Decision Trees, Random Forest and Ensemble Methods

WQD 7006

Decision trees

- Non-linear classifier - Target function is discrete valued
- Easy to use
- Easy to interpret
- Susceptible to overfitting but can be avoided.

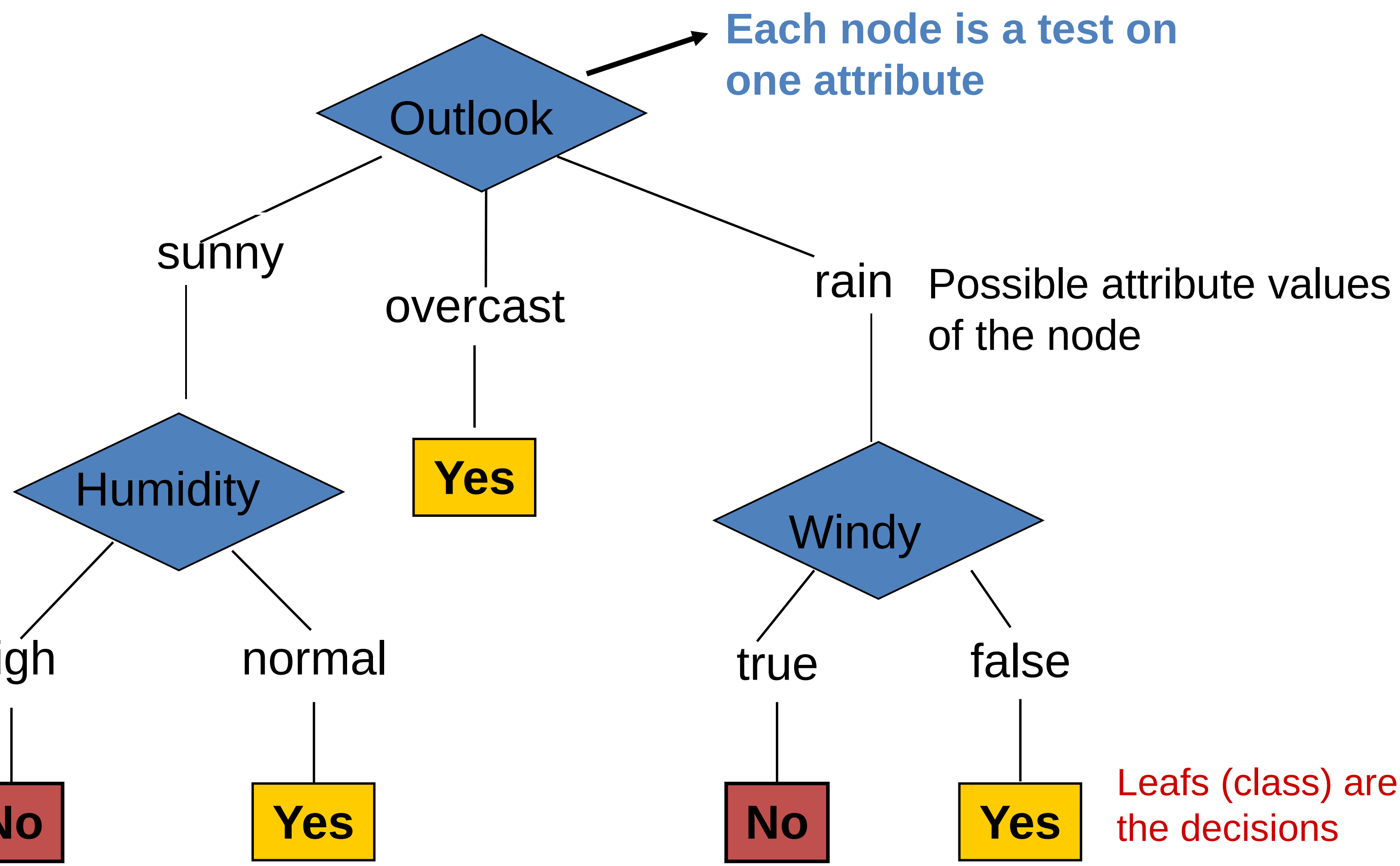
Examples:

Medical diagnosis

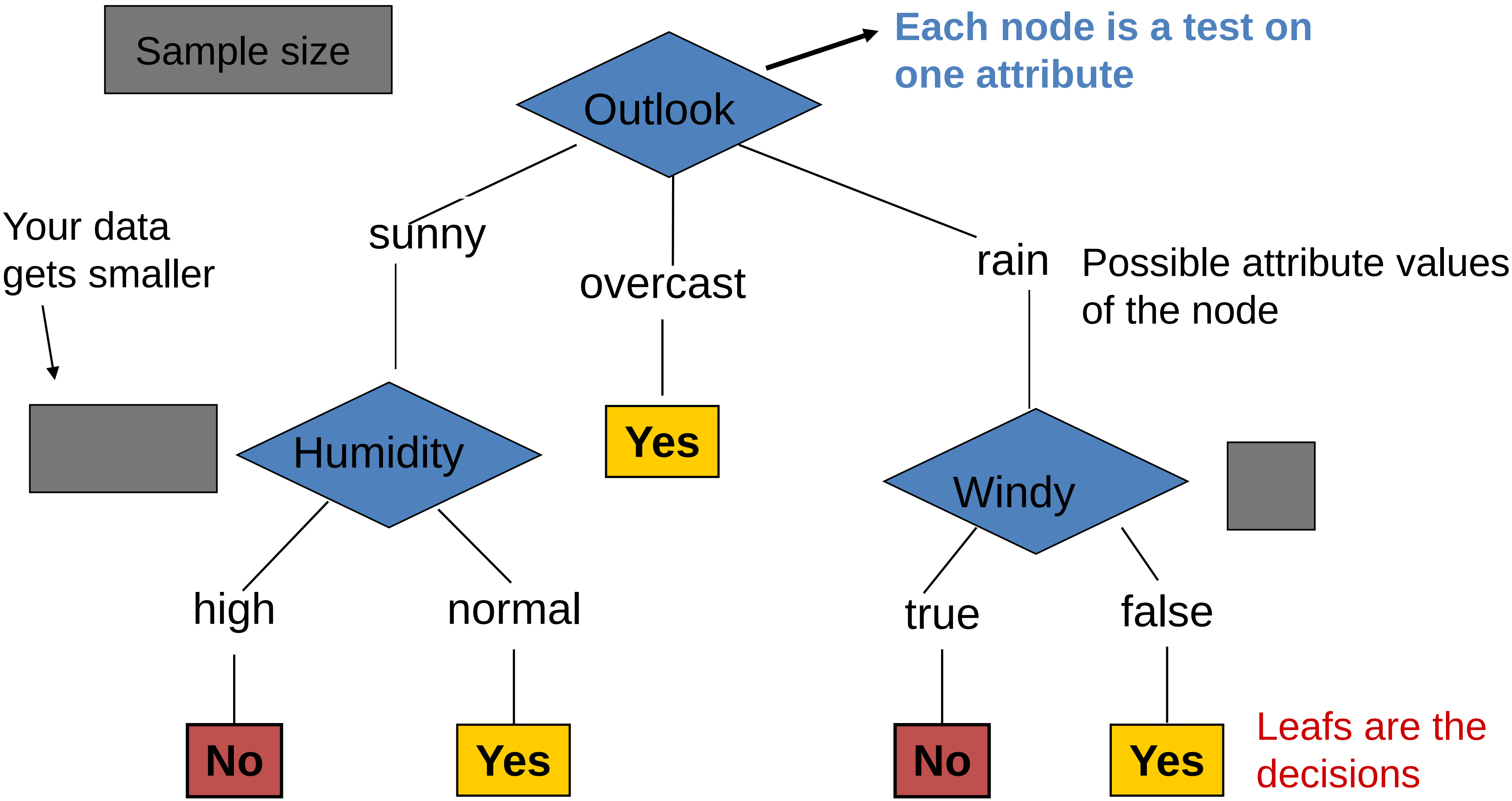
Credit risk analysis

Natural Language Processing

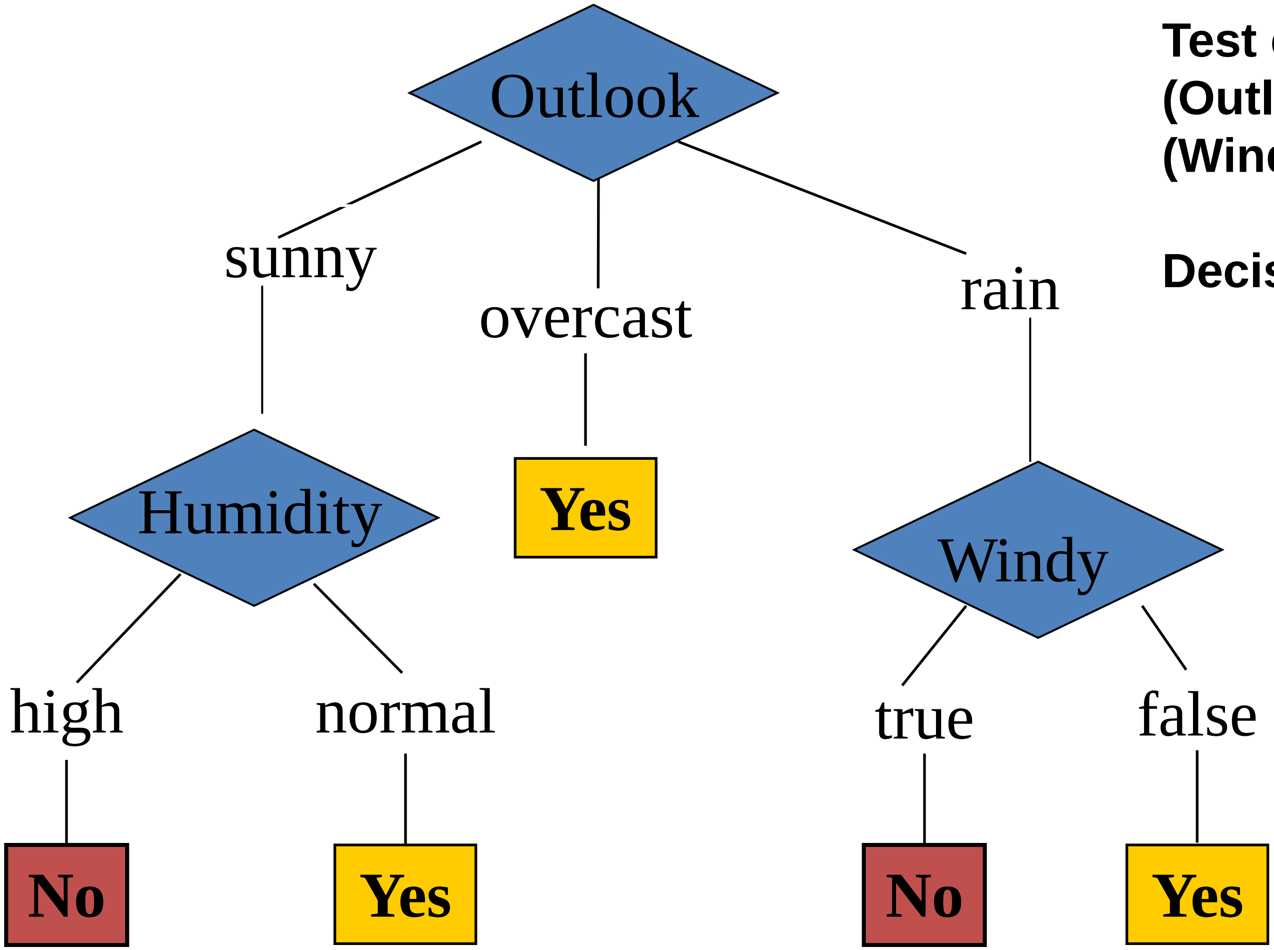
Anatomy of a decision tree



Anatomy of a decision tree



To ‘play tennis’ or not.

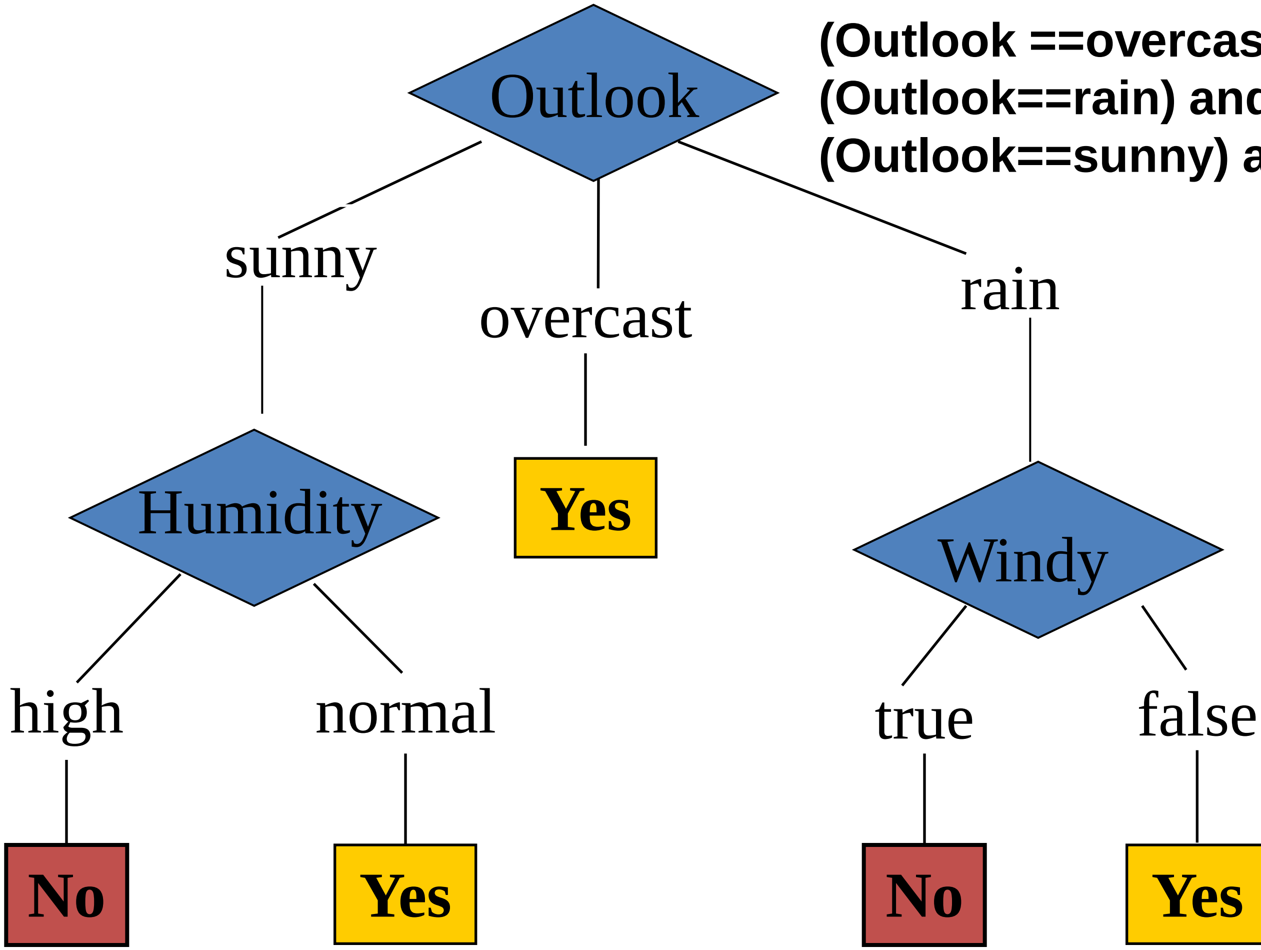


Test example:
(Outlook==rain) and
(Windy==false)

Decision is yes.

To ‘play tennis’ or not.

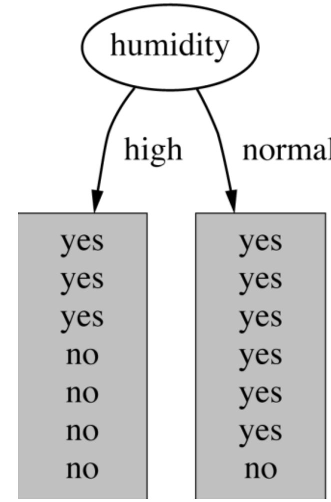
(Outlook ==overcast) -> yes
(Outlook==rain) and (not Windy==True) ->yes
(Outlook==sunny) and (Humidity=normal) ->yes



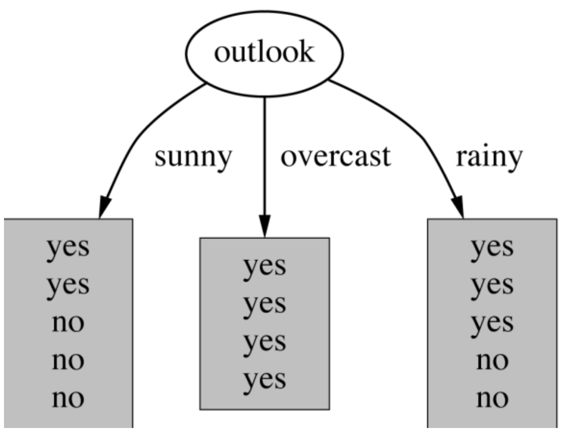
Day	Outlook	Temp.	Humidity	Wind	Play Tennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

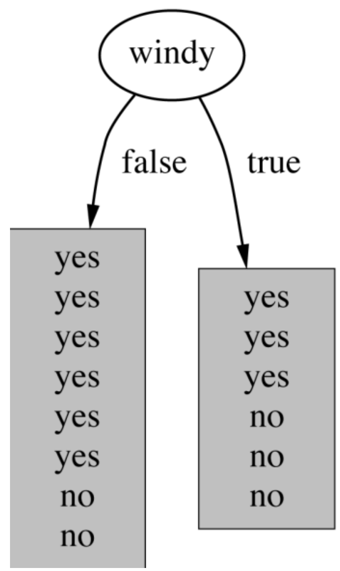
How do we choose the test ?

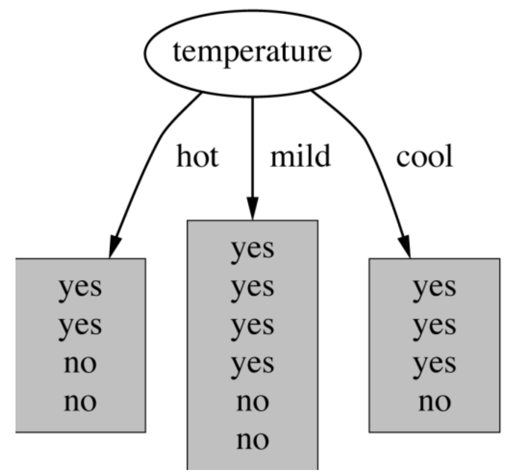
Which attribute should be used as the test?



Intuitively, you would prefer the one that *separates* the training examples as much as possible.







Im(purity) Measures

- Gini
- **Information gain/entropy (HOMEWORK)**
- Chi-square test

Entropy

Entropy measures the amount of **uncertainty** in a probability distribution

$$H(X) = E(I(X)) = \sum_i p(x_i) I(x_i) = -\sum_i p(x_i) \log_2 p(x_i)$$

Entropy

Play Tennis dataset - two target classes: *yes* and *no*

Out of 14 instances, 9 classified *yes*, rest *no*

$$p_{yes} = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) = 0.41$$

$$p_{no} = -\left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) = 0.53$$

$$E(S) = p_{yes} + p_{no} = 0.94$$

Information Gain

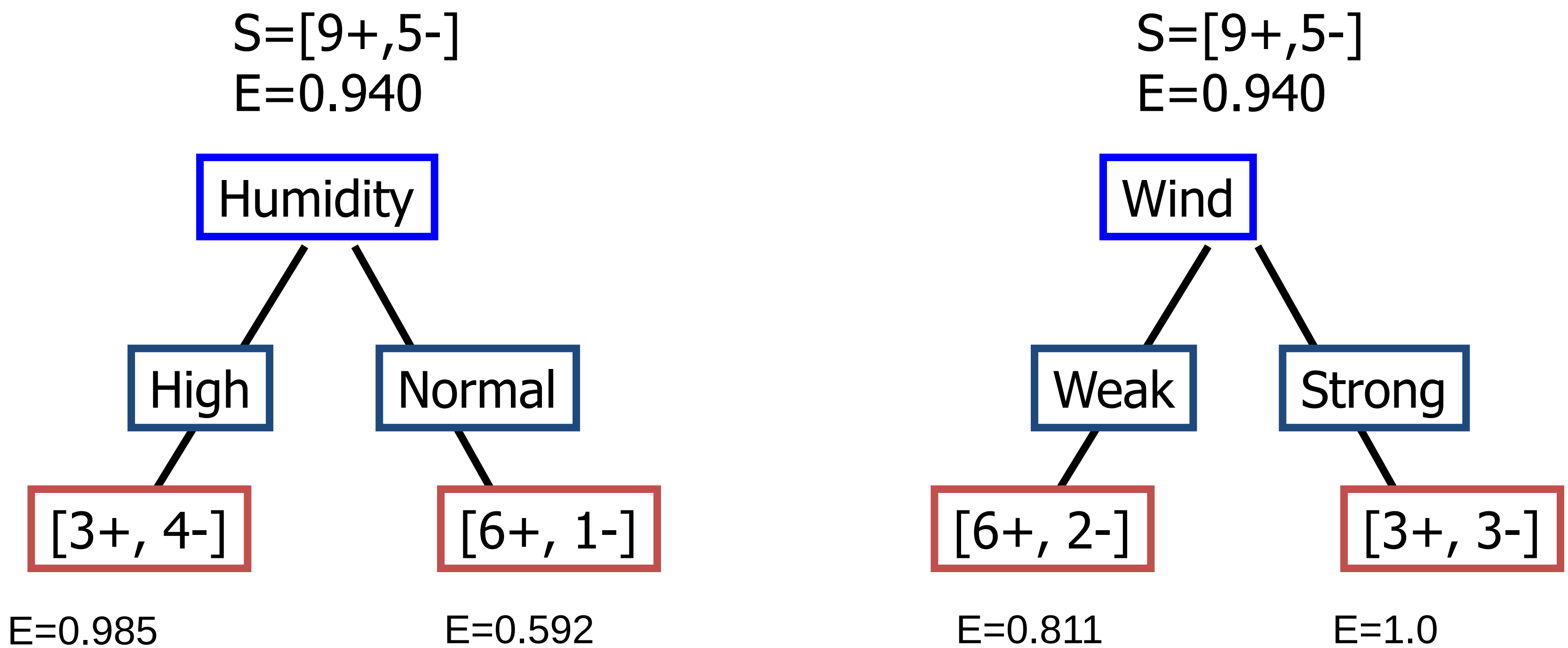
The expected **reduction in entropy** caused by partitioning the instances from S according to a given discrete variable.

$$Gain(S, X_i) = E(S) - \sum_j \frac{|S_{x_{ij}}|}{|S|} E(S_{x_{ij}})$$

where $S_{x_{ij}}$ is the subset of instances from S s.t. $X_i = x_{ij}$.

Information gain =
(information before split) – (information after split)

Selecting the Next Attribute

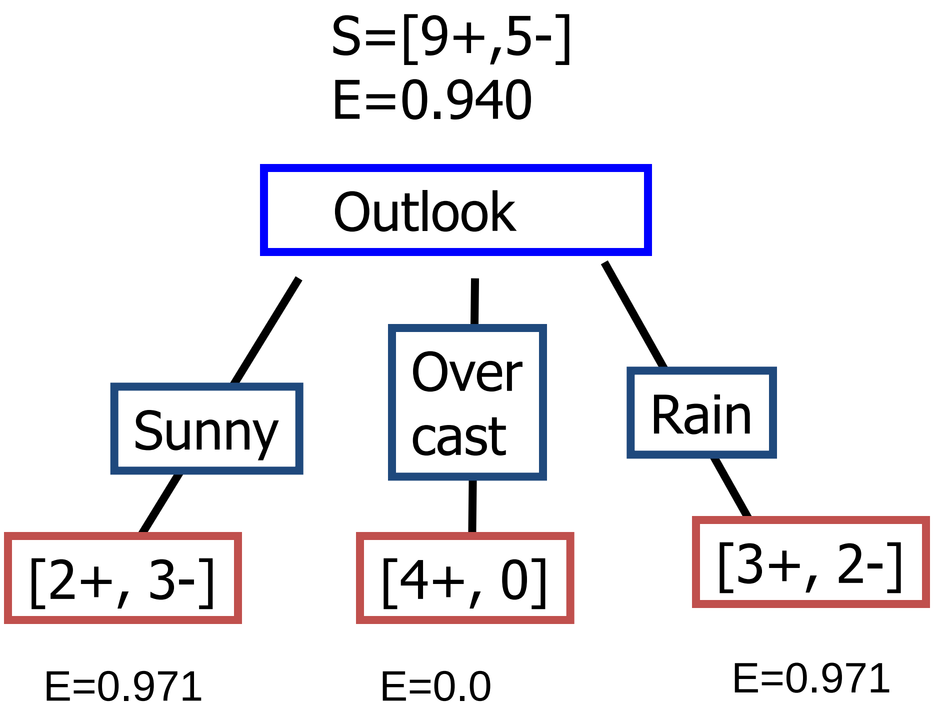


Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

Humidity provides greater info. gain than Wind = target classification.

Selecting the Next Attribute



Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.0971
=0.247

Overfitting

- You can perfectly fit to any training data
- Zero bias, high variance

Two approaches:

1. **Pre-pruning** - Stop growing the tree when further splitting the data does not yield an improvement
2. Grow a full tree, then prune the tree, by eliminating nodes **(post-prune)**

Decision tree

- Not so ideal for predicting - **inaccuracy**

  - not so flexible in classifying based on new dataset

- **Instability** – a change in data can change the look of a tree

Random Forest

- Create a bootstrapped dataset
- Build Random Forest
- Evaluate
- Repeat Step 2 with more features

Bagging

- Bagging or ***Bootstrap aggregating*** **-** a technique for reducing the variance of an estimated prediction function.
  - the variance in the prediction is reduced (no big loss from the random errors that a single classifier is bound to make).

- Bagging:
  - Sample M bootstrap samples.
  - Train M different classifiers on these bootstrap samples
  - For a new query, let all classifiers predict and take an average (or majority vote)

Bagging

A bootstrap sample is chosen at random *with* replacement from the data. Some observations end up in the bootstrap sample more than once, while others are not included ("***out of bag***").

Bagging reduces the ***variance*** of the base learner but has limited effect on the ***bias*****.**

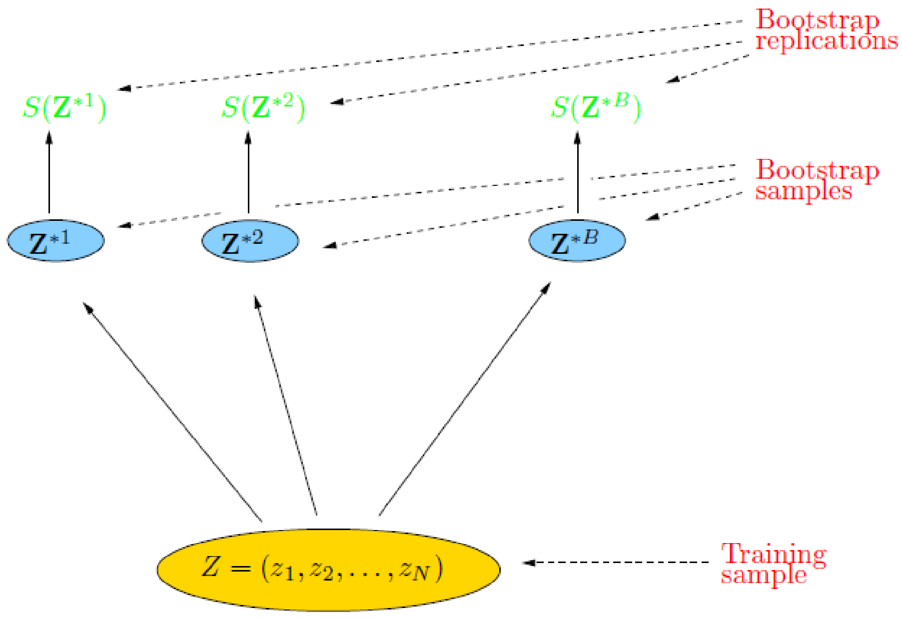
•It's most effective if we use *strong* base learners that have very little bias but high variance (unstable). E.g. trees.

•Both **bagging** and **boosting** are examples of "ensemble learners" that were popular in machine learning in the '90s.

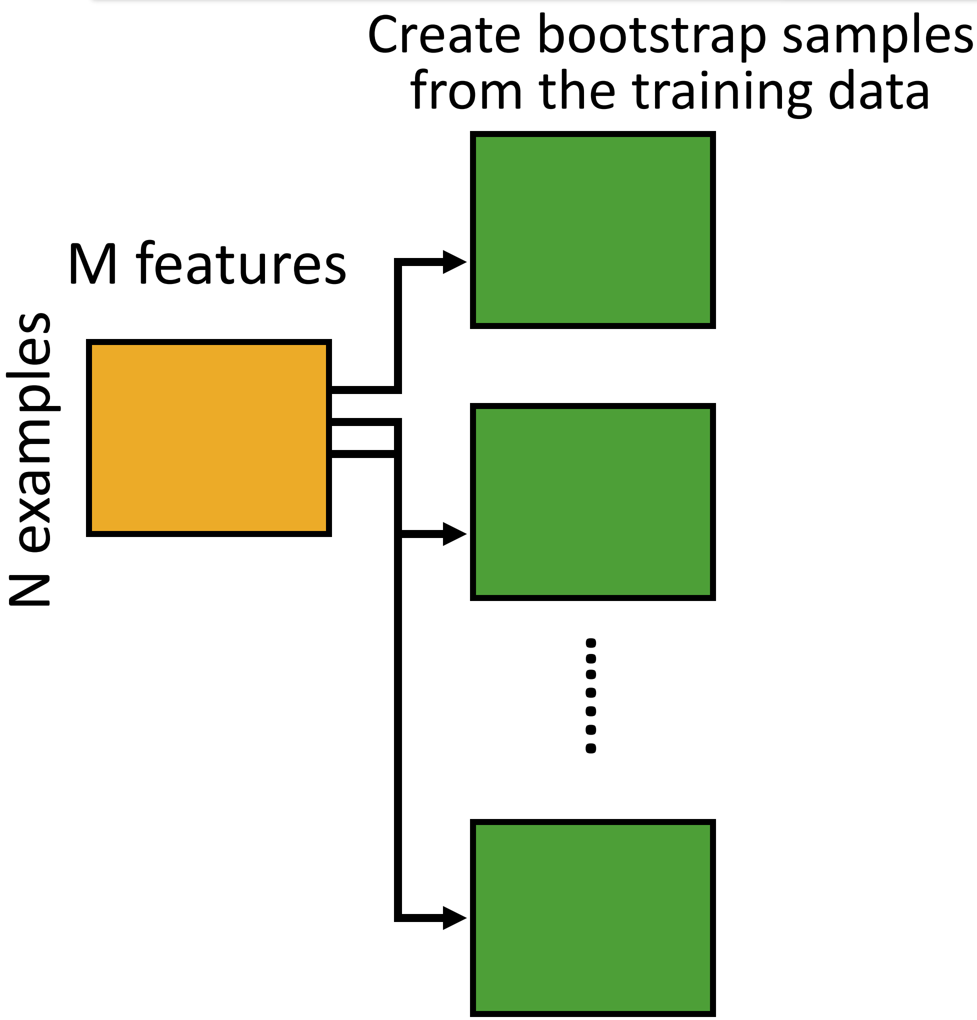
Bootstrap

The basic idea:

randomly draw datasets *with replacement* from the training data, each sample *the same size as the original training set*

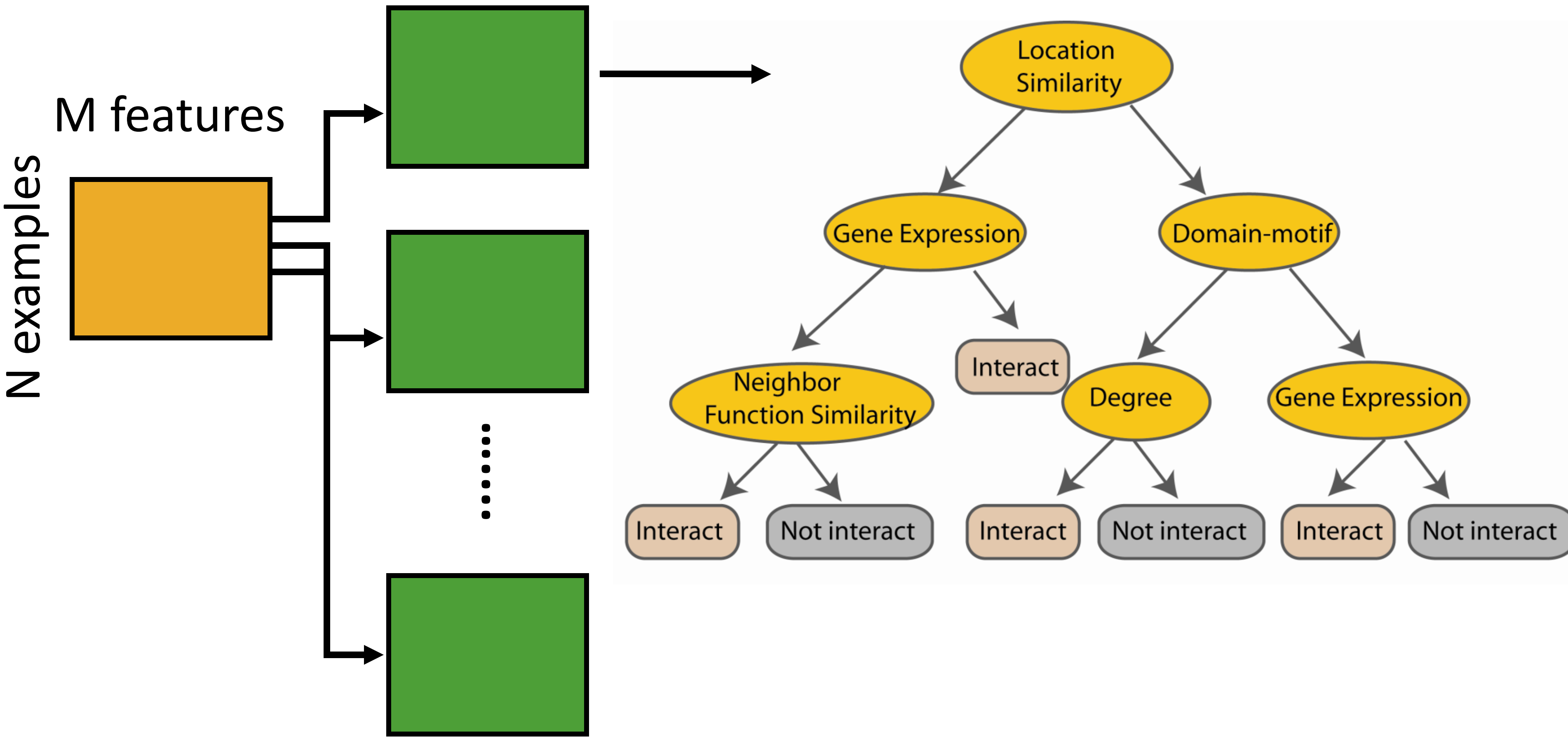


Bagging


Create bootstrap samples
from the training data
M features
N examples

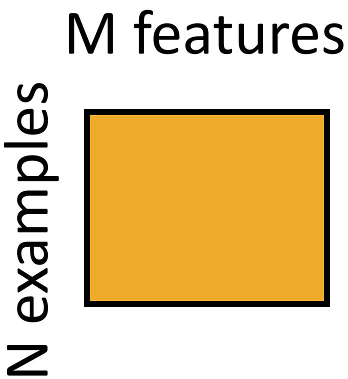
Random Forest Classifier

Construct a decision tree



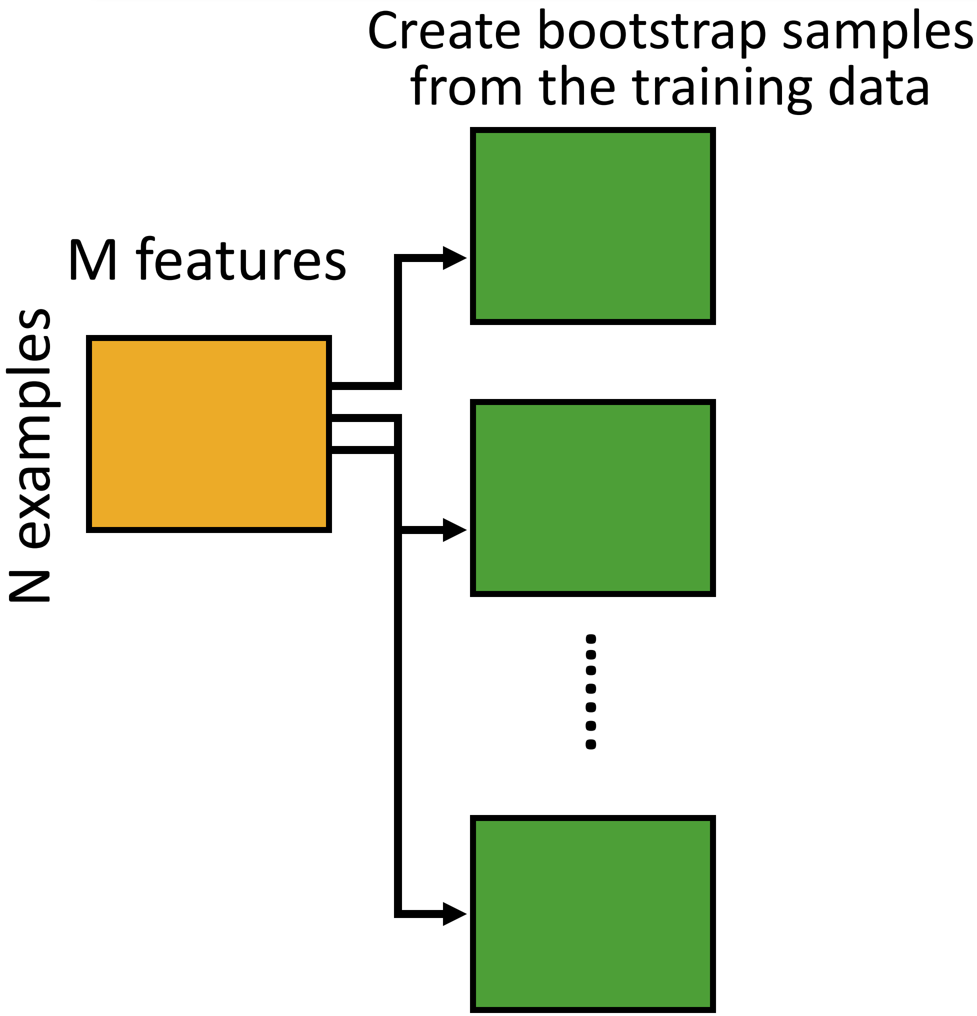
Random Forest Classifier

Training Data



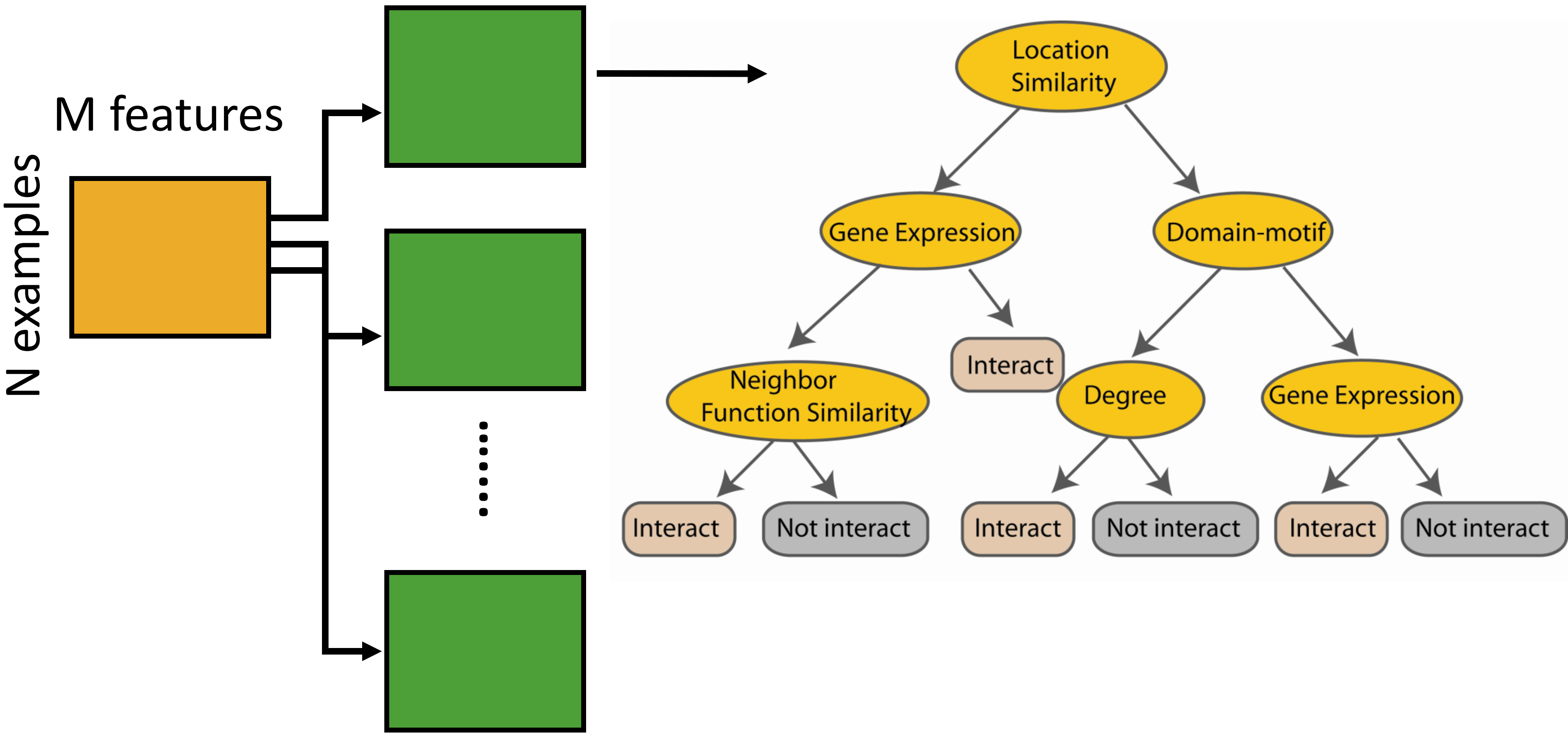
Random Forest is an ensemble classifier. Briefly, given N data and M features, bootstrap samples are created from the training data

Random Forest Classifier



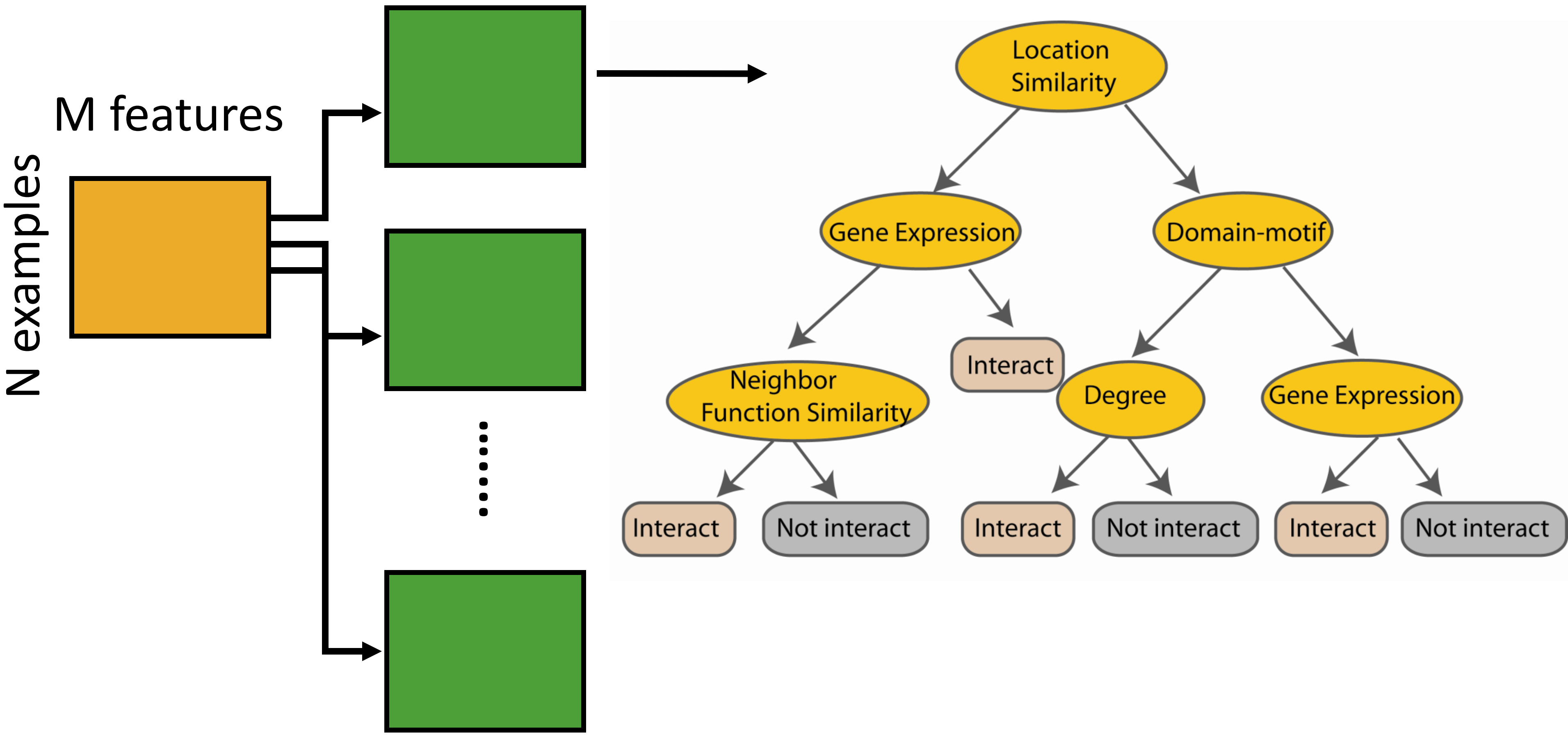
Random Forest Classifier

Construct a decision tree



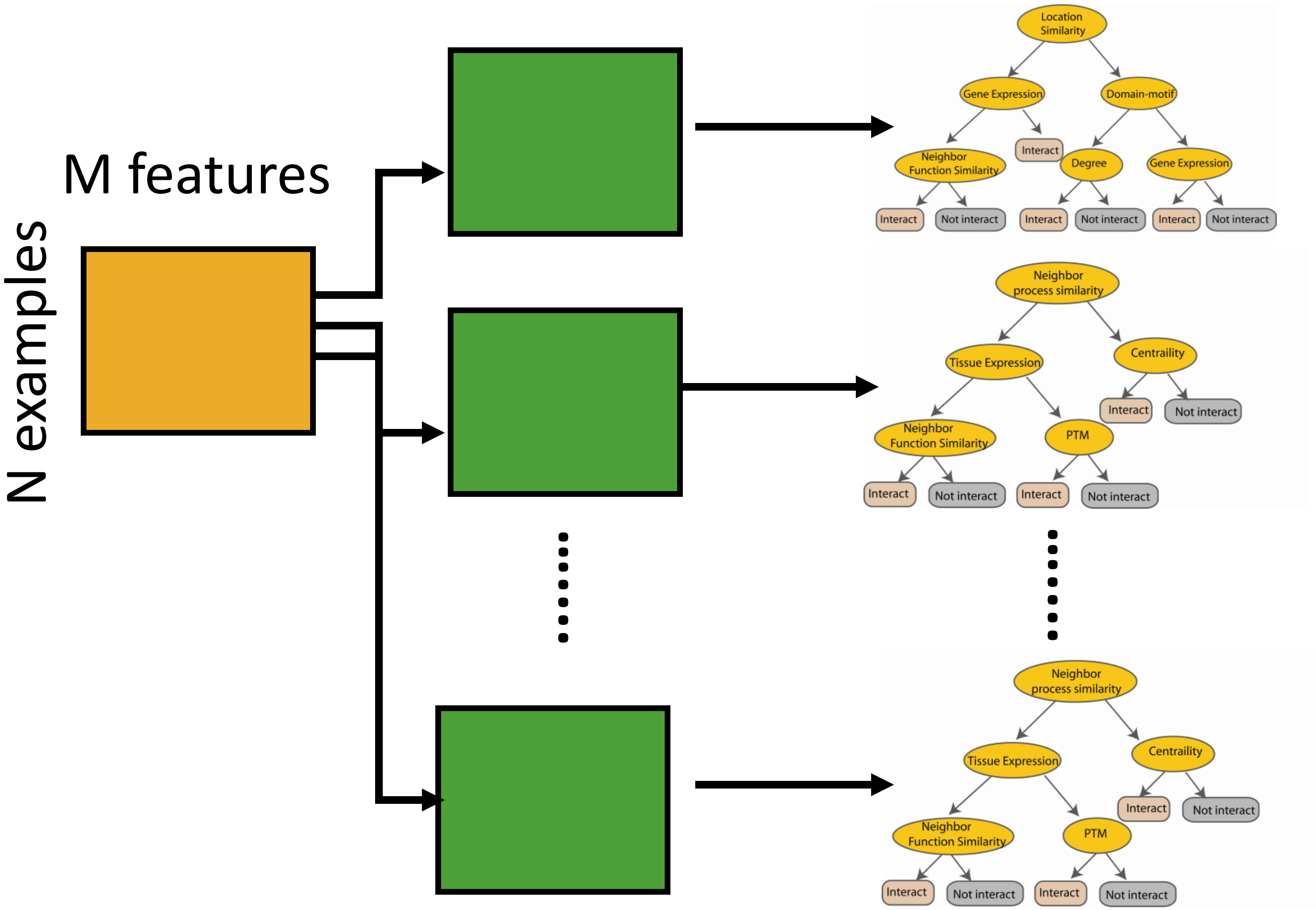
Random Forest Classifier

At each node in choosing the split feature choose only among *m*<*M* features

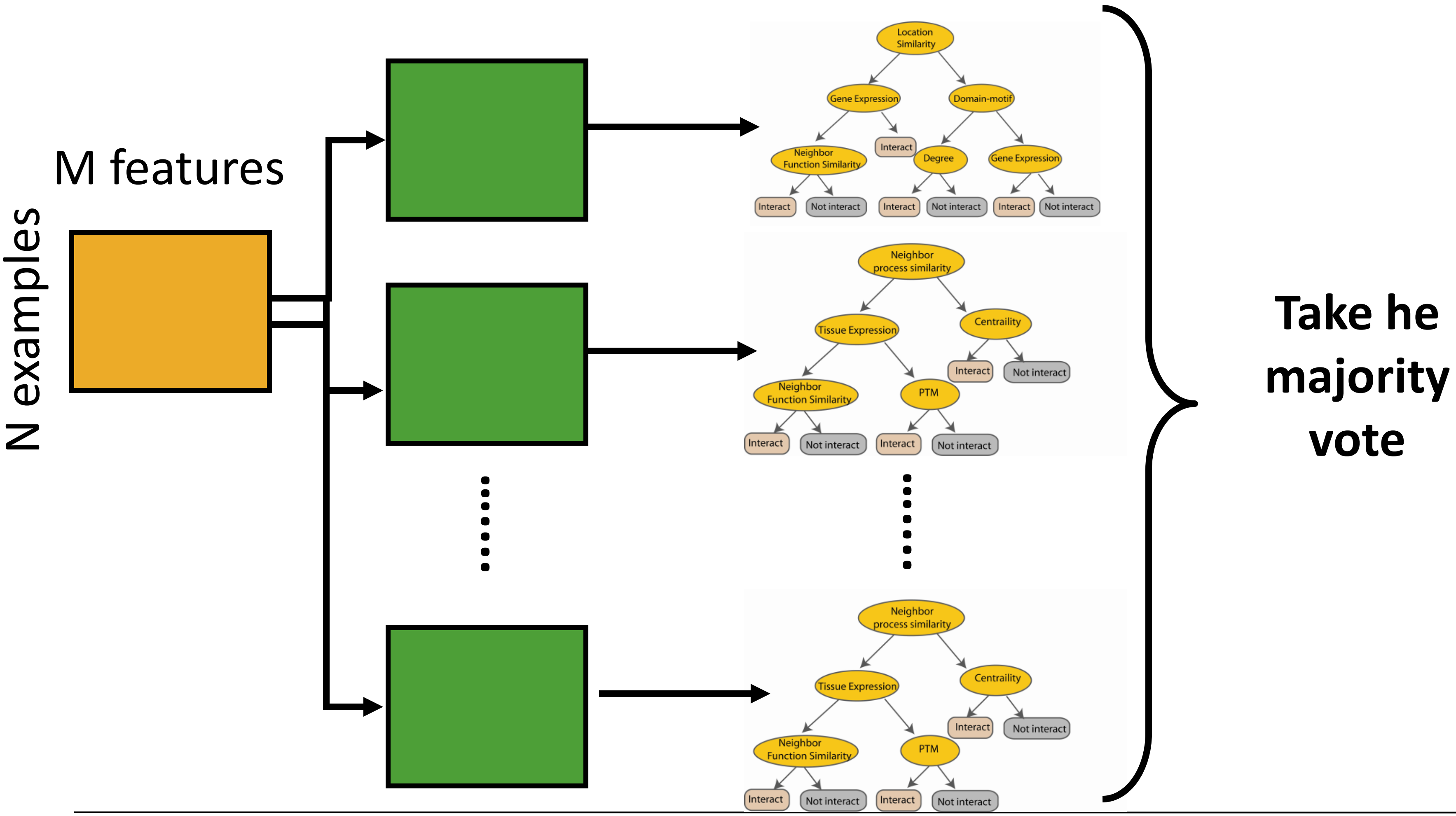


Random Forest Classifier

Create decision tree from each bootstrap sample



Random Forest Classifier



Additional Info - Boosting

Main idea:

- train classifiers (e.g. decision trees) in a sequence.
- a new classifier should focus on those cases which were incorrectly classified in the last round (get higher weights)
- Each boosting round learns a new (simple) classifier on the weighed dataset.
- These classifiers are weighed to combine them into a single powerful classifier.
- Combine the classifiers by letting them vote on the final prediction (like bagging).
- Stop by using monitoring a hold out set (cross-validation).

Boosting in a Picture