Random Forests

Nishant Mehta

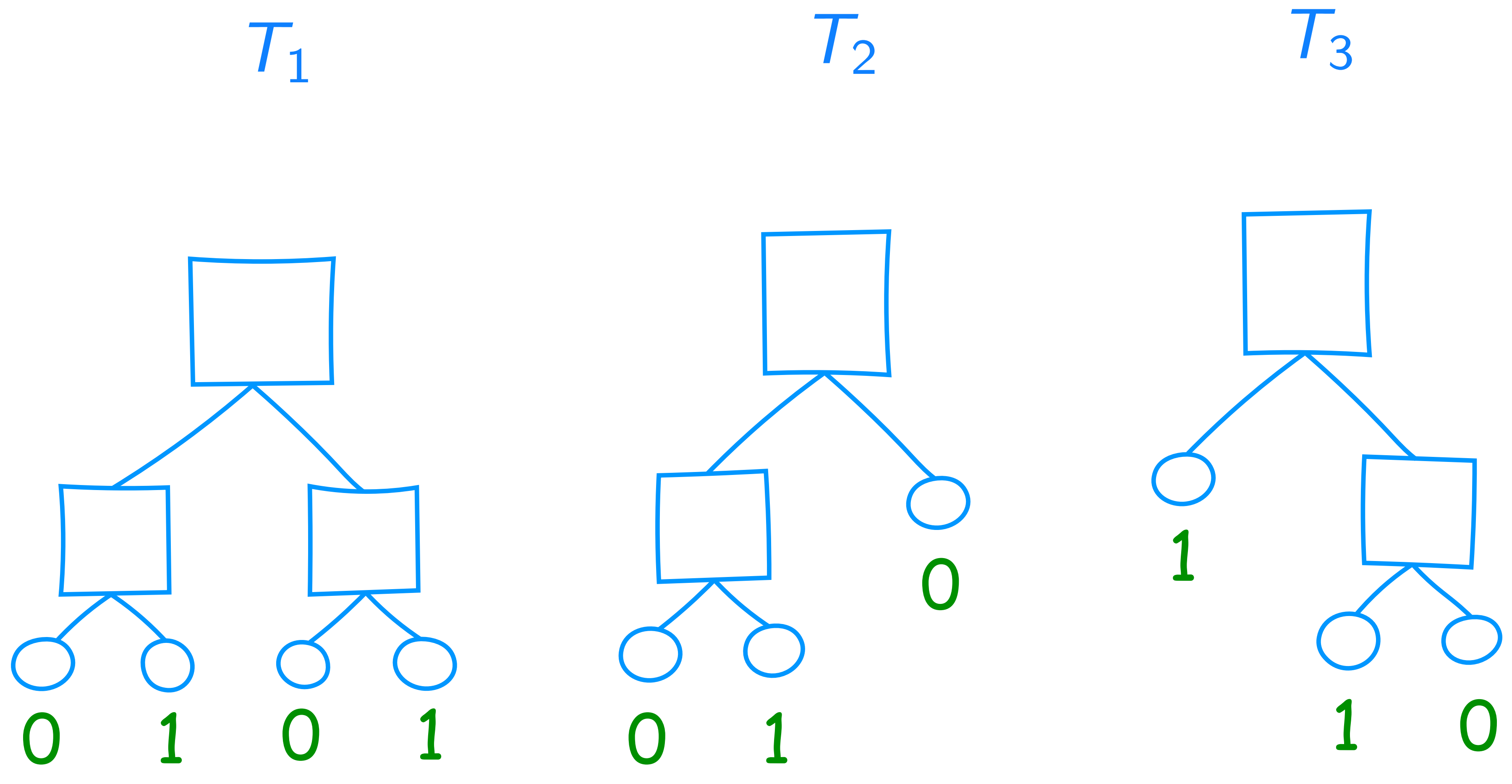
Lecture 3

Ensemble methods

An *ensemble method*: a method that predicts by aggregating the predictions of many hypotheses

Bagged Trees and *Random Forests* are both ensemble methods

- Common idea: Take the majority vote (for classification) among a large, diverse set of classifiers, each of which is a decision tree (trained on a random ***bootstrap*** sample of the training set)

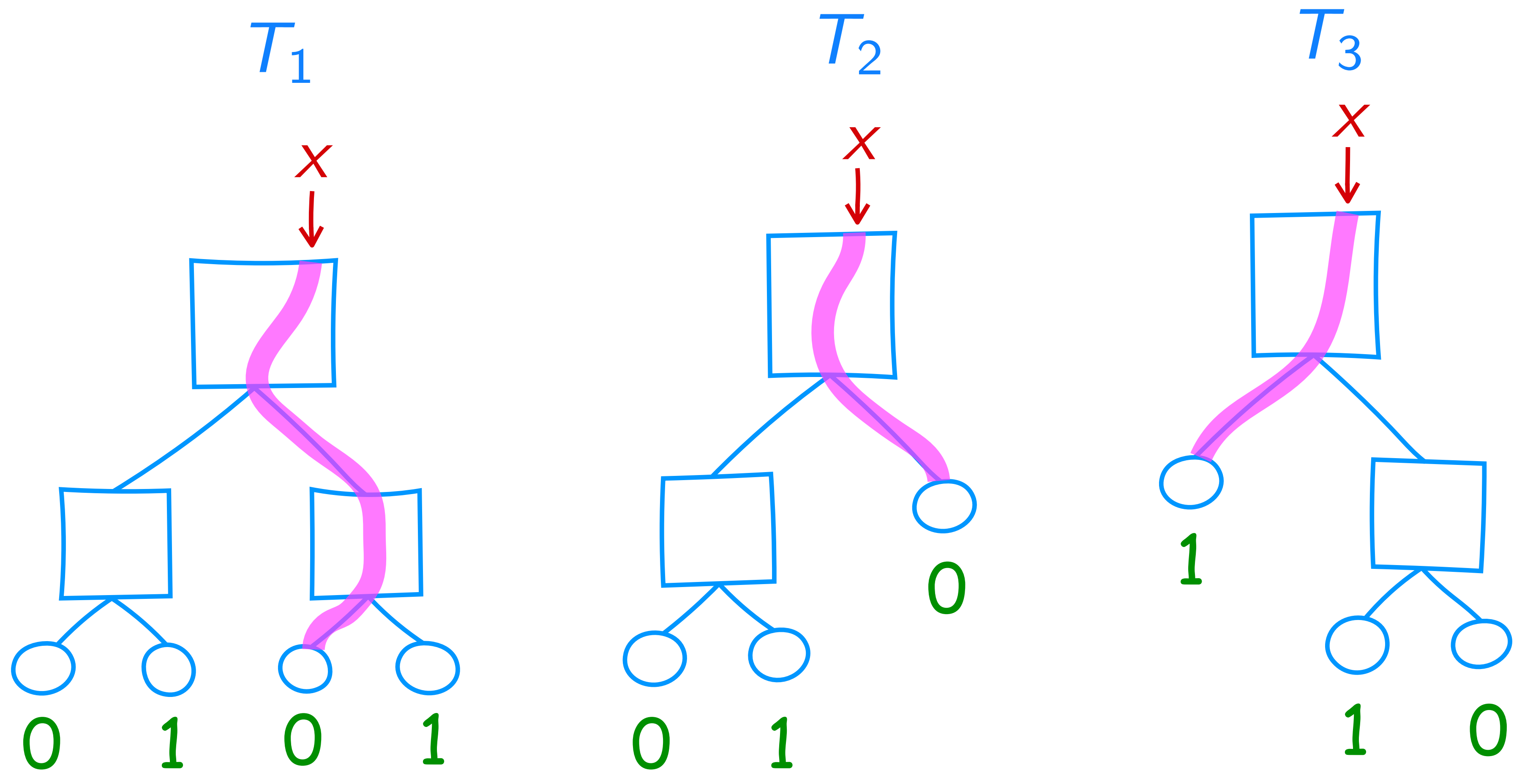


Ensemble methods

An *ensemble method*: a method that predicts by aggregating the predictions of many hypotheses

Bagged Trees and *Random Forests* are both ensemble methods

- Common idea: Take the majority vote (for classification) among a large, diverse set of classifiers, each of which is a decision tree (trained on a random ***bootstrap*** sample of the training set)



Bootstrap Aggregation (Bagging)

Given a training sample $D = ((X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n))$ of size n, a ***bootstrap sample*** is obtained by drawing n examples *with replacement* from D.

For a given learning algorithm, ***bootstrap aggregation***, or ***bagging***, trains the learning algorithm M times as follows:

For j = 1, 2, …, M

- Draw a new bootstrap sample $\tilde{D}_j$ from D
- Train the algorithm on $\tilde{D}_j$, giving hypothesis $\hat{h}_j$

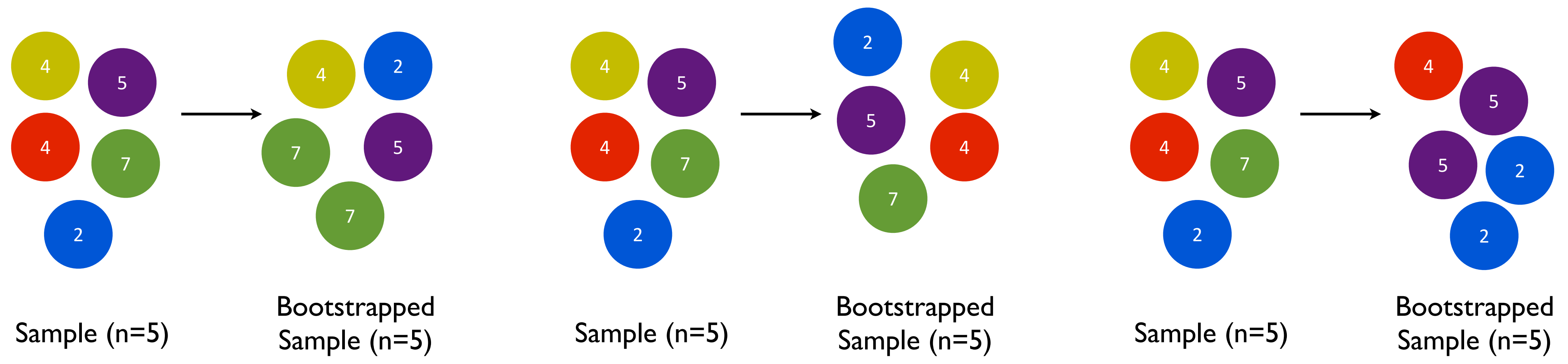
Bootstrap Aggregation (Bagging)

Given a training sample $D = ((X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n))$ of size *n*, a ***bootstrap sample*** is obtained by drawing *n* examples *with replacement* from *D*.

For a given learning algorithm, ***bootstrap aggregation***, or ***bagging***, trains the learning algorithm *M* times as follows:

For $j = 1, 2, \ldots, M$

- Draw a new bootstrap sample $\tilde{D}_j$ from D
- Train the algorithm on $\tilde{D}_j$, giving hypothesis $\hat{h}_j$


4
5
4
7
2
Sample (n=5)
4
2
7
5
7
Bootstrapped Sample (n=5)
4
5
4
7
2
Sample (n=5)
2
4
5
4
7
Bootstrapped Sample (n=5)
4
5
4
7
2
Sample (n=5)
4
5
5
2
2
Bootstrapped Sample (n=5)

(Ong, 2014) A primer to bootstrapping

Bootstrap Aggregation (Bagging)

Given a training sample $D = ((X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n))$ of size *n*, a ***bootstrap sample*** is obtained by drawing *n* examples *with replacement* from *D*.

For a given learning algorithm, ***bootstrap aggregation***, or ***bagging***, trains the learning algorithm *M* times as follows:

For *j* = 1, 2, …, *M*

- Draw a new bootstrap sample $\tilde{D}_j$ from *D*
- Train the algorithm on $\tilde{D}_j$, giving hypothesis $\hat{h}_j$

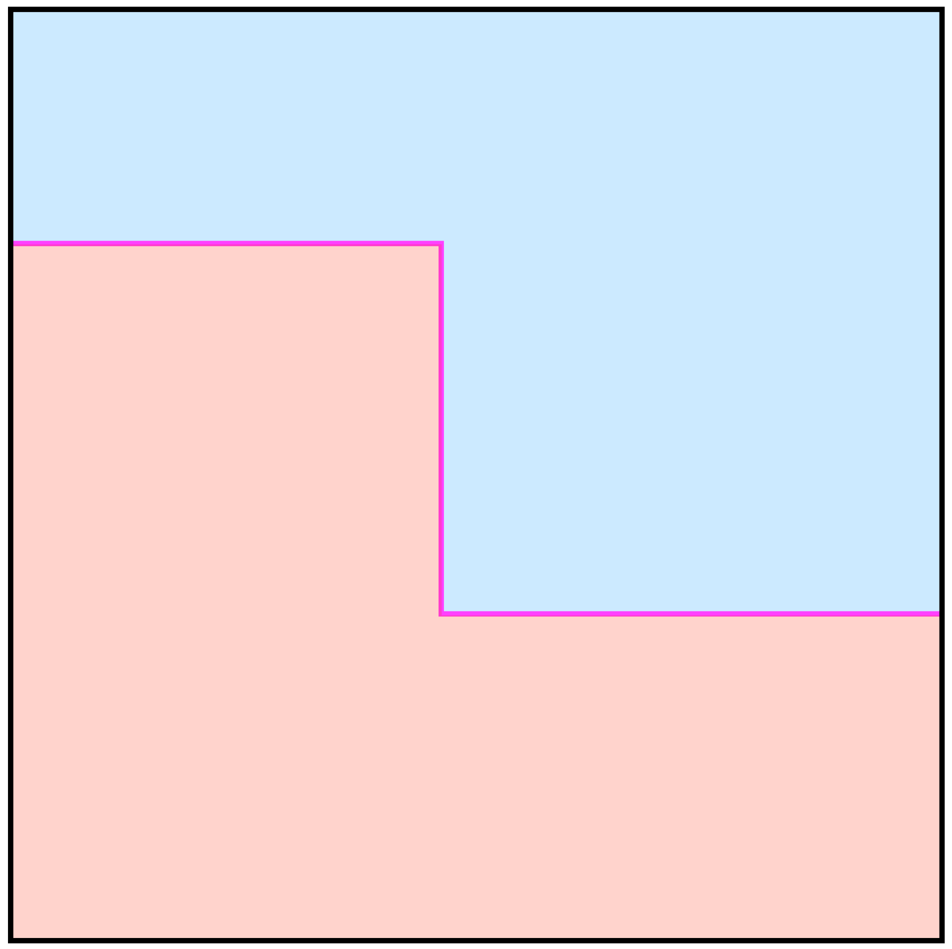
When given a new example, the bagged predictor predicts:

- the ***majority vote*** among $\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_M$ for classification

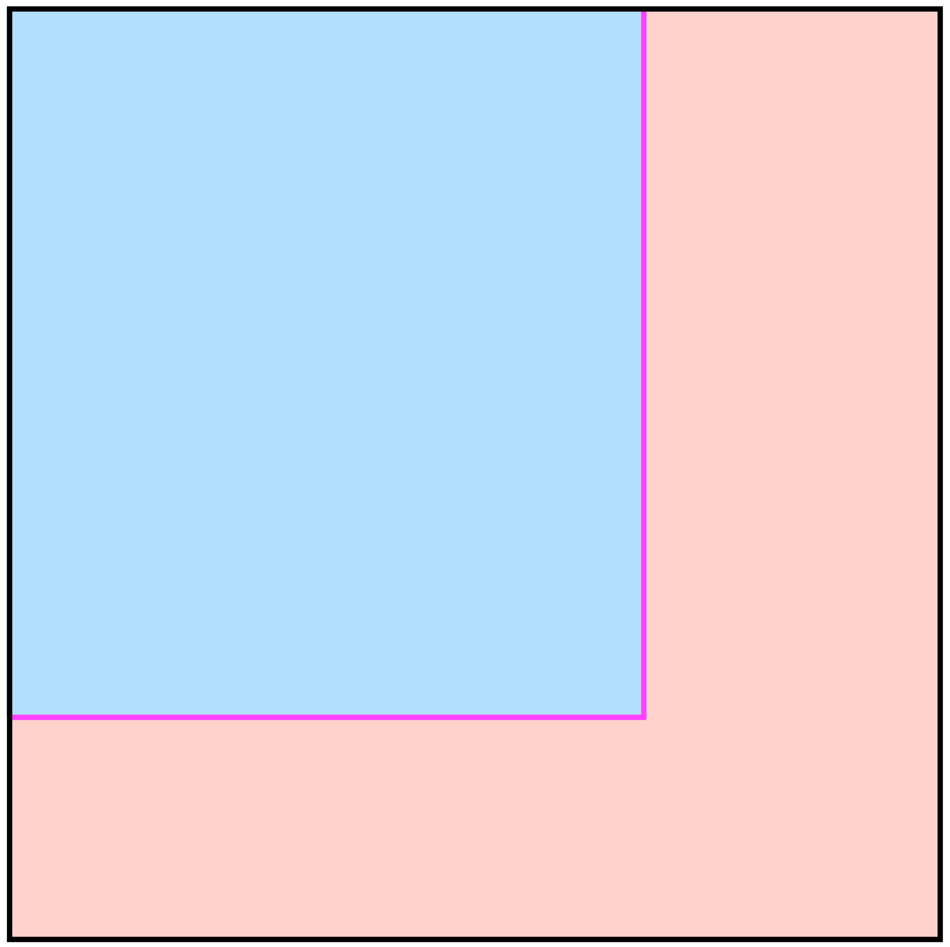
$$\operatorname*{arg\,max}_{y \in \{1,2,\ldots,k\}} \left|\{j \in [M] \colon \hat{h}_j(x) = y\}\right|$$

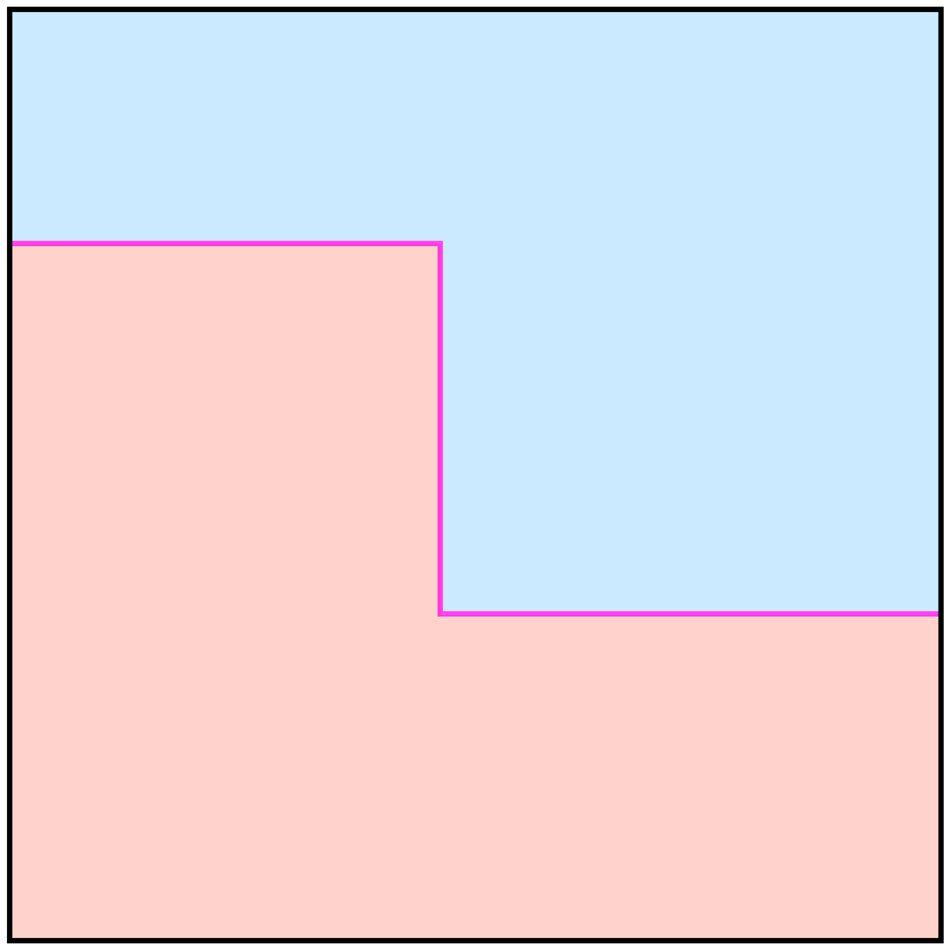
- the mean of $\hat{h}_1, \hat{h}_2, \ldots, \hat{h}_M$ for regression

Geometry: One decision tree

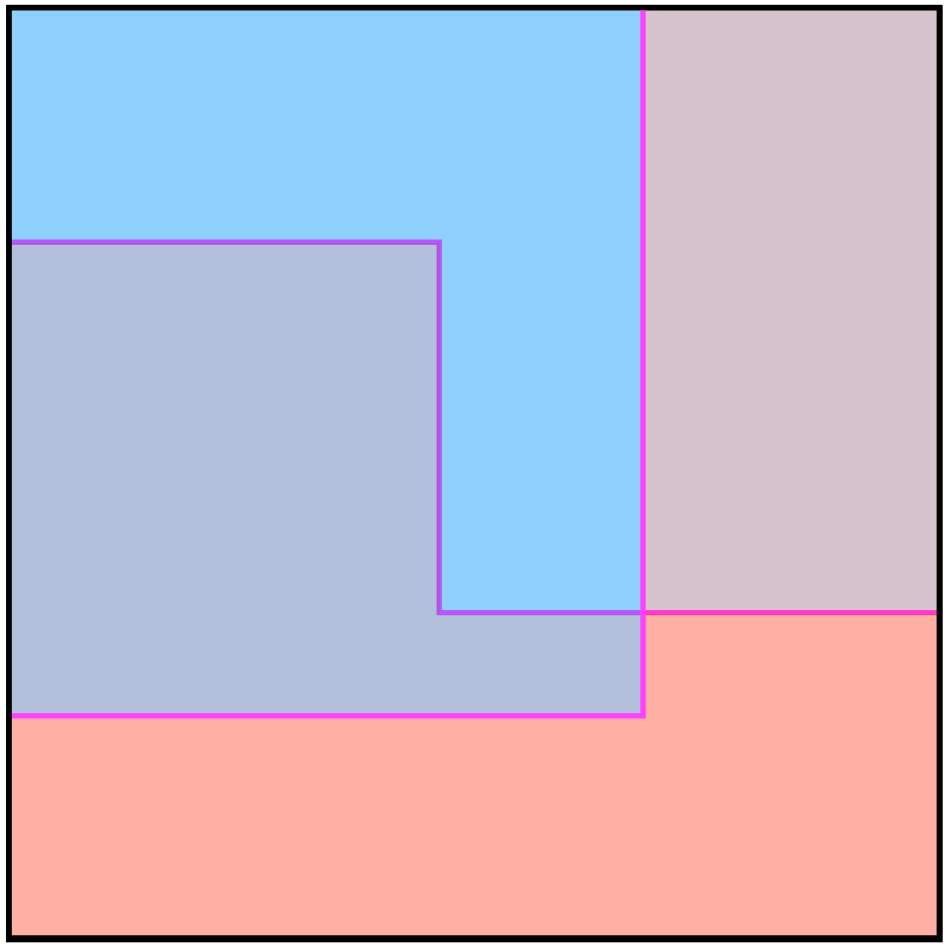


Geometry: Towards majority vote of 2 decision trees

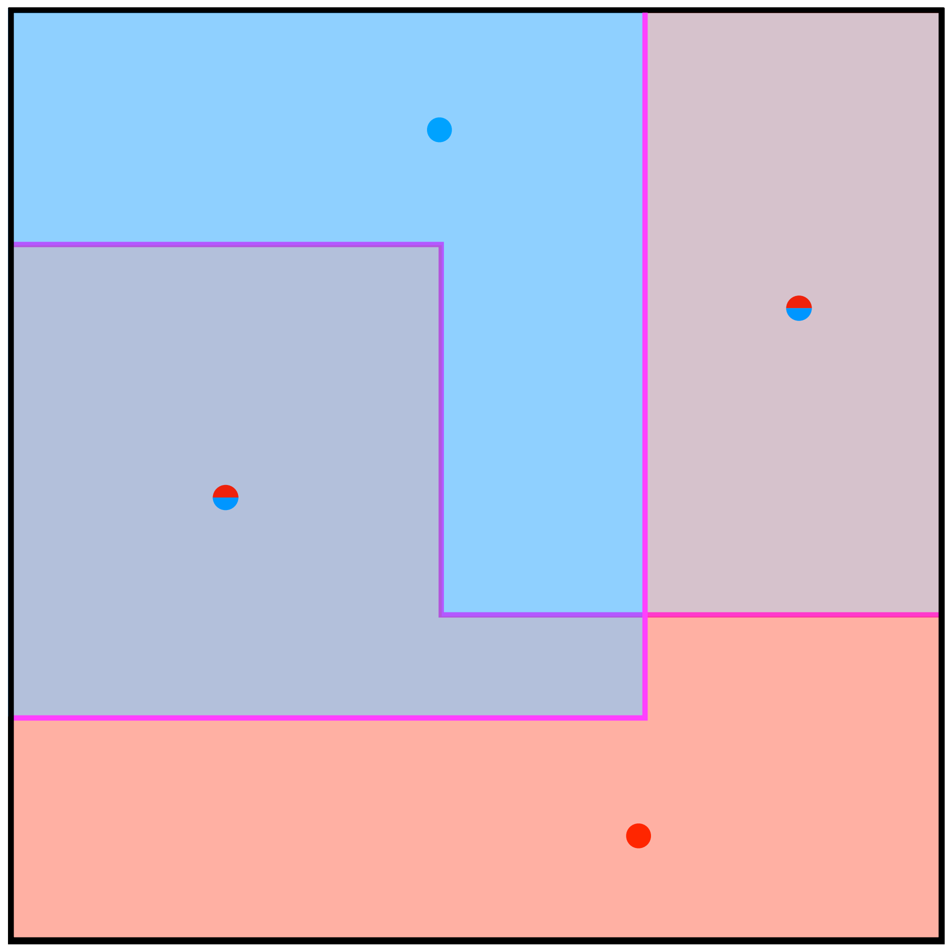




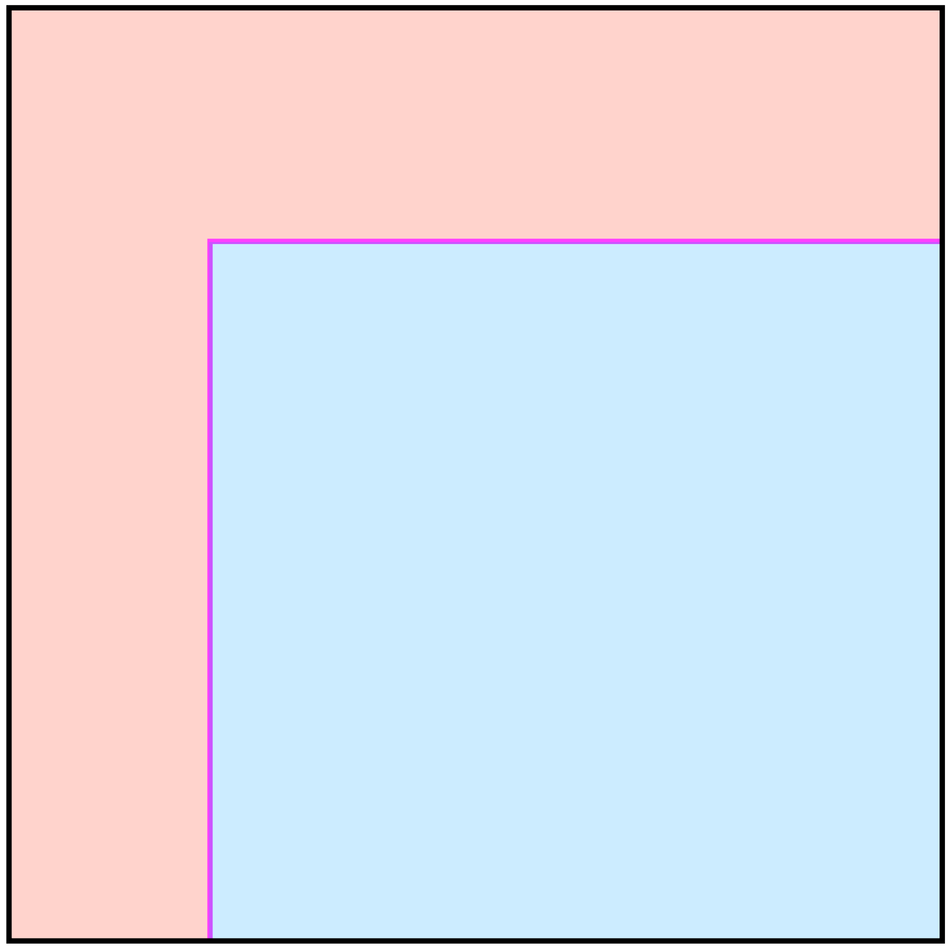
Geometry: Majority vote of 2 decision trees

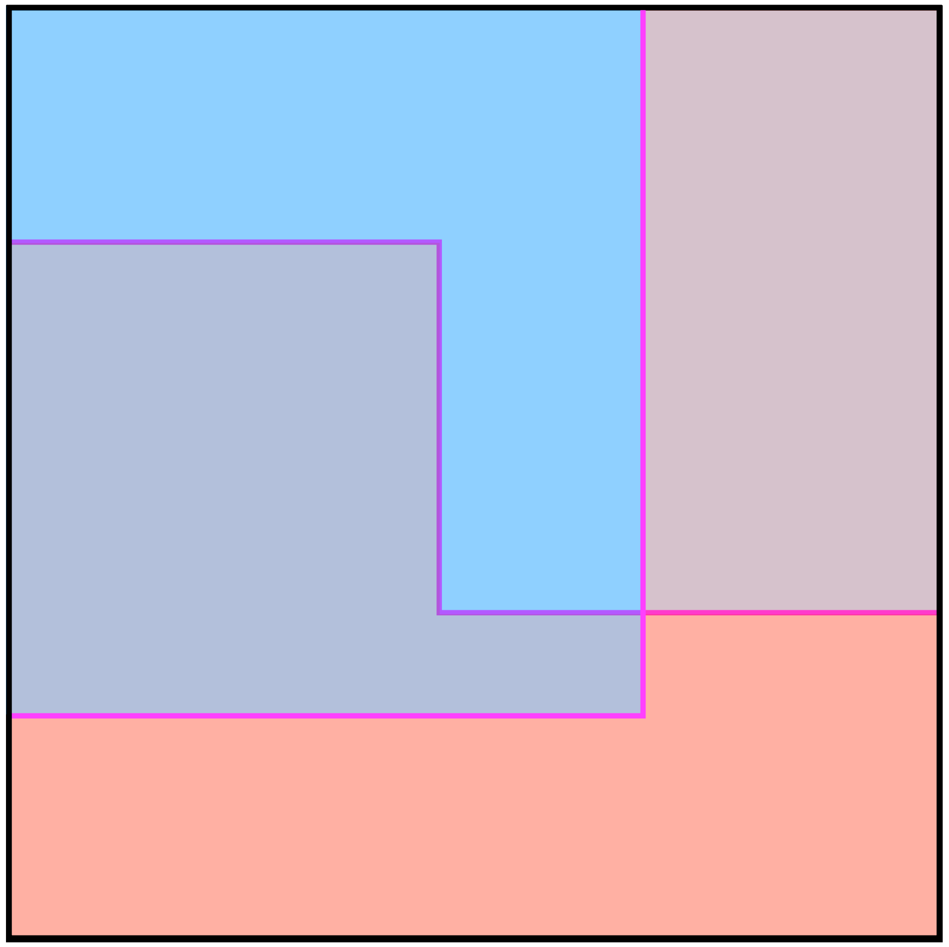


Geometry: Majority vote of 2 decision trees

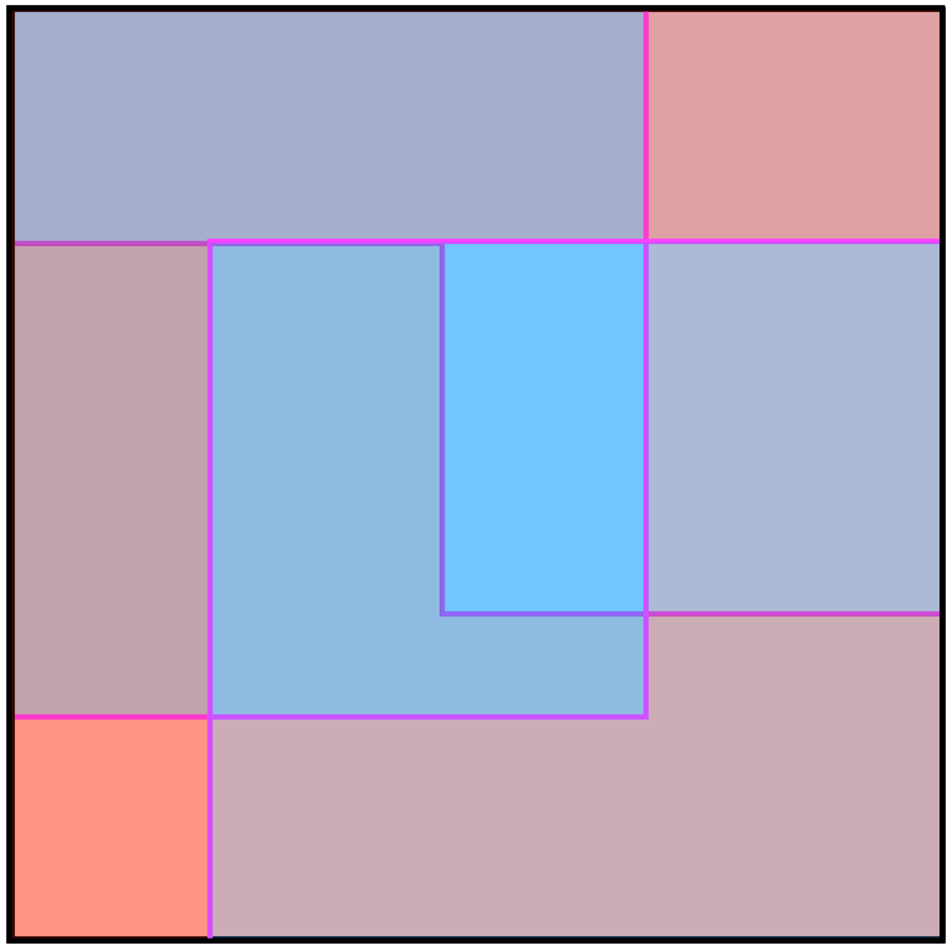


Geometry: Towards majority vote of 3 decision trees

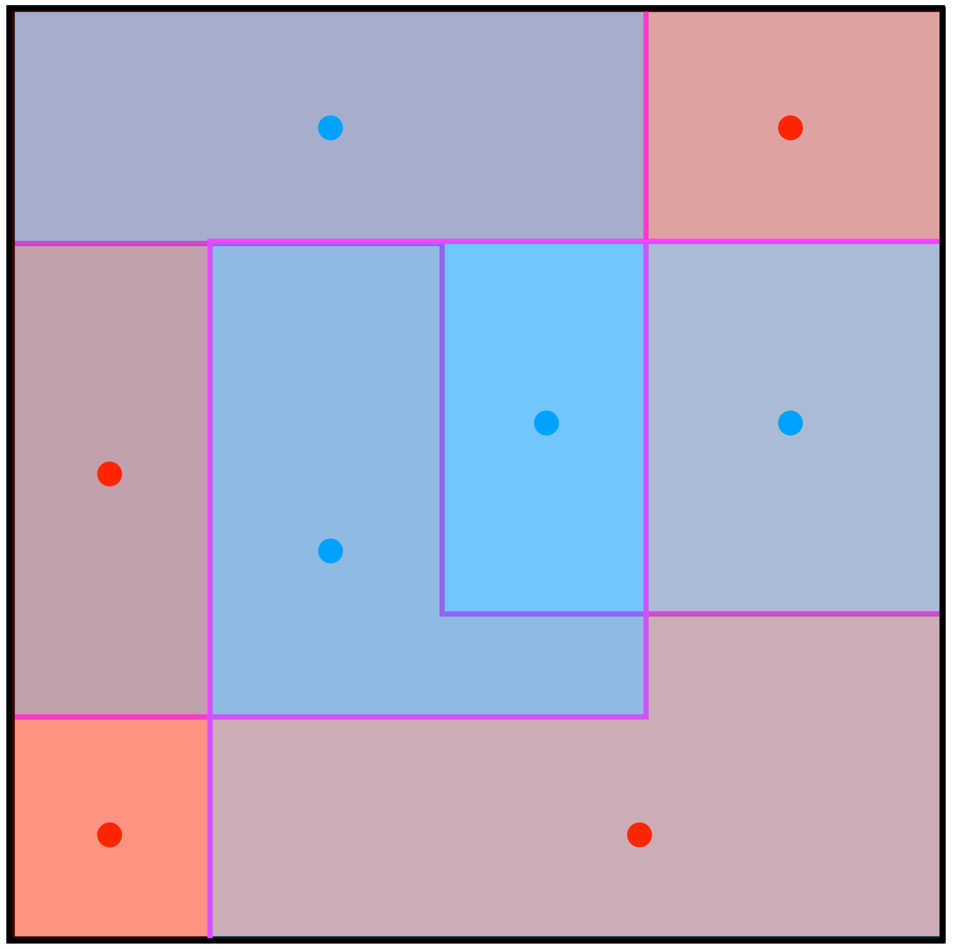




Geometry: Majority vote of 3 decision trees



Geometry: Majority vote of 3 decision trees



Bagged Trees

Each tree is constructed as follows:

Input

- a training set of n examples with d features
- Number of trees, M

Algorithm

For each of the M trees:

- Sample (with replacement) n examples from the training set
- Train the tree using the sample

Are bagged trees a good idea? Yes for regression. For classification, bagging could be better or worse than a single decision tree.

Random Forest Construction

Each tree is constructed as follows:

Input

- a training set of *n* examples with *d* features
- n' $(\leq n)$ - number of examples in each decision tree's training set
- d' $(< d)$ - number of features used to determine the decision at each node of the tree
- Number of trees, *M*

Algorithm

For each of the *M* trees:

- Sample (with replacement) n' examples from the training set
- During tree construction, for each node of the tree, randomly select d' distinct features on which to base the decision at the node. So, the best split will be computed based on these d' features.

Random Forest Construction

Each tree is constructed as follows:

Input

- a training set of n examples with d features
- n' $(\leq n)$ - number of examples in each decision tree's training set
- d' $(< d)$ - number of features used to determine the decision at each node of the tree
- Number of trees, M

Rules of Thumb

Try $n' = n$ (bootstrap samples)
Classification: try $d' = \sqrt{d}$
Regression: try $d' = d/3$

Algorithm

For each of the M trees:

- Sample (with replacement) n' examples from the training set
- During tree construction, for each node of the tree, randomly select d' distinct features on which to base the decision at the node. So, the best split will be computed based on these d' features.

Random Forests

Quite old, and yet one of the best methods out there

Intuition for why they work well:

- Trees have low bias (high representational power) but can have high variance (high sensitivity to particular choice of training sample). Achieve some variance control by aggregating over many trees trained on bootstrapped data

- If errors of individual models tend not to be correlated (different models make different types of errors),
  the majority vote can far outperform the best individual model

Random forests have this advantage,
but bagged trees do not. Why?

Prediction using a Random Forest

(1) Get a prediction for each tree T in the forest $\mathcal{T}$

(2) Take the majority vote of the predictions (for classification)

$$\text{Predict} \quad \underset{y \in \{1,2,\ldots,k\}}{\arg\max} \left| \{ T \in \mathcal{T} \colon \; T(x) = y \} \right|$$

set of potential labels

Random Forests

Good news: Good performance in practice

Bad news: A random forest hypothesis is not easily interpretable (whereas a single decision tree typically is interpretable)