# COMPSCI 589
# Lecture 12: Ensemble Methods

Benjamin M. Marlin

College of Information and Computer Sciences
University of Massachusetts Amherst

Slides by Benjamin M. Marlin (marlin@cs.umass.edu).
Created with support from National Science Foundation Award# IIS-1350522.

# Ensembles

- An *ensemble* is simply a collection of models that are all trained to perform the same task.
- An ensemble can consist of many different versions of the same model, or many different types of models.
- The final output for an ensemble is typically obtained through a (weighted) average or vote of the predictions of the different models in the ensemble.
- An ensemble of different models that all achieve similar generalization performance often outperforms any of the individual models.
- **Question:** How is this possible?

# Ensemble Intuition for Regression

- Suppose we have an ensemble of regression functions $f_k(\mathbf{x})$ for $k = 1, ..., K$.
- Suppose that on average they have the same expected MSE $\epsilon = E_{p(x,y)}[(y - f_k(\mathbf{x}))^2]$, but that the errors they make are *independent*.
- The intuition is that the average of the models $\frac{1}{K}\sum_{k=1}^{K} f_k(\mathbf{x})$ can perform significantly better than any single model because the errors the individual models make will tend to cancel out.
- Again, a simple average can significantly improve regression performance by *decreasing variance*.
- **Question:** How can we come up with such an ensemble?

# Ensemble Intuition for Classification

- Suppose we have an ensemble of binary classification functions $f_k(\mathbf{x})$ for $k = 1, ..., K$.
- Suppose that on average they have the same expected error rate $\epsilon = E_{p(x,y)}[y \neq f_k(\mathbf{x})] < 0.5$, but that the errors they make are *independent*.
- The intuition is that the majority of the $K$ classifiers in the ensemble will be correct on many examples where any individual classifier makes an error.
- A simple majority vote can significantly improve classification performance by *decreasing variance*.
- **Question:** How can we come up with such an ensemble?

# Independent Training Sets

- Suppose we collect multiple independent training sets $Tr_1, ..., Tr_K$ and use each of these training sets to train a different instance of the same model obtaining $K$ prediction functions $f_1(\mathbf{x}), ..., f_K(\mathbf{x})$.
- Models trained in this way are guaranteed to make independent errors on test data.
- For regression, if the expected errors of the individual regressors are approximately equal, independence will lead to a variance reduction and a corresponding reduction in expected error.
- In the classification setting, if the expected error of each classifier is less than 0.5, then the weighted majority vote is guaranteed to reduce the expected generalization error.

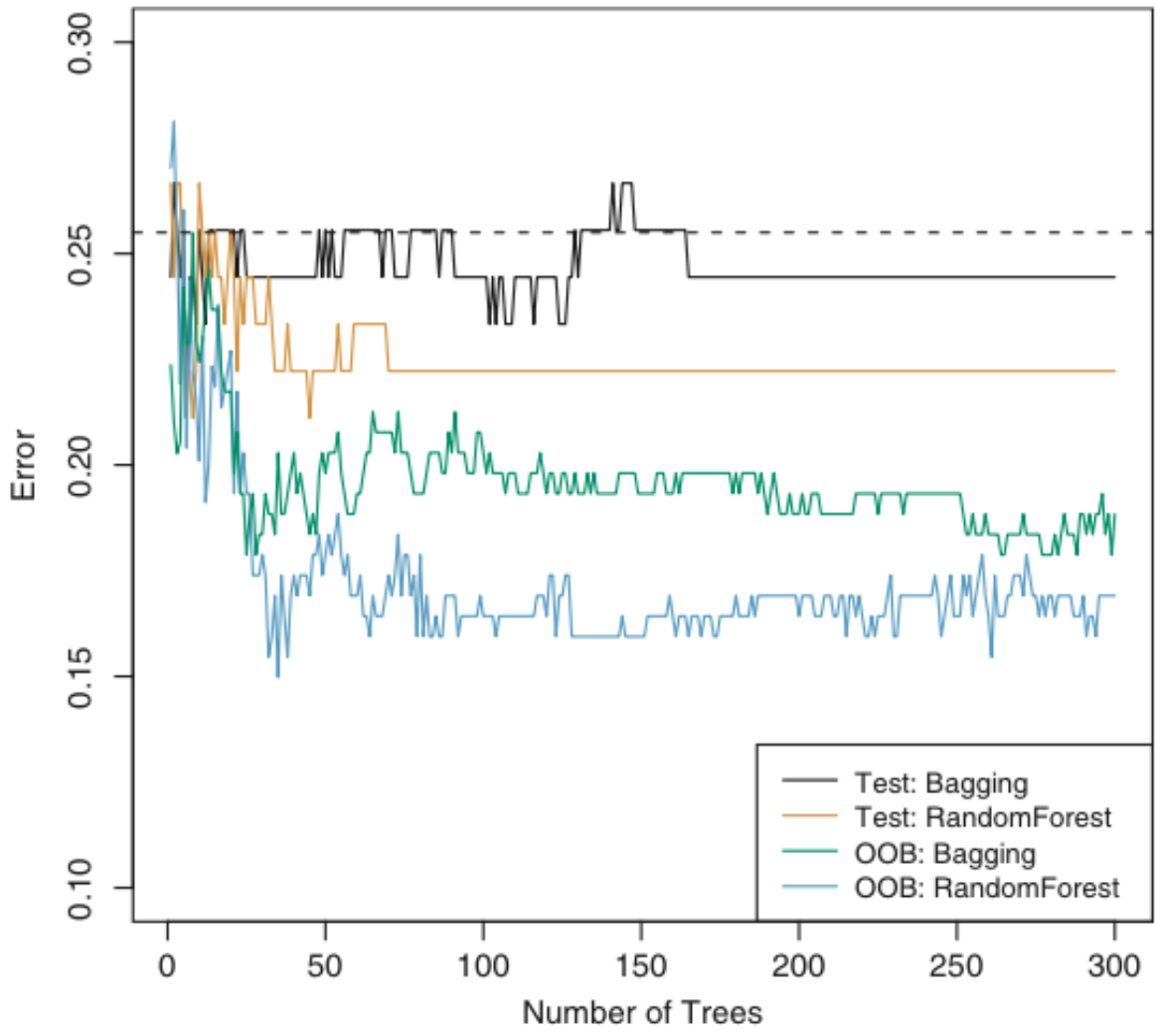
# Bagging

- Bootstrap aggregation or *Bagging* is an approximation to the previous method that takes a single training set $Tr$ and randomly re-samples it $K$ times (with replacement) to form $K$ training sets $Tr_1, ..., Tr_K$ of size equal to the original training set.
- Each of the re-sampled training sets contains about 63% of the unique instances from the original training set.
- Each of these training sets is used to train a different instance of the same model obtaining $K$ prediction functions $f_1(\mathbf{x}), ..., f_K(\mathbf{x})$.
- The errors won't by totally independent because the data sets aren't independent, but the random re-sampling usually introduces enough diversity to decrease variance and give improved performance.

# Bagging and Random Forests

- Bagging is particularly useful for high-variance, high-capacity models.
- Historically, it is most closely associated with decision tree models.
- A very successful extension of bagged trees is the random forest.
- The random forest algorithm further decorrelates the learned trees by only considering a random sub-set of the available features when deciding which variable to split on at each node in the tree.
- The same idea can be applied in the regression and classification settings.

# Example: Bagging vs Random Forests



# Boosting

- Boosting is an ensemble method based on adding new models to the ensemble sequentially to minimize error.
- The main idea is to identify data cases that currently result in errors in the ensemble, and then add a next model that will focus on the data cases that are causing the errors.
- In the regression setting, boosting non-linear models (like regression trees) can often improve predictive performance. Gradient boosting is a common approach that supports different loss functions.
- In the classification setting, assuming that the base classifier can always achieve an error rate of less than 0.5 on any data sample, algorithms such as Adaboost can be shown to decrease error.

# Gradient Boosting

**Input:** Training data $\mathcal{D}_{tr} = \{(\mathbf{x}_i, y_i) | 1 \leq i \leq N\}$, number of iterations $T$, learning rate $\epsilon$
**Initialize:** $f_0(x) \leftarrow \frac{1}{n} \sum_{i=1}^{n} y_i$
**for** $t = 1$ to $T$ **do**
    **for** $i = 1$ to $N$ **do**
        Compute residuals: $r_i^{(t)} \leftarrow y_i - f_{t-1}(\mathbf{x}_i)$
    **end for**
    Fit regression tree $h_t$ to $\{(\mathbf{x}_i, r_i^{(t)}) | 1 \leq i \leq N\}$
    Update model: $f_t \leftarrow f_{t-1} + \epsilon \cdot h_t$
**end for**
**Output:** Final model $f_T$

# Stacking (or Blending)

- Unlike bagging and boosting, stacking is an algorithm for combining several different types of models.
- The main idea is to form a train-validation-test split and train many different kinds of models $f_k(\mathbf{x})$ on the training data.
- The trained models are used to make predictions on the validation data set and a new feature representation is then created where each data case consists of the vector of predictions of each model in the ensemble $\tilde{\mathbf{x}} = [f_1(\mathbf{x}), ..., f_K(\mathbf{x})]$.
- Finally, a meta-classifier called a *combiner* is trained to minimize the validation error given the data $\{(y_i, \tilde{\mathbf{x}}_i) | i = 1, .., N\}$.
- The extra layer of combiner learning can deal with correlated classifiers as well as classifiers that perform poorly.

# Example: Netflix Prize (2009)


## Leaderboard

Showing Test Score. Click here to show quiz score

Display top 20 leaders.

| Rank | Team Name | Best Test Score | % Improvement | Best Submit Time |
|---|---|---|---|---|
| Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos | | | | |
| 1 | BellKor's Pragmatic Chaos | 0.8567 | 10.06 | 2009-07-26 18:18:28 |
| 2 | The Ensemble | 0.8567 | 10.06 | 2009-07-26 18:38:22 |
| 3 | Grand Prize Team | 0.8582 | 9.90 | 2009-07-10 21:24:40 |
| 4 | Opera Solutions and Vandelay United | 0.8588 | 9.84 | 2009-07-10 01:12:31 |
| 5 | Vandelay Industries ! | 0.8591 | 9.81 | 2009-07-10 00:32:20 |
| 6 | PragmaticTheory | 0.8594 | 9.77 | 2009-06-24 12:06:56 |
| 7 | BellKor in BigChaos | 0.8601 | 9.70 | 2009-05-13 08:14:09 |
| 8 | Dace | 0.8612 | 9.59 | 2009-07-24 17:18:43 |

Winning team used stacked predictor of 450+ different models.