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)
  - \( P(\text{John, buys_computer=yes}) = 20\% \)
  - \( P(\text{John, buys_computer=no}) = 80\% \)

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 *posterior 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 $P(h)$:
    - $P(\text{John, buys_computer}=\text{yes})$
    - $P(\text{John, buys_computer}=\text{no})$
  - For $P(\text{John, buys_computer}=\text{yes})$ we are going to use:
    - $P(\text{age}=27 \land \text{income}=\text{high} \land \text{student}=\text{no} \land \text{credit_rating}=\text{fair})$ given that $P(\text{buys_computer}=\text{yes})$
    - $P(\text{buys_computer}=\text{yes})$
    - $P(\text{age}=27 \land \text{income}=\text{high} \land \text{student}=\text{no} \land \text{credit_rating}=\text{fair})$
  - 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:

$$ P(X|C_j) = \prod P(v_i|C_j), X=v_1 \land v_2 \land \ldots \land v_n $$

- This is the *naïve hypothesis* (attribute independence assumption)
Naïve Bayesian Classifier

Example:
- Given that for John (X)
  - age=27, income=high, student=no, credit_rating=fair
- \( P(\text{John, buys_computer=\text{yes}}) = P(\text{buys_computer=\text{yes}}) \times P(\text{age=27 | buys_computer=\text{yes}}) \times P(\text{income=high | buys_computer=\text{yes}}) \times P(\text{student=no | buys_computer=\text{yes}}) \times P(\text{credit_rating=fair | buys_computer=\text{yes}}) \)

- Greatly reduces the computation cost, by only counting the class distribution.
- Sensitive to cases where there are strong correlations between attributes
  - E.g. \( P(\text{age=27} \land \text{income=high}) \gg P(\text{age=27}) \times P(\text{income=high}) \)

---

**Naive Bayesian Classifier Example**

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>N</td>
</tr>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>N</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>N</td>
</tr>
<tr>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>P</td>
</tr>
<tr>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>N</td>
</tr>
<tr>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>rain</td>
<td>mild</td>
<td>normal</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>true</td>
<td>P</td>
</tr>
<tr>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>P</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>false</td>
<td>P</td>
</tr>
<tr>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>N</td>
</tr>
</tbody>
</table>
Naive Bayesian Classifier Example

Given the training set, we compute the probabilities:

- We also have the probabilities:
  - P = 9/14
  - N = 5/14
The classification problem is formalized using a\textit{-}\text{posteriori} probabilities:

\[ P(C|X) = \text{prob. that the sample tuple } X = \langle x_1, \ldots, x_k \rangle \text{ is of class } C. \]

E.g. \( P(\text{class}=\text{N} \mid \text{outlook}=\text{sunny}, \text{windy}=\text{true}, \ldots) \)

Assign to sample \( X \) the class label \( C \) such that \( P(C|X) \) is maximal

Naive assumption: attribute independence

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

To classify a new sample \( X \):
- outlook = sunny
- temperature = cool
- humidity = high
- windy = false

\[
\begin{align*}
\text{Prob}(P|X) &= \text{Prob}(P) \cdot \text{Prob}(\text{sunny}|P) \cdot \text{Prob}(\text{cool}|P) \cdot \\
& \quad \text{Prob}(\text{high}|P) \cdot \text{Prob}(\text{false}|P) = \\
& = 9/14 \cdot 2/9 \cdot 3/9 \cdot 3/9 \cdot 6/9 = 0.01 \\
\text{Prob}(N|X) &= \text{Prob}(N) \cdot \text{Prob}(\text{sunny}|N) \cdot \text{Prob}(\text{cool}|N) \cdot \\
& \quad \text{Prob}(\text{high}|N) \cdot \text{Prob}(\text{false}|N) = \\
& = 5/14 \cdot 3/5 \cdot 1/5 \cdot 4/5 \cdot 2/5 = 0.013 \\
\end{align*}
\]

Therefore \( X \) takes class label \( N \)
Naive Bayesian Classifier Example

- Second example \(X = \langle \text{rain, hot, high, false} \rangle\)

\[
P(X|p) \cdot P(p) = P(\text{rain}|p) \cdot P(\text{hot}|p) \cdot P(\text{high}|p) \cdot P(\text{false}|p) \cdot P(p) = \frac{3}{9} \cdot \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{6}{9} \cdot \frac{9}{14} = 0.010582
\]

\[
P(X|n) \cdot P(n) = P(\text{rain}|n) \cdot P(\text{hot}|n) \cdot P(\text{high}|n) \cdot P(\text{false}|n) \cdot P(n) = \frac{2}{5} \cdot \frac{2}{5} \cdot \frac{4}{5} \cdot \frac{2}{5} \cdot \frac{5}{14} = 0.018286
\]

- 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

Family History → LungCancer
   Smoker → LungCancer
   Emphysema
   Dyspnea

PositiveXRays

<table>
<thead>
<tr>
<th>(FH, S)</th>
<th>(FH, ~S)</th>
<th>(~FH, S)</th>
<th>(~FH, ~S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LC</td>
<td>0.8</td>
<td>0.5</td>
<td>0.7</td>
</tr>
<tr>
<td>~LC</td>
<td>0.2</td>
<td>0.5</td>
<td>0.3</td>
</tr>
</tbody>
</table>

The conditional probability table for the variable LungCancer

Bayesian Belief Networks (III)

- **Using** Bayesian Belief Networks:
  \[ P(v_1, ..., v_n) = \prod P(v_i/\text{Parents}(v_i)) \]

- **Example**:
  \[ P(LC = \text{yes} \land FH = \text{yes} \land S = \text{yes}) = \]
  \[ P(FH = \text{yes}) \times P(S = \text{yes}) \times 0.8 \]
  \[ P(LC = \text{yes}|FH = \text{yes} \land S = \text{yes}) = \]
  \[ P(FH = \text{yes}) \times P(S = \text{yes}) \times 0.8 \]
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 $x_q$.
- Voronoi diagram: the decision surface induced by 1-NN for a typical set of training examples.

\[ w = \frac{1}{d(x_q, x_i)^2} \]

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
  - 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

- **Linear regression**: \( Y = \alpha + \beta X \)
  - Two parameters, \( \alpha \) and \( \beta \), specify the line and are to be estimated by using the data at hand.
  - Using the least squares criterion to the known values of \((x_1,y_1),(x_2,y_2),\ldots,(x_S,y_S)\):
    \[
    \beta = \frac{\sum_{i=1}^{S}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{S}(x_i - \bar{x})^2}
    \]
    \[
    \alpha = \bar{y} - \beta \bar{x}
    \]

- **Multiple regression**: \( Y = b_0 + b_1 X_1 + b_2 X_2 \).
  - Many nonlinear functions can be transformed into the above. E.g., \( Y = b_0 + b_1 X + b_2 X^2 + b_3 X^3 \), \( X_1 = X \), \( X_2 = X^2 \), \( X_3 = X^3 \)

- **Log-linear models**:
  - 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

\[
Y = x + 1
\]

\( Y_1 \)

\( x \)

(years of experience)

\( Y \) (salary)
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

![Diagram showing support vector machines with different possible solutions.]

- Another possible solution

Support Vector Machines

![Diagram showing support vector machines with multiple possible solutions.]

- Other possible solutions
Support Vector Machines

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

Find hyperplane maximizes the margin => B1 is better than B2
Support Vector Machines

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \]
\[ \mathbf{w} \cdot \mathbf{x} + b = -1 \]
\[ \mathbf{w} \cdot \mathbf{x} + b = +1 \]

\[ f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b \geq 1 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b \leq -1 \end{cases} \]

\[ \text{Margin} = \frac{2}{||\mathbf{w}||^2} \]

---

Support Vector Machines

- **We want to maximize:** \[ \text{Margin} = \frac{2}{||\mathbf{w}||^2} \]
  - Which is equivalent to minimizing: \[ L(\mathbf{w}) = \frac{||\mathbf{w}||^2}{2} \]
  - But subjected to the following constraints: \[ f(\mathbf{x}_i) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \geq 1 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \end{cases} \]

- This is a constrained optimization problem
  - Numerical approaches to solve it (e.g., quadratic programming)
What if the problem is not linearly separable?

Introduce slack variables

Need to minimize:

\[ L(w) = \frac{||\vec{w}||^2}{2} + C \sum_{i=1}^{N} \xi_i \]

Subject to:

\[ f(\vec{x}_i) = \begin{cases} 
1 & \text{if } \vec{w} \cdot \vec{x}_i + b > 1 - \xi_i \\
-1 & \text{if } \vec{w} \cdot \vec{x}_i + b \leq -1 + \xi_i 
\end{cases} \]
Nonlinear Support Vector Machines

- What if decision boundary is not linear?

- Transform data into higher dimensional space