

# Foliage:

Decision Trees of A Different Color

Brett Borghetti

October 10, 2007

# Motivation

- Classical decision trees are powerful classifiers:
  - Capable of arbitrary partitions of the data space
  - Very fast to use once trained
  - Humans can read and understand them
- But they lack a number of desirable features of a classifier...

# What are the problems with Classical Decision Trees?

- Can't process very large training data sets
- Not efficiently capable of “online learning”
- Don't handle “concept drift” efficiently
- Noisy data can cause structural sensitivities
- May take many nodes to represent non-axis parallel data
- May require additional processing to reduce overfitting and improve generalization accuracy

# Classical Decision Trees: Quick Review (Wikipedia/Mitchel)

- Training Data has one or more attributes  $[X_{1..n}]$  and a label  $[y]$
- A decision tree can be constructed top-down using the information gain in the following way:
  1. Begin at the root node
  2. Determine the attribute with the highest *information gain* which is not used in an ancestor node
  3. Add a child node for each possible value of that attribute
  4. Attach all examples to the child node where the attribute values of the examples are identical to the attribute value attached to the node
  5. If all examples attached to the child node can be classified uniquely add that classification to that node and mark it as leaf node
  6. Go back to step two if there is at least one more unused attribute left, otherwise add the classification of most of the examples attached to the child node

# Information Gain Review

- Entropy of a dataset

$$H(X) = \mathbb{E}_X[I(x)] = - \sum_{x \in \mathcal{X}} p(x) \log p(x).$$

- Information Gain: the difference between the entropy of the original dataset and the weighted entropy of each data subset once separated by the split. Since the split reduces the entropy of the remaining data, gain should be positive.

$$IG(E_x, a) = H(E_x) - \sum_{v \in \text{values}(a)} \frac{|\{x \in E_x \mid \text{value}(x, a) = v\}|}{|E_x|} \bullet H(\{x \in E_x \mid \text{value}(x, a) = v\})$$

Original Data Set  
Entropy

Weight (fraction of  
data in subset)

Entropy of Subset

# Overview

- “Very Fast Machine Learning” extensions:
  - Very Fast Decision Trees
  - Very Fast Decision Trees with concept drift
- Oblique Decision Trees
- Random (Decision) Forests

# Very Fast Decision Trees

- Objective
  - Make a decision-tree-like classifier that can handle incoming “real-time” data, interleaving predictions and training
  - Also can be used to handle larger-than-memory-size datasets by streaming data from storage as if it was coming in real time
  - Resulting trees must have accuracy within tolerance of (or better than) trees generated with classical method

# Very Fast Decision Trees (Domingos & Hulten)

- Start with a root node (which is actually a leaf), then recursively
  - Take the next data item and predict its class according to the current tree, report the prediction
  - Increment counts of actual classes at the leaf node
  - Relabel that leaf (if necessary) with the label of the majority of examples at that leaf
  - If all of the data at that leaf node are not of all the same class then consider *growing* sub-branches to further classify the non-homogeneous data

# Very Fast Decision Trees (Domingos & Hulten)

- Growing sub-branches:
  - Calculate the “split evaluation” value for each of the attributes at the potential branch point (Information Gain or Gini Impurity, for example)
  - Consider the 2 attributes with the highest values
  - Using *Hoeffding bounds*, determine if the value difference is significant, and if so, build a branch that splits based on the higher valued attribute
  - For each branch in the split, add a leaf node, increment the counts of the classes at each new leaf.

# Hoeffding Bounds

- Want to determine how much of a difference between split attribute evaluation warrants growing a new branch
- $R$  is the range of a split attribute value
- With probability  $1-\delta$  the true mean of the variable is at least

$$\bar{r} - \epsilon$$

- Given a  $1-\delta$ ,  $n$ , and  $R$ , solve for epsilon for each attribute

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

- If the epsilons are small enough to disambiguate the two split attribute values than a split is warranted

# Very Fast Decision Trees

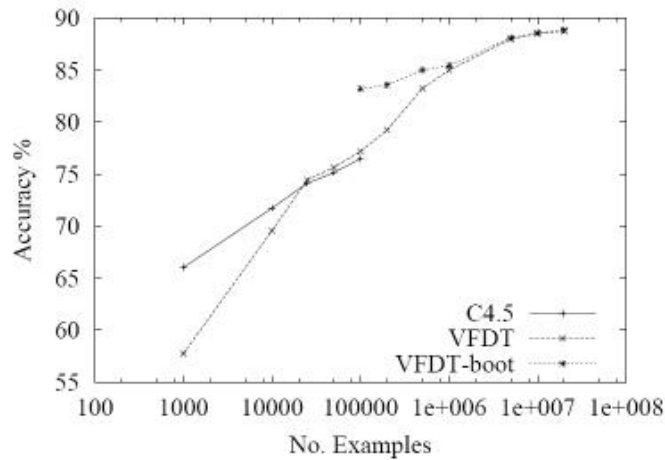


Figure 1: Accuracy as a function of the number of training examples.

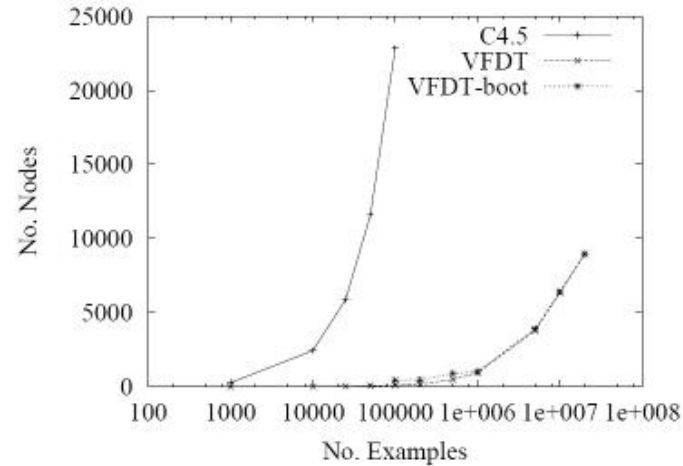


Figure 2: Tree size as a function of the number of training examples.

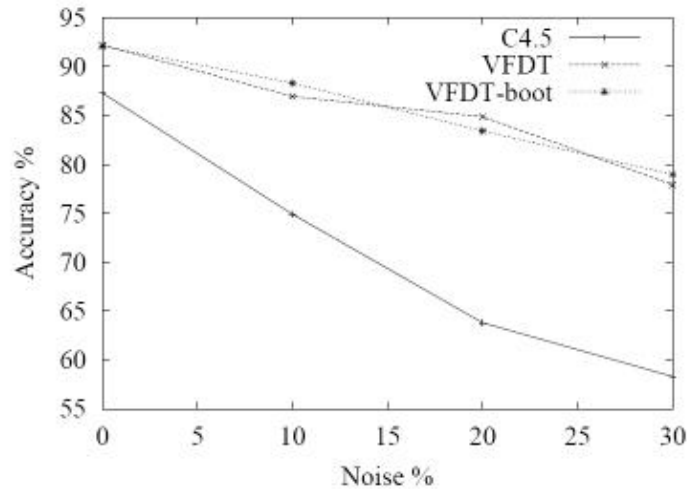


Figure 3: Accuracy as a function of the noise level.

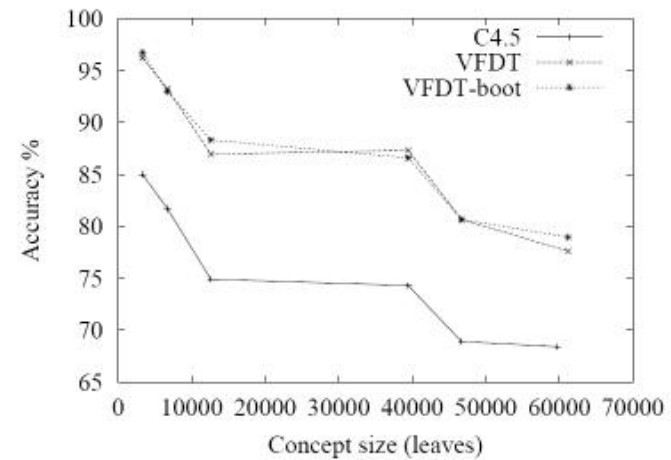


Figure 4: Accuracy as a function of the complexity of the true concept.

# Concept Drift

- Suppose that instead of coming from one stationary process your data was being generated from two or more different stationary processes and that there was a secret mechanism determining which process was currently generating data
- Alternately, consider an parameterized generation process where one or more of the parameters was varying over time in a manner which you do not have visibility
- Example: customer purchase data over the course of several months changes because of seasonal purchases as well as the introduction of new products in the market

# Concept-adapting Very Fast Decision Trees (Hulten, Spencer, Domingos)

- Concept adaptive Modifications to VFDT
  - Keep a sliding window record of statistics for the nodes, “forgetting” examples that are too old to remain in the window by removing them from the statistics of the applicable nodes
  - When a node’s feature split appears to be no longer be correct, start growing an *alternate* tree using a better split, but keep using the original tree
  - Periodically *test* each node over the next  $m$  incoming data points to see if the data supports the original or alternate tree better. If the alternate sub-tree is better than the original sub-tree, remove the original sub-tree and replace it with the alternate
  - Prune alternate trees that don’t seem to be making progress in accurate predictions

# Concept-adapting Very Fast Decision Trees (Hulten, Spencer, Domingos)

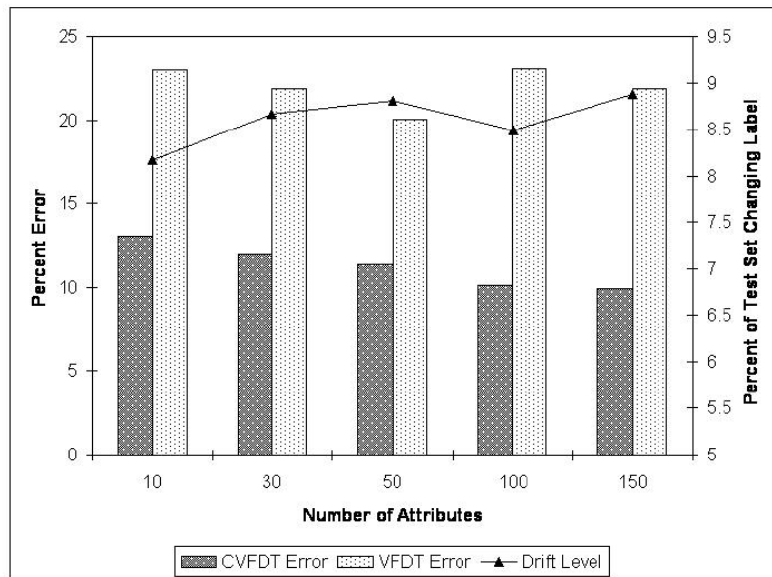


Figure 1: Error rates as a function of the number of attributes.

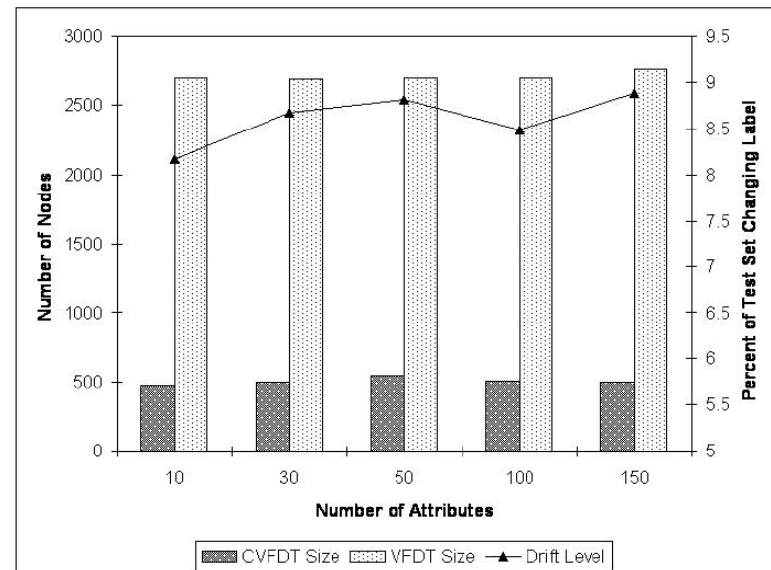
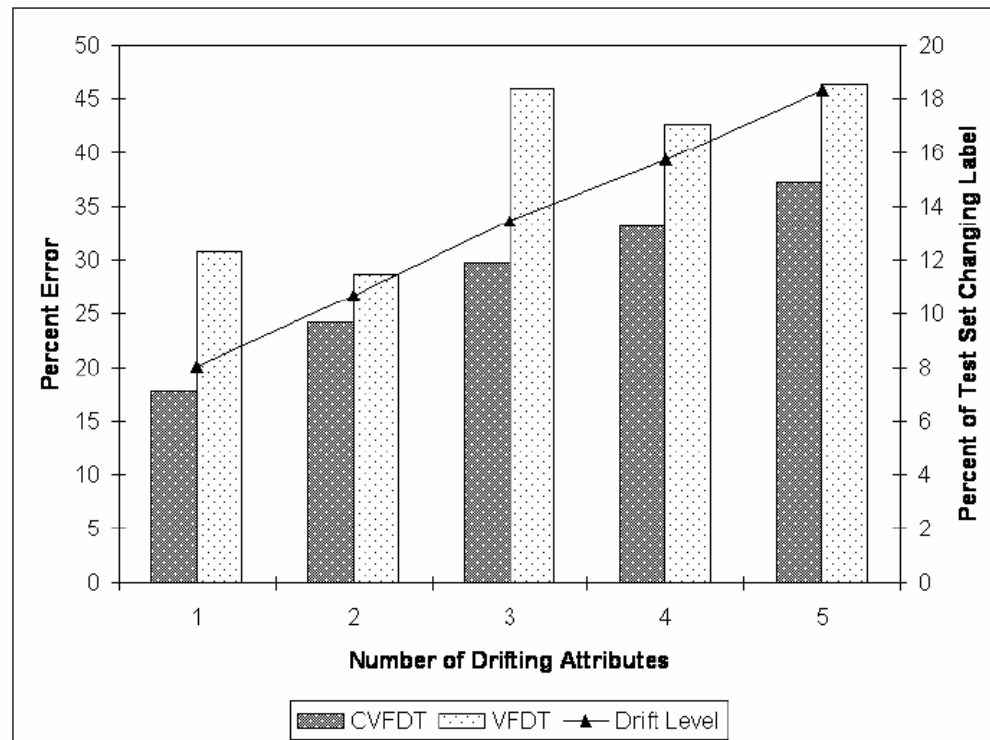


Figure 2: Tree sizes as a function of the number of attributes.

- Used rotating hyperplane in attribute-space to generate data with concept drift

# Concept-adapting Very Fast Decision Trees (Hulten, Spencer, Domingos)



**Figure 4: Error rates as a function of the amount of concept drift.**

# Oblique Decision Trees (Heath, Kasif, Salzberg)

- Problem with classical decision trees: Data that is “diagonal” in the feature space requires larger trees than data that is aligned with one or more features
- Solution: Instead of forcing each node to represent an axis-parallel cut through the data, develop a new kind of tree which is composed of arbitrary linear separators (hyperplanes) at each node

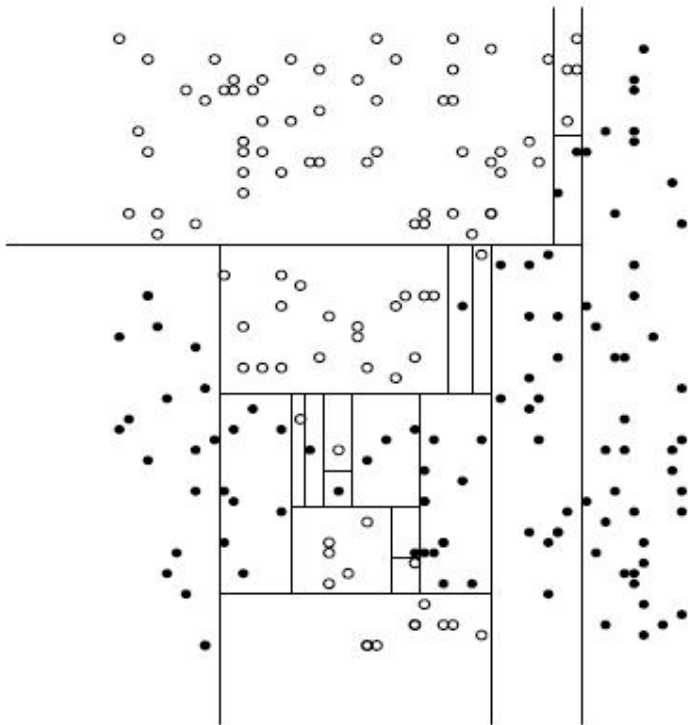
# Oblique Decision Trees (Heath, Kasif, Salzberg)

- Recursive Process (in batch mode)
  - *Select a hyperplane* that is a decent classifier on the current data
  - Separate the set of training examples using the hyperplane
  - For each sub-set, repeat the process
  - Continue the process until some desired goodness is achieved

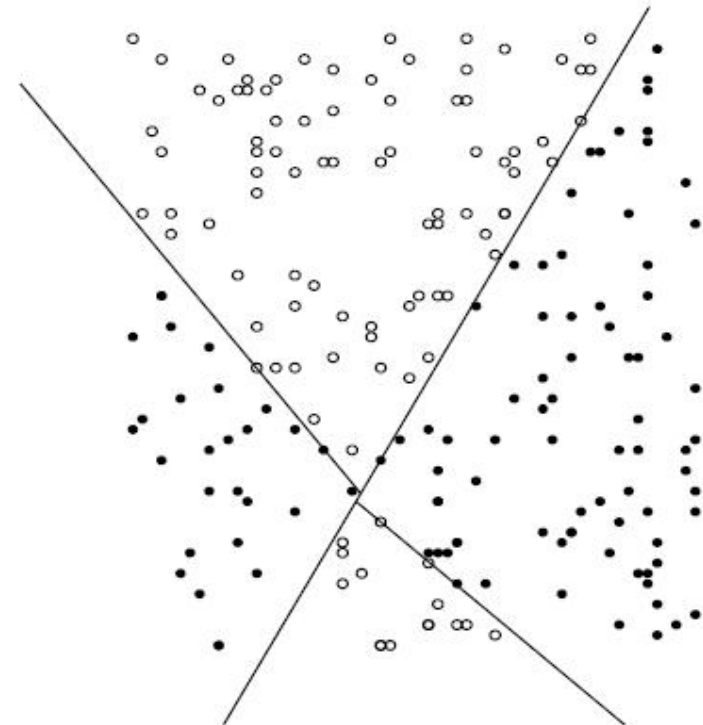
# Oblique Decision Trees (Heath, Kasif, Salzberg)

- Selecting a hyperplane using simulated annealing:
  - Start with an arbitrary hyperplane based on the unit vectors in the featurespace
  - Repeat
    - Create a new candidate hyperplane by perturbing one dimension by  $[-0.5, 0.5)$
    - Determine the *energy* difference of the hyperplanes
    - Select the better candidate with a simulated annealing scheme: probability of switching =
    - Temperature  $T$  of the system is reduced over time to simulate “cooling” of the learning process and create convergence

# Oblique Decision Trees (Heath, Kasif, Salzberg)



(A)



(B)

Figure 1: Tree generated by (A) ID3 and (B) SADT

# Random Decision Forests

- Problem with classical decision trees: In a tree of a given size, there is an implicit tradeoff between performance on the training data and performance on the unseen data (generalization accuracy). It is often unclear how large to grow a tree for a given dataset.
- Solution: Use multiple trees grown stochastically and combine their predictive power in complementary ways

# Random Decision Forests (Ho)

- Use oblique decision trees
  - Tested several hyperplane generation methods
    - Central axis projection (one pass)
    - Perceptron Training (hard limit on max iterations)
- Grow a forest
  - For each tree:
    - Randomly choose a subset of features to consider
    - Build an oblique decision tree using only attribute values from the chosen subset of features, classifying *all* of the training data correctly

# Random Decision Forests (Ho)

- To classify new data with the forest:
  - Let each tree have an equal vote over the classification of the observation
  - Alternately, weight the votes of each tree using a scheme that gives more weight to trees tending to correctly predict the observations (vector quantization, weighted exponential regret, etc.)

# Random Decision Forests (Ho)

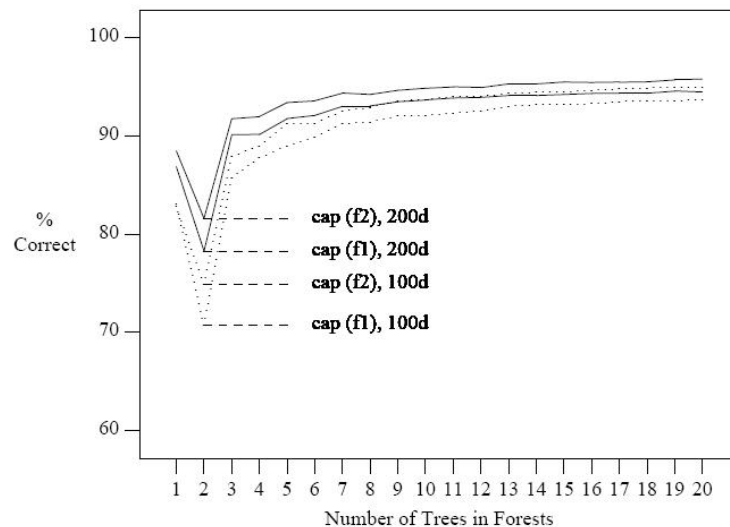


Figure 1: Classification Accuracy (% correct) of Forests Constructed by Central Axis Projection (in 100- and 200- dimensional random subspaces)

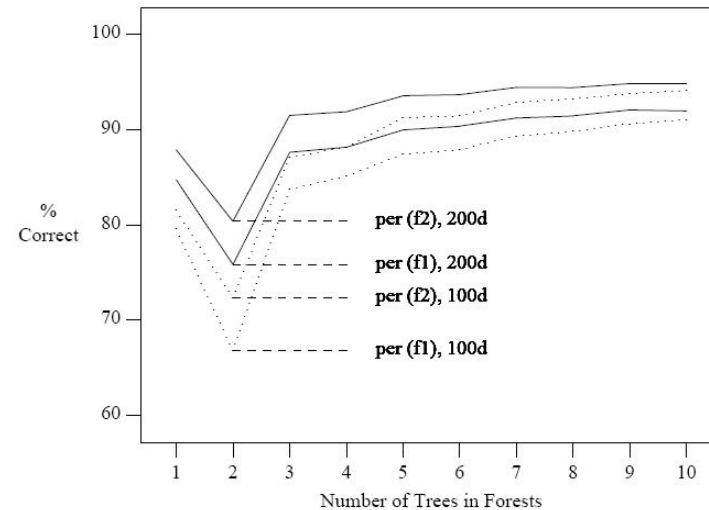


Figure 2: Classification Accuracy (% correct) of Forests Constructed by Perceptron Training Algorithm (in 100- and 200- dimensional random subspaces)

- In general, classification accuracy improves as the number of trees increases

# Summary

- We explored four representative techniques for mitigating some of the problems with classical decision trees
  - Improved tree-building speed and max size of data while minimizing noise effects using heuristics for node expansion
  - Improved concept drift adaptivity through heuristic-based creation of alternate trees
  - Reduced the size of trees using oblique decisions
  - Improved generalization through use of forests of trees in subsets of the feature space

# Sources

Mining High-Speed Data Streams (Domingos & Hulten):

Sixth International conf on knowledge discovery and data mining (2000) pp 71-80

<http://www.cs.washington.edu/homes/pedrod/papers/kdd00.pdf>

Mining Time-Changing Data Streams (Hulten, Spencer, Domingos): Proceedings of the seventh ACM SIGKDD international conf. on knowledge discovery and data mining (2001), pp 97-

106 <http://www.cs.washington.edu/homes/pedrod/papers/kdd01b.pdf>

Induction of oblique decision trees (Heath, Kasif, Salzberg): In proceedings of the 13th IJCAI, pp 1002-1007, Chamb'ery, France, 1993 <http://citeseer.ist.psu.edu/95282.html>

Random Decision Forests (Ho): Proceedings of the 3rd intl conf on document analysis and recognition, Montréal, Canada, Aug 14-18, 1995, pp278-282

<http://citeseer.ist.psu.edu/ho95random.html>

also more info on basics of Random Forests & extra sources at Wikipedia:

[http://en.wikipedia.org/wiki/Random\\_forest](http://en.wikipedia.org/wiki/Random_forest)