Random Forests

Last time: $V_1(x)$ = Win Probability if you have a 1st down and 10 at game-state x = (yardline, game seconds remaining, pointspread, timeouts, receive 2nd half kickoff, ___)

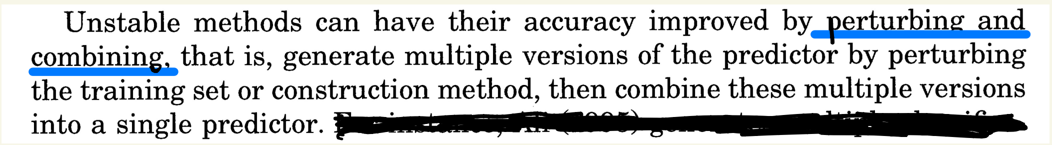
We introduced Decision Trees as a machine learning model to capture interacting relationships between variables.

Problem: decision trees are unstable and prone to overfitting.

Unstable: if you slightly perturb the training dataset, you can get a very different model.

Overfitting: memorizing the random noise in the training dataset rather than the "true" underlying trend (signal)

What to do about this? Ask Leo Breiman.



Unstable methods can have their accuracy improved by perturbing and combining, that is, generate multiple versions of the predictor by perturbing the training set or construction method, then combine these multiple versions into a single predictor. [redacted]

- small changes in the training set or in the construction of a decision tree can lead to larger changes in the aggregated predictor

---

<u>**Bagging**</u> = Bootstrap - Aggregating

Training set T consists of N instance $n=1,\dots,N$ (e.g. N football plays)

put equal probabilities $p(n) = \frac{1}{N}$ on each instance.

Sample with Replacement (Bootstrap) N times from the training set T, forming a "new" Resampled training set $T^{(B)}$.

Then fit a decision tree $\hat{f}^{(B)}$ on $T^{(B)}$.

Lets do this, say, 1000 times, so we have 1000 bootstrapped decision trees $\{\hat{f}^{(1)}, \dots, \hat{f}^{(1000)}\}$.

Aggregate $\hat{f} = \frac{1}{1000} \sum_{B=1}^{1000} \hat{f}^{(B)}$.

Why might bagging be a good idea?

1. First, pretend you have 1000 independently drawn datasets of size N, $\{T^{(1)}, \dots, T^{(1000)}\}$.

Fit 1000 decision trees $\{\hat{f}^{(1)}, \dots, \hat{f}^{(1000)}\}$ where $\hat{f}^{(B)}$ fit on $T^{(B)}$,

$$\hat{f} = \frac{1}{1000} \sum_{B=1}^{1000} \hat{f}^{(B)}$$ and

$$\mathbb{E}\hat{f} = \frac{1}{1000} \sum_{B=1}^{1000} \mathbb{E}\hat{f}^{(B)}$$ by linearity of expectation

$$= \frac{1}{1000} \sum_{B=1}^{1000} \mathbb{E}\hat{f}^{(1)}$$ because $\mathbb{E}\hat{f}^{(B)} = \mathbb{E}\hat{f}^{(1)}$ for all B

$$= \mathbb{E}\hat{f}^{(1)}.$$

The aggregated predictor has the same expected value as 1 individual decision tree.

$$\mathrm{var}(\hat{f}) = \mathrm{var}\left(\frac{1}{1000}\sum_{B=1}^{1000}\hat{f}^{(B)}\right)$$

$$= \frac{1}{1000^2}\sum_{B=1}^{1000}\mathrm{var}(\hat{f}^{(B)})$$ by independence

$$= \frac{1}{1000^2}\sum_{B=1}^{1000}\sigma^2$$

$$= \frac{\sigma^2}{1000}$$

$\mu = \mathbb{E}\hat{f}^{(B)}$, $\sigma^2 = \mathrm{var}(\hat{f}^{(B)})$, then

$\mathbb{E}\hat{f} = \mu$ $\quad$ $\mathrm{var}(\hat{f}) = \frac{\sigma^2}{1000}$.

Aggregating independent decision trees has same expected value but lower variance.

Assumed 1000 independently drawn training sets from some "true" function.

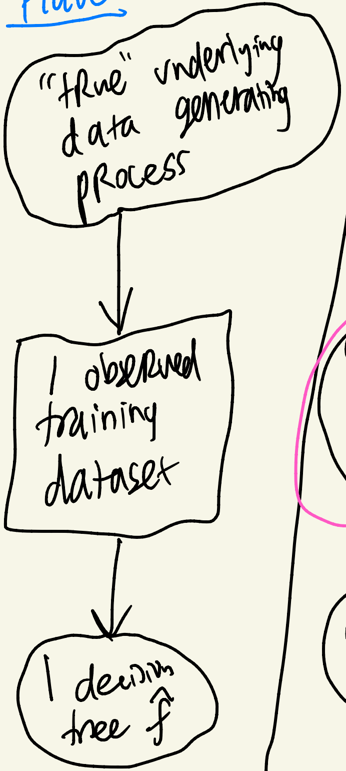
* In reality we have only onee independently drawn dataset of size N, not 1000.

Bootstrapping: created 1000 different training datasets by Resampling N Rows with Replacement.

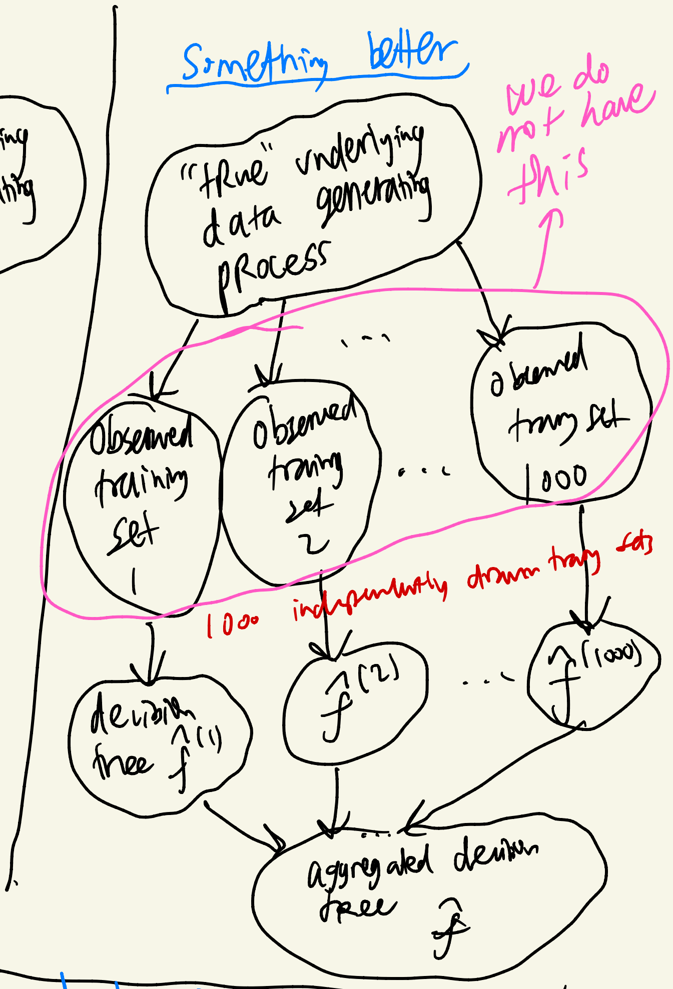
Idea behind Bagging one bootstrapped dataset is a reasonably close approximation to one independently drawn dataset from the "true" data source.

Why is bootstrapping a reasonable approximation to independently drawing a new training set?

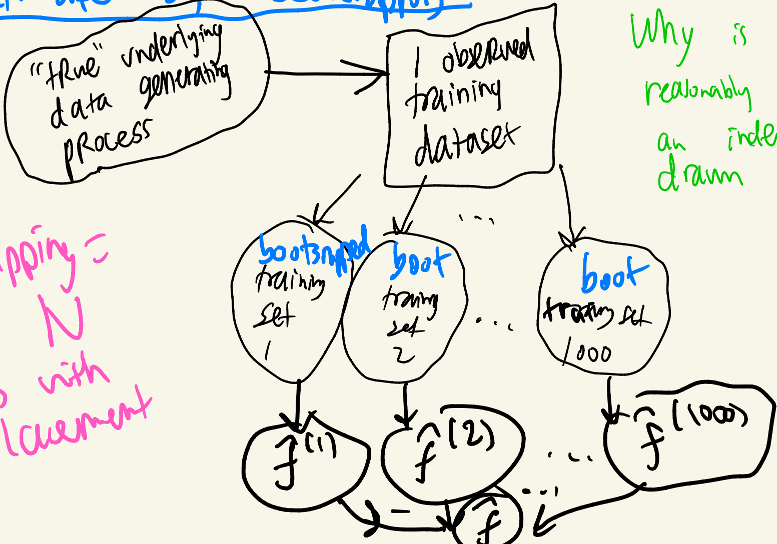
Have


"true" underlying data generating process
1 observed training dataset
1 decision tree $\hat{f}$

Something better

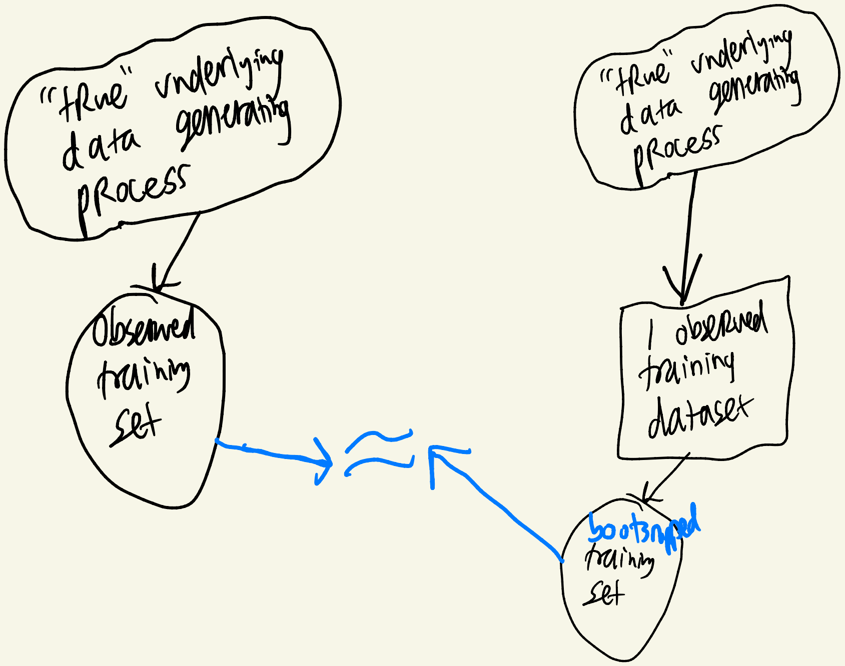

"true" underlying data generating process
Observed training set 1
Observed traning set 2
...
observed traing set 1000
we do not have this
1000 independently drawn traing sets
decision tree $\hat{f}^{(1)}$
$\hat{f}^{(2)}$
$\hat{f}^{(1000)}$
aggregated decision tree $\hat{f}$

Approximate by bootstrapping

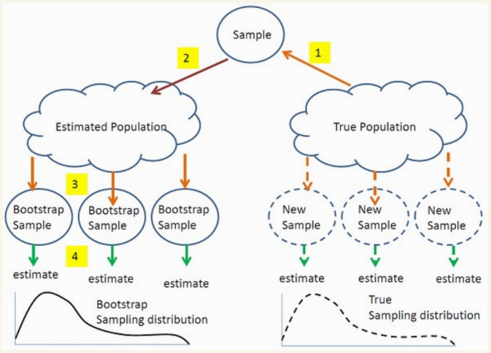


Why is bootstrapped $T^{(B)}$ reasonably similar to an independently drawn training set?

bootstrapping = sample N rows with replacement


"true" underlying data generating process
Observed training set
"true" underlying data generating process
1 observed training dataset
bootstrapped training set
≈

sampling from the "true" underlying data generating process is similar to sampling from the observed training set because the observed training set itself is sampled from the "true" underlying data generating process.



Random Forest

Input: observed training data $T = (X, y)$

Hyperparameters: $n, m, A = 1000$

for $B = 1, \ldots, A$:

1. Bootstrap (Row subsampling):
   sample n of N rows of T with replacement

2. Column subsampling:
   sample m of the columns (features) of X

These 2 things yield a new training set $T^{(B)}$ with n rows and m columns.

3. fit a decision tree $\hat{f}^{(B)}$ on $T^{(B)}$.

Aggregate: $\hat{f} = \frac{1}{1000} \sum_{B=1}^{1000} \hat{f}^{(B)}$.

* Random forest is much better than a single decision tree because it reduces variance.

Bias-Variance Tradeoff

$$\text{out-of-sample prediction error} = \underbrace{\text{Bias}}_{\mathbb{E}(\hat{f}-f)^2} + \underbrace{\text{Variance}} + \underbrace{\text{Irreducible error}}$$

$\Downarrow$ lower for R.F. than dec. tree

Bias: same for 1 dec. tree and 1 Random Forest

Variance: R.F. has lower variance than 1 decision tree

Irreducible error: same no matter what model you use

* one decision tree $\hat{f}(T)$ is more unstable and sensitive to the Random idiosyncracies of the training set than 1000 aggregated decision trees

$$\hat{f}(T) = \frac{1}{1000}\sum^{1000} \hat{f}^{(B)}(T).$$