Introduction to Ensemble methods

- We have looked at different machine learning models now
  - k - Nearest Neighbours
  - Classification & Regression Trees (CART)
  - Logistic Regression
  - Neural Networks
  - Support Vector Machines

- Question: Given a dataset, which ML algorithm should we pick, and how do you know which technique will perform the best?

- Unfortunately, there is no good answer to this question.
  - It is mostly a process of trial-and-error
  - Each kind of ML algorithm yields a different model/hypothesis
  - But there is no perfect model/hypothesis in practice

- So you may ask could we combine several imperfect models into a better model?

- Analogies of combining multiple models in our society
  - Elections combine voter's choices to pick a "good" candidate
  - Committees combine several experts' opinion to make better decisions

- Intuition behind combining multiple models / hypotheses
  - Individuals (or individual models) often make mistakes, but the "majority" is less likely to make mistakes
  - Individuals often have partial knowledge, but a committee can pool expertise to make better decisions

- Ensemble learning can combine an ensemble of
  - Different types of base models (e.g. Neural networks, CART and SVM)
  - Same base model trained slightly differently ✓ We are going to follow this approach

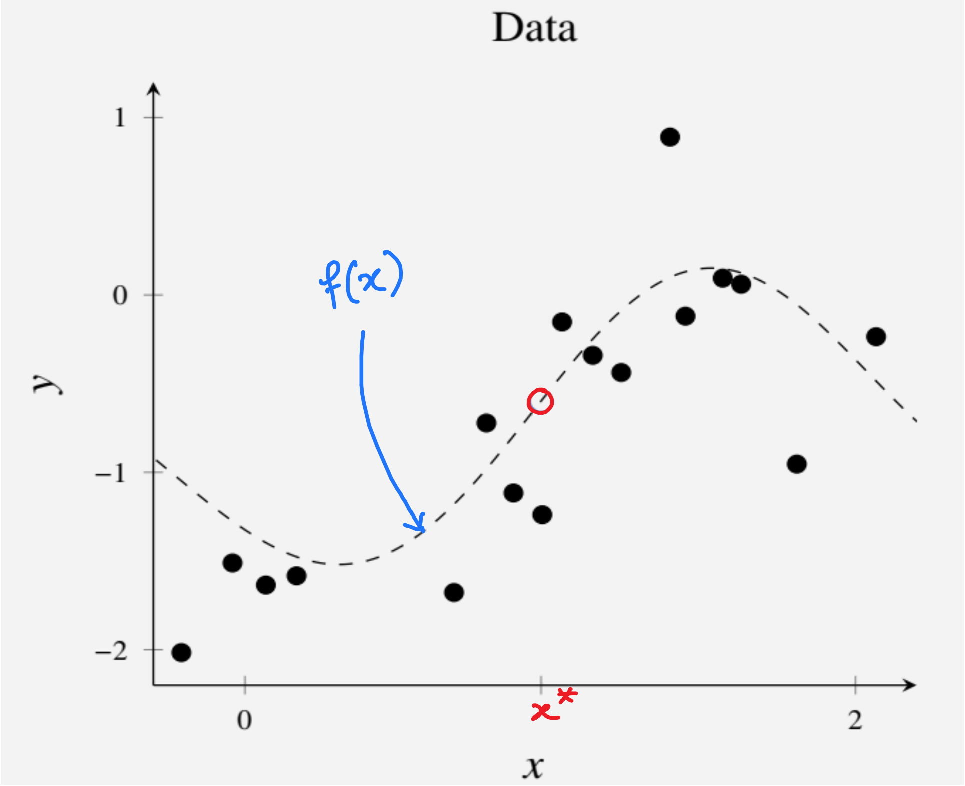
Bagging (or Bootstrap Aggregating)

- A central concept in ML is the bias-variance tradeoff
  - The more flexible a model is, the lower its bias will be
- Examples of highly flexible models that can represent complicated input-output relationships are K-Nearest Neighbours, CART, NNs, etc.
- The downside of such highly flexible models is the risk of overfitting
- Overfitted models lead to unwanted high variance in predictions
- By using bagging, we can reduce the variance of the base model without increasing its bias
- Lets take an example of regression trees with bagging

- Consider the data obtained as

$$y = f(x) + \underbrace{\epsilon}_{\text{noise}}$$

- We would like to train an ML model using this data, so as to be able to predict new data points well

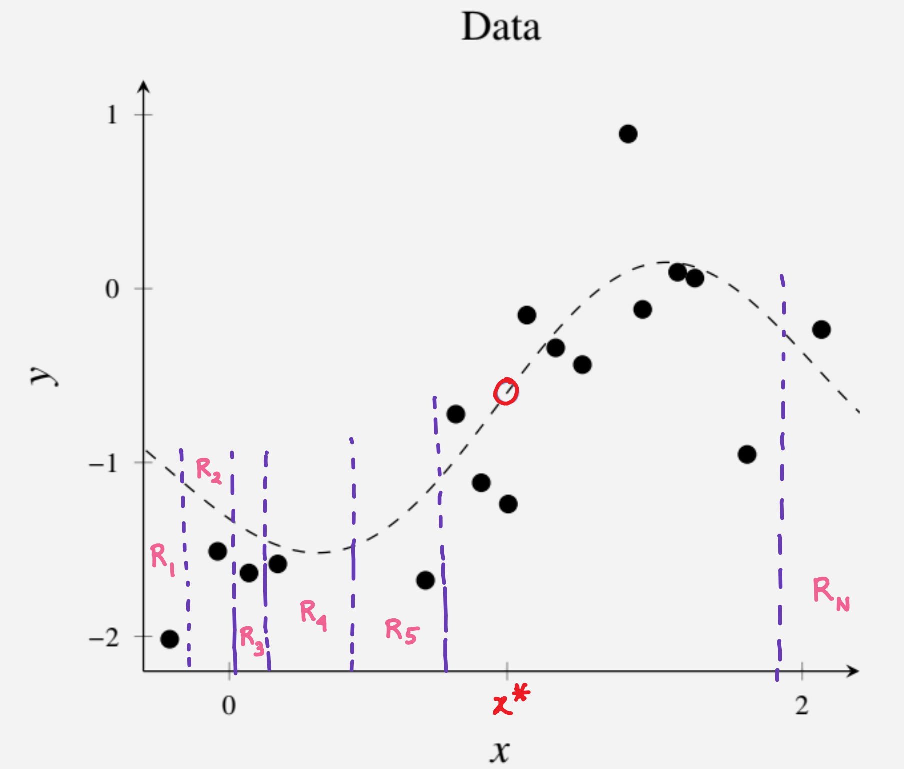


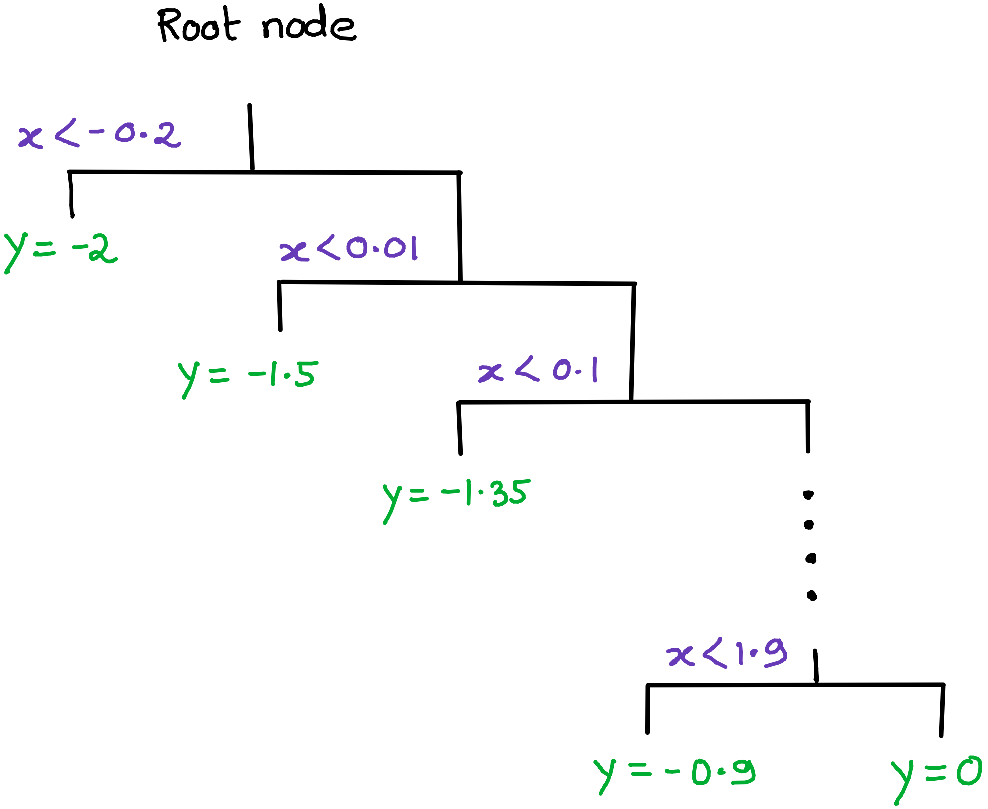
- A good prediction would mean that the trained model should predict $f(x)$ shown by the dotted line well at x^*

- For this problem, let us use Regression Trees as the chosen ML method, since they are non-parametric methods and are very flexible

- Recall that in Classification and Regression trees, we partition the input space using box-shaped decision boundaries

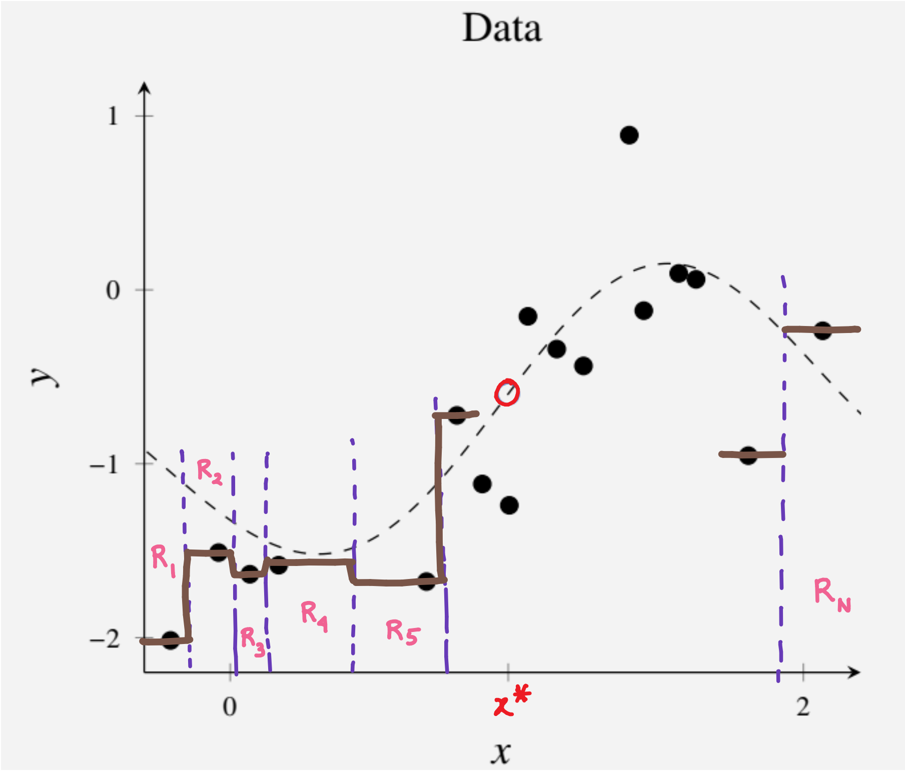
- Lets consider a Regression tree which is grown until each leaf node has only one data point

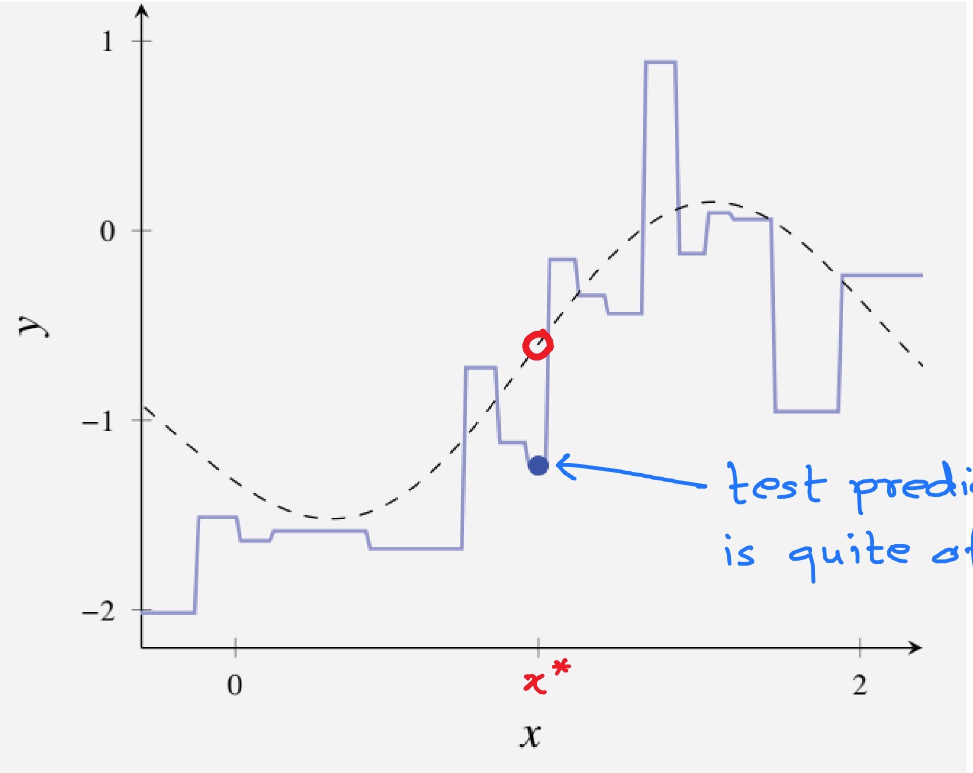




- On fitting a regression tree (with one data point in each leaf),

  we get an OVERFITTED Regression Tree



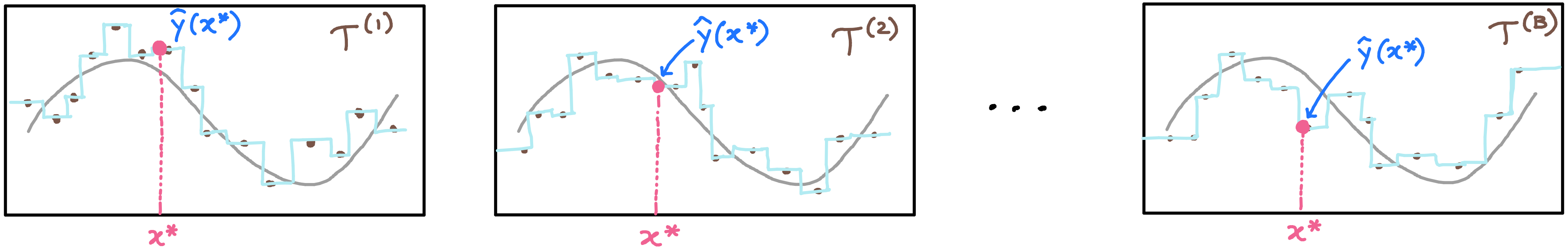


test prediction is quite off

- Due to overfitting, the resulting regression tree is a low-bias-high-variance model

  – high variance means the trained model is very sensitive to the training data; if the training data changes, the predictions change a lot

- Because of the noise in training data, we can think of the prediction $\hat{y}(x^*)$ from the trained model as a random variable
  - It means that if we had multiple datasets and we trained different RTs on them, each of their predictions $\hat{y}(x^*)$ would be different



- So if we assumed that we had access to B independent datasets $T^{(1)}, T^{(2)}, \dots, T^{(B)}$, then we could train a separate tree for each dataset and obtain separate predictions $\hat{y}_b(\underline{x}^*)$, $b = 1, 2, \dots, B$, then:
  - Each $y_b(\underline{x}^*)$ would have low bias and high variance
  - By averaging $\hat{y}(\underline{x}^*) = \frac{1}{B}\sum_{b=1}^{B}\hat{y}_b(\underline{x}^*)$, the bias is kept small, but the variance is reduced by a factor of B ! (Proof ?)

Probability detour – Variance reduction by averaging

Let $z_1, z_2, \dots, z_B$ be a collection of identically distributed but possibly dependent random variables, with

$$\left.\begin{array}{ll} \text{Mean :} & \mathbb{E}[z_b] = \mu \\ \text{Variance:} & \mathrm{Var}(z_b) = \sigma^2 \end{array}\right\} \quad \text{for } b = 1, 2, \dots, B$$

$$\text{Correlation:} \quad \mathrm{Corr}(z_i, z_j) = \rho \qquad i \neq j, \quad i,j = 1, 2, \dots, B$$

Then one can show that the mean and variance of the average $\frac{1}{B}\sum_{b=1}^{B} z_b$ are: (assuming $\rho > 0$)

$$\boxed{\mathbb{E}\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] = \mu, \qquad \mathrm{Var}\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] = \underbrace{\frac{1-\rho}{B}}_{\text{small for large } B}\sigma^2 + \rho\sigma^2}$$

- Problem : We only have access to one training dataset

- Solution : Bootstrap the data!

- Bootstrap is a method of artificially creating multiple datasets (of size N) out of one dataset (also of size N)
  - Sample N times with replacement from the original training data $T = \{ \underline{x}^{(i)}, y^{(i)} \}_{i=1}^{N}$
  - Repeat B times to generate B "bootstrapped" training datasets $\tilde{T}^{(1)}, \tilde{T}^{(2)}, \ldots, \tilde{T}^{(B)}$

- BAGGING
  - For each bootstrapped dataset $\tilde{T}^{(b)}$, we train a tree (basemodel) Averaging them,

$$\boxed{\hat{y}_{bag} = \frac{1}{B} \sum_{b=1}^{B} \tilde{y}^{b}(\underline{x})}$$

Bagging example with regression trees as basemodel

Assume that we have a training set

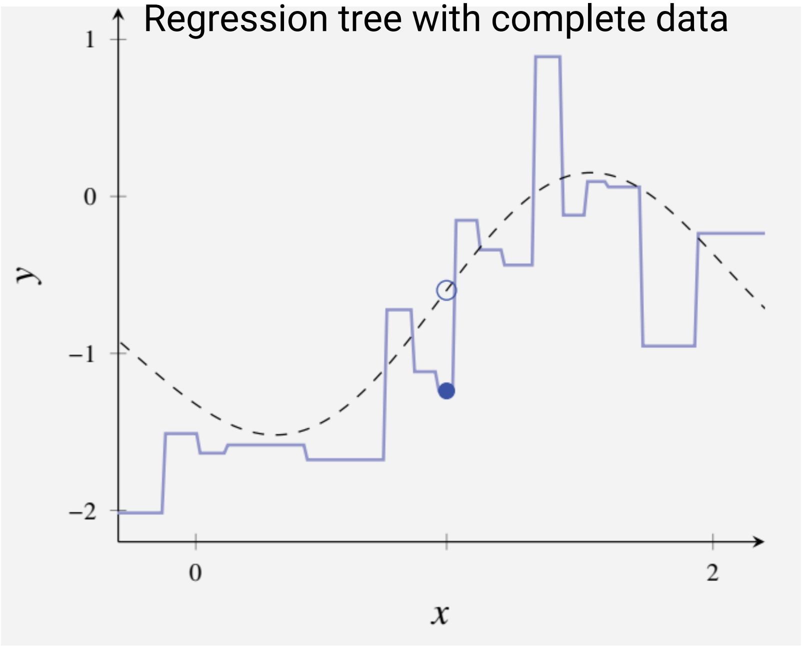
$$\mathcal{T} = \left\{ \left(\underline{x}^{(1)}, y^{(1)}\right), \left(\underline{x}^{(2)}, y^{(2)}\right), \left(\underline{x}^{(3)}, y^{(3)}\right), \dots, \left(\underline{x}^{(N)}, y^{(N)}\right) \right\}$$

- We generate, say, $B = 9$ datasets by bootstrapping:

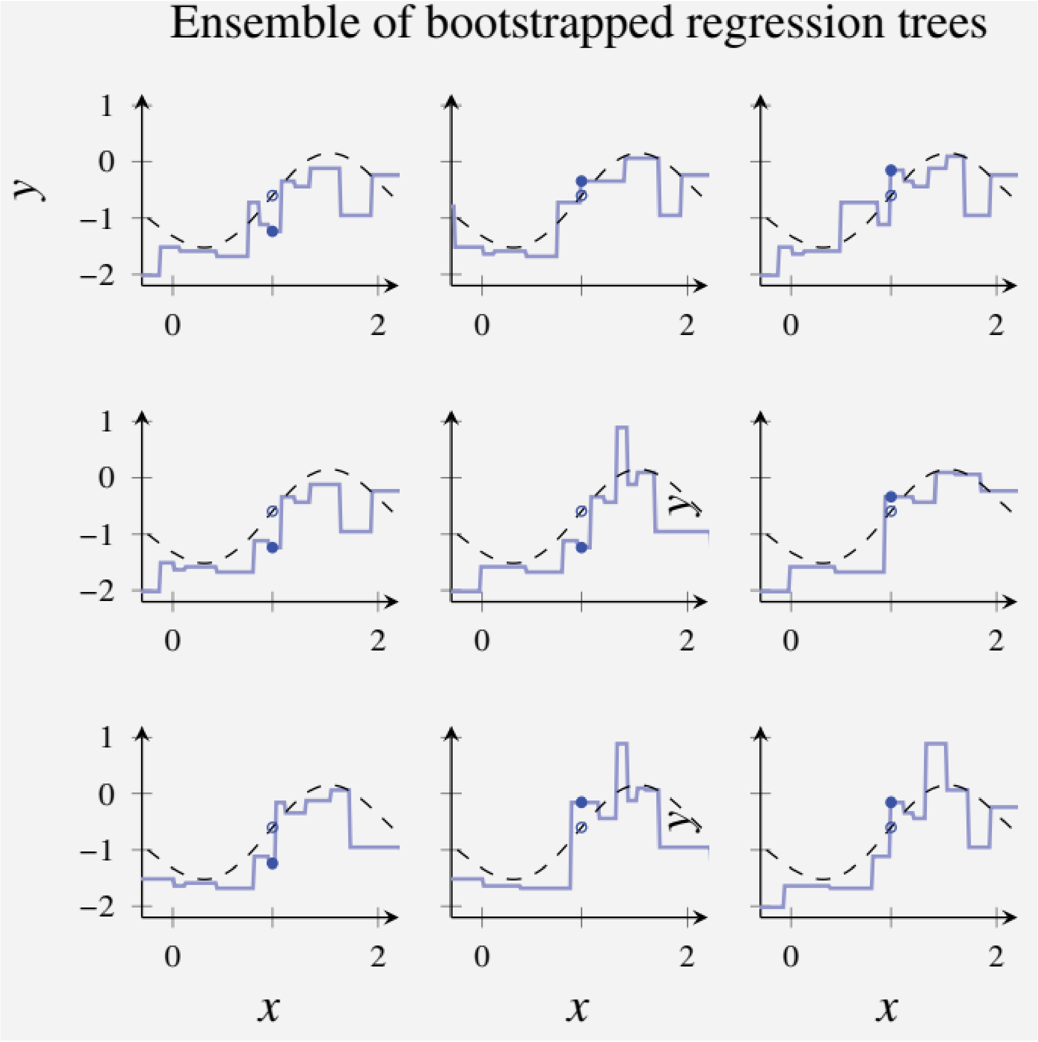
$$\begin{aligned}
\tilde{\mathcal{T}}^{(1)} &= \left\{ \left(\underline{x}^{(1)}, y^{(1)}\right), \left(\underline{x}^{(2)}, y^{(2)}\right), \left(\underline{x}^{(3)}, y^{(3)}\right), \dots, \left(\underline{x}^{(3)}, y^{(3)}\right) \right\} \\
\tilde{\mathcal{T}}^{(2)} &= \left\{ \left(\underline{x}^{(1)}, y^{(1)}\right), \left(\underline{x}^{(N)}, y^{(N)}\right), \left(\underline{x}^{(N)}, y^{(N)}\right), \dots, \left(\underline{x}^{(N)}, y^{(N)}\right) \right\} \\
&\;\;\vdots \qquad\qquad\qquad\qquad\qquad \vdots \\
\tilde{\mathcal{T}}^{(9)} &= \left\{ \left(\underline{x}^{(1)}, y^{(1)}\right), \left(\underline{x}^{(1)}, y^{(1)}\right), \left(\underline{x}^{(2)}, y^{(2)}\right), \dots, \left(\underline{x}^{(3)}, y^{(3)}\right) \right\}
\end{aligned}$$

- We compute $B = 9$ (deep) regression trees $\tilde{y}^{(1)}(\underline{x}), \tilde{y}^{(2)}(\underline{x}), \dots, \tilde{y}^{(9)}(\underline{x})$ one for each dataset $\tilde{\mathcal{T}}^{(1)}, \tilde{\mathcal{T}}^{(2)}, \dots, \tilde{\mathcal{T}}^{(9)}$, and average $\hat{y}_{\text{bag}} = \frac{1}{9} \sum_{b=1}^{9} \tilde{y}^{(b)}(\underline{x})$

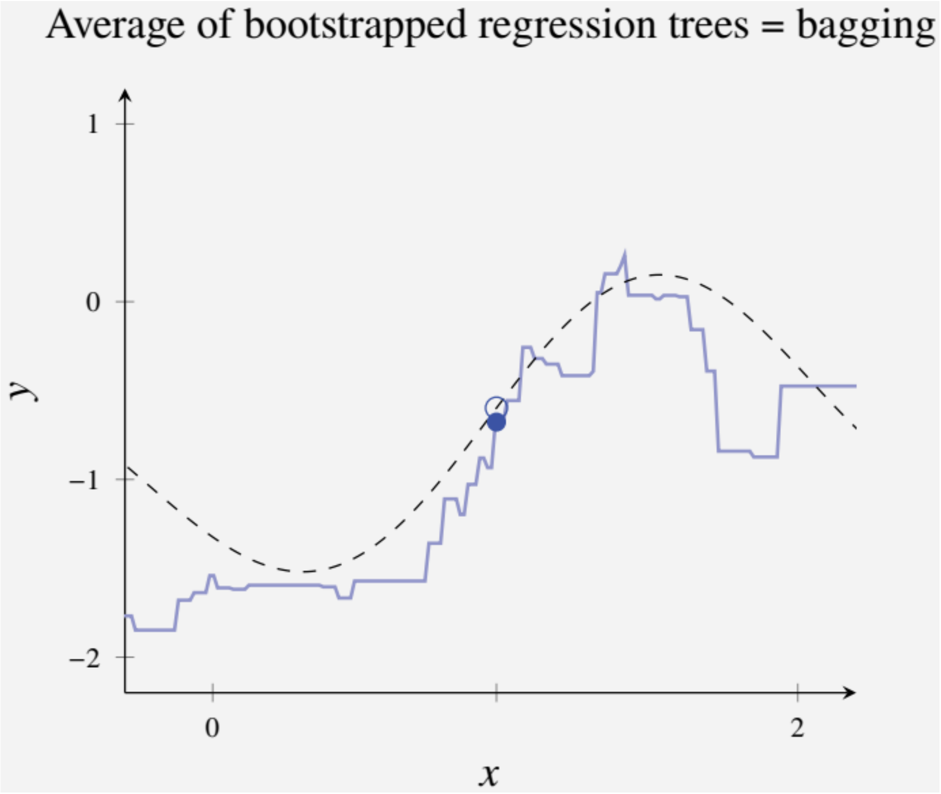
Bagging example with regression trees



Regression tree with complete data



Ensemble of bootstrapped regression trees



Average of bootstrapped regression trees = bagging

Bagging algorithm

- Training : Learn all base models

Data : Training dataset $\mathcal{T} = \{ \underline{x}^{(i)}, y^{(i)} \}_{i=1}^{N}$

Result : 'B' base models

```
for b = 1, ...., B do
    - Generate a bootstrap dataset $\widetilde{\mathcal{T}}^{(b)} = \{ \tilde{\underline{x}}^{(i)}, \tilde{y}^{(i)} \}_{i=1}^{N}$
    - Learn a base model from $\widetilde{\mathcal{T}}^{(b)}$
end
```

Obtain $\hat{y}_{bag}(\underline{x})$ by averaging : $\hat{y}_{bag}(\underline{x}) = \frac{1}{B} \sum_{i=1}^{B} \tilde{y}^{(b)}(\underline{x})$

The choice of 'B' is mainly guided by computational constraints

- Prediction with the base models

Data : 'B' base models and test input $\underline{x}^*$

Result : A prediction $\hat{y}_{bag}(\underline{x}^*)$

Use same formula

RANDOM FORESTS

- Bagging can greatly improve the performance of CART
  - Averaging over ensemble prediction, in case of regression trees
  - Majority vote over ensemble prediction, for classification trees

- However, the 'B' bootstrapped dataset are correlated!

  Therefore, the variance reduction due to averaging is diminished

(Recall

$$\mathrm{Var}\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] = \frac{1-\rho}{B}\sigma^2 + \rho\sigma^2$$

- No variance reduction when $\rho = 1$
- Highest variance reduction when $\rho = 0$)

- Idea of Random Forest : De-correlate the 'B' trees by injecting additional randomness when constructing each tree

Random forest algorithm

Inputs : $T = \{ \underline{x}^{(i)}, y^{(i)} \}_{i=1}^{N}$; $\underline{x} \in \mathbb{R}^{p}$

```
for $b = 1$ to $B$, do   (can run in parallel)
    (a) Draw a bootstrap dataset $\tilde{T}^{(b)}$ of size $N$ from $T$
    (b) Grow a regression (or classification) tree by repeating the steps
        below, until a minimum node size is reached :
        - Select a random subset consisting of $q \leq p$ inputs
        - Find the best splitting variable $x_j$ among the 'q' selected inputs
        - Split the node into two children with $\{x_j \leq s\}$ and $\{x_j > s\}$
```

Thumb rule

$q = \sqrt{p}$ (for CT)

$q = p/3$ (for RT)

Final model is the average of the 'B' ensemble members

$$\boxed{\hat{y}^{rf} = \frac{1}{B} \sum_{b=1}^{B} \tilde{y}^{(b)}}$$

- For identically distributed random variables $\{z_b\}_{b=1}^{B}$

$$\mathrm{Var}\left[\frac{1}{B}\sum_{b=1}^{B} z_b\right] = \frac{1-\rho}{B}\sigma^2 + \rho\sigma^2$$

- The random input selection used in random forests:

  - increases the bias, but often very slowly ↓
  - adds to the variance (σ^2) of each tree ↓
  - reduces the correlation (ρ) between member trees ↑↑↑

- The reduction in correlation typically has a dominant effect

  ⇒ leads to an overall reduction in error

- Bagging is a general technique → can be used with any base model

  Random forest consider base models as classification or regression trees