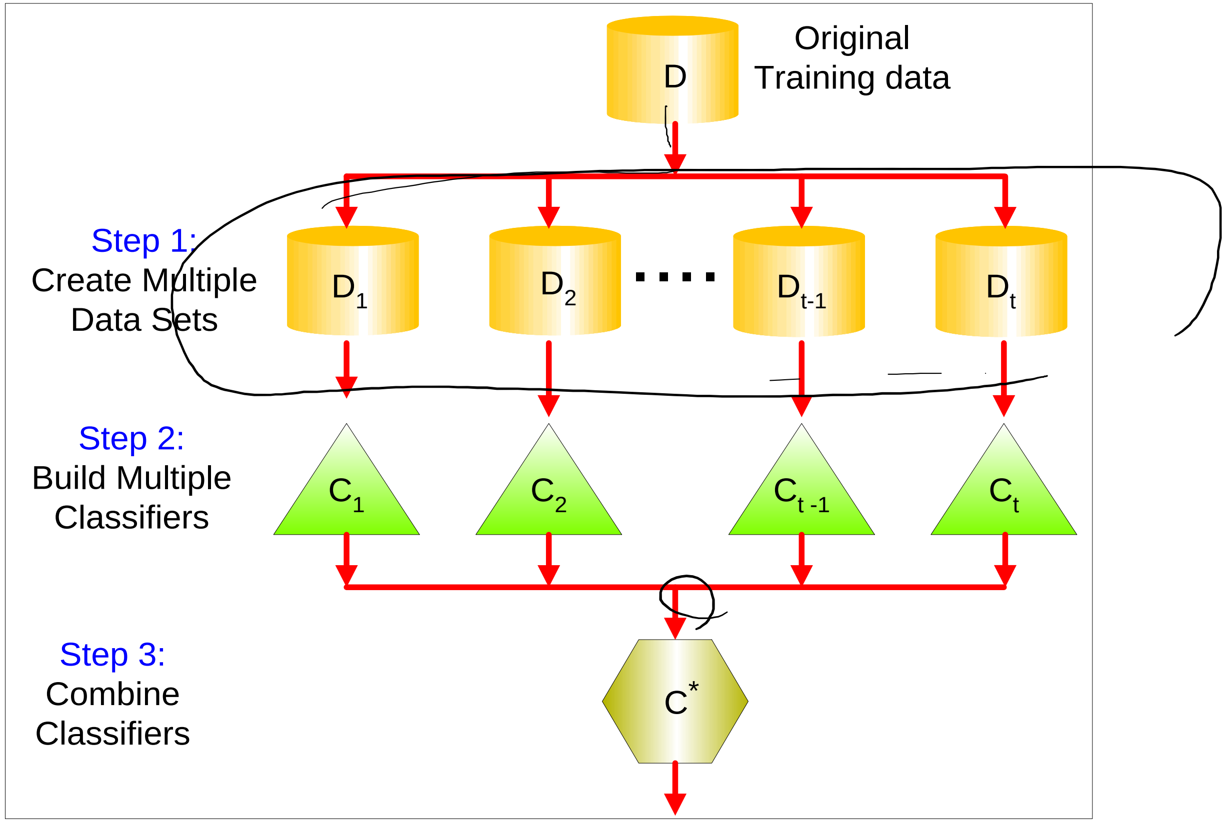
Ensemble Methods+Random Forest

Machine learning 2021

Mansoor Rezghi

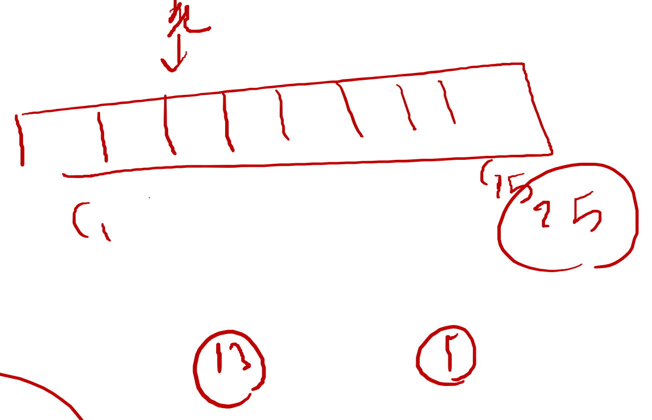
Ensemble Methods

- ✓ improving classification accuracy by aggregating the predictions of multiple classifiers
- ✓ Construct a set of base classifiers from the training data
- ✓ Predict class label of previously unseen records by aggregating predictions made by multiple classifiers



Tor.

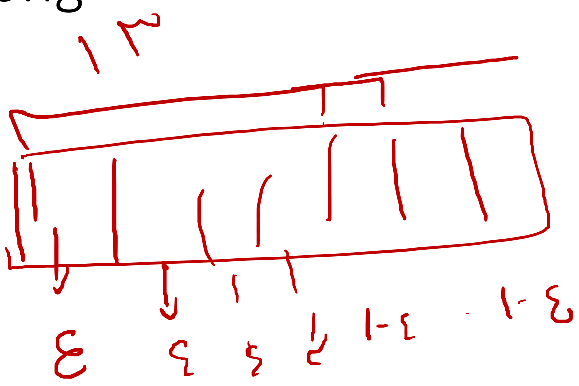
Ensemble Methods



- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$ /.65
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

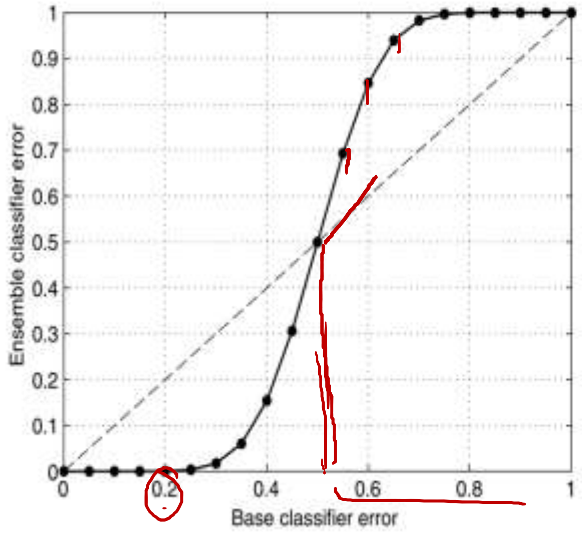
$$\sum_{i=13}^{25} \binom{25}{i} \boxed{\varepsilon^{i}(1-\varepsilon)^{25-i}} = 0.06$$

.94



V

base classifiers should be independent of each other
base classifiers should do better than a classifier that performs random guessing.



k

How to generate an ensemble of classifiers?

- Bagging: bootstrap aggregating
- Boosting

Bagging

- Sampling with replacement
- Build classifier on each bootstrap sample

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

Tr

C_1 4:1

C_2 6:1

D

D_1 $C_1 \leftarrow$ 6:1 4

D_2

D_3

$2 \leftarrow$ 9:1 3

Algorithm 5.6 Bagging algorithm.

1: Let k be the number of bootstrap samples.
2: **for** $i = 1$ to k **do**
3: Create a bootstrap sample of size N, D_i.
4: Train a base classifier C_i on the bootstrap sample D_i.
5: **end for**
6: $C^*(x) = \underset{y}{\operatorname{argmax}} \sum_i \delta(C_i(x) = y)$.
$\{\delta(\cdot) = 1$ if its argument is true and 0 otherwise$\}$.

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of boosting round

1. how the weights of the training examples are updated at the end of each boosting round
2. how the predictions made by each classifier are combined.

$N \quad w_j = \frac{1}{N}$

$X \to W = [\frac{1}{N}, \dots, \frac{1}{N}]$?

$C_1 \to \alpha_i \to \epsilon_i$

D, Evaluate

$D_1 \to C_1$ Train

Boosting: AdaBoost

Let $\{(\mathbf{x}j, yj) \mid j = 1, 2, \dots, N\}$ denote a set of N training examples.
Unlike bagging, importance of a base classifier Ci depends on its error rate

C_i

$$\alpha_i \longrightarrow \quad \epsilon_i = \frac{1}{N}\left[\sum_{j=1}^{N} w_j\, I\big(C_i(\mathbf{x}_j) \neq y_j\big)\right], \qquad \alpha_i = \frac{1}{2}\ln\left(\frac{1-\epsilon_i}{\epsilon_i}\right)$$

C_1

$(C_1(x_i) \neq y_i)$

weight assigned to example ($\mathbf{x}i$, yi) during the *jth* boosting round

$$e^{2\alpha_i} = \frac{1-\epsilon_i}{\epsilon_i}$$

$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \times \begin{cases} \exp^{-\alpha_j} & \text{if } C_j(\mathbf{x}_i) = y_i \\ \exp^{\alpha_j} & \text{if } C_j(\mathbf{x}_i) \neq y_i \end{cases}$$

ضعیف $C_1 \to \epsilon_i \to$ بزرگ α_i

Normalization factor

$$\epsilon_i(1+e^{2\alpha_i}) = 1 \to \epsilon_i = \frac{1}{1+e^{2\alpha_i}}$$

intermediate rounds produce an error rate higher than 50%, the weights wi =

قوی $C_1 \to \epsilon_i\ 1/N \to \alpha_i$ خیلی بزرگ

$\alpha_i \to$ بیشتر

$\alpha_1 \to \infty$

$\alpha_i \to -\infty$

C_{i+1}

D_{i+1}

D x_j

Boosting: AdaBoost

D_i

Let {($\mathbf{x}j$, yj) | j = 1, 2, . . . , N} denote a set of N training examples.
Unlike bagging, importance of a base classifier Ci depends on its error rate

$$\epsilon_i = \frac{1}{N}\left[\sum_{j=1}^{N} w_j \boxed{I\left(C_i(\mathbf{x}_j) \neq y_j\right)}\right], \qquad \boxed{\alpha_i} = \frac{1}{2}\ln\left(\frac{1-\epsilon_i}{\epsilon_i}\right)$$

$C_i \rightarrow \alpha_i$

$$w_j^{(i+1)} = \frac{w_j^{(i)}}{Z_i}\begin{cases} e^{-\alpha_i} & C_i(x_j) = y_j \\ e^{\alpha_i} & C_i(x_j) \neq y_j \end{cases}$$

weight assigned to example ($\mathbf{x}i$, yi)
during the jth boosting round

$$w_j^{(j+1)} = \frac{w_j^{(j)}}{Z_j} \times \begin{cases} \exp^{-\alpha_j} & \text{if } C_j(\mathbf{x}_j) = y_j \\ \exp^{\alpha_j} & \text{if } C_j(\mathbf{x}_j) \neq y_j \end{cases}$$

Normalization factor

intermediate rounds produce an error rate higher than 50%, the weights wi = 1/N

$C_i \rightarrow \alpha_i$

$C_i(x_j) = y_j$ $e^{-\alpha_i}$

$C_i(x_j) \neq y_j$ e^{α_i}

AdaBoost

Algorithm 5.7 AdaBoost algorithm.

1: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Initialize the weights for all N examples.}
2: Let k be the number of boosting rounds.
3: **for** $i = 1$ to k **do**
4: Create training set D_i by sampling (with replacement) from D according to $\mathbf{w}$.
5: Train a base classifier C_i on D_i.
6: Apply C_i to all examples in the original training set, D.
7: $\epsilon_i = \frac{1}{N}\left[\sum_j w_j\ \delta\big(C_i(x_j) \neq y_j\big)\right]$ {Calculate the weighted error.}
8: **if** $\epsilon_i > 0.5$ **then**
9: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Reset the weights for all N examples.}
10: Go back to Step 4.
11: **end if**
12: $\alpha_i = \frac{1}{2} \ln \frac{1-\epsilon_i}{\epsilon_i}$.
13: Update the weight of each example according to Equation 5.69.
14: **end for**
15: $C^*(\mathbf{x}) = \underset{y}{\operatorname{argmax}} \sum_{j=1}^{T} \alpha_j \delta\big(C_j(\mathbf{x}) = y\big)$.

Example

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	−1	−1	−1	−1	1	1	1

Boosting Round 1:

x	0.1	0.4	0.5	0.6	0.6	0.7	0.7	0.7	0.8	1
y	1	-1	-1	-1	-1	-1	-1	-1	1	1

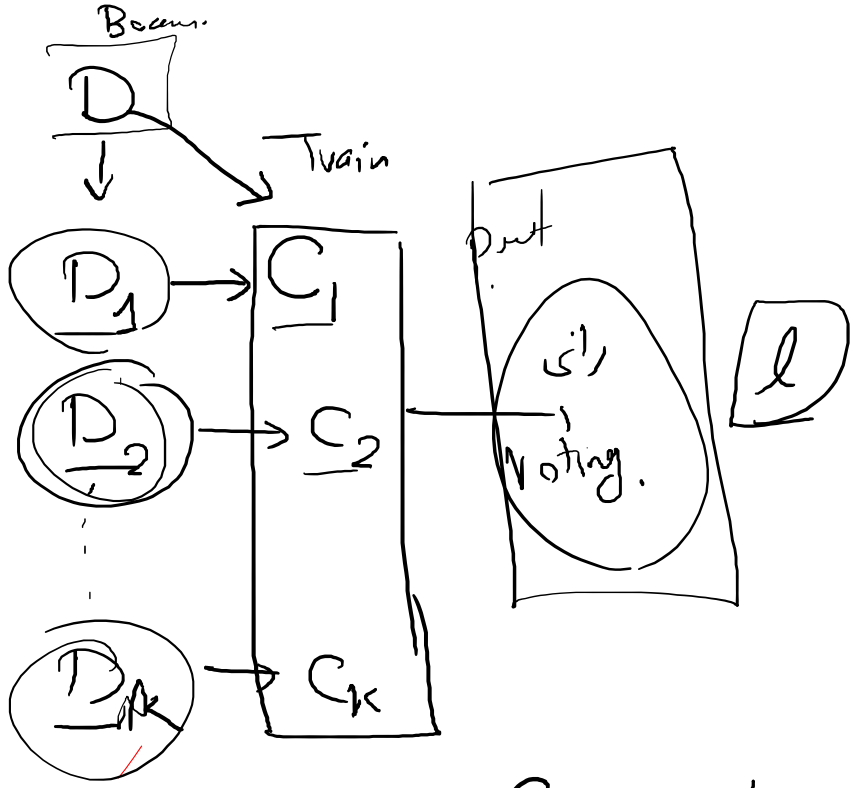
Boosting Round 2:

x	0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3
y	1	1	1	1	1	1	1	1	1	1

Boosting Round 3:

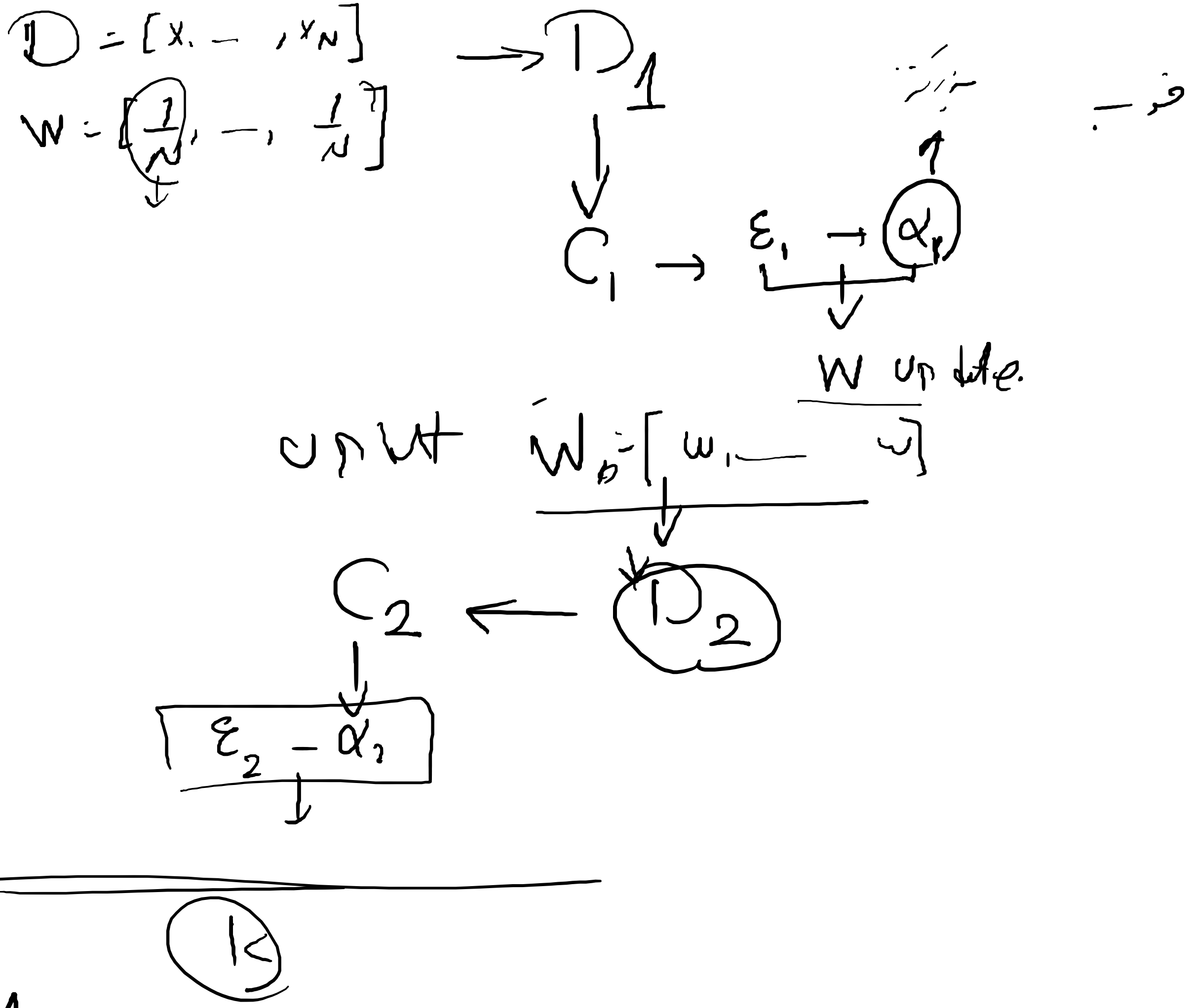
x	0.2	0.2	0.4	0.4	0.4	0.4	0.5	0.6	0.6	0.7
y	1	1	-1	-1	-1	-1	-1	-1	-1	-1

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	0.311	0.311	0.311	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.029	0.029	0.029	0.228	0.228	0.228	0.228	0.009	0.009	0.009



$C_1 — \alpha_1$

$C_n = \alpha_n$



Random Forest

- ✓ ensemble methods specifically designed for decision tree classifiers
- ✓ multiple decision trees where each tree is generated based on the values of an independent set of random vectors

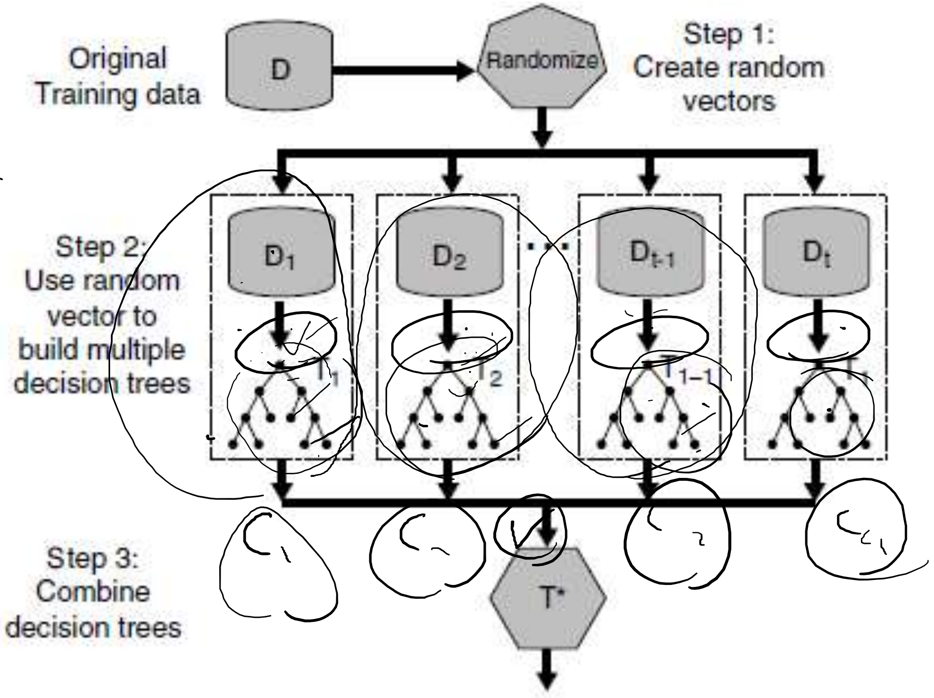
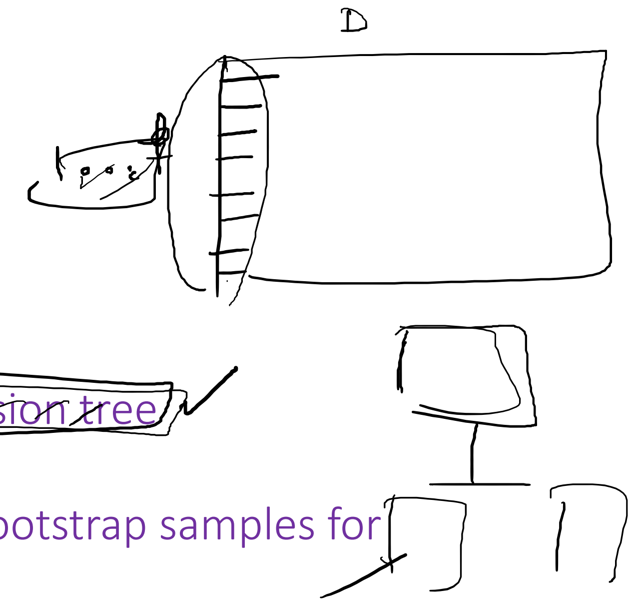
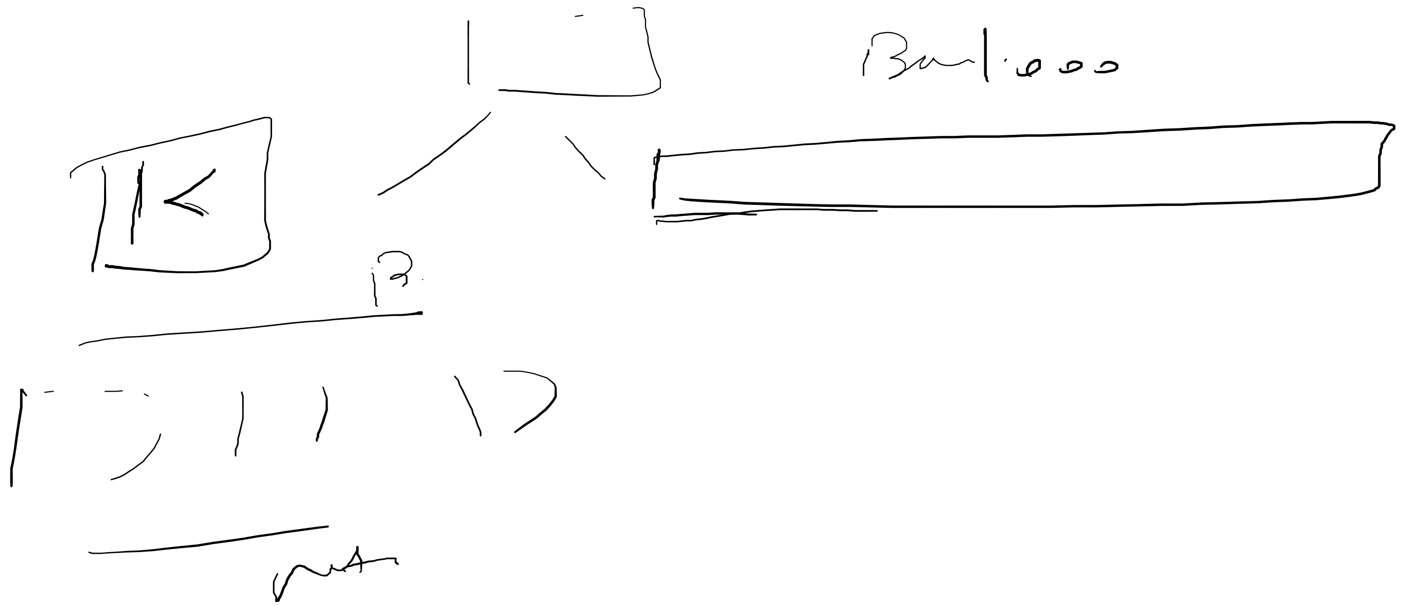


Figure 5.40. Random forests.

Random Forest

- ✓ Bagging using decision trees is a special case of random forests
- ✓ randomly select *F* input features to split at each node of the decision tree
- ✓ majority voting scheme
- ✓ To increase randomness, bagging can also be used to generate bootstrap samples for Forest-RI





Metrics for class imbalance problem

Imbalance

- ✓ Data sets with imbalanced class distributions
- ✓ in credit card fraud detection, fraudulent transactions are outnumbered by legitimate transactions
- ✓ accuracy measure, used extensively for classifiers, may not be well suited for evaluating models derived from imbalanced data sets

example : 1% of the credit card transactions fraudulent,
a model that predicts every transaction as legitimate
accuracy 99%
it fails to detect any of the fraudulent activities.

binary classification, the rare class is often denoted as the positive class against negative class

		Predicted Class	
		+	−
Actual Class	+	f_{++} (TP)	f_{+-} (FN)
	−	f_{-+} (FP)	f_{--} (TN)

confusion matrix

Imbalance

Precision : fraction of records that actually turns out to be positive in the group the classifier has declared as a positive class

$$\boxed{\text{Precision, } p = \frac{TP}{TP + FP}}$$

Recall measures the fraction of positive examples correctly predicted by the classifier

$$\boxed{\text{Recall, } r = \frac{TP}{TP + FN}}$$

maximizes both precision and recall

Imbalance

Precision and recall can be summarized into another metric known as the F1 measure

$$\boxed{F_1 = \frac{2}{\frac{1}{r} + \frac{1}{p}}.}$$

tends to be closer to the smaller of the two numbers
a high value of *F*1-measure ensures that both precision and recall are reasonably high

$$\boxed{\text{Weighted accuracy} = \frac{w_1 TP + w_4 TN}{w_1 TP + w_2 FP + w_3 FN + w_4 TN}.}$$