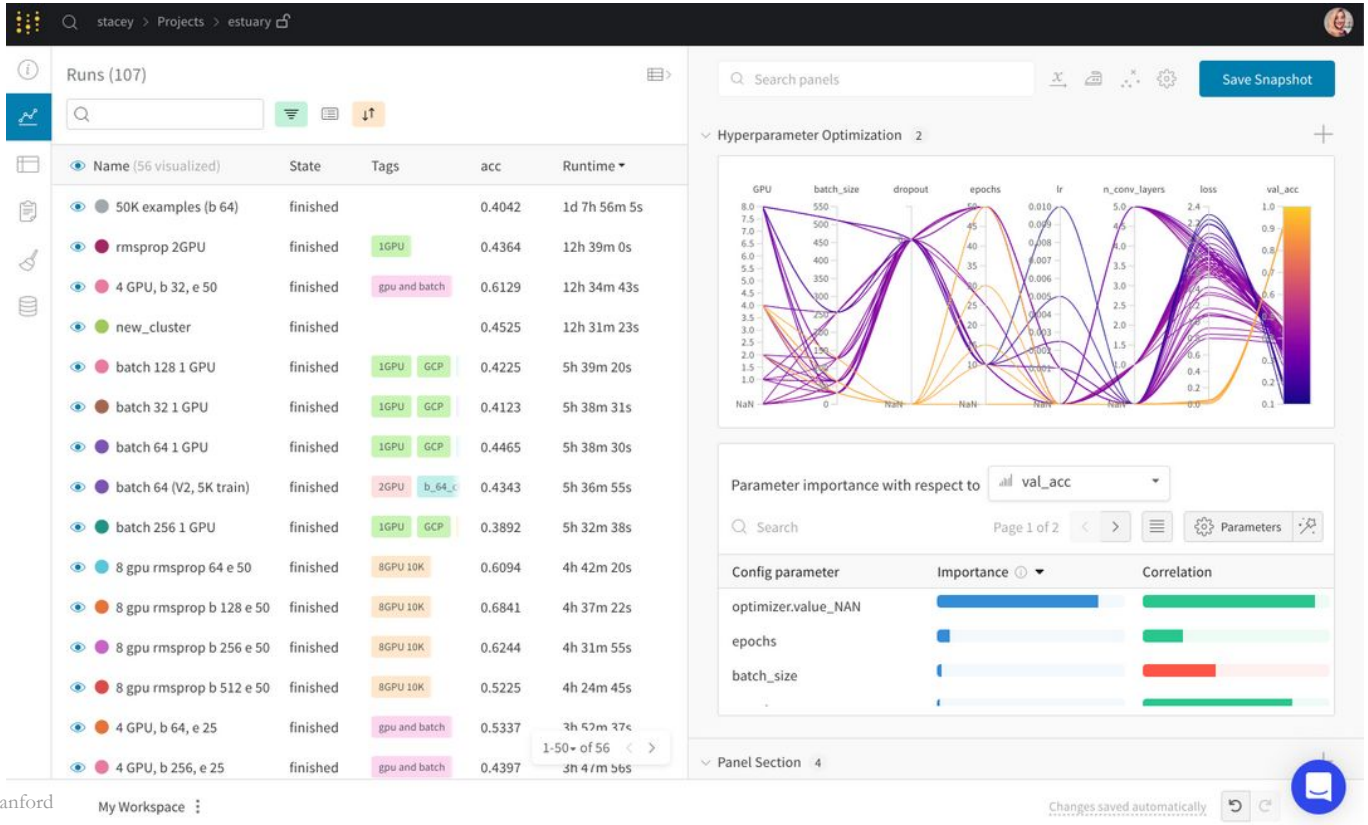
Table 5: **Timing** (ms) on a K40 GPU, except SS proposal is evaluated in a CPU. "Region-wise" includes NMS, pooling, fully-connected, and softmax layers. See our released code for the profiling of running time.

model	system	conv	proposal	region-wise	total	rate
VGG	SS + Fast R-CNN	146	1510	174	1830	0.5 fps
VGG	RPN + Fast R-CNN	141	**10**	47	**198**	**5 fps**
ZF	RPN + Fast R-CNN	31	**3**	25	**59**	**17 fps**

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Slide Credit: cs329s, stanford

Track experiments – After W&B



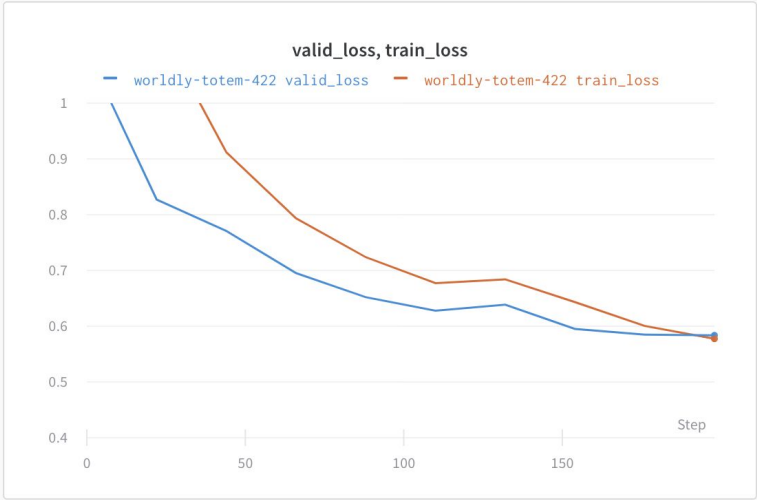
Slide Credit: cs329s, stanford

Live

bit.ly/demo-run

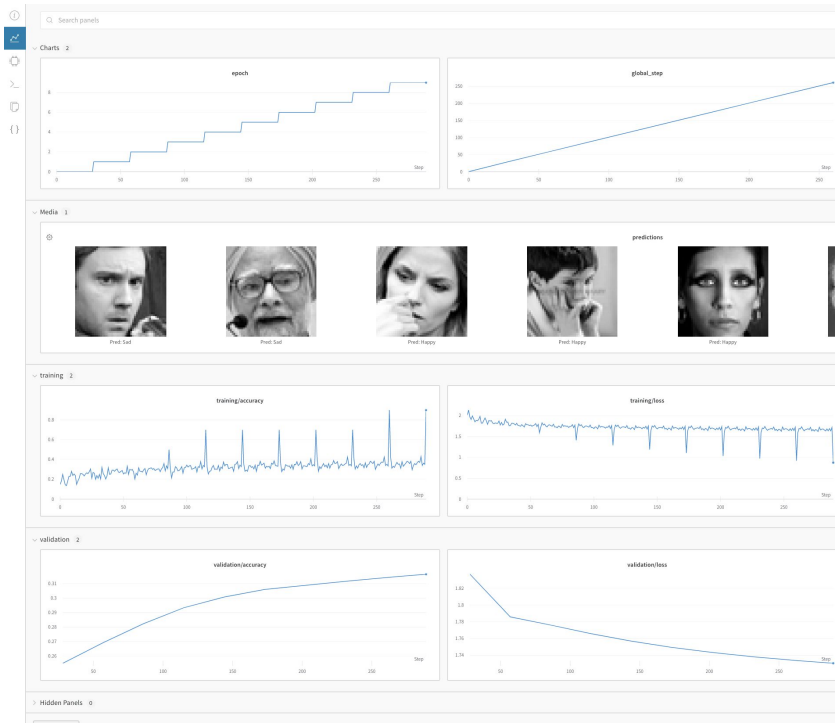
Reproduce a model

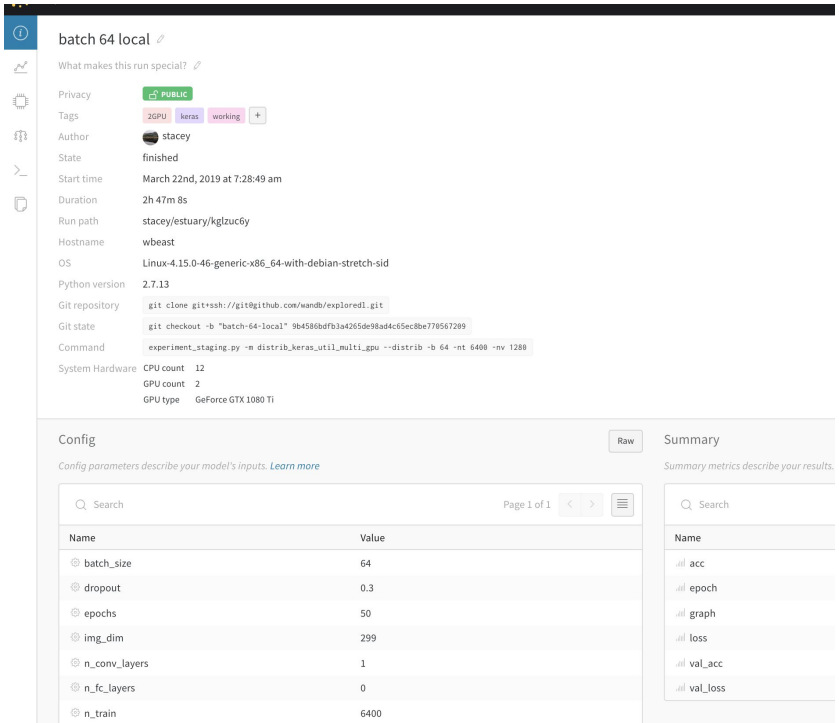
See live updates on model performance, check for overfitting, and visualize how a model performs on different classes.



Slide Credit: cs329s, stanford

Track experiments – After W&B



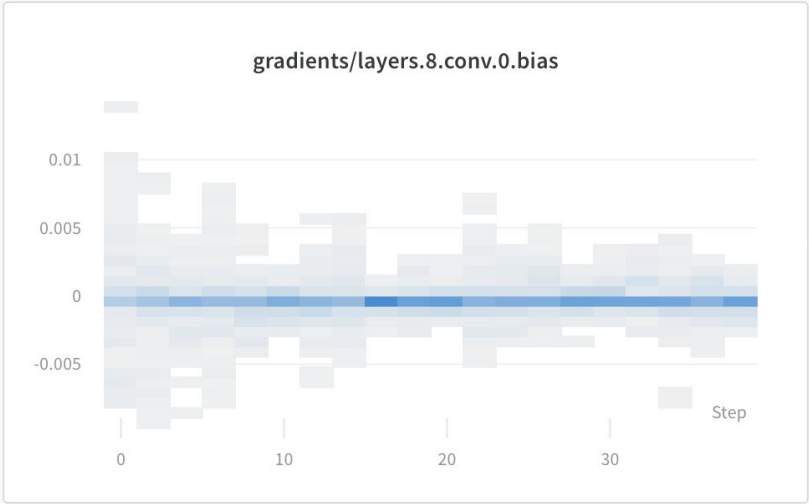


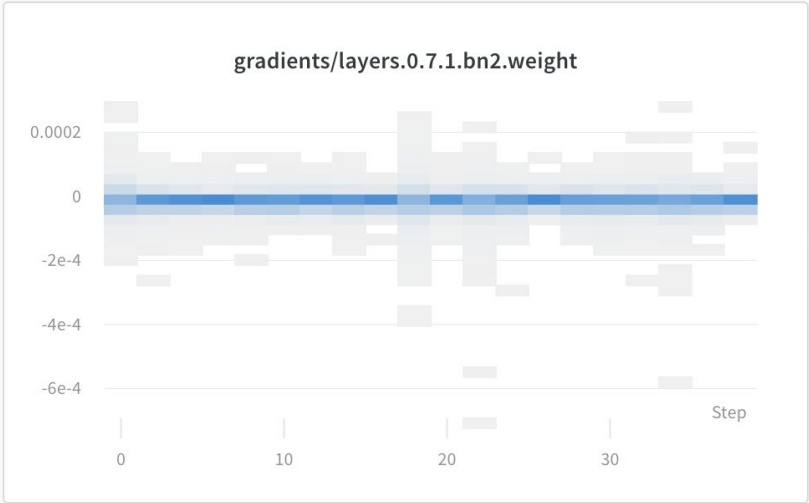
Slide Credit: cs329s, stanford

bit.ly/demo-run

Visualize gradients

See the inner workings of your model, look for convergence during training, and check for exploding gradients.



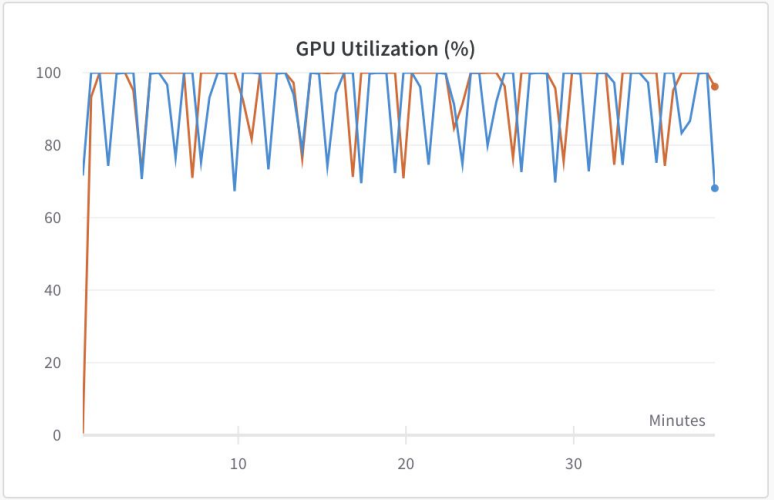


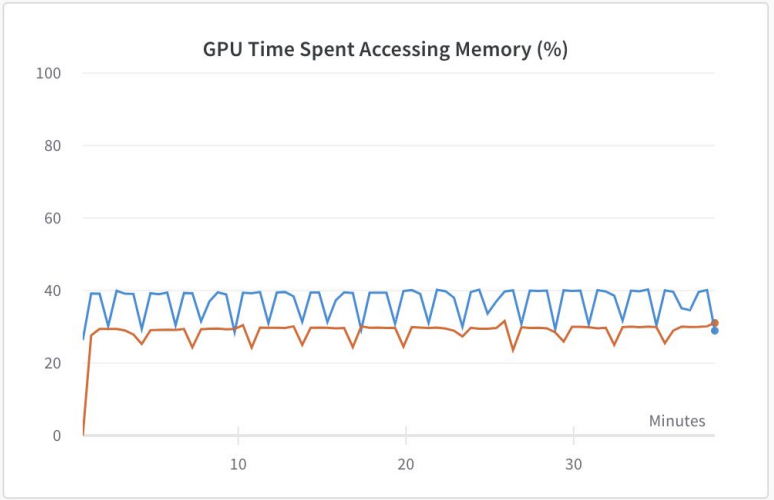
Slide Credit: cs329s, stanford

bit.ly/demo-run

System metrics

Get the most out of your GPUs and identify opportunities for optimizing hardware utilization.





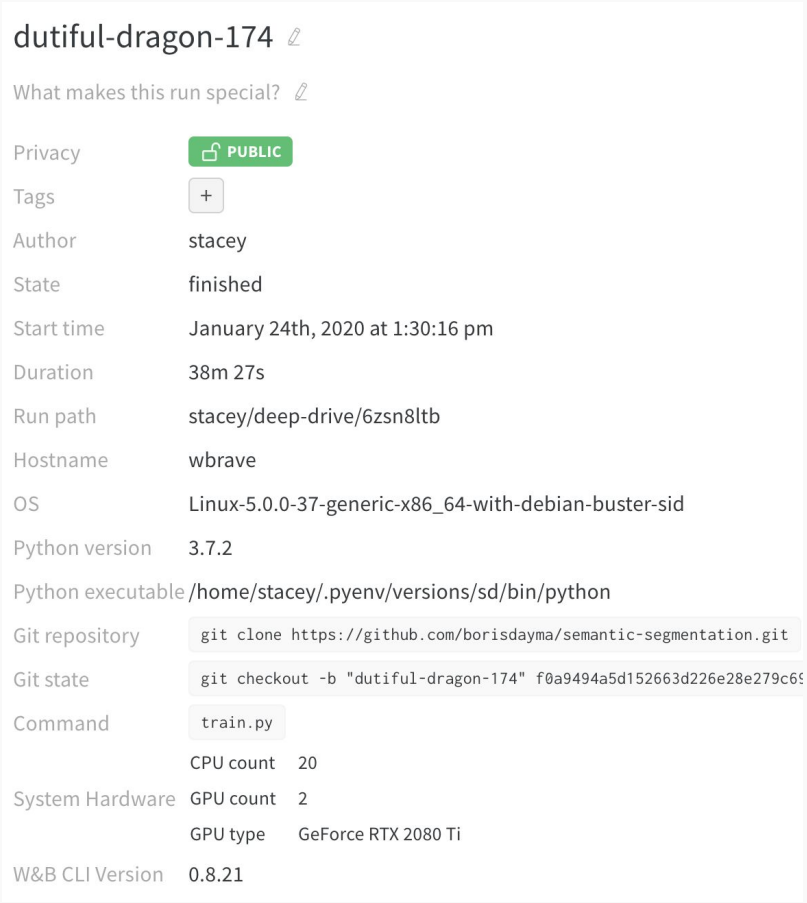
Slide Credit: cs329s, stanford

bit.ly/demo-run

Capture the code

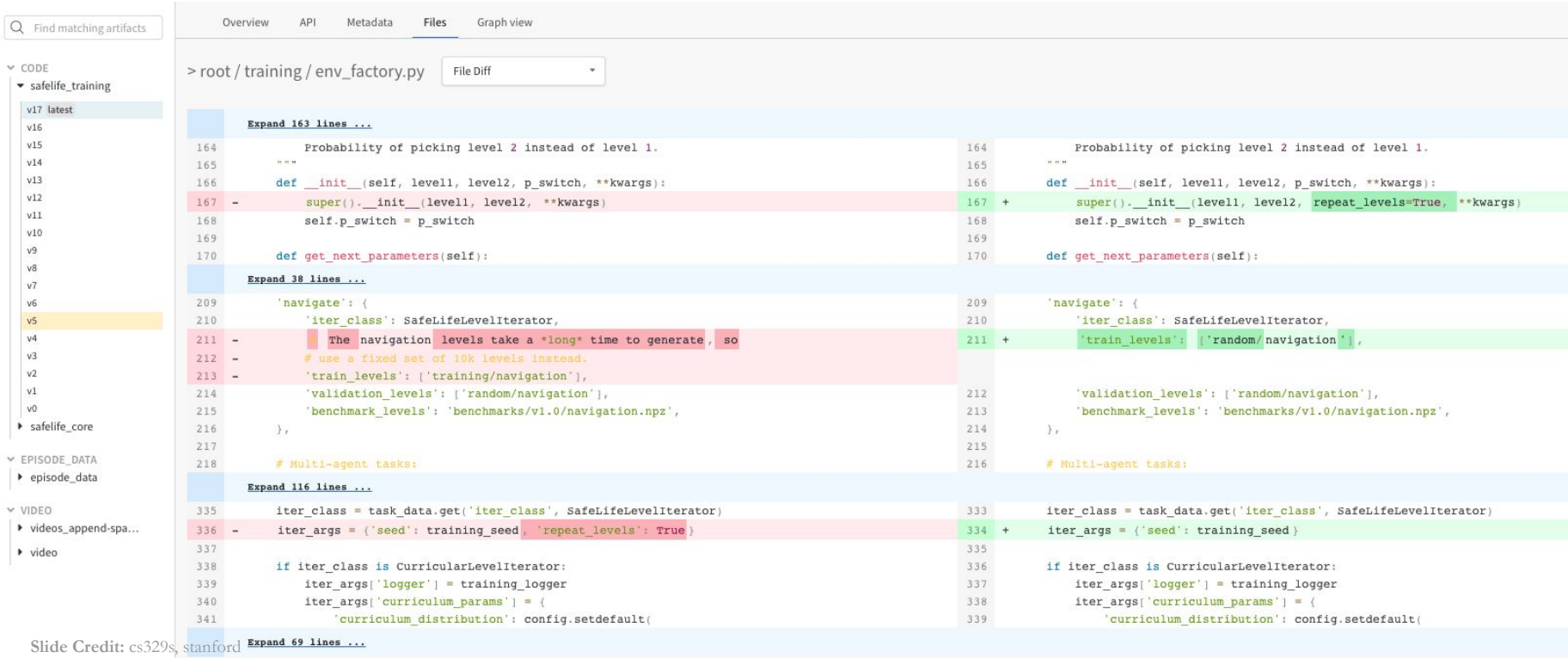
Save the most recent git commit, the command and args, and system hardware setup.

W&B also saves a patch file with uncommitted changes so you can reproduce the exact code that trained the model.



Slide Credit: cs329s, stanford

Code tab with versioning – spot changes



Slide Credit: cs329s, stanford

P.S. Store seeds & randomization settings in wandb.config

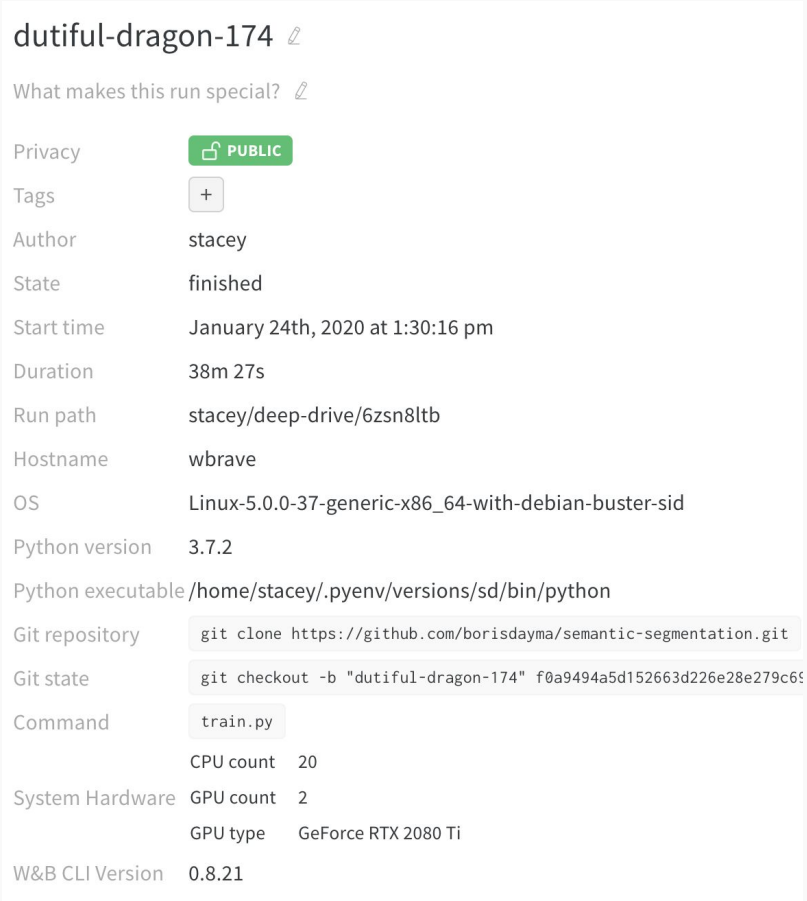
```python
def main(config):
    config.name = config_desc(config)
    if config.use_wandb:
        run.save()

    # set random seed
    tf.random.set_seed(config.random_seed)
    # also set numpy seed to control train/val dataset split
    np.random.seed(config.random_seed)
```

```python
def set_global_seed(config):
    from safelife.random import set_rng

    # Make sure the seed can be represented by floating point exactly.
    # This is just because we want to pass it over the web, and javascript
    # doesn't have 64 bit integers.
    if config.get('seed') is None:
        config['seed'] = np.random.randint(2**53)
    seed = np.random.SeedSequence(config['seed'])
    logger.info("SETTING GLOBAL SEED: %i", seed.entropy)
    set_rng(np.random.default_rng(seed))
    torch.manual_seed(seed.entropy & (2**31 - 1))

    if config['deterministic']:
        # Note that this may slow down performance
        # See https://pytorch.org/docs/stable/notes/randomness.html#cudnn
        torch.backends.cudnn.deterministic = True
```

Slide Credit: cs329s, stanford

Save each script run

- Run overview tab: see all the details
- Save the most recent git commit, the command and args, and system hardware setup (+ requirements.txt!)
- W&B also saves a patch file with uncommitted changes so you can reproduce the exact code that trained the model
- Example run



Slide Credit: cs329s, stanford

bit.ly/deep-drive

Persistent system of record

Query and filter across thousands of runs, and easily keep projects organized.

Runs (395)

3 Filters | Group | Sort | Tag | Move | Columns

Name (228 visualized)	Tags	Runtime	batch_size	encoder	learning_ra	num_train	num_valid	weight_dec	iou	train_loss	valid_loss	acc	traffic_acc	road
best car acc (50% data)	seg_masks	47m 52s	6	resnet34	0.001311	3524	492	0.08173	0.7997	0.5375	0.4427	0.8823	0.8664	0.93
best traffic acc (50% data))	seg_masks	46m 42s	8	resnet18	0.001	3523	492	0.097	0.8073	0.4919	0.4203	0.888	0.8718	0.94
best human iou (50% data)	seg_masks	31m 34s	7	alexnet	0.0009084	1405	190	0.097	0.716	0.6222	0.6259	0.8334	0.8042	0.9
best overall IOU (20% data)	seg_masks	20m 23s	7	resnet34	0.001367	1376	205	0.06731	0.7948	0.5096	0.4659	0.8726	0.8593	0.93
vibrant-cherry-426		6m 12s	8	resnet34	0.001	347	42	0.097	0.752	0.7114	0.6055	0.8474	0.8282	0.95
worldly-totem-422		12m 54s	8	resnet34	0.001	682	97	0.097	0.7523	0.5774	0.5836	0.8566	0.8349	0.91
jumping-voice-421		11m 59s	8	resnet34	0.001	725	92	0.097	0.7449	0.5633	0.5334	0.8504	0.8296	0.91
logical-energy-420	test_only	2m 14s	8	resnet34	0.001	66	10	0.097	0.4297	1.459	1.221	0.626	0.5958	0.76

Slide Credit: cs329s, stanford

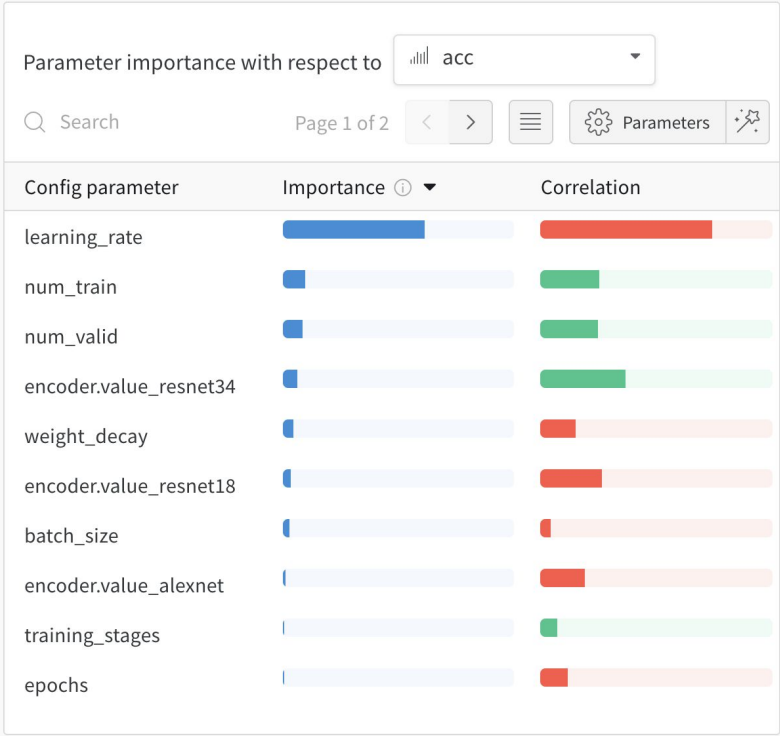
Understand models through interactive visualizations

bit.ly/deep-drive

Compare runs

Visualize the relationships between hyperparameters and model metrics.

Explore the space of possible models quickly, without getting bogged down setting up manual visualizations.



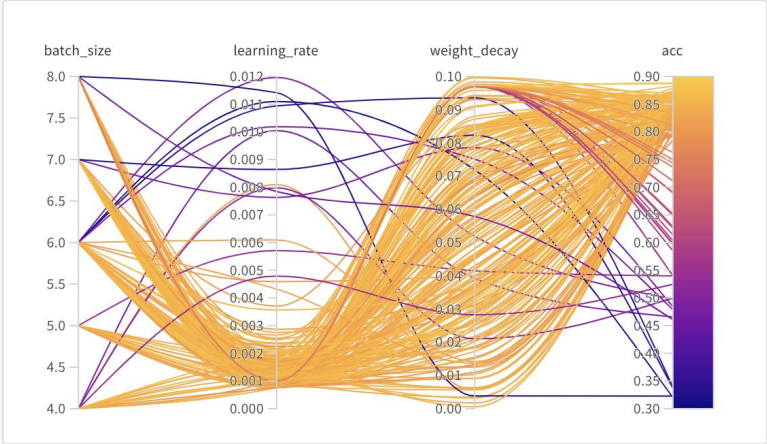
Slide Credit: cs329s, stanford

bit.ly/deep-drive

Iterate quickly
Accelerate time to insight

Use the interactive dashboard to spot issues in real time.

Stop underperforming runs early to optimize resource utilization.



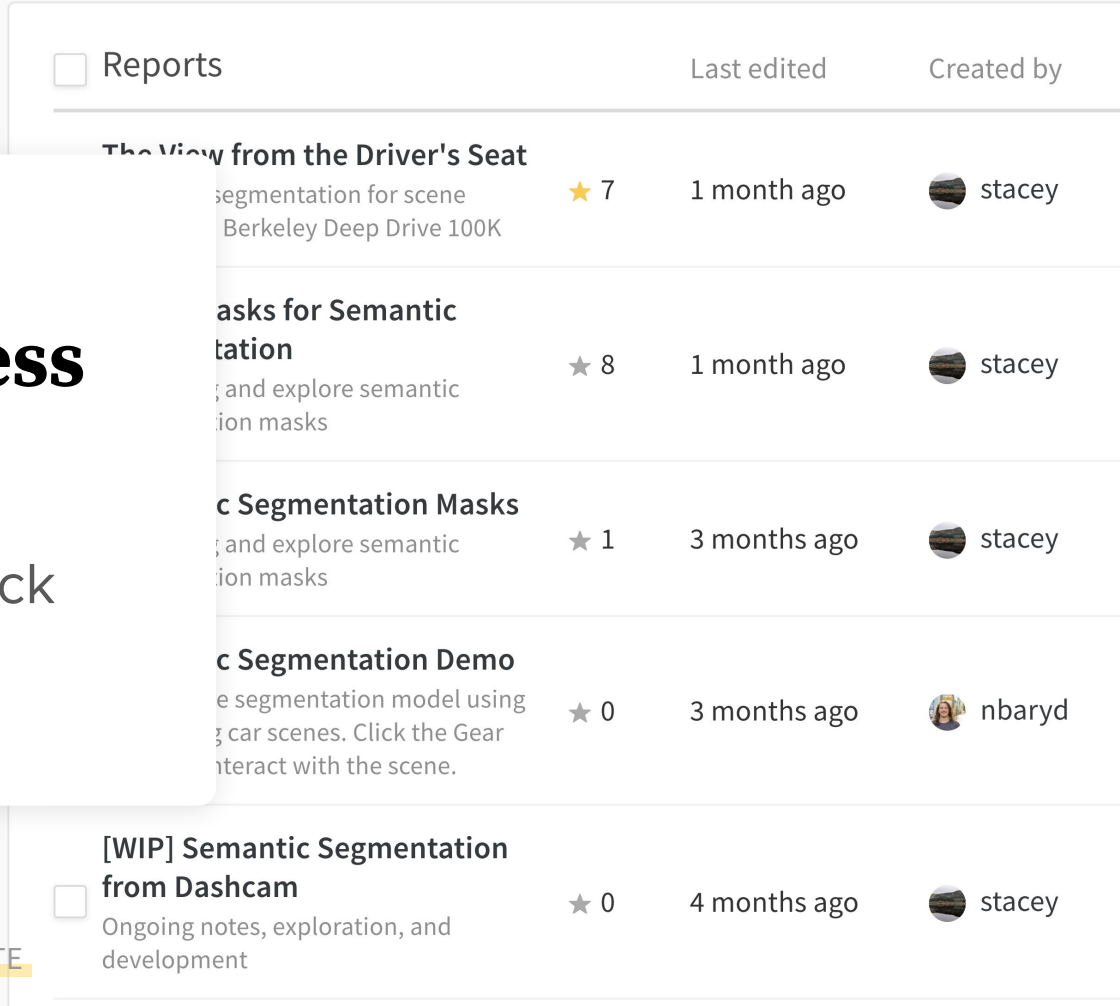
Slide Credit: cs329s, stanford

bit.ly/deep-drive-report

REPORTS

Annotate progress

Use reports to keep a work log and quickly pick up where you left off.



Slide Credit: cs329s, stanford

INSTALL • TRACK • COMPARE • OPTIMIZE • COLLABORATE

bit.ly/deep-drive-report

REPORTS

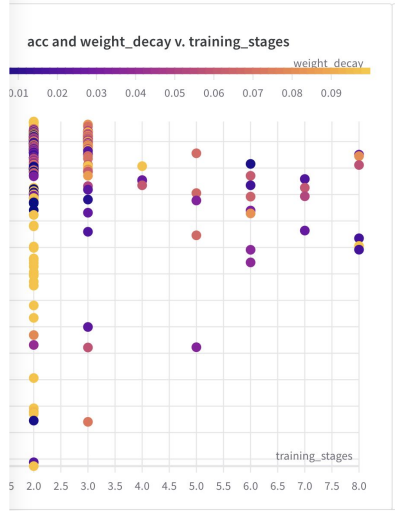
Collaborate easily

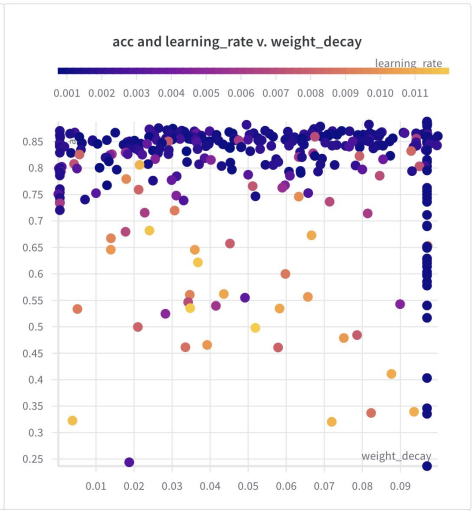
Give team members access to your exploratory results, sharing the context they need to quickly build on your work.

Weight decay: inconclusive

Initially increasing the weight decay 5X improved the accuracy by 9%. Increasing by 200X causes the same amount of improvement though.

erparameter Sweep Insights



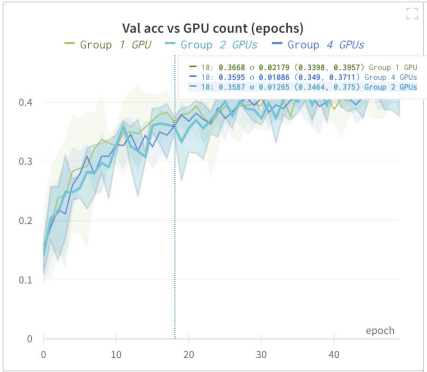


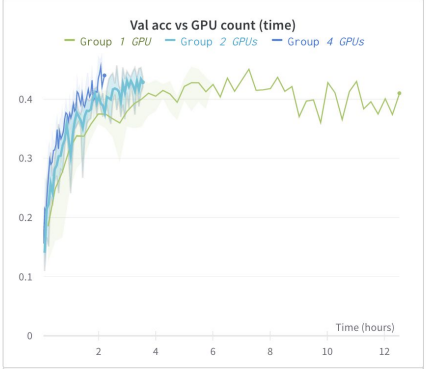
Slide Credit: cs329s, stanford

INSTALL • TRACK • COMPARE • OPTIMIZE • COLLABORATE

Sharing model insights with W&B

Scaling to 4 GPUs





Speed up: 4 GPUs: 2.5X, 2 GPUs: 1.6X

2.5X faster training on 4 GPUs vs 1 GPU

- model reaches a slightly higher validation accuracy 2.5X as fast when using 4 versus 1 GPU—this is the main advantage
- train/val acc/loss are not significantly affected by parallelizing the job across 1, 2, or 4 GPUs—this is expected and reassuring
- could continue tuning to improve relative speed-up—improvement is less than linear
- need better metrics (data throughput, batches per unit time, time to convergence) to quantify the added value of distributed training

Experiment

Train a 7-layer convnet on main iNaturalist dataset (5000 train / 800 val) as a proof of concept for the Keras multi_gpu_model function.

Notes

- initially no noticeable difference between 1 GPU and 2 GPUs—masked by one extremely slow run, batch 64_2, which stalled for 2 hours during training for unknown reasons. Leaving it out of the average shows that 2 GPUs yield a 1.6x acceleration
- accuracy vs batch size: consider effect of both batch size and number of GPUs
- for 2 GPUs, batch 64 > batch 32 / 128 > batch 256, but the effect is not super clear. 64 seems to be the optimal choice for batch size
- some combinations might be slower—resource sharing

Slide Credit: cs329s, stanford

5 lines of code

```
!pip install wandb
import wandb
wandb.init(project="classify_photos")

# save any experiment configuration
wandb.config = {"learning_rate" : 0.001, "epochs" : 10}

# define & train your model

# log any metric from your training script
wandb.log({"acc": accuracy, "val_acc": val_accuracy})
```

Slide Credit: cs329s, stanford

Quickstart docs docs.wandb.com

Understand datasets through interactive visualizations

dataset / pedestrians-at-night

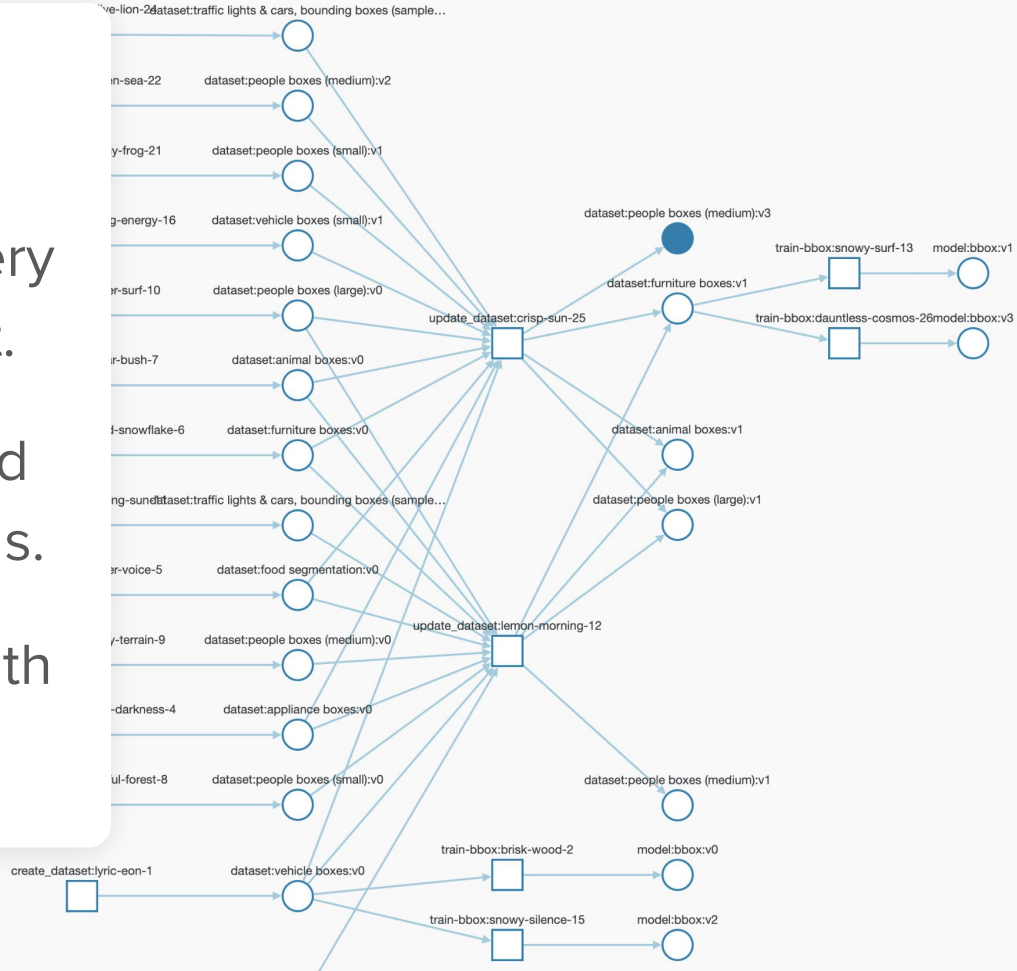
Overview API Metadata Files Graph view

Track artifacts

Get a bird's eye view of every step of model development.

Automatically checksum and version datasets and models.

Host your files anywhere with our infra-agnostic tracking.



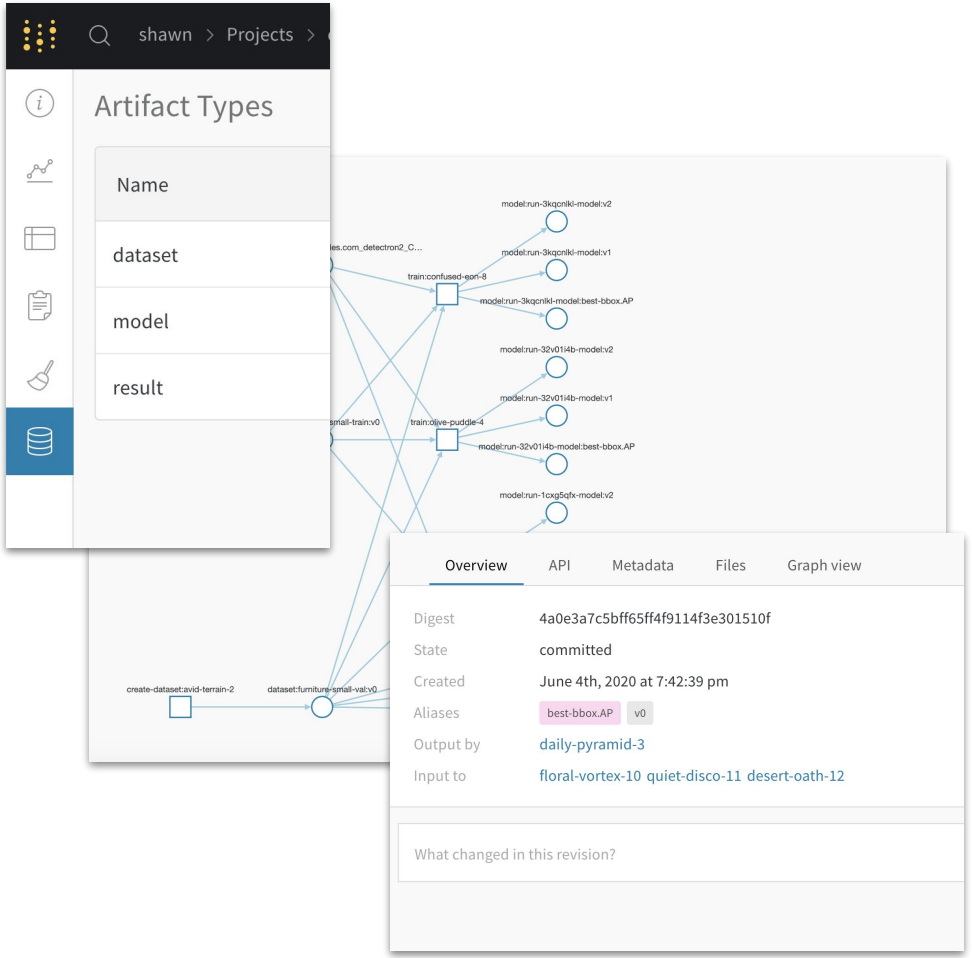
Slide Credit: cs329s, stanford

https://docs.wandb.com/artifacts

Slide Credit: cs329s, stanford

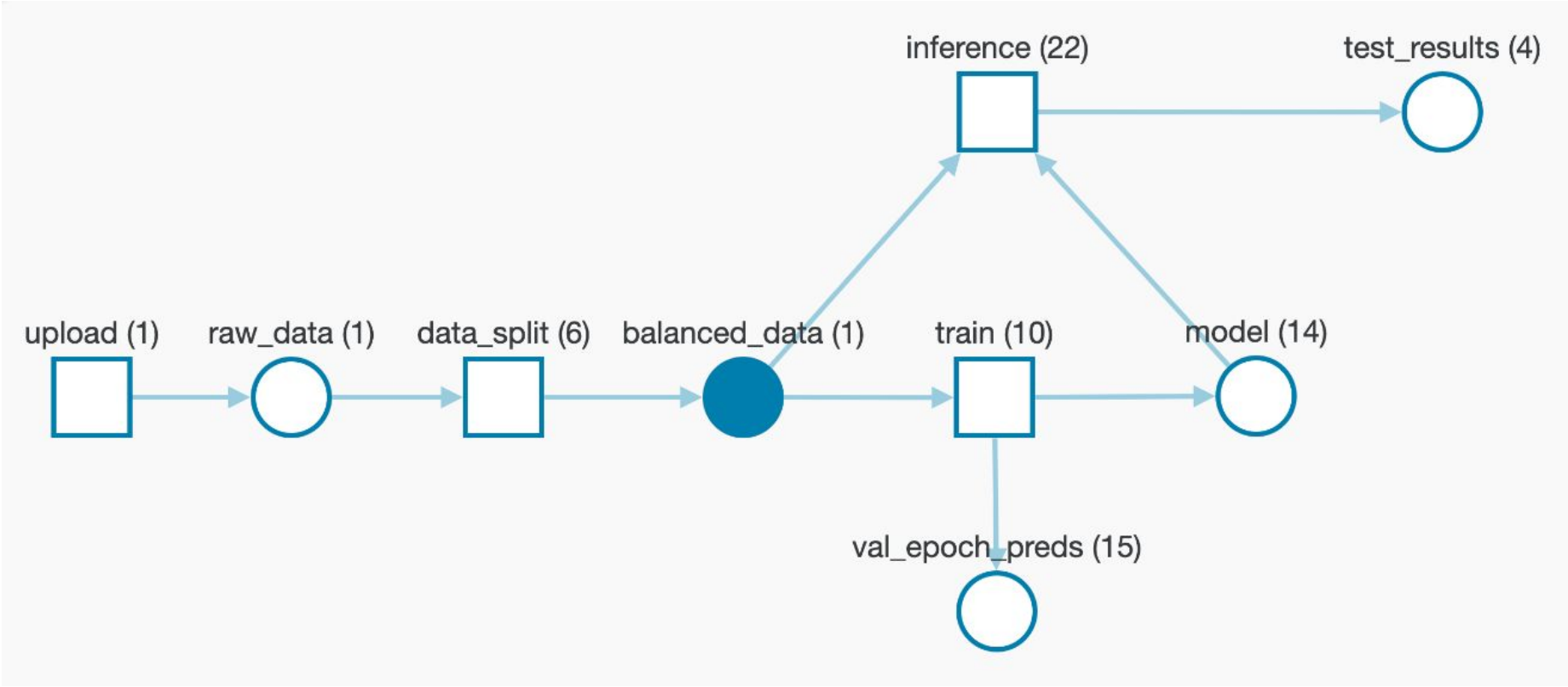
Artifacts

- Integrated with the project
  - Get a clear understanding of what is consumed by and generated from runs and projects
- Can be consumed by any run with access to the project
  - Artifacts as a serving agent for downstream systems
- Automatically generates lineage graphs



Slide Credit: cs329s, stanford

Version datasets, models & any files



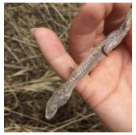
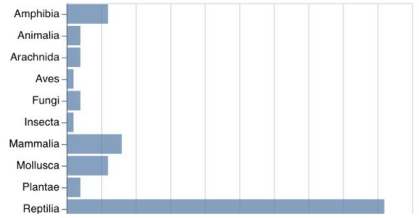
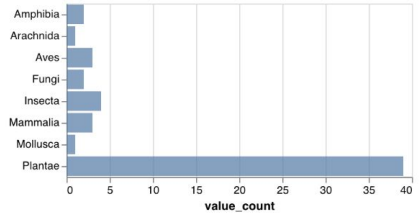
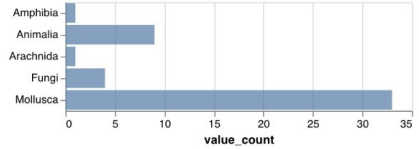
Artifacts Quickstart →

Organize, visualize, and share workflows

Slide Credit: cs329s, stanford

Example →

Images

Group by (guess) ↓	image	truth
Reptilia	   1-3 of 76	
Plantae	   10-12 of 55	
Mollusca	   1-3 of 48	

Slide Credit: cs329s, stanford

[Example →]

[Report →]

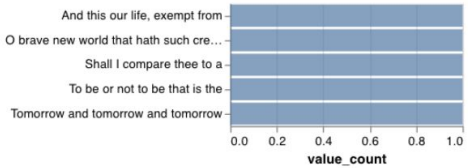
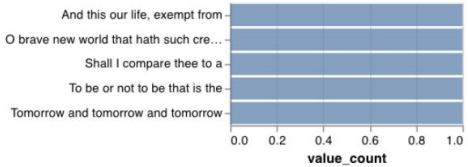
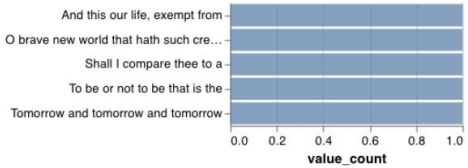
Segmentation masks

id	prediction	ground truth	overall IOU	false positive	false negative	iou_road
2b9bc4f8-780bd01f			0.6061	171737	213406	0.5211
134f6849-00000000			0.4365	350147	61326	0.8978
382e37b6-139b8d9f			0.4425	298525	206063	0.8833
83a3ef4b-9e72764d			0.7372	157810	113137	0.7578

Slide Credit: cs329s, stanford

Example →

Text

Group by (temperature)	prompt	response
0.3		To be or not to be that is the dearty to the brocaty of the persing stands and be is the beard to the sease that that the last may 1-3 of 5
0.6		To be or not to be that is the his visitition of prose, and may make to his tree the best kinous and it be herart. MERCUTIO: Why s 1-3 of 5
0.8		To be or not to be that is the see and hence not it aradements, the stands all one. ATBON: As stay the strange and I am before gon 1-3 of 5

Slide Credit: cs329s, stanford

Example →

Report →

Videos

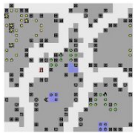
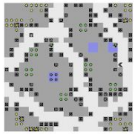
Type: video

- videos_append-spawn
  - v1 latest
  - v0
- video

Overview | API | Metadata | Files | Graph view

Filter

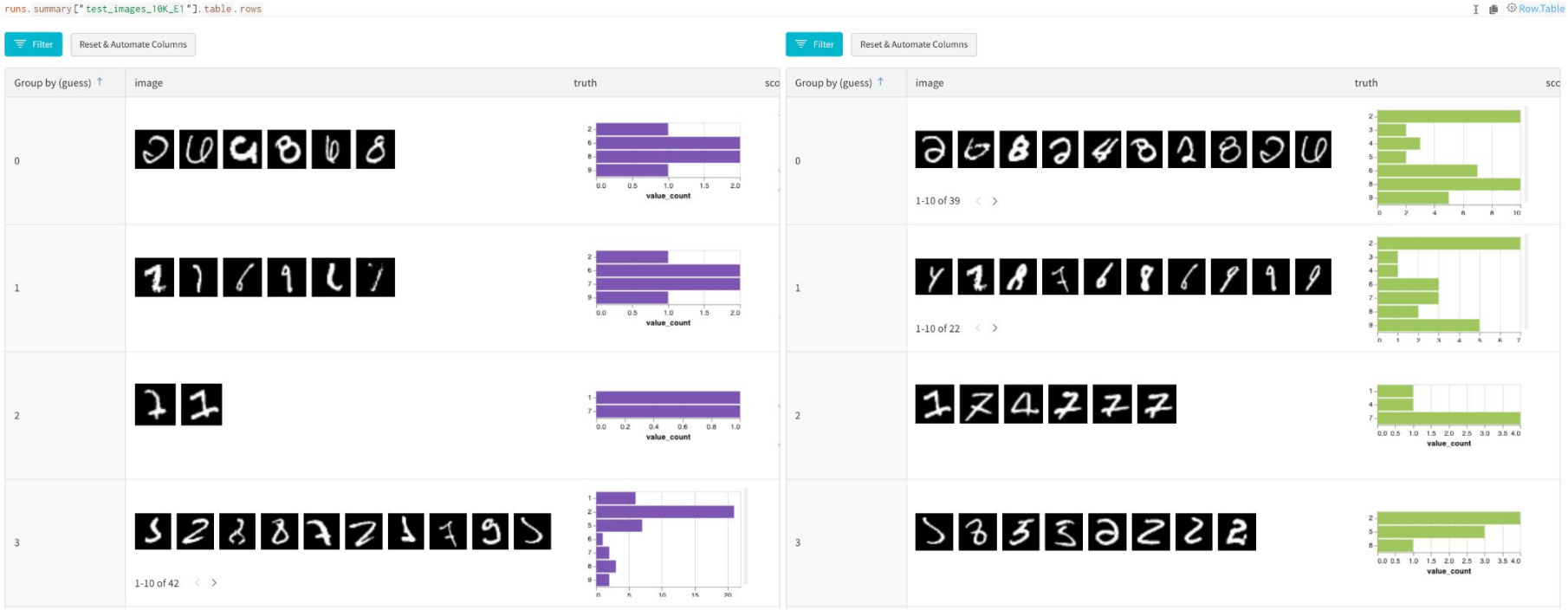
1-10 of 40002

video	side_effects	length	reward	success	score ↓	steps	episodes
	0	50	0.9706	0	96.544	1279994	25597
	0	50	0.9706	0	96.544	1561417	31228
	0	50	0.9706	0	96.544	1893888	37877

Slide Credit: cs329s, stanford

Example →

Interactive exploration & comparison



Slide Credit: cs329s, stanford

[Single table →]

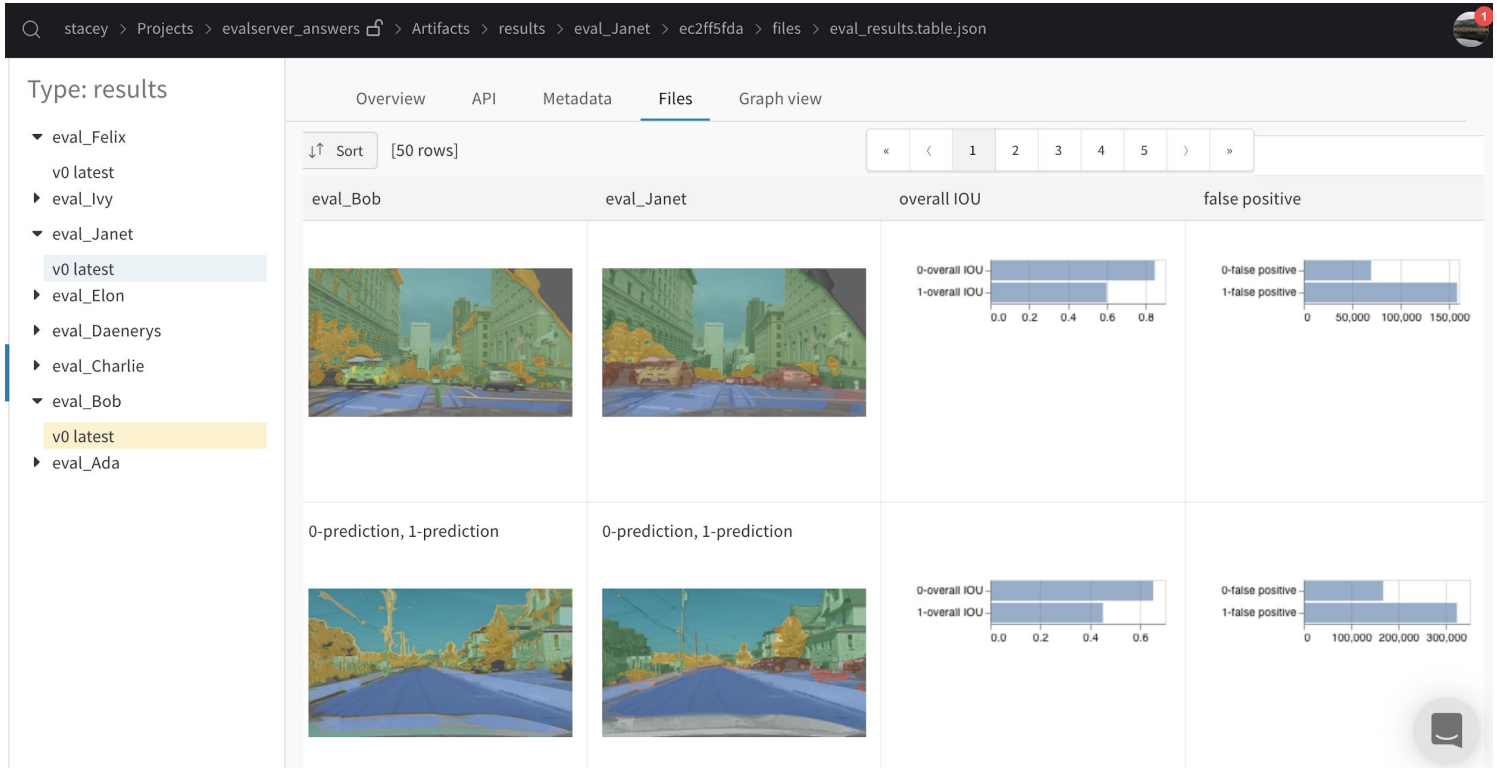
[Comparison →]

Tables Use Cases

- Log interactive media
- Manage data splits & iteratively refine datasets
- Visualize predictions
- Explore dynamically: filter, sort, group, add columns, & more
- Compare model variants
- Save Tables to workspace, report, etc.

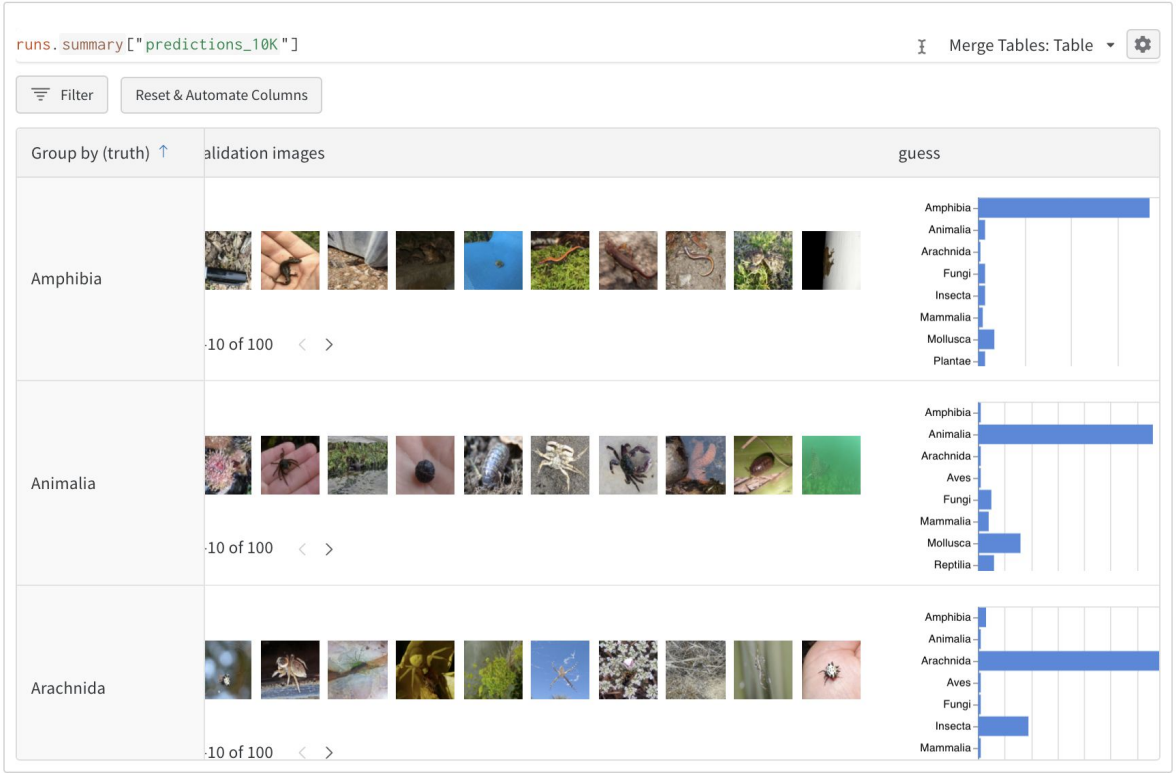
Slide Credit: cs329s, stanford

Example: why are two models different?



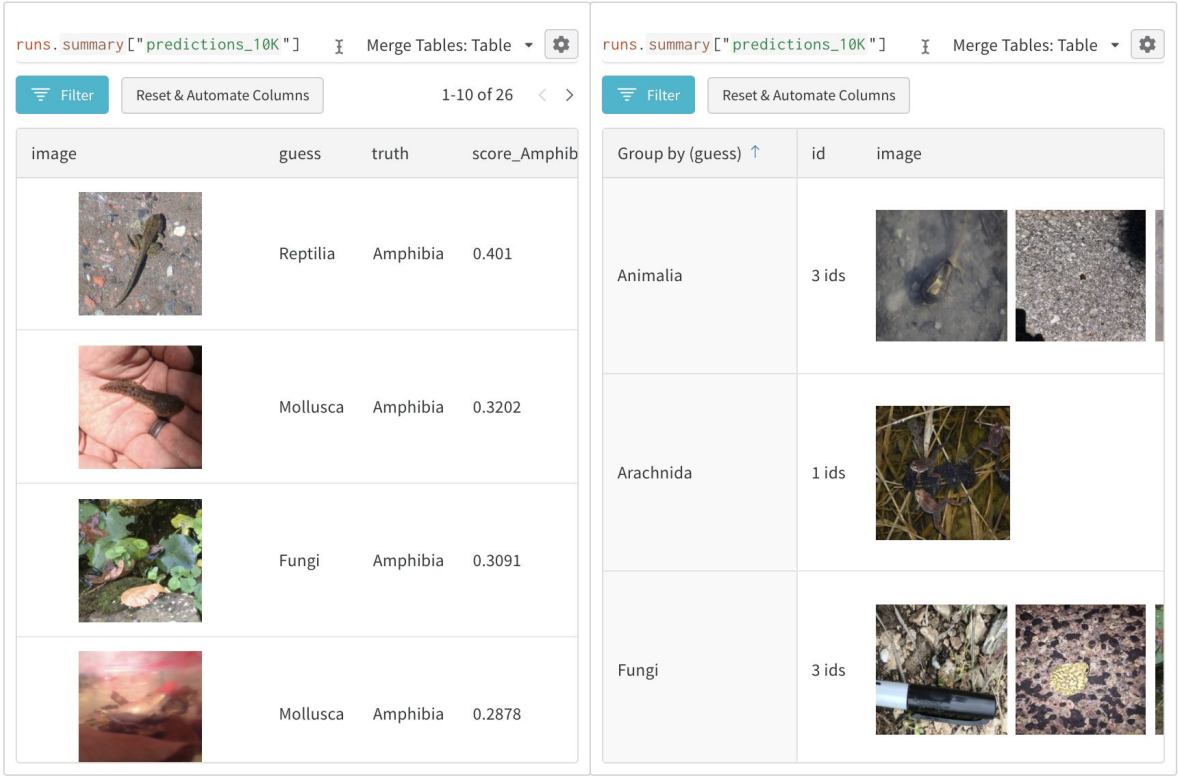
Slide Credit: cs329s, stanford

Spot classes that confound the model



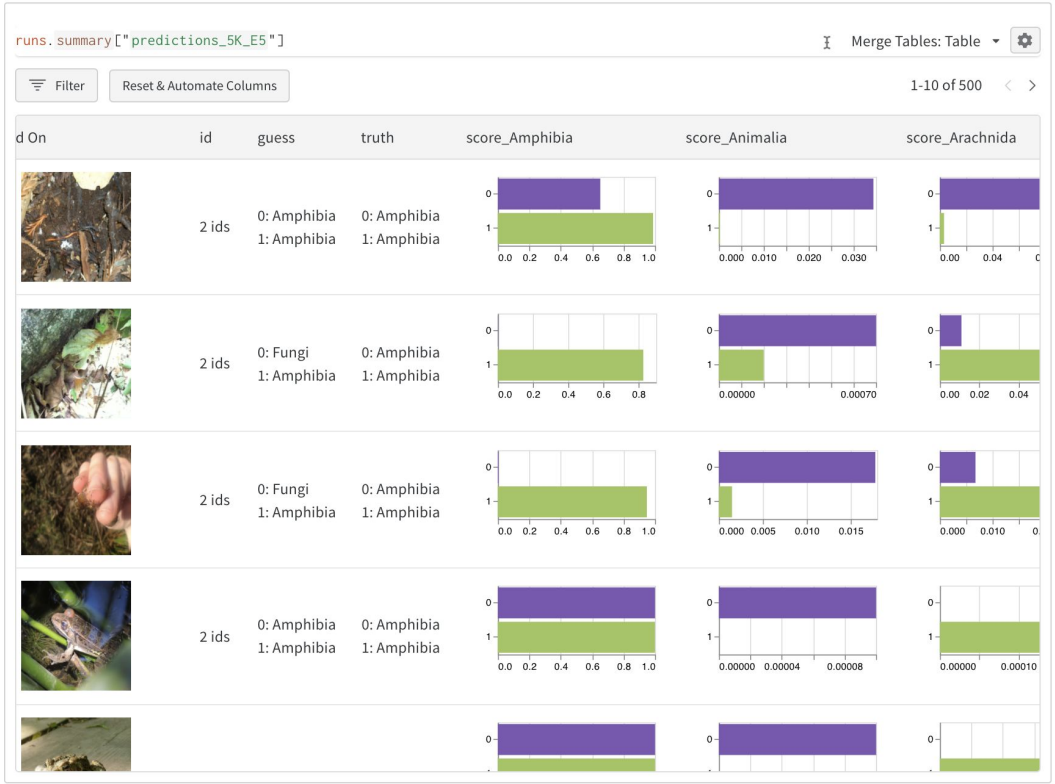
Slide Credit: cs329s, stanford

Spot images that confound the model



Slide Credit: cs329s, stanford

Compare performance across models



Slide Credit: cs329s, stanford