Unit 2
Machine Learning

Artificial Intelligence

School of Cyber Security & Digital Forensics

M. Sc. Cyber Security (Semester-I)

What is Machine Learning ?

- Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed. —Arthur Samuel, 1959
- A computer program is said to learn from **experience E** with respect to some **task T** and some performance **measure P**, if its performance on T, as measured by P, improves with experience E. —Tom Mitchell, 1997

Types of machine learning systems

- **Whether or not they are trained with human supervision**
  - **Supervised learning**: Tasks:- Classification and Prediction

    (Popular algorithms: k-Nearest Neighbors, Linear Regression, Logistic Regression, Support Vector Machines, Decision Trees and Random Forests, Neural networks*)
  - **Unsupervised learning:** Tasks:- Clustering , Anomaly detection, Novel detection & Association rule learning

    (Popular algorithms: K-Means, DBSCAN, Hierarchical Cluster Analysis (HCA)
  - **Semi-supervised learning:** Mostly combination of supervised and unsupervised learning
  - **Reinforcement learning:** Learning system called **agent** learns best strategy based on **rewards** and **penalty**

Types of machine learning systems

- **<u>Whether or not the system can learn incrementally from a stream of incoming data.</u>**
  - Batch learning or offline learning
  - Online learning
- **<u>Based on how they generalize.</u>**
  - Instance-based learning
  - Model-based learning

Challenges of Machine Learning

- Data challenges
  - Insufficient Quantity of Training Data
  - Non-representative Training Data
  - Poor quality of data
  - Irrelevant features
- Algorithm challenges
  - Overfitting
  - Underfitting

Testing and Validating

- **Training set and Testing set**
- Generalization error or out-of-sample error : Error rate on new cases
- Testing set gives an estimate of generalization error
- If training error < generalization error ; means overfitting
- **Hyperparameter Tuning and Model Selection**
- **Validation set –** if size too small cross validation

Machine learning project: Main steps

- Data Collection
- Discover and visualize data …(Understanding datasets)
- Prepare data for machine learning algorithms … (Data cleaning, Feature Encoding ,Feature Selection, Feature Normalization,)
- Select a model and train
- Fine tune model
- Present Model
- Launch, monitor and maintain the system

Note: *Reference for contents till this slide*: Chapter 1 of 2-Aurélien-Géron-Hands-On-Machine-Learning-with-Scikit-Learn-Keras-and-Tensorflow_-Concepts-Tools-and-Techniques-to-Build-Intelligent-Systems-O'Reilly-Media-2019

Prepare data for machine learning algorithms

- Data Cleaning (source: 3.2 Han Kamber Data mining)

  1. Missing Values

     i. Ignore the tuple

     ii. Fill missing values manually

     iii. Use a global constant

     iv. Use a measure of central tendency

     v. Use mean/median for all samples belonging to same class as given tuple

     vi. Use most probable value

     Although we can try our best to clean the data after it is seized, good database and data entry procedure design should help minimize the number of missing values or errors in the first place.

Prepare data for machine learning algorithms

- Data Cleaning (source: 3.2 Han Kamber Data mining)

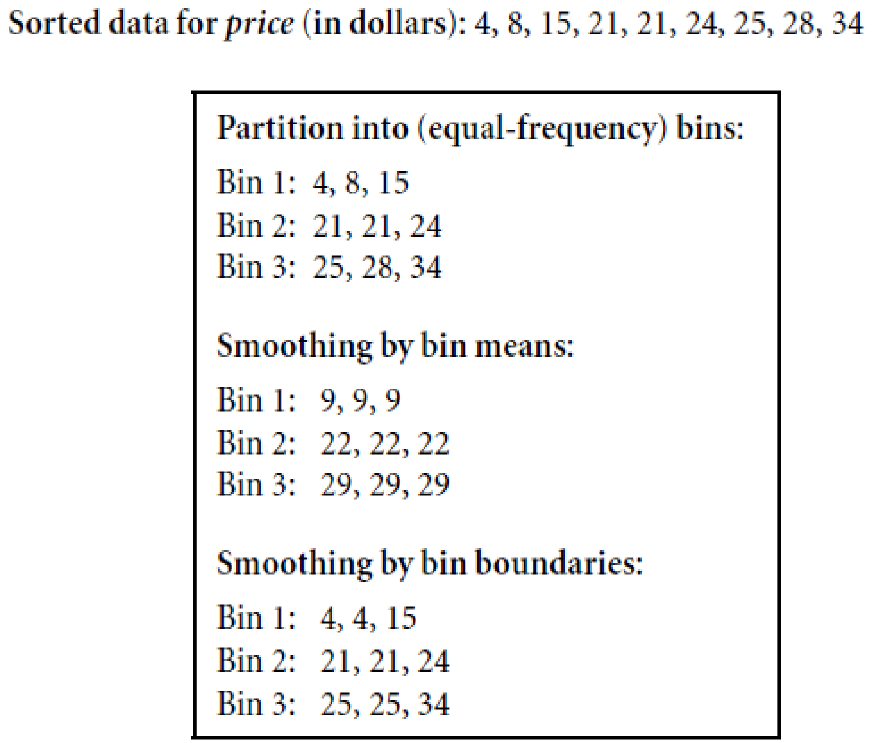
  1. Missing Values
  2. Noisy Data : **Noise** is a random error or variance in a measured variable

  Following are data smoothing techniques:

  i. **Binning** : (Equal width and Equal frequency) Smoothing by bin means, Smoothing by bin medians, Smoothing by bin boundaries

  ii. **Regression**

  iii. **Outlier analysis**



Sorted data for *price* (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

Partition into (equal-frequency) bins:

Bin 1: 4, 8, 15
Bin 2: 21, 21, 24
Bin 3: 25, 28, 34

Smoothing by bin means:

Bin 1: 9, 9, 9
Bin 2: 22, 22, 22
Bin 3: 29, 29, 29

Smoothing by bin boundaries:

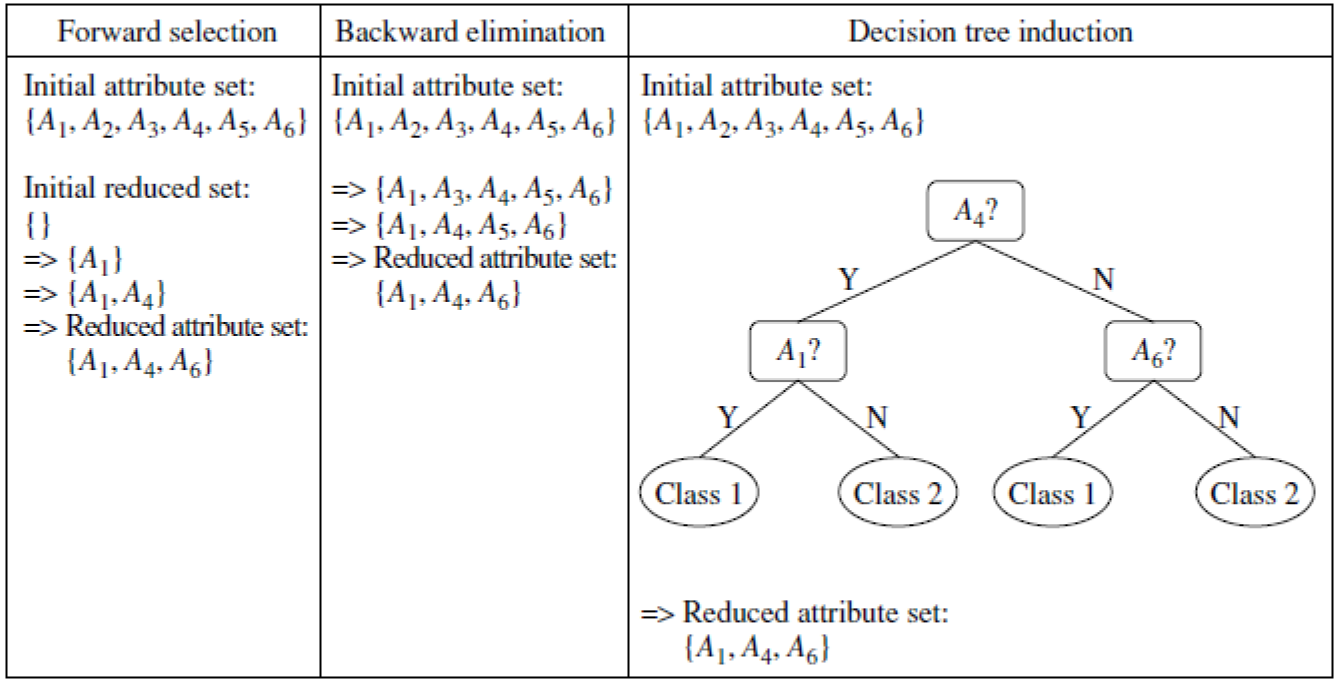
Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

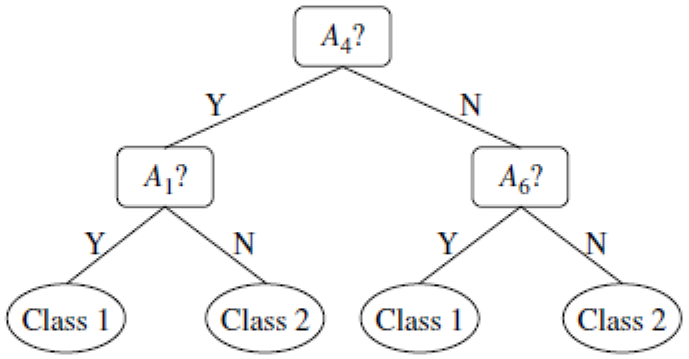
Prepare data for machine learning algorithms

- **Feature Selection/ Attribute subset selection** (source: 3.4.4 Han Kamber Data mining) : reduces the data set size by **removing irrelevant** or **redundant** attributes (or dimensions). The goal of attribute subset selection is to find a minimum set of attributes such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.
- **Feature Selection:** The "best" (and "worst") attributes are typically determined using tests of statistical significance, which assume that the attributes are independent of one another. Many other attribute evaluation measures can be used such as the information gain measure

Prepare data for machine learning algorithms

- **Feature Selection**: Following are some heuristic approaches:
  1. Stepwise forward selection
  2. Stepwise backward elimination
  3. Combination of forward selection and backward elimination
  4. Decision tree induction
  5. Regression



Forward selection	Backward elimination	Decision tree induction
Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ Initial reduced set: $\{\}$ => $\{A_1\}$ => $\{A_1, A_4\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$ => $\{A_1, A_3, A_4, A_5, A_6\}$ => $\{A_1, A_4, A_5, A_6\}$ => Reduced attribute set: $\{A_1, A_4, A_6\}$	Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$  => Reduced attribute set: $\{A_1, A_4, A_6\}$

Prepare data for machine learning algorithms

- **Feature Normalization:** (Source :Section 3.5.2 Han Kamber) Normalizing the data attempts to give all attributes an equal weight. Normalization is particularly useful for classification algorithms involving neural networks or distance measurements such as nearest-neighbor classification and clustering. Following are some methods for normalization

  1. Min-max normalization $$v_i' = \frac{v_i - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A.$$

  2. Z-score normalization $$v_i' = \frac{v_i - \bar{A}}{\sigma_A},$$

  3. Decimal Scaling $$v_i' = \frac{v_i}{10^j},$$

  where j is smallest integer such that max($|\boldsymbol{v_i'}|$) **< 1**

Performance measures

- **Classification evaluation**:
  - Accuracy = (TP + TN) / (P + N)
  - Precision = (TP) / (TP + FP)
  - Recall / Sensitivity / TPR = TP /P
  - Specificity / TNR = TN/N
  - F1-score = (2 * precision * recall) / (precision + recall)
- Confusion Matrix is a tabular visualization of the **ground-truth labels versus model predictions**.

		Predicted	
		Positive (P)	**Negative (N)**
Actual	**Positive (P)**	TP	FN
	Negative (N)	FP	TN

Performance measures

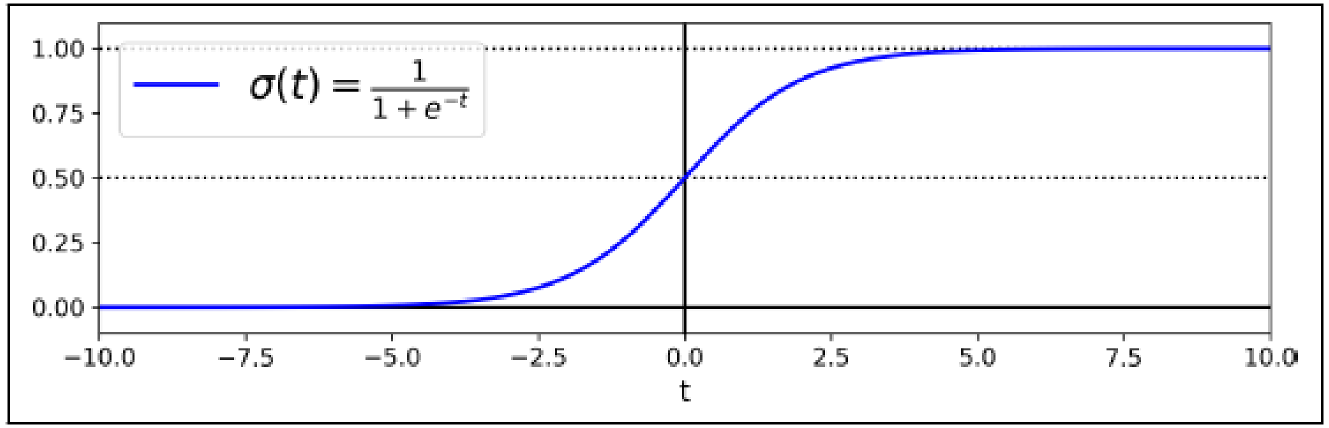
- **Regression evaluation**:
  - Mean Absolute Error = $\frac{\sum_{i=1}^{N} |\widehat{y_j} - y_j|}{N}$
  - Mean Squared Error = $\frac{\sum_{i=1}^{N} (\widehat{y_j} - y_j)^2}{N}$
  - Root Mean Squared Error = $\sqrt{\frac{\sum_{i=1}^{N} (\widehat{y_j} - y_j)^2}{N}}$

Models of Supervised learning

- Linear Regression: Prediction
- Logistic Regression: Classification
- k-Nearest Neighbour (kNN) : Classification
- Naïve Bayes : Classification
- Decision Trees : Classification
- Support Vector Machine (SVM) : Classification

Logistic Regression /Logit Regression

- Binary Classification problems: Email spam /not spam ; (Source: Hands on ML pg. no 144)
- Just like a Linear Regression model, a Logistic Regression model computes a weighted sum of the input features (plus a bias term), but instead of outputting the result directly like the Linear Regression model does, it outputs the *logistic* of this result $\widehat{p} = \sigma\,(x^T\beta)$
- $\sigma(t) = \frac{1}{1+e^{-t}}$ Logistic / Sigmoid function (S-shaped)
- $\hat{y} = \begin{cases} 0, \widehat{p} < 0.5 \\ 1, \widehat{p} \geq 0.5 \end{cases}$



kNN Model

- No learning required; (Source: Machine learning mastery pg. no 98)
- KNN makes predictions using the training dataset directly. Predictions are made for a new data point by searching through the entire training set for the K most similar instances
- To determine which of the K instances in the training dataset are most similar to a new input a distance measure is used. For real-valued input variables, the most popular distance measure is **Euclidean distance**.
- $dist(a,b) = \sqrt{\sum_{i=1}^{N}(a_i - b_i)^2}$

kNN Model

- Instance –based learning ; Lazy learning
- If you are using K and you have an even number of classes (e.g. 2) it is a good idea to choose a K value with an odd number to avoid a tie. And the inverse, use an even number for K when you have an odd number of classes.

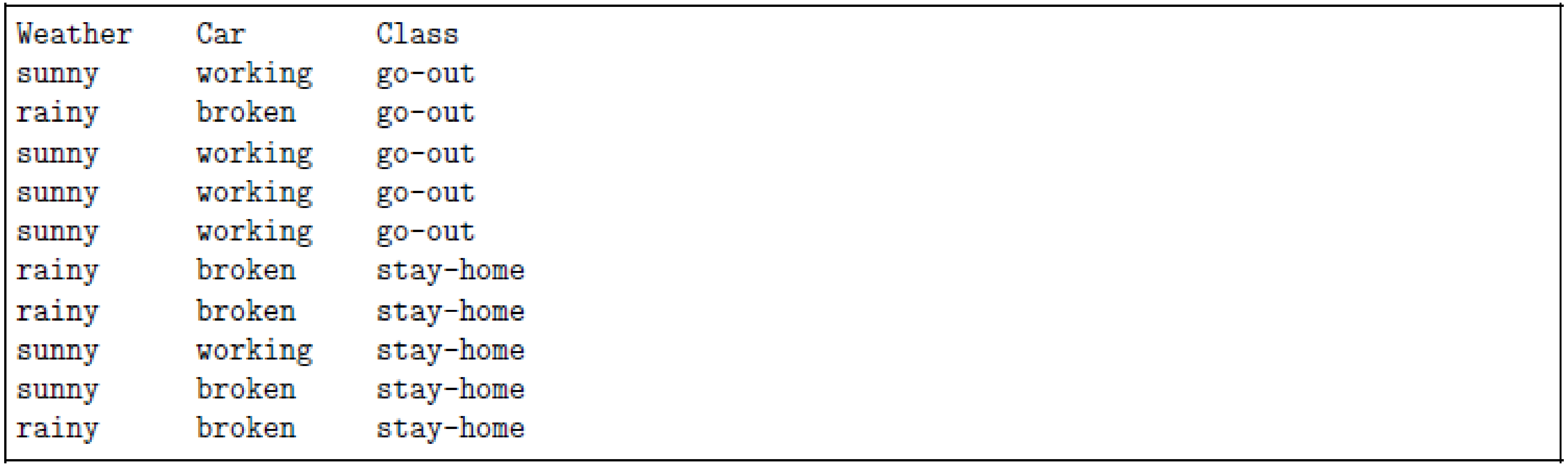
X1	X2	Y
7	7	B
7	4	B
3	4	G
1	4	G
3	7	?

Naïve Bayes Model

- Naïve Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. (Source: Han Kamber Pg. 350 and Master Machine learning algorithm)
- Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X and represents each C_i class.
- $P(C_i|X) = \frac{P(X|C_i)\,P(C_i)}{P(X)}$ $maximize\, P(C_i|X)$
- *Laplacian correction*

Naïve Bayes Model

- $Weather = sunny\,, Car = Working\,, class = ?$



Weather	Car	Class
sunny	working	go-out
rainy	broken	go-out
sunny	working	go-out
sunny	working	go-out
sunny	working	go-out
rainy	broken	stay-home
rainy	broken	stay-home
sunny	working	stay-home
sunny	broken	stay-home
rainy	broken	stay-home

Decision Tree

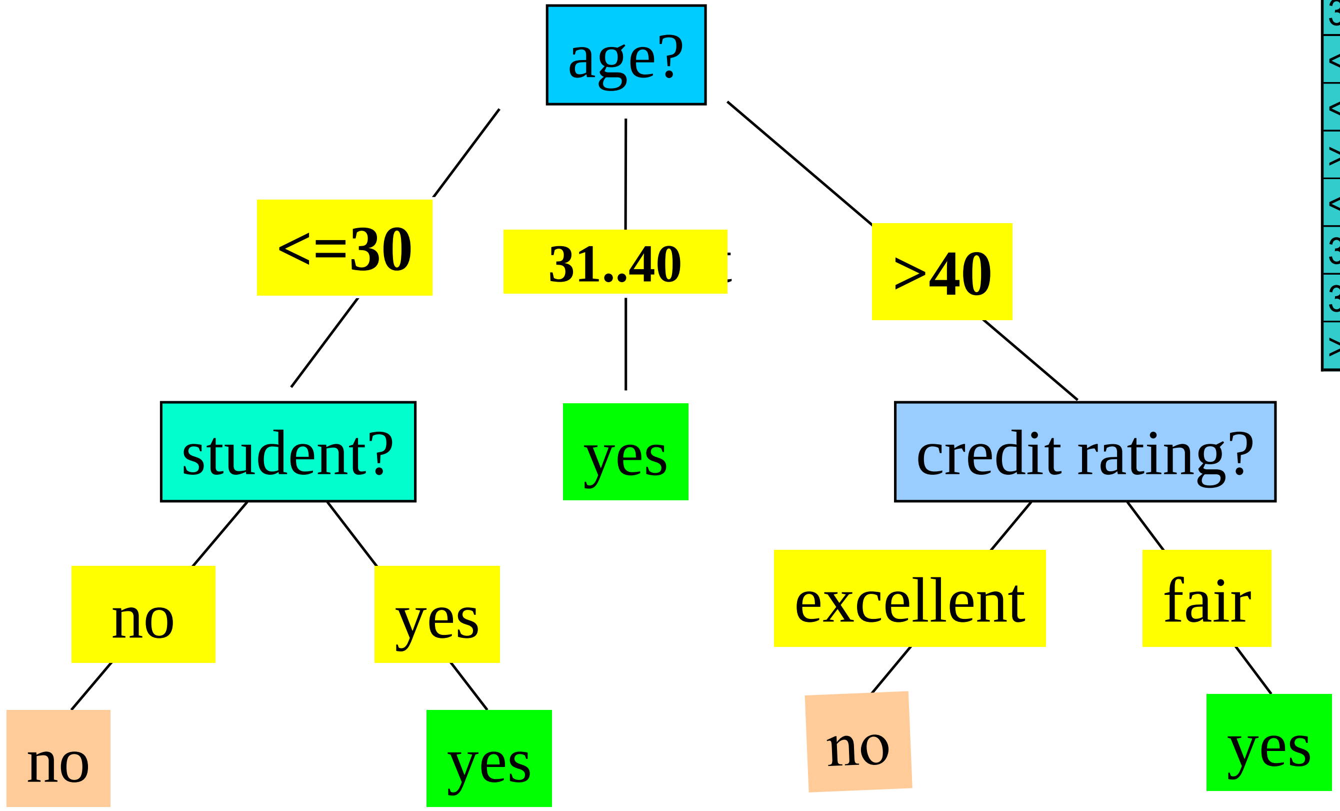
- **Decision tree induction** is the learning of decision trees from class-labeled training tuples. (Source: Han Kamber Pg. 330)
- A **decision tree** is a flowchart-like tree structure, where each **internal node** (nonleaf node) denotes a test on an attribute, each **branch** represents an outcome of the test, and each **leaf node** (or *terminal node*) holds a class label. The topmost node in a tree is the **root** node.

Decision Tree

- An **attribute selection measure** is a heuristic for selecting the splitting criterion that "best" separates a given data partition, *D*, of class-labeled training tuples into individual classes.
- Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split.
- ID3 uses Information Gain , CART uses Gini Index, C4.5 uses Gain ratio

Decision Tree Induction: An Example

- Training data set: Buys_computer
- Resulting tree:



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

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)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- 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

Brief Review of Entropy

- Entropy (Information Theory)
  - A measure of uncertainty associated with a random variable
  - Calculation: For a discrete random variable *Y* taking *m* distinct values $\{y_1, \ldots, y_m\}$,
    - $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$
  - Interpretation:
    - Higher entropy => higher uncertainty
    - Lower entropy => lower uncertainty
- Conditional Entropy
  - $H(Y|X) = \sum_x p(x) H(Y|X = x)$

Attribute Selection Measure: Information Gain (ID3)

- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i, estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D:

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

- Information gained by branching on attribute A

$$Gain(A) = Info(D) - Info_A(D)$$

Attribute Selection: Information Gain

- Class P: buys_computer = “yes”
- Class N: buys_computer = “no”

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$$

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

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

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

$\frac{5}{14}I(2,3)$ means “age <=30” has 5 out of 14 samples, with 2 yes’es and 3 no’s. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

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

Computing Information-Gain for

- Let attribute A be a continuous-valued attribute
- Must determine the *best split point* for A
  - Sort the value A in increasing order
  - Typically, the midpoint between each pair of adjacent values is considered as a possible *split point*
    - $(a_i+a_{i+1})/2$ is the midpoint between the values of a_i and a_{i+1}
  - The point with the *minimum expected information requirement* for A is selected as the split-point for A
- Split:
  - D1 is the set of tuples in D satisfying A ≤ split-point, and D2 is the set of tuples in D satisfying A > split-point