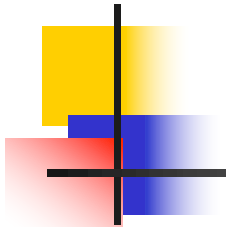
Classification vs. Prediction

- Classification:
  - predicts categorical class labels
  - classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data
- Prediction:
  - models continuous-valued functions, i.e., predicts unknown or missing values
- Typical Applications
  - credit approval
  - target marketing
  - medical diagnosis
  - treatment effectiveness analysis

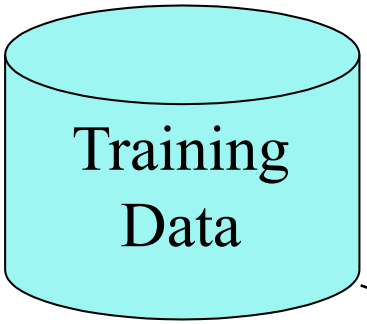
Classification—A Two-Step Process

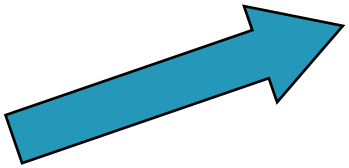
- **Model construction**: describing a set of predetermined classes or concepts
  - Input:
    - Database tuples described by attributes
  - Each tuple/sample is assumed to belong to a predefined class
    - determined by the class label attribute
    - Thus, it is supervised learning
  - Training data set
    - The set of tuples used for model construction
  - Training samples
    - Randomly selected from the sample polulation
  - The model is represented as classification rules, decision trees, or mathematical formulae

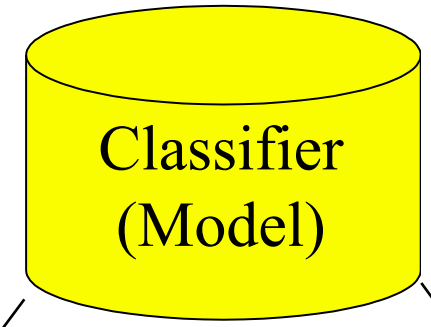


- **Model usage**: for classifying future or unknown objects
  - Estimate accuracy of the model (or classifier)
    - Holdout method
      - Uses a test set of class-labeled samples
  - The known label of test sample is compared with the classified result from the model
  - Accuracy rate
    - the percentage of test set samples that are correctly classified by the model
  - Test set is independent of training set, otherwise over-fitting will occur

Classification Process (1): Model Construction



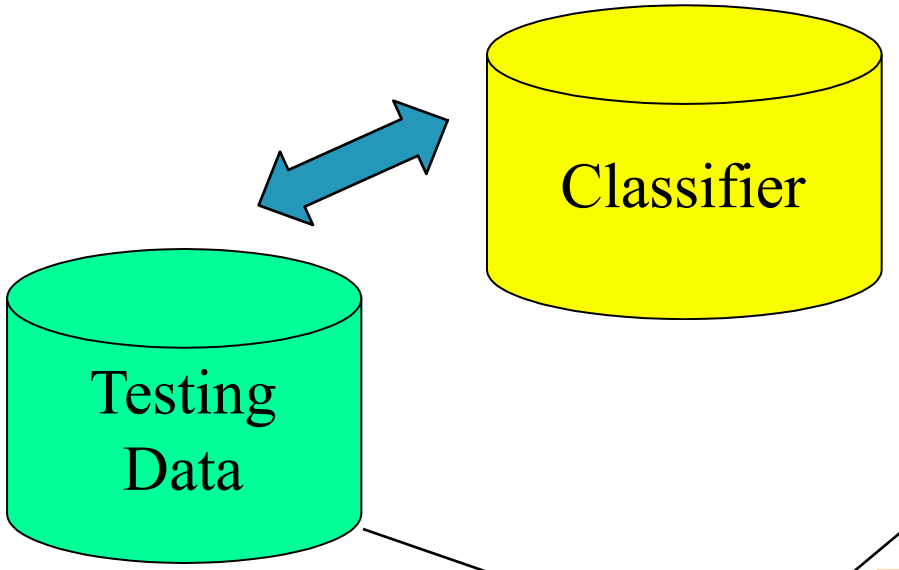




NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = ‘professor’
OR years > 6
THEN tenured = ‘yes’

Classification Process (2): Use the Model in Prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

(Jeff, Professor, 4)

Tenured?

Yes

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

Prediction vs. Classification

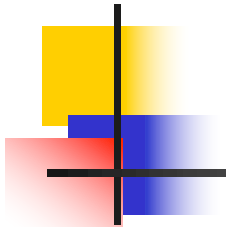
- Prediction
  - Assess the class of an unlabeled sample
  - Assess the value (or range) of an attribute of a given sample
- Classification and regression – two major types of prediction
  - Classification
    - Predict discrete or nominal values
  - Regression (commonly referred to as Prediction)
    - Predict continuous or ordered values

Classification and Prediction

- What is classification? What is prediction?
- **Classification by decision tree induction**
- Bayesian Classification
- Prediction
- Classification accuracy

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
- Decision tree generation consists of two phases
  - Tree construction
    - At start, all the training examples (samples) are at the root
    - Partition examples recursively based on selected attributes
  - Tree pruning
    - Identify and remove branches that reflect noise or outliers
      - To improve classification accuracy on unseen data

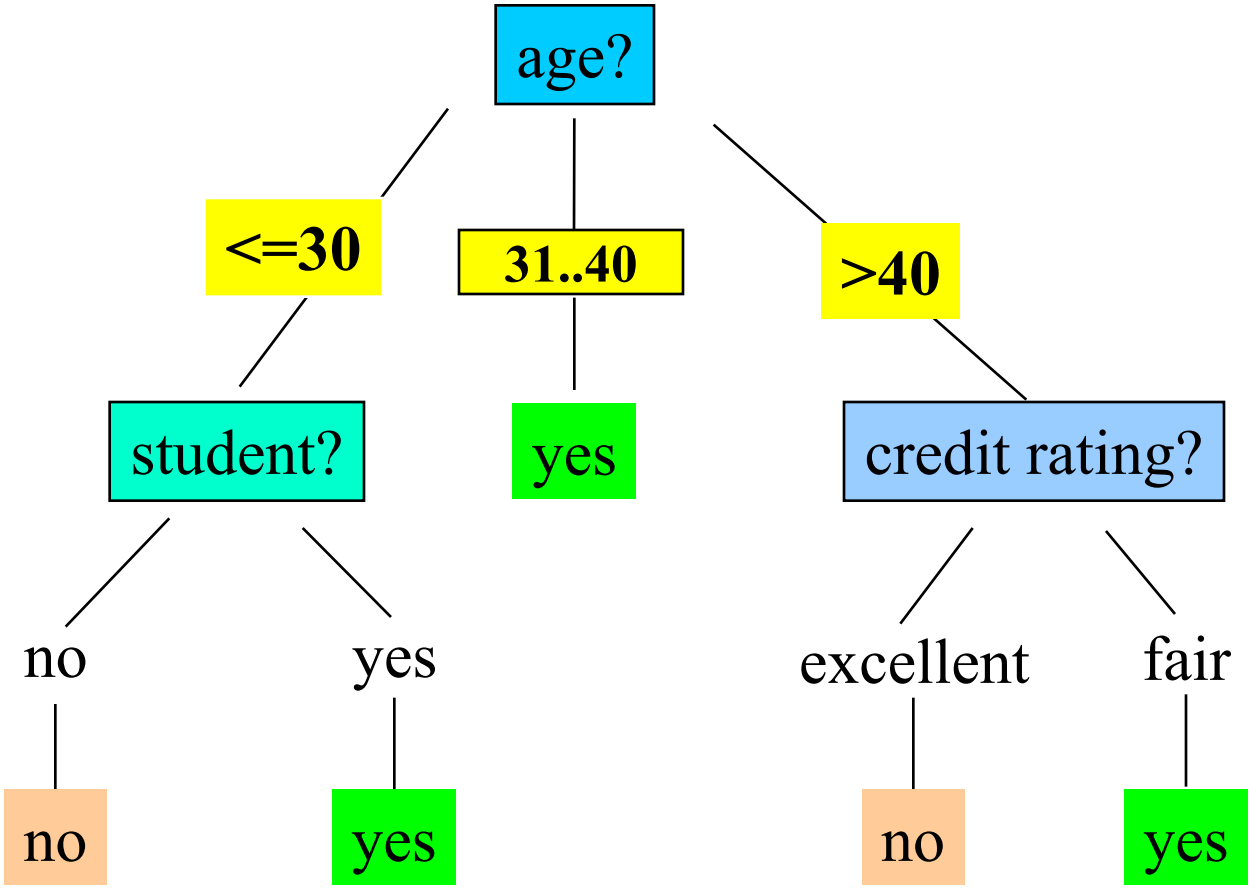


- Use of decision tree: Classifying an unknown sample
  - Test the attribute values of the sample against the decision tree
  - Trace the path from the root to a leaf node
    - The leaf node shows the class prediction for that sample

Training Dataset

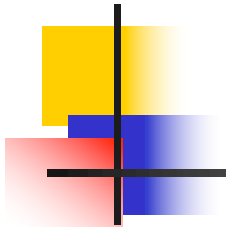
age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent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

Output: A Decision Tree for "*buys_computer*"



Algorithm for Decision Tree Induction

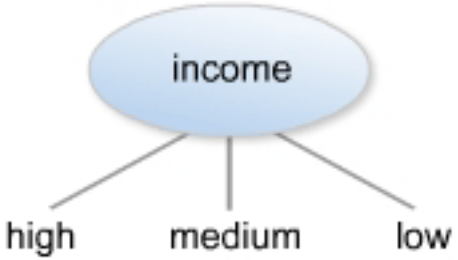
- Basic algorithm (a greedy algorithm)
  - Tree is constructed in a top-down recursive divide-and-conquer manner
    - ID3 – decision tree induction algorithm
  - At start, all the training samples are at the root
  - Attributes are categorical (if continuous-valued, they are discretized in advance)
  - 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)



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

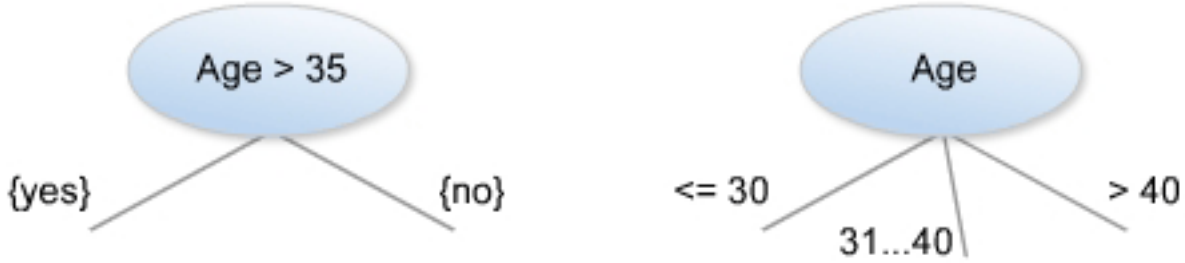
How to Split an Attribute?

- Multi-way Split



- Binary Split

- Numeric attribute



Attribute Selection Measure

- Information gain (ID3/C4.5)
  - All attributes are assumed to be categorical
  - Can be modified for continuous-valued attributes
- Gini index
  - All attributes are assumed continuous-valued
  - Assume there exist several possible split values for each attribute
  - May need other tools, such as clustering, to get the possible split values
  - Can be modified for categorical attributes

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 samples *S* contain *p* elements of class *P* and *n* elements of class *N*
  - The amount of information, needed to decide if an arbitrary sample 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 that has *v* distinct values, a set *S* will be partitioned into sets $\{S_1, S_2, \ldots, S_v\}$
  - If S_i contains p_i samples of *P* and n_i samples 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

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

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

$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.69$$

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

= 0.25

Similarly

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

Information Gain Example

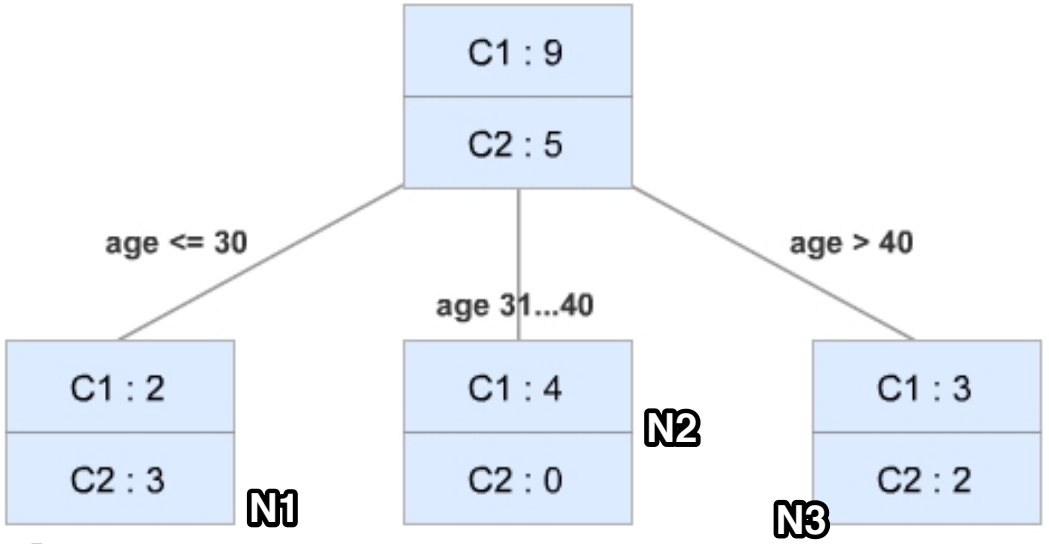
- Before Split

P(C1) : P(BuysComputer = yes) = 9/14
P(C2) : P(BuysComputer = no) = 5/14

- Entropy of the Total Data Set

$$Info(D) = -\frac{9}{14}\log(\frac{9}{14}) - \frac{5}{14}\log(\frac{5}{14}) = -0.643 * -0.637 - 0.357 * -1.485 = 0.94$$

- Split on Age



$$Info(N1) = -\frac{2}{5}\log(\frac{2}{5}) - \frac{3}{5}\log(\frac{3}{5}) = -0.4 * -1.322 - 0.6 * -0.737 = 0.971$$

$$Info(N2) = -\frac{4}{4}\log(\frac{4}{4}) - \frac{0}{4}\log(\frac{0}{4}) = -1 * 0 - 0 * \text{inf} = 0$$

$$Info(N3) = -\frac{3}{5}\log(\frac{3}{5}) - \frac{2}{5}\log(\frac{2}{5}) = -0.6 * -0.737 - 0.4 * -1.322 = 0.971$$

The combined expected information for this split is then:

$$Info_{age}(N) = \frac{5}{14} Info(N1) + \frac{4}{14} Info(N2) + \frac{5}{14} Info(N3) = 0.694$$

The information gain associated with this split is

$$Gain(age) = Info(D) - Info_{age}(N) = 0.94 - 0.694 = 0.246$$

Gini Index

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

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

  where p_j is the relative frequency of class *j* in *T*.
- If a data set *T* is split into two subsets T_1 and T_2 with sizes N_1 and N_2 respectively, the *gini* index of the split data contains examples from *n* classes, the *gini* index *gini*(*T*) is defined as

$$gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

- The attribute provides the smallest $gini_{split}(T)$ is chosen to split the node (*need to enumerate all possible splitting points for each attribute*).

Gini Example

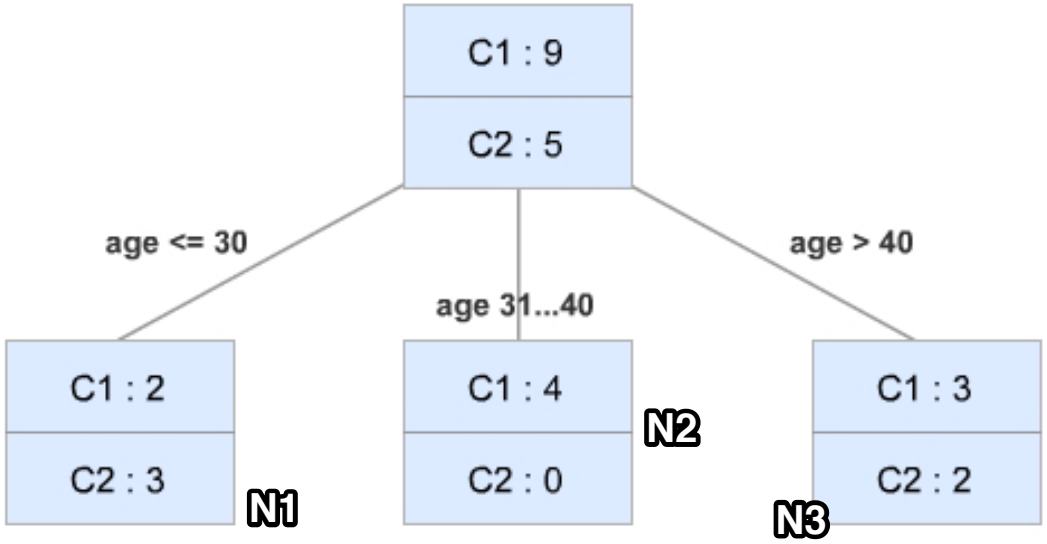
- Before Split

P(C1) : P(BuysComputer = yes) = 9/14
P(C2) : P(BuysComputer = no) = 5/14

- Gini of the Total Data Set

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

Split on Age



- For the child node N1, P(C1) = 2/5 and P(C2) = 3/5
  - Hence, Gini(N1) = $1 - (2/5)^2 - (3/5)^2 = 0.48$
- For the child node N2, P(C1) = 4/4 and P(C2) = 0/4
  - Hence, Gini(N2) = $1 - (1)^2 - (0)^2 = 0$
- For the child node N3, P(C1) = 3/5 and P(C2) = 2/5
  - Hence, Gini(N3) = $1 - (3/5)^2 - (2/5)^2 = 0.48$
- Now, the weighted Gini for the attribute "age" is:
  - Gini(age) = (5/14) * Gini(N1) + (4/14) * Gini(N2) + (5/14) * Gini(N3) = 0.343
- The information gain association with this split is
  - Gain(age) = Gini(Before split) – Gini(age) = 0.459 – 0.343 = 0.116

Extracting Classification Rules from Trees

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to a leaf
- Each attribute-value pair along a path forms a conjunction
- The leaf node holds the class prediction
- Rules are easier for humans to understand
- Example

IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"

IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

IF *age* = "31…40" THEN *buys_computer* = "*yes*"

IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*no*"

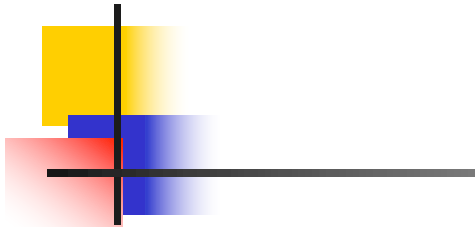
IF *age* = ">40" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*yes*"

Bayesian Classification

- Statistical classifiers
  - Predict class membership probabilities
    - Probability that a given sample belongs to a particular class
  - Naïve Bayesian classifier
    - Effect of an attribute value on a given class is independent of the values of the other attributes
      - Class conditional independence
      - Computationally simple
      - Comparable in performance with decision tree classifiers
  - Bayesian belief networks
    - Allow representation of dependencies among attributes

Bayesian Theorem

- X – a data sample whose class label is unknown
- H – a hypothesis that the data sample X belongs to a specified class C
- To determine P(H | X)
  - The probability that the hypothesis H holds given the observed data sample X
  - Also called the posterior probability (a posteriori probability) of H conditioned on X
- P(H) – prior probability (a priori probability)
- Example: X (red and round); H (X is an apple)
  - P(H|X) – confidence that X is an apple given X is read and round
  - P(H) – probability that any given data sample is an apple



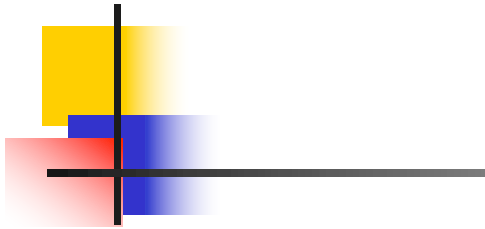
- P(X|H) – posterior probability of X conditioned on H
  - Probability that X is red and round given that we know that X is an apple
- P(X) – prior probability of X
  - Probability that a data sample is red and round
- Estimate P(X), P(H), and P(X|H) from the given data
- Using Bayes theorem

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)}$$

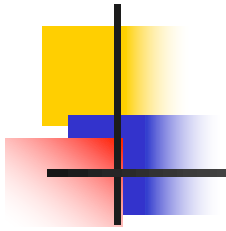
Naïve Bayesian Classification

- Given *n* attributes A_1, A_2, …, A_n
- Each data sample is an n-dimensional vector
  - $X = (x_1, x_2, \ldots, x_n)$
- Given m classes, C_1, C_2, …, C_m
- Given an unknown data sample X
  - Predict that X belongs to the class having the highest posterior probability, conditioned on X
- X belongs to class C_i if and only if
  - $P(C_i \mid X) > P(C_j \mid X)$ for all $1 \leq j \leq m, j \neq i$

- By Bayes theorem $$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$$
- P(X) is constant for all classes
- So, maximize $P(X \mid C_i)P(C_i)$
- If the class prior probabilities are not known
  - Assume that the classes are equally likely
    - $P(C_1) = P(C_2) = \ldots = P(C_m)$
  - So, maximize $P(X \mid C_i)$
- Otherwise, estimate $P(C_i) = s_i / s$,
  - s_i is the number of training samples of class C_i
  - s is the total number of training samples



- How to compute $P(X|C_i)$
  - Computationally expensive for datasets with many attributes
  - Assuming class conditional independence
    - No dependence relationships among attributes
    - $P(X \mid C_i) = \prod P(x_k \mid C_i)$
      - $P(x_1|C_i)$, $P(x_2|C_i)$, …, $P(x_n|C_i)$ are estimated from the training samples
    - If A_k is categorical, then $P(x_k|C_i) = s_{ik} / s_i$
      - s_{ik} is the number of training samples of class C_i having the value x_k for A_k
      - s_i is the number of training samples belonging to C_i



- Finally, the unknown sample X is assigned to class C if and only if
  - $P(X \mid C_i)P(C_i) > P(X \mid C_j)\ P(C_j)$ for all $1 \leq j \leq m, j \neq I$
- Example
  - Attributes
    - age, income, student, credit_rating
  - Class label attribute
    - buys_computer
      - Two distinct values {yes, no}
  - Class C1 – corresponds to the class buys_computer = “yes”
  - Class C2 – corresponds to the class buys_computer = “no”

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

- To classify the unknown sample
  - X = (age = "<=30", income="medium", student="yes", credit_rating="fair")
- To do
  - Find maximum of $P(X|C_1)P(C_1)$ and $P(X|C_2)P(C_2)$
- Prior probabilities of each class
  - $P(C_1)$= P(buys_computer="yes") = 9/14 = 0.643
  - $P(C_2)$ = P(buys_computer="no") = 5/14 = 0.357

- Conditional probabilities $P(x_k|C_i)$
  - P(age="<=30" | buys_computer="yes") = 2/9 = 0.222
  - P(age="<=30" | buys_computer="no") = 3/5 = 0.600
  - P(income="medium" | buys_computer="yes") = 4/9 = 0.444
  - P(income="medium" | buys_computer="no") = 2/5 = 0.400
  - P(student="yes" | buys_computer="yes") = 6/9 = 0.667
  - P(student="yes" | buys_computer="no") = 1/5 = 0.200
  - P(credit_rating="fair" | buys_computer="yes") = 6/9 = 0.667
  - P(credit_rating="fair" | buys_computer="no") = 2/5 = 0.400
- So,
  - P(X| buys_computer="yes") = 0.222 * 0.444 * 0.667 * 0.667 = 0.044
  - P(X| buys_computer="no") = 0.6 * 0.4 * 0.2 * 0.4 = 0.019
- Hence
  - P(X| buys_computer="yes") P(buys_computer="yes") = 0.044 * 0.643 = 0.028
  - P(X| buys_computer="no") P(buys_computer="no") = 0.019 * 0.357 = 0.007
- Prediction: **buys_computer = "yes"**

Naive Bayesian Classifier (II)

- Given a training set, we can compute the probabilities

Outlook	P	N
sunny	2/9	3/5
overcast	4/9	0
rain	3/9	2/5
Tempreature		
hot	2/9	2/5
mild	4/9	2/5
cool	3/9	1/5

Humidity	P	N
high	3/9	4/5
normal	6/9	1/5
Windy		
true	3/9	3/5
false	6/9	2/5

Play-tennis example: estimating $P(x_i|C)$

Outlook	Temperature	Humidity	Windy	Class
sunny	hot	high	false	N
sunny	hot	high	true	N
overcast	hot	high	false	P
rain	mild	high	false	P
rain	cool	normal	false	P
rain	cool	normal	true	N
overcast	cool	normal	true	P
sunny	mild	high	false	N
sunny	cool	normal	false	P
rain	mild	normal	false	P
sunny	mild	normal	true	P
overcast	mild	high	true	P
overcast	hot	normal	false	P
rain	mild	high	true	N

P(p) = 9/14
P(n) = 5/14

Outlook	
P(sunny|p) = 2/9	**P(sunny|n) = 3/5**
P(overcast|p) = 4/9	**P(overcast|n) = 0**
P(rain|p) = 3/9	**P(rain|n) = 2/5**
temperature	
P(hot|p) = 2/9	**P(hot|n) = 2/5**
P(mild|p) = 4/9	**P(mild|n) = 2/5**
P(cool|p) = 3/9	**P(cool|n) = 1/5**
humidity	
P(high|p) = 3/9	**P(high|n) = 4/5**
P(normal|p) = 6/9	**P(normal|n) = 2/5**
windy	
P(true|p) = 3/9	**P(true|n) = 3/5**
P(false|p) = 6/9	**P(false|n) = 2/5**

Play-tennis example: classifying X

- An unseen sample X = <rain, hot, high, false>

- P(X|p)·P(p) = P(rain|p)·P(hot|p)·P(high|p)·P(false|p)·P(p) = 3/9·2/9·3/9·6/9·9/14 = 0.010582
- P(X|n)·P(n) = P(rain|n)·P(hot|n)·P(high|n)·P(false|n)·P(n) = 2/5·2/5·4/5·2/5·5/14 = 0.018286

- Sample X is classified in class n (don't play)

Avoiding the Zero-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be **non-zero**. Otherwise, the predicted prob. will be zero

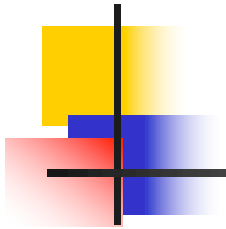
$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)$$

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income= medium (990), and income = high (10)
- Use **Laplacian correction** (or Laplacian estimator)
  - *Adding 1 to each case*

    Prob(income = low) = 1/1003

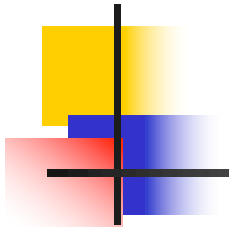
    Prob(income = medium) = 991/1003

    Prob(income = high) = 11/1003
  - The "corrected" prob. estimates are close to their "uncorrected" counterparts



Classification and Prediction

- What is classification? What is prediction?
- Classification by decision tree induction
- Bayesian Classification
- **Prediction**
- Classification accuracy



What Is Prediction?

- Prediction is similar to classification
  - First, construct a model
  - Second, use model to predict unknown value
    - Major method for prediction is regression
      - Linear and multiple regression
      - Non-linear regression
- Prediction is different from classification
  - Classification refers to predict categorical class label
  - Prediction models continuous-valued functions

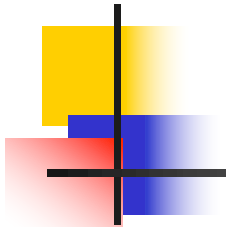
Regress Analysis and Log-Linear Models in Prediction

- Linear regression:
  - Data modeled using a straight line
    - Model a response variable (Y) as a linear function of a predictor variable (X)
      - $Y = \alpha + \beta X$
    - Two parameters , α and β, specify the line and are to be estimated by using the data at hand.
      - using the least squares criterion to minimize the error between the actual data and the estimate of the line
    - Given *s* samples of data (x_1, y_1), (x_2, y_2), …, (x_s, y_s),
      -

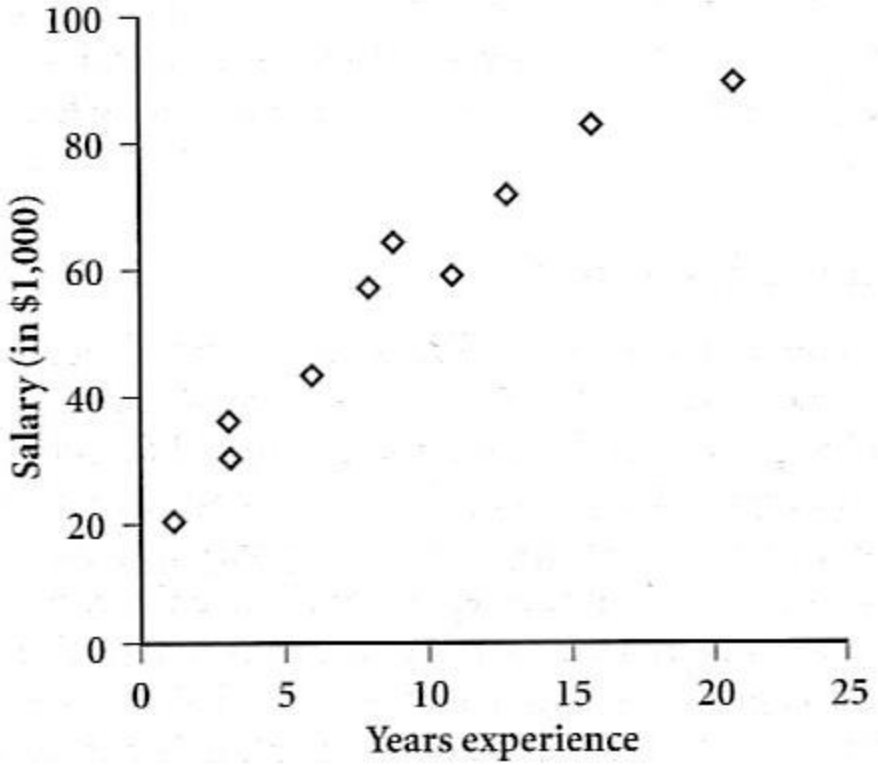
$$\beta = \frac{\sum_{i=1}^{s} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{s} (xi - \bar{x})^2}$$

$$\alpha = \bar{y} - \beta\bar{x}$$

$\bar{x}$ is the average of $x_1, x_2, \ldots, x_s$ and $\bar{y}$ is the average of $y_1, y_2, \ldots, y_s$.



- Example
  - X (years of experience)
    - (3,8,9,13,3,6,11,21,1,16)
  - Y (Salary in $1000s)
    - (30,57,64,72,36,43,59,90,20,93)
  - $\beta = 3.5$, $\alpha = 23.6$
  - $Y = 23.6 + 3.5\ X$
  - Predict the salary with 10 years of experience to be $58.6K


100
80
60
40
20
0
Salary (in $1,000)
0
5
10
15
20
25
Years experience

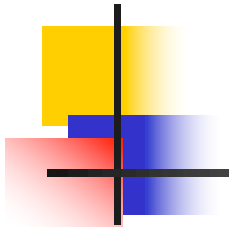
- Multiple regression: $Y = \alpha + \beta_1 X_1 + \beta_2 X_2$
  - More than one predictor variable
- Nonlinear regression
  - $Y = \alpha + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$
  - transformed into linear form and solved by the method of least squares

Classification and Prediction

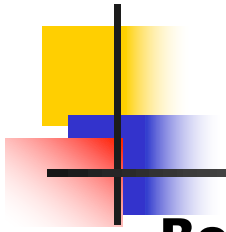
- What is classification? What is prediction?
- Classification by decision tree induction
- Bayesian Classification
- Prediction
- **Classification accuracy**

Classification Accuracy: Estimating Error Rates

- From randomly sampled portions of the given data
- **Holdout** method
- Partition the data into two independent sets
  - training set (2/3 of the data),
  - test set(1/3 of the data)
- Use the training set to derive the classifier
- Estimate the accuracy using the test set
  - used for data set with large number of samples
- Random subsampling
  - The holdout method is repeated *k* times
    - Overall accuracy = average of the accuracies from each iteration



- **k-fold Cross-validation**
  - divide the data set randomly into *k* subsamples or folds
  - use *k-1* subsamples as training data and one sub-sample as test data
    - Training and testing is repeated *k* times
  - for data set with moderate size
- **Stratified Cross-validation**
  - Class distribution of the samples in each fold is approximately the same as that in the initial data



- **Bootstrapping**
  - Sample the given training instances uniformly with replacement
    - Leave one out
    - Like the *k*-fold cross-validation with *k* set to *s*, the number of initial samples
  - for small size data
- Recommended method to estimate classifier accuracy
  - 10-fold cross validation
    - Low bias and variance