Data, Environment and Society: Lecture 23: Boosting, Bagging and Random Forests

Instructor: Duncan Callaway
GSI: Seigi Karasaki

November 8, 2018

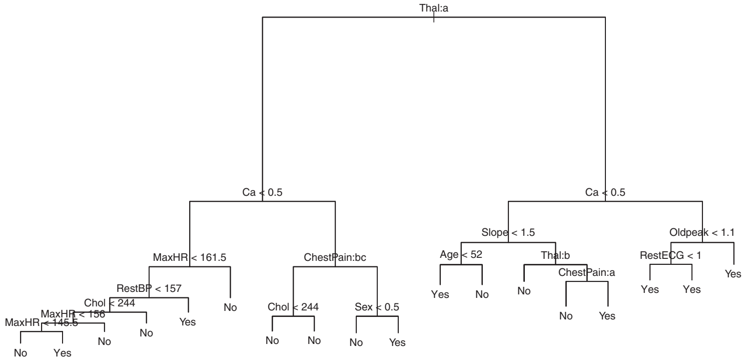
Announcements

- Midterm: Average 81, Median 84
- Projects!
  - Data identified and “resource allocation” question by 11/19 lab.
  - First pass on running a prediction model by 11/26 lab.
  - Lab notebook due Dec 11, 6pm (when the final exam would have ended).
  - Come to office hours, or schedule time with us, to discuss your ideas!
- Next week
  - We’ll begin talking about support vector machines Tuesday
  - Thursday, guest lectures from Grace Wu and Diego Ponce de Leon Barido.
    - Support vector machines played a prominent role in Grace’s PhD work on impact of dams on deforestation!
    - Diego focuses on energy data analytics and sharing in Latin America
- Remainder of semester
  - Finish support vector machines on 11/20
  - Quick intro to neural networks on 11/27
  - Duncan traveling 11/29...

Today's lecture setup

Thus far we've talked about regression trees and classification trees.

- We build multiple trees in the cross validation process
- But then build a single tree with all folds.

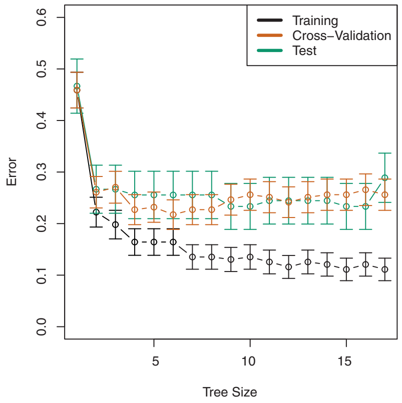


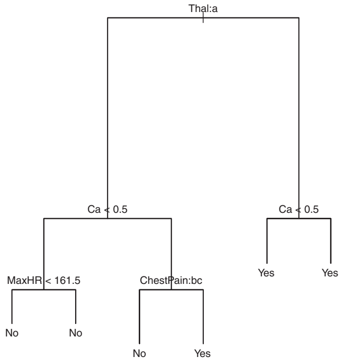
From ISLR: Heart disease classification tree

Today's lecture setup

Thus far we've talked about regression trees and classification trees.

- We build multiple trees in the cross validation process
- But then build a single tree with all folds.





From ISLR: Heart disease classification tree

Today's lecture setup

Thus far we've talked about regression trees and classification trees.

- We build multiple trees in the cross validation process
- But then build a single tree with all folds.

As we've discussed:

- Recursive binary splitting quickly narrows the set of candidate trees
- The "best" tree might be one with a different first split than the "greedy optimal" one.

Trees built using the approaches we've learned so far tend to have *high variance.*

Why?

Today’s lecture setup

Thus far we’ve talked about regression trees and classification trees.

- We build multiple trees in the cross validation process
- But then build a single tree with all folds.

As we’ve discussed:

- Recursive binary splitting quickly narrows the set of candidate trees
- The “best” tree might be one with a different first split than the “greedy optimal” one.

Trees built using the approaches we’ve learned so far tend to have *high variance.*

Why?

- Split the data randomly into two data sets
- Each one could yield very different trees, depending on the nature of the random split.

Today we’ll talk about tools to bring the variance down.

How much does this cow weigh?

How much does this cow weigh?

According to James Surowiecki's book, *The Wisdom of Crowds*, in 1906 Frances Galton averaged all of a crowd's guesses for a heffer and they were only 1% off.

Another anecdote

The Netflix Prize: Use historical viewing data to predict what movies customers will like.

Another anecdote

The Netflix Prize: Use historical viewing data to predict what movies customers will like.

The winners were like the crowd at that 1906 county fair: they were a coalition of three separate competitors from the prior year. They averaged their models, won, and split the winnings three ways.

Overview of how “the power of crowds” works for regression trees.

Step 1: Accept that interpretability is a casualty for what you’re about to do

Overview of how “the power of crowds” works for regression trees.

Step 1: Accept that interpretability is a casualty for what you’re about to do

Step 2: Build many different trees from the data you’re given

Overview of how “the power of crowds” works for regression trees.

Step 1: Accept that interpretability is a casualty for what you’re about to do

Step 2: Build many different trees from the data you’re given

Step 3: *Average* the predictions from many different trees.

Overview of how “the power of crowds” works for regression trees.

Step 1: Accept that interpretability is a casualty for what you’re about to do

Step 2: Build many different trees from the data you’re given

Step 3: *Average* the predictions from many different trees.

Interpretability is lost: now there are many trees with splits in different locations on different predictors.

But if it’s done right: You’ll get better *predictions* because you’ll reduce variance

Different methods differ in step two: how do you manipulate the tree building process so that you get different trees built from the same data?

Three ways to build many trees from the same data

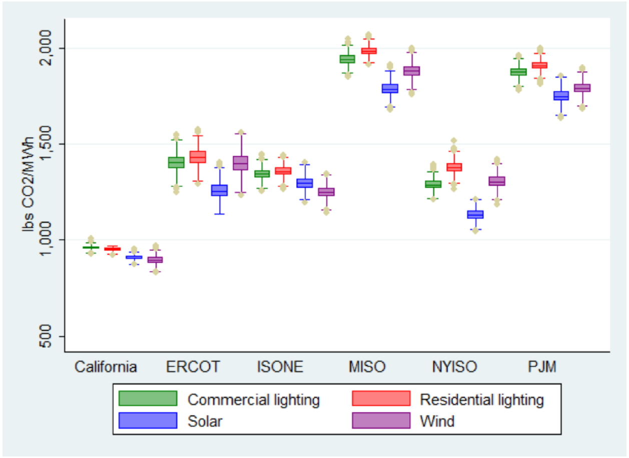
- **Bagging** (Bootstrap aggregation): Build many trees from random samples of the data
- **Random forests**: Build many trees from bootstrapped samples, but each binary split is chosen from a random subset of predictors
- **Boosting**: choose new trees to minimize the residual of an existing aggregation of trees.

Bagging = Bootstrap aggregating

Leo Breiman retired from UC Berkeley Statistics in 1993 and published the first paper on bagging in 1994.

Reminder: Bootstrapping

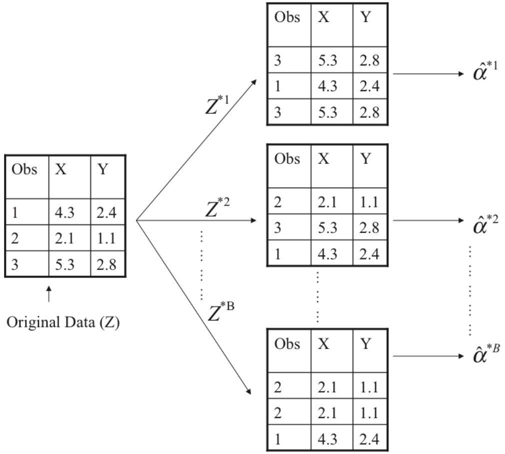
Suppose you want to compute the standard error for a metric, but you can't get there with conventional statistics:



(Callaway, Fowlie, McCormick 2017)

Reminder: Bootstrapping, Part 2

- *First key idea*: create a new sample of N observations from your original (N observation) sample by sampling with replacement.
- Repeat this B times
- Record the metrics you care about each time you build a new model from a new bootstrap
- The average parameter estimate will equal the true parameter
- *Second key idea*: the standard error of the parameter estimates from the population of boostrapped estimates will be roughly the true standard error.



Obs	X	Y
1	4.3	2.4
2	2.1	1.1
3	5.3	2.8

Obs	X	Y
3	5.3	2.8
1	4.3	2.4
3	5.3	2.8

Obs	X	Y
2	2.1	1.1
3	5.3	2.8
1	4.3	2.4

Obs	X	Y
2	2.1	1.1
2	2.1	1.1
1	4.3	2.4

Examples of things you might bootstrap

- A prediction of PM2.5 concentration at a new location
- An estimate for how many CO2 emissions were displaced by a low-carbon technology
- An estimate of the number of people that died due to ozone for a given set of predictors

Moving on to **B**ootstap **AGG**regat**ING** = Bagging

It's very simple:

$$\hat{f}_{\mathsf{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x)$$

where $\hat{f}^{*b}(x)$ is the regression tree estimate from the b^{th} bootstrapped data set.

Each individual tree has high variance but low bias. But averaging deals with the variance.

Moving on to **B**ootstap **AGG**regat**ING** = Bagging

It's very simple:

$$\hat{f}_{\mathsf{bag}}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$$

where $\hat{f}^{*b}(x)$ is the regression tree estimate from the b^{th} bootstrapped data set.

Each individual tree has high variance but low bias. But averaging deals with the variance.

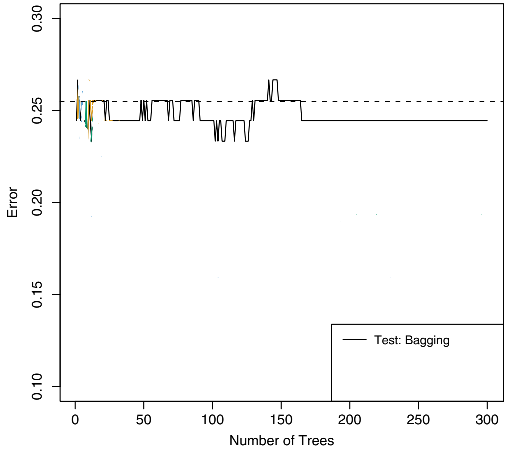
Important: because we deal with variance by averaging, there is no more need for cost-complexity pruning or tuning of the α parameter.

Consequence: Grow the trees deep!

Further details

Data set and model: predict patients' heart disease condition (yes or no) based on many observations.

How many bootstraps? As with conventional boostrapping, keep adding boostraps until the estimate converges.

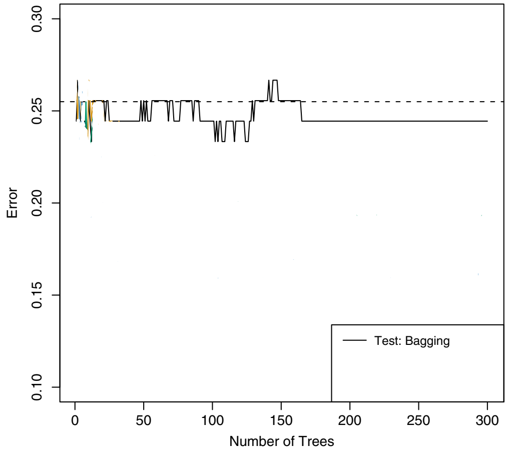


Bagged model (black) vs single regression tree (dashed). Only look at *Test: Bagging* for now.

Further details

Data set and model: predict patients' heart disease condition (yes or no) based on many observations.

How many bootstraps? As with conventional boostrapping, keep adding boostraps until the estimate converges.



Bagged model (black) vs single regression tree (dashed). Only look at *Test: Bagging* for now.

Classification with Bagging: just take the "majority vote" from all the trees you build.

“Out of bag”? The terminology keeps getting better!

When you bootstrap with replacement, a certain number of observations don’t get chosen for inclusion in the sample.

- Folks call those the Out Of Bag observations.
- On average an observation will be left out of about 1/3rd of the bootstrap samples

“Out of bag”? The terminology keeps getting better!

When you bootstrap with replacement, a certain number of observations don’t get chosen for inclusion in the sample.

- Folks call those the Out Of Bag observations.
- On average an observation will be left out of about 1/3rd of the bootstrap samples

“Out of bag”? The terminology keeps getting better!

When you bootstrap with replacement, a certain number of observations don’t get chosen for inclusion in the sample.

- Folks call those the Out Of Bag observations.
- On average an observation will be left out of about 1/3rd of the bootstrap samples

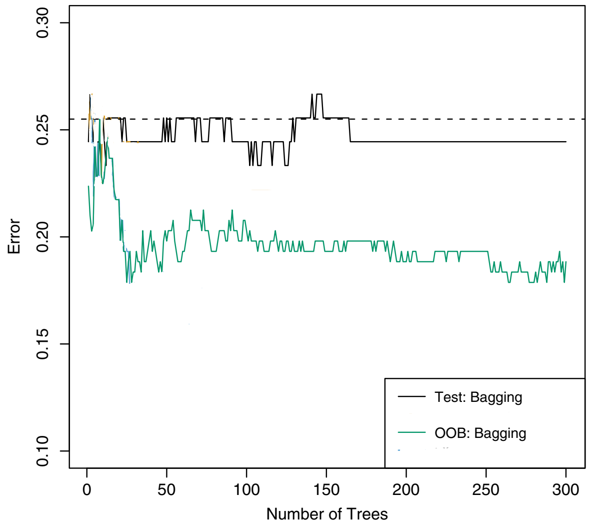
OOB error for an observation: difference between an observation and its prediction using the average of the models built with the observation out-of-bag. This can be MSE or classification error.

OOB error for the model is the mean OOB across all observations.

Why OOB is useful

- Valid estimate of the test error for the bagged model,
  - The response for each observation is predicted using only the trees that were not fit using that observation.
- With B sufficiently large, OOB error is virtually equivalent to leave-one-out cross-validation error
  - This means you can get an estimate of LOOCV error without actually having to do LOOCV.
  - Unlike many other nonlinear estimators, random forests can be fit in one sequence, with cross-validation being performed along the way

Let's look at OOB error

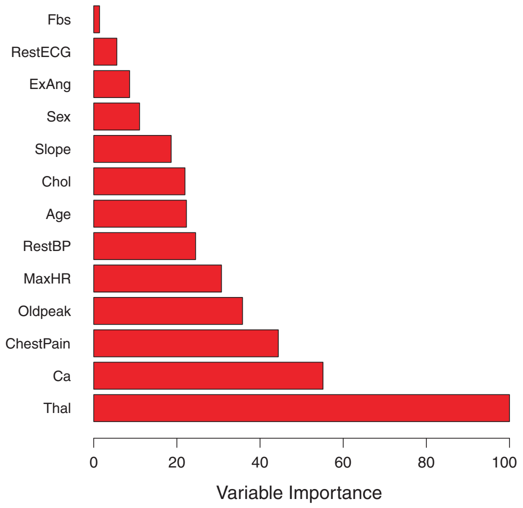


Interpretation is not entirely lost: Variable importance measures.

Basic idea:

1. For each predictor indexed by j, initialize a variable $\rho_j = 0$
2. Then as you grow the trees, every time you split on predictor j add the improvement in RSS for the split to ρ_j.
    - (As you move to a new tree you can keep adding to the same ρ_js, no need to start anew.)
3. When you're done growing trees, normalize all ρ to $\frac{\rho_j}{\max \rho_j} * 100$

Example Variable Importance Measures - Heart data



Thal = Thalium stress test

Random Forests (also due to Leo Breiman)

Random Forests are a simple modification to bagging:

On each split, evaluate only $m < p$ randomly chosen predictors.

Why? What might this accomplish?

Random Forests are a simple modification to bagging:

On each split, evaluate only $m < p$ randomly chosen predictors.

Why? What might this accomplish?

Suppose there is a predictor that is *always* the best first split. If you always split on it first, the space of subtrees you can grow is limited. Splitting on only a subset ensures you don't always split on the winner.

- This "decorrelates" the trees
- The average has less variance as a result.

Random Forests are a simple modification to bagging:

On each split, evaluate only $m < p$ randomly chosen predictors.

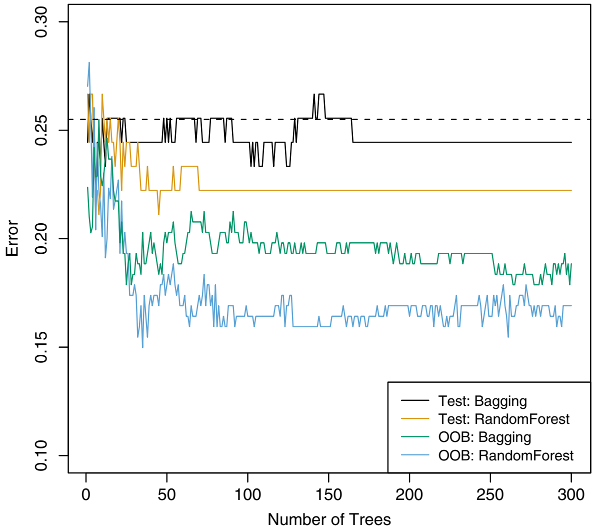
Why? What might this accomplish?

Suppose there is a predictor that is *always* the best first split. If you always split on it first, the space of subtrees you can grow is limited. Splitting on only a subset ensures you don't always split on the winner.

- This "decorrelates" the trees
- The average has less variance as a result.

$m = \sqrt{p}$ works well for classifiers, $m = \frac{p}{3}$ is good for regression.

Let's look at the heart data again



Let's take the quiz

Boosting

The Boosting algorithm

1. Set $\hat{f} = 0$ and $r_i = y_i$ for all *i* in the training set.
2. For $b = 1, 2, ..., B$, repeat:
    1. Fit a tree $\hat{f}_b$ with d splits ($d + 1$ terminal nodes) to the training data (X, r).
    2. Update $\hat{f}$ by adding in a shrunken version of the new tree:
    $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$
    3. Update the residuals
    $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$
3. Output the boosted model
$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

Why does boosting work?

Why does boosting work?

It's chasing the residuals for the model.

- instead of choosing new models that explain the data well...
- ...it chooses models that explain the *residuals* between observations and the "current" model

Comments on boosting

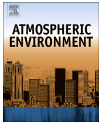
Tuning parameters:

1. B is the number of “boosts”. Because we’re not averaging models fit to the same thing, there is a risk of over-fit. Choose B via cross validation.
2. λ is like the “learning rate”, and we choose pretty small values (0.001 or 0.01). This slows down learning and avoids making residuals worse in some places in order to push residuals down on average.
3. d is the number of splits in each tree. This can be very low (1 or 2)

Contents lists available at ScienceDirect

Atmospheric Environment

journal homepage: www.elsevier.com/locate/atmosenv



Identifying pollution sources and predicting urban air quality using ensemble learning methods

Kunwar P. Singh [a,b,*], Shikha Gupta [a,b], Premanjali Rai [a,b]

[a] *Academy of Scientific and Innovative Research, Council of Scientific & Industrial Research, New Delhi, India*
[b] *Environmental Chemistry Division, CSIR – Indian Institute of Toxicology Research, Post Box 80, Mahatma Gandhi Marg, Lucknow 226 001, India*

HIGHLIGHTS

- Developed tree ensemble models for seasonal discrimination and air quality prediction.
- PCA used to identify air pollution sources; air quality indices used for health risk.
- Bagging and boosting algorithms enhanced predictive ability of ensemble models.
- Ensemble classification and regression models performed better than SVMs.
- Proposed models can be used as tools for air quality prediction and management.

GRAPHICAL ABSTRACT

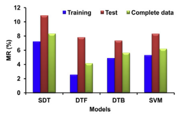
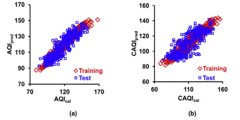
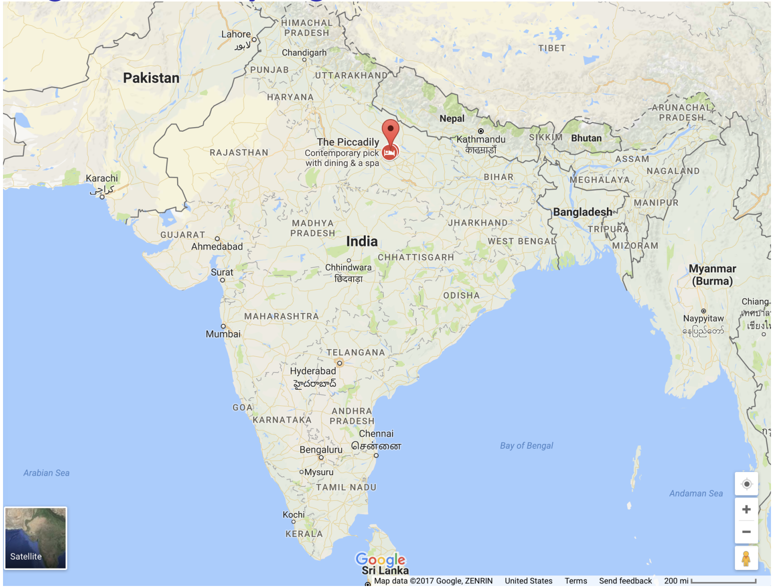


Figure shows misclassification rate in seasonal discrimination of air quality of Lucknow yielded by different models and suggest that the ensemble learning classification models (DTF and DTB) performed relatively better than SDT and SVM.



Figures show correlative distribution of calculated and model predicted values of (a) AQI, and (b) CAQI for Lucknow ambient air using DTB model.

Singh *et al* study region – Lucknow



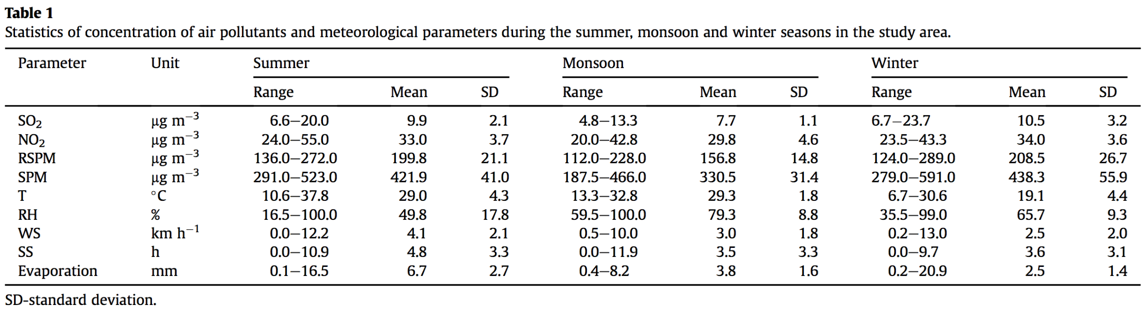


Table 1
Statistics of concentration of air pollutants and meteorological parameters during the summer, monsoon and winter seasons in the study area.

Parameter	Unit	Summer			Monsoon			Winter		
		Range	Mean	SD	Range	Mean	SD	Range	Mean	SD
SO_2	μg m^{-3}	6.6–20.0	9.9	2.1	4.8–13.3	7.7	1.1	6.7–23.7	10.5	3.2
NO_2	μg m^{-3}	24.0–55.0	33.0	3.7	20.0–42.8	29.8	4.6	23.5–43.3	34.0	3.6
RSPM	μg m^{-3}	136.0–272.0	199.8	21.1	112.0–228.0	156.8	14.8	124.0–289.0	208.5	26.7
SPM	μg m^{-3}	291.0–523.0	421.9	41.0	187.5–466.0	330.5	31.4	279.0–591.0	438.3	55.9
T	°C	10.6–37.8	29.0	4.3	13.3–32.8	29.3	1.8	6.7–30.6	19.1	4.4
RH	%	16.5–100.0	49.8	17.8	59.5–100.0	79.3	8.8	35.5–99.0	65.7	9.3
WS	km h^{-1}	0.0–12.2	4.1	2.1	0.5–10.0	3.0	1.8	0.2–13.0	2.5	2.0
SS	h	0.0–10.9	4.8	3.3	0.0–11.9	3.5	3.3	0.0–9.7	3.6	3.1
Evaporation	mm	0.1–16.5	6.7	2.7	0.4–8.2	3.8	1.6	0.2–20.9	2.5	1.4

SD-standard deviation.

RSPM = respirable suspended PM10; SPM = suspended PM. RSPM permissible levels in India = 100 $\mu g/m^{-3}$; In U.S., 150 $\mu g/m^{-3}$ is unhealthy for sens. groups.

Singh *et al* objective and result

Use ensemble-learning tree-based methods to predict combined AQI using meteorological variables (temp, RH, wind speed, evaporation rate, sun hours)

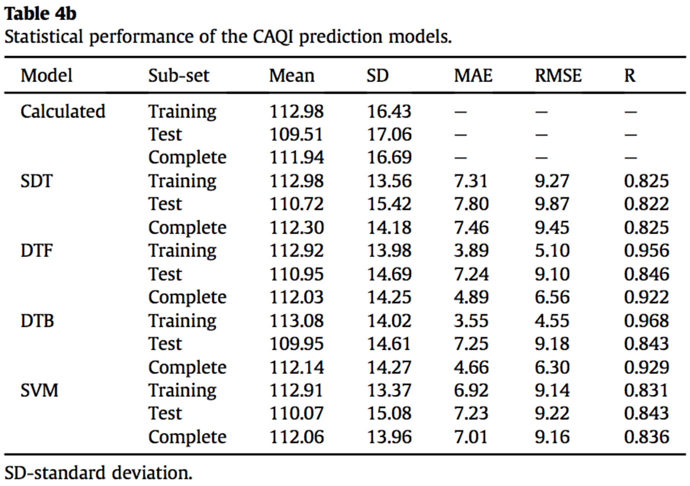


Table 4b
Statistical performance of the CAQI prediction models.

Model	Sub-set	Mean	SD	MAE	RMSE	R
Calculated	Training	112.98	16.43	–	–	–
	Test	109.51	17.06	–	–	–
	Complete	111.94	16.69	–	–	–
SDT	Training	112.98	13.56	7.31	9.27	0.825
	Test	110.72	15.42	7.80	9.87	0.822
	Complete	112.30	14.18	7.46	9.45	0.825
DTF	Training	112.92	13.98	3.89	5.10	0.956
	Test	110.95	14.69	7.24	9.10	0.846
	Complete	112.03	14.25	4.89	6.56	0.922
DTB	Training	113.08	14.02	3.55	4.55	0.968
	Test	109.95	14.61	7.25	9.18	0.843
	Complete	112.14	14.27	4.66	6.30	0.929
SVM	Training	112.91	13.37	6.92	9.14	0.831
	Test	110.07	15.08	7.23	9.22	0.843
	Complete	112.06	13.96	7.01	9.16	0.836

SD-standard deviation.

SDT = single decision tree

DTF = decision tree forest

DTB = decision tree boosted

SVM = support vector machine