Prediction models

Seminar Data Science for Economics

Madina Kurmangaliyeva
m.kurmangaliyeva@uvt.nl

Spring 2021

Tilburg University

Decision trees are intuitive.

But they are unstable.

Decision trees are intuitive.

But they can be very unstable.

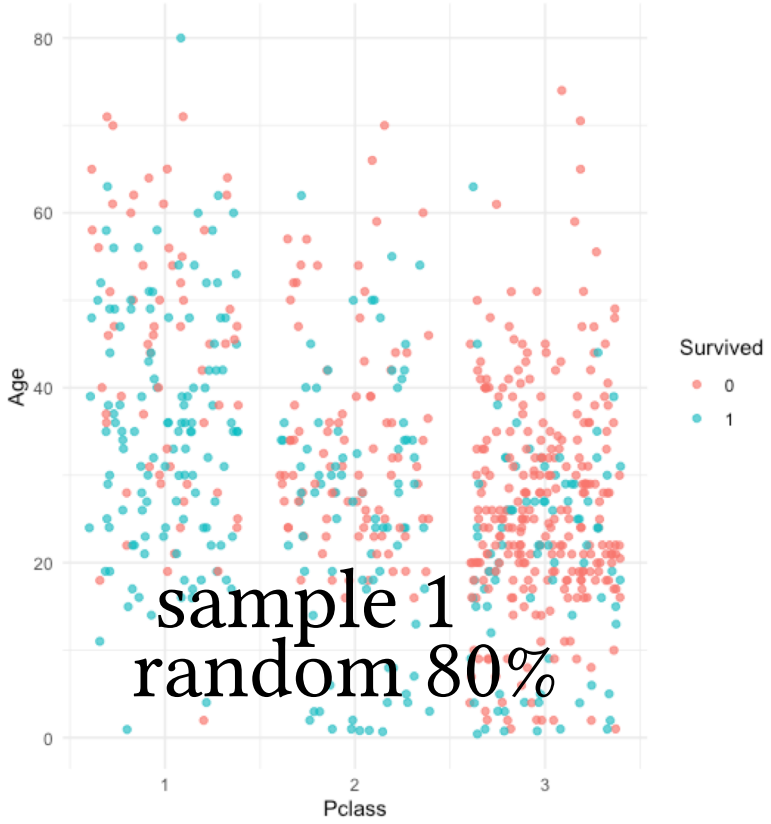
Let's see the predictions on who survives among the Titanic's passengers

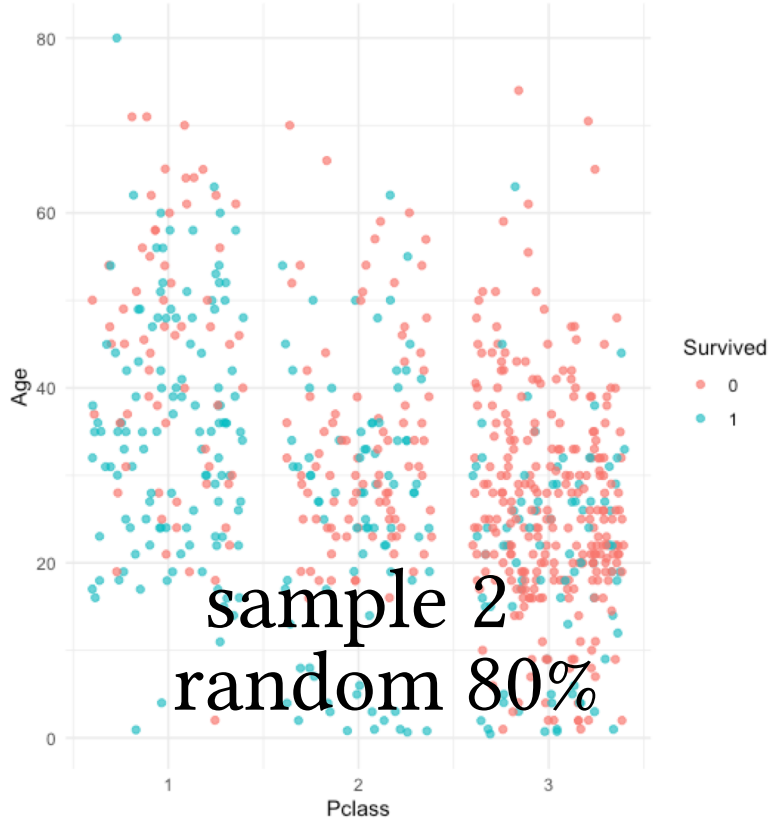
The sinking of the Titanic

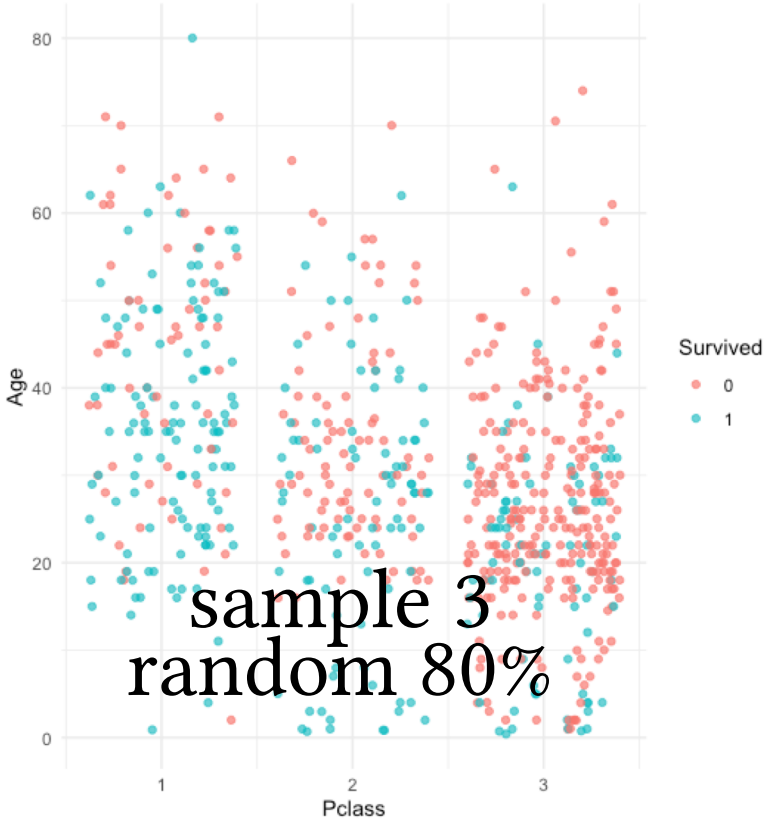
The sinking of the Titanic

1st class 3rd class

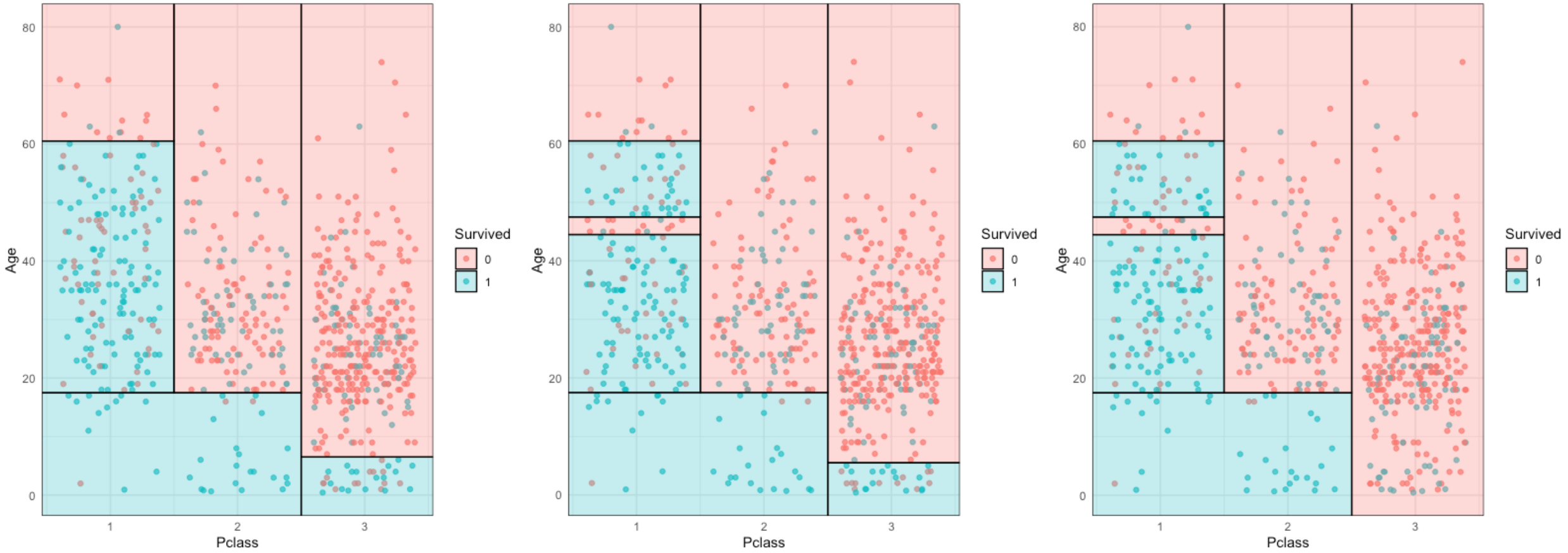
Data on the Titanic passengers
by age and passenger class



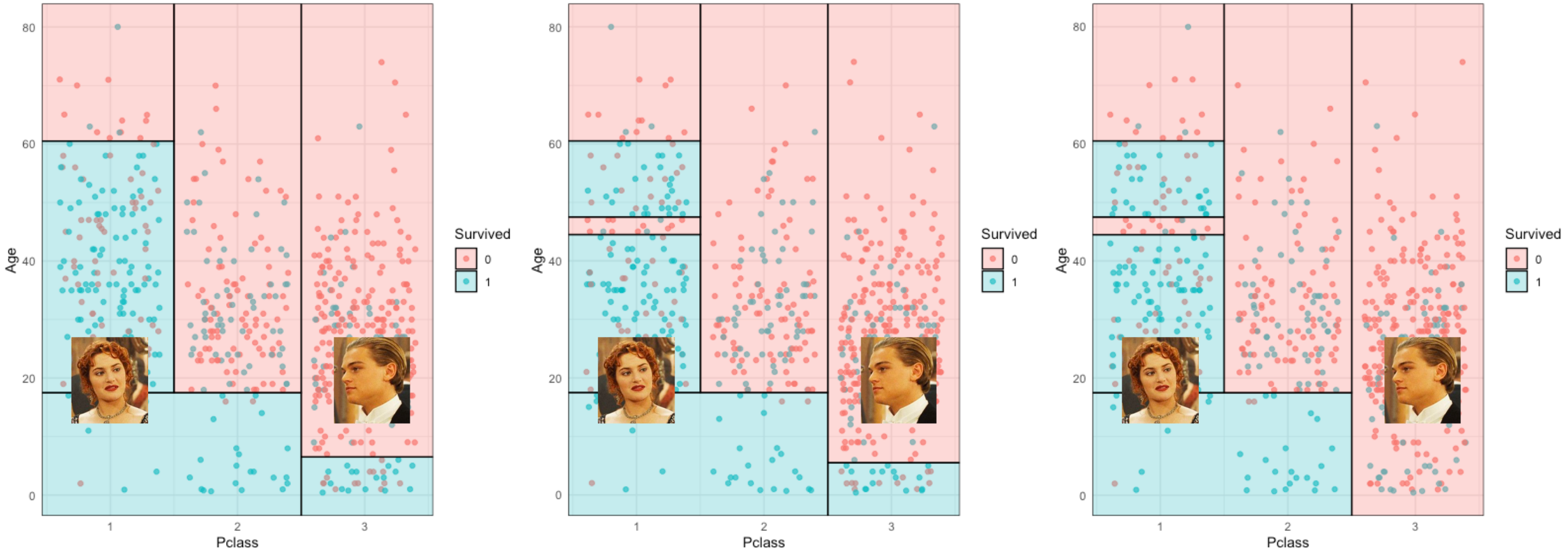




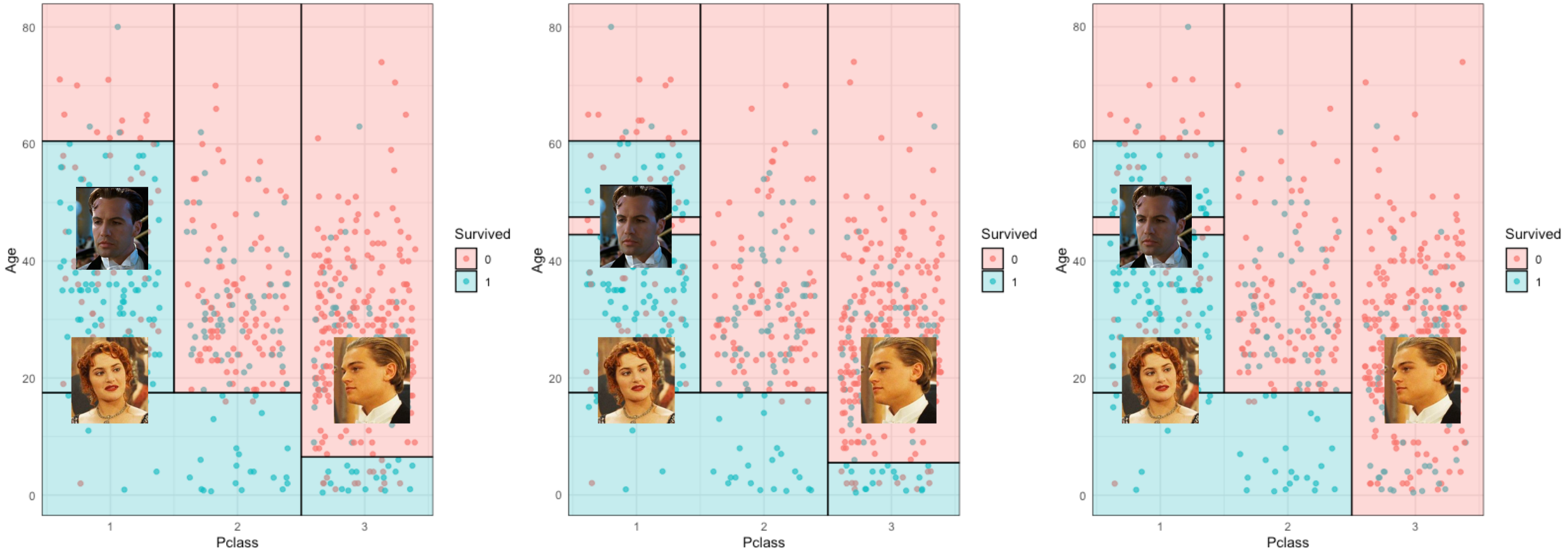
Data on the Titanic passengers
by age and passenger class



Data on the Titanic passengers



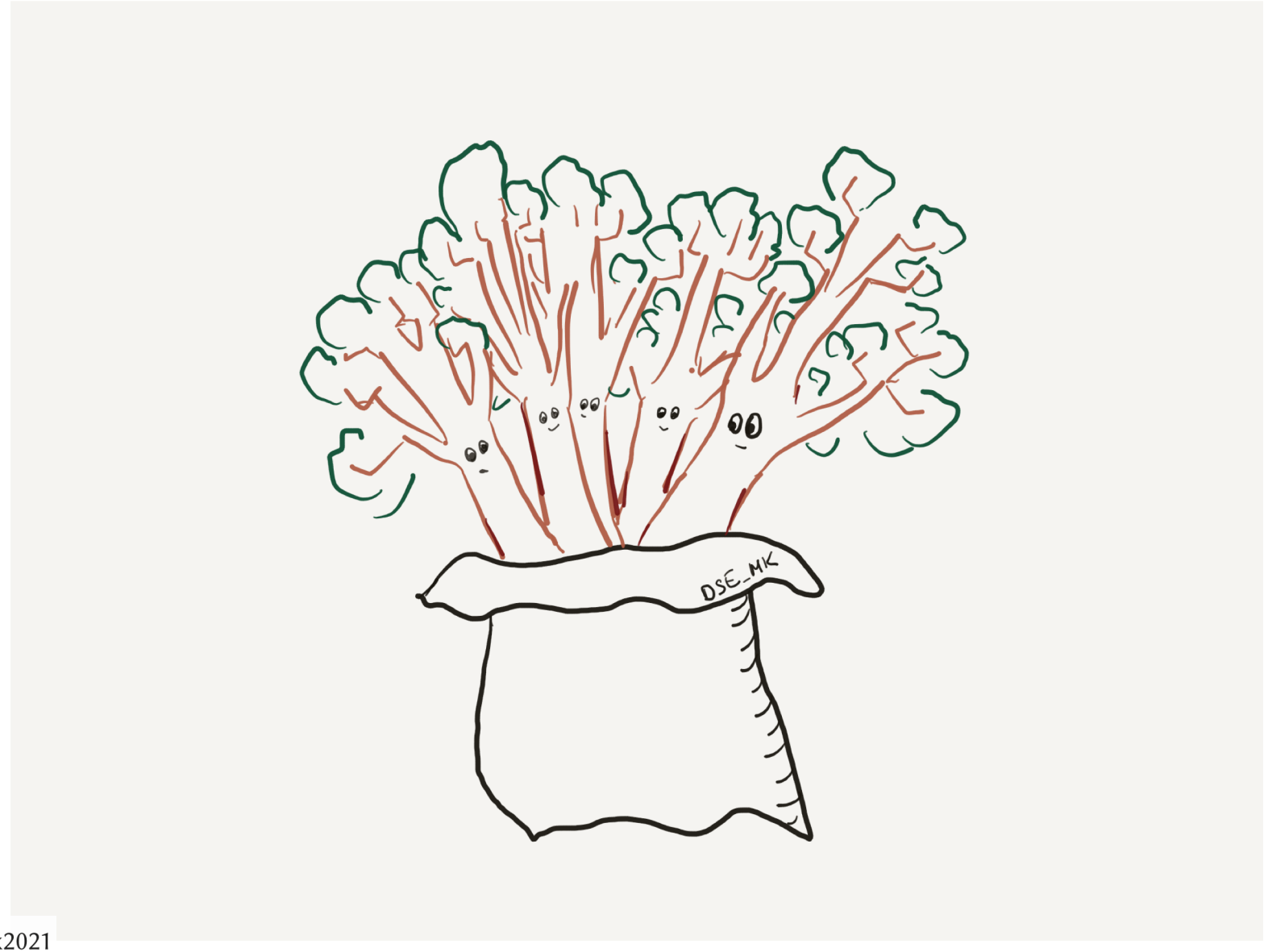
Data on the Titanic passengers

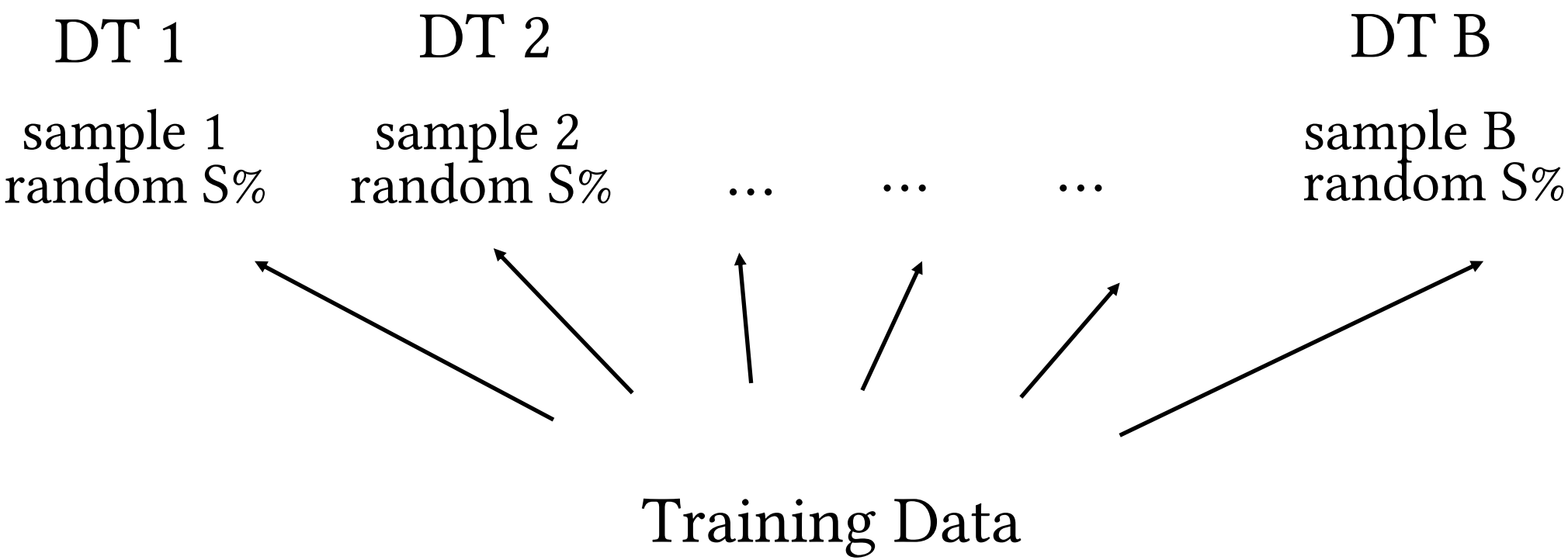


Data on the Titanic passengers

Decision Tree: Low bias but High variance $\Longrightarrow$ Low prediction accuracy

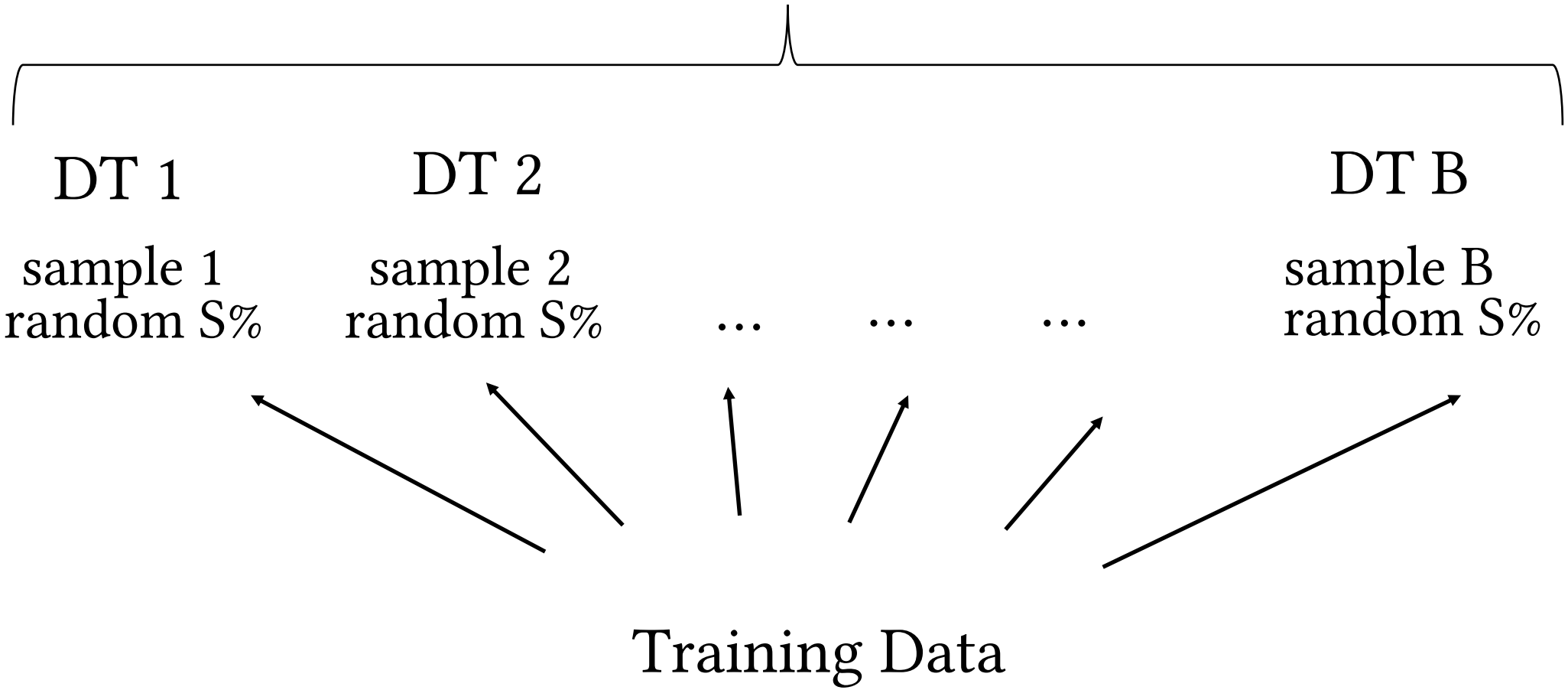
Bagging

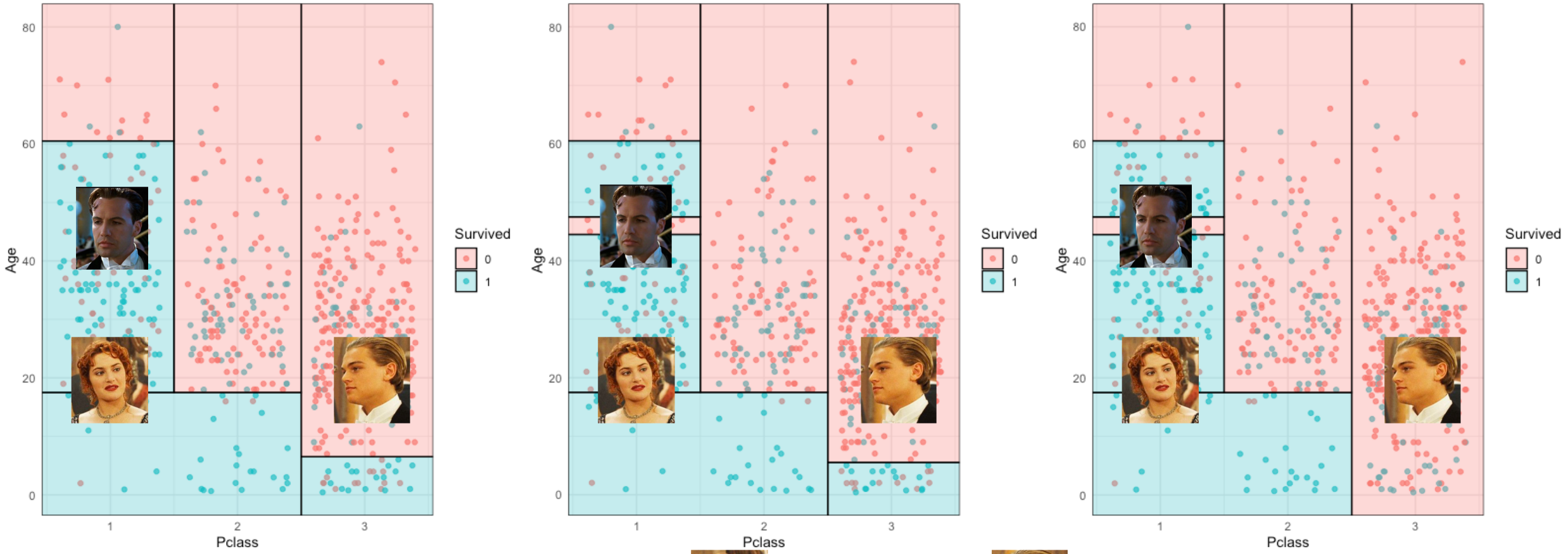



DT 1
DT 2
DT B
sample 1
random S%
sample 2
random S%
…
…
…
sample B
random S%
Training Data

Forecast for any new X

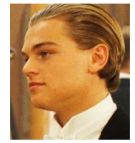
- the **average** of all B predictions for regression trees:
  $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B}\hat{y}^i(X)$
- the **majority vote** by all B predictions for classification trees:
  $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$





Majority vote:

Survived



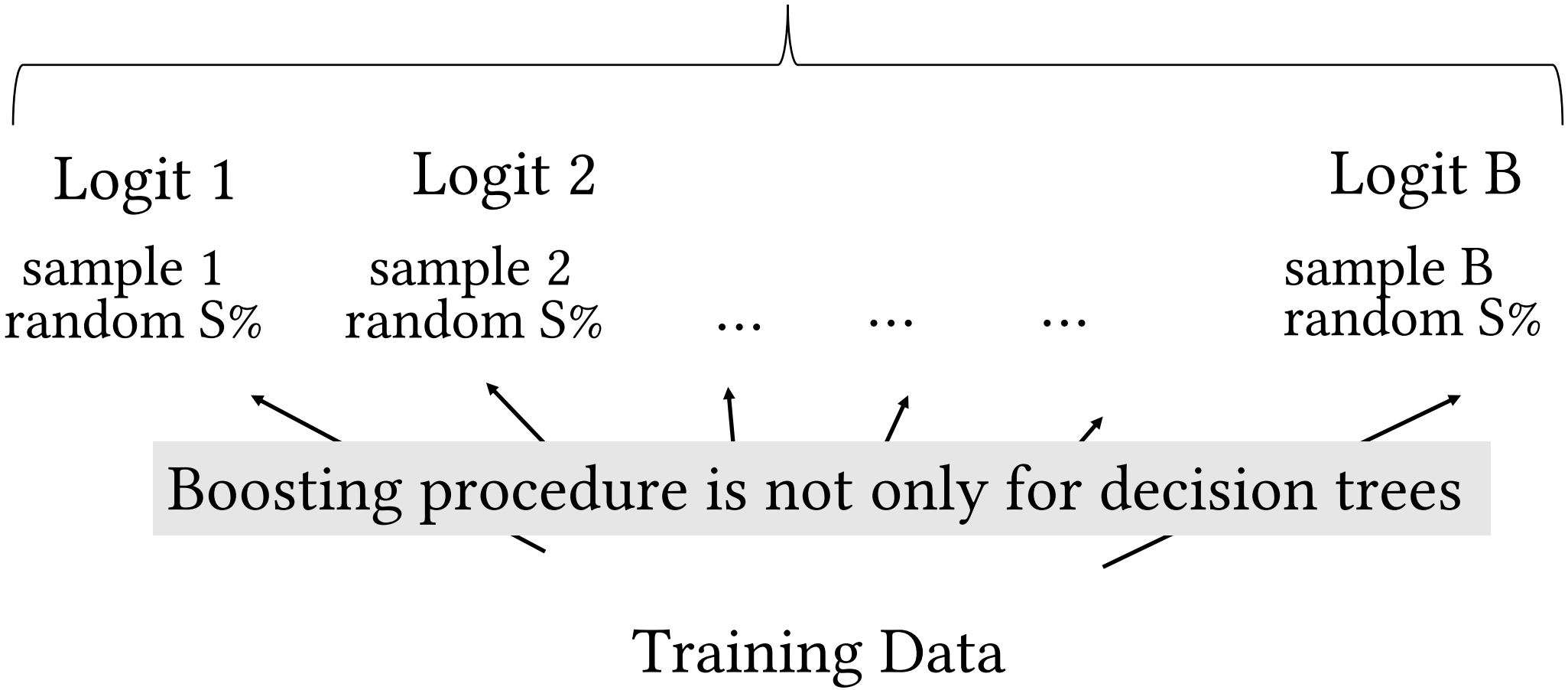
Died

Died

Data on the Titanic passengers

Forecast for any new X

- the **average** of all B predictions for regression trees:
  $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B}\hat{y}^i(X)$
- the **majority vote** by all B predictions for classification trees:
  $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$

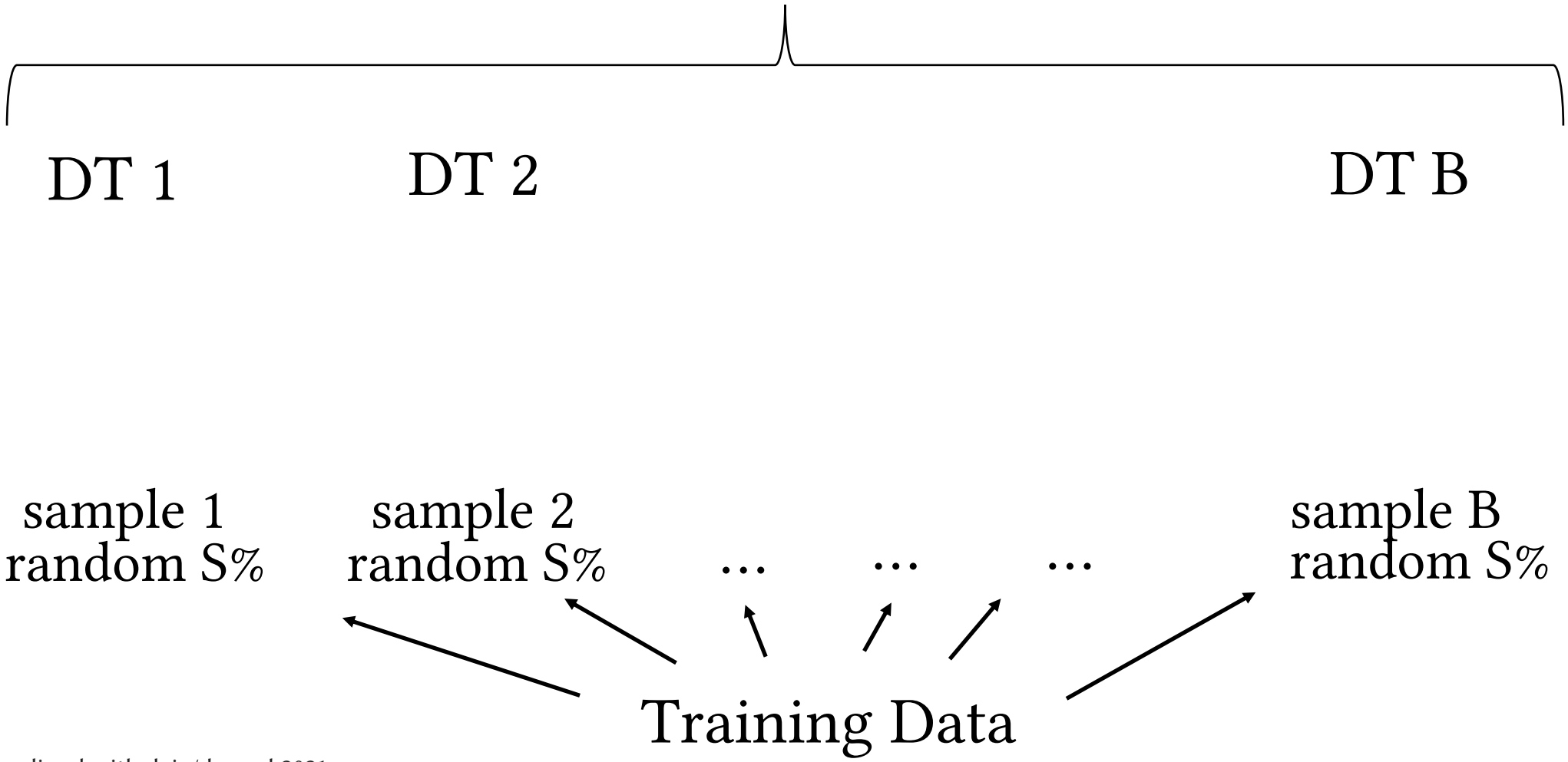


By averaging over many low bias but high variance models, we reduce the overall variance $\Longrightarrow$ more accurate model.

Random Forest

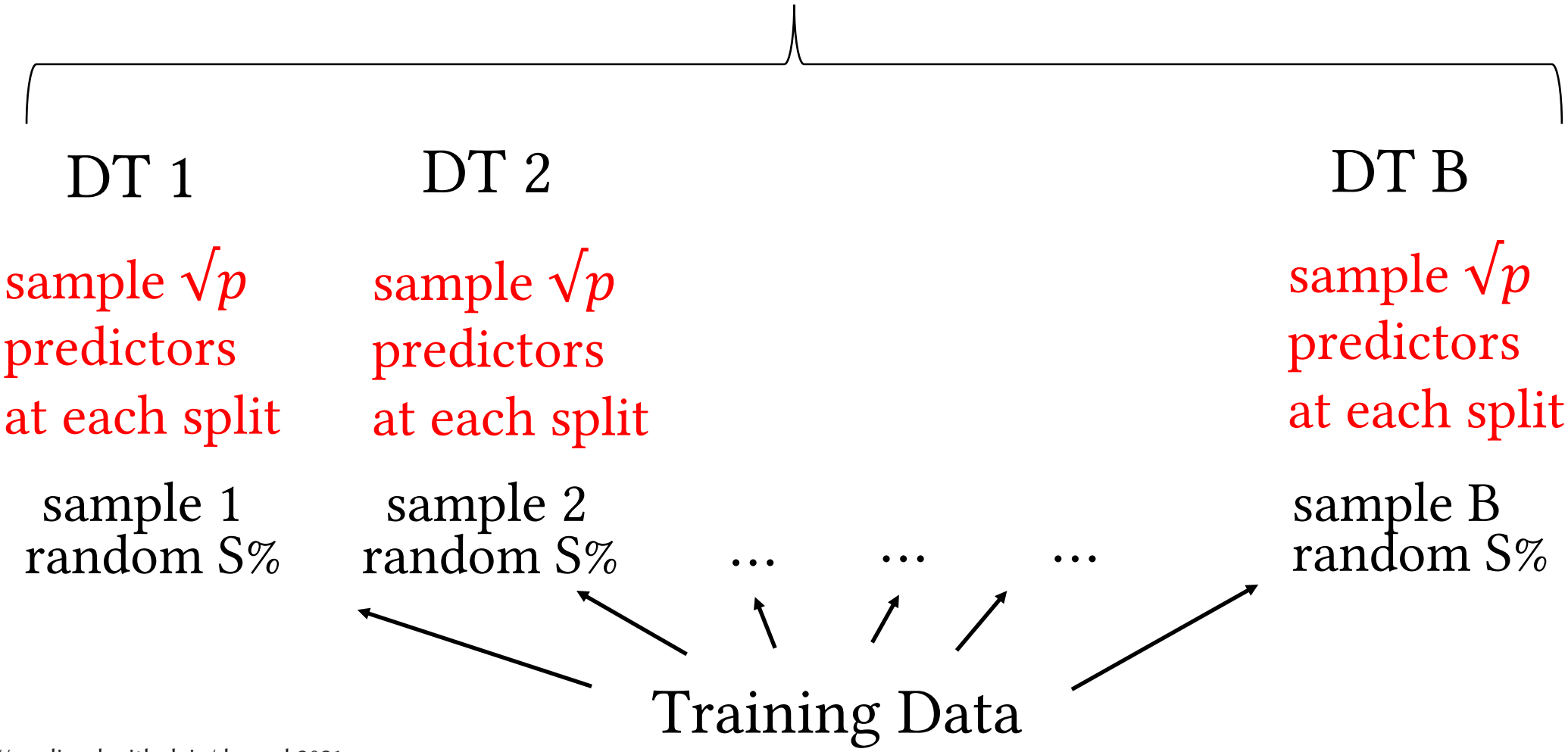
Forecast for any new X

- the **average** of all B predictions for regression trees:
  $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B}\hat{y}^i(X)$
- the **majority vote** by all B predictions for classification trees:
  $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$



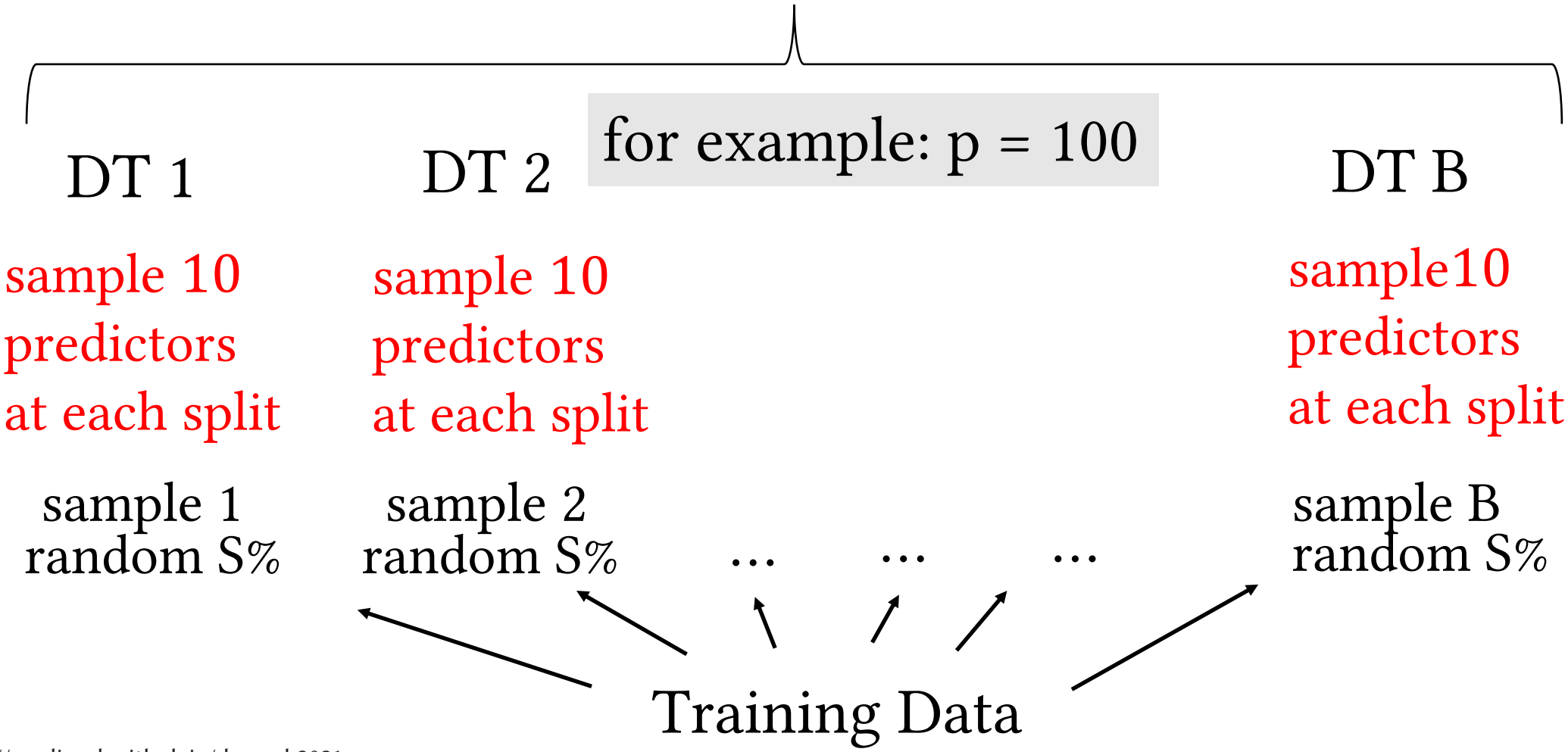
Forecast for any new X

- the **average** of all B predictions for regression trees:
  $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B}\hat{y}^i(X)$
- the **majority vote** by all B predictions for classification trees:
  $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$



Forecast for any new X

- the **average** of all B predictions for regression trees:
  $\hat{y}_{bag}(X) = \frac{1}{B}\sum_{i=1}^{B} \hat{y}^i(X)$
- the **majority vote** by all B predictions for classification trees:
  $\hat{y}_{bag}(X) = \max_k \sum_{i=1}^{B} I\{\hat{y}^i(X) = k\}$

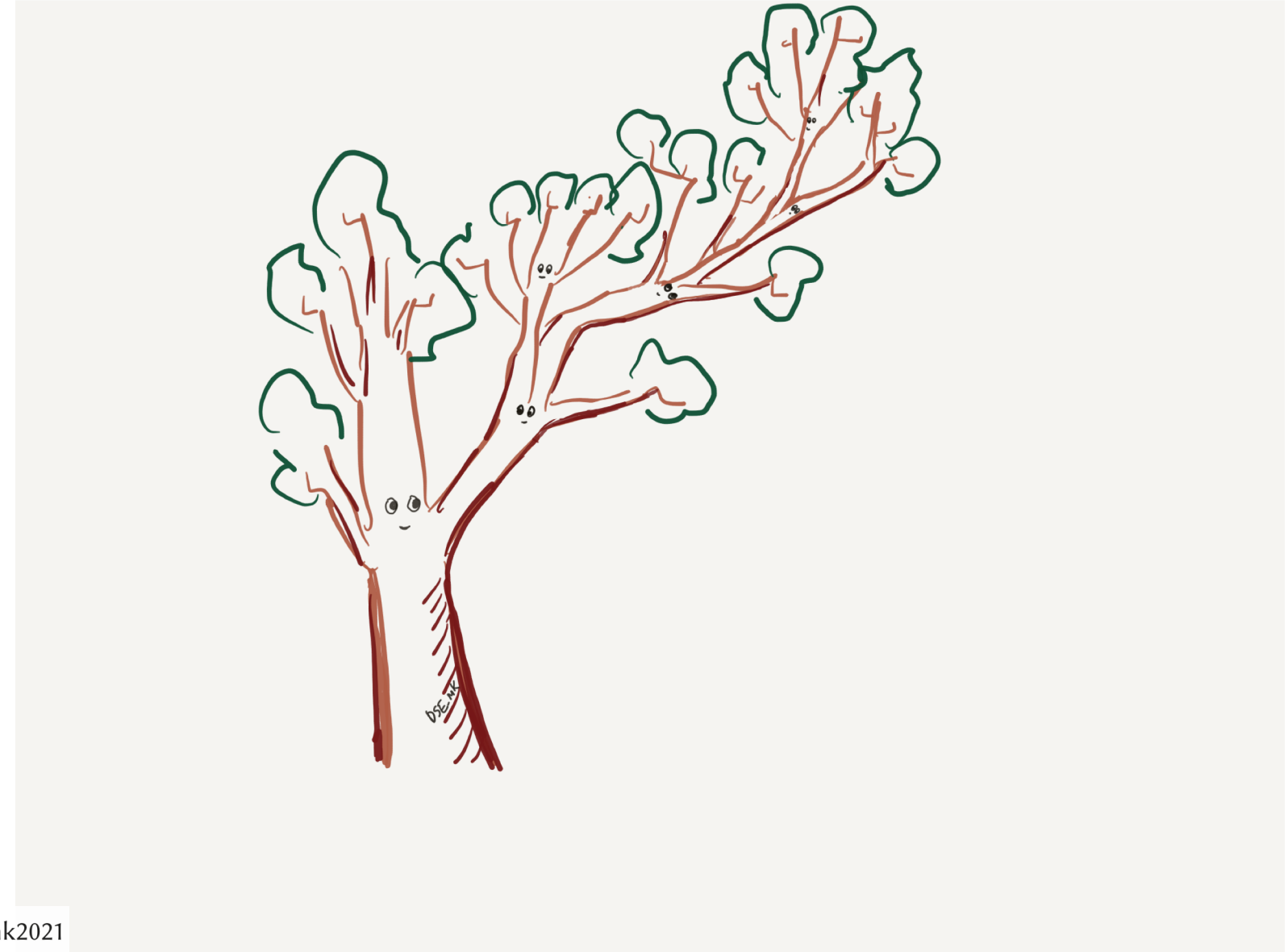


Random Forest

Random forest is especially helpful when there is one strong predictor or a set of predictors that are highly correlated with each other.

By randomly removing some predictors at each split, we reduce the correlation between different trees across bootstrapped samples $\Longrightarrow$ reduce variance $\Longrightarrow$ more accurate model.

Boosting



Boosting

Boosting will grow decision trees sequentially (iteratively):

- Each new decision tree will **"attack" the residuals** (unexplained errors) of the previous tree
- Each tree is usually **shallow**
- But there are **many of them** and each has a chance to improve over the previous models in its own turn
- Hence, boosting is a **slow learning** procedure, which accumulates the "wisdom" of many trees

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
...	...	...
obs n	y_n	X_n

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
...	...	...
obs n	y_n	X_n

	Prediction 0
obs 1	0
obs 2	0
...	...
obs n	0

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
...	...	...
obs n	y_n	X_n

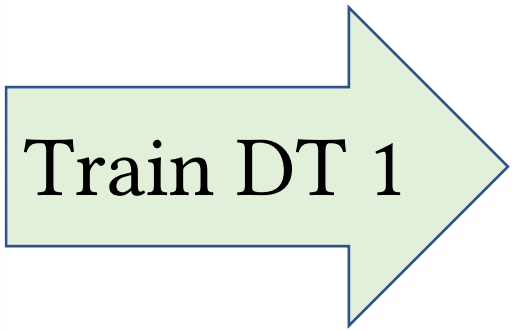
	Prediction 0
obs 1	0
obs 2	0
...	...
obs n	0

	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
...	...	...
obs n	y_n	X_n

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

	Prediction 0	
obs 1		0
obs 2		0
…	…	
obs n		0

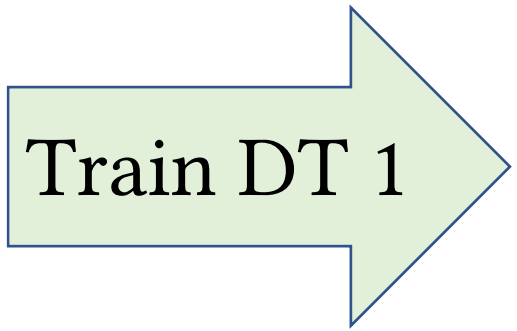
	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n



	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

	Prediction 0
obs 1	0
obs 2	0
…	…
obs n	0

	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n



	Prediction 1
obs 1	$0 + \lambda \hat{y}1(X\_1)$
obs 2	$0 + \lambda \hat{y}1(X\_2)$
…	…
obs n	$0 + \lambda \hat{y}1(X\_n)$

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

	Prediction 0
obs 1	0
obs 2	0
…	…
obs n	0

	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

Train DT 1

	Prediction 1
obs 1	0 + λŷ1(X_1)
obs 2	0 + λŷ1(X_2)
…	…
obs n	0 + λŷ1(X_n)

	resid	Predictors
obs 1	y_1 + λy^1(X_1)	X_1
obs 2	y_2 + λy^1(X_2)	X_2
…	…	…
obs n	y_3 + λy^1(X_2)	X_n

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

	Prediction 0
obs 1	0
obs 2	0
…	…
obs n	0

	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

Train DT 1

	Prediction 1
obs 1	0 + λŷ1(X_1)
obs 2	0 + λŷ1(X_2)
…	…
obs n	0 + λŷ1(X_n)

	resid	Predictors
obs 1	y_1 + λy^1(X_1)	X_1
obs 2	y_2 + λy^1(X_2)	X_2
…	…	…
obs n	y_3 + λy^1(X_2)	X_n

Train DT 2

	Target	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

	Prediction 0
obs 1	0
obs 2	0
…	…
obs n	0

	resid	Predictors
obs 1	y_1	X_1
obs 2	y_2	X_2
…	…	…
obs n	y_n	X_n

Train DT 1

	Prediction 1
obs 1	0 + λŷ1(X_1)
obs 2	0 + λŷ1(X_2)
…	…
obs n	0 + λŷ1(X_n)

	resid	Predictors
obs 1	y_1 + λy^1(X_1)	X_1
obs 2	y_2 + λy^1(X_2)	X_2
…	…	…
obs n	y_3 + λy^1(X_2)	X_n

Train DT 2

	Prediction 1
obs 1	old pred + λŷ2(X_1)
obs 2	old pred + λŷ2(X_2)
…	…
obs n	old pred + λŷ2(X_n)

Boosting for regression trees (Algorithm 8.2 from ISLR)

1. Set predictions to zero, $\hat{y}(X) = 0$, and residuals to y, $r_i = y_i$ for all i in training set.
2. For $b = 1, 2, \ldots, B$ repeat:
   a) Fit a tree $\hat{y}^b$ with d splits (i.e., $d + 1$ terminal nodes) to the training data (X, r) [Note: you do not pass y, you pass residuals]
   b) Update predictions over all domain of x by adding a shrunken prediction of the new tree:

$$\hat{y}^{new} = \hat{y}^{old} + \lambda \hat{y}^b(x) \tag{1}$$

   c) **Update the residuals** by deducting a shrunken prediction of the new tree:

$$r_i^{new} = r_i^{old} - \lambda \hat{y}^b(x_i) \tag{2}$$

3. Output the **boosted model**

$$\hat{y}(x) = \sum_{b=1}^{B} \lambda \hat{y}^b(x) \tag{3}$$

Free parameters

In the algorithm for boosting there are three free parameters:

1. The **number of trees** B. Choosing too high B may lead to overfitting (unlike in bagging or RF) $\Longrightarrow$ need to cross-validate this parameter
2. The **shrinkage parameter** λ, usually between 0.001 and 0.01. It controls the speed of learning. Smaller λ requires higher B.
3. The **number of splits** d, controls interaction depth. Usually, $d = 1$ works very well.

Think about all those trees as little ants: each is small and powerless, but their strength is in numbers and joint attack on one goal!