Ensemble Learning (Bagging and Boosting)

Foundations of Machine Learning

(Sunita Sarawagi)

Motivation

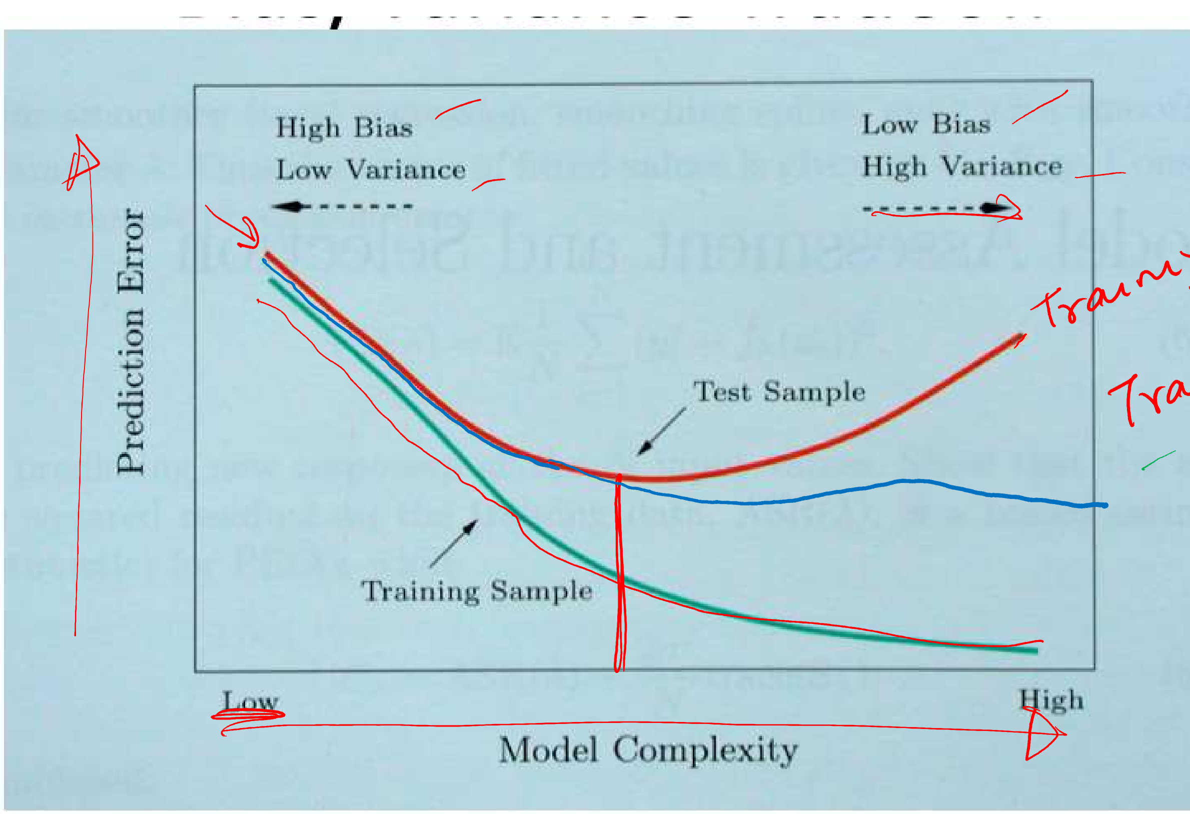
- Ensembling→Learning from more than one classifier
- A single classifier may not be powerful enough
  - Limited hypothesis class: example linear classifier, or decision tree of limited depth
- A single classifier may overfit
  - Decision tree without no limit on length
  - Neural network without regularizer

- Many competitions won because of ensembling!
- Ensembling continues to be useful even in the era of deep learning

Bias and Variance of a model class

- Bias:
  - Simplifying assumptions about the form of the model so as to be easy to learn.
  - Example: linear regression can only learn linear separators, naïve Bayes assumes conditional independence
  - Low bias classifiers: Kernel SVMs, nearest neighbor classifier, feed forward NN, decision trees,
  - High bias classifiers: linear regression, naïve Bayes, perceptron, in general any parameteric model has high bias. with small # of parameters
- Variance:
  - Change in accuracy of the model with changes in the training dataset.
  - For a high variance classifier the test accuracy will change a lot when we sample different N instances for training from the data distribution.
  - Examples of high variance classifier: Kernel SVMs, nearest neighbor classifier, feed forward NN with many units, unbounded depth decision tree
  - Examples of low variance classifier: linear regression, naïve Bayes, perceptron, small depth decision trees.

Bias variance tradeoff



Training sample 1 ← D_1
Training sample 2 ← D_2

Hastie, Tibshirani, Friedman "Elements of Statistical Learning" 2001

Bagging

- Accuracy from a single classifier may have high variance
- Bagging creates a committee of classifiers and averages the predictions from the committee to make the final prediction
- D = training set, m = number of classifiers or committee members
- Train M classifiers: $C_1(x)$, $C_2(x)$, … $C_m(x)$
- For a test instance, get m predictions $\widehat{y_1}, \ldots, \widehat{y_m}$
- Final prediction: majority label among the m predictions

D m classifiers

Committee members

$c_1 \quad c_2 \quad c_3 \quad - - - - - - \quad c_m$

$C_1(x) \quad C_2(x) \quad \ldots \quad C_m(x) \qquad \vec{x} \in \mathbb{R}^d$

$= \hat{y}_1 \quad = \hat{y}_2 \quad - - - \quad = \hat{y}_m$

Predicted label = majority $(\hat{y}_1, \ldots \hat{y}_m) = \hat{y}_B$

Why should bagging reduce error?

- Simple Example (Synthetic)

- Let p = probability that a classifier gives correct label on a x

Let $C_1, \dots, C_m$ be independent of each other.

What is the probability that $\hat{y}_B$ is correct?

$$= \boxed{\sum_{k=\frac{m}{2}}^{m} \binom{m}{k} p^{k} (1-p)^{m-k}} \overset{?}{>} p$$

For binary classifiers above is true as long as p > 0.5

$m = 2$

what is the probability that 1 or both classifiers are correct?

$C_1 \qquad C_2$

1) Prob that both are correct

p^2

$k = m$

2) $2p(1-p)$

$p^2 + 2p(1-p) > p$

$p^2 + 2p - 2p^2 - p > 0$

$-p^2 + p \geq 0$

$(\quad p >$

Methods of creating committee

Two goals during creation of committee members: each classifier should be as accurate as possible, the ~~predictions~~ errors from different classifiers should be as independent as possible.

- Bootstrap sampling
  - Create different training samples from the given training dataset D
- Random forests
  - Above + Create different random attribute subsets from D

Bagging by Bootstrap sampling

$D = \{ (x^1, y^1) \ldots (x^N, y^N) \}$

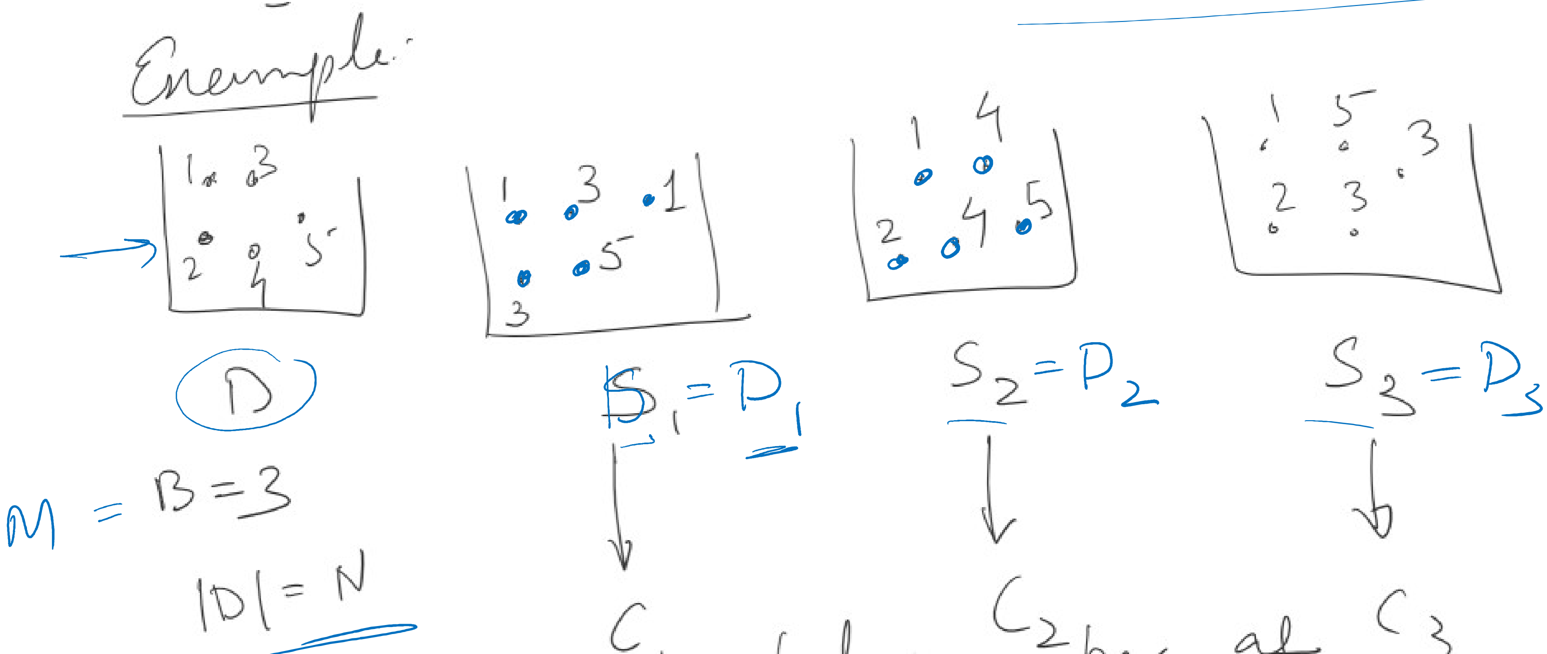
Convert this into a point distribution

$P_D(x,y) = \frac{1}{N}$ if $(x,y) \in D$

$= 0$ otherwise.

- For j = 1 to M
  - /* Create training set for Dj using bootstrap sampling as follows */
  - For i=1 to N
    - Sample an instance from D (uniformly from P_D) /* Sampling with replacement */
  - $C_j(x)$ = Train j-th committee member using training data Dj

Example of bootstrap samples for bagging



Expected number of distinct samples in any one bag?

N*Probability that an instance is selected in any of N trials

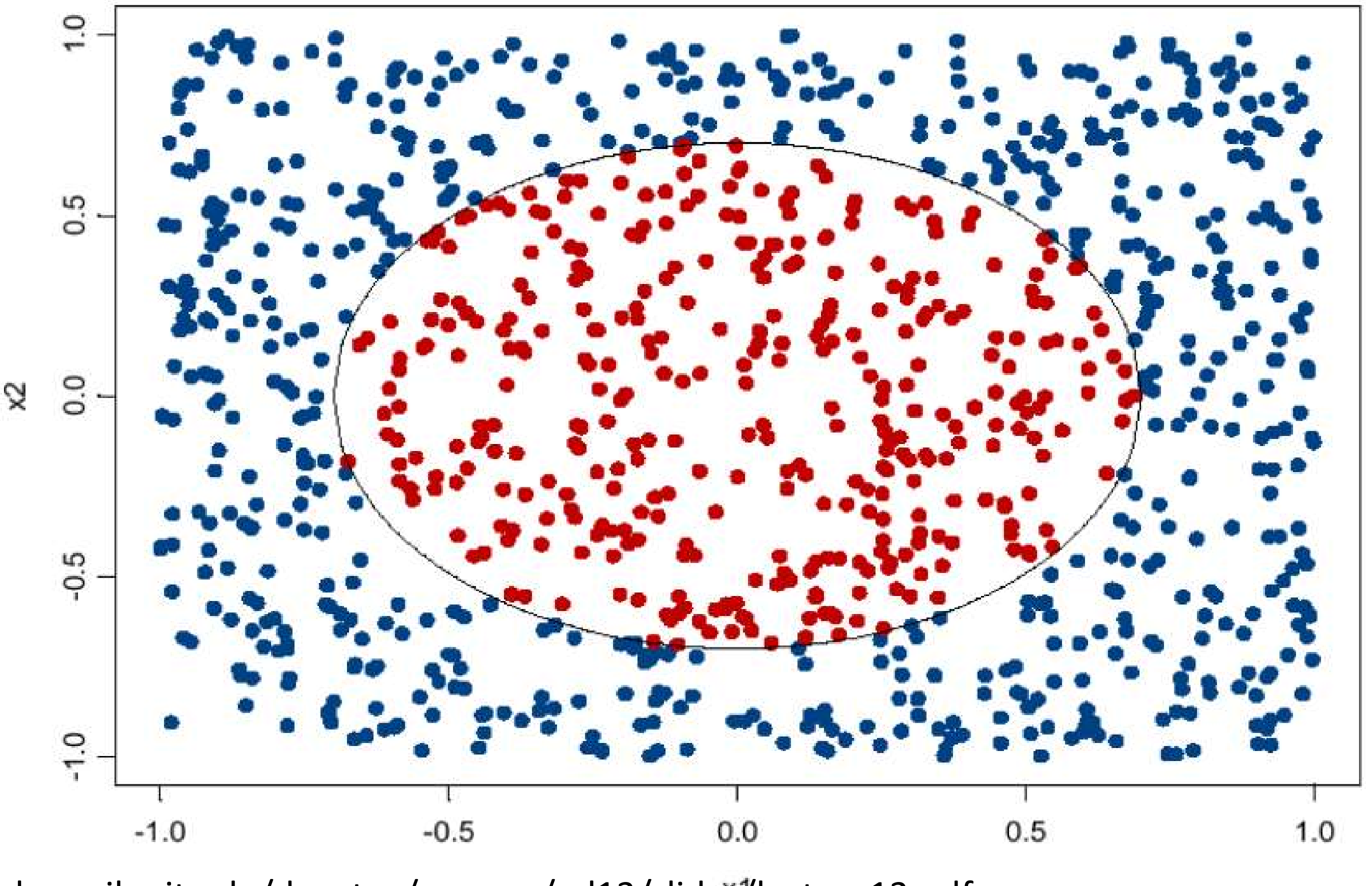
$$N\left(1-\left(1-\frac{1}{N}\right)^{N}\right) \xrightarrow[N\to\infty]{as} N\left(1-\frac{1}{e}\right) \approx 0.63N$$

2.7

Bagging

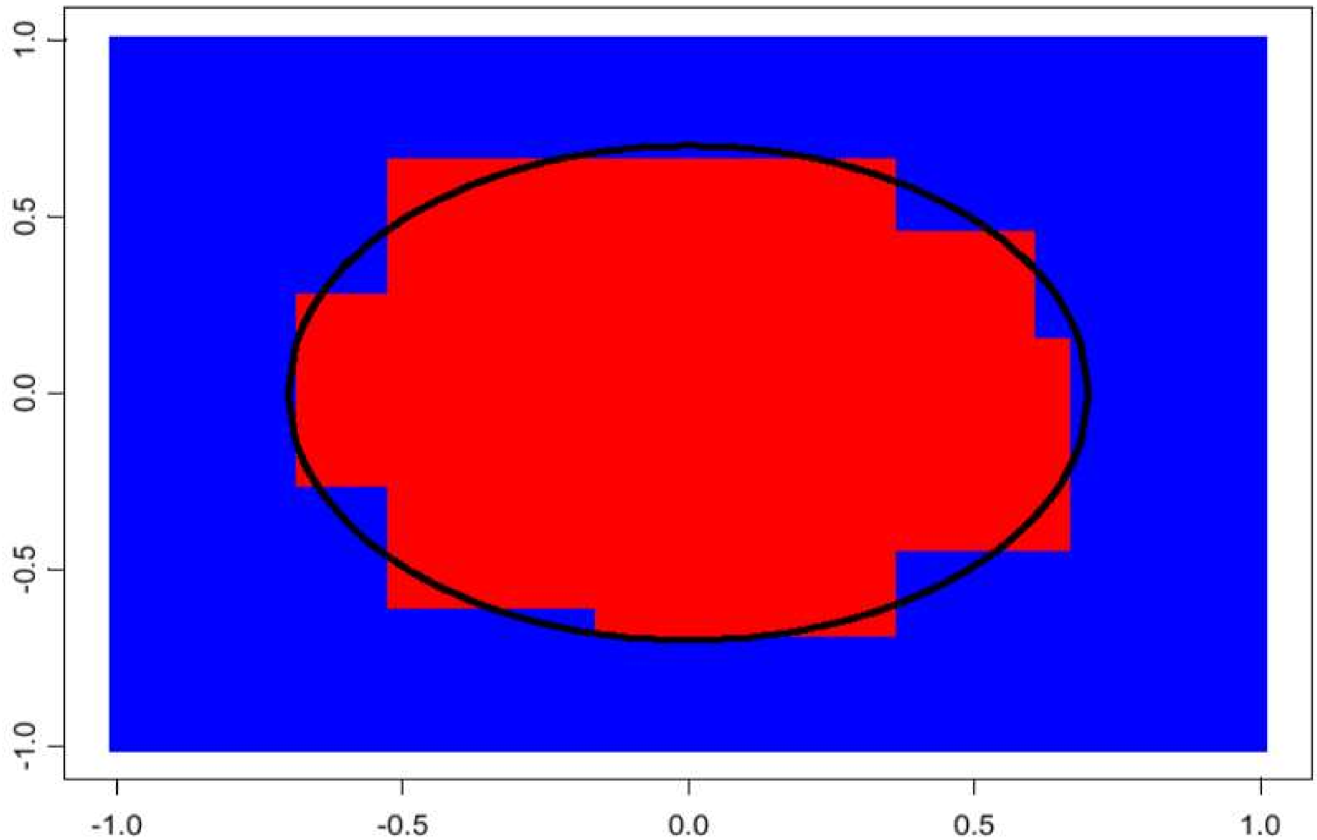
- Useful when individual classifiers are over-fitting, e.g. a decision tree without pruning.
- Bagging by averaging the predictions from multiple over-fitted trees reduces variance

Bagging Example

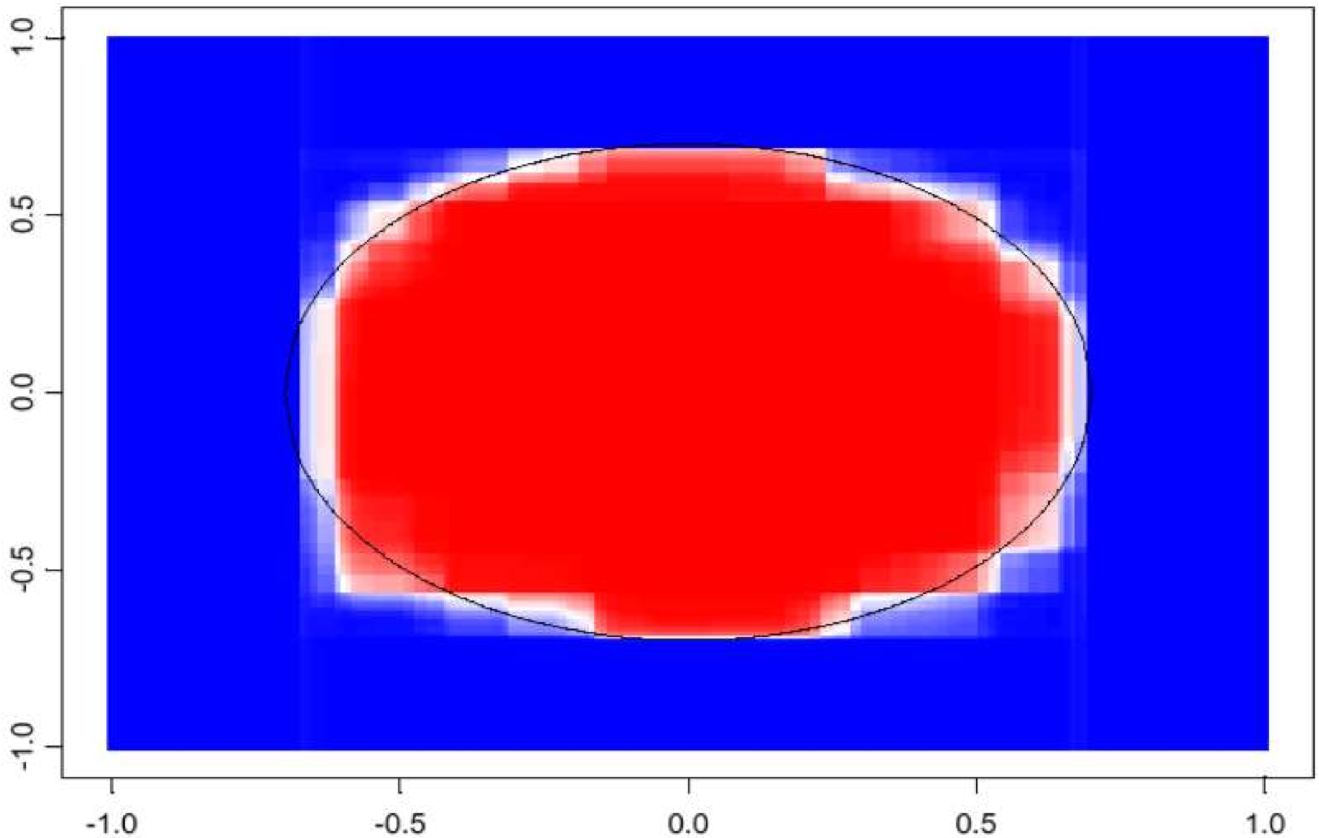


http://people.csail.mit.edu/dsontag/courses/ml13/slides/lecture13.pdf decision tree learning algorithm; very similar to ID3

CART decision boundary



100 bagged trees



shades of blue/red indicate strength of vote for particular classification

Random Forests

- Create m decision trees.
- Each tree uses a bootstrap sample from D
- For each node of each tree
  - Randomly sample $\sqrt{d}$ attributes from the available d attributes
  - Select the best split from among the sampled attributes

Many other variants of randomize selection of attributes [skipping those]

Random forests one of the best performing of the traditional classifiers in many applications.

Random forests Algorithm

1. For $b = 1$ to $\cancel{B}$: M D_j
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached. d
      $\sqrt{d}$
      i. Select $\cancel{m}$ variables at random from the p variables.
      ii. Pick the best variable/split-point among the $\cancel{m}$. $\sqrt{d}$
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Regression: $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

From http://people.csail.mit.edu/dsontag/courses/ml13/slides/lecture13.pdf