Decision Trees and Their Ensembles

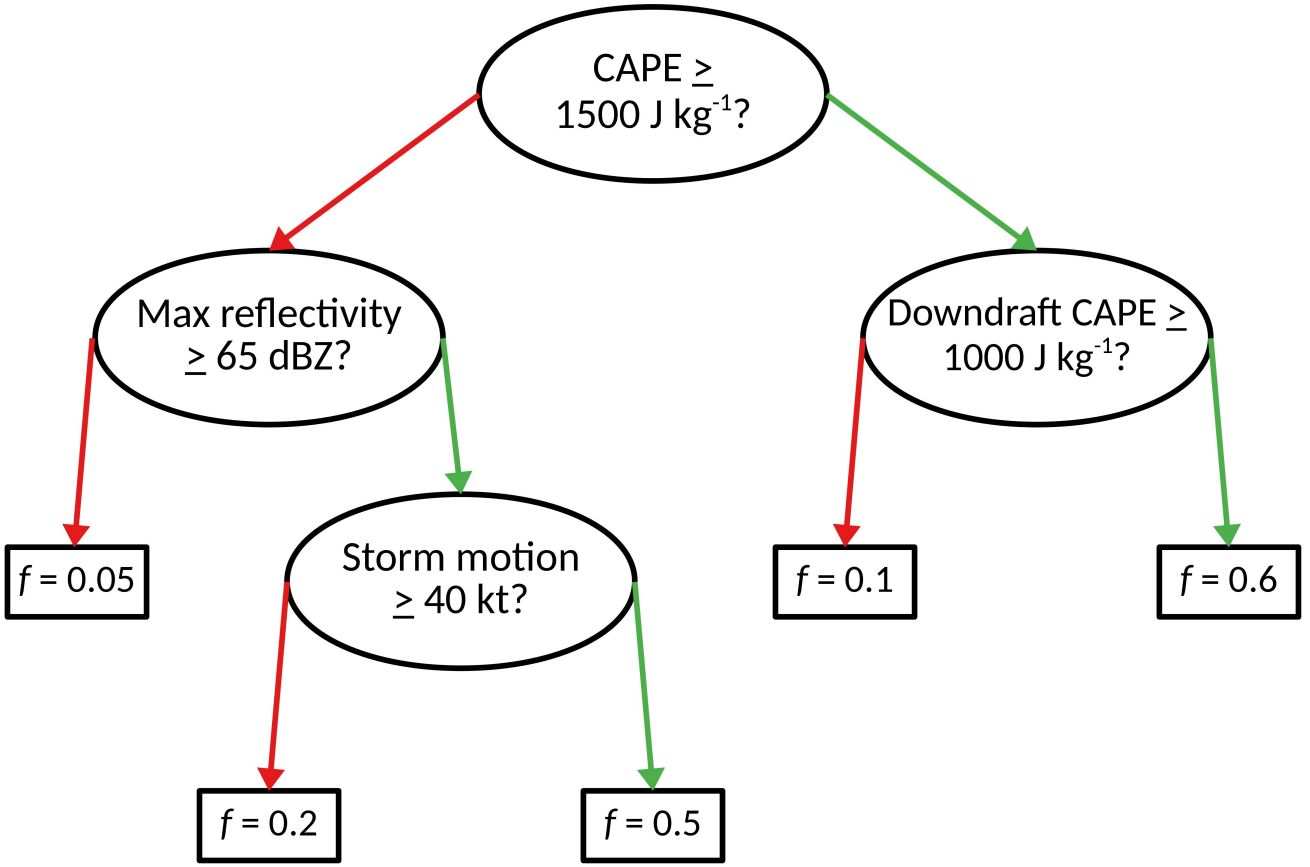
Ryan Lagerquist
(@ralager_Wx, ryan.lagerquist@noaa.gov)

Lecture for NCAR AI summer school

June 22, 2020

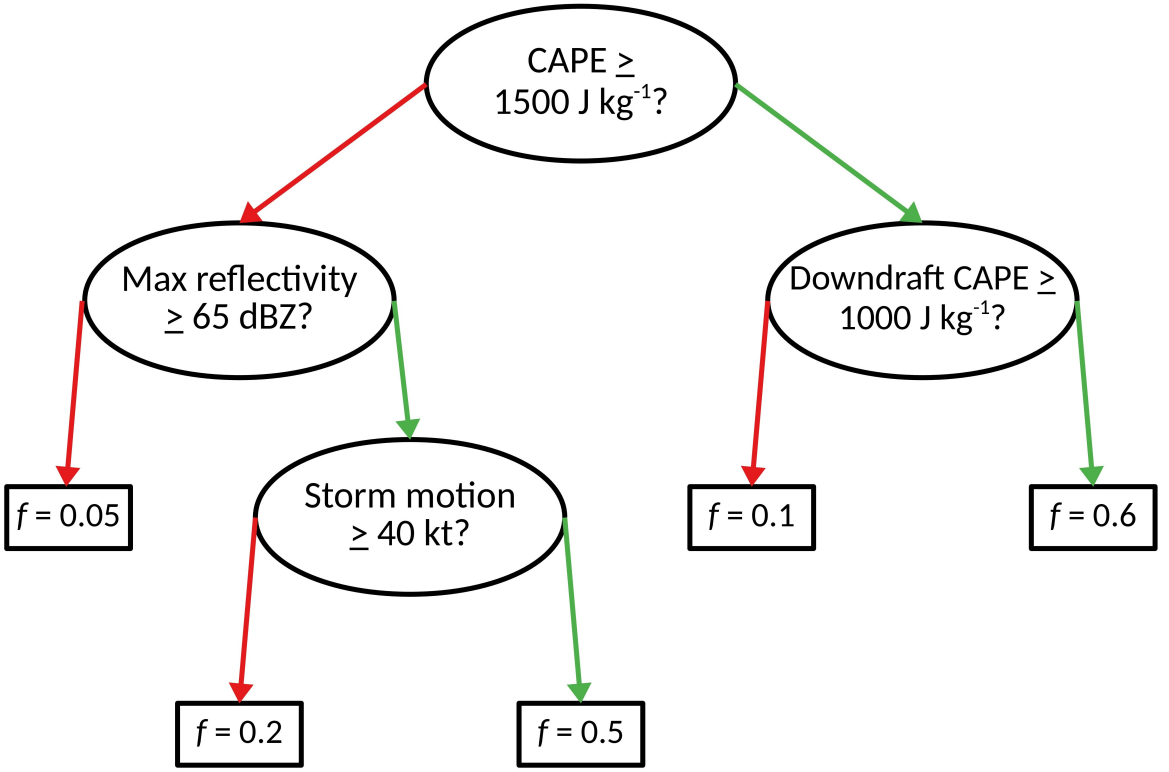
Section 1: Theory

- A decision tree is a flow chart with **branch nodes** (ellipses) and **leaf nodes** (rectangles).
- In the contrived example below, *f* is the predicted probability of severe weather.



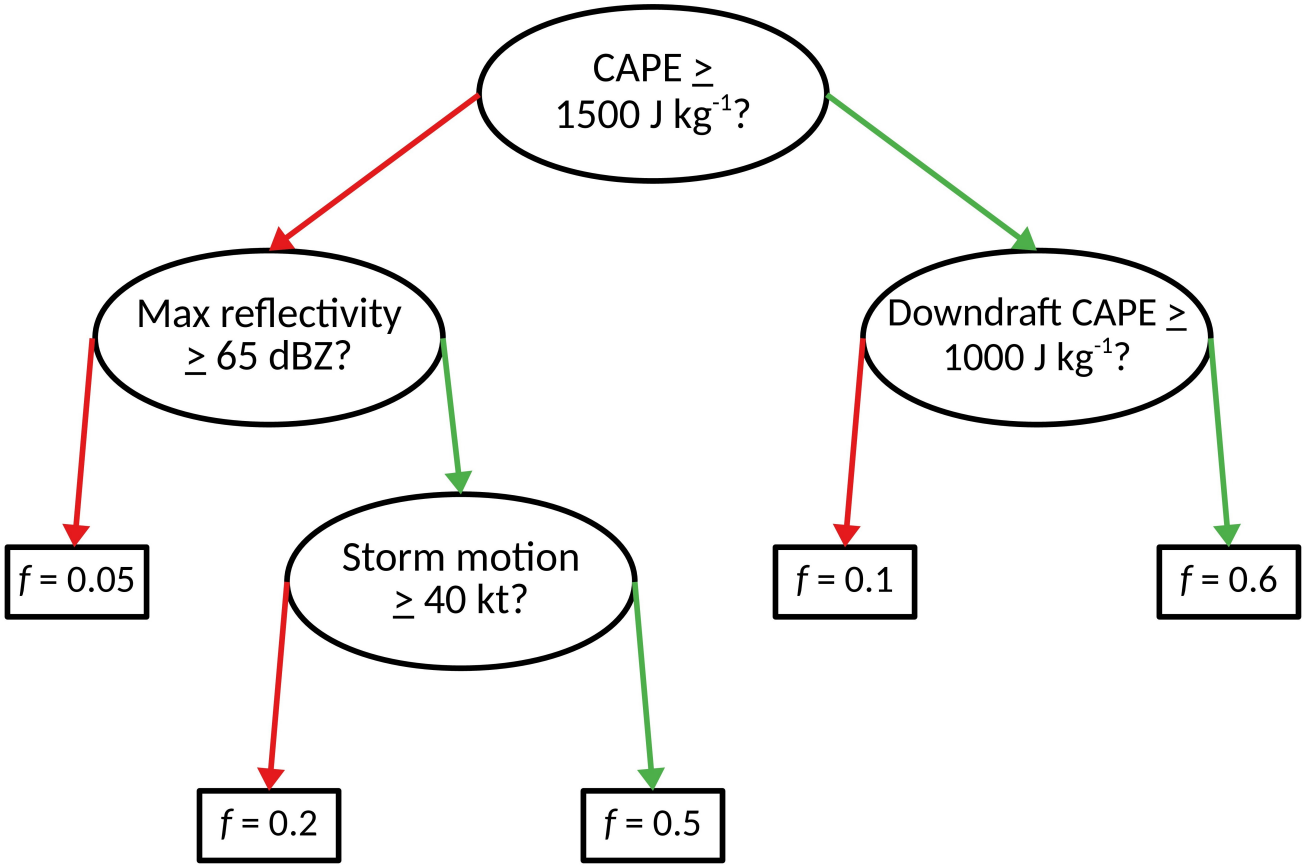
Section 1: Theory

- The branch nodes are bifurcating, and the leaf nodes are terminal.
- In other words, each branch node has 2 children and each leaf node has 0 children.
- **Predictions are made at the leaf nodes, and questions are asked at the branch nodes.**
- Since the branch nodes are bifurcating, **questions asked at the branch nodes must be yes-or-no.**



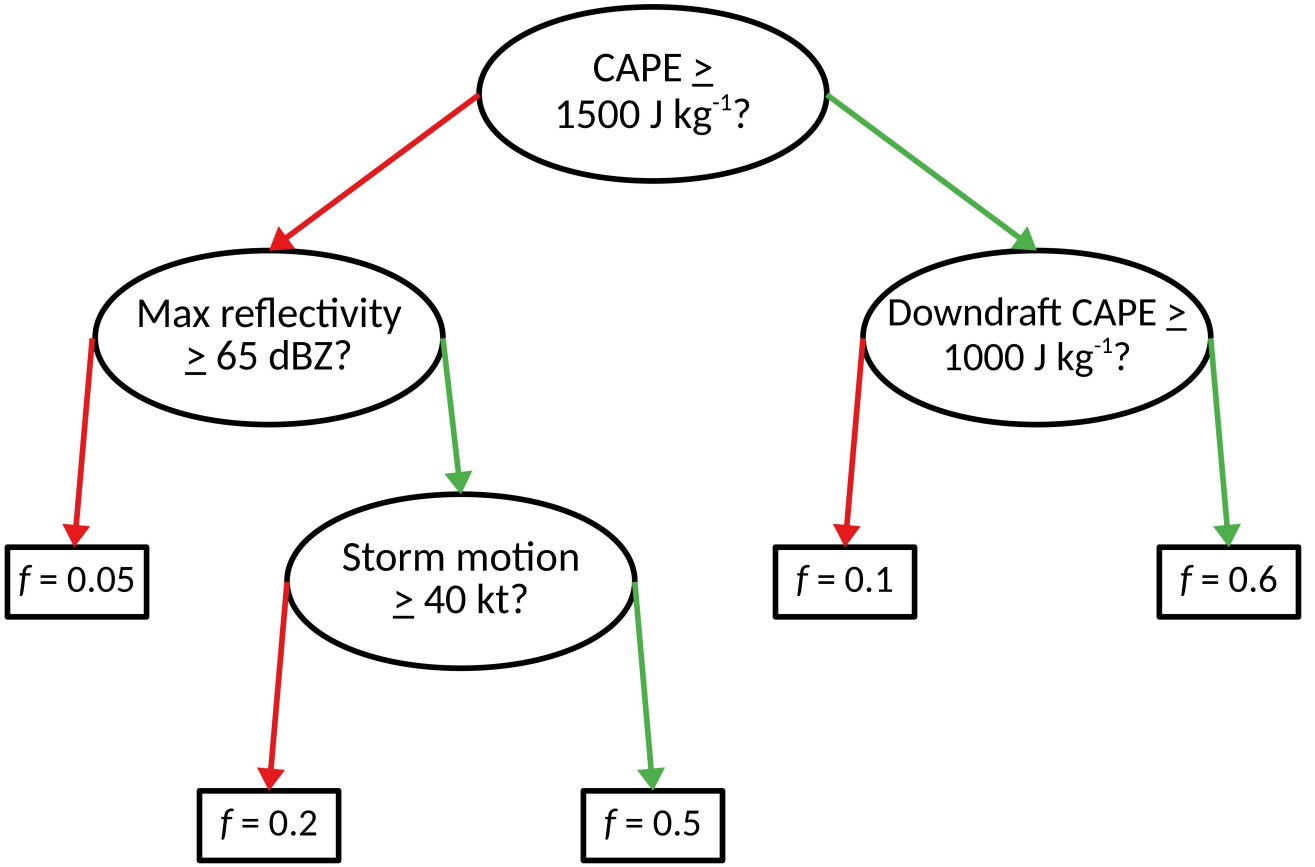
Section 1: Theory

- Decision trees have been used in meteorology since the 1960s (Chisholm 1968).
- They were built subjectively by human experts until the 1980s, when an objective algorithm (Quinlan 1986) was developed to "train" them (determine the best question at each branch node).
- **The prediction at leaf node *L* the average of all training examples that reached *L*.**
- For regression this is a real value (average hail size of storms that reached *L*).
- For classification this is a probability (fraction of storms that reached *L* with severe hail).



Section 1: Theory

- **The question chosen at each branch node is that which maximizes info gain.**
- For regression, this is done by minimizing mean squared error (between predicted and actual values, *e.g.*, of hail size).
- For classification, this is done by minimizing the "remainder," which is based on entropy of the child nodes.



Section 1: Theory

- The entropy of one node is defined below:

$$E = -\frac{1}{n}\left[f \log_2(f) + (1-f)\log_2(1-f)\right]$$

- n = number of examples that reached node
- f = fraction of these examples in the positive class (*e.g.,* severe hail as opposed to non-severe hail)

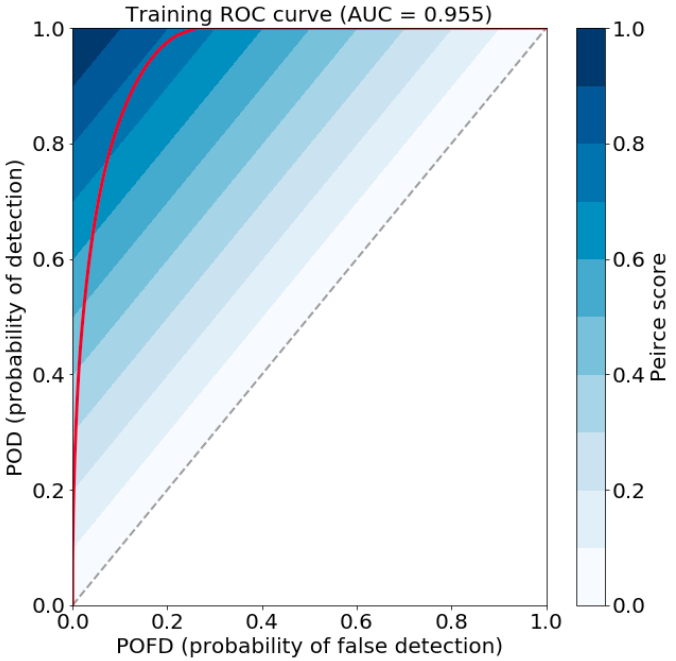
- Then the “remainder” is defined as follows:

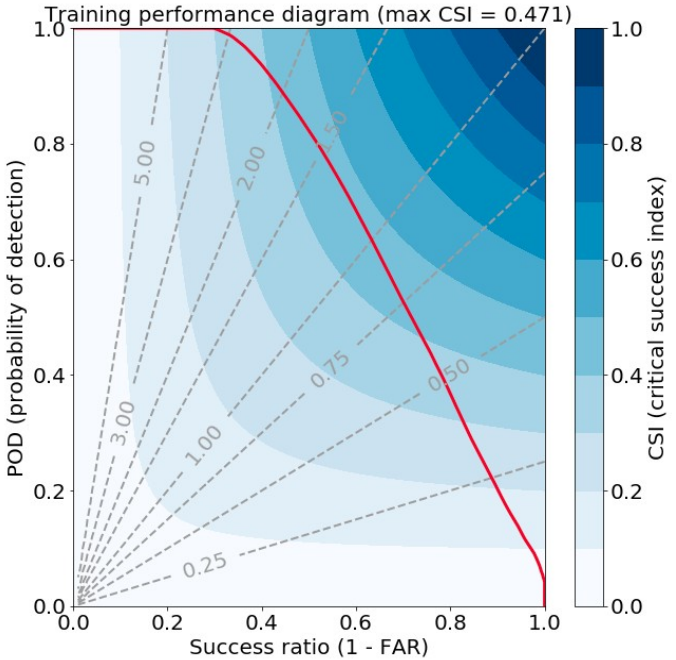
$$R = \frac{n_{\text{left}} E_{\text{left}} + n_{\text{right}} E_{\text{right}}}{n_{\text{left}} + n_{\text{right}}}$$

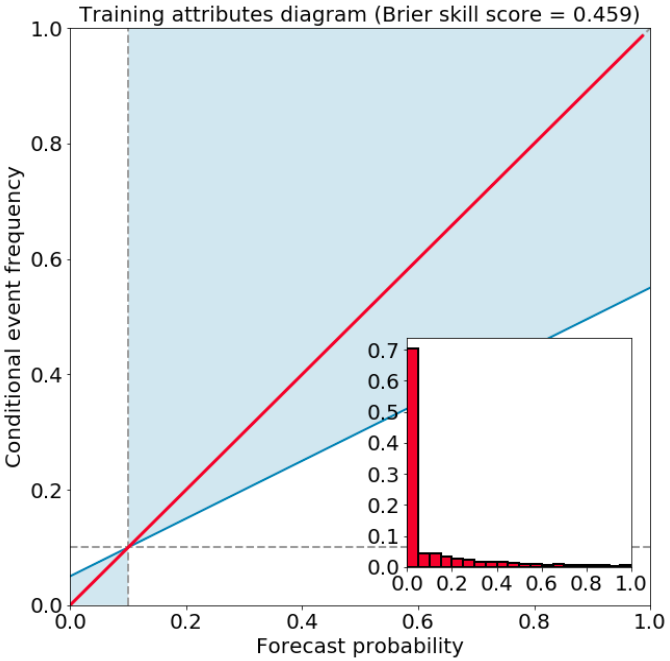
- n_{left} = number of examples sent to left child (for which the answer to the question is "no")
- n_{right} = number of examples sent to right child (for which answer is "yes")
- E_{left} = entropy of left child
- E_{right} = entropy of right child

Section 2: Default Decision Tree

- **Shown below are results on the training data for a default decision tree trained in Python.**
- The tree is trained with `sklearn.tree.DecisionTreeClassifier`, using default input arguments (see code here).
- The tree is trained to predict whether a storm will develop strong rotation (vorticity $\geq$ 0.00385 s^{-1}) in the future.



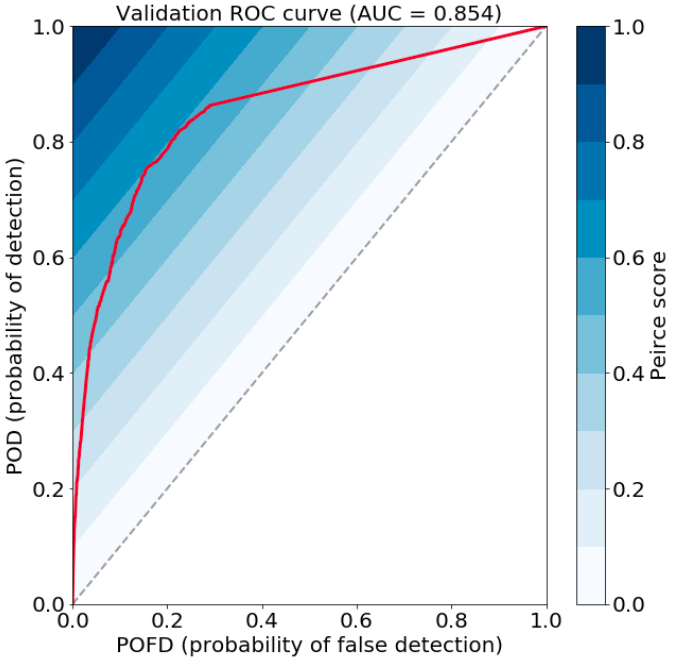


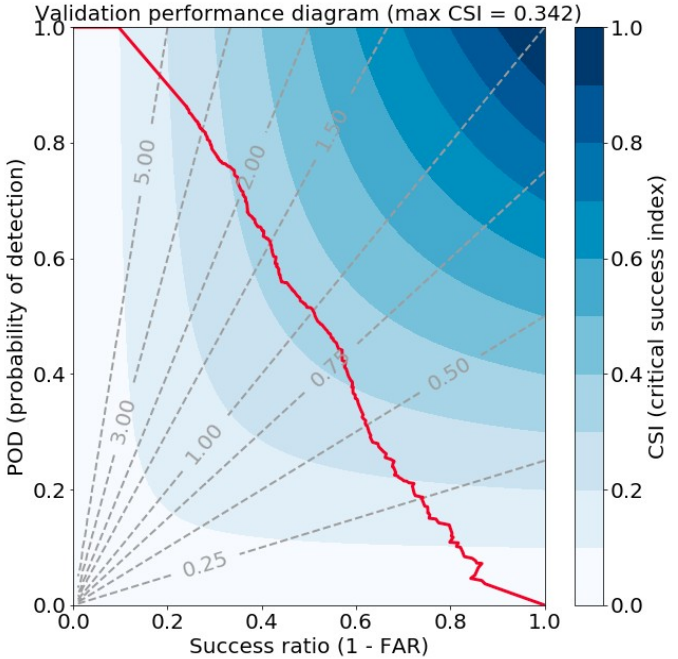


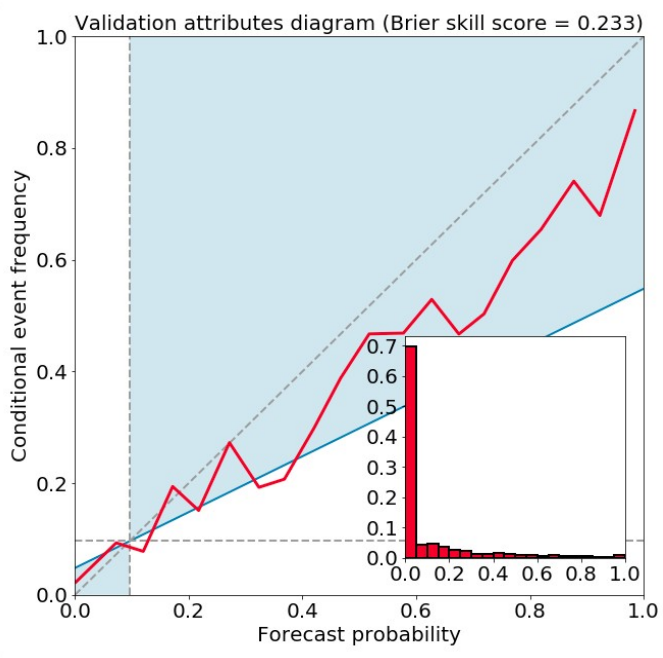
Section 2: Default Decision Tree

- **Below: results on validation data for same tree.**

- **Decrease in skill shows that tree overfit training data strongly:**
  - AUC drops by 0.101
  - Maximum CSI (shown in performance diagram) drops by 0.129
  - Brier skill score (shown in attributes diagram) drops by 0.226
  - Reliability (shown in attributes diagram) goes from perfect to consistent overprediction





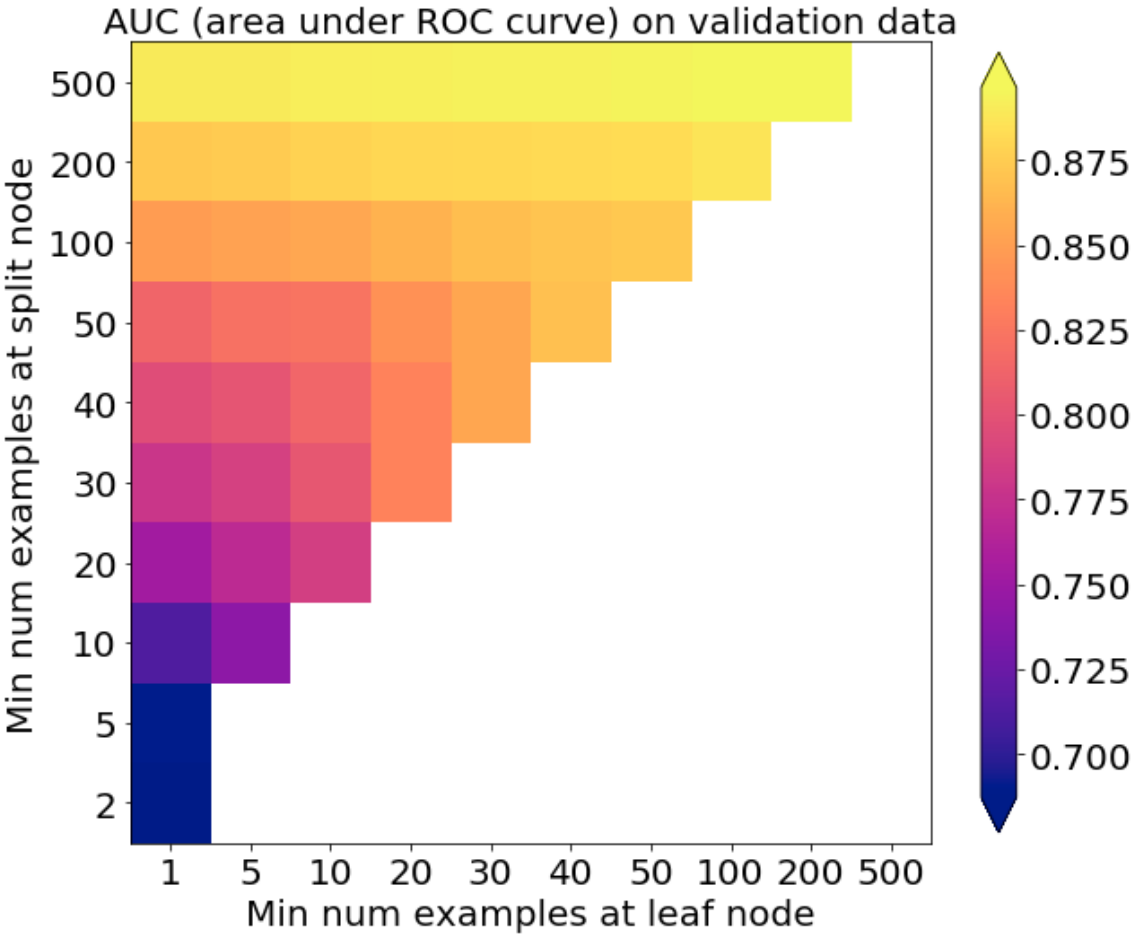


Section 3: Fancier Decision Tree

- **Overfitting can be controlled by limiting the depth of the tree.**
- **Two hyperparameters (among others) control the depth of a decision tree:**
  - Minimum sample size (number of examples) at a branch node
  - Minimum sample size at a leaf node
- If both values are set to 1, the tree can become very deep, making it easier to overfit.
- You can think of this another way:
  - If there is only one example per leaf node, all predictions will be based on only one training example.
  - These predictions will probably not generalize well outside of training data.
- If minimum sample sizes are too high, the tree will not be deep enough, causing it to underfit.

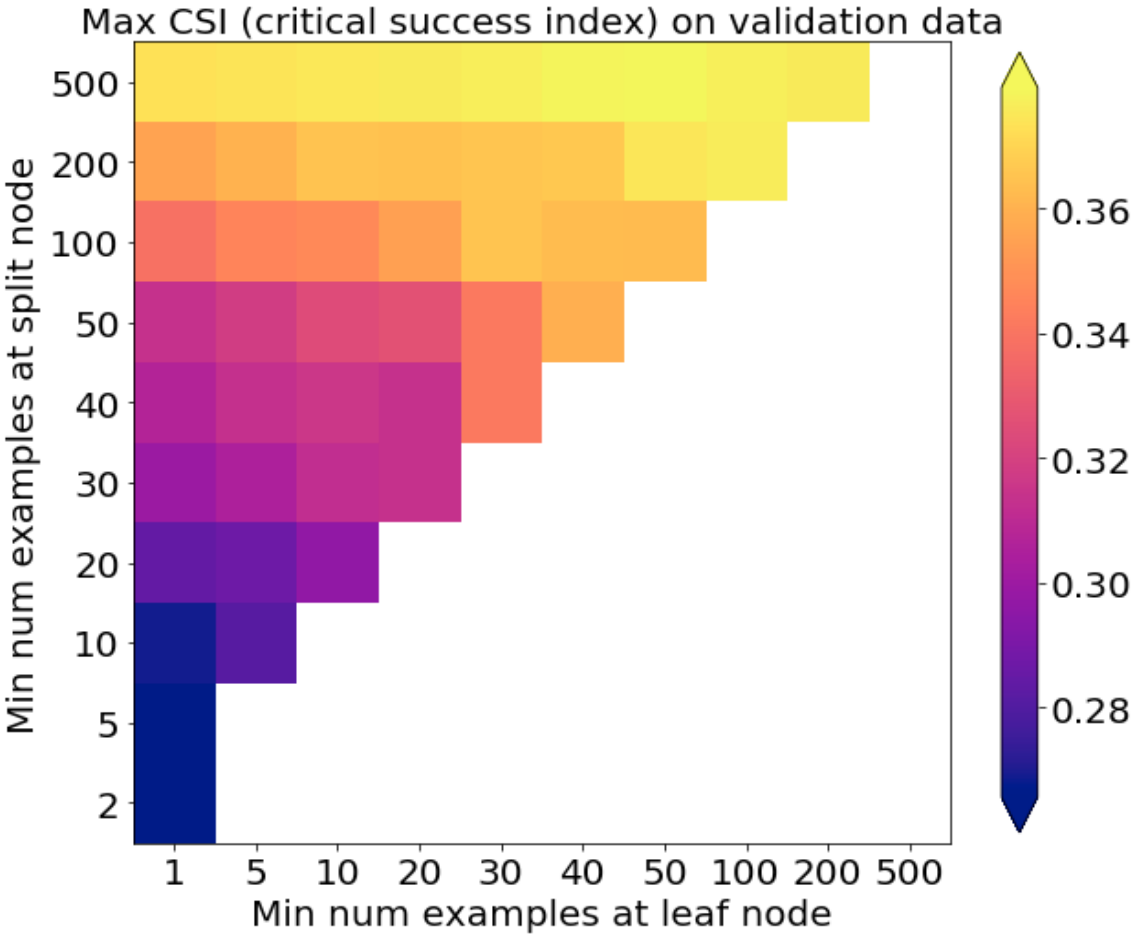
Section 3: Fancier Decision Tree

- I played with different minimum sample sizes to find the combo that works best on validation data.
- This is called a “hyperparameter experiment”.



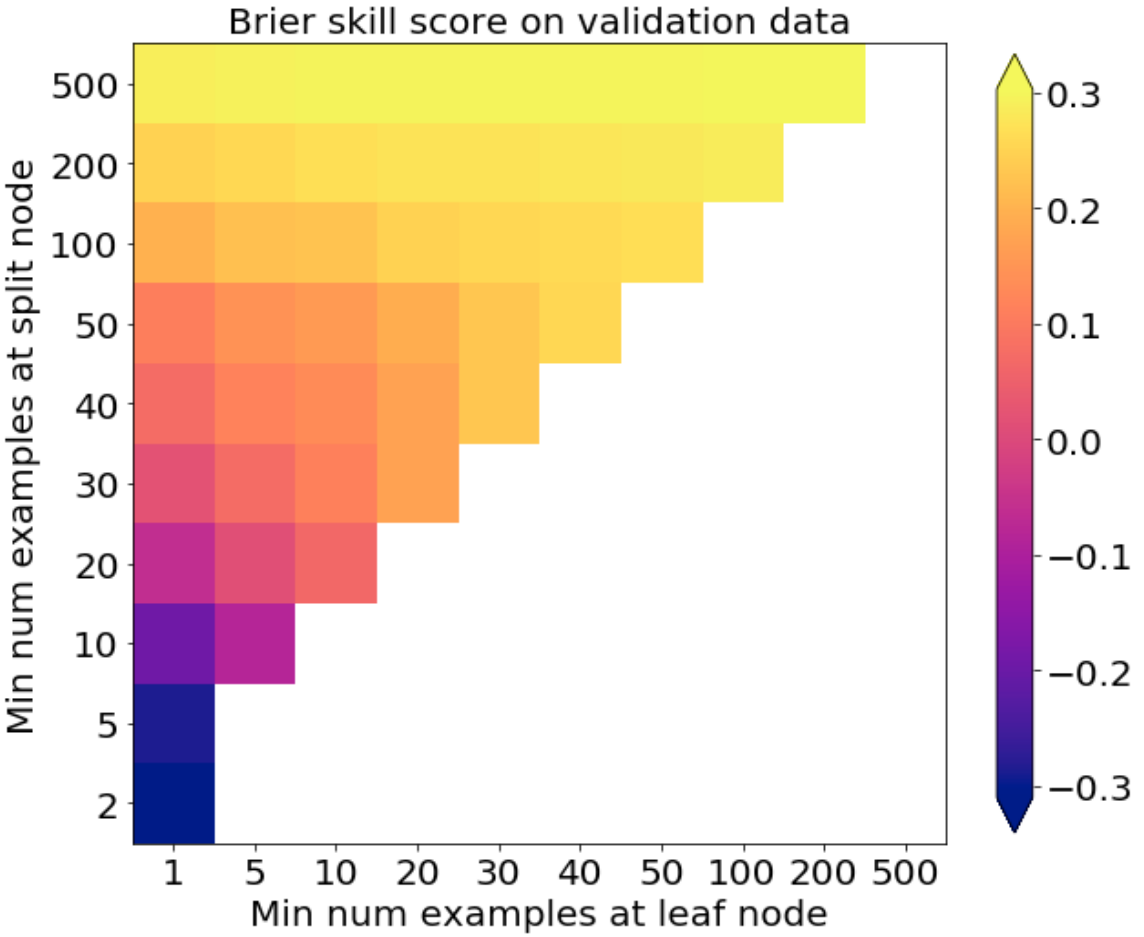
Section 3: Fancier Decision Tree

- I played with different minimum sample sizes to find the combo that works best on validation data.
- This is called a “hyperparameter experiment”.



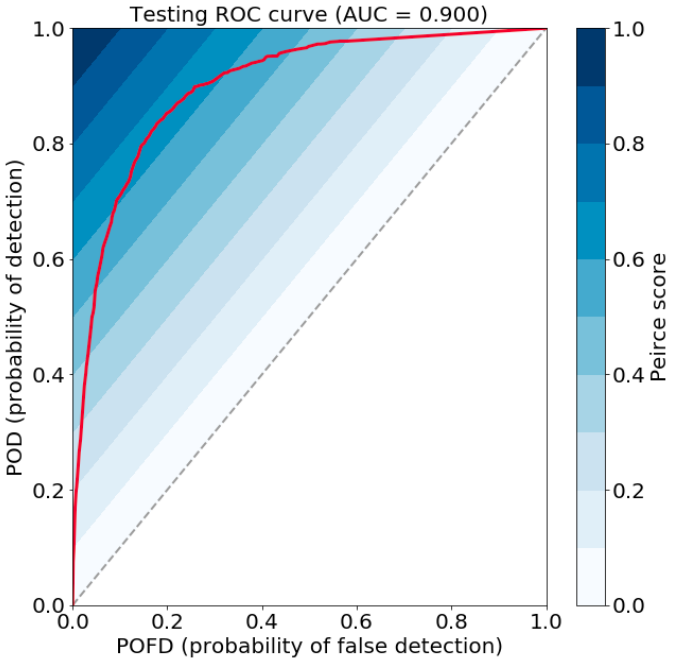
Section 3: Fancier Decision Tree

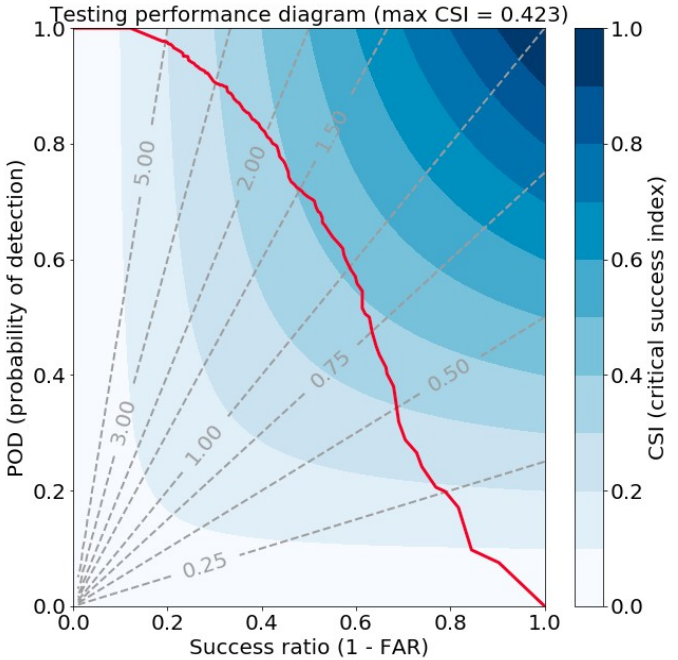
- I played with different minimum sample sizes to find the combo that works best on validation data.
- This is called a “hyperparameter experiment”.

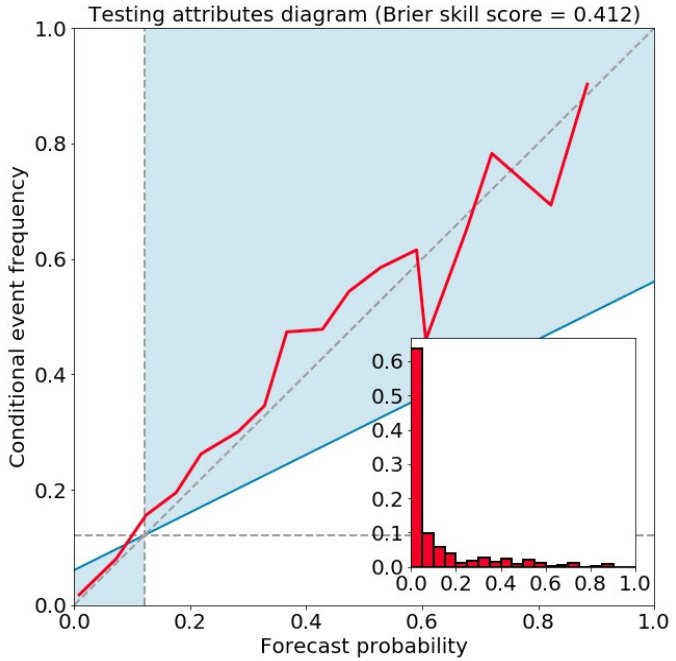


Section 3: Fancier Decision Tree

- I picked the tree with the best (highest) validation BSS:
  - Minimum sample size per branch node = 500
  - Per leaf node = 200
- **Testing results for the new tree are shown below.**
- **The new tree still overfits** (skill drops from training to validation/testing data), **but less than the default tree.**

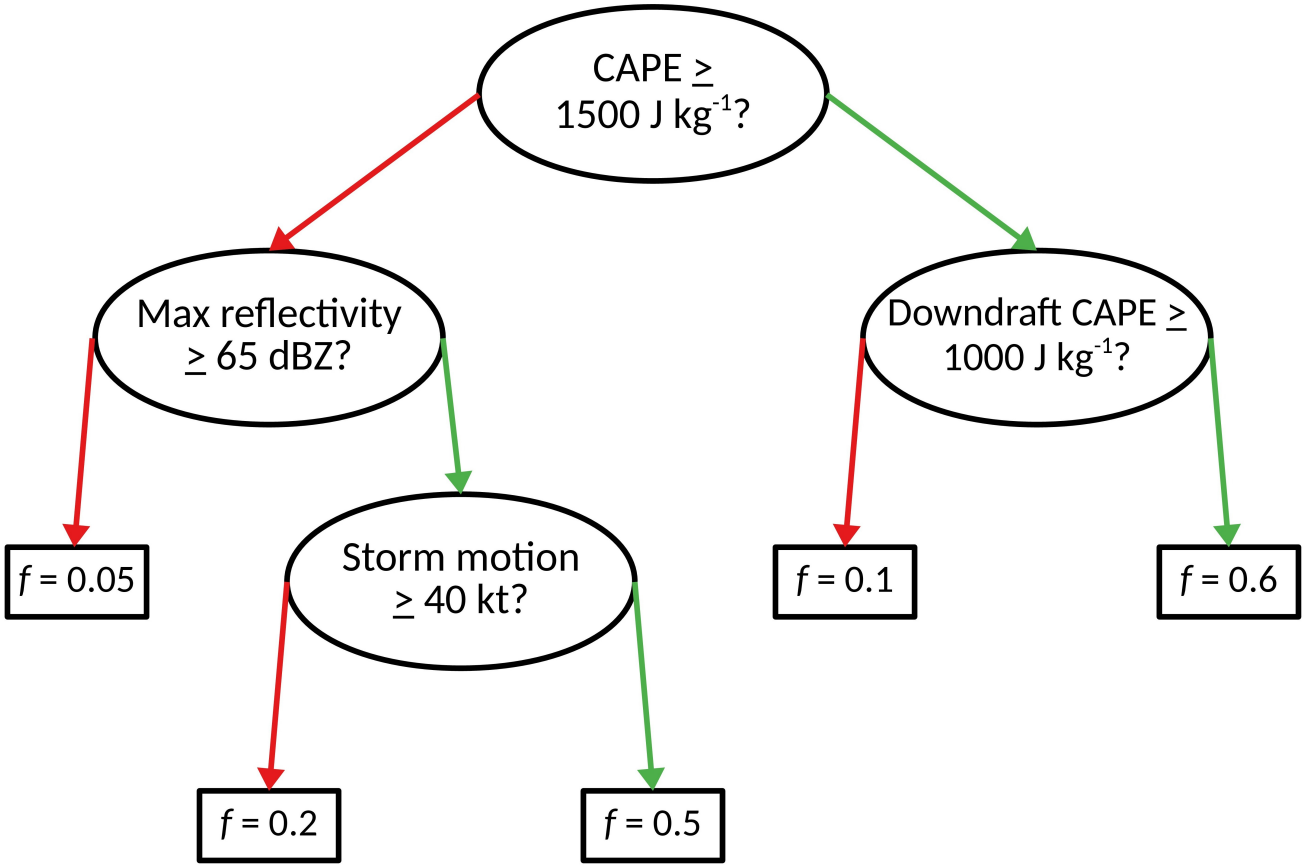






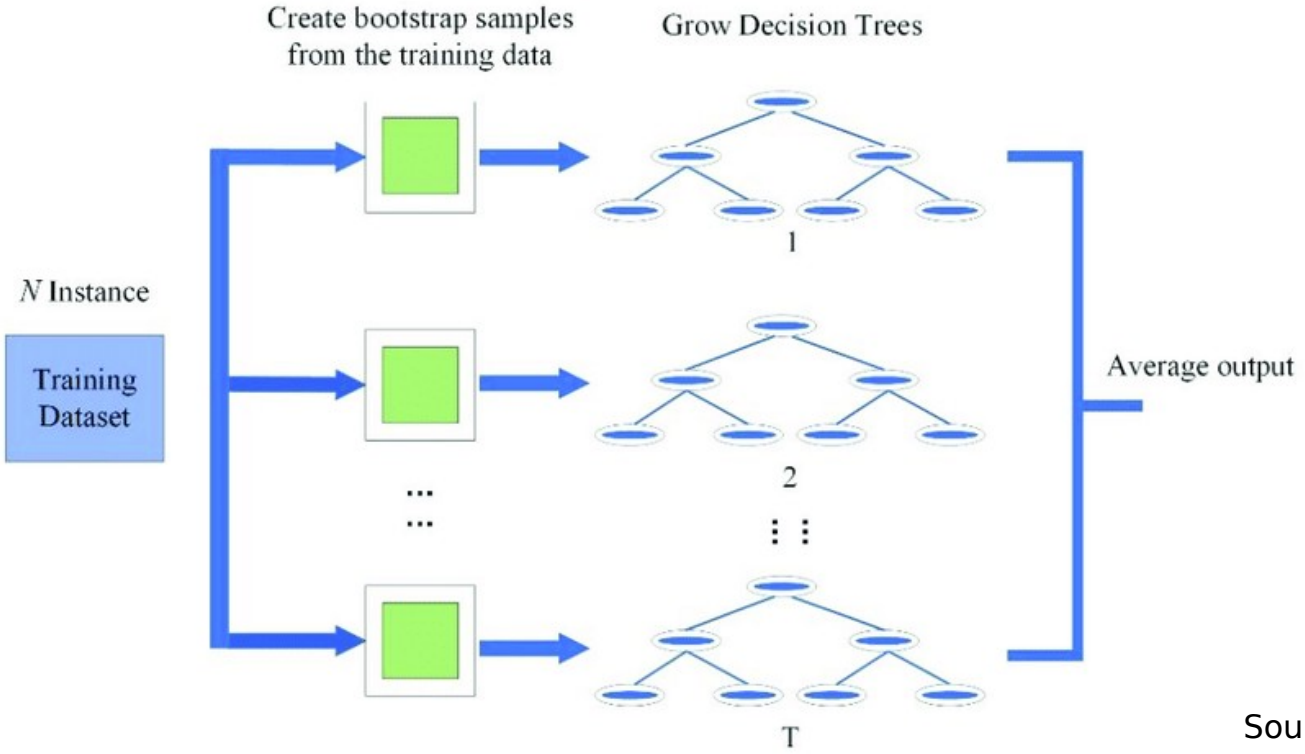
Section 4: Random Forests

- **A random forest (Breiman 2001) is an ensemble of decision trees.**
- In the single decision trees in Section 3, there was a lot of **overfitting**.
- This is a **common problem with decision trees**, because they rely on exact thresholds, which introduce "jumps" into the decision function.
- For example, in the tree shown here, a difference of 0.0001 J kg^{-1} in CAPE could lead to a difference of 55% in severe-Wx probability.



Section 4: Random Forests

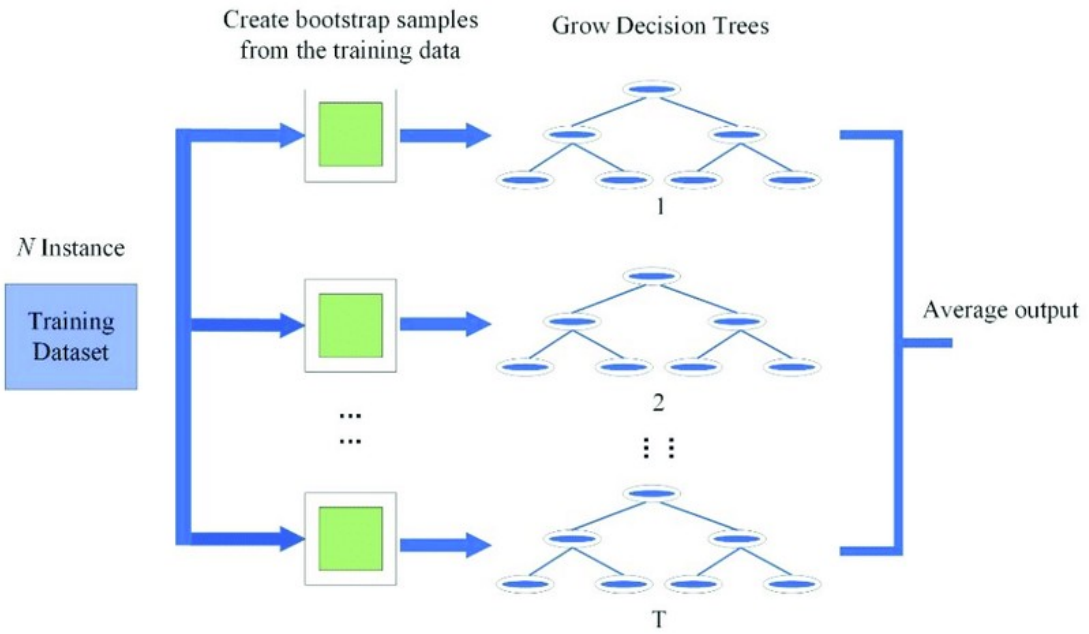
- One way to mitigate this overfitting is: train a bunch of decision trees.
- **If the trees are diverse enough, they will hopefully have offsetting biases** (overfit in different ways).
- **Random forests ensure diversity in two ways:**
  - Example-bagging (sometimes called “bootstrapping”)
  - Predictor-bagging (sometimes called “feature-bagging” or “subsetting”)



Source: Liu *et al.* (2019)

Section 4: Random Forests

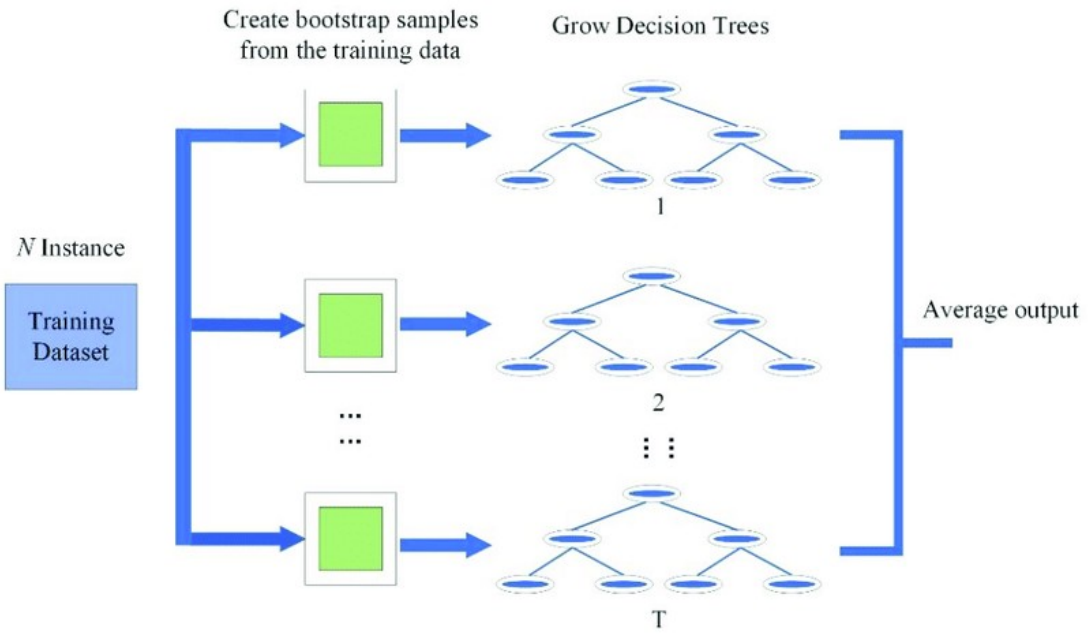
- **Example-bagging is done by training each tree with a bootstrapped replicate of the training data.**
- For a training set with *N* examples, a "bootstrap replicate" is created by randomly sampling *N* examples with replacement.
- Sampling with replacement leads to duplicates. On average, each bootstrapped replicate contains only 63.2% $(1 - e^{-1})$ of unique examples, with the other 37.8% being duplicates.
- **This ensures that each tree is trained with a different set of unique examples.**



Source: Liu *et al.* (2019)

Section 4: Random Forests

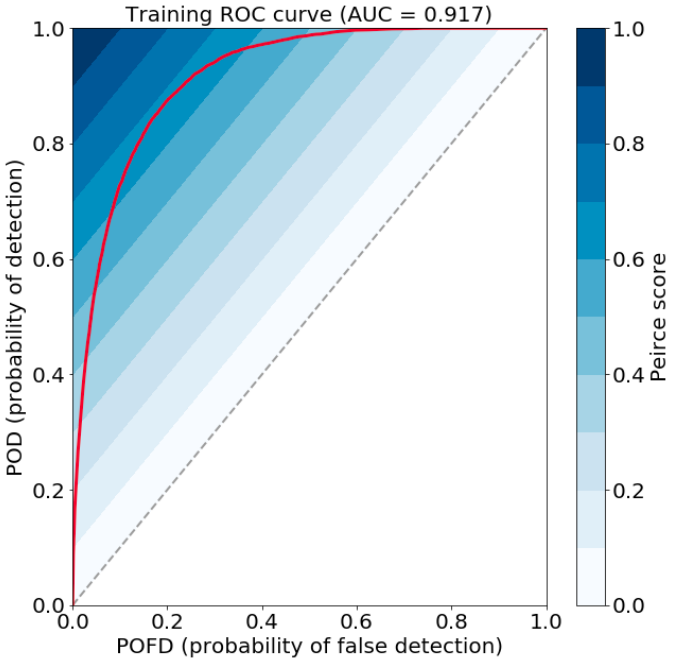
- **Predictor-bagging is done by looping over a random subset of predictors at each branch node.**
- **In other words, instead of trying all predictors to find the best question, try only a few predictors.**
- If there are *M* predictors, the general rule is to try $M^{1/2}$ predictors at each branch node.
- *e.g.,* If there are 41 predictors, each branch node will loop over 6 randomly chosen predictors.

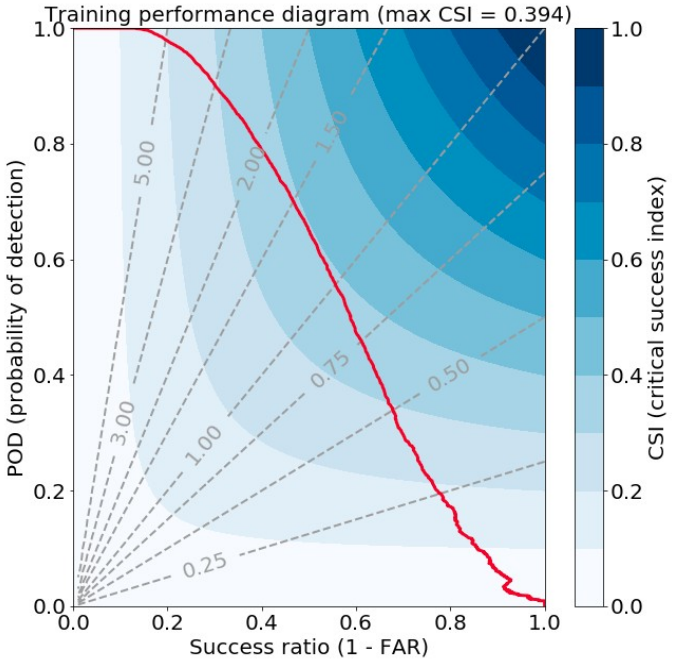


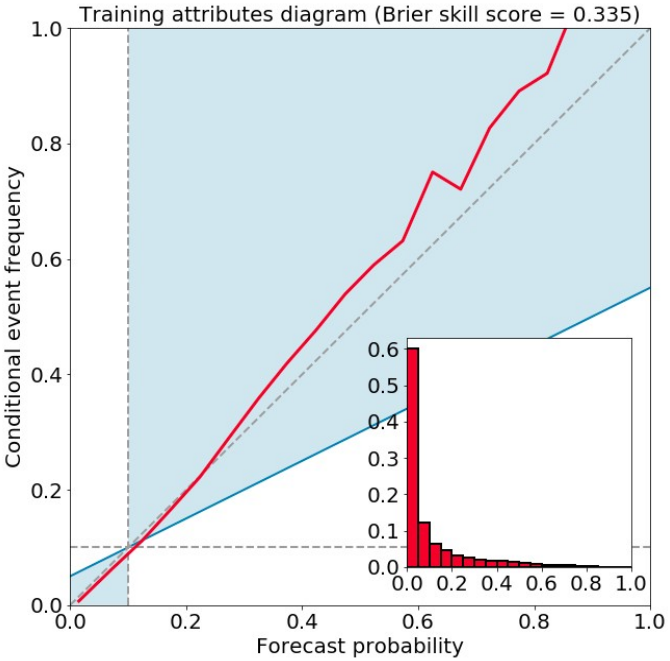
Source: Liu *et al.* (2019)

Section 4: Random Forests

- **Shown below are results on the training data for a random forest trained in Python.**
- The forest is trained with `sklearn.ensemble.RandomForestClassifier` (see code here).
- Like single trees in Section 3, the forest is trained to predict whether a storm will develop strong rotation (vorticity $\geq$ 0.00385 s^{-1}) in the future.

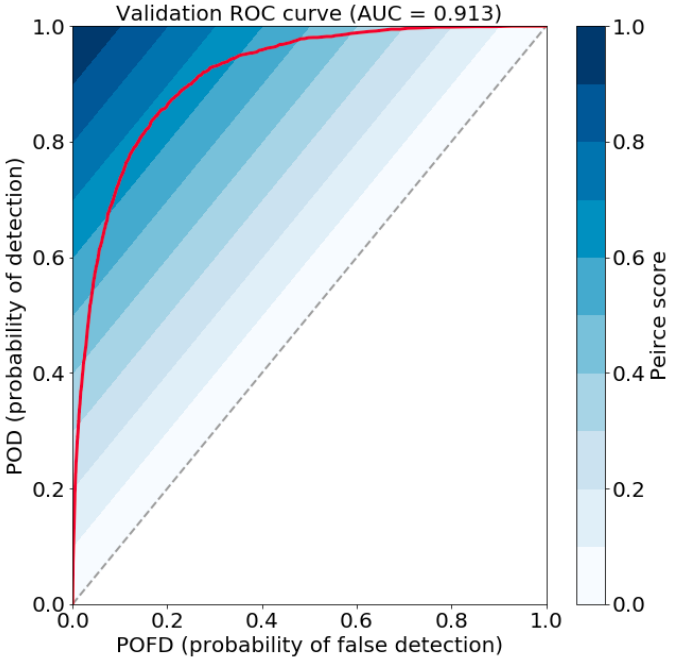


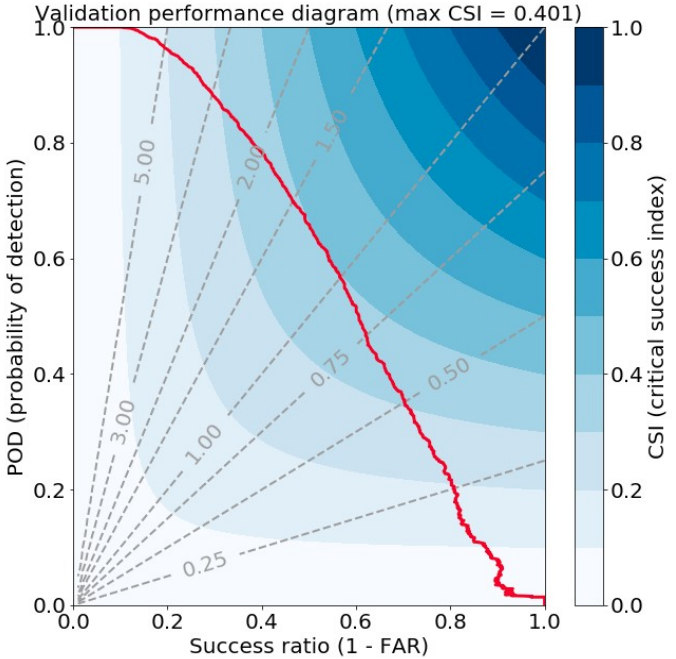


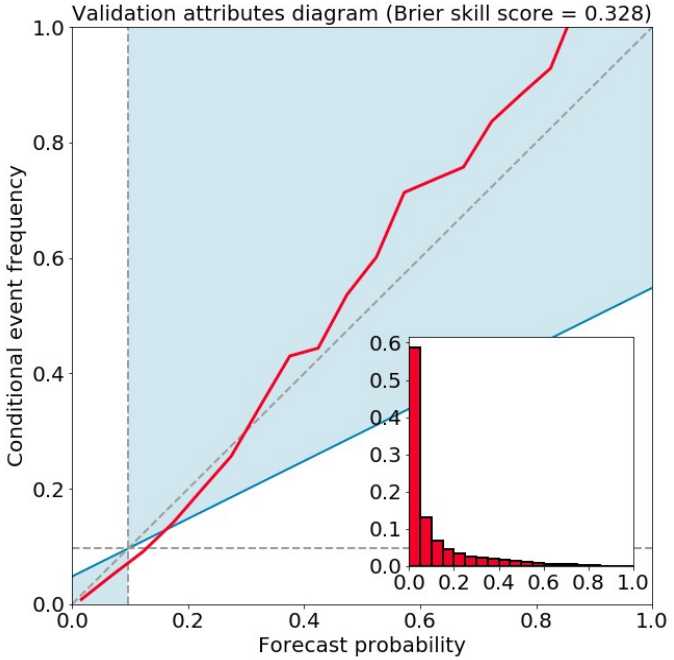


Section 4: Random Forests

- **Below: results on validation data for same forest.**
- **Very small change in skill shows that, unlike single trees, random forest did not overfit badly:**
  - AUC drops by 0.004
  - Maximum CSI (shown in performance diagram) increases by 0.007
  - Brier skill score (shown in attributes diagram) drops by 0.007
  - Likely that none of these differences are statistically significant!

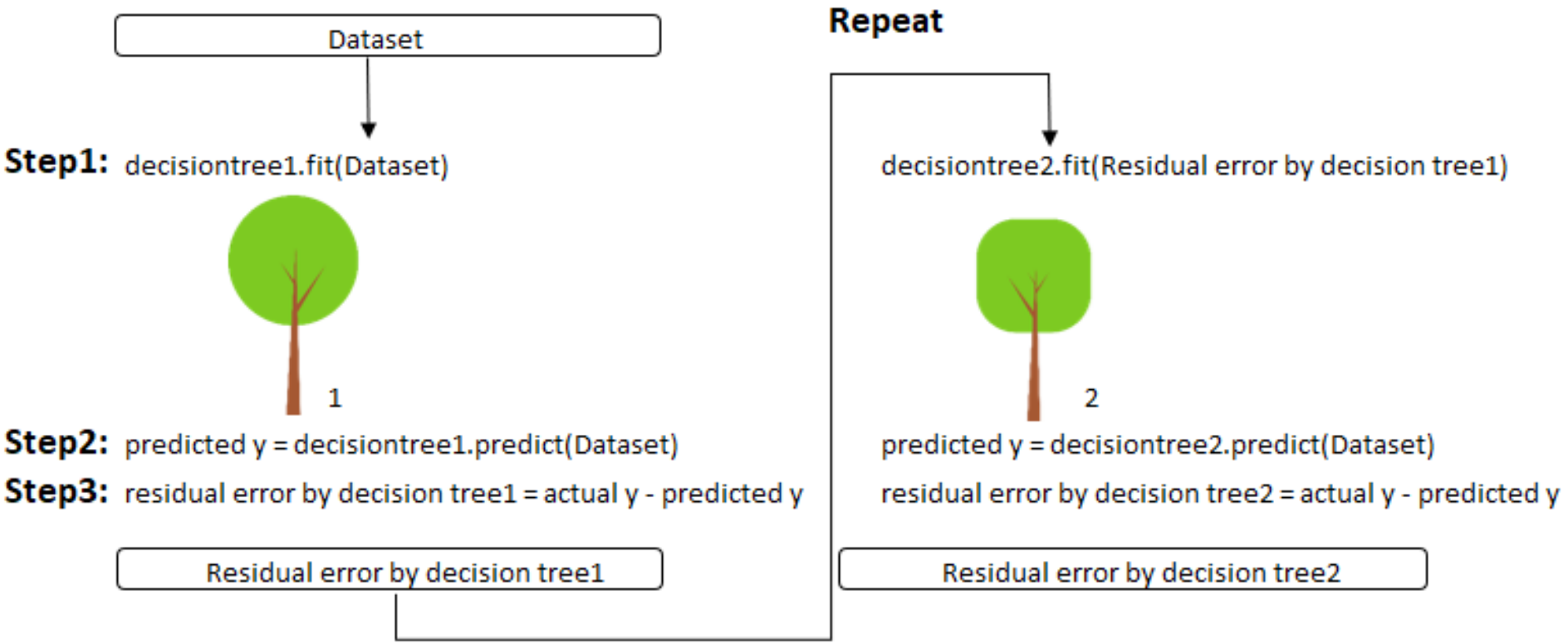






Section 5: Gradient-boosted Forests

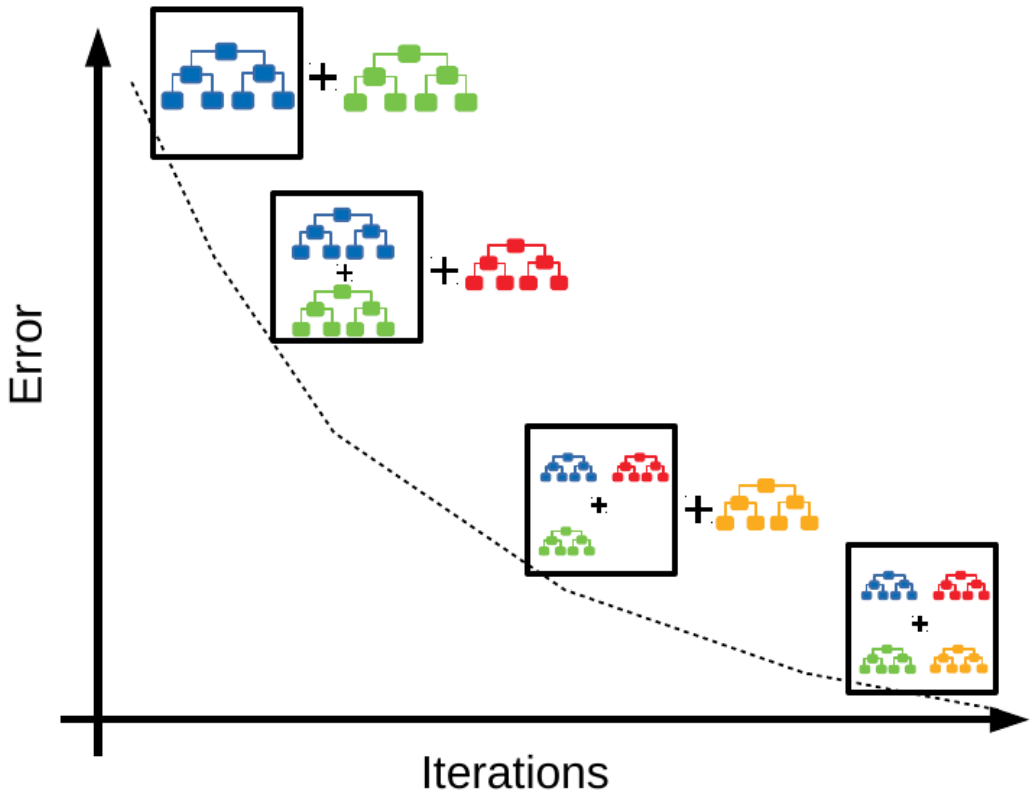
- **Gradient-boosting (Friedman 2002) is another way of ensembling decision trees.**
- In a random forest the trees are trained independently of each other.
- Conversely, in a GBF, **the k^{th} tree is trained to predict the residual from the first k – 1 trees.**



Source: https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

Section 5: Gradient-boosted Forests

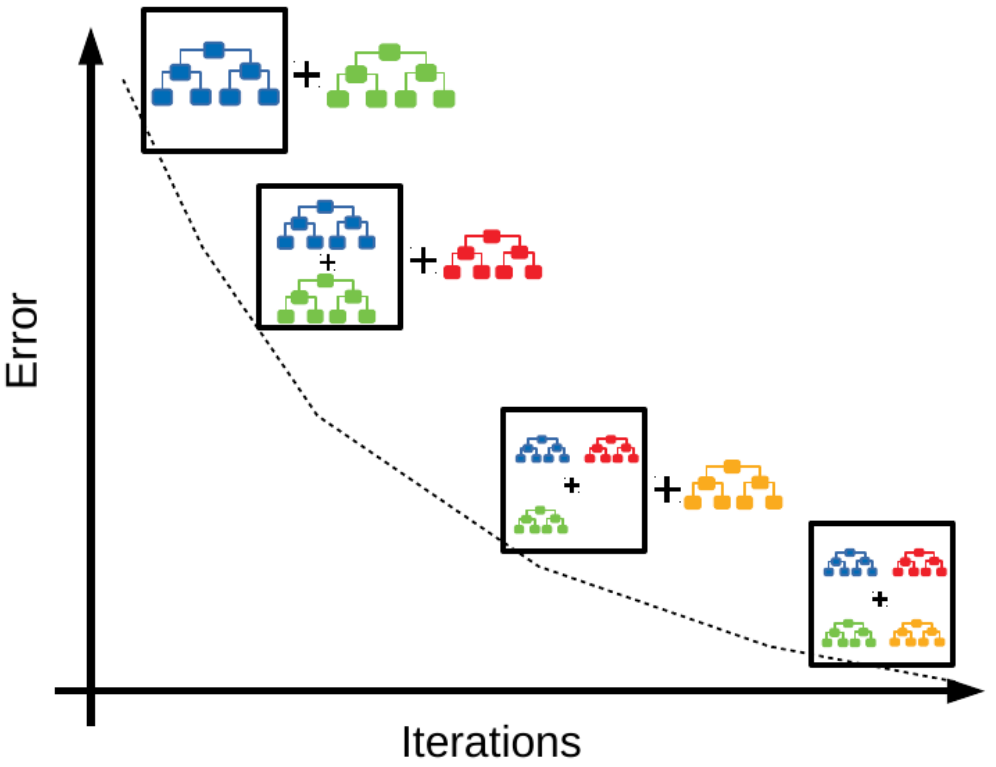
- **Gradient-boosting (Friedman 2002) is another way of ensembling decision trees.**
- In a random forest the trees are trained independently of each other.
- Conversely, in a GBF, **the k^{th} tree is trained to predict the residual from the first k – 1 trees.**



Source: http://tvas.me/articles/2019/08/26/Block-Distributed-Gradient-Boosted-Trees.html

Section 5: Gradient-boosted Forests

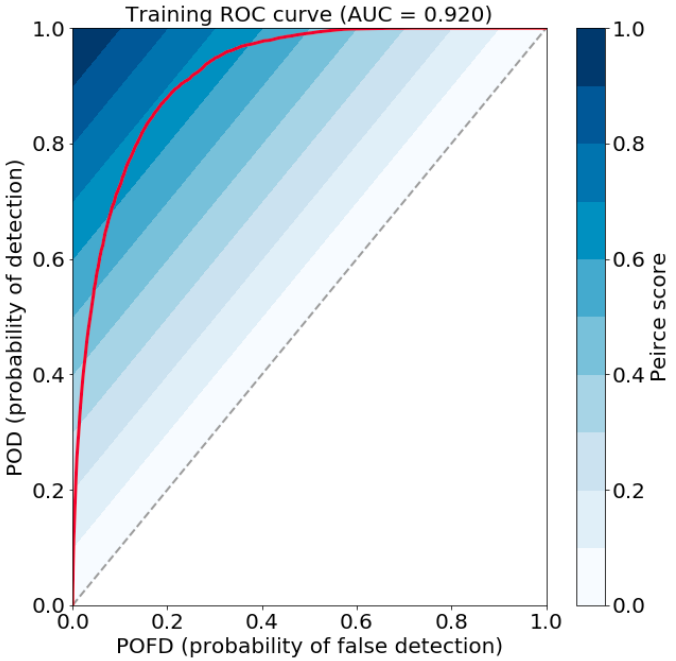
- GBFs can still use example-bagging and predictor-bagging.
- However, in most libraries the default is no example-bagging or predictor-bagging.
  - *i.e.,* Train each tree with all examples and attempt all predictors at each branch node.
- **In a random forest the trees can be trained in parallel** (each tree is independent of the others), **which makes random forests faster.**
- **In a GBF the trees must be trained in series, which makes them slower.**
- **However, in practice GBFs usually outperform random forests.**
- In a recent contest for solar-energy prediction, the top 3 teams all used GBFs (McGovern *et al.* 2015).

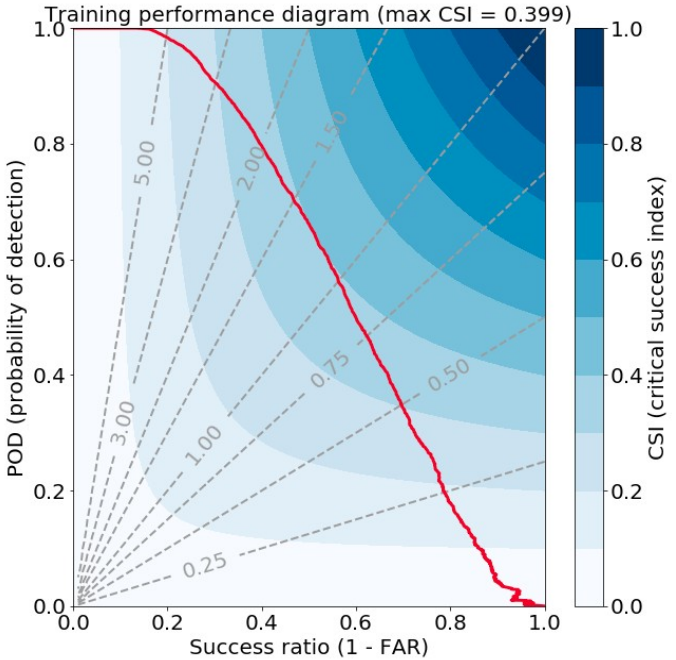


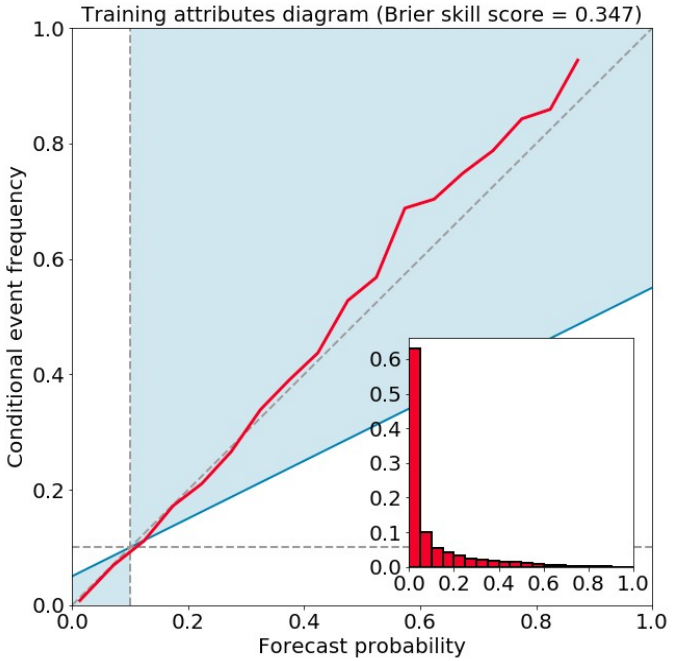
Source: http://tvas.me/articles/2019/08/26/Block-Distributed-Gradient-Boosted-Trees.html

Section 5: Gradient-boosted Forests

- **Shown below are results on the training data for a GBF trained in Python.**
- The forest is trained with `sklearn.ensemble.GradientBoostingClassifier` (see code here).
- Like single trees in Section 3 and random forests in Section 4, GBF is trained to predict whether a storm will develop strong rotation (vorticity ≥ 0.00385 s^{-1}) in the future.

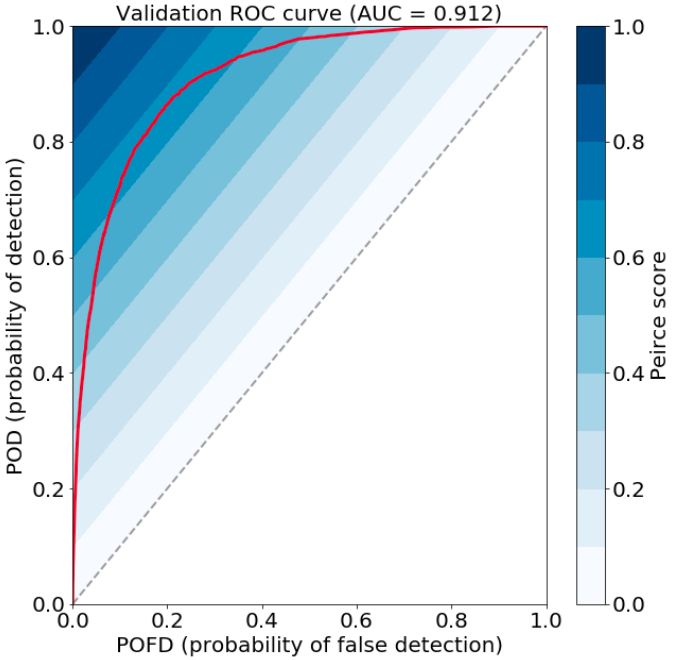


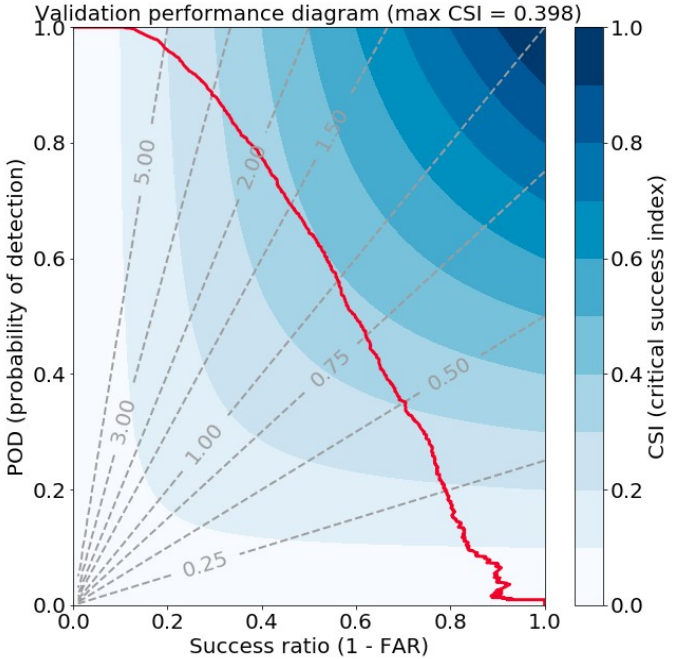


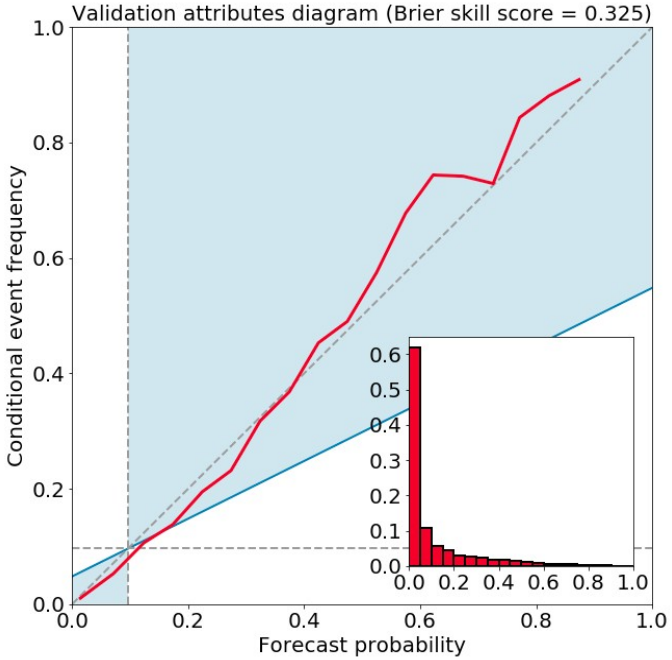


Section 5: Gradient-boosted Forests

- **Below: results on validation data for same GBF.**
- **Like the random forest but unlike single trees, GBF did not overfit badly:**
  - AUC drops by 0.008
  - Maximum CSI (shown in performance diagram) increases by 0.001
  - Brier skill score (shown in attributes diagram) drops by 0.022
  - Likely that none of these differences are statistically significant!
- Validation results slightly worse for GBF than for random forest (but again, differences probably not significant).







Section 6: Summary

- Decision trees can solve a regression or classification problem, using a series of yes-or-no questions.
- Main advantage of decision trees: human-readability.
- Main disadvantage: they commonly overfit.
- Overfitting can be mitigated by ensembling decision trees, using a random or gradient-boosted forest.
- Disadvantage of forests: not human-readable.
  - The individual trees are, but there are usually hundreds of trees.
- You can find interactive code for all the experiments shown here:
  https://github.com/djgagne/ams-ml-python-course/blob/master/module_2/ML_Short_Course_Module_2_Basic.ipynb
- Applications of decision trees and forests in atmospheric science:
  - https://link.springer.com/content/pdf/10.1007/s10994-013-5346-7.pdf
  - https://link.springer.com/content/pdf/10.1007/s10994-013-5343-x.pdf
  - https://journals.ametsoc.org/waf/article/30/6/1781/40289
  - https://journals.ametsoc.org/waf/article/32/6/2175/41018
  - https://journals.ametsoc.org/jamc/article/57/7/1575/68277
  - https://journals.ametsoc.org/waf/article/35/2/537/345548
  - https://journals.ametsoc.org/waf/article/32/5/1819/41181

References

Breiman, L., 2001: “Random forests.” *Machine Learning*, **45 (1)**, 5–32, https://link.springer.com/content/pdf/10.1023/A:1010933404324.pdf.

Chisholm, D., J. Ball, K. Veigas, and P. Luty, 1968: "The diagnosis of upper-level humidity." *Journal of Applied Meteorology*, **7 (4)**, 613-619.

Friedman, J., 2002: “Stochastic gradient boosting.” *Computational Statistics and Data Analysis*, **38 (4)**, 367–378, https://www.sciencedirect.com/science/article/abs/pii/S016794730100065 2.

McGovern, A., D. Gagne II, J. Basara, T. Hamill, and D. Margolin, 2015: "Solar energy prediction: An international contest to initiate interdisciplinary research on compelling meteorological problems." *Bulletin of the American Meteorological Society*, **96 (8)**, 1388-1395, https://journals.ametsoc.org/bams/article/96/8/1388/69447.

Quinlan, J., 1986: "Induction of decision trees." *Machine Learning*, **1 (1)**, 81–106, https://link.springer.com/article/10.1007/BF00116251.