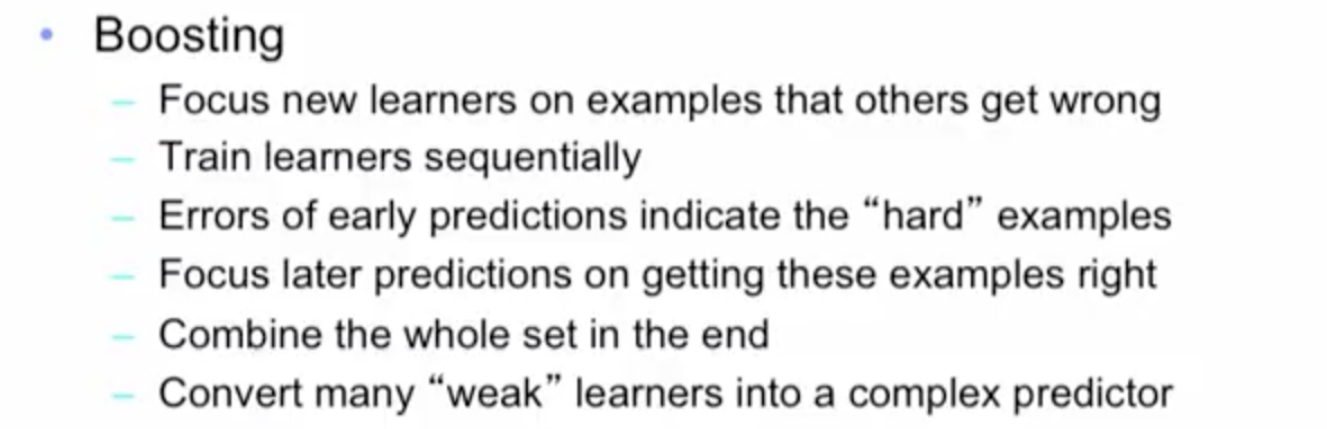
random forests & gradient boosting

Overview

- A decision tree is learnt by splitting a subset of data (that maps to a current leaf) by the most useful features to split across.
- A random forest is an ensemble of decision tress → ensemble to reduce variance
- each decision tree in the forest considers a random subset of features when forming questions and only has access to a random set of the training data points.
- in case of regression, we average the outputs of each tree in the ensemble
- in case of classification, we let the trees vote and take a majority
- Bagging vs boosting
  - **Bagging** is a simple ensembling technique in which we build many *independent* predictors/models/learners and combine them using some model averaging techniques. (e.g. weighted average, majority vote or normal average)
  - Example of bagging ensemble is **Random Forest models.**
  - **Boosting** is an ensemble technique in which the predictors are not made independently, but sequentially.
  - This technique employs the logic in which the subsequent predictors learn from the mistakes of the previous predictors.
  - Gradient boosting is an example of a boosting ensemble vs random forests which is a bagging ensemble.
  - Because we keep training on the mistakes in the boosting model, we need to be vary of overfitting and have a good stopping criterion



- math for gradient boosting

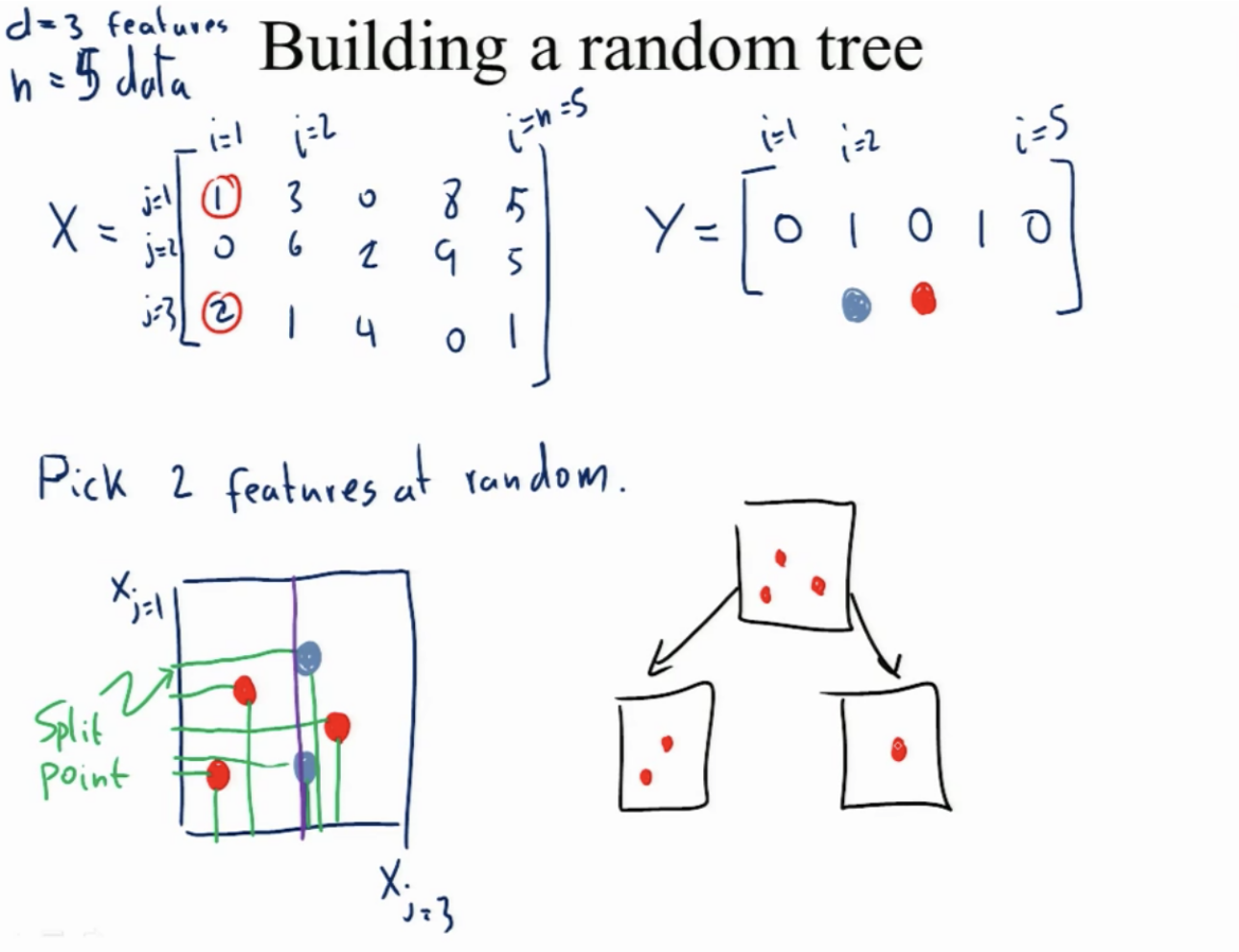
  - pick the next leaf to split based on which feature gives a good split → sequentially picking the worst performing leaf next

- we use an additive strategy: fix what we have learned, and add one new tree at a time.
- for an overview of gradient boosting, see xgboost intro

Random forests

- the tree is built by choosing splitting on which feature gives us the best information gain. if the feature is a set of continuous valued functions, sample at random.

- One measure for information gain is entropy minimization: $-\sum p_i log(1/p_i)$ where p_i is the probability (by counting) of a class i. Note that we get better information gain (lower entropy) if the split is something like [3/4, 1/8, 1/8] rather than [2/3, 2/3, 2/3].
- Another could be $\sum_i p_i^2$ over all classes in that split.
- using something like the above measures, we choose a split
- how to build a random tree? lets say we have a dataset of 5 inputs with 3 features in each input. pick a random set of features → lets say two features, use that to generate some candidate split points (here we use the projection onto axes as candidates). pick the one that gives the most information gain

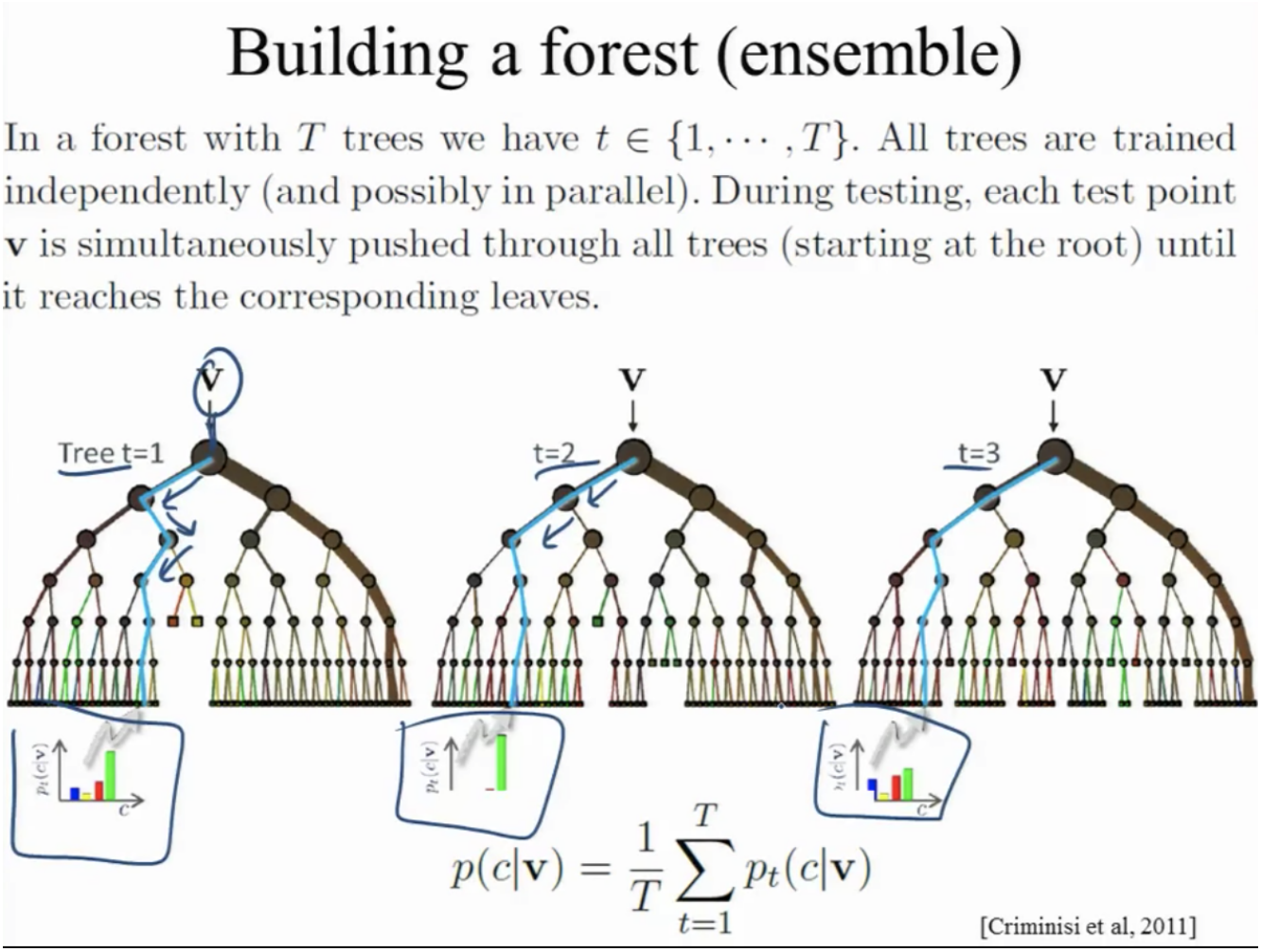


- why pick at random? we have tons of features and tons of split points. lets pick at random and build many trees (like a 1000 trees). over time we hope to pick up good split points.

Random Forests algorithm

1. For $b = 1$ to B:
   (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.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

- note that the algorithm has randomness both in the data chosen to train each tree but also randomness in the feature picked as split point.
- because we build many independent trees, this is trivially parallelizable and if we have a cluster of machines we can take advantage of it.
- **Bootstrapping** is the idea of sampling data at random and training a set of classifiers with the thus sampled subsets of data. we use bootstrapping here (see algorithm above)
- each of these trees has very high variance. like really high
- for a ML algorithm to benefit from bagging, they need to be uncorrelated errors.



Text classification example

In news categorization, a possible term is *Bill Clinton.* A corresponding **weak learner (node)** is: If the term *Bill Clinton* appears in the document predict that the document belongs ~~to~~ **News**.