Lecture 11: Ensembles

Announcements

- Final Exam Dates

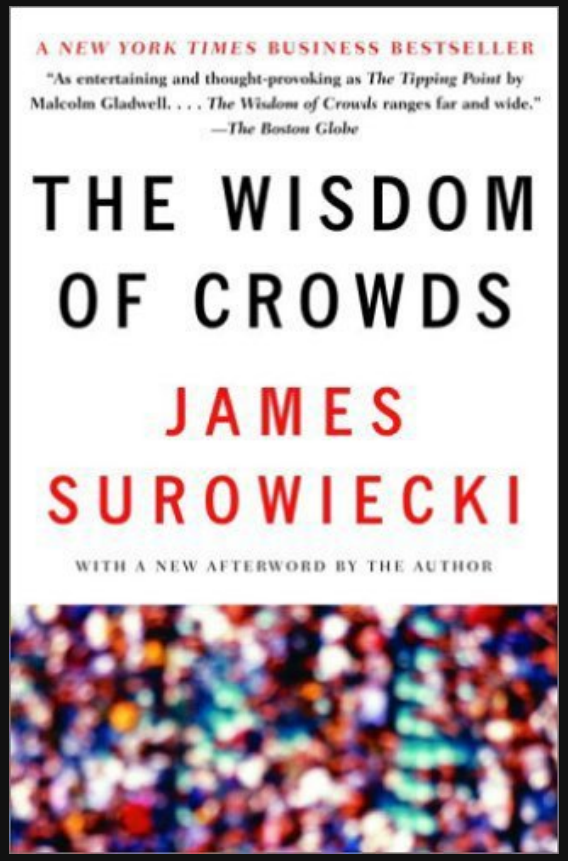
iClicker Exercise 11.0

Which of the following scenarios has worked effectively for you in the past?

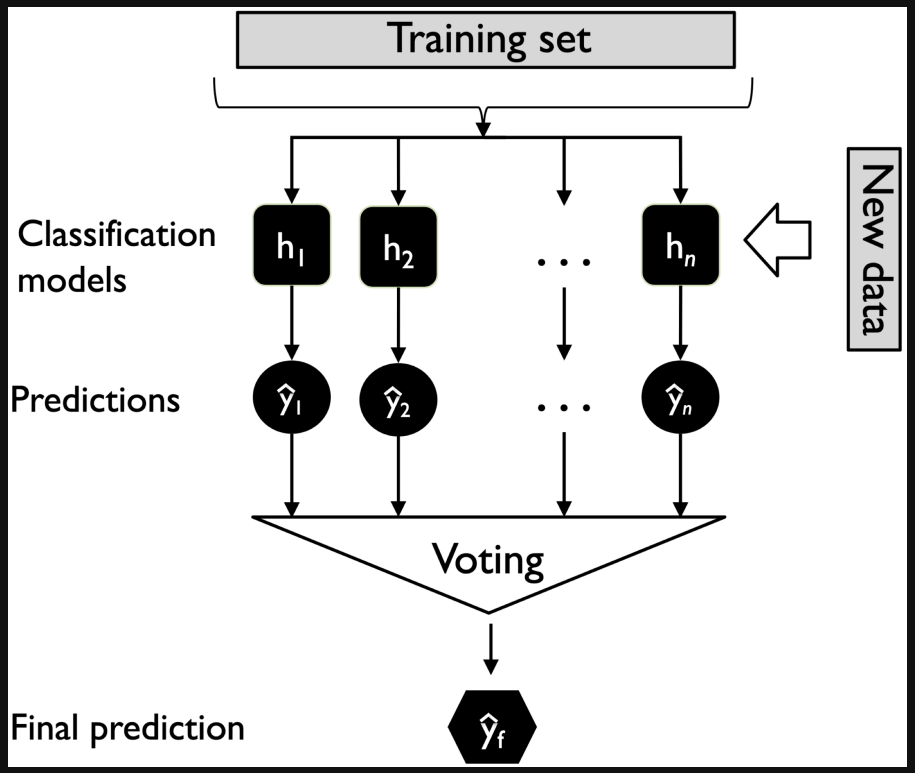
- a. Working independently on a project/assignment.
- b. Working with like-minded people.
- c. Teaming up with a diverse group offering varied perspectives and skills.

The Wisdom of Crowds

Groups can often make better decisions than individuals, especially when group members are diverse enough.



Ensembles key idea



- Combine predictions from multiple models for a more accurate and stable result
- Classification: either take majority class or average predicted probabilities
- Regression: Average (mean or median) predictions
- Averaging reduces individual model errors, making the ensemble more robust than any single model

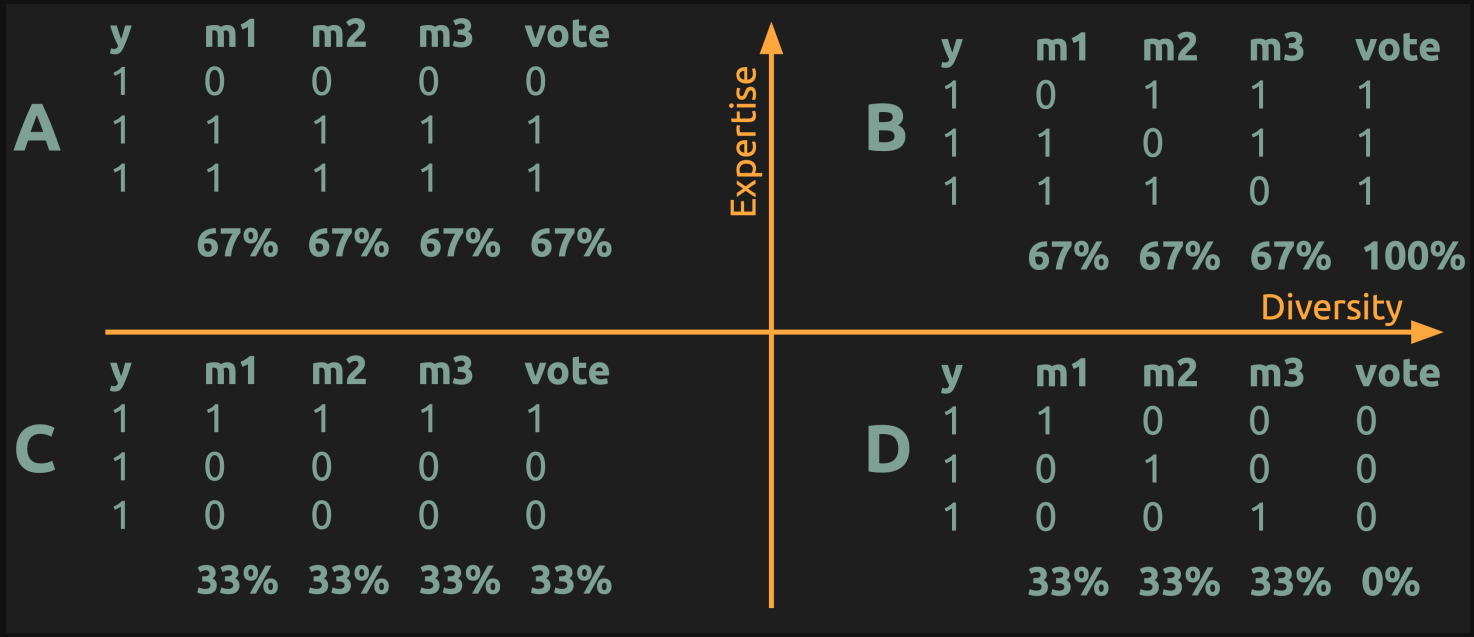
Source

Ensembles

- Ensemble methods are **widely used in industry** and dominate **machine learning competitions** (e.g., Kaggle).
- In this course, we'll explore
- **Tree-based ensembles:**
  - Random Forests
  - Gradient Boosted Trees (at a high level)
- **Ways to combine models:**
  - **Averaging:** combine model outputs directly
  - **Stacking:** use one model to learn how to combine others

When do ensembles help?

In which of these scenarios does an ensemble improve performance?



Source

When do ensembles help?

Ensembles improve performance when the individual models are both **competent** (have expertise) and **make different kinds of errors** (are diverse).

Diversity

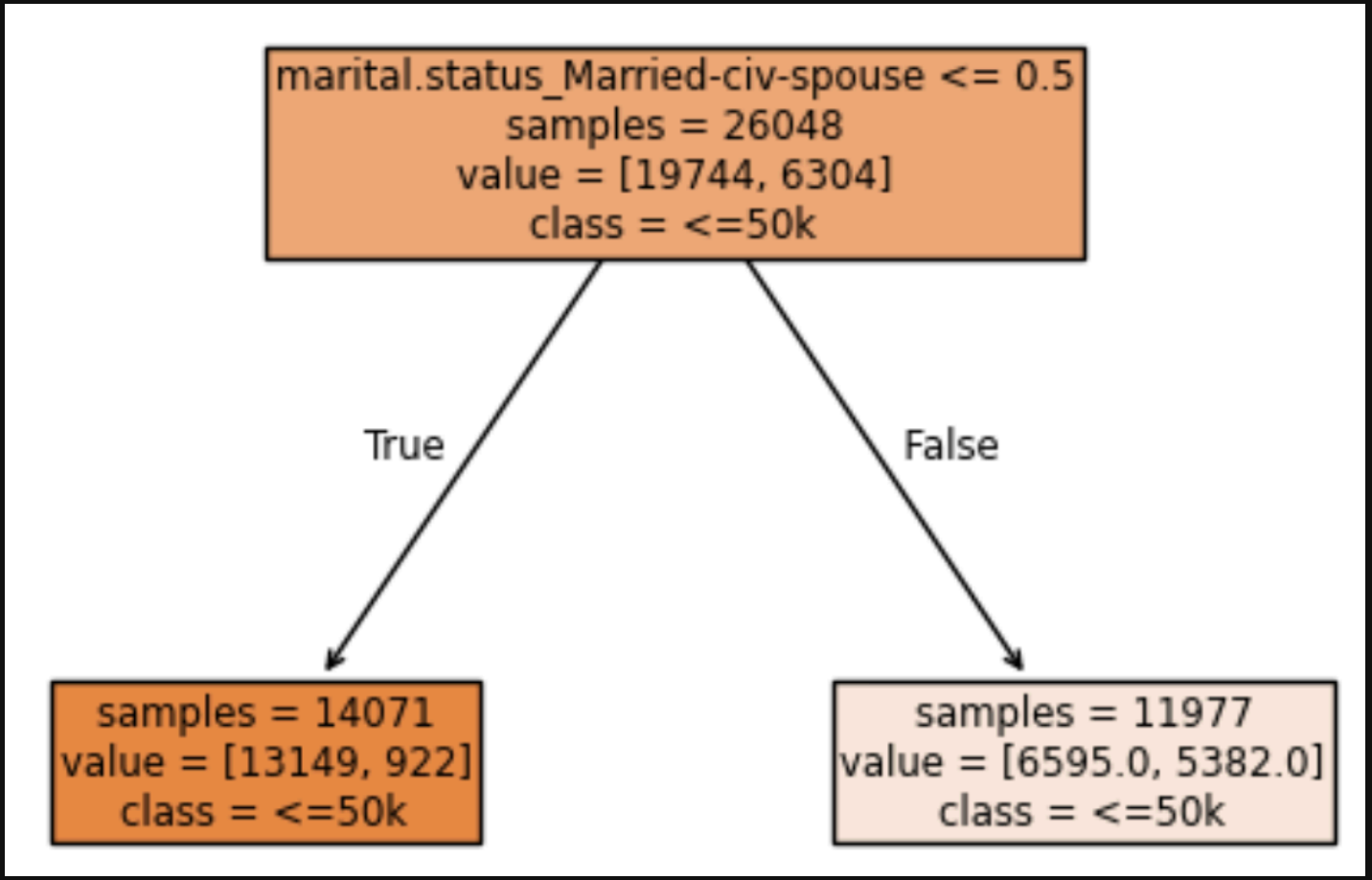
- Ensembles work best when their models make **different types of mistakes**.
- If all models make the same errors, averaging won't help.
- We can **encourage diversity** by:
  - Introducing randomness (e.g., bootstrapped samples and random feature subsets (*Random Forests*))
  - Sequentially focusing on previous errors (e.g., *Gradient Boosting*)

Expertise

- Diversity alone is not enough. Individual models should still have **some predictive skill**.
- If each model performs poorly (worse than random), the ensemble can't recover.
- Best ensembles strike a **balance** between:
  - **Diversity:** models disagree in useful ways
  - **Expertise:** each model performs better than chance

Random forest

What can we randomize between trees?



Two sources of randomness in random forests

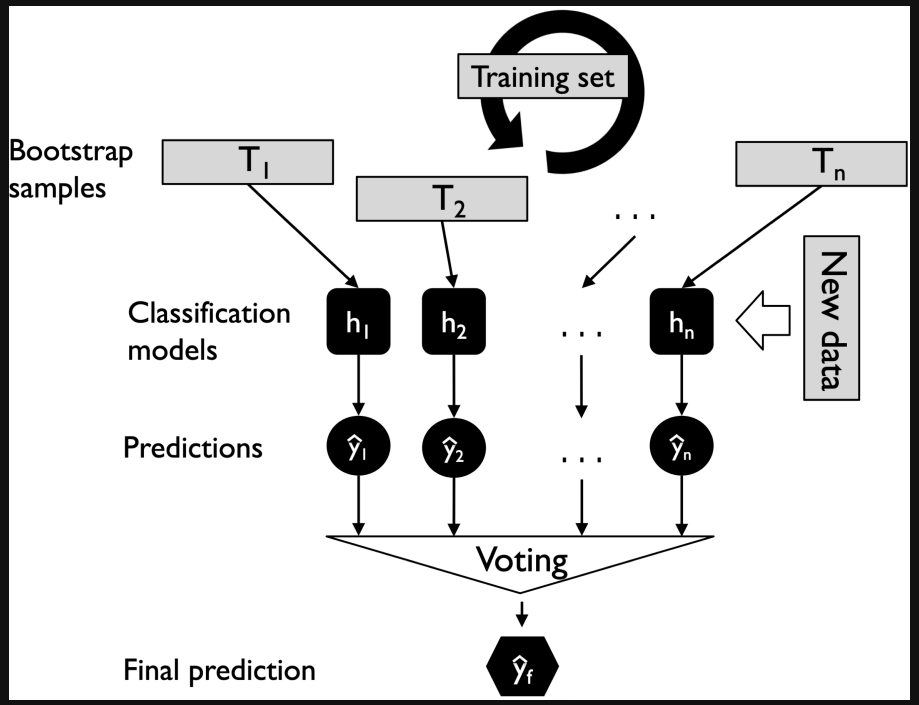
- **Bootstrap sampling of data**
  Each tree is trained on a slightly different dataset → decorrelates trees.
- **Random subset of features at each split**
  Each split considers only a random subset of features → further reduces correlation.

Bootstrap sample

- Ensures trees **see different versions of the data**
- Creates **diversity** while preserving **similar overall structure**

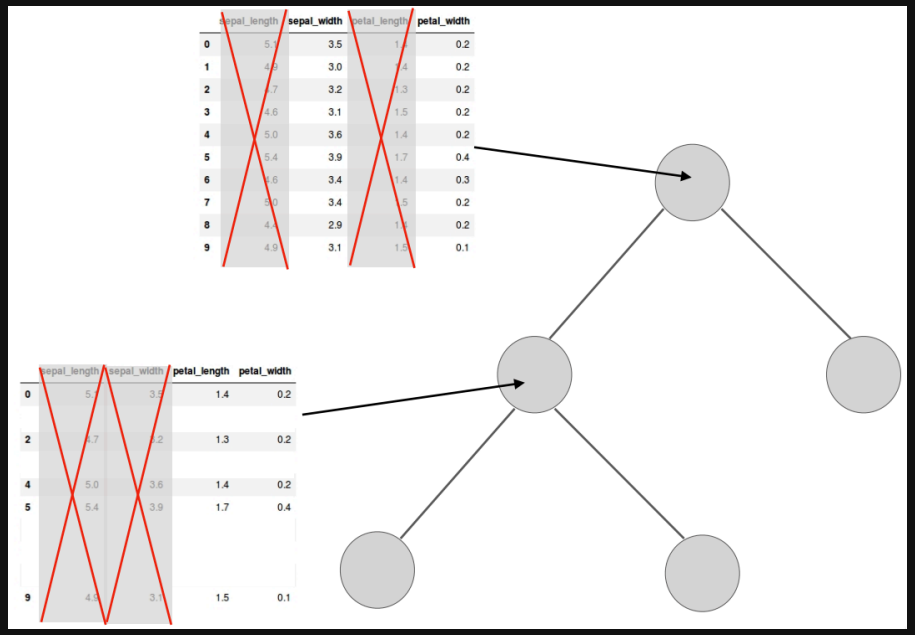
Training example indices	Bootstrap sample 1	Bootstrap sample 2	…
1	2	7	…
2	2	3	…
3	1	2	…
4	3	1	…
5	7	1	…
6	2	7	…
7	4	7	…

h_1 h_2 h_n



Source

Random subset of features at each split



- Select a different subset of features at every split
- Each tree now explores different parts of the feature space
- 🤔 But why not pick one subset per tree instead?

Source

Why use a new subset at each split?

Strategy	Diversity	Expertise	Issue
One subset per tree	Very high	Often low	May miss important features
New subset per split	Moderate	High	Stronger, less correlated trees

- Using a new subset per split strikes a **balance** between diversity and strength
- Prevents trees from becoming too weak while still keeping them different

> "Random selection of features at each node gives substantial additional accuracy gains over bagging alone."

Interim summary

- Random Forests reduce overfitting by **decorrelating trees**
- Two ingredients make this happen: Random data (bootstrapping) and random features (per split)
- Together, they build a strong yet diverse ensemble

iClicker Exercise 11.1

Select the most accurate option below.

- a. Every tree in a random forest uses a different bootstrap sample of the training set.
- b. To train a tree in a random forest, we first randomly select a subset of features. The tree is then restricted to only using those features.
- c. The `n_estimators` hyperparameter of random forests should be tuned to get a better performance on the validation or test data.
- d. In random forests we build trees in a sequential fashion, where the current tree is dependent upon the previous tree.
- e. Let classifiers A, B, and C have training errors of 10%, 20%, and 30%, respectively. Then, the best possible training error from averaging A, B and C is 10%.

Gradient boosting (high level)

From random forests to boosting

- In **random forests**, we build trees **independently** and **average** their predictions
  - reduces **overfitting** by combining diverse models
- In **boosting**, we build trees **sequentially**
  - each new tree **learns from the mistakes** of the previous ones
  - reduces **underfitting** by making the model smarter over time

Why it works

- **Sequential correction:** each tree improves on what's left of the error
- **Bias reduction:** starts simple and **adds complexity gradually**
- **Controlled learning:** small trees and a learning rate prevent overfitting

Most commonly used boosting models

- **XGBoost:** extremely optimized and widely adopted in industry
- **LightGBM:** faster training with large datasets
- **CatBoost:** handles categorical features efficiently

> *"More recently, gradient boosting machines (GBMs) have become a Swiss army knife in many a Kaggler's toolbelt."*
> — Sebastian Raschka, Joshua Patterson, & Corey Nolet (2020)

Averaging and Stacking

Goal: Combining diverse models for better generalization

Averaging (Voting)

- Combines predictions from **multiple base models**
- Works for both **classification** and **regression**
- Improves stability and performance by averaging over models
- Random forest is a type of averaging classifier

Types of Voting

Hard Voting

- Each model votes for a class label
- Final prediction = **majority vote**

Soft Voting

- Averages the **predicted probabilities** from each model
- Final prediction = class with **highest average probability**
- Usually performs better

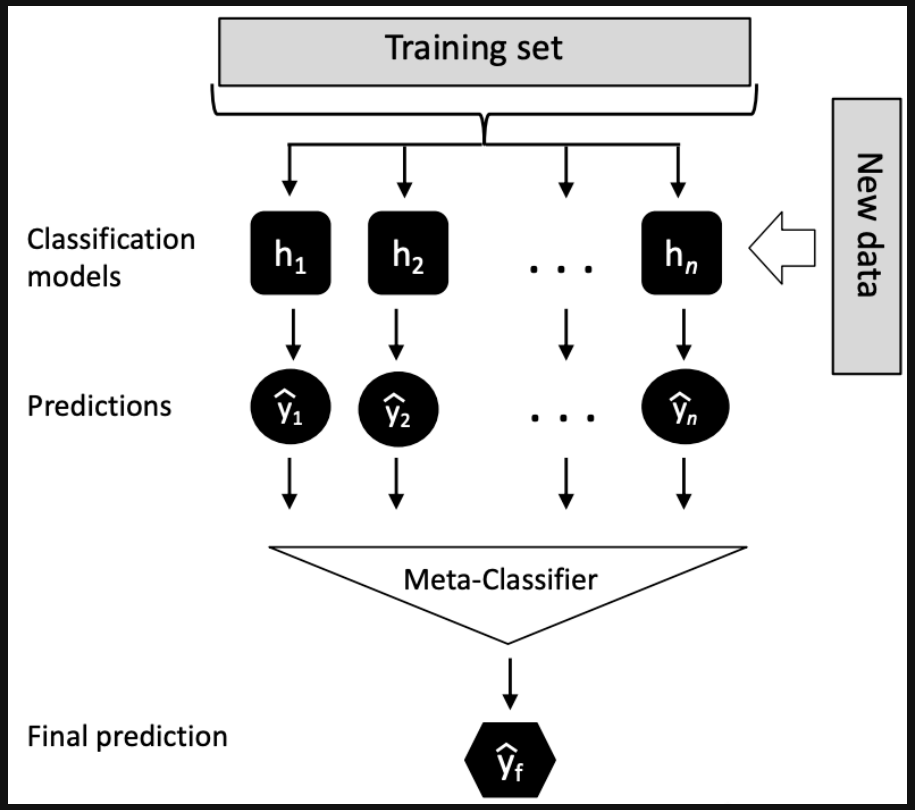
Stacking

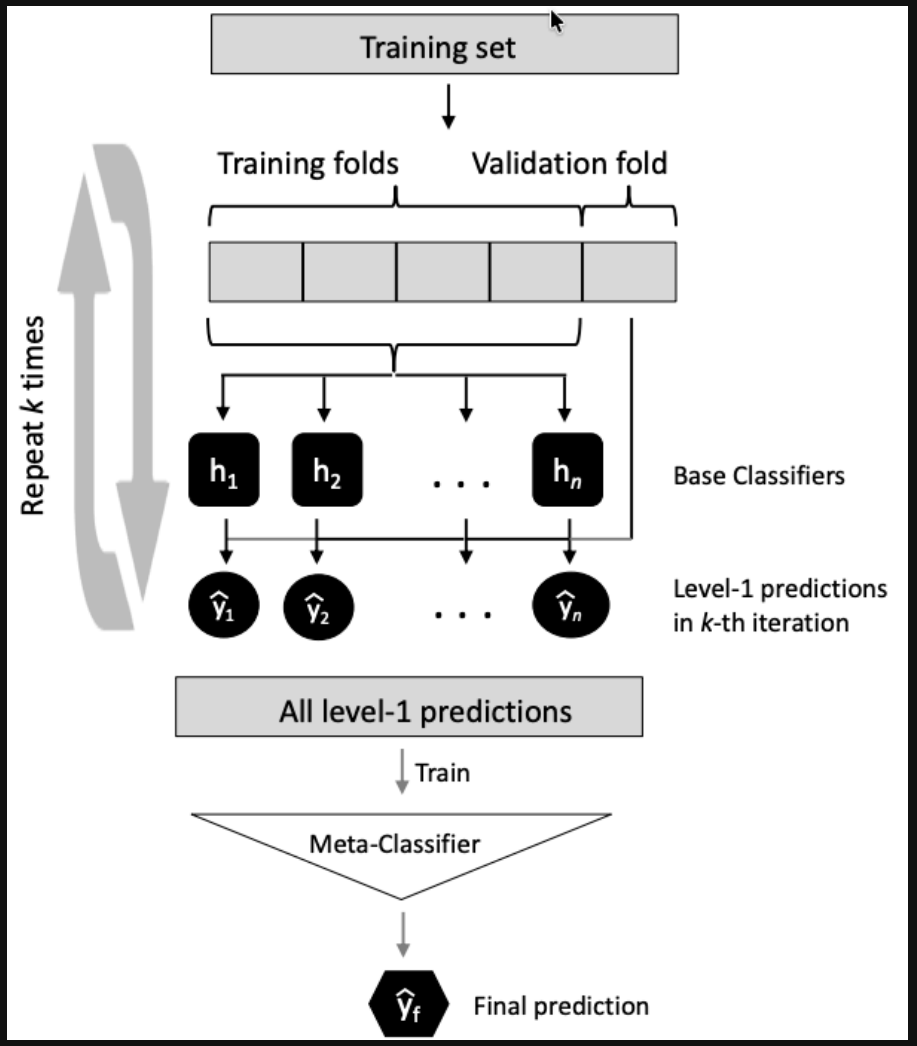
- Goes one step beyond averaging
- Instead of simple averaging, trains a meta-model to learn how to best combine base models' predictions

How stacking works

- Train several base models on the training data
- Collect their predictions on a validation set
- Train a meta-model on those predictions to learn the optimal combination
- Final prediction = output of the meta-model

Stacking

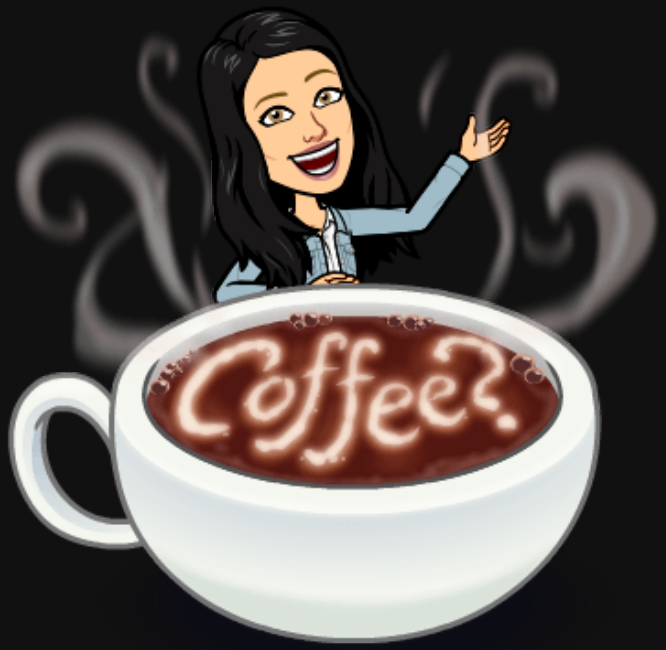




Source

Break

Let’s take a break!



/

Group Work: Class Demo & Live Coding

For this demo, each student should click this link to create a new repo in their accounts, then clone that repo locally to follow along with the demo from today.