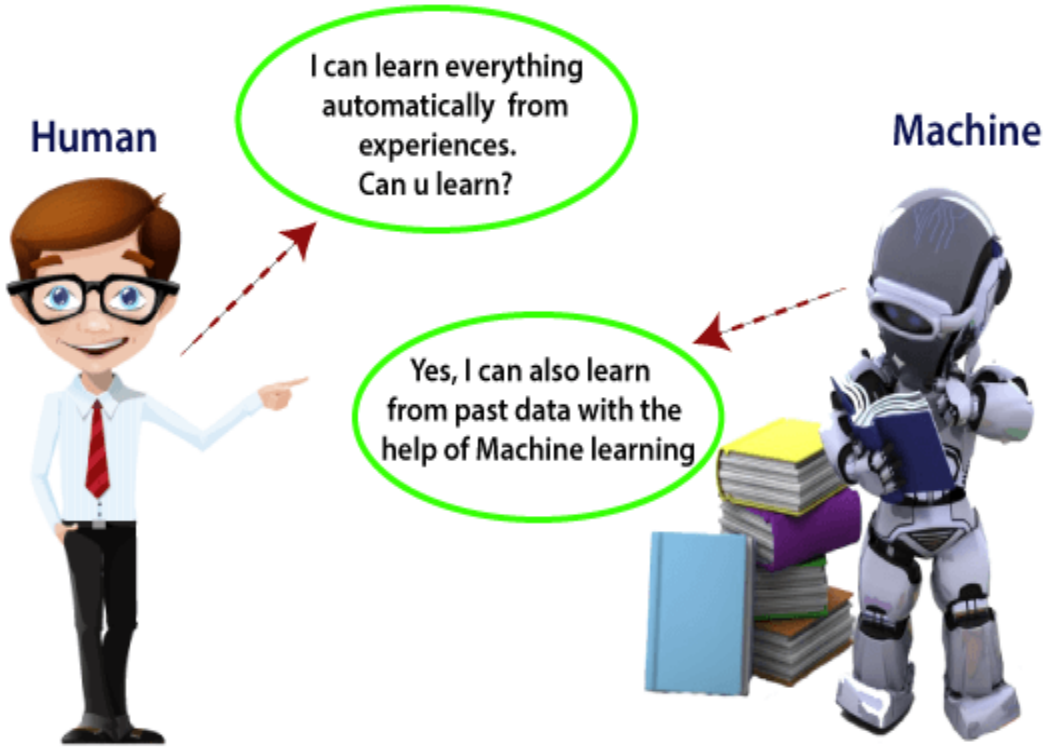
Module 5

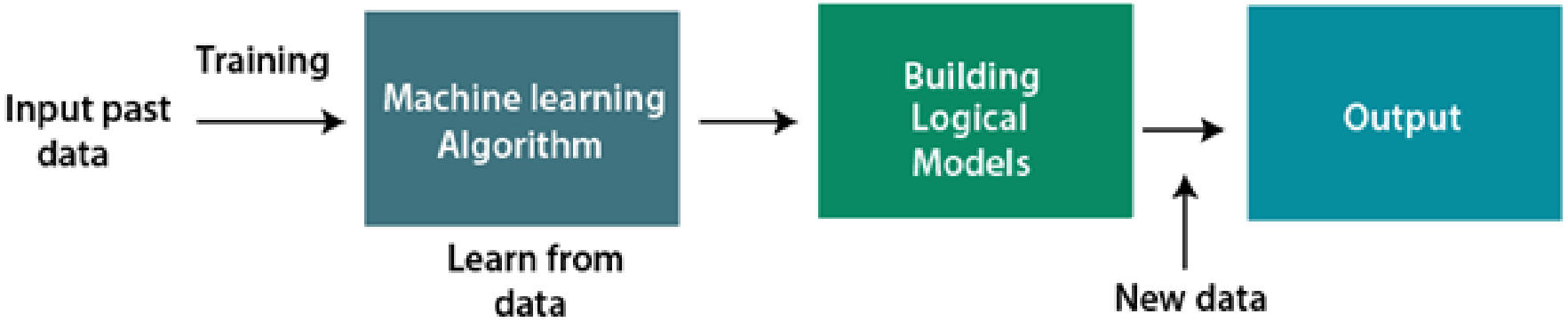
Module-5 **(Machine Learning)**

- Learning from Examples
- Forms of Learning,
- Supervised Learning,
- Learning Decision Trees,
- Evaluating and choosing the best hypothesis,
- Regression and classification with Linear models.

Machine Learning



- Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.



- A Machine Learning system **learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it.**

Classification of Machine Learning

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

Supervised learning

- We provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.
- The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done
- Then we test the model by providing a sample data to check whether it is predicting the exact output or not.

- The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as **when a student learns things in the supervision of the teacher**.
- **TYPES**
  - o Classification
  - o Regression

Unsupervised Learning

- Unsupervised learning is a learning method in which a machine learns without any supervision.
- The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision.
- The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.
- In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

- **TYPES**
  - o Clustering
  - o Association

Reinforcement Learning

- Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
- The agent learns automatically with these feedbacks and improves its performance.
- In reinforcement learning, the agent interacts with the environment and explores it.
- The goal of an agent is to get the most reward points, and hence, it improves its performance.

- The robotic dog, which automatically learns the movement of his arms, is an example of Reinforcement learning.

Supervised vs. Unsupervised Learning

- Supervised learning (classification)
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set
- Unsupervised learning (clustering)
  - The class labels of training data is unknown
  - Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data

Decision tree induction

Ross Quinlan developed a decision tree algorithm known as ID3(iterative dichotomiser)

Later he presented C4.5(successor of ID3)

L.Breiman,J.Friedman,R.Olshen)-Another algorithm CART(classification and regression tree)

ID3,C4.5 and CART adopt a greedy(no backtracking)approach in which decision trees are constructed in top-down, recursive divide and conquer manner

Basic Algorithm for Decision Tree Induction

- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - Samples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

Classification by Decision Tree Induction

- Decision tree
  - A flow-chart-like tree structure
  - Internal node denotes a test on an attribute
  - Branch represents an outcome of the test
  - Leaf nodes represent class labels or class distribution
  - Root node-top most node
  - Decision tree induction is the learning of decision tree from class labeled training tuples

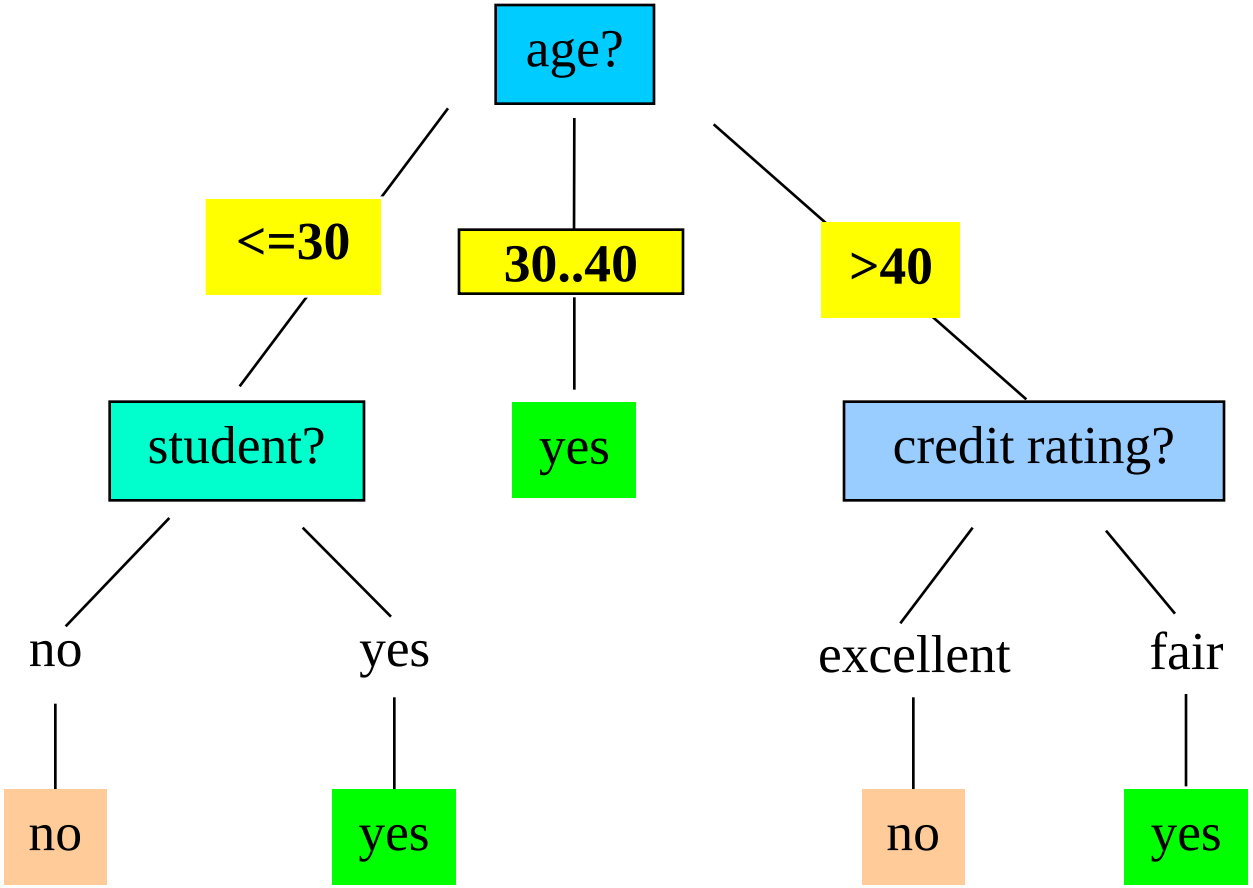
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Training Dataset

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Example: A Decision Tree for "buys_computer"



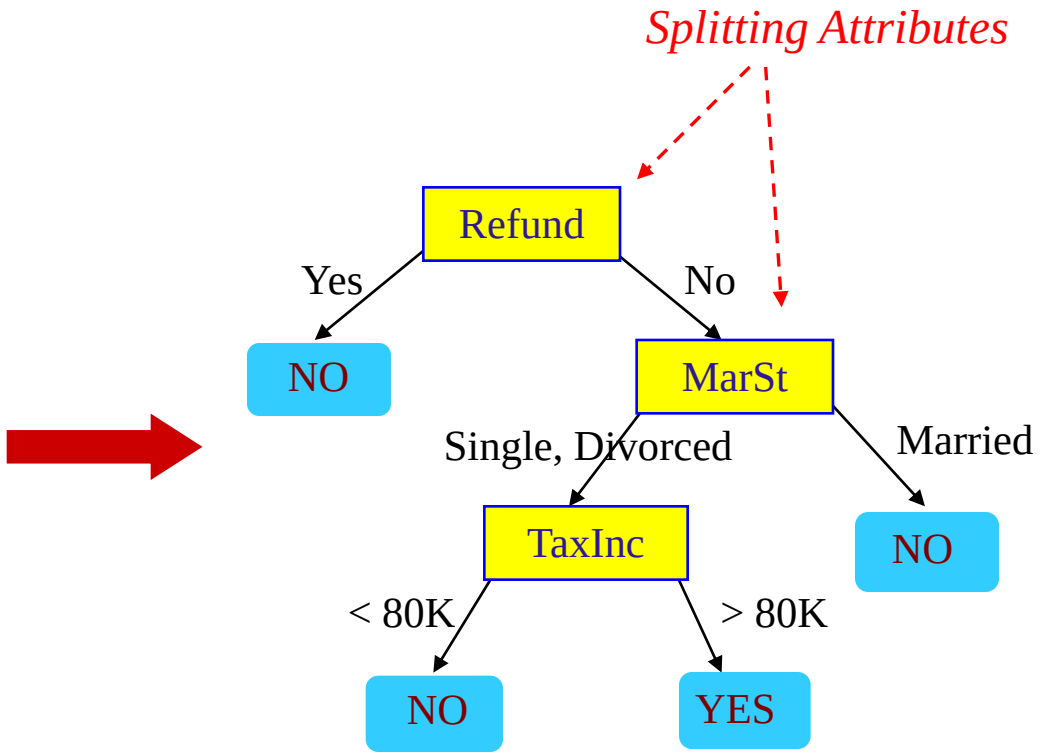
Non-leaf nodes – test on an attribute

Leaf nodes – class (buys_computer)

Example of a Decision Tree

Tid	Refund (categorical)	Marital Status (categorical)	Taxable Income (continuous)	Cheat (class)
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

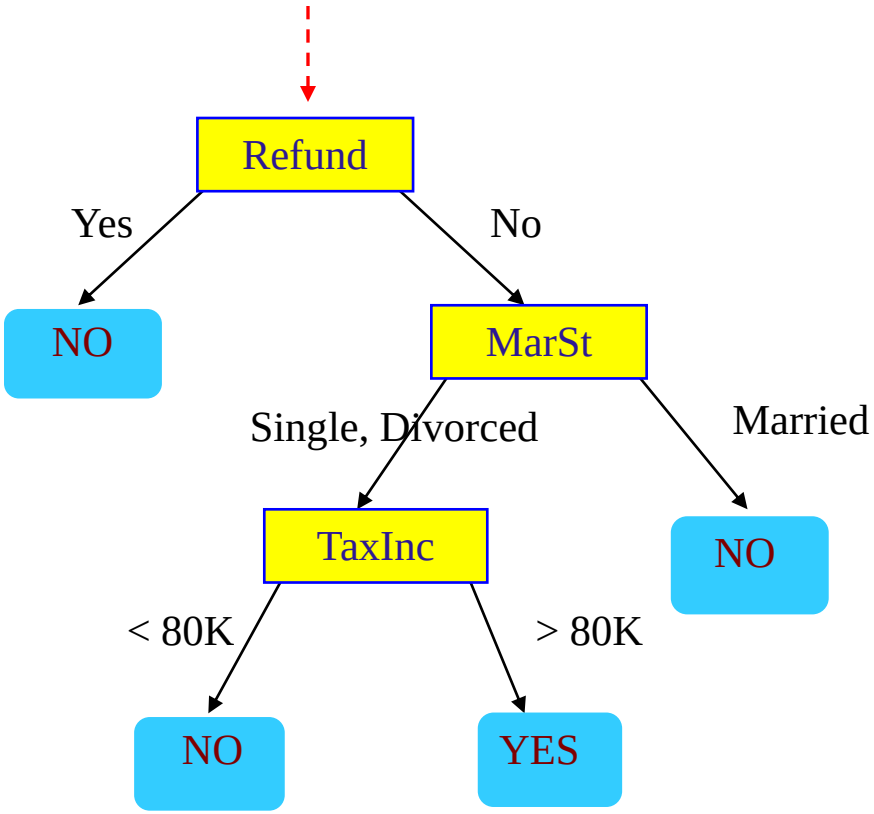
Training Data



Model: Decision Tree

Apply Model to Test Data

Start from the root of tree.



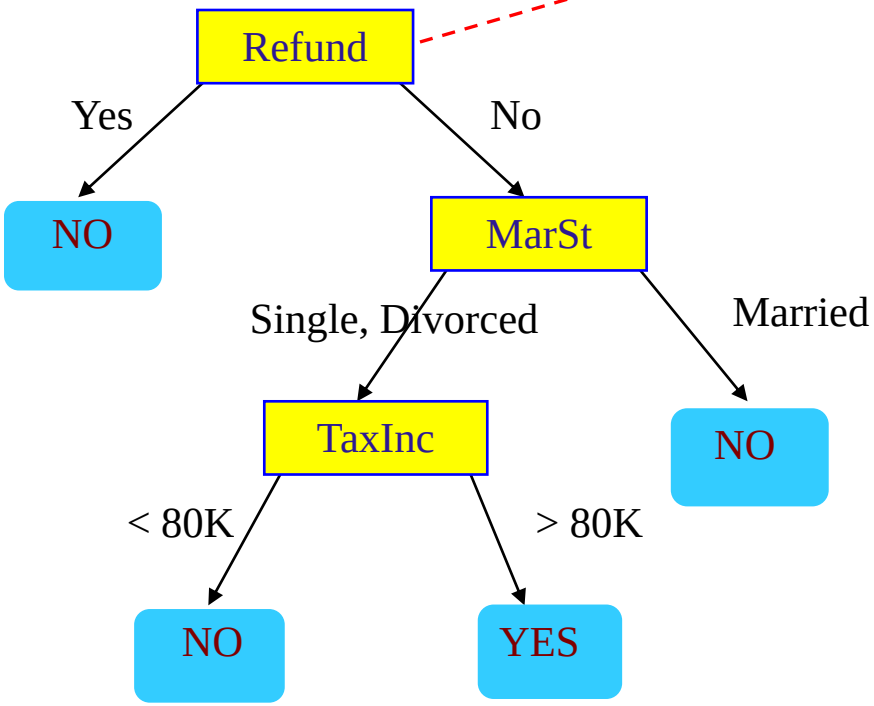
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Test Data

Apply Model to Test Data

Test Data

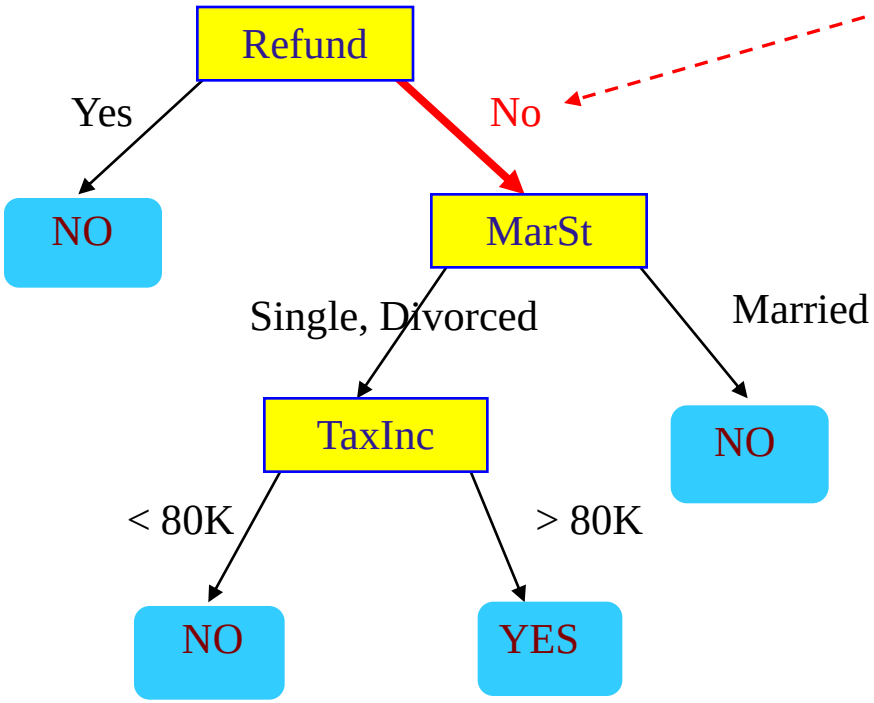
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

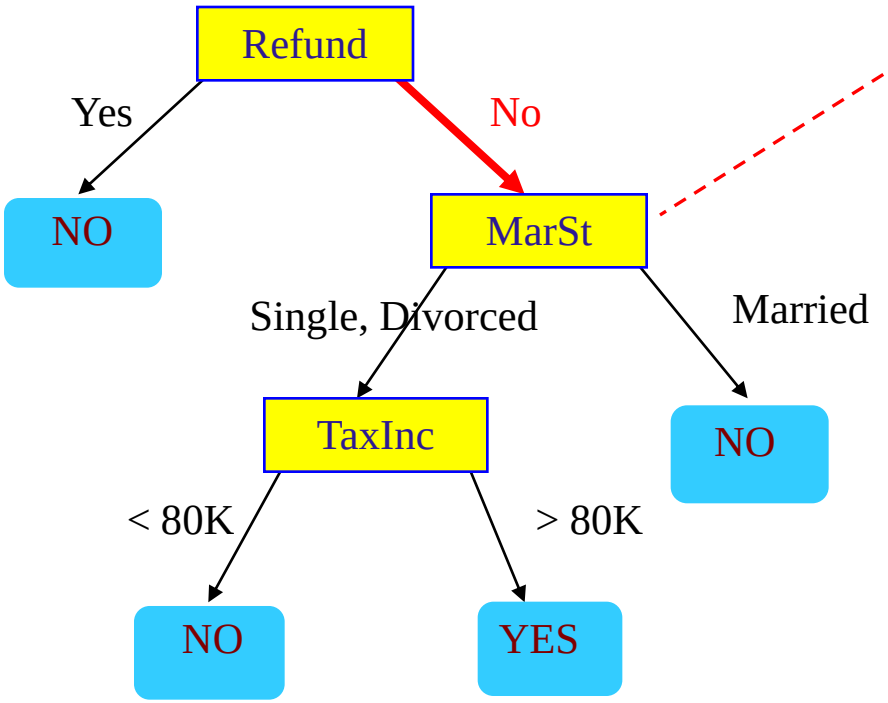
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

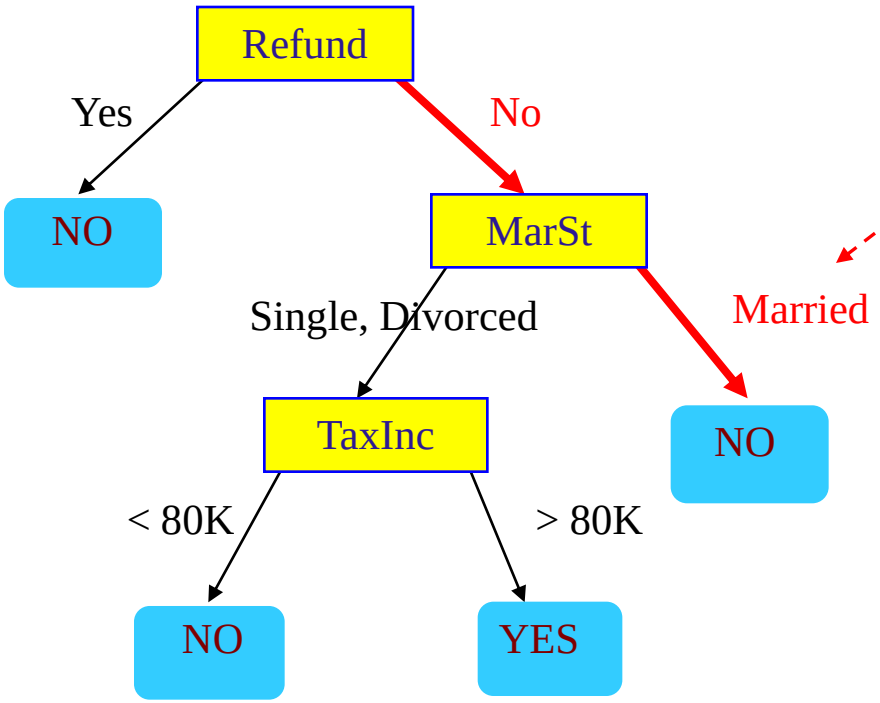
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

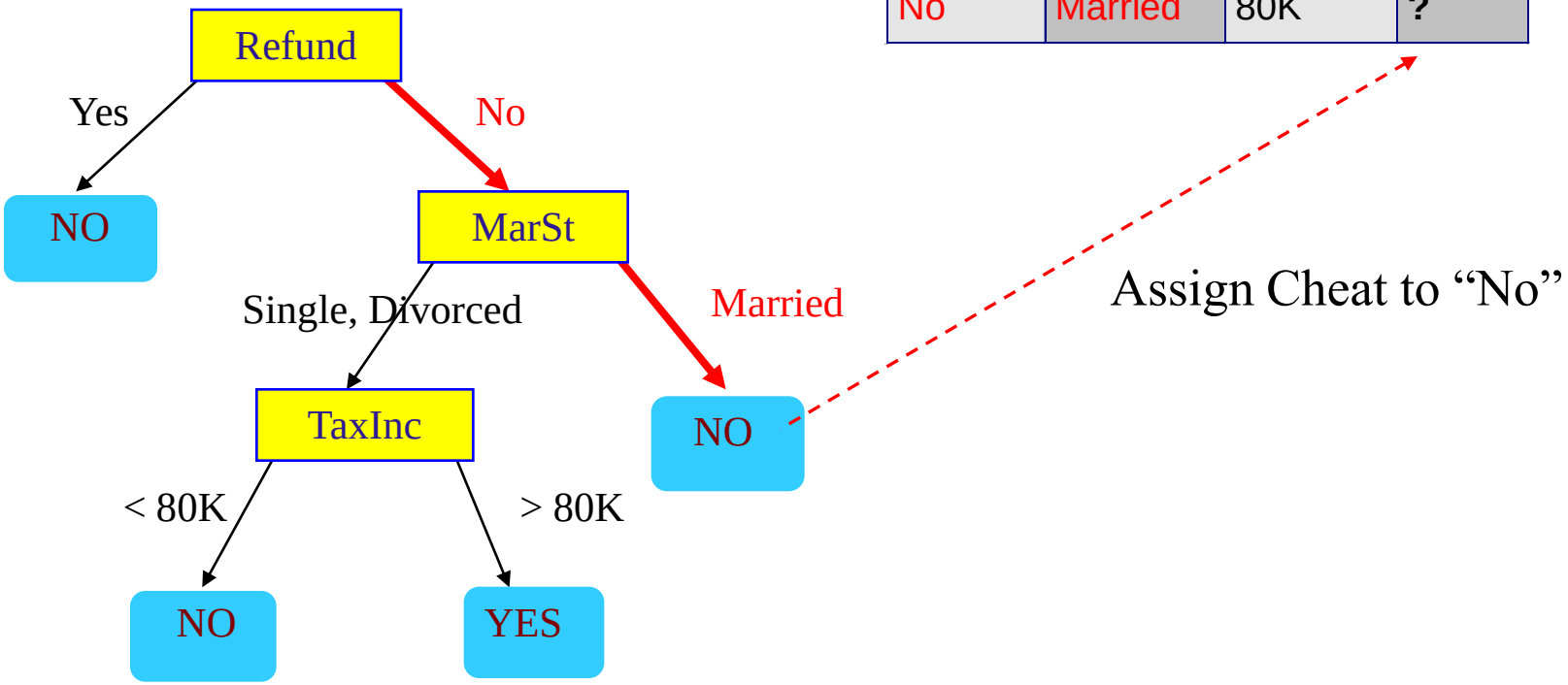
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

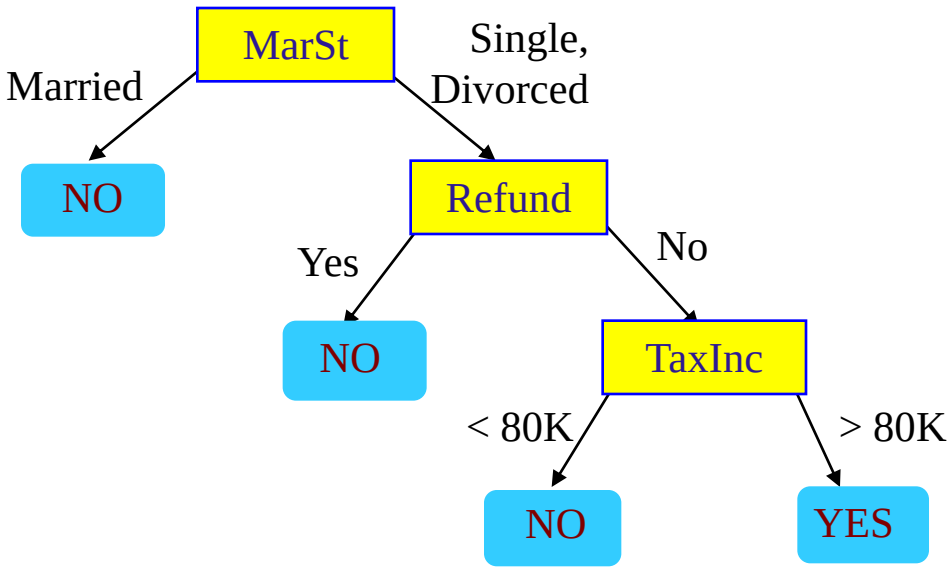
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Another Example of Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



There could be more than one tree that fits the same data!

DECISION TREE LEARNING ALGORITHM

function DECISION-TREE-LEARNING($examples, attributes, parent\_examples$) **returns** a tree

if $examples$ is empty **then return** PLURALITY-VALUE($parent\_examples$)
else if all $examples$ have the same classification **then return** the classification
else if $attributes$ is empty **then return** PLURALITY-VALUE($examples$)
else
  $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE($a, examples$)
  $tree \leftarrow$ a new decision tree with root test A
  for each value v_k of A **do**
    $exs \leftarrow \{e : e \in examples$ **and** $e.A = v_k\}$
    $subtree \leftarrow$ DECISION-TREE-LEARNING($exs, attributes - A, examples$)
    add a branch to $tree$ with label $(A = v_k)$ and subtree $subtree$
  return $tree$

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

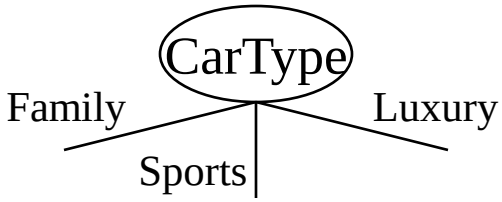
- The node N is labeled with the splitting criterion, which serves as a test at the node
- A branch is grown from node N for each of the outcomes of the splitting criterion.
- The tuples in D are partitioned accordingly

How to Specify Test Condition?

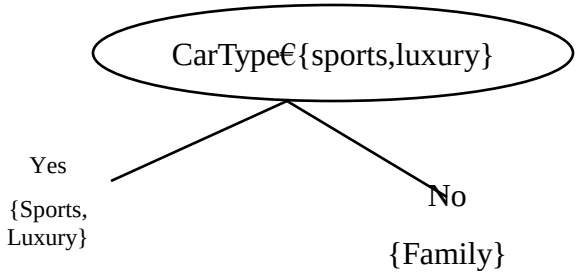
- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

- Multi-way split: Use as many partitions as distinct values.

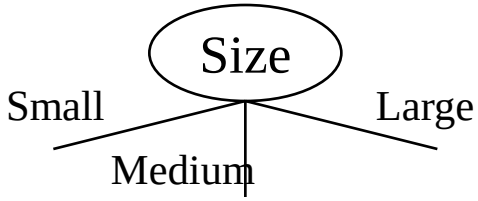


- Binary split: Divides values into two subsets. Need to find optimal partitioning.

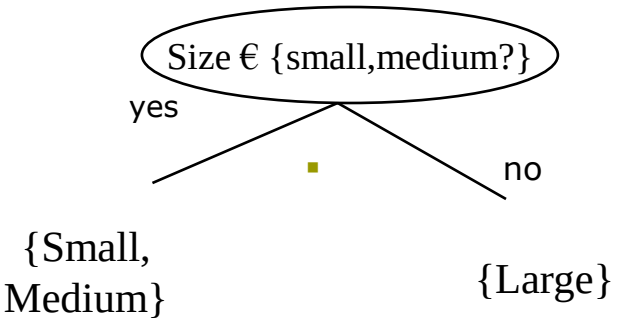


Splitting Based on Ordinal Attributes

- Multi-way split: Use as many partitions as distinct values.



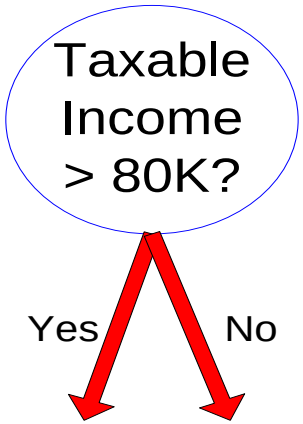
- Binary split: Divides values into two subsets. Need to find optimal partitioning.



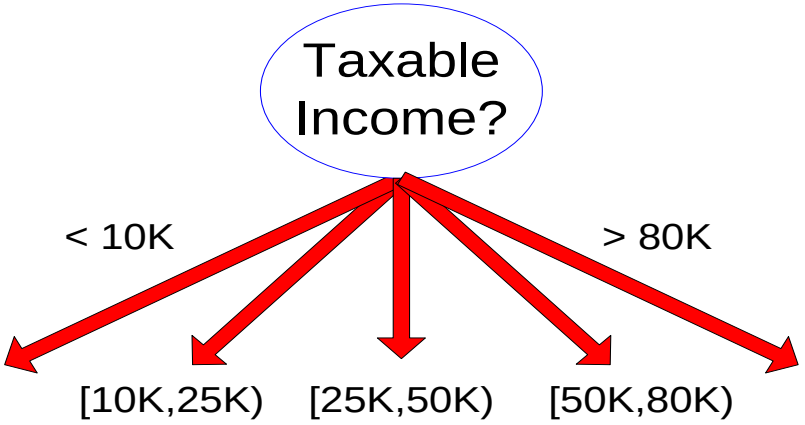
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

Attribute Selection Measure

- Information gain (ID3/C4.5)
  - All attributes are assumed to be categorical
  - Can be modified for continuous-valued attributes

- Conditions for stopping partitioning
  - All samples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left
  - Complexity=O(n* |D| * log(D))
  - n-no of attributes
  - |D|=no of training tuples

Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Assume there are **two classes, *….P* and *….N***
  - Let the set of examples *S* contain
    - ***…p* elements of class *P***
    - ***…n* elements of class *N***
  - The amount of information, needed to decide if an arbitrary example in *S* belongs to *P* or *N* is defined as $$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

Information Gain in Decision Tree Induction

- Assume that using attribute A

  a set *S* will be partitioned into sets $\{S_1, S_2, \ldots, S_v\}$

  - If S_i contains p_i examples of *P* and n_i examples of *N* ,the entropy, or the expected information needed to classify objects in all subtrees S_i is

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- The encoding information that would be gained by branching on *A*

$$Gain(A) = I(p, n) - E(A)$$

Attribute Selection by Information Gain Computation

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

Attribute Selection by Information Gain Computation

$$Gain(A) = I(p,n) - E(A)$$

$$I(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$$-9/(9+5)\log_2 9/(9+5) - 5/(9+5)\log_2 5/(9+5)$$

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"
- I(p, n) = I(9, 5) = 0.940
- Compute the entropy for *age*:

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

9 5

Attribute Selection by Information Gain Computation

- I(p, n) = I(9, 5) =0.940

$$Gain(age) = I(p,n) - E(age)$$

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i)$$

Gain(age) = 0.940 –0.692 = 0.248

$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.692$$

Similarly

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

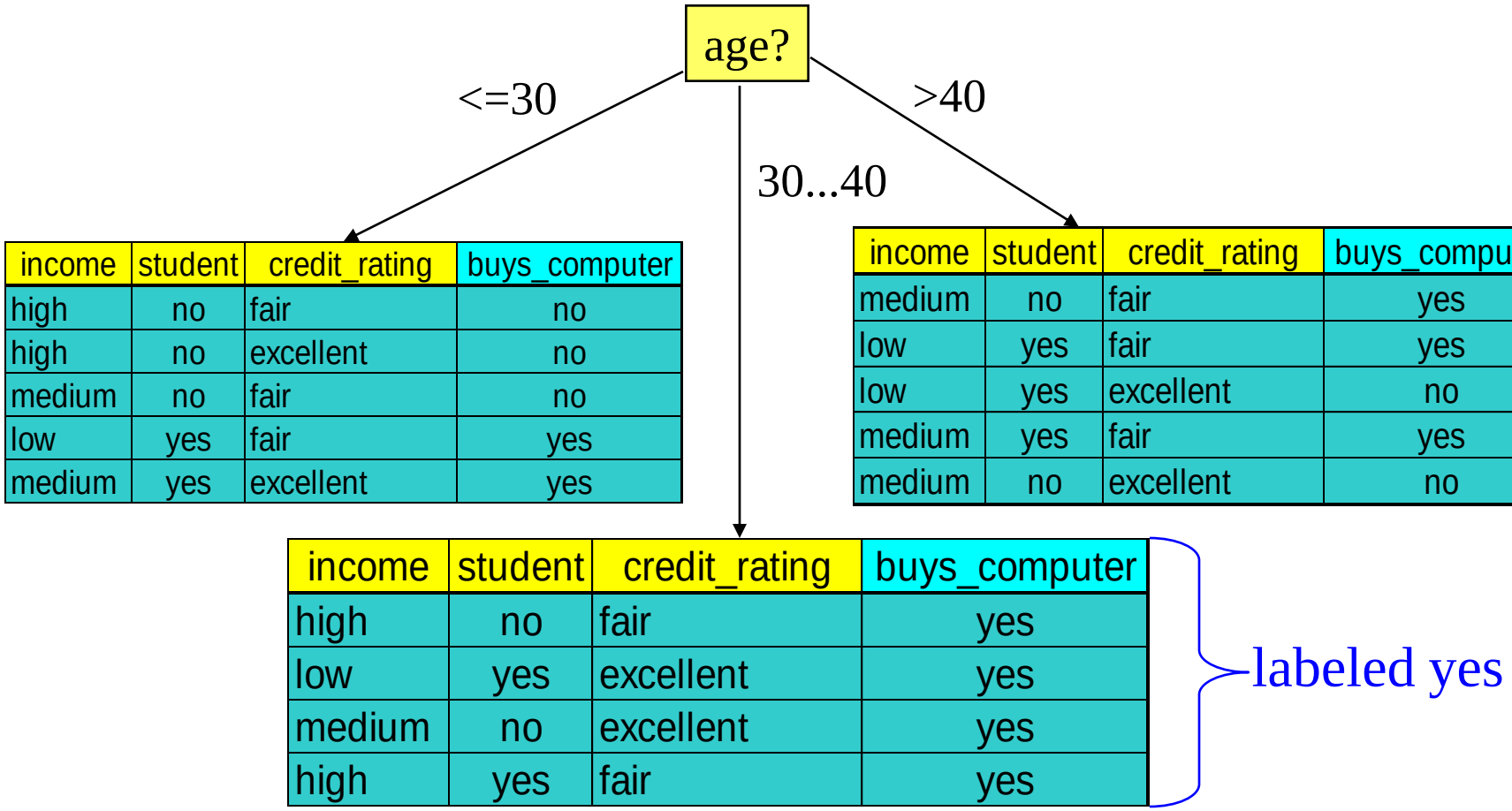
$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

- The attribute with the maximum Information Gain is selected as the splitting attribute

Splitting the samples using *age*



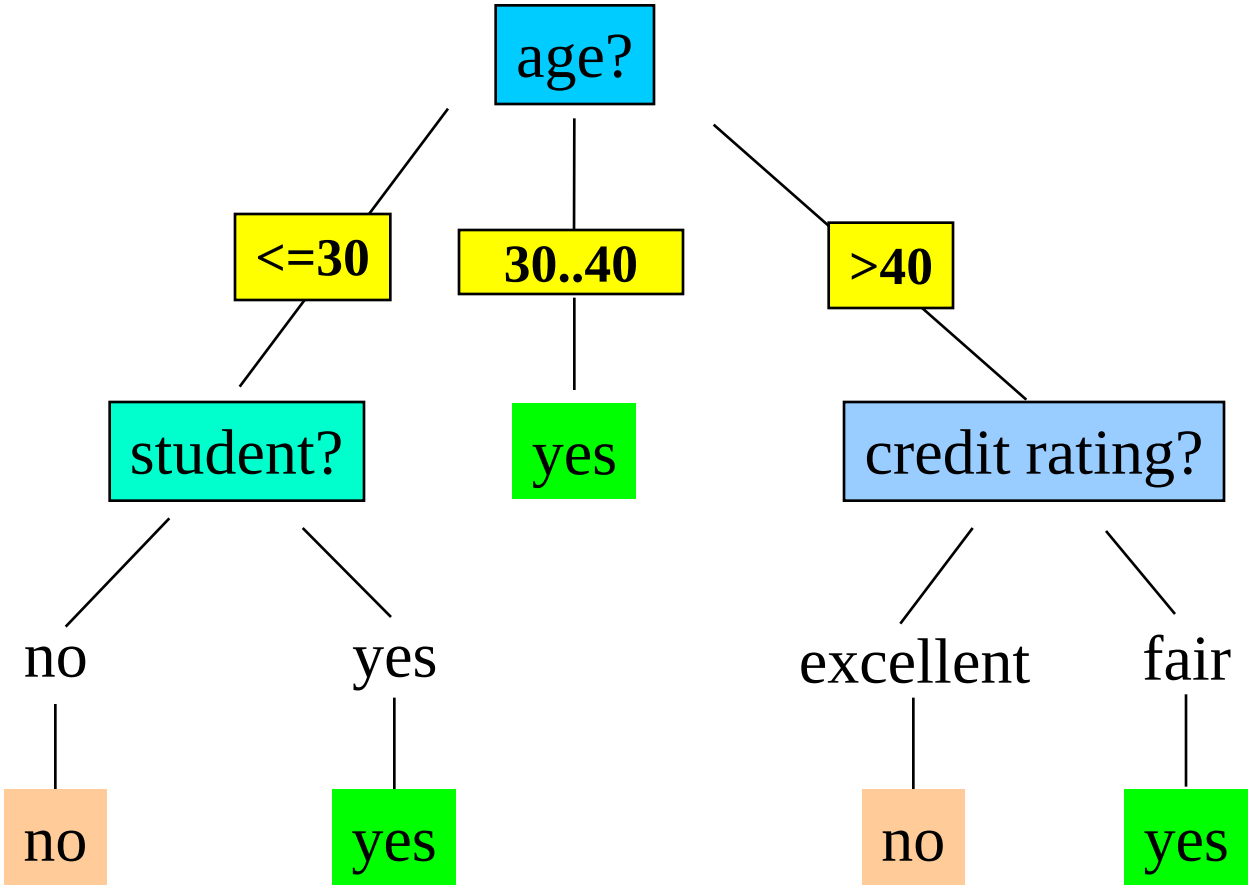
income	student	credit_rating	buys_computer
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	buys_computer
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	buys_computer
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

labeled yes

Output: A Decision Tree for "*buys_computer*"



Gini index (CART, IBM IntelligentMiner)

- Gini index measures the impurity of data set D

- If a data set *D* contains examples from *n* classes, gini index, *gini*(*D*) is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is probability that a tuple in D belong to class C

- Gini index considers a **binary split for each attribute**
- A is discrete valued and have v distinct values $\{a_1, a_2, a_3 \ldots\ldots a_v\}$
- To find best binary split on A consider all the subset formed by the values of A(exclude power set (all elements) and empty split)
- Each subset S_A can be considered as a binary test for attribute A of the form $A \in S_A$

•When considering a binary split we compute a weighted sum of the impurity of each resulting partition

---

•If a data set D is split on A into two subsets D_1 and D_2, the *gini* index *of D* is defined as

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

•For each attribute each of the possible binary split is considered

•. For a discrete valued attribute the subset that gives **minimum gini index** is selected

•For continuous valued attribute each possible split point must be considered.

–value sorted

-midpoint between adjacent values

-point which gives minimum gini index is selected

Gini index (CART, IBM IntelligentMiner)

Reduction in impurity that could be incurred by a binary split on a discrete or continuous valued attribute A is

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- **The attribute <u>that maximizes</u> the reduction in impurity(or have the minimum gini index)**

Gini index (CART, IBM IntelligentMiner)

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Gini index (CART, IBM IntelligentMiner)

- Ex. D has 9 tuples in buys_computer = “yes” and 5 in “no”

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Suppose the attribute income partitions D into 10 in D_1: {low, medium} and 4 in D_2 {high}

$$gini_{income \in \{low,medium\}}(D) = \left(\frac{10}{14}\right) Gini(D_1) + \left(\frac{4}{14}\right) Gini(D_2)$$

$$\begin{aligned} &Gini_{income \in \{low,medium\}}(D) \\ &= \frac{10}{14} Gini(D_1) + \frac{4}{14} Gini(D_2) \\ &= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) \\ &= 0.443 \\ &= Gini_{income \in \{high\}}(D). \end{aligned}$$

- Calculate the Gini index of remaining subsets Eg:- {medium, high} and{low}.
- Select the subset with the minimum Gini index value.

Gini index (CART, IBM IntelligentMiner)

Similarly, the Gini index values for splits on the remaining subsets are **0.458** (for the subsets {low, high} and {medium}) and 0.450 (for the subsets {medium, high} and {low}). Therefore, the best binary split for attribute income is on ***{low, medium} (or {high})*** because it minimizes the Gini index.

Evaluating age, we obtain ***{youth, senior} (or {middle_aged}***) as the best split for age with a Gini index of **0.357**; the attributes student and credit_rating are both binary, with Gini index values of 0.367 and 0.429, respectively.

Gini index (CART, IBM IntelligentMiner)

The attribute age and splitting subset {youth, senior} therefore give the **minimum Gini index overall**, with a reduction in impurity of . 0.459-0.357 =.102

The binary split “age {youth, senior?}” results in the maximum reduction in impurity of the tuples in D and is returned as the splitting criterion.

Comparing Attribute Selection Measures

- The three measures, in general, return good results but
  - Information gain:
    - biased towards multivalued attributes
  - Gini index:
    - biased towards multivalued attributes
    - has difficulty when no of classes is large
    - tends to favor tests that result in equal-sized partitions

Advantages of the Decision Tree

It is simple to understand as it follows the same process which a human follow while making any decision in real-life.

It can be very useful for solving decision-related problems

It helps to think about all the possible outcomes for a problem.

There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

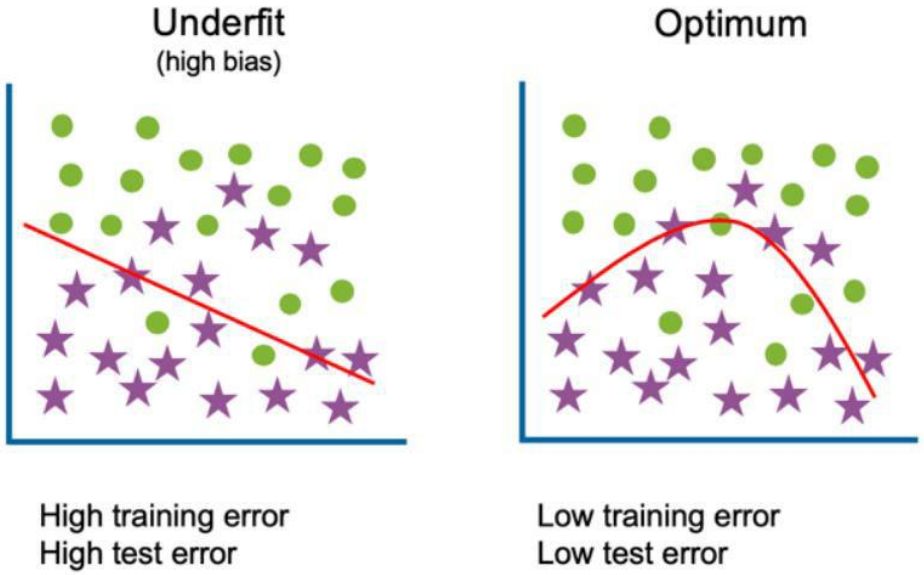
The decision tree contains lots of layers, which makes it complex.

It may have an overfitting issue, which can be resolved using the Random Forest algorithm.

For more class labels, the computational complexity of the decision tree may increase.

Avoid Overfitting in Classification

- When a decision tree is built many of the branches will reflect anomalies due to noise or outliers. The generated tree may overfit the training data
- Perform very well on the seen dataset but perform badly on unseen data or unknown instances. In such cases, the model is said to be Overfitting
- Result is in poor accuracy for unseen samples



Overfitting & underfitting are the two main errors/problems in the machine learning model, which cause poor performance in Machine Learning.

Overfitting occurs when the model fits more data than required, and it tries to capture each and every datapoint fed to it. Hence it starts capturing noise and inaccurate data from the dataset, which degrades the performance of the model.

An overfitted model doesn't perform accurately with the test/unseen dataset and can’t generalize well.

An overfitted model is said to have low bias and high variance.

Noise: Noise is meaningless or irrelevant data present in the dataset. It affects the performance of the model if it is not removed.

Bias: Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.

Variance: If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

Generalization: It shows how well a model is trained to predict unseen data

How to detect Overfitting?

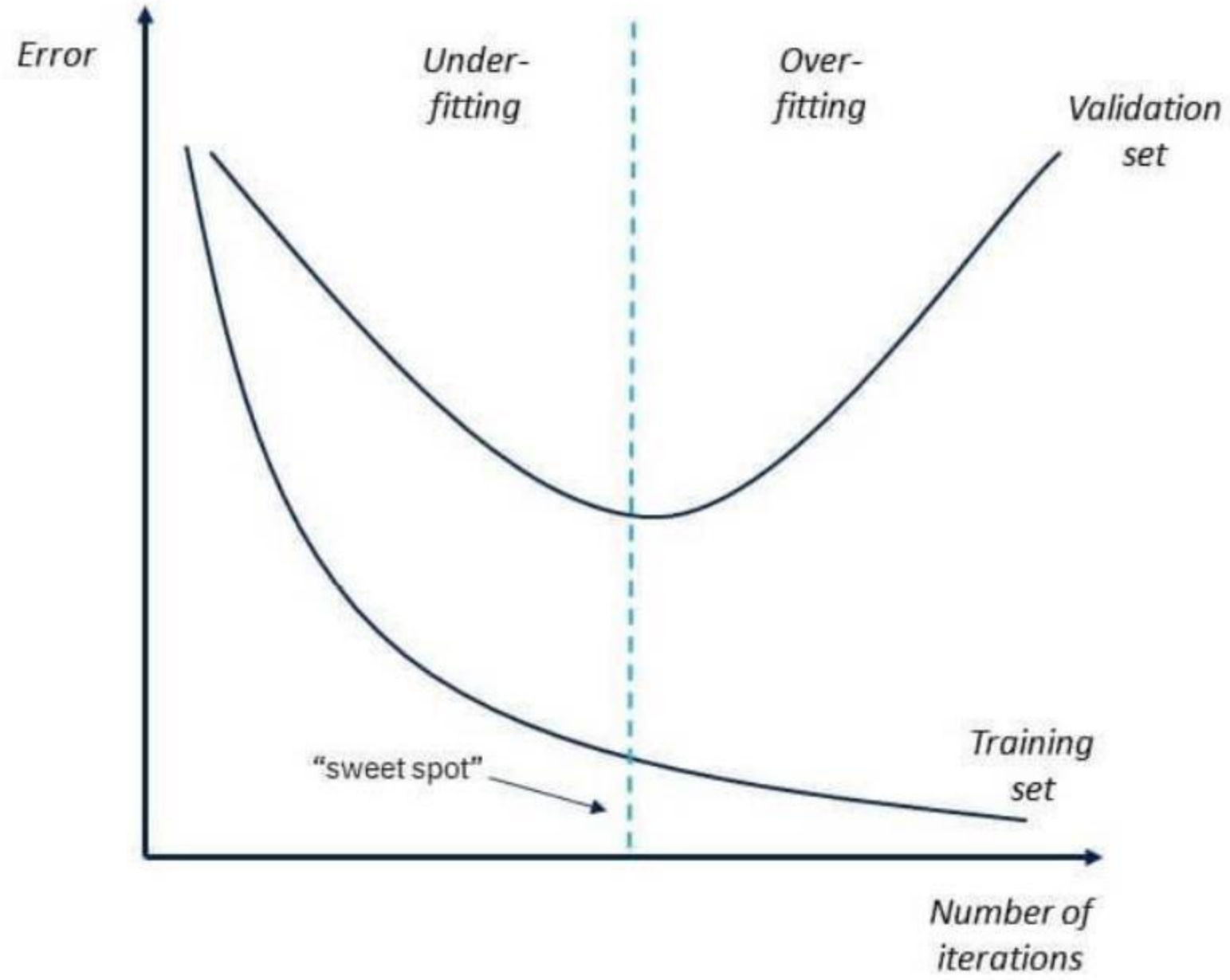
Overfitting in the model can only be detected once you test the data. To detect the issue, we can perform Train/test split.

In the train-test split of the dataset, we can divide our dataset into random test and training datasets.

We train the model with a training dataset which is about 80% of the total dataset. After training the model, we test it with the test dataset, which is 20 % of the total dataset.

Now, if the model performs well with the training dataset but not with the test dataset, then it is likely to have an overfitting issue.

For example, if the model shows 85% accuracy with training data and 50% accuracy with the test dataset, it means the model is not performing well.


Error
Under-
fitting
Over-
fitting
Validation
set
Training
set
"sweet spot"
Number of
iterations

Ways to prevent the Overfitting

Although overfitting is an error in Machine learning which reduces the performance of the model, however, we can prevent it in several ways.

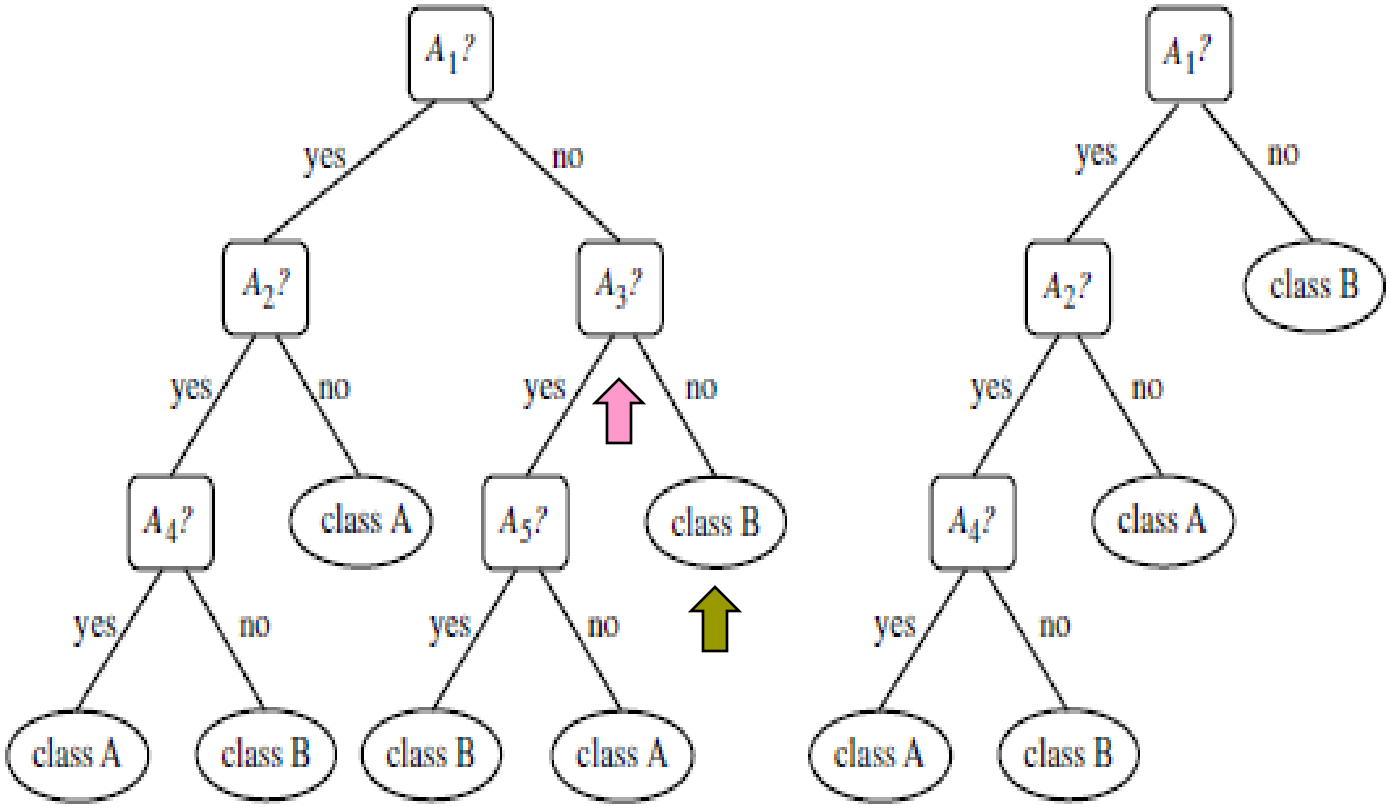
With the use of the linear model, we can avoid overfitting; however, many real-world problems are non-linear ones. Below are several ways that can be used to prevent overfitting:

1. Early Stopping
2. Train with more data
3. Feature Selection
4. Cross-Validation
5. Data Augmentation
6. Regularization

Tree Pruning

- Two approaches to avoid overfitting
  - Prepruning: Halt tree construction early—do not split a node if this would result in the goodness measure falling below a threshold
    - When constructing a tree measures such as information gain,gini index etc are used to assess the goodness of split
      - Difficult to choose an appropriate threshold
  - Postpruning: removes subtrees from a "fully grown" tree—subtree is pruned by removing its branches and replacing it with a leaf.
  - The leaf is labeled with most frequent class among the subtree being replaced

Tree Pruning



An unpruned decision tree and a pruned version of it.

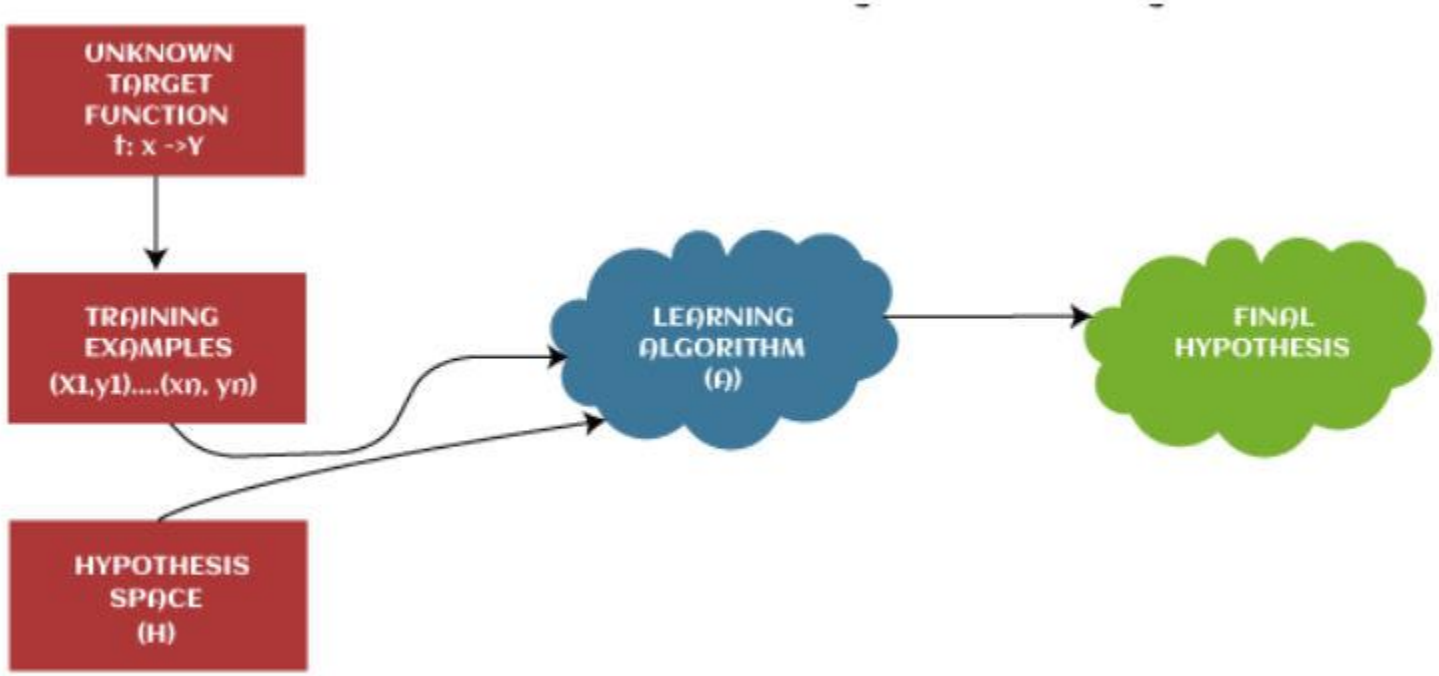
Hypothesis in Machine Learning

The hypothesis is defined as the supposition or proposed explanation based on insufficient evidence or assumptions

It is just a guess based on some known facts but has not yet been proven. A good hypothesis is testable, which results in either true or false.

Example: Let's understand the hypothesis with a common example. Some scientist claims that ultraviolet (UV) light can damage the eyes then it may also cause blindness.

In this example, a scientist just claims that UV rays are harmful to the eyes, but we assume they may cause blindness. However, it may or may not be possible. Hence, these types of assumptions are called a hypothesis.


UNKNOWN TARGET FUNCTION f: x ->Y
TRAINING EXAMPLES (X1,y1)....(xn, yn)
HYPOTHESIS SPACE (H)
LEARNING ALGORITHM (A)
FINAL HYPOTHESIS

In supervised learning techniques, the main aim is to determine the possible hypothesis out of hypothesis space that best maps input to the corresponding or correct outputs.

There are some common methods given to find out the possible hypothesis from the Hypothesis space, where hypothesis space is represented by uppercase-h (H) and hypothesis by lowercase-h (h).

Hypothesis space (H):

Hypothesis space is defined as a set of all possible legal hypotheses; hence it is also known as a hypothesis set.

It is used by supervised machine learning algorithms to determine the best possible hypothesis to describe the target function or best maps input to output.

It is often constrained by choice of the framing of the problem, the choice of model, and the choice of model configuration

Hypothesis (h)

It is defined as the approximate function that best describes the target in supervised machine learning algorithms. It is primarily based on data as well as bias and restrictions applied to data.

Hence hypothesis (h) can be concluded as a single hypothesis that maps input to proper output and can be evaluated as well as used to make predictions.

The hypothesis (h) can be formulated in machine learning as follows: $y= mx + b$

Where,

Y: Range

m: Slope of the line which divided test data or changes in y divided by change in x.

x: domain

c: intercept (constant)

Evaluating and Choosing the Best Hypothesis

- We assume that there is a probability distribution over examples that remains stationary over time
  - Each observed value is sampled from that distribution and is independent of previous examples and
  - Each example has identical prior probability distribution
- Examples that satisfy these assumptions are called independent and identically distributed (i.i.d.)
- The ***error rate*** of a hypothesis h is the proportion of mistakes it makes
  - The proportion of times that $h(x) \neq y$ for an (x, y) example
- Just because a hypothesis h has low error rate on the training set does not mean that it will generalize well

Model selection: Complexity vs. goodness of fit

- We can think of finding the best hypothesis as two tasks:
  - Model selection defines the hypothesis space and
  - Optimization finds the best hypothesis within that space
- How to select among models that are parameterized by size
  - With polynomials we have size = 1 for linear functions, size = 2 for quadratics, and so on
  - For decision trees, the size could be the # of nodes in the tree
- We want to find the value of the size parameter that best balances underfitting and overfitting to give the best test set accuracy

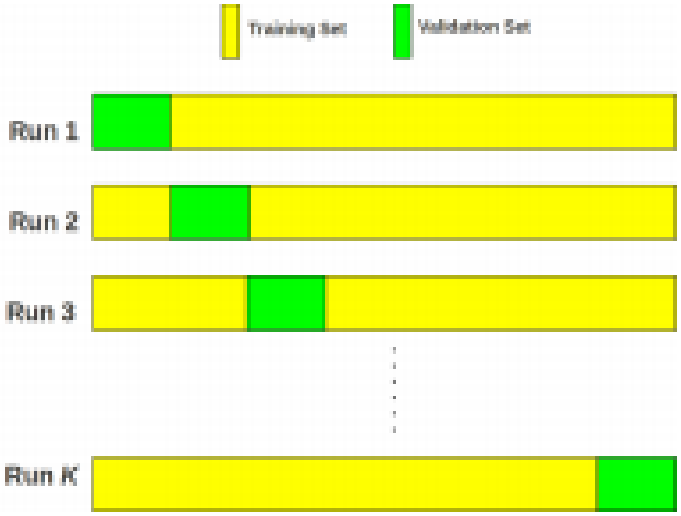
Model selection: Complexity vs. goodness of fit

- A wrapper takes a learning algorithm as an argument (DT learning for example)
- The wrapper enumerates models according to the size parameter
- For each size, it uses cross validation (say) on the learner to compute the average error rate on training and test sets
- We start with the smallest, simplest models (which probably underfit the data), and iterate, considering more complex models at each step, until the models start to overfit
- The cross validation picks the value of size with the lowest validation set error
- We then generate a hypothesis of that size using all the data (without holding out any of it; eventually we should evaluate the returned hypothesis on a separate test set)

- Cross Validation
  - K-Fold Cross-Validation
  - Leave-one-out Cross-Validation

Cross-Validation

- K-fold Cross-Validation
  - Create K equal sized partitions of the training data
  - Each partition has N/K examples
  - Train using $K-1$ partitions, validate on the remaining partition
  - Repeat the same K times, each with a different validation partition

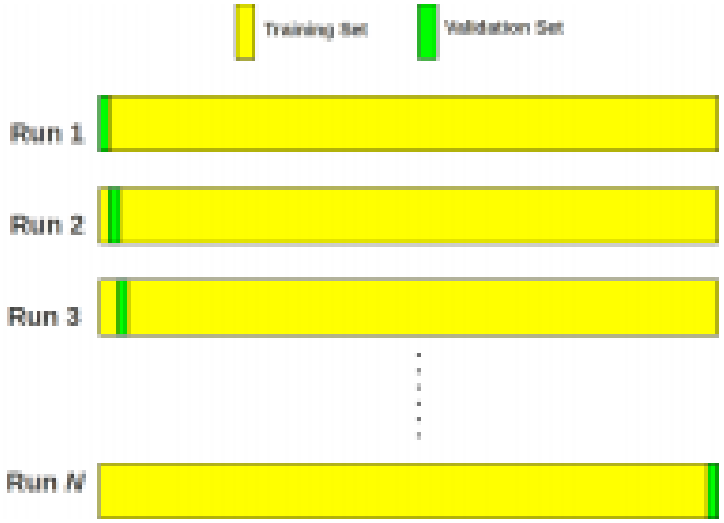


  - Finally, choose the model with smallest **average** validation error
  - Usually K is chosen as 10

Leave-One-Out (LOO) Cross-Validation

Special case of K-fold CV when $K = N$ (number of training examples)

- Each partition is now an example
- Train using $N - 1$ examples, validate on the remaining example
- Repeat the same N times, each with a different validation example



- Finally, choose the model with smallest **average** validation error
- Can be expensive for large N. Typically used when N is small

Regression and classification with Linear models

Video Lecture :

https://www.fau.tv/clip/id/30379

Material :

https://people.cs.umass.edu/~barto/courses/CS383-Fall11/383-Fall11-Lec19.pdf