Selección de modelos

Presentación con una selección de las slides de las dos primeras presentaciones de Chip Huyen que se pueden consultar en stanford-cs329s.github.io/syllabus.html

ML algorithm

- Function to be learned
  - E.g. model architecture, number of hidden layers
- Objective function to optimize (minimize)
  - Loss function
- Learning procedure (optimizer)
  - Adam, Momentum

6 tips for evaluating ML algorithms

1. Avoid the state-of-the-art trap

- SOTA's promise
  - Why use an old solution when a newer one exists?
  - It's exciting to work on shiny things
  - Marketing

1. Avoid the state-of-the-art trap

- SOTA's reality
  - SOTA on research data != SOTA on your data
  - Cost
  - Latency
  - Proven industry success
  - Community support

2. Start with the simplest models

- Easier to deploy
  - Deploying early allows validating pipeline
- Easier to debug
- Easier to improve upon

2. Start with the simplest models

- Easier to deploy
  - Deploying early allows validating pipeline
- Easier to debug
- Easier to improve upon
- Simplest models != models with the least effort
  - BERT is easy to start with pretrained model, but not the simplest

3. Avoid human biases in selecting models

- A tale of human biases
  - Papers proposing LSTM variants show that the variants improve upon the vanilla LSTM.
  - Do they?

3. Avoid human biases in selecting models

- A tale of human biases
  - Papers proposing LSTM variants show that the variants improve upon the vanilla LSTM.
  - Do they?

LSTM: A Search Space Odyssey

Klaus Greff, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber

We conclude that the most commonly used LSTM architecture (vanilla LSTM) performs reasonably well on various datasets. None of the eight investigated modifications significantly improves performance.

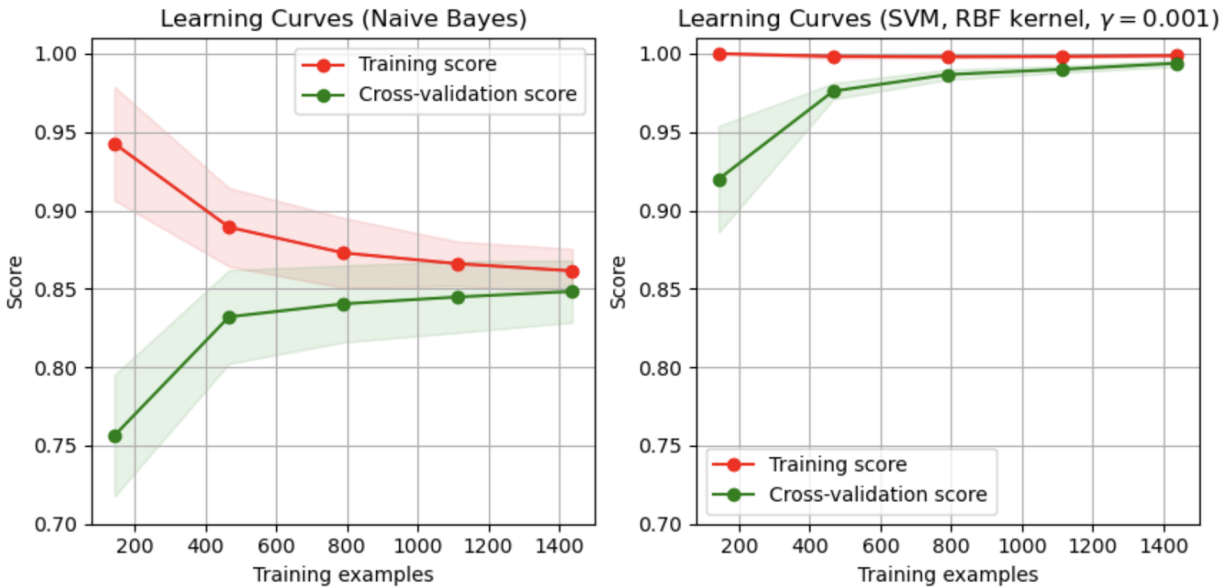
3. Avoid human biases in selecting models

- It’s important to evaluate models under comparable conditions
  - It’s tempting to run more experiments for X because you’re more excited about X
- Near-impossible to make blanketed claims that X is *always* better than Y

Better now vs. better later

- Best model now != best model in 2 months
  - Improvement potential with more data
  - Ease of update

Learning curve



Good for estimating if performance can improve with more data

https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html

5. Evaluate trade-offs

- False positives vs. false negatives
- Accuracy vs. compute/latency
- Accuracy vs. interpretability

6. Understand your model's assumption

- IID
  - **Neural networks** assume that examples are **independent and identically distributed**
- Smoothness
  - **Supervised algorithms** assume that there's a set of functions that can transform inputs into outputs such that **similar inputs are transformed into similar outputs**
- Tractability
  - Let X be the input and Z be the latent representation of X. **Generative models** assume that it's tractable to compute P(Z|X).
- Boundaries
  - **Linear classifiers** assume that **decision boundaries are linear**.
- Conditional independence
  - **Naive Bayes classifiers** assume that the **attribute values are independent of each other given the class.**

6 tips for evaluating ML algorithms

1. Avoid the state-of-the-art trap
2. Start with the simplest models
3. Avoid human biases in selecting models
4. Evaluate good performance now vs. good performance later
5. Evaluate trade-offs
6. Understand your model’s assumptions

Ensembles

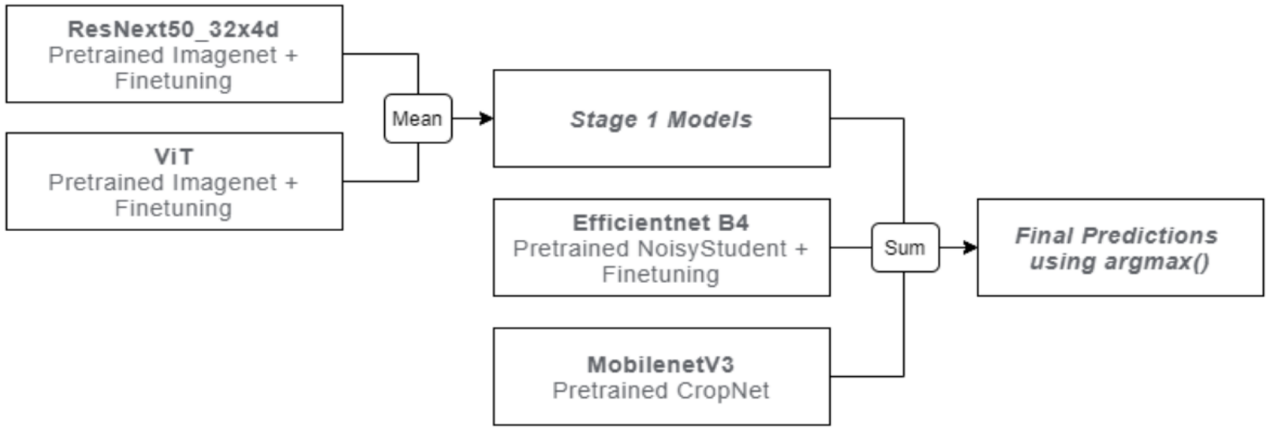
Ensemble

- Creating a strong model from an ensemble of weak models

Base learners

Ensembles: extremely common in leaderboard style projects

- 20/22 winning solutions on Kaggle in Jan - Aug 2021 use ensembles
- One solution uses 33 models!



https://farid.one/kaggle-solutions

Why does ensembling work?

- Task: email classification (SPAM / NOT SPAM)
- 3 uncorrelated models, each with accuracy of 70%
- Ensemble: majority vote of these 3 models
    - Ensemble is correct is at least 2 models are correct

Why does ensembling work?

- 3 models, each with 70% accuracy
- Ensemble is correct if at least 2 models are correct
- Probability at least 2 models are correct: 34.3% + 44.1% = 78.4%

Outputs of 3 models	**Probability**	**Ensemble's output**
All 3 are correct	0.7 * 0.7 * 0.7 = 0.343	Correct
Only 2 are correct	(0.7 * 0.7 * 0.3) * 3 = 0.441	Correct
Only 1 is correct	(0.3 * 0.3 * 0.7) * 3 = 0.189	Wrong
None is correct	0.3 * 0.3 * 0.3 = 0.027	Wrong

Why does ensembling work?

- 3 models, each with 70% accuracy
- Ensemble is correct if at least 2 models are correct
- Probability at least 2 models are correct: 34.3% + 44.1% = 78.4%

- The less correlation among base learners, the better
- Common for base learners to have different architectures

Ensemble

- Bagging
- Boosting
- Stacking

Bagging

- Sample with replacement to create different datasets
- Train a classifier with each dataset
- Aggregate predictions from classifiers
    - e.g. average, majority vote

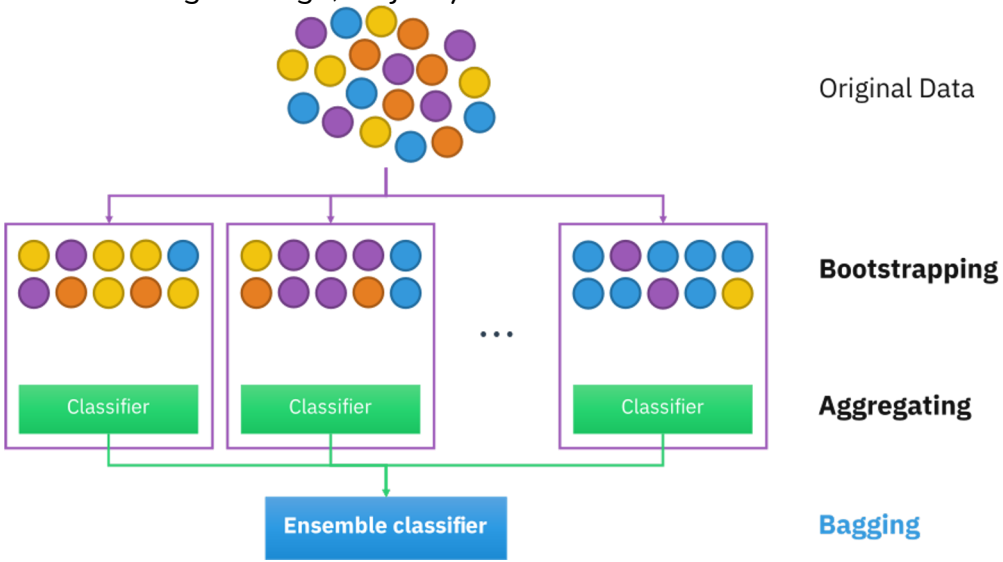
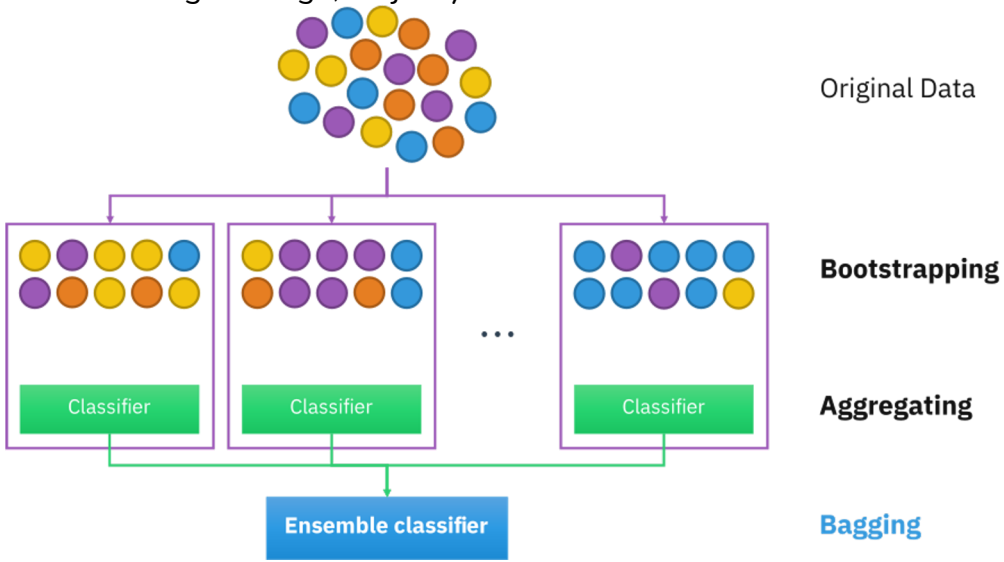


Illustration by Sirakorn

Bagging

- Sample with replacement to create different datasets
- Train a classifier with each dataset
- Aggregate predictions from classifiers
  - e.g. average, majority vote



- Generally improves unstable methods e.g. neural networks, trees
- Can degrade stable methods e.g. kNN

Bagging Predictors (Leo Breiman, 1996)

Illustration by Sirakorn

Boosting

1. Train a weak classifier
2. Give samples misclassified by weak classifier higher weight
3. Repeat (1) on this reweighted data as many iterations as needed
4. Final strong classifier: weighted combination of existing classifiers
    a. classifiers with smaller training errors have higher weights

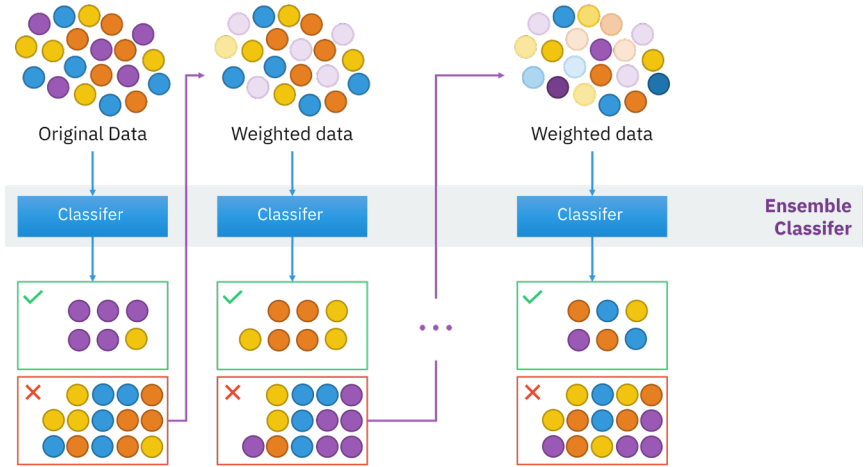
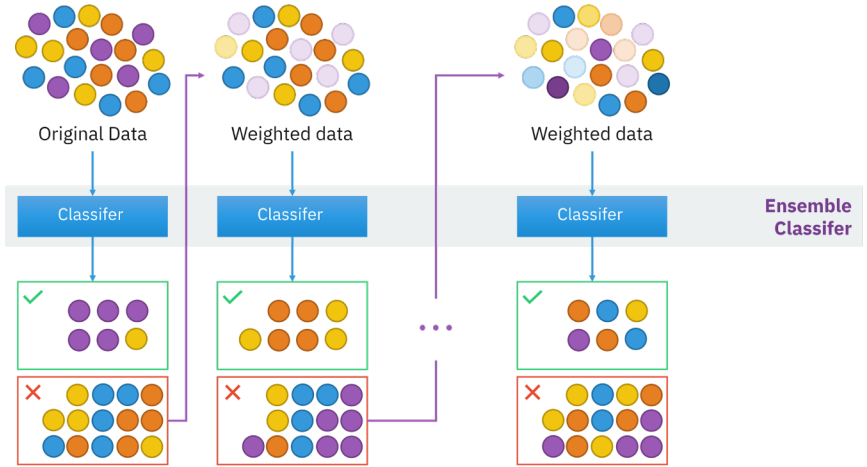


Illustration by Sirakorn

Boosting

1. Train a weak classifier
2. Give samples misclassified by weak classifier higher weight
3. Repeat (1) on this reweighted data as many iterations as needed
4. Final strong classifier: weighted combination of existing classifiers
    a. classifiers with smaller training errors have higher weights

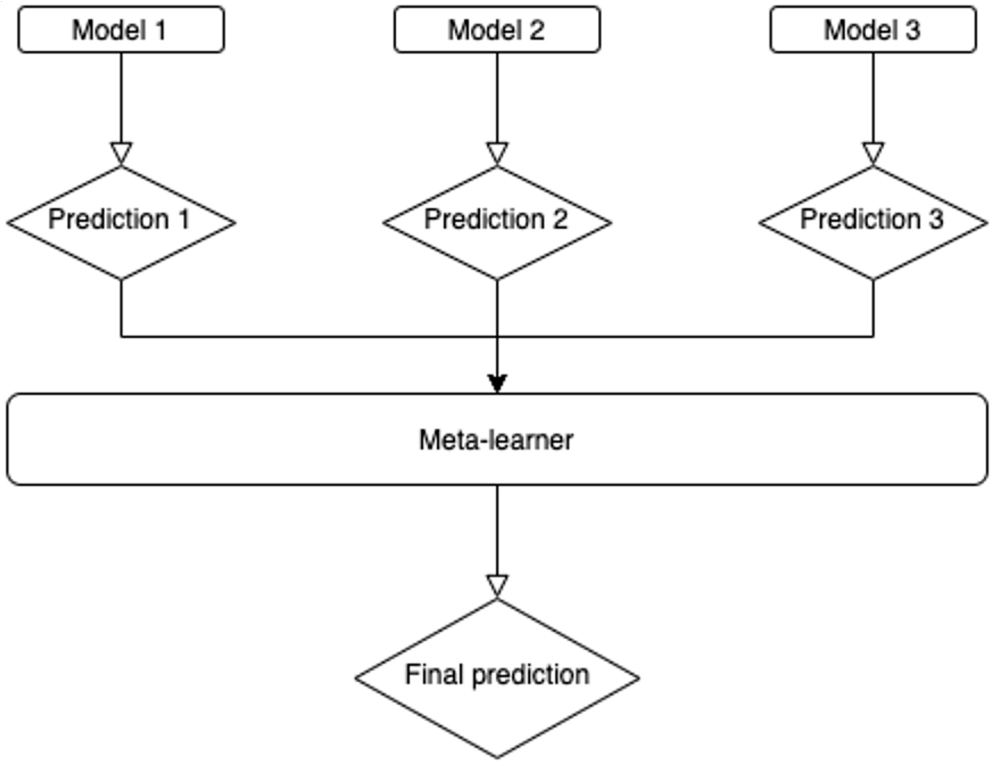


Extremely popular:

- XGBoost
- LightGBM

Illustration by Sirakorn

Stacking

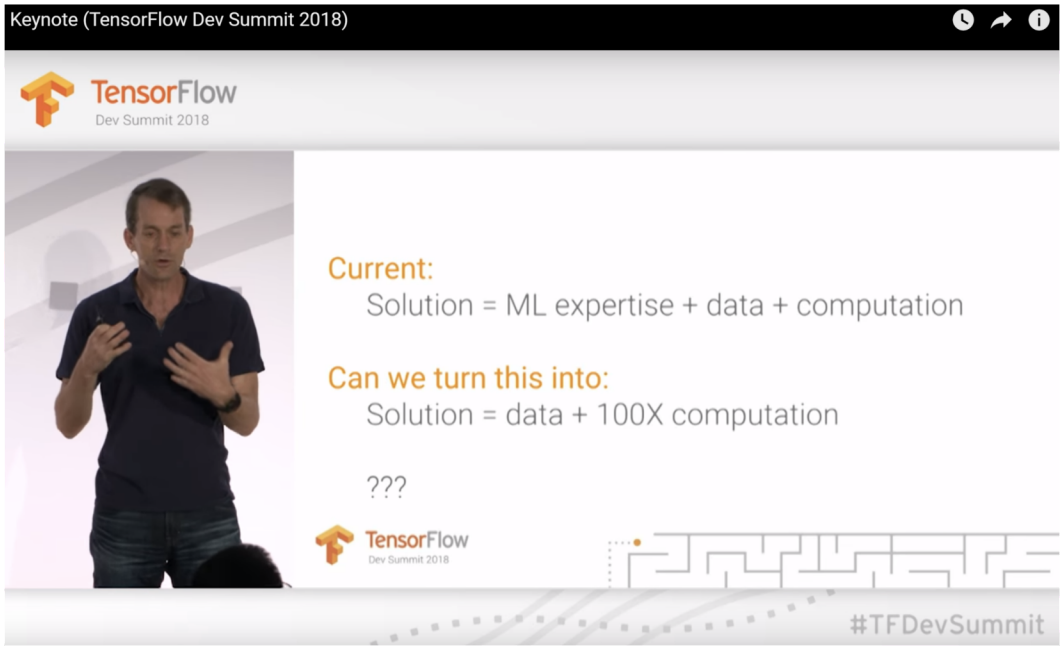


Majority vote, logistic regression, simple NN

AutoML

AutoML

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



AutoML

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

More computationally expensive

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)

Soft AutoML: Hyperparameter tuning

- Many hyperparameters to tune

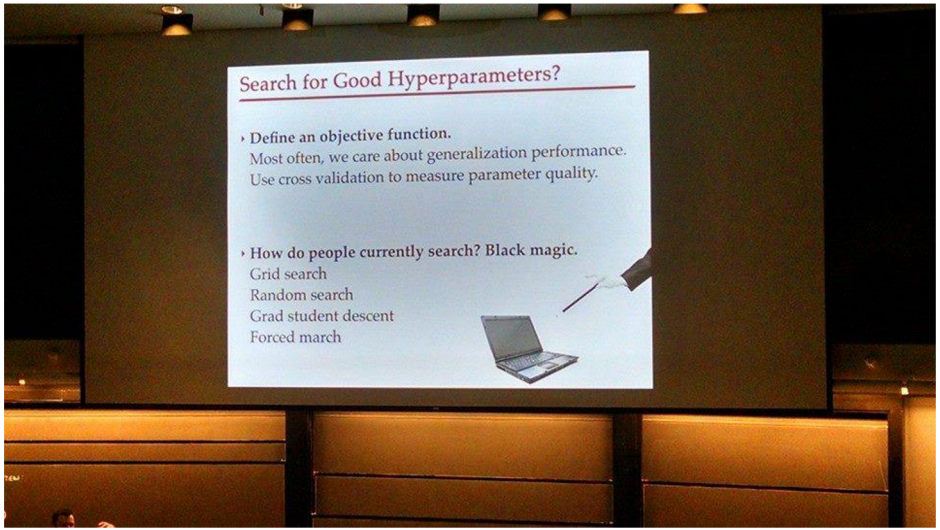
```
model_type = "bert"

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
):
```


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



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

NAS: Neural architecture search

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

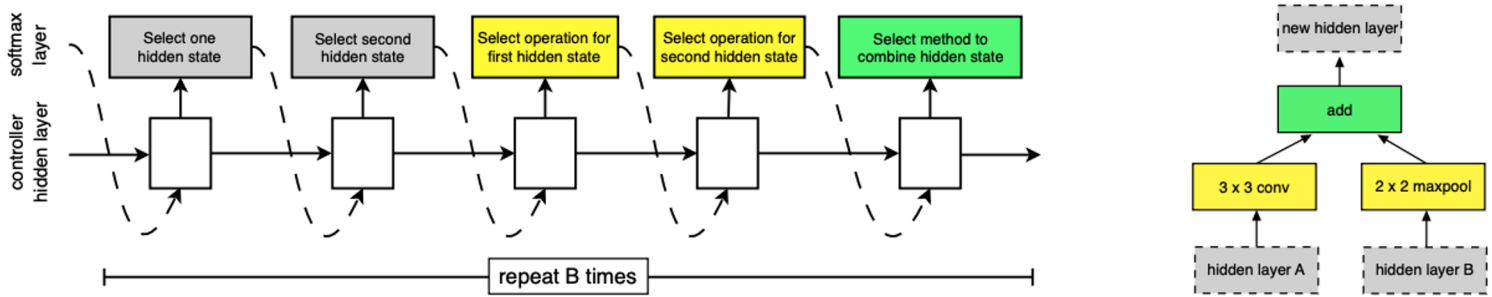


Figure 3. Controller model architecture for recursively constructing one block of a convolutional cell. Each block requires selecting 5 discrete parameters, each of which corresponds to the output of a softmax layer. Example constructed block shown on right. A convolutional cell contains B blocks, hence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

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.

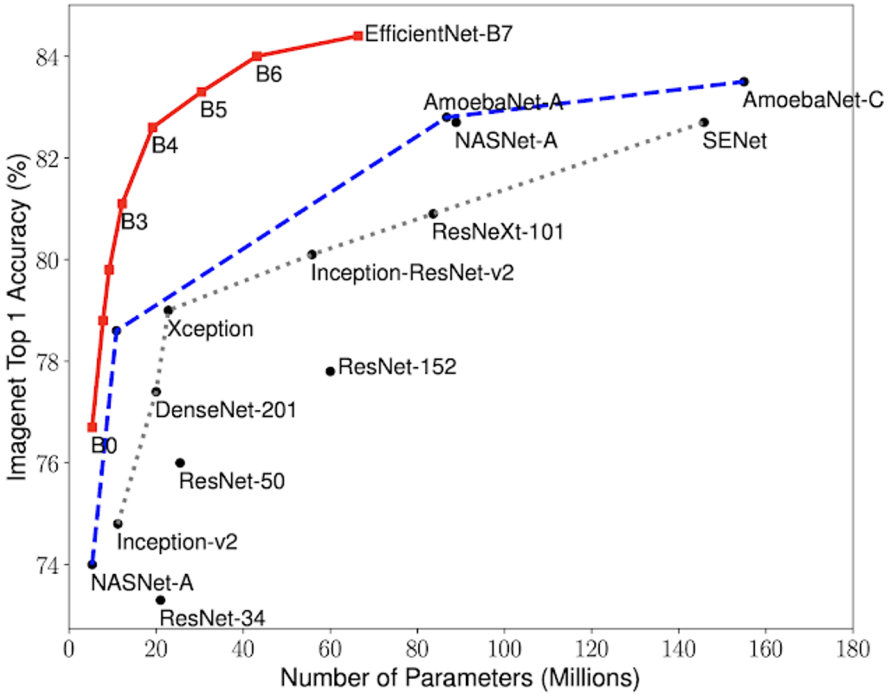
NAS: Neural architecture search

- 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

NAS: Neural architecture search

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

Very successful



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

Learning: architecture + learning algorithm

- Learning algorithm:
  - A set of functions that specifies how to update the weights.
  - Also called **optimizers**
    - Adam, Momentum, SGD

Learned optimizer

Deep learning

engineering features → learning features

SIFT (Lowe et. al. 1999)
HOG (Dalal et. al. 2005)

LeNet (LeCun et. al. 1998)
AlexNet (Krizhevsky et. al. 2012)

Meta learning

engineering to learn → learning to learn

SGD (Robbins et. al. 1951, Bottou 2010)
Autoencoders (Hinton et. al. 2006)

Learning To Learn (Hochreiter et. al. 2001)
Learned Optimizers (Andrychowicz et. al. 2016, Li et. al. 2016, Wichrowska et. al. 2017, Metz et. al. 2018, 2019)

Slide courtesy of Luke Metz

Learned optimizer

- Learn how to learn on a set of tasks
- Generalize to new tasks

Using a thousand optimization tasks to learn hyperparameter search strategies

Luke Metz [1] Niru Maheswaranathan [1] Ruoxi Sun [1] C. Daniel Freeman [1] Ben Poole [1] Jascha Sohl-Dickstein [1]

Slide courtesy of Luke Metz

Learned optimizer

- Learn how to learn on a set of tasks
- Generalize to new tasks
- The learned optimizer can then be used to train a better version of itself!

Using a thousand optimization tasks to learn hyperparameter search strategies

Luke Metz [1] Niru Maheswaranathan [1] Ruoxi Sun [1] C. Daniel Freeman [1] Ben Poole [1] Jascha Sohl-Dickstein [1]

Slide courtesy of Luke Metz

Ways a model can scale

1. In complexity: architecture, number of parameters

Ways a model can scale

1. In complexity: architecture, number of parameters
2. In prediction traffic

Ways a model can scale

1. In complexity: architecture, number of parameters
2. In prediction traffic
3. In number of models

Rise of Incredibly Large DL Models

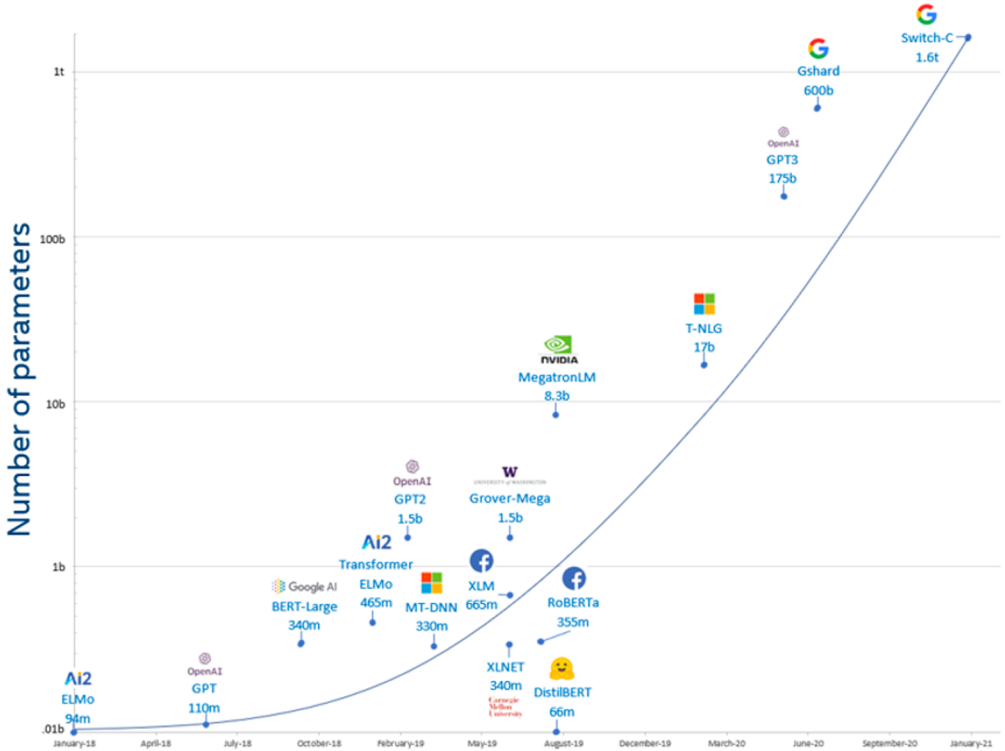
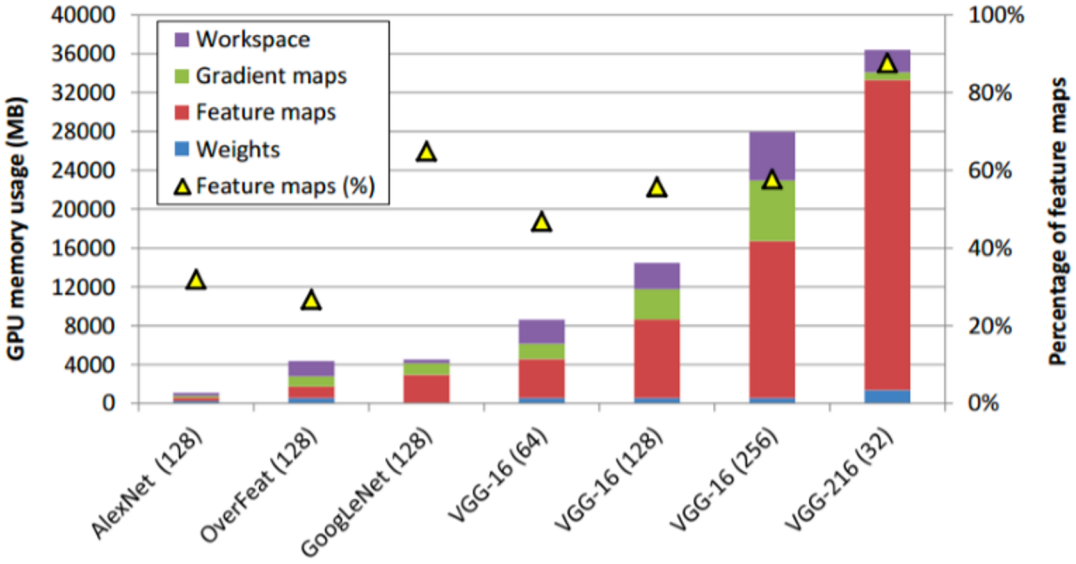


Figure 1: Exponential growth of number of parameters in DL models

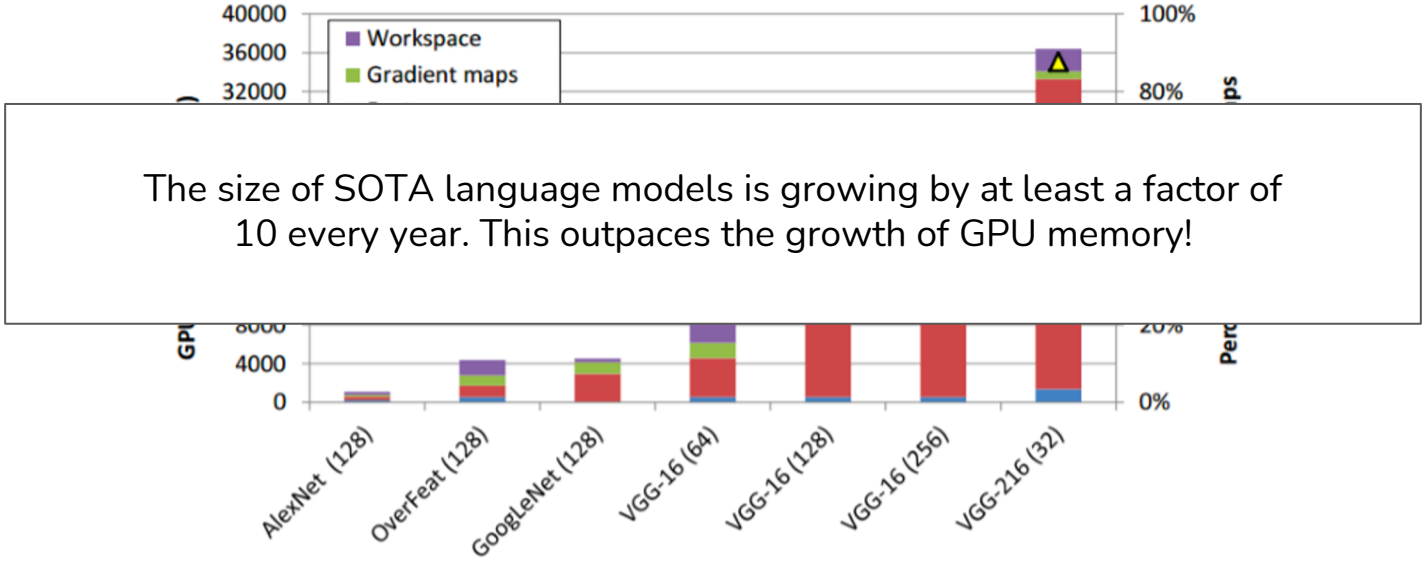
Illustration by [Towards Data Science]

GPU Usage



Rhu, M., et al, vDNN: Virtualized Deep Neural Networks for Scalable, Memory-Efficient Neural Network Design

GPU Usage



The size of SOTA language models is growing by at least a factor of 10 every year. This outpaces the growth of GPU memory!

Rhu, M., et al, vDNN: Virtualized Deep Neural Networks for Scalable, Memory-Efficient Neural Network Design

Breakout exercise

What went wrong with Zillow Offers?

Zillow, facing big losses, quits flipping houses and will lay off a quarter of its staff.

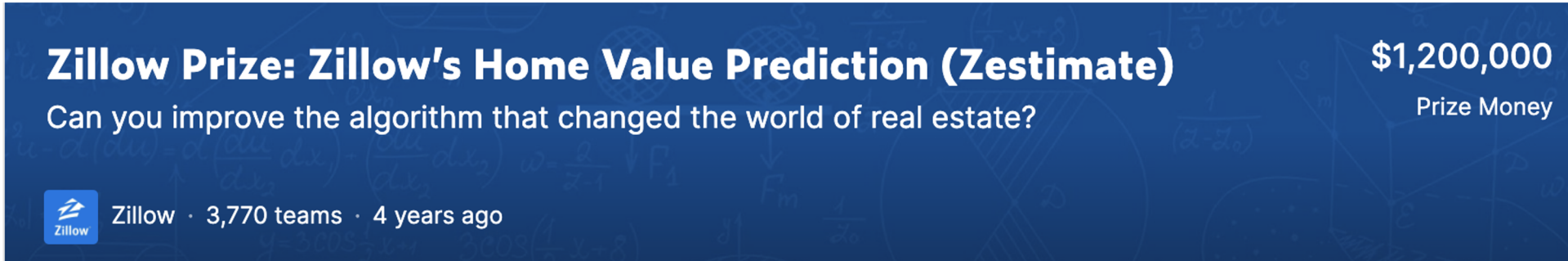
The real estate website had been relying on its algorithm that estimates home values to buy and resell homes. That part of its business lost about $420 million in three months.

Zillow is sitting on thousands of houses worth less than what the company paid for them. Caitlin O'Hara for The New York Times

https://www.nytimes.com/2021/11/02/business/zillow-q3-earnings-home-flipping-ibuying.html (November, 2021)

Blaming game

1. Prophet
2. Kaggle-style data science
3. Leadership
4. ML/DS team



What went wrong with Zillow Offers?

1. Use ML to predict home prices
2. Use predicted prices to flip houses
3. ML models over-predict house prices
4. Buy houses at higher prices

Group of 5, 10 minutes

1. What might be the causes of ML models over-predicting house prices?
    a. Hint: what market conditions have changed in the last 2 years?
2. If you were on their team, what would you have done to prevent this problem?

ML offline evaluation

Facebook translates 'good morning' into 'attack them', leading to arrest

Palestinian man questioned by Israeli police after embarrassing mistranslation of caption under photo of him leaning against bulldozer

0 People Like You

Model evaluation

- Offline evaluation: before deployed
- Online evaluation: after deployed

Test in production. Will cover this later!

Model offline evaluation

- Baselines
- Evaluation methods

Baselines

- Numbers by themselves mean little
- Task: binary classification, 90% POSITIVE, 10% NEGATIVE
- F1 score: 0.90

Is it model good or bad?

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	?
Random [label distribution]	0.98	?

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	0.02
Random [label distribution]	0.98	0.01

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution
- **Zero rule baseline**
  - Always predict the most common class

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	0.02
Random [label distribution]	0.98	0.01
Most common [preds = [0] * n]	?	?

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution
- **Zero rule baseline**
  - Always predict the most common class
- **Simple heuristics**
  - E.g.: classify tweets based on whether they contain links to unreliable sources

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	0.02
Random [label distribution]	0.98	0.01
Most common [preds = [0] * n]	?	?
Simple heuristics	?	?

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution
- **Zero rule baseline**
  - Always predict the most common class
- **Simple heuristics**
  - E.g.: classify tweets based on whether they contain links to unreliable sources
- **Human baseline**
  - What's human-level performance?

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	0.02
Random [label distribution]	0.98	0.01
Most common [preds = [0] * n]	?	?
Simple heuristics	?	?
Human expert	?	?

Model selection: baselines

- **Random baseline**
  - Predict at random:
    - uniform
    - following label distribution
- **Zero rule baseline**
  - Always predict the most common class
- **Simple heuristics**
  - E.g.: classify tweets based on whether they contain links to unreliable sources
- **Human baseline**
  - What's human-level performance?
- **Existing solutions**

- **Example**: misinformation classification
  - n = 1,000,000
  - 99% negative (label = 0)
  - 1% positive (label = 1)

	Accuracy	**F1**
Random [uniform]	0.5	0.02
Random [label distribution]	0.98	0.01
Most common [preds = [0] * n]	?	?
Simple heuristics	?	?
Human expert	?	?
3rd party API	?	?

Evaluation methods

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

Perturbation tests

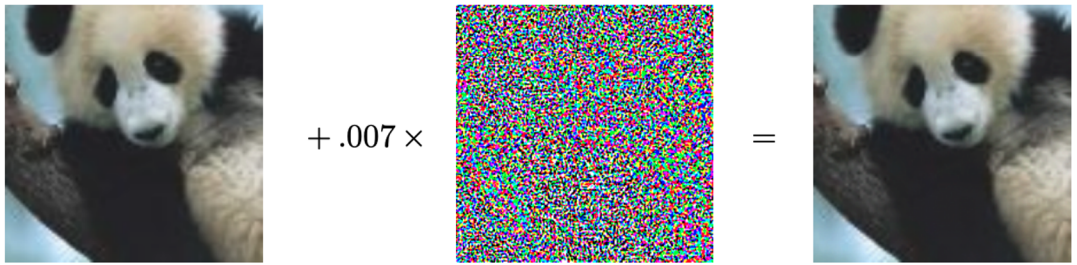
- Problem: users input might contain noise, making it different from test data
  - Examples:
    - Speech recognition: background noise
    - Object detection: different lighting
    - Text inputs: typos, intentional misspelling (e.g. looooooooong)
  - Model does well on test set, but fails in production

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

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



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

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.

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!

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

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!

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.”

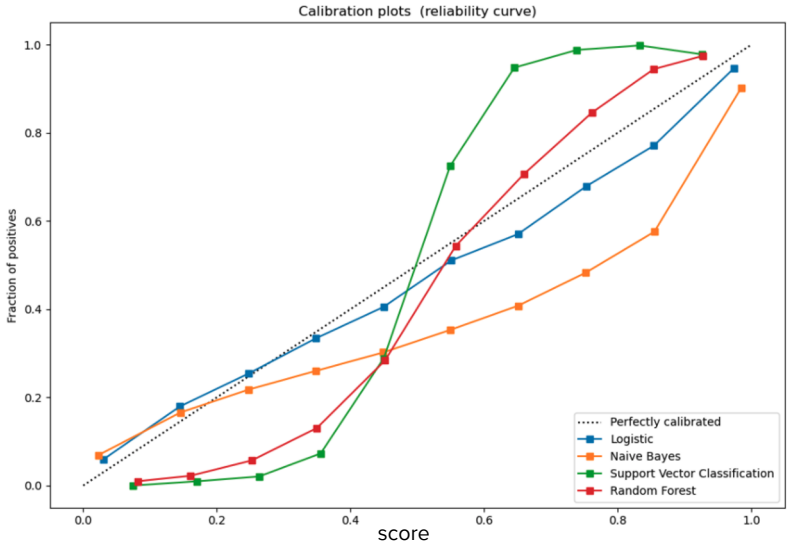
Nate Silver, **The Signal and the Noise**

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!

Model calibration: binary 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**

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

Model calibration: recsys

- Recommend movies to a user who watches 70% comedy, 30% romance
- What happens if you recommend most likely watched movies?

Movie title	**Watch probability**
Comedy 1	0.8
Comedy 2	0.73
Comedy 3	0.68
Comedy 4	0.67
Romance 1	0.29
Romance 2	0.2
Science fiction	0.04

Model calibration: recsys

- Recommend movies to a user who watches 70% comedy, 30% action
- What happens if you recommend most likely watched movies?

Need to calibrate recommendations to include 70% comedy, 30% action

Movie title	Watch probability
Comedy 1	0.8
Comedy 2	0.73
Comedy 3	0.68
Comedy 4	0.67
Action 1	0.29
Action 2	0.2
Science fiction	0.04

Model calibration: CTR

- 2 ads: A & B
- Model predicts click probability: A (10%), B (8%)
- How to estimate number of clicks you'll actually get if model isn't calibrated?

Confidence measurement

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

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

Slice-based evaluation

Different performance on different slices

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

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 ⚠️

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%

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%

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

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)**

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?

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 ⚠️

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

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

How to identify slices?

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

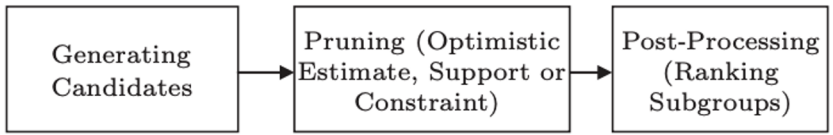


Fig.1. Methodology for subgroup discovery.

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

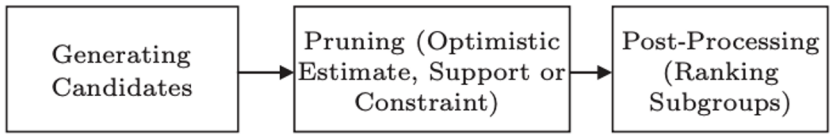


Fig.1. Methodology for subgroup discovery.

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

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