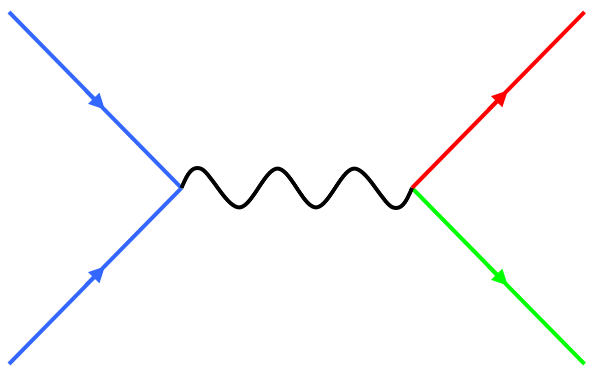
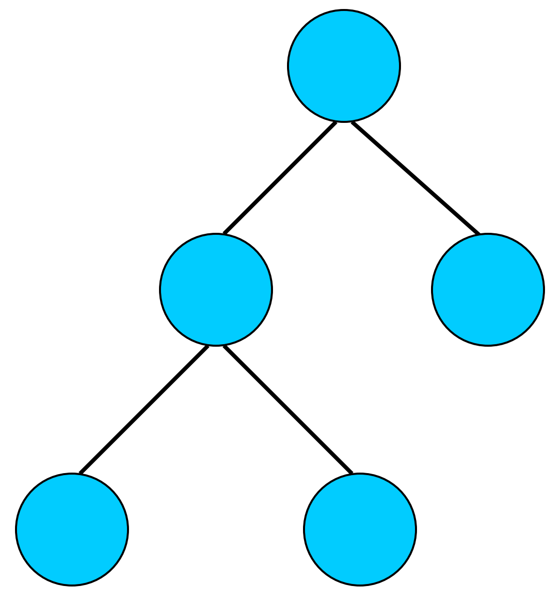
Decision Trees in the Single Top Search

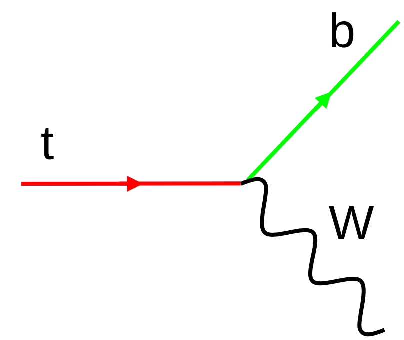




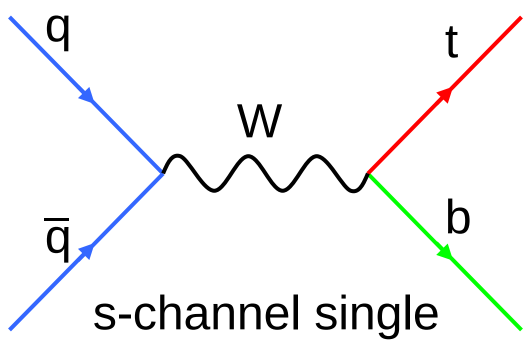
Steve Herrin
University of Washington
REU 2006
Advisor: Gordon Watts

Single Top Quark Production

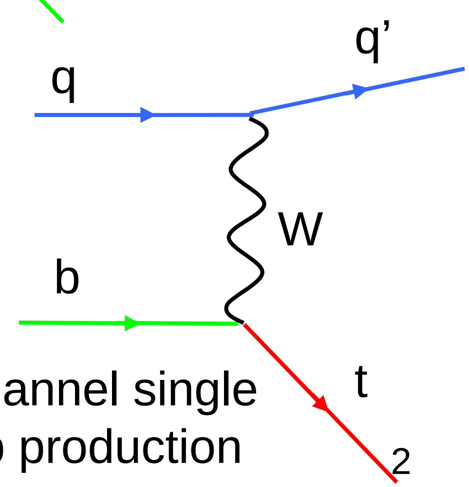
- Top quark decays into a bottom quark and a W boson
- The process can occur in reverse, producing single top
- Cross sections:
  – s-channel: 0.88 pb
  – t-channel: 1.98 pb
  – (about 1 out of every 75 billion collisions)



top quark decay



s-channel single top production



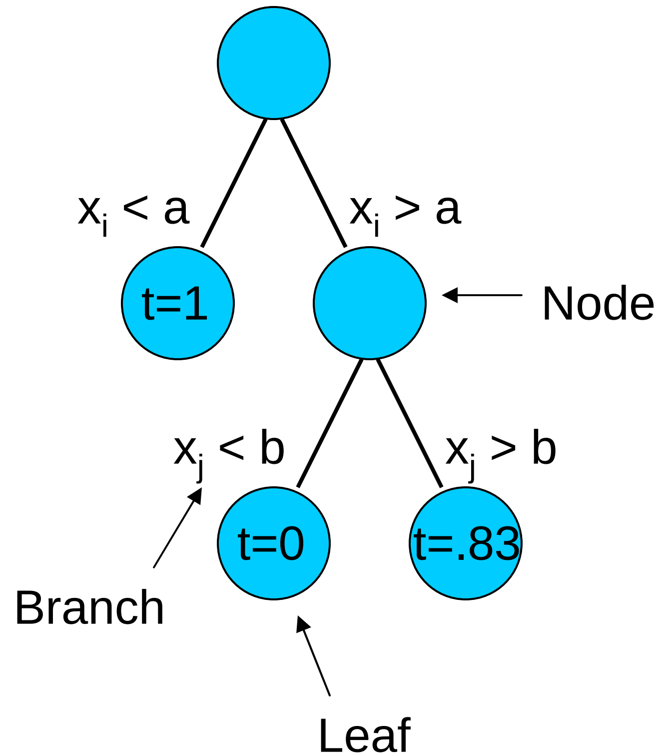
t-channel single top production

Backgrounds

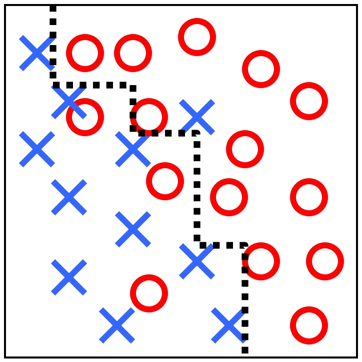
- We have enough data that we should see about 1000 single top events, but it's not so easy
- Several backgrounds that look similar in terms of:
  - Number of jets
  - Energy
  - Sphericity
- Main backgrounds:
  - W+jets: look similar, cross section ~1000 times larger
  - $t\bar{t}$: usually more jets, but imperfect detector loses some, cross section ~5 times larger

Decision Trees

- One type of classifier:
  - Train with known events $(X, t) = (x_1, \ldots, x_n, t)$
  - For unknown events, predict t given X
  - X is various variables from detector, t is 1 for signal, 0 for background
- Tree-like structure



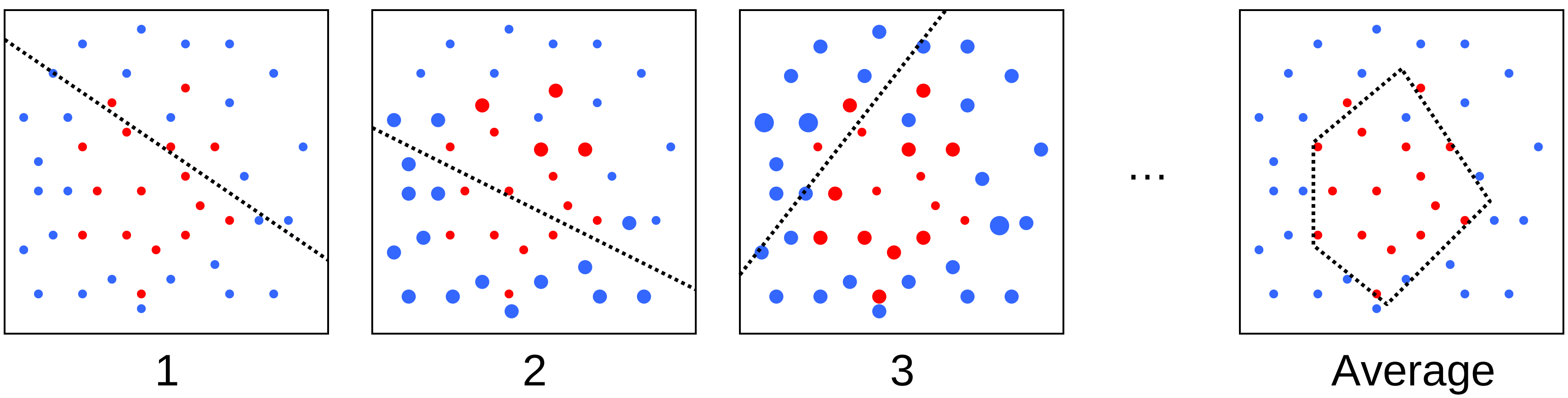
- Advantages:
  - Faster than similar-performing classifiers (e.g. neural nets)
  - Easy for humans to parse
- Disadvantages:
  - Unstable: a small change in input data can cause large changes in output predictions
  - Doesn’t notice correlations between variables (though a human can consider this and make a new variable for input)
  - Discrete output



An iso-prediction in 2D variable space

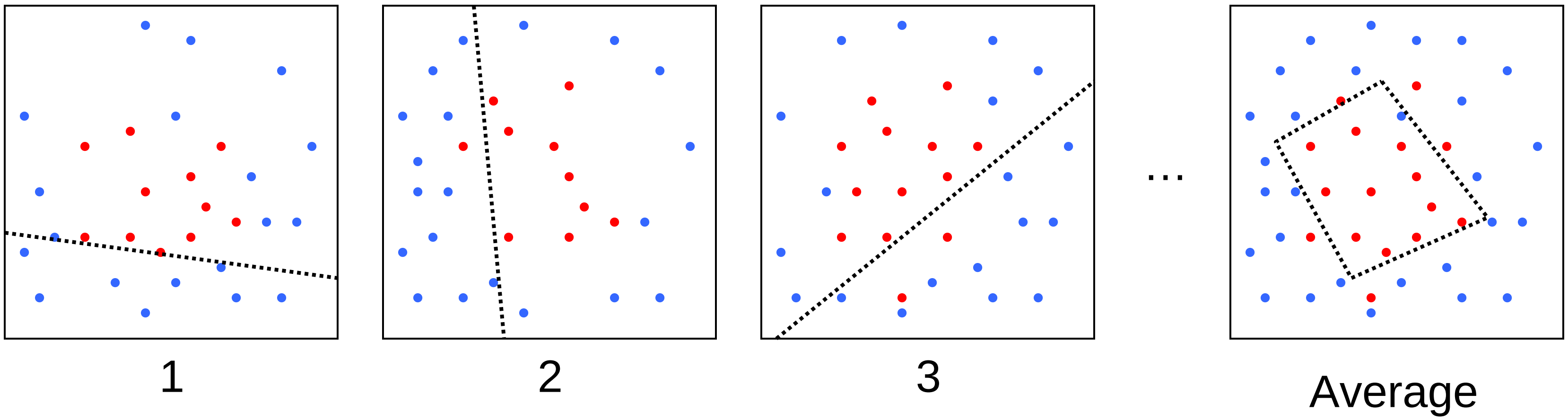
Boosting

- Create multiple trees
- Each tree gives more weight to those events misclassified by previous tree
- Take average of all trees



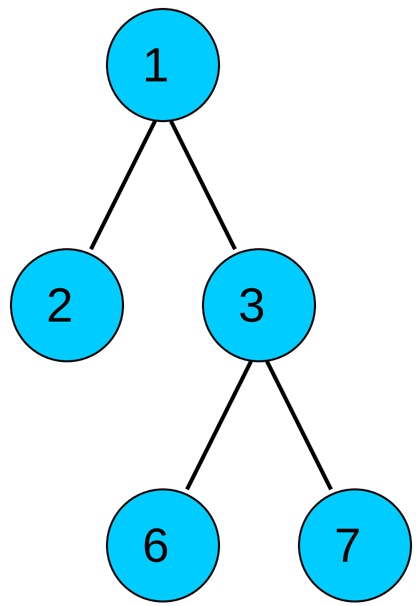
Bagging

- Grow multiple trees again
- Only give each tree a subset of the total training data
- Take average of trees

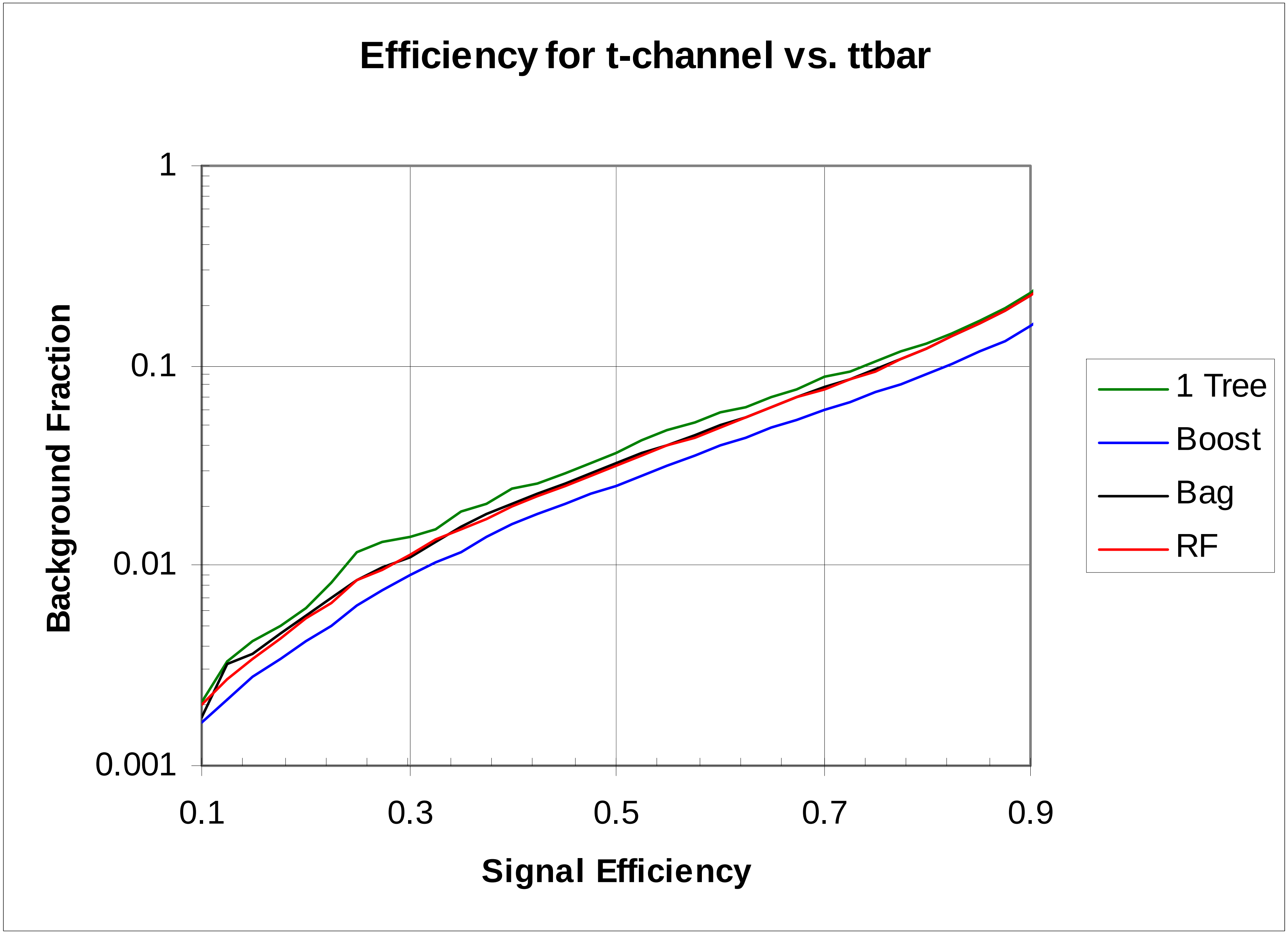


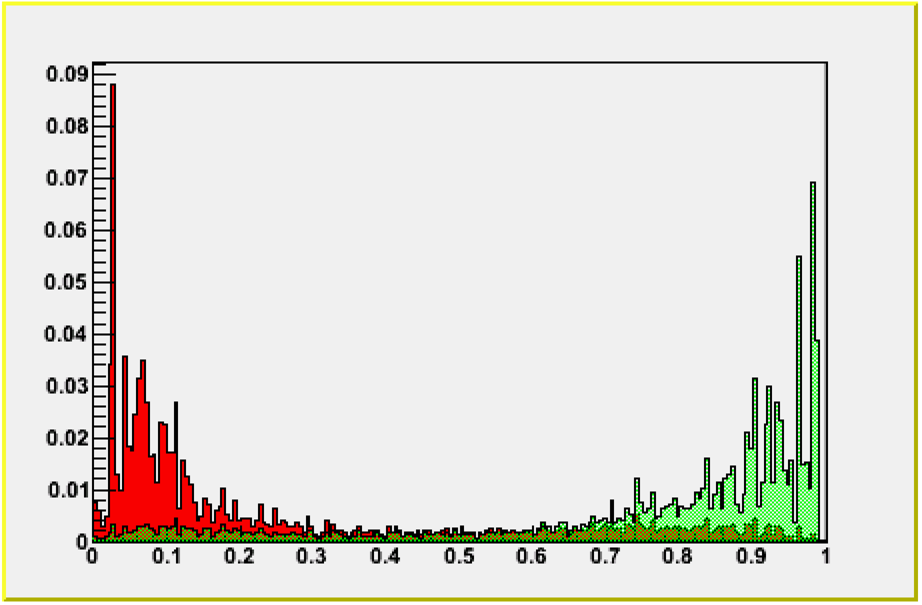
Random Forest

- Like bagging, train with random subsets of data
- Additionally, add randomness to each node
  - Randomly limit the variables on which tree can split at each node

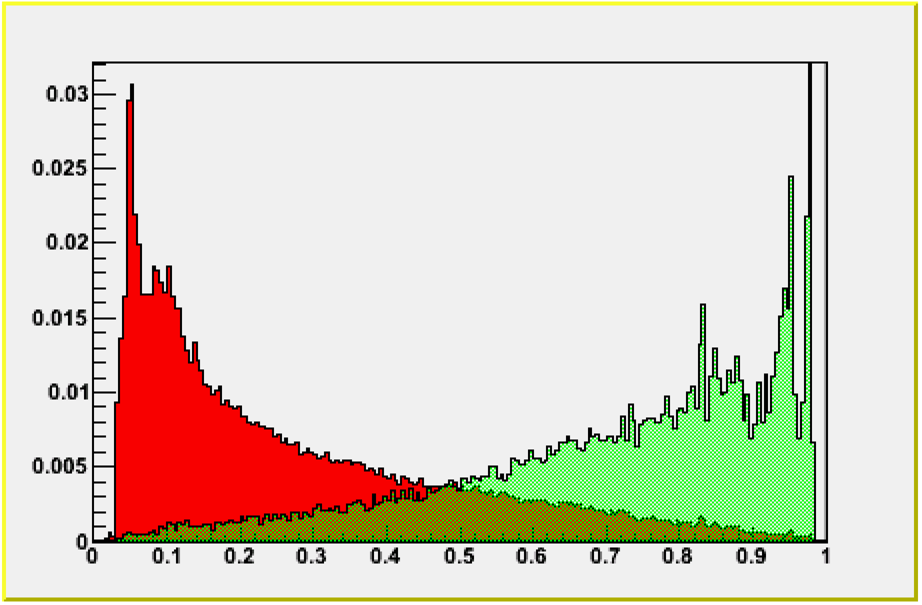


Node	Variables (Ranked)	Variable Used
1	d, (c), b, (a), f, e	d
2	(b), (g), a, (f), e, d	None (a, e are not good enough)
3	a, (b), g, (e), f, d	a
6	(b), (g), e, (f), d, a	None
7	(e), g, (f), b, d, (a)	None

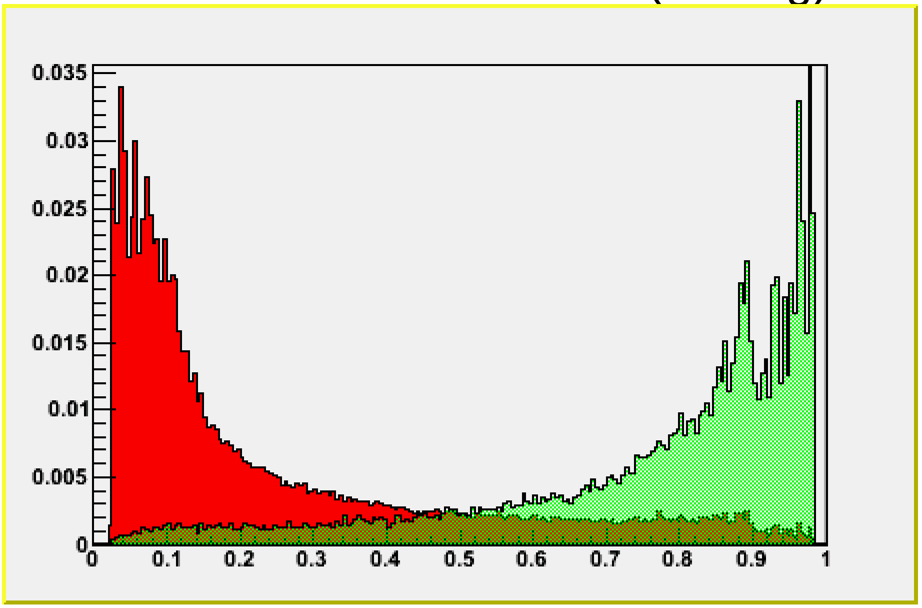

Efficiency for t-channel vs. ttbar
1
0.1
0.01
0.001
Background Fraction
0.1
0.3
0.5
0.7
0.9
Signal Efficiency
1 Tree
Boost
Bag
RF



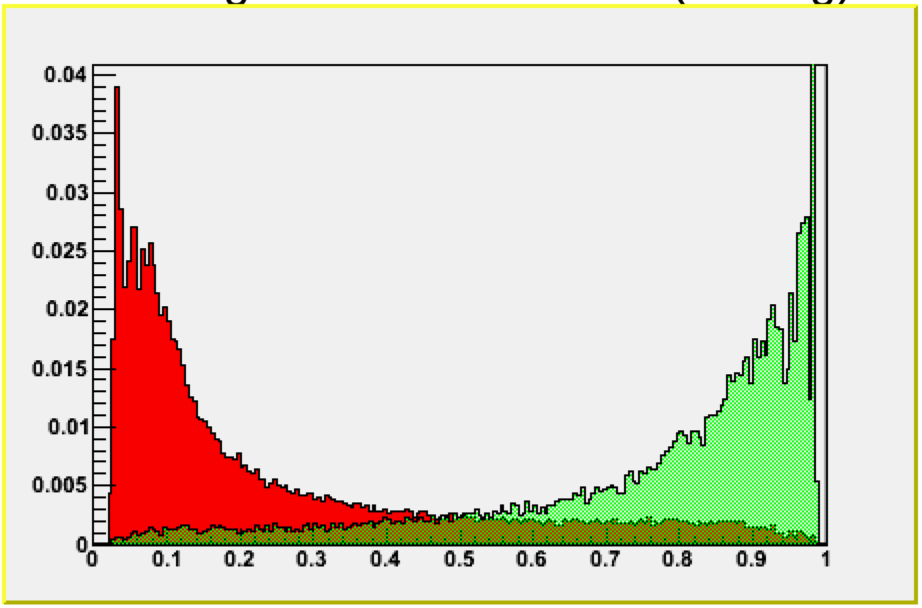
1 Tree: t-channel vs. ttbar (1 b-tag)



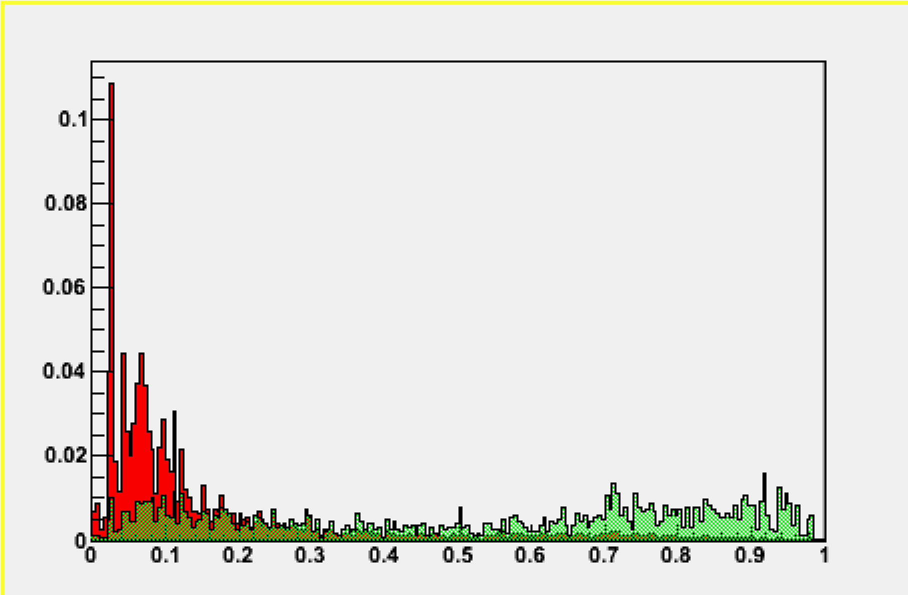
Boosting: t-channel vs. ttbar (1 b-tag)



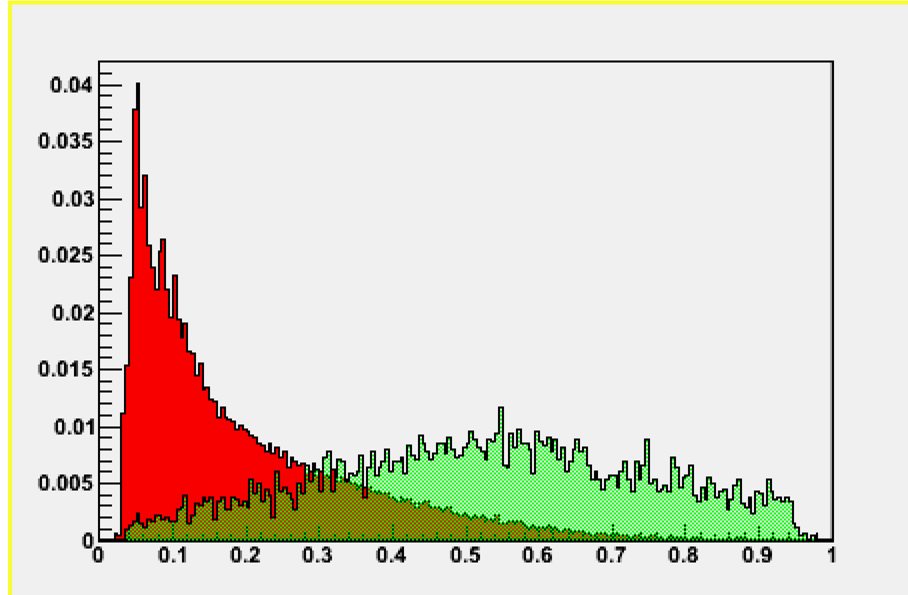
Bagging: t-channel vs. ttbar (1 b-tag)



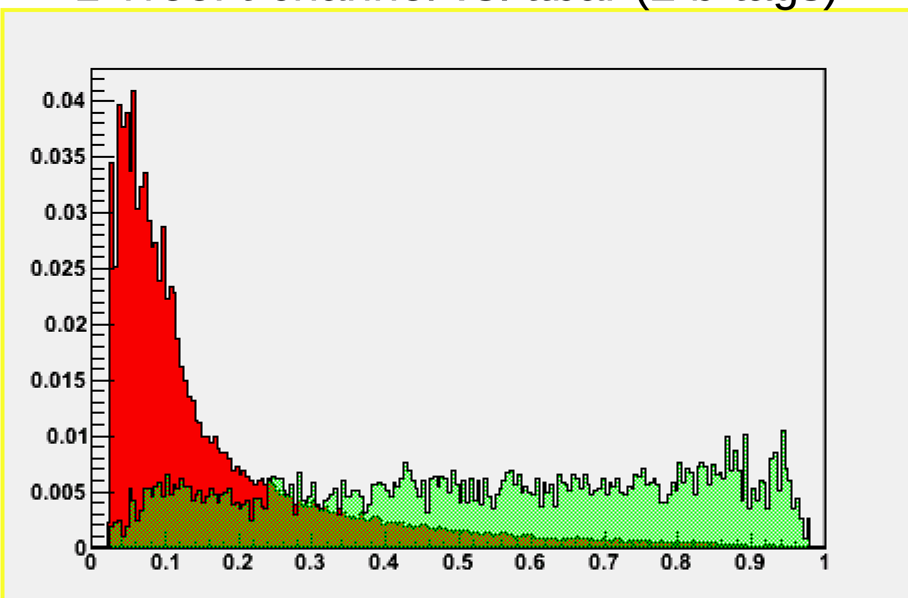
Random Forest: t-channel vs. ttbar (1 b-tag)



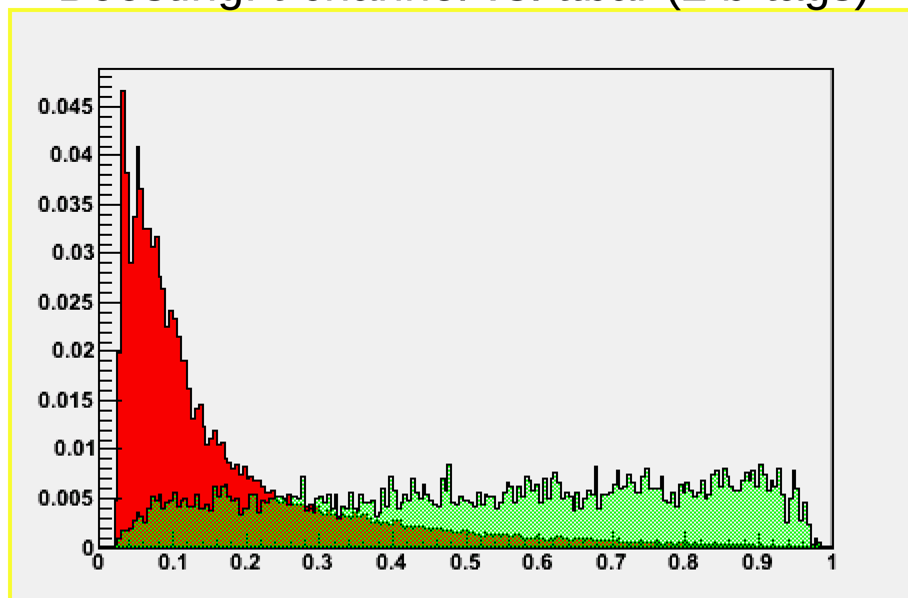
1 Tree: t-channel vs. ttbar (2 b-tags)



Boosting: t-channel vs. ttbar (2 b-tags)



Bagging: t-channel vs. ttbar (2 b-tags)



Random Forest: t-channel vs. ttbar (2 b-tags)

Conclusions & Comments

- Boosting provides best performance
  - Less BG Fraction for given Sig. Efficiency
  - Also runs about twice as fast
    - (Random subset generation slows Bagging, RF)
- Creating different trees for different numbers of b-tags might improve discrimination
- Still need to optimize for size of random subset and also try more trees
  - Bagging/RF may improve over Boosting
  - RF should outperform Bagging, but we don't see this, indicating we need more randomness