Week 11:
Ensemble Learning Models

CMPS 320 : Machine Learning

Intro

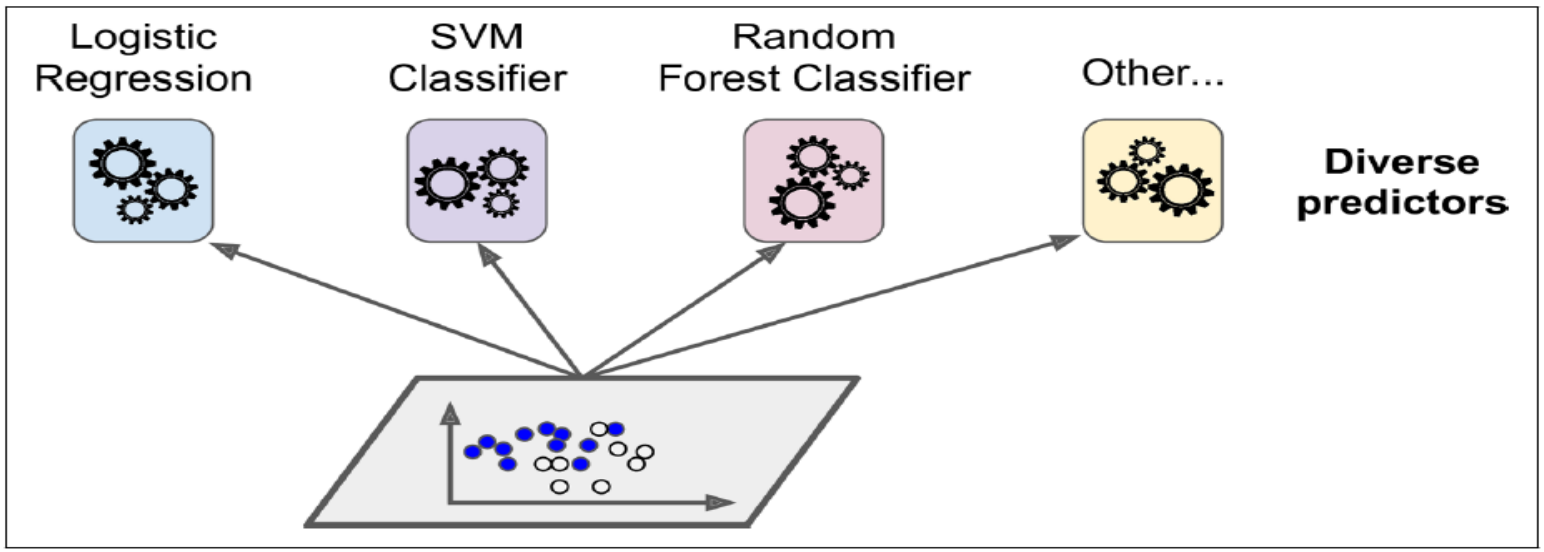
- Suppose you pose a complex question to thousands of random people, then aggregate their answers.
- In many cases you will find that this aggregated answer is better than an expert's answer.
  - This is called the **wisdom of the crowd**.
- Similarly, aggregating the predictions of a group of predictors (such as classifiers or regressors), will often produce better predictions than with the best individual predictor.
- A group of predictors is called an **ensemble**; thus, this technique is called **Ensemble Learning**.
  - an Ensemble Learning algorithm is called an Ensemble method.

Intro- Cont.

- An example of an Ensemble method is training a group of Decision Tree classifiers each on a different random subset of the training set.
- To make predictions, you obtain the predictions of all the individual trees, then predict the class that gets the most votes.
- Such an ensemble of Decision Trees is called a **Random Forest**.
- In today's lectures we will discuss the most popular Ensemble methods, including
  - Bagging,
  - Random Forest, and
  - Boosting

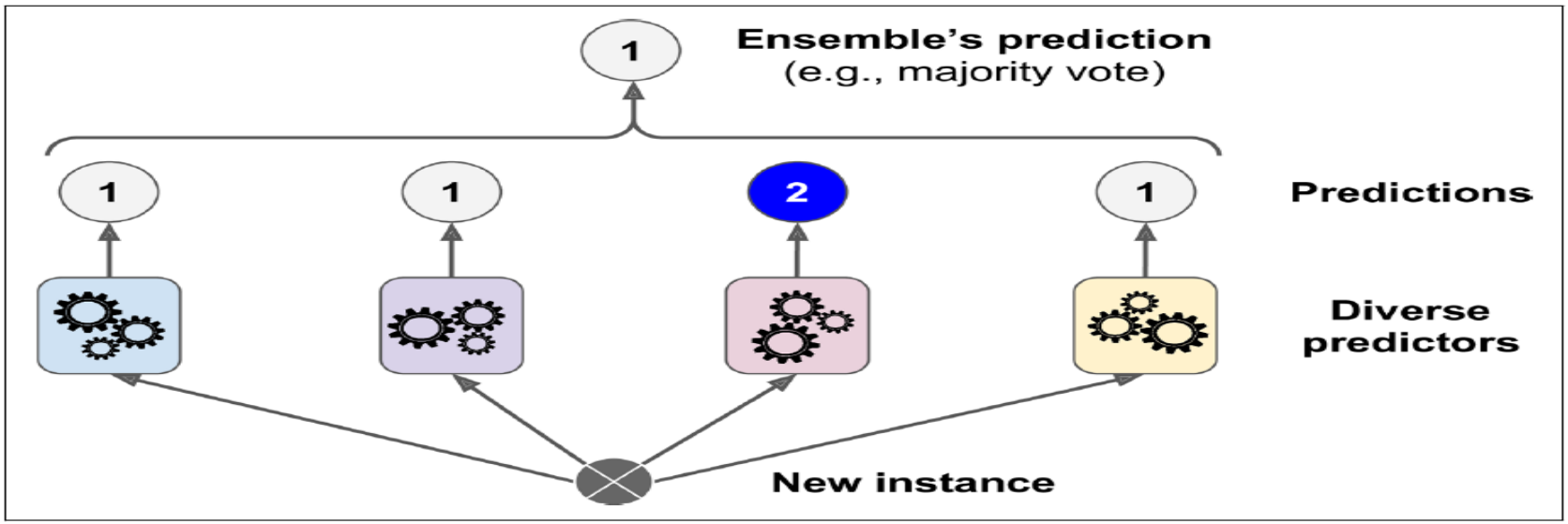
Voting Classifiers

- Suppose you have trained a few classifiers, each one achieving about 80% accuracy.
- We may have a Logistic Regression classifier, an SVM classifier, a Random Forest classifier, a K-Nearest Neighbors classifier etc.



Voting Classifiers- Cont.

- A simple way to create a better classifier is to aggregate the predictions of each classifier and predict the class that gets the most votes.
  - This majority-vote classifier is called a **hard voting** classifier.
- This voting classifier often achieves a higher accuracy than the best classifier in the ensemble.



Voting Classifiers- Cont.

- How is this possible?
- Suppose you have a biased coin that has a 51% chance of coming up heads and 49% chance of coming up tails.
- If you toss it 1,000 times, you will generally get approximately 510 heads and 490 tails, and hence a majority of heads.
- The probability of obtaining a majority of heads after 1,000 tosses is close to 75%.
- In fact, the more you toss the coin, the higher the probability (e.g., with 10,000 tosses, the probability climbs over 97%).
- This is due to the law of large numbers: as you keep tossing the coin, the ratio of heads gets closer and closer to the probability of heads (51%).

Voting Classifiers- Cont.

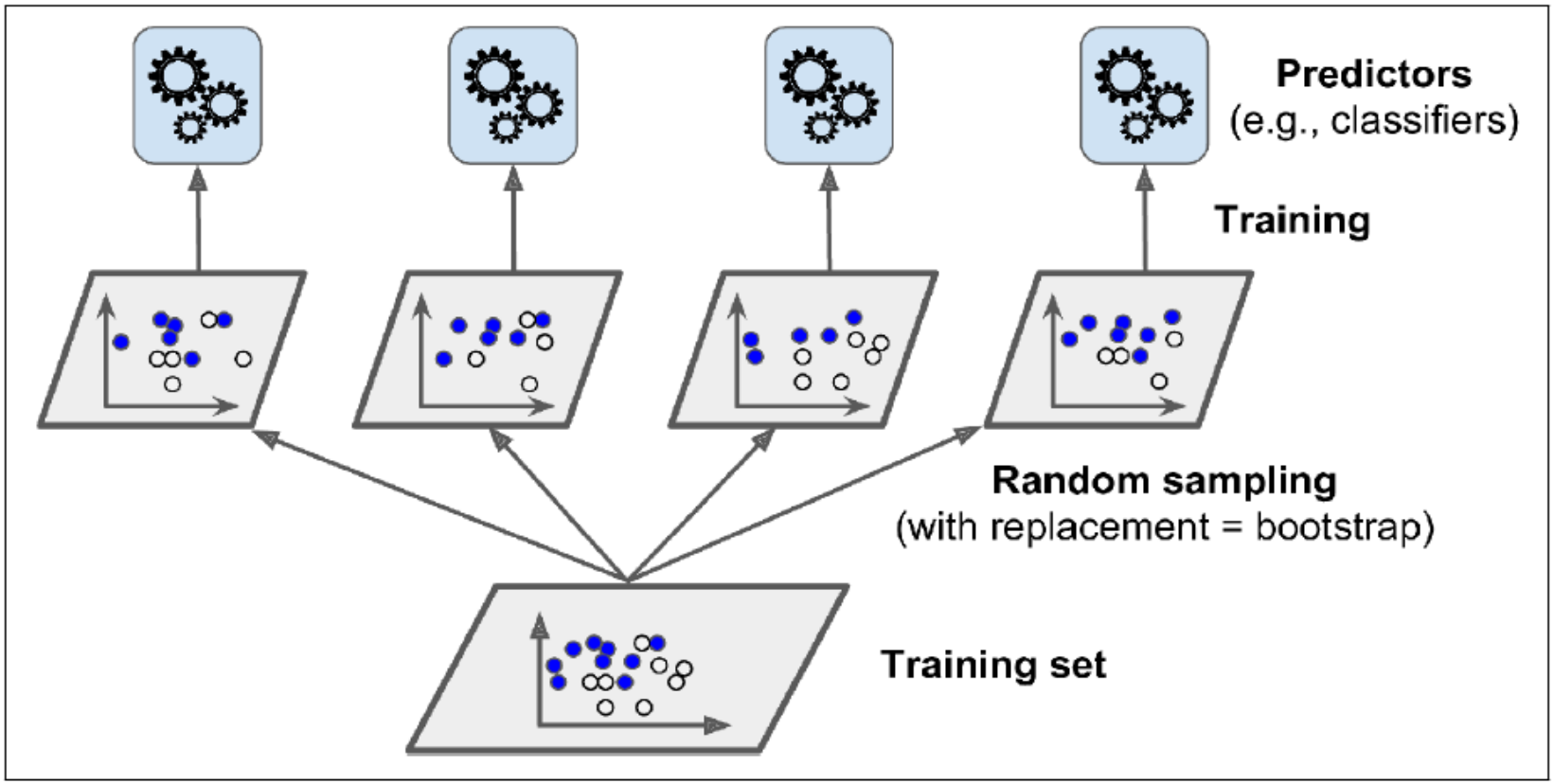
- Similarly, suppose you build an ensemble containing 1,000 classifiers that are individually correct only 51% of the time.
- If you predict the majority voted class, you can hope for up to 75% accuracy.
  - This is because the probability of obtaining a majority of heads after 1,000 tosses is close to 75%.
- This is true if all classifiers are perfectly independent, making uncorrelated errors
- Note:
  - Ensemble methods work best when the predictors are independent from one another

Bagging and Pasting

- One way to get a diverse set of classifiers is to use very different training algorithms.
- Another approach is to use the same training algorithm for every predictor and train them on different random subsets of the training set.
- When sampling is performed with replacement, this method is called **bagging** (bootstrap aggregating).
- When sampling is performed without replacement, it is called **pasting**.
- Bagging and pasting allow training instances to be sampled several times across multiple predictors.
  - Only bagging allows training instances to be sampled several times for the same predictor.

Bagging and Pasting Cont.

- Bagging and pasting involves training several predictors on different random samples of the training set



Bagging and Pasting Cont.

- Once all predictors are trained, the ensemble can make a prediction for a new instance by simply aggregating the predictions of all predictors.
- The aggregation function is typically:
  - the statistical mode (i.e., the most frequent prediction, just like a hard voting classifier) for classification or
  - the average for regression.
- Each individual predictor has a higher bias than if it were trained on the original training set, but aggregation reduces both bias and variance.
- The net result is that the ensemble has a similar bias but a lower variance than a single predictor trained on the original training set.
- The predictors can all be trained in parallel, via different CPU cores or even different servers:
  - Predictions can be made in parallel.
  - They scale very well.

Random Forest

- The **Random Forest** algorithm introduces extra randomness when growing trees:
  - Instead of searching for the very best feature when splitting a node, it searches for the best feature among a random subset of features.
- The algorithm results in greater tree diversity, which trades a higher bias for a lower variance, generally yielding an overall better model.
- Yet another great quality of Random Forests is that they make it easy to measure the relative importance of each feature.
  - Scikit-Learn measures a feature's importance by looking at how much the tree nodes that use that feature reduce impurity on average (across all trees in the forest).

Boosting

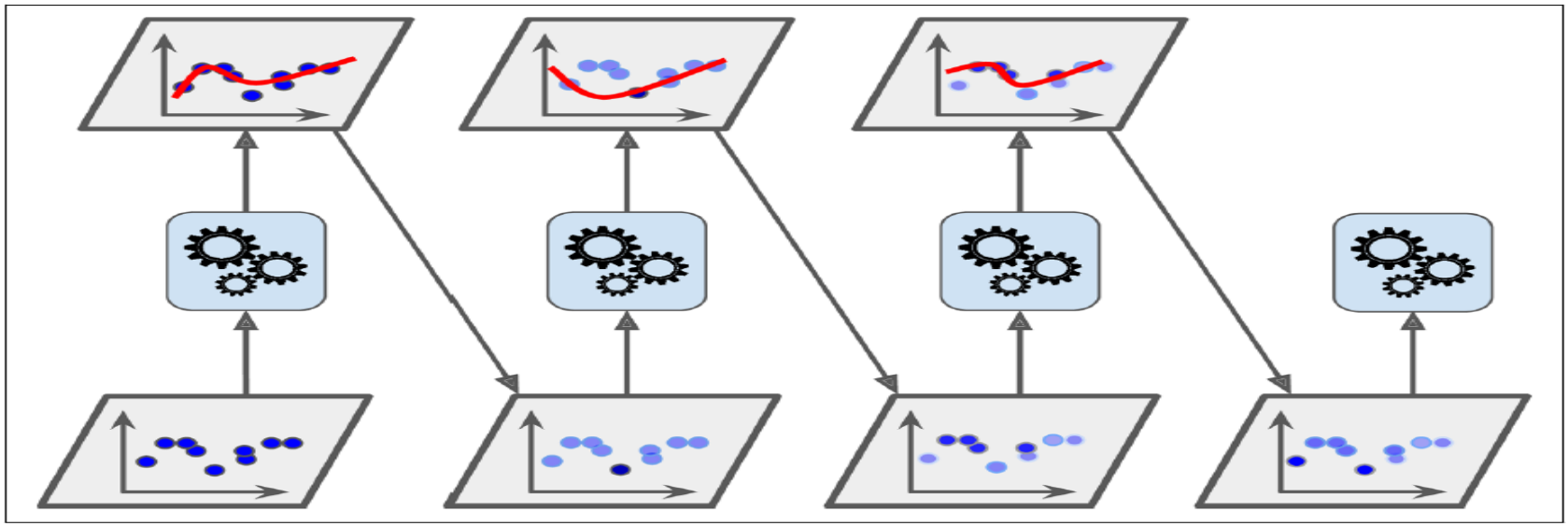
- **Boosting** (originally called hypothesis boosting) refers to any Ensemble method that can combine several weak learners into a strong learner.
- The general idea of most boosting methods is to train predictors sequentially, each trying to correct its predecessor.
- There are many boosting methods available:
  - AdaBoost (Adaptive Boosting)
  - Gradient Boosting

AdaBoost

- One way for a new predictor to correct its predecessor is to pay a bit more attention to the training instances that the predecessor underfitted.
- This results in new predictors focusing more and more on the hard cases.
  - This is the technique used by AdaBoost.

AdaBoost- Cont.

- To train an AdaBoost classifier, the algorithm first trains a base classifier (such as a Decision Tree) and uses it to make predictions on the training set.
  - The algorithm then increases the relative weight of misclassified training instances.
- Then it trains a second classifier, using the updated weights, and again makes predictions on the training set, updates the instance weights, and so on...



AdaBoost- Cont.

- Once all predictors are trained, the ensemble makes predictions very much like bagging or pasting, except that predictors have different weights depending on their overall accuracy on the weighted training set.
- One important drawback to this sequential learning technique:
  - It cannot be parallelized (or only partially), since each predictor can only be trained after the previous predictor has been trained and evaluated
  - It does not scale as well as bagging or pasting.

Gradient Boosting

- Gradient Boosting works by sequentially adding predictors to an ensemble, each one correcting its predecessor.
- However, instead of tweaking the instance weights at every iteration like AdaBoost does, this method tries to fit the new predictor to the residual errors made by the previous predictor.