Classification and Prediction

Classification and Prediction

- What is classification? What is prediction?
- Issues regarding classification and prediction
- Classification by decision tree induction
- Bayesian Classification
- Other Classification Methods
- Prediction

What is Bayesian Classification?

- Bayesian classifiers are statistical classifiers
- For each new sample they provide a probability that the sample belongs to a class (for all classes)
- Example:
  - sample John (age=27, income=high, student=no, credit_rating=fair)
  - P(John, buys_computer=yes) = 20%
  - P(John, buys_computer=no) = 80%

Bayesian Classification: Why?

- Probabilistic learning: Calculate explicit probabilities for hypothesis, among the most practical approaches to certain types of learning problems
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.
- Probabilistic prediction: Predict multiple hypotheses, weighted by their probabilities
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayes' Theorem

- Given a data sample *X*, the *posteriori probability of a hypothesis h*, *P(h|X)* follows the Bayes theorem

$$P(h|X) = \frac{P(X|h)P(h)}{P(X)}$$

- Example:
  - Given that for John (*X*) has
    - age=27, income=high, student=no, credit_rating=fair
  - We would like to find P(*h*):
    - P(John, buys_computer=yes)
    - P(John, buys_computer=no)
- For P(John, buys_computer=yes) we are going to use:
  - P(age=27 $\wedge$ income=high $\wedge$ student=no $\wedge$ credit_rating=fair) given that P(buys_computer=yes)
  - P(buys_computer=yes)
  - P(age=27 $\wedge$ income=high $\wedge$ student=no $\wedge$ credit_rating=fair)
- Practical difficulty: require initial knowledge of many probabilities, significant computational cost

Naïve Bayesian Classifier

- A simplified assumption: attributes are conditionally independent:

$$P(C_j | X) = P(C_j)\prod_{i=1}^{n} P(v_i | C_j)$$

- Notice that the class label C_j plays the role of the hypothesis.
- The denominator is removed because the probability of a data sample *P(X)* is constant for all classes.
- Also, the probability $P(X|C_j)$ of a sample X given a class C_j is replaced by:
  - $P(X|C_j) = \prod P(v_i|C_j)$, $X = v_1 \wedge v_2 \wedge \ldots \wedge v_n$
- This is the naive hypothesis (attribute independence assumption)

Naïve Bayesian Classifier

- Example:
  - Given that for John (*X*)
    - age=27, income=high, student=no, credit_rating=fair
  - P(John, buys_computer=yes) =
    P(buys_computer=yes)*
    P(age=27|buys_computer=yes)*
    P(income=high |buys_computer=yes)*
    P(student=no |buys_computer=yes)*
    P(credit_rating=fair |buys_computer=yes)
- Greatly reduces the computation cost, by only counting the class distribution.
- Sensitive to cases where there are strong correlations between attributes
  - E.g. P(age=27 $\wedge$ income=high) >> P(age=27)*P(income=high)

play tennis?

Naive Bayesian Classifier Example

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

Naive Bayesian Classifier Example

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

9

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

5

Naive Bayesian Classifier Example

- Given the training set, we 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

- We also have the probabilities
  - P = 9/14
  - N = 5/14

Naive Bayesian Classifier Example

- The classification problem is formalized using a-posteriori probabilities:
- $P(C|X)$ = prob. that the sample tuple $X=<x_1,\ldots,x_k>$ is of class C.
- E.g. P(class=N | outlook=sunny,windy=true,…)
- Assign to sample X the class label C such that P(C|X) is maximal
- Naïve assumption: attribute independence

$$P(x_1,\ldots,x_k|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$$

Naive Bayesian Classifier Example

- To classify a new sample X:
  - outlook = sunny
  - temperature = cool
  - humidity = high
  - windy = false
- Prob(P|X) = Prob(P)*Prob(sunny|P)*Prob(cool|P)* Prob(high|P)*Prob(false|P) = 9/14*2/9*3/9*3/9*6/9 = 0.01
- Prob(N|X) = Prob(N)*Prob(sunny|N)*Prob(cool|N)* Prob(high|N)*Prob(false|N) = 5/14*3/5*1/5*4/5*2/5 = 0.013
- Therefore X takes class label N

Naive Bayesian Classifier Example

- Second example X = <rain, hot, high, false>

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

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

Categorical and Continuous Attributes

- Naïve assumption: attribute independence

$$P(x_1,\ldots,x_k|C) = P(x_1|C)\cdot\ldots\cdot P(x_k|C)$$

- If i-th attribute is categorical:
  $P(x_i|C)$ is estimated as the relative freq of samples having value x_i as i-th attribute in class C
- If i-th attribute is continuous:
  $P(x_i|C)$ is estimated thru a Gaussian density function
- Computationally easy in both cases

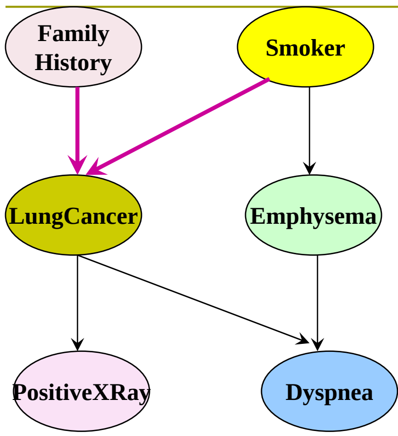
The independence hypothesis…

- … makes computation possible
- … yields optimal classifiers when satisfied
- … but is seldom satisfied in practice, as attributes (variables) are often correlated.
- Attempts to overcome this limitation:
  - Bayesian networks, that combine Bayesian reasoning with causal relationships between attributes
  - Decision trees, that reason on one attribute at the time, considering most important attributes first

Bayesian Belief Networks (I)

- A directed acyclic graph which models dependencies between variables (values)
- If an arc is drawn from node Y to node Z, then
  - Z depends on Y
  - Z is a child (descendant) of Y
  - Y is a parent (ancestor) of Z
- Each variable is conditionally independent of its nondescendants given its parents

Bayesian Belief Networks (II)



Bayesian Belief Networks

	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

The conditional probability table for the variable LungCancer

Bayesian Belief Networks (III)

- Using Bayesian Belief Networks:
  - $P(v_1, \ldots, v_n) = \prod P(v_i/\text{Parents}(v_i))$
- Example:
  - P(LC = yes ∧ FH = yes ∧ S = yes) =

    P(FH = yes)* P(S = yes)*

    P(LC = yes|FH = yes ∧ S = yes) =

    P(FH = yes)* P(S = yes)*0.8

Bayesian Belief Networks (IV)

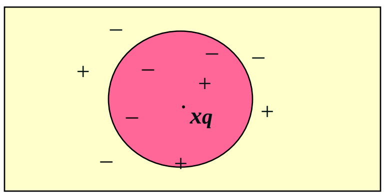
- Bayesian belief network allows a *subset* of the variables conditionally independent
- A graphical model of causal relationships
- Several cases of learning Bayesian belief networks
  - Given both network structure and all the variables: easy
  - Given network structure but only some variables
  - When the network structure is not known in advance

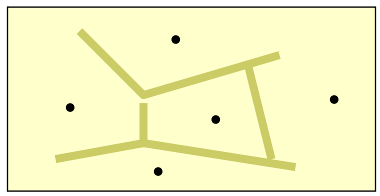
Instance-Based Methods

- Instance-based learning:
  - Store training examples and delay the processing ("lazy evaluation") until a new instance must be classified
- Typical approaches
  - *k*-nearest neighbor approach
    - Instances represented as points in a Euclidean space.
  - Locally weighted regression
    - Constructs local approximation
  - Case-based reasoning
    - Uses symbolic representations and knowledge-based inference

The *k*-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space.
- The nearest neighbor are defined in terms of Euclidean distance.
- The target function could be discrete- or real- valued.
- For discrete-valued function, the *k*-NN returns the most common value among the k training examples nearest to *xq*.
- Vonoroi diagram: the decision surface induced by 1-NN for a typical set of training examples.





Discussion on the *k*-NN Algorithm

- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the k neighbors according to their distance to the query point x_q
    - give greater weight to closer neighbors
  - Similarly, for real-valued target functions

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

- Robust to noisy data by averaging k-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes.
  - To overcome it, axes stretch or elimination of the least relevant attributes.

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

Predictive Modeling in Databases

- Predictive modeling: Predict data values or construct generalized linear models based on the database data.
- One can only predict value ranges or category distributions
- Method outline:
  - Minimal generalization
  - Attribute relevance analysis
  - Generalized linear model construction
  - Prediction
- Determine the major factors which influence the prediction
  - Data relevance analysis: uncertainty measurement, entropy analysis, expert judgement, etc.

Regress Analysis and Log-Linear Models in Prediction

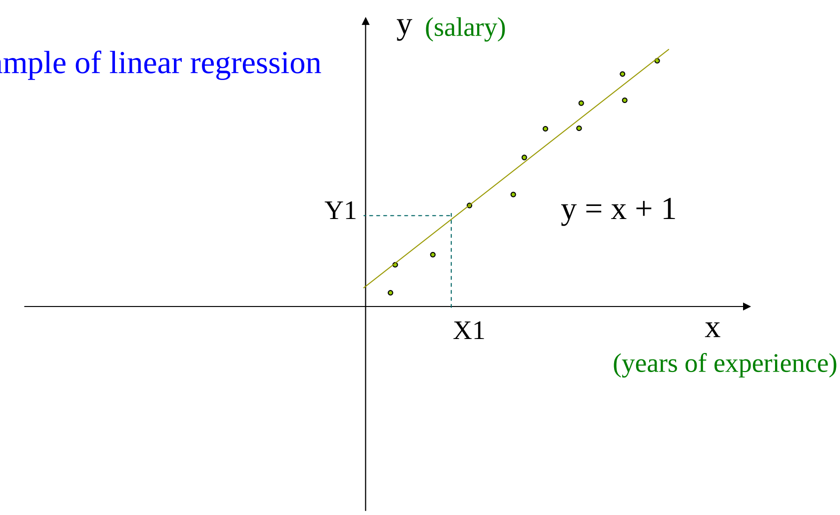
- <u>Linear regression</u>: Y = α + β X
  - Two parameters , α and β specify the line and are to be estimated by using the data at hand.
  - using the least squares criterion to the known values of $(x_1,y_1),(x_2,y_2),\ldots,(x_s,y_S)$:

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

- <u>Multiple regression</u>: Y = b0 + b1 X1 + b2 X2.
  - Many nonlinear functions can be transformed into the above. E.g., $Y=b0+b_1X+b_2X^2+b_3X^3$, $X1=X$, $X2=X^2$, $X3=X^3$
- <u>Log-linear models</u>:
  - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
  - Probability: $p(a, b, c, d) = \alpha ab\ \beta ac \chi ad\ \delta bcd$

Regression

Example of linear regression



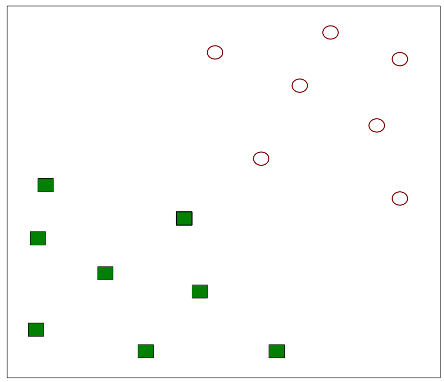
Boosting

- Boosting increases classification accuracy
  - Applicable to decision trees or Bayesian classifiers
- Learn a series of classifiers, where each classifier in the series pays more attention to the examples misclassified by its predecessor
- Boosting requires only linear time and constant space

Boosting Technique (II) — Algorithm

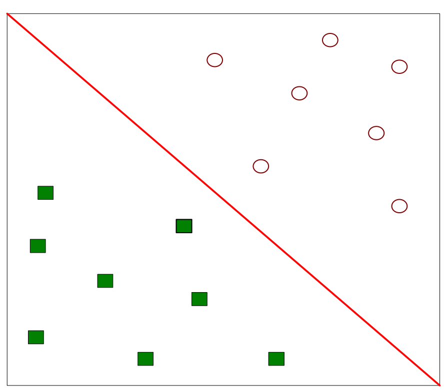
- Assign every example an equal weight *1/N*
- *For t = 1, 2, ..., T Do*
  - Obtain a hypothesis (classifier) $h^{(t)}$ under $w^{(t)}$
  - Calculate the error of *h(t)* and re-weight the examples based on the error
  - Normalize $w^{(t+1)}$ to sum to 1
- Output a weighted sum of all the hypothesis, with each hypothesis weighted according to its accuracy on the training set

Support Vector Machines



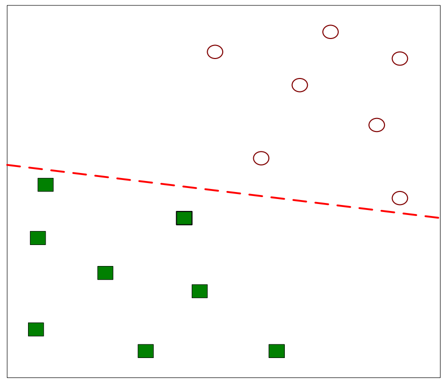
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



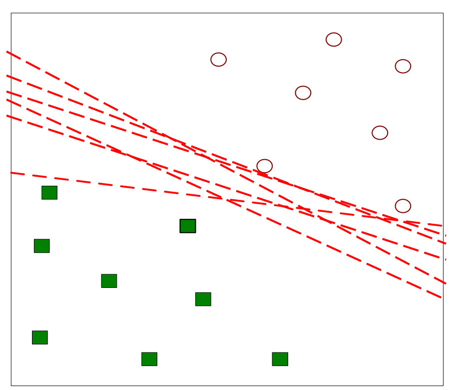
- One Possible Solution

Support Vector Machines



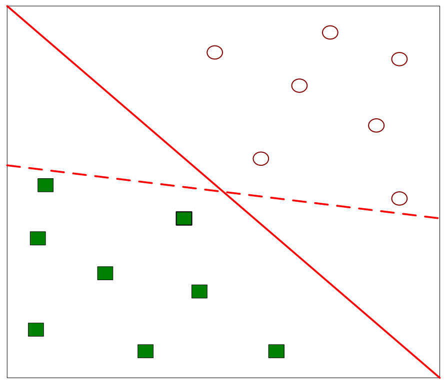
- Another possible solution

Support Vector Machines



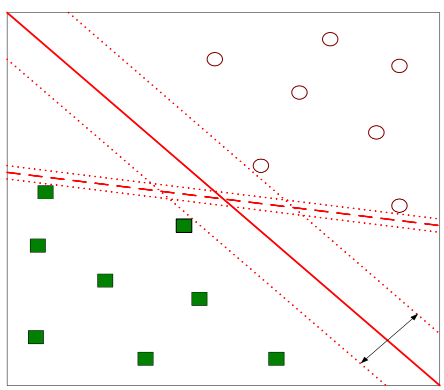
- Other possible solutions

Support Vector Machines



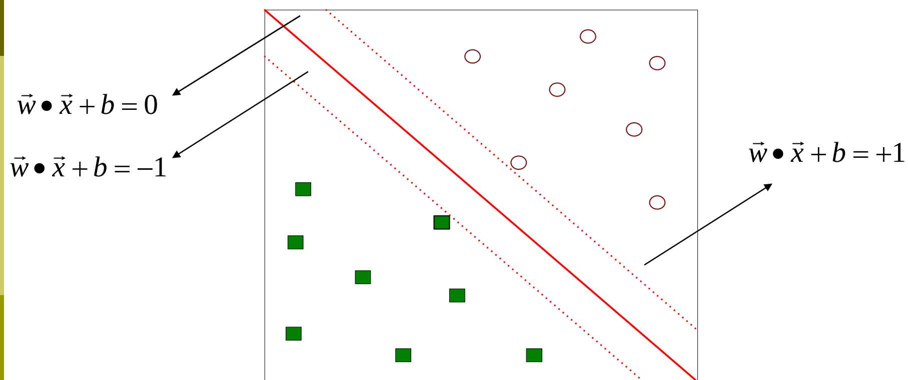
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Support Vector Machines

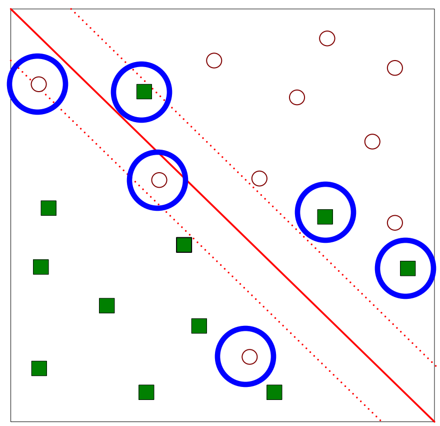
- We want to maximize: $\text{Margin} = \frac{2}{\|\vec{w}\|^2}$
  - Which is equivalent to minimizing: $L(w) = \frac{\|\vec{w}\|^2}{2}$
  - But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x}_i + b \leq -1 \end{cases}$$

    - This is a constrained optimization problem
      - Numerical approaches to solve it (e.g., quadratic programming)

Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

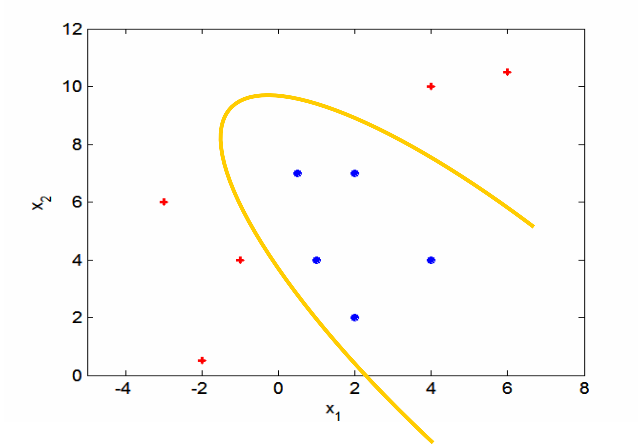
$$L(w) = \frac{\|\vec{w}\|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k\right)$$

    - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 - \xi_{\mathrm{i}} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 + \xi_{\mathrm{i}} \end{cases}$$

Nonlinear Support Vector Machines

- What if decision boundary is not linear?



Nonlinear Support Vector Machines

- Transform data into higher dimensional space