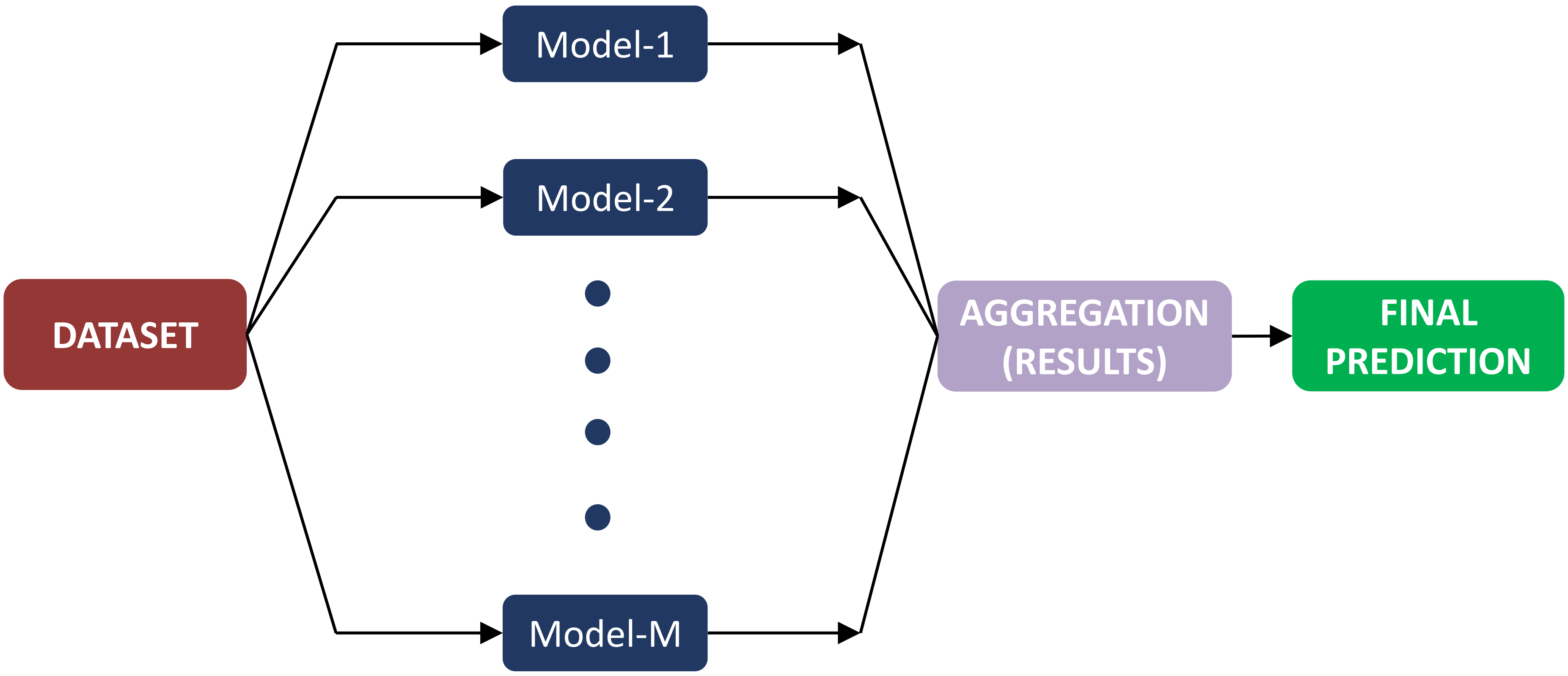
Ensemble Methods

Dripta Mj
Department of Mathematics
Ramakrishna Mission Vivekananda Educational and Research Institute
Belur Math, India

Ensemble



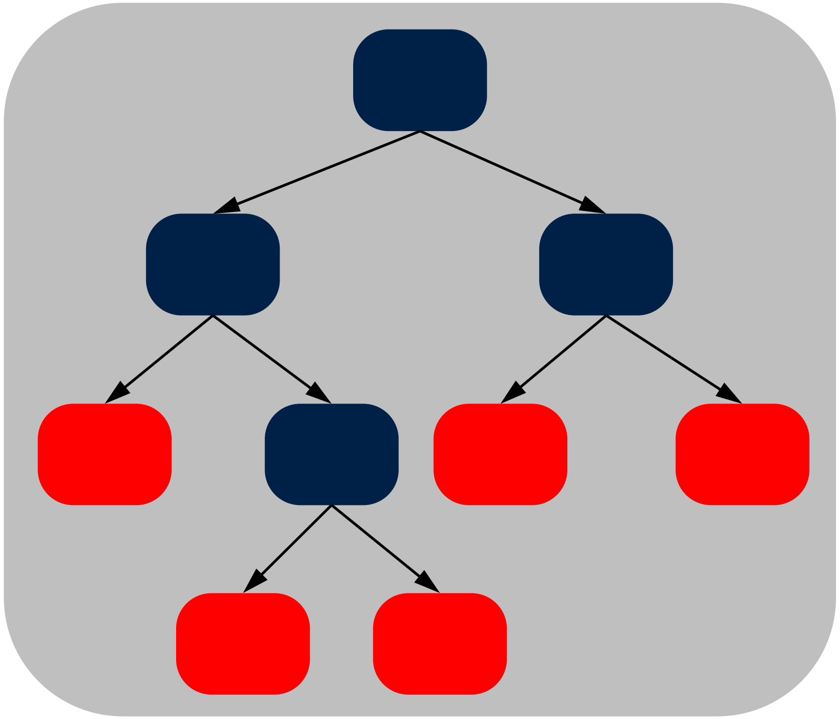
Model-1
Model-2
DATASET
AGGREGATION (RESULTS)
FINAL PREDICTION
Model-M

Ensemble methods

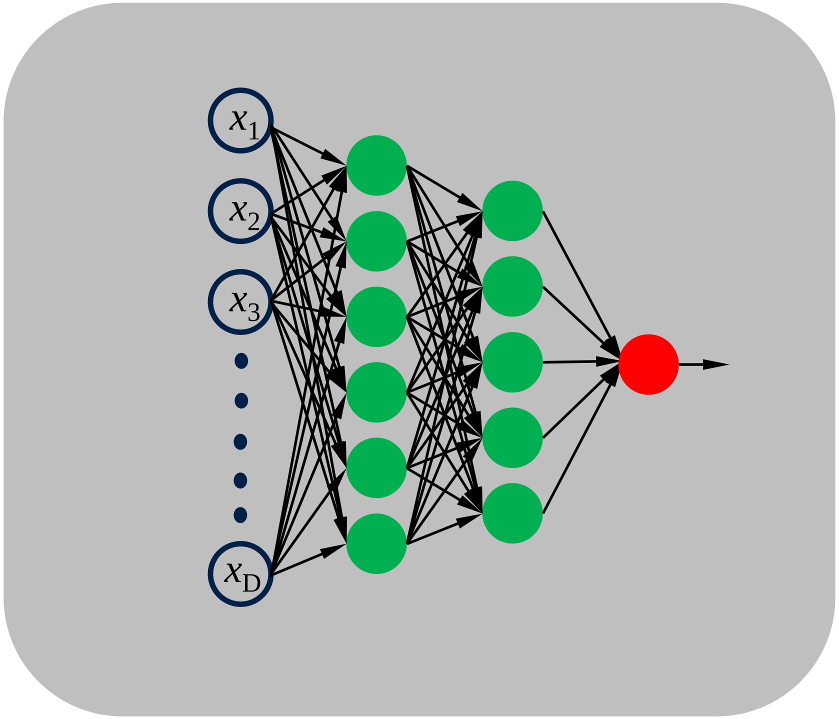
- Idea is to use multiple learners and combine their predictions.
  - E.g. in ensemble of classifiers, predictions from a set of classifiers are combined
- Consider a committee of M models with uncorrelated errors, then by simply averaging the outputs of the M models the average error can be reduced by a factor of M.
  - Although in practice the errors are typically correlated and so the reduction is smaller.
- Ensemble methods can transform a "weak" learner into a strong model by taking combinations of the former.
- Ensemble methods combine models such that the ensemble achieves better performance than an individual model on average.

Base Models – examples

Decision Tree



Neural Network



Can we reduce variance?

Original decomposition:

$$\mathbb{E}_{\mathbf{x},y,\mathcal{D}}\left[\left(g_{\mathcal{D}}(\mathbf{x}) - y\right)^2\right] = \underbrace{\mathbb{E}_{\mathbf{x},\mathcal{D}}\left[\left(g_{\mathcal{D}}(\mathbf{x}) - \overline{g}(\mathbf{x})\right)^2\right]}_{\text{Variance}} + \underbrace{\mathbb{E}_{\mathbf{x}}\left[\left(\overline{g}(\mathbf{x}) - \overline{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathbf{x},y}\left[\left(\overline{y}(\mathbf{x}) - y\right)^2\right]}_{\text{Noise}}$$

- Suppose we have M different training datasets: $\mathcal{D}_1$, $\mathcal{D}_2$, ..., $\mathcal{D}_M$
- Can train a separate model on each of them: $g_{\mathcal{D}_1}, g_{\mathcal{D}_2}, \ldots., g_{\mathcal{D}_M}$
- Predictions can be obtained as the average of the trained models

$$\widehat{g}(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} g_{\mathcal{D}_m}(\mathbf{x}) \to \overline{g}(\mathbf{x}) \qquad \text{as} \quad M \to \infty$$

- As $\widehat{g}(\mathbf{x}) \to \overline{g}(\mathbf{x})$, the variance term $\mathbb{E}\left[(\widehat{g}(\mathbf{x}) - \overline{g}(\mathbf{x}))^2\right] \to 0$
- Issue: Don't have M different training datasets.

Bootstrap Aggregating

- Bagging: Bootstrap Aggregating
- Bootstrap: Replicate given dataset by sampling with replacement.
- Example:

$$\boxed{\text{Original data}: \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(5)}\}}$$

$$\text{Bootstrap 1}: \{\mathbf{x}^{(4)}, \mathbf{x}^{(1)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(2)}\}$$

$$\text{Bootstrap 2}: \{\mathbf{x}^{(5)}, \mathbf{x}^{(5)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(2)}\}$$

$$\text{Bootstrap 3}: \{\mathbf{x}^{(2)}, \mathbf{x}^{(1)}, \mathbf{x}^{(1)}, \mathbf{x}^{(3)}, \mathbf{x}^{(1)}\}$$

- Bootstrap samples are independent realizations of the original data.

Algorithm

for $m = 1$ to M **do**

- Draw a bootstrap sample dataset $\mathcal{D}_m$ from the training dataset $\mathcal{D}$.
  - The size of $\mathcal{D}_m$ should be same as $\mathcal{D}$.
- Train a base model T_m on the dataset $\mathcal{D}_m$.

end for

- Output ensemble models: $\{T_1, T_2, ..., T_M\}$
- Prediction for a new example $\mathbf{x}^*$:
  - Regression:

$$\overline{y}_M(\mathbf{x}^*) = \frac{1}{M} \sum_{m=1}^{M} T_m(\mathbf{x}^*)$$

  - Classification:

$$\overline{y}_M(\mathbf{x}^*) = \text{majority vote}\big\{C_1(\mathbf{x}^*), C_2(\mathbf{x}^*), ..., C_M(\mathbf{x}^*)\big\}$$

where $C_m(\mathbf{x}^*)$ is the class prediction of the mth model.

Bagging – Random Forests

- Bagging gives the average of predictions of a model fit to many Bootstrap samples.
- Bagging reduces the variance as it averages the fits from many independent datasets (bootstrap samples).
- Issue with Bagging:
  - Similar decision trees can be formed by different Bootstrap samples.
- Random Forests address the issue.
- In Random Forests, each Bootstrap sample produces a different decision tree.
- The final output is the average of the predictions from all the trees.

Bootstrap samples

Original Dataset

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
1	Hot	Good	Interesting	Medium	Yes
2	Cold	Average	Boring	High	Yes
3	Cold	Sick	Mediocre	Medium	No
4	Mild	Average	Interesting	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
7	Rainy	Good	Mediocre	Medium	No
8	Mild	Good	Mediocre	Medium	Yes

BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No

BOOTSTRAP-2

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
6	Hot	Good	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
1	Hot	Good	Interesting	Medium	Yes
4	Mild	Average	Interesting	High	Yes
5	Rainy	Sick	Mediocre	Low	No
1	Hot	Good	Interesting	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No
5	Rainy	Sick	Mediocre	Low	No

BOOTSTRAP-3

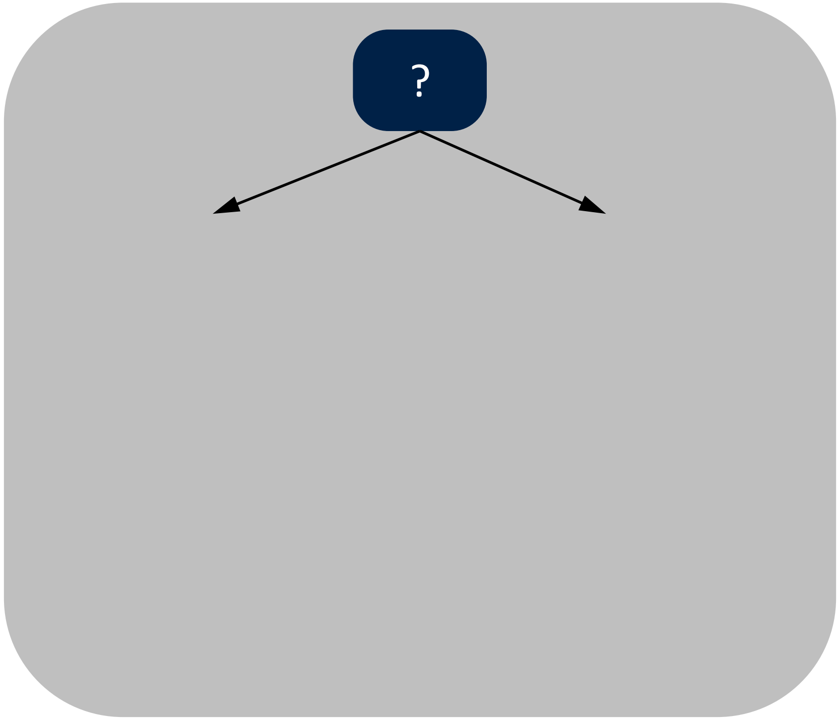
Instance	Weather	Health	Teaching	Topic Importance	Going to class?
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
4	Mild	Average	Interesting	High	Yes
4	Mild	Average	Interesting	High	Yes
1	Hot	Good	Interesting	Medium	Yes
5	Rainy	Sick	Mediocre	Low	No
2	Cold	Average	Boring	High	Yes
4	Mild	Average	Interesting	High	Yes

Random Forests – example

- k variables are selected at random, where $k < D$.
  - Here $k = 2$.
- Of the k selected features, the best feature (according to some criteria) is used for splitting.

BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No



Random Forests – example

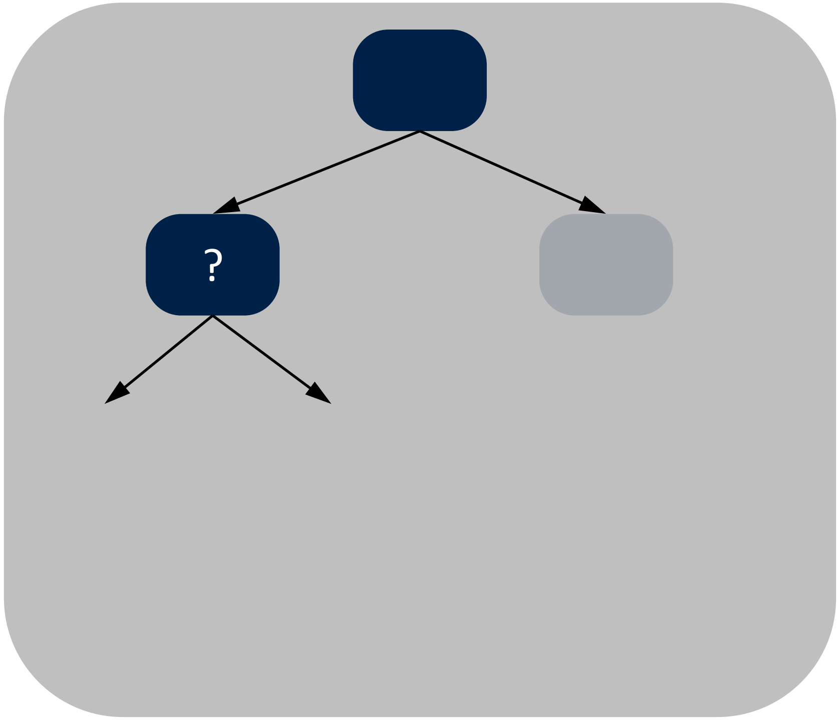
- k variables are selected at random, where $k < D$.
  - Here $k = 2$.

- Of the k selected features, the best feature (according to some criteria) is used for splitting.

- At the next node, k features are again selected at random and splitting is done using the best feature.

BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No

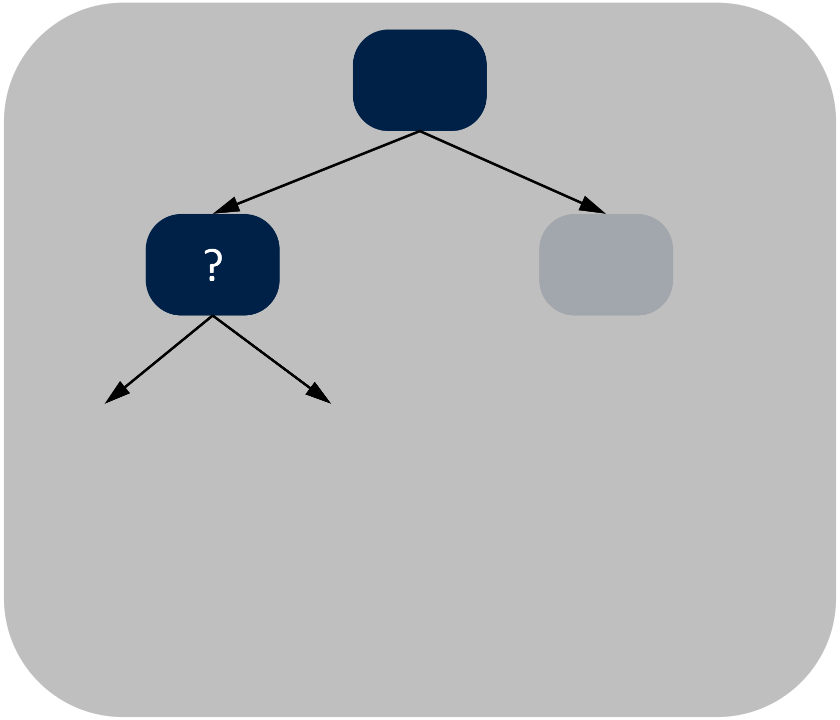


Random Forests – example

- k variables are selected at random, where $k < D$.
  - Here $k = 2$.
- Of the k selected features, the best feature (according to some criteria) is used for splitting.
- At the next node, k features are again selected at random and splitting is done using the best feature.

BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No

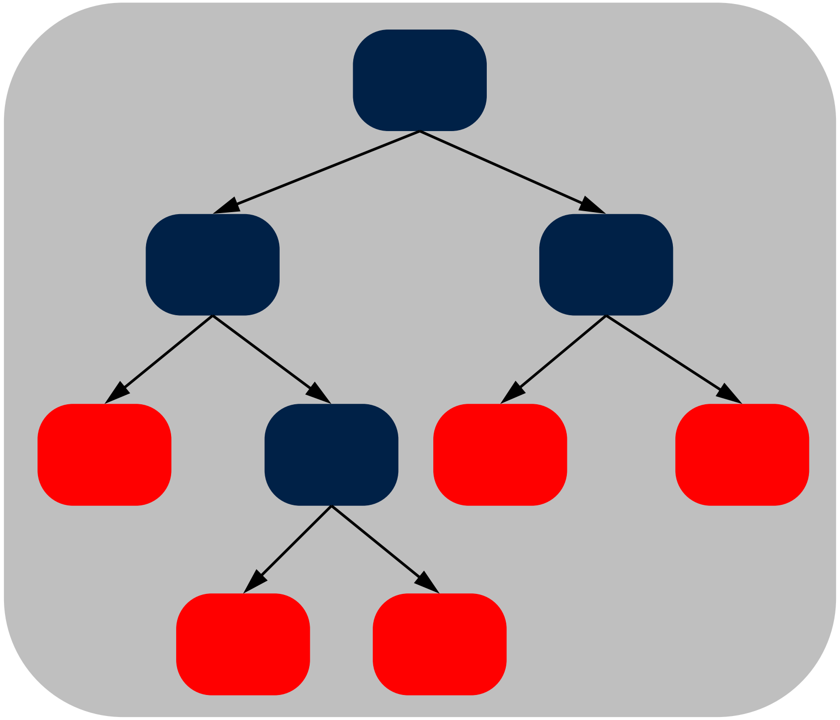


Random Forests – example

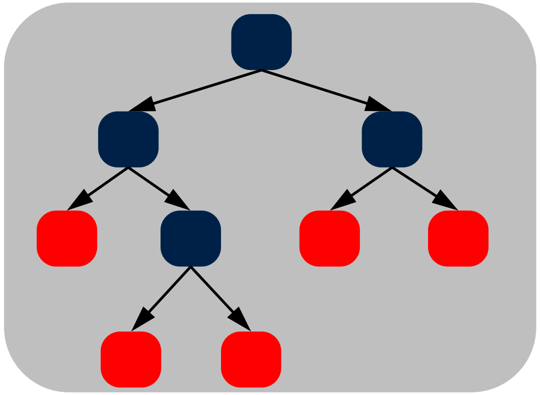
- k variables are selected at random, where $k < D$.
  - Here $k = 2$.
- Of the k selected features, the best feature (according to some criteria) is used for splitting.
- At the next node, k features are again selected at random and splitting is done using the best feature.
- The process is repeated till the end.

BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No

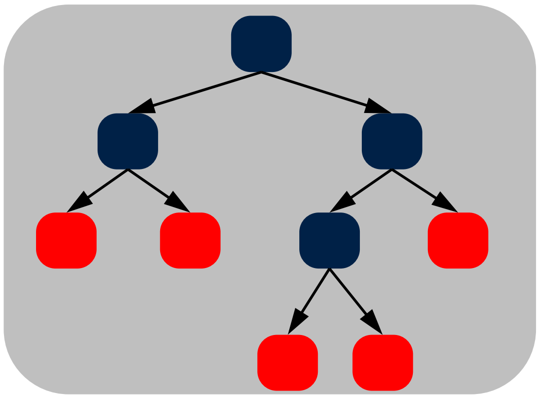


Tree ensembles



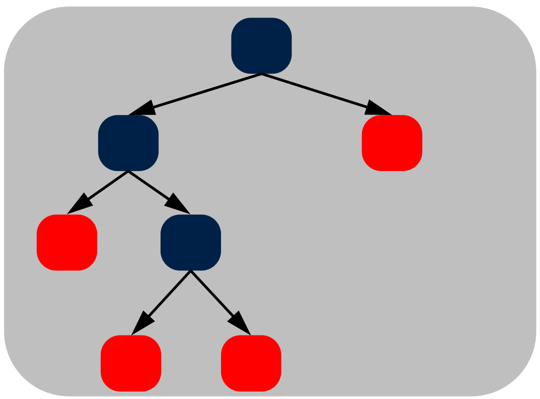
BOOTSTRAP-1

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
7	Rainy	Good	Mediocre	Medium	No
2	Cold	Average	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
5	Rainy	Sick	Mediocre	Low	No
6	Hot	Good	Boring	High	Yes
8	Mild	Good	Mediocre	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No



BOOTSTRAP-2

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
6	Hot	Good	Boring	High	Yes
6	Hot	Good	Boring	High	Yes
1	Hot	Good	Interesting	Medium	Yes
4	Mild	Average	Interesting	High	Yes
5	Rainy	Sick	Mediocre	Low	No
1	Hot	Good	Interesting	Medium	Yes
7	Rainy	Good	Mediocre	Medium	No
5	Rainy	Sick	Mediocre	Low	No



BOOTSTRAP-3

Instance	Weather	Health	Teaching	Topic Importance	Going to class?
6	Hot	Good	Boring	High	Yes
2	Cold	Average	Boring	High	Yes
4	Mild	Average	Interesting	High	Yes
4	Mild	Average	Interesting	High	Yes
1	Hot	Good	Interesting	Medium	Yes
5	Rainy	Sick	Mediocre	Low	No
2	Cold	Average	Boring	High	Yes
4	Mild	Average	Interesting	High	Yes

Algorithm – regression

for m = 1 to M **do**

- Draw a bootstrap sample dataset $\mathcal{D}_m$ from the training dataset $\mathcal{D}$. The size of $\mathcal{D}_m$ should be same as $\mathcal{D}$.
- Construct a decision tree T_m using the bootstraped dataset $\mathcal{D}_m$ based on the following rules:
  - Select k features randomly from the D features.
  - From the k features, select the best feature (based on some criteria) for splitting
  - Split the node using the best feature.
  - Repeat the process till the stopping criteria is attained.

end for

- Output tree ensembles: $\{T_1, T_2, ..., T_M\}$
- Prediction at a new point $\mathbf{x}^*$:
  - Regression:

$$\overline{y}_M(\mathbf{x}^*) = \frac{1}{M}\sum_{m=1}^{M} T_m(\mathbf{x}^*)$$

Algorithm – classification

for m = 1 to M **do**

- Draw a bootstrap sample dataset $\mathcal{D}_m$ from the training dataset $\mathcal{D}$. The size of $\mathcal{D}_m$ should be same as $\mathcal{D}$.
- Construct a decision tree T_m using the bootstraped dataset $\mathcal{D}_m$ based on the following rules:
  - Select k features randomly from the D features.
  - From the k features, select the best feature (based on some criteria) for splitting
  - Split the node using the best feature.
  - Repeat the process till the stopping criteria is attained.

end for

- Output tree ensembles: $\{T_1, T_2, ..., T_M\}$
- Prediction at a new point $\mathbf{x}^*$:
  - Classification:

$$\overline{y}_M(\mathbf{x}^*) = \text{majority vote}\{C_1(\mathbf{x}^*), C_2(\mathbf{x}^*), ..., C_M(\mathbf{x}^*)\}$$

where $C_m(\mathbf{x}^*)$ is the class prediction of the mth random forest.

Out-of-bag error

- Test error can be assessed without cross-validation or validation set
- On an average, each bagged tree uses around two-third of the original training dataset.
  - The left-out examples are known as "out-of-bag" (OOB) examples.
- Example:

$$\text{Original data}: \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(5)}\}$$

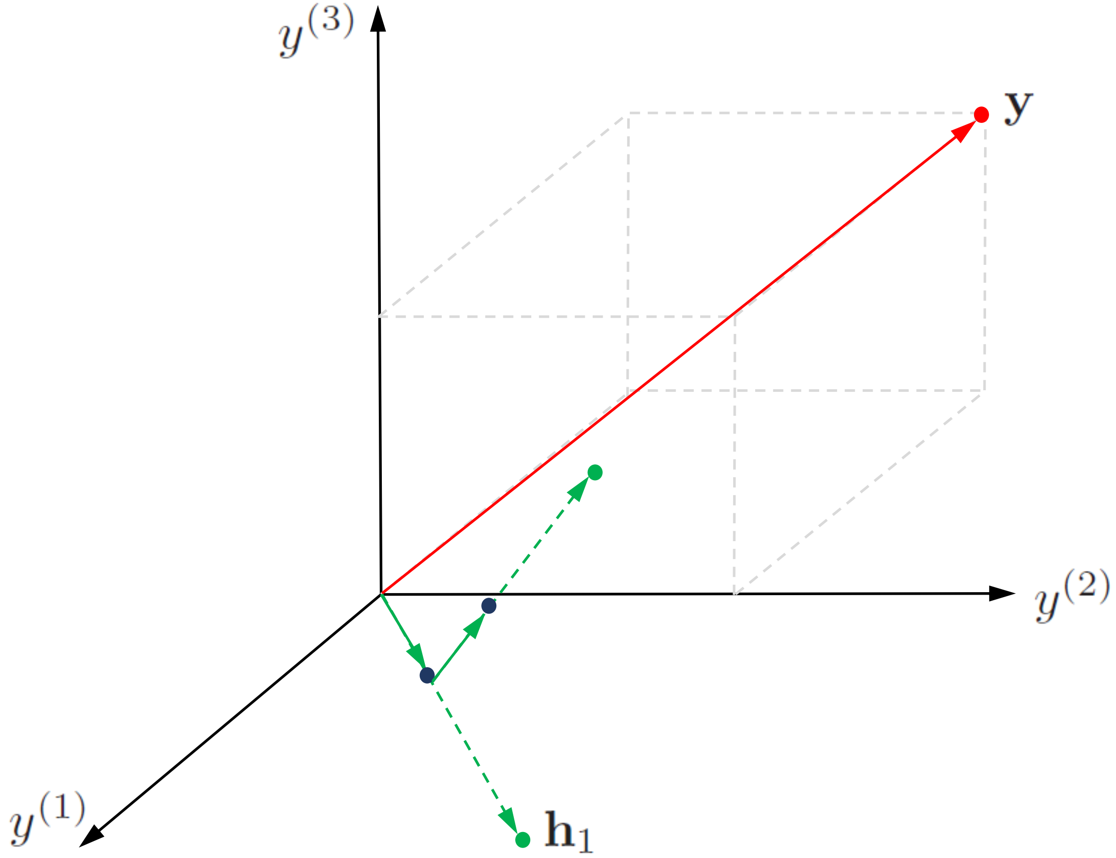
	Bootstraps	OOB examples
Bootstrap 1 :	$\{\mathbf{x}^{(4)}, \mathbf{x}^{(1)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(2)}\}$	$\{\mathbf{x}^{(5)}\}$
Bootstrap 2 :	$\{\mathbf{x}^{(5)}, \mathbf{x}^{(5)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(2)}\}$	$\{\mathbf{x}^{(1)}, \mathbf{x}^{(4)}\}$
Bootstrap 3 :	$\{\mathbf{x}^{(2)}, \mathbf{x}^{(1)}, \mathbf{x}^{(1)}, \mathbf{x}^{(3)}, \mathbf{x}^{(1)}\}$	$\{\mathbf{x}^{(4)}, \mathbf{x}^{(5)}\}$

Out-of-bag error

- Test error can be assessed without cross-validation or validation set
- On an average, each bagged tree uses around two-third of the original training dataset.
  - The left-out examples are known as "out-of-bag" (OOB) examples.
- The prediction for the nth example $\mathbf{x}^{(n)}$ can be made using the bagging trees where $\mathbf{x}^{(n)}$ was an OOB example.
- Final OOB prediction:
  - Regression: Average of the predicted outputs.
  - Classification: Majority vote.
- In this way OOB predictions can obtained for all the N examples in the training dataset.
- OOB error: Error can be computed from the OOB predictions of the N examples.

BOOSTING

Intuition

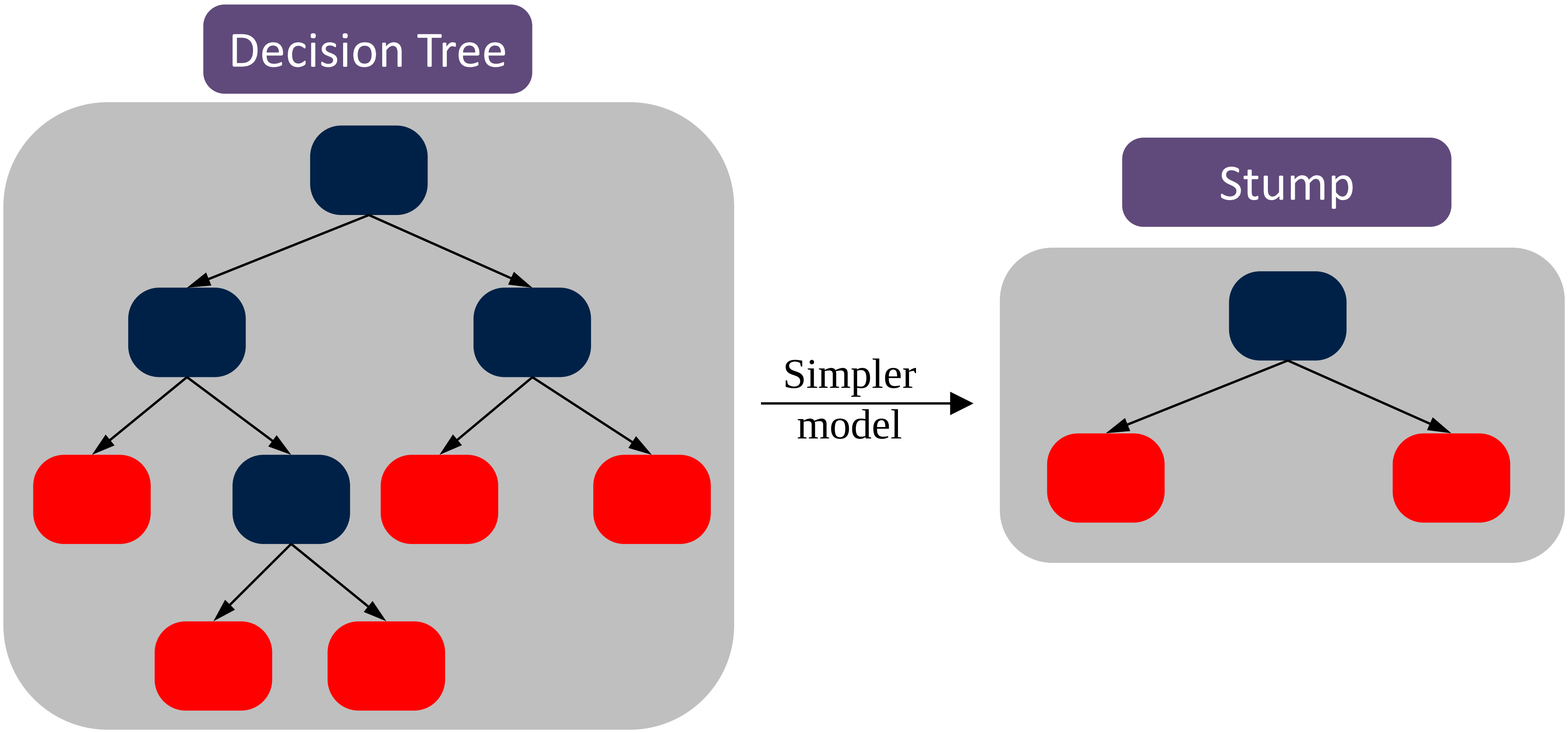


- Outputs: $\mathbf{y} = \left[y^{(1)}, y^{(2)}, y^{(3)}\right]^{\mathrm{T}}$
- Predictions of 1st weak learner: $\mathbf{h}_1 = \left[h_1(\mathbf{x}^{(1)}), h_1(\mathbf{x}^{(2)}), h_1(\mathbf{x}^{(3)})\right]^{\mathrm{T}}$

Figures for illustration only.

Weak learner

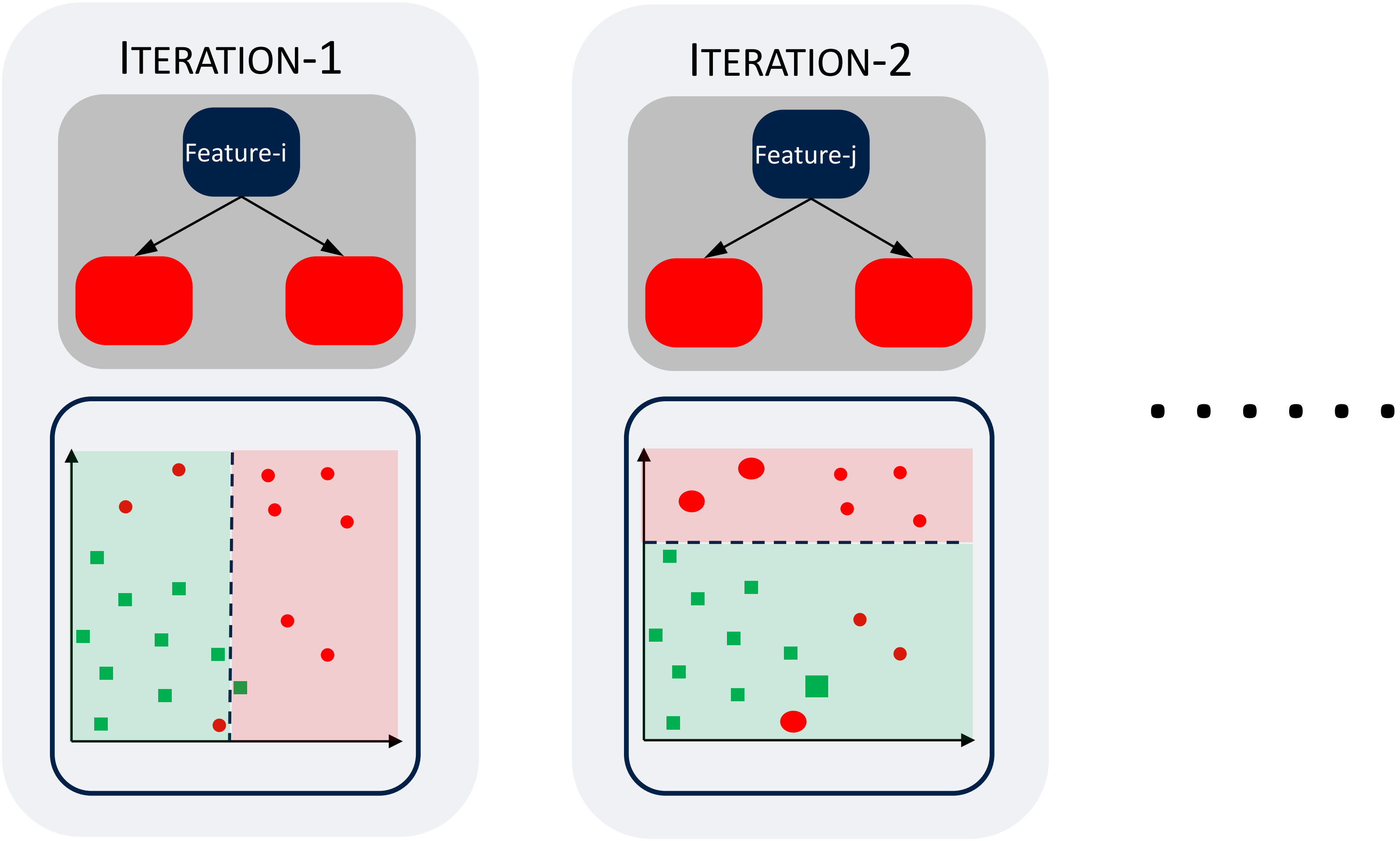
- Transforms a weak learning algorithm into a strong one.



– A weak learner performs just better than random guessing.

Additive ensemble

- At each stage, a weak learner is introduced to compensate for the shortcomings of existing weak learners.



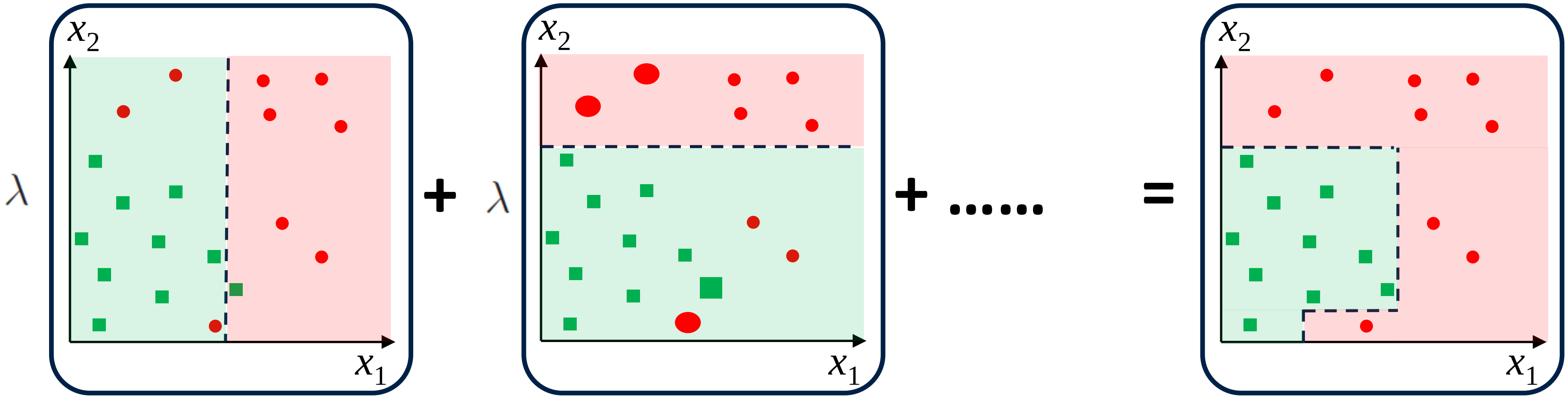
Figures for illustration only.

Additive ensemble

- Define an additive ensemble model at the end of the Tth iteration as

$$H_T(\mathbf{x}) = \sum_{t=1}^{T} \lambda h_t(\mathbf{x})$$

where λ is the step-size and h_t is the classifier that is added to the ensemble at the tth iteration.



Figures for illustration only.

Loss function

- Let $\widehat{\mathcal{L}}(H_T(\mathbf{x}^{(n)}), y^{(n)})$ be a convex, differentiable loss function where $H_T(\mathbf{x}^{(n)})$ is the ensemble prediction and $y^{(n)}$ is the observed output for a input $\mathbf{x}^{(n)}$.

- The overall loss can then written as

$$\mathcal{L}(H_T) = \frac{1}{N} \sum_{n=1}^{N} \widehat{\mathcal{L}}(H_T(\mathbf{x}^{(n)}), y^{(n)})$$

- At the $(t+1)$th iteration, a new weak learner is added to the ensemble such that

$$h_{t+1} = \arg\min_{h \in \mathcal{H}} \mathcal{L}(H_t + \lambda h)$$

where λ is the step size.

- Employing Taylor approximation on $\mathcal{L}(H_t + \lambda h)$ gives

$$\begin{aligned} \mathcal{L}(H_t + \lambda h) &\approx \mathcal{L}(H_t) + \lambda < \nabla \mathcal{L}(H_t), h > \\ &\approx \mathcal{L}(H_t) + \lambda \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial (H_t(\mathbf{x}^{(n)}))} h(\mathbf{x}^{(n)}) \end{aligned}$$

Loss function

- Therefore

$$
\begin{aligned}
h_{t+1} &= \arg\min_{h \in \mathcal{H}} \mathcal{L}\big(H_t + \lambda h\big) \\
&= \arg\min_{h \in \mathcal{H}} \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial\big(H_t(\mathbf{x}^{(n)})\big)} h(\mathbf{x}^{(n)}) \\
&= \arg\min_{h \in \mathcal{H}} \sum_{n=1}^{N} r_{t,n} h(\mathbf{x}^{(n)})
\end{aligned}
$$

where $r_{t,n} = \dfrac{\partial \mathcal{L}}{\partial\big(H_t(\mathbf{x}^{(n)})\big)}$

- The loss function $\mathcal{L}$ is reduced as long as $\sum_{n=1}^{N} r_{t,n} h(\mathbf{x}^{(n)}) < 0$.

General boosting algorithm

Intialize $H_0 = \mathbf{0}$

for $t = 0$ to $T - 1$ **do**

$$h_{t+1} = \arg\min_{h \in \mathcal{H}} \sum_{n=1}^{N} r_{t,n} h(\mathbf{x}^{(n)})$$

if $\displaystyle\sum_{n=1}^{N} r_{t,n} h(\mathbf{x}^{(n)}) < 0$ **then**

$$H_{t+1} = H_t + \lambda h_{t+1}$$

else

return H_t

end if

end for

return H_T

AdaBoost

- Binary classification problem – $y^{(n)} \in \{-1, 1\}$.
- Weak learners $h \in \mathcal{H}$ also have outputs $h(\mathbf{x}^{(n)}) \in \{-1, 1\}$.
- Loss function:

$$\mathcal{L}(H) = \sum_{n=1}^{N} \exp\left[-y^{(n)} H(\mathbf{x}^{(n)})\right]$$

- Gradient of loss function:

$$r_{t,n} = \frac{\partial \mathcal{L}}{\partial \left[H(\mathbf{x}^{(n)})\right]} = -y^{(n)} \exp\left[-y^{(n)} H(\mathbf{x}^{(n)})\right]$$

- Method enables computation of best step-size λ.

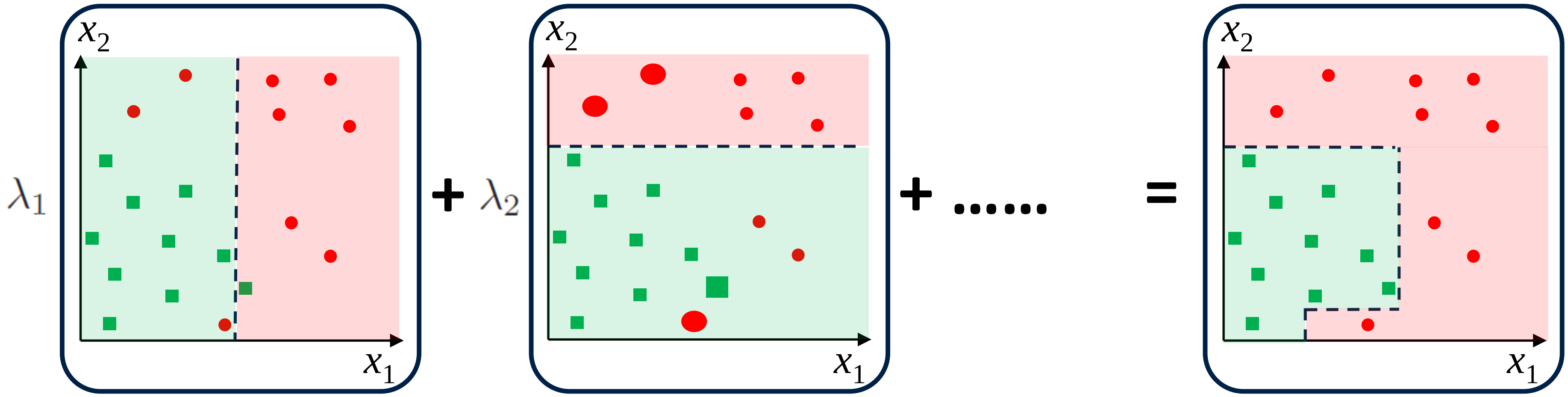
Weights

- Define $w^{(n)}$ as

$$w^{(n)} = \frac{\exp\left[-y^{(n)}H(\mathbf{x}^{(n)})\right]}{\sum_{j=1}^{N}\exp\left[-y^{(j)}H(\mathbf{x}^{(j)})\right]}$$

  - The denominator acts as a normalizing factor to give $\sum_{n=1}^{N} w^{(n)} = 1$.

- Each weight $w^{(n)}$ is the relative contribution of the data point $(\mathbf{x}^{(n)}, y^{(n)})$ to the overall loss.



Figures for illustration only.

Adaboost training

- Optimal "weak-learner" at the $(t+1)$th iteration:

$$h_{t+1} = \arg\min_{h \in \mathcal{H}} \sum_{n=1}^{N} r_{t,n} h(\mathbf{x}^{(n)})$$

$$= \arg\min_{h \in \mathcal{H}} \epsilon \qquad \text{where} \qquad \underbrace{\epsilon = \sum_{n:h(\mathbf{x}^{(n)}) \neq y^{(n)}} w^{(n)}}_{\text{weighted classification error}}$$

- Optimal step-size λ at the $(t+1)$th iteration:

$$\lambda_{t+1} = \arg\min_{\lambda} \mathcal{L}(H_t + \lambda h_{t+1})$$

- Differentiating the objective function w.r.t. λ and equating to 0:

$$\frac{\partial \sum_{n=1}^{N} \exp\left[-y^{(n)} (H_t + \lambda h_{t+1}) \right]}{\partial \lambda} = 0 \qquad \Rightarrow \boxed{\lambda_{t+1} = \frac{1}{2} \log\left(\frac{1-\epsilon}{\epsilon} \right)}$$

- The optimal step-size leads to fast convergence of the AdaBoost algorithm.

Adaboost algorithm

Initialize $H_0 = \mathbf{0}$ and $w^{(n)} = 1/N$, $n = 1, 2, \ldots N$

for $t = 0$ to $T - 1$ **do**

$$h_{t+1} = \arg\min_{h \in \mathcal{H}} \epsilon$$

if $\epsilon < 1/2$ **then**

$$\lambda_{t+1} = \frac{1}{2} \log\left(\frac{1-\epsilon}{\epsilon}\right)$$

$$H_{t+1} = H_t + \lambda_{t+1} h_{t+1}$$

$$w^{(n)} \leftarrow \frac{w^{(n)} \exp(-\lambda_{t+1} h(\mathbf{x}^{(n)}) y^{(n)})}{2\sqrt{\epsilon(1-\epsilon)}} \qquad n = 1, 2, \ldots N$$

else

return H_t

end if

end for

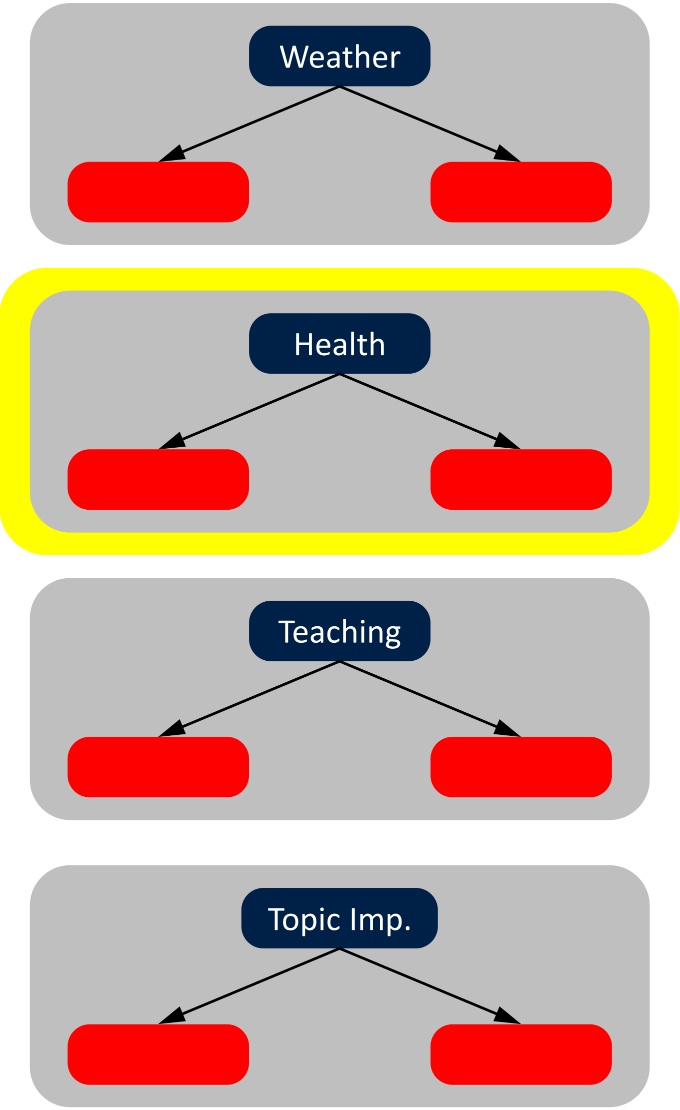
return H_T

AdaBoost (with decision stumps)

Original Dataset

Instance	Weather	Health	Teaching	Topic Importance	Going to class?	$w_1^{(n)}$
1	Hot	Good	Interesting	Medium	Yes	1/8
2	Cold	Average	Boring	High	Yes	1/8
3	Cold	Sick	Mediocre	Medium	No	1/8
4	Mild	Average	Interesting	High	Yes	1/8
5	Rainy	Sick	Mediocre	Low	No	1/8
6	Hot	Good	Boring	High	Yes	1/8
7	Rainy	Good	Mediocre	Medium	No	1/8
8	Mild	Good	Mediocre	Medium	Yes	1/8

- One example of weak classifier – Decision stumps.
  - A decision stump is a one-level decision tree comprising one root and terminal nodes.
- Can use a separate stump for splitting w.r.t. a particular feature.
- Select the stump giving the least weighted error
  - Suppose the feature Health gives the best result.

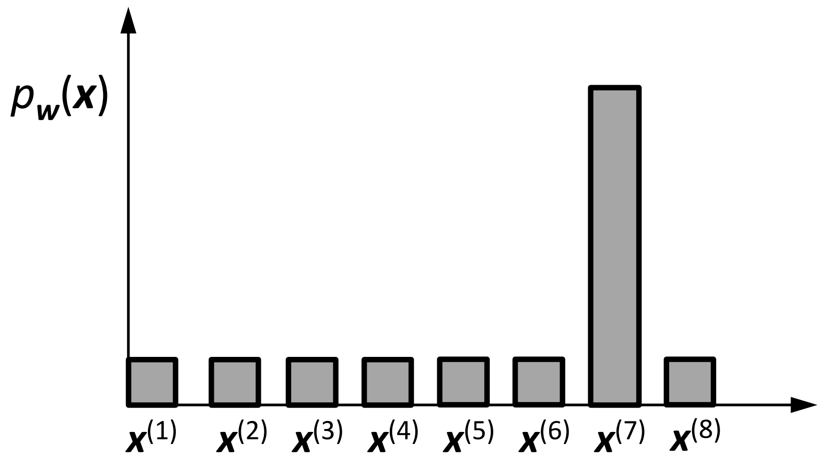


Figures for illustration only.

AdaBoost (with decision stumps)

Original Dataset

Instance	Weather	Health	Teaching	Topic Importance	Going to class?	$w_1^{(n)}$	$w_2^{(n)}$
1	Hot	Good	Interesting	Medium	Yes	1/8	0.07
2	Cold	Average	Boring	High	Yes	1/8	0.07
3	Cold	Sick	Mediocre	Medium	No	1/8	0.07
4	Mild	Average	Interesting	High	Yes	1/8	0.07
5	Rainy	Sick	Mediocre	Low	No	1/8	0.07
6	Hot	Good	Boring	High	Yes	1/8	0.07
7	Rainy	Good	Mediocre	Medium	No	1/8	0.50
8	Mild	Good	Mediocre	Medium	Yes	1/8	0.07



- Compute λ.
- Update the weights.
- For determining the next weak classifier, algorithms use one of the following two ways:
  - same data with updated weights.
  - samples from the training dataset according to the distribution $p_{\mathbf{w}}(\mathbf{x})$.

Figures for illustration only.