Classification fundamentals

Elena Baralis
Politecnico di Torino

Classification

- Objectives
  - prediction of a class label
  - definition of an interpretable model of a given phenomenon
Data mining: classification

**Classification**

- **Approaches**
  - decision trees
  - bayesian classification
  - classification rules
  - neural networks
  - k-nearest neighbours
  - SVM

**Classification**

- **Requirements**
  - accuracy
  - interpretability
  - scalability
  - noise and outlier management
Classification

- Applications
  - detection of customer propension to leave a company (churn or attrition)
  - fraud detection
  - classification of different pathology types
  - ...

Classification: definition

- Given
  - a collection of class labels
  - a collection of data objects labelled with a class label

- Find a descriptive profile of each class, which will allow the assignment of unlabeled objects to the appropriate class
Definitions

- Training set
  - Collection of labeled data objects used to learn the classification model

- Test set
  - Collection of labeled data objects used to validate the classification model

Classification techniques

- Decision trees
- Classification rules
- Association rules
- Neural Networks
- Naïve Bayes and Bayesian Networks
- k-Nearest Neighbours (k-NN)
- Support Vector Machines (SVM)
- ...
Evaluation of classification techniques

- **Accuracy**
  - quality of the prediction
- **Efficiency**
  - model building time
  - classification time
- **Scalability**
  - training set size
  - attribute number
- **Robustness**
  - noise, missing data
- **Interpretability**
  - model interpretability
  - model compactness

---

Decision trees

Elena Baralis
Politecnico di Torino
Data mining: classification

**Example of decision tree**

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Another example of decision tree**

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

There could be more than one tree that fits the same data!
Data mining: classification

Apply Model to Test Data

Start from the root of tree.

Refund

Yes

No

MarSt

Single, Divorced

Married

TaxInc

< 80K

> 80K

NO

YES

Test Data

Refund Marital Status Taxable Income Cheat
No Married 80K ?

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Apply Model to Test Data

Refund

Yes

No

MarSt

Single, Divorced

Married

TaxInc

< 80K

> 80K

NO

YES

Refund Marital Status Taxable Income Cheat
No Married 80K ?

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Apply Model to Test Data

Test Data

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Data mining: classification

Apply Model to Test Data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

Apply Model to Test Data

Assign Cheat to "No"

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Decision tree induction

- Many algorithms to build a decision tree
  - Hunt’s Algorithm (one of the earliest)
  - CART
  - ID3, C4.5, C5.0
  - SLIQ, SPRINT

General structure of Hunt’s algorithm

Basic steps
- If \( D_t \) contains records that belong to the same class \( y_t \)
  - then \( t \) is a leaf node labeled as \( y_t \)
- If \( D_t \) contains records that belong to more than one class
  - select the “best” attribute \( A \) on which to split \( D_t \) and label node \( t \) as \( A \)
  - split \( D_t \) into smaller subsets and recursively apply the procedure to each subset
- If \( D_t \) is an empty set
  - then \( t \) is a leaf node labeled as the default (majority) class, \( y_d \)

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Hunt’s algorithm

```
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  80K   Yes
```

Decision tree induction

- Adopts a greedy strategy
  - “Best” attribute for the split is selected locally at each step
    - not a global optimum
- Issues
  - Structure of test condition
    - Binary split versus multiway split
  - Selection of the best attribute for the split
  - Stopping condition for the algorithm
Structure of test condition

- Depends on attribute type
  - nominal
  - ordinal
  - continuous
- Depends on number of outgoing edges
  - 2-way split
  - multi-way split

Splitting 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

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
### Splitting 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

What about this split?

### Splitting on continuous attributes

- **Different techniques**
  - **Discretization** to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – discretize during tree induction
  - 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 find the best cut
    - more computationally intensive
Splitting on continuous attributes

(i) Binary split

- Taxable Income > 80K?
  - Yes
  - No

(ii) Multi-way split

- Taxable Income?
  - < 10K
  - [10K,25K)
  - [25K,50K)
  - [50K,80K]
  - > 80K

Selection of the best attribute

Before splitting:
10 records of class 0,
10 records of class 1

- Own Car?
  - Yes
  - No

- Car Type?
  - Family
  - Sport

- Luxury

- Student ID?
  - $c_0$
  - $c_1$
  - $c_{10}$
  - $c_{11}$

Which attribute (test condition) is the best?
Selection of the best attribute

- Attributes with \textit{homogeneous} class distribution are preferred
- Need a measure of node impurity

\begin{center}
\begin{tabular}{c|c|c|c}
 & \text{C0: 5} & \text{C1: 5} & \text{C0: 9} \text{ C1: 1} \\
\hline
\text{Non-homogeneous,} & & & \\
\text{high degree of impurity} & & & \\
\text{Homogeneous, low} & & & \\
\text{degree of impurity} & & & \\
\end{tabular}
\end{center}

Measures of node impurity

- Many different measures available
  - Gini index
  - Entropy
  - Misclassification error
- Different algorithms rely on different measures

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
How to find the best attribute

Before Splitting:

A?

Yes No

Node N1 Node N2

C0 N10 C0 N20
C1 N11 C1 N21

M1 M2

B?

Yes No

Node N3 Node N4

C0 N30 C0 N40
C1 N31 C1 N41

M3 M4

Gain = M0 - M12 vs M0 - M34

GINI impurity measure

- Gini Index for a given node t

\[ GINI(t) = 1 - \sum_{j} [p(j|t)]^2 \]

- \( p(j|t) \) is the relative frequency of class j at node t

- Maximum (1 - 1/nc) when records are equally distributed among all classes, implying higher impurity degree

- Minimum (0.0) when all records belong to one class, implying lower impurity degree

\[
\begin{array}{cccc}
\text{C1} & 0 & \text{C1} & 1 & \text{C1} & 2 & \text{C1} & 3 \\
\text{C2} & 6 & \text{C2} & 5 & \text{C2} & 4 & \text{C2} & 3 \\
\text{Gini}=0.000 & \text{Gini}=0.278 & \text{Gini}=0.444 & \text{Gini}=0.500
\end{array}
\]
Examples for computing GINI

\[ GINI(t) = 1 - \sum_j [p(j \mid t)]^2 \]

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

- P(C1) = 0/6 = 0  \( P(C2) = 6/6 = 1 \)
- Gini = 1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

- P(C1) = 1/6  \( P(C2) = 5/6 \)
- Gini = 1 - (1/6)^2 - (5/6)^2 = 0.278

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

- P(C1) = 2/6  \( P(C2) = 4/6 \)
- Gini = 1 - (2/6)^2 - (4/6)^2 = 0.444

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Splitting based on GINI

- Used in CART, SLIQ, SPRINT
- When a node p is split into k partitions (children), the quality of the split is computed as

\[ GINI_{\text{split}} = \sum_{i=1}^{k} \frac{n_i}{n} \cdot GINI(i) \]

where

- \( n_i \) = number of records at child i
- \( n \) = number of records at node p

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Computing GINI index: Boolean attribute

- Splits into two partitions
  - larger and purer partitions are sought for

![Diagram of a binary decision tree with a decision at node B?
Yes: Node N1, Gini(N1) = 0.408
No: Node N2, Gini(N2) = 0.32

\[
\text{Gini}(\text{split on } B) = \frac{7}{12} \times 0.408 + \frac{5}{12} \times 0.32 = 0.371
\]

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Computing GINI index: Categorical attribute

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

### CarType
- **Multi-way split**
  - | CarType | Count |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Family</td>
<td>1</td>
</tr>
<tr>
<td>Sports</td>
<td>2</td>
</tr>
<tr>
<td>Luxury</td>
<td>1</td>
</tr>
</tbody>
</table>
  - **Gini** = 0.393

### CarType
- **Two-way split**
  - (find best partition of values)
  - **(Sports, Luxury)**
    - | CarType | Count |
    |--------|-------|
    | C1     | 3     |
    | C2     | 2     |
    - **Gini** = 0.400

- **(Sports)**, **(Family, Luxury)**
  - | CarType | Count |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>2</td>
</tr>
<tr>
<td>C2</td>
<td>1</td>
</tr>
</tbody>
</table>
  - **Gini** = 0.419

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Computing GINI index: Continuous attribute

- Binary decision on one splitting value
  - Number of possible splitting values
    = Number of distinct values
  - Each splitting value \( v \) has a count matrix
    - class counts in the two partitions
      - \( A < v \)
      - \( A \geq v \)

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>50K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Entropy impurity measure (INFO)

- Entropy at a given node $t$

$$\text{Entropy}(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

$p(j \mid t)$ is the relative frequency of class $j$ at node $t$

- Maximum ($\log n_c$) when records are equally distributed among all classes, implying higher impurity degree
- Minimum (0.0) when all records belong to one class, implying lower impurity degree
- Entropy based computations are similar to GINI index computations

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Examples for computing entropy

$$\text{Entropy}(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

<table>
<thead>
<tr>
<th>C1</th>
<th>0</th>
<th>P(C1) = 0/6 = 0    P(C2) = 6/6 = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>6</td>
<td>Entropy = −0 \log 0 − 1 \log 1 = −0 − 0 = 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C1</th>
<th>1</th>
<th>P(C1) = 1/6       P(C2) = 5/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>5</td>
<td>Entropy = −(1/6) \log_2 (1/6) − (5/6) \log_2 (5/6) = 0.65</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C1</th>
<th>2</th>
<th>P(C1) = 2/6       P(C2) = 4/6</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2</td>
<td>4</td>
<td>Entropy = −(2/6) \log_2 (2/6) − (4/6) \log_2 (4/6) = 0.92</td>
</tr>
</tbody>
</table>

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
**Splitting Based on INFO**

- **Information Gain**

  \[ GAIN = \text{Entropy}(p) - \left( \sum_{i=1}^{k} \frac{n_i}{n} \text{Entropy}(i) \right) \]

  Parent Node, p is split into k partitions; 
  \( n_i \) is number of records in partition i

  - Measures reduction in entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN)

  - Used in ID3 and C4.5

  - Disadvantage: Tends to prefer splits yielding a large number of partitions, each small but pure

- **Gain Ratio**

  \[ \text{GainRatio} = \frac{GAIN}{\text{SplitINFO}} \]

  \[ \text{SplitINFO} = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n} \]

  Parent Node, p is split into k partitions 
  \( n_i \) is the number of records in partition i

  - Adjusts Information Gain by the entropy of the partitioning (SplitINFO). Higher entropy partitioning (large number of small partitions) is penalized

  - Used in C4.5

  - Designed to overcome the disadvantage of Information Gain

---

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Classification error impurity measure

Classification error at a node \( t \)

\[
\text{Error}(t) = 1 - \max_i P(i \mid t)
\]

Measures misclassification error made by a node

- Maximum \( (1 - 1/n_c) \) when records are equally distributed among all classes, implying least interesting information
- Minimum \( (0.0) \) when all records belong to one class, implying most interesting information

Examples for computing error

\[
\text{Error}(t) = 1 - \max_i P(i \mid t)
\]

<table>
<thead>
<tr>
<th>( C1 )</th>
<th>( C2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 0 )</td>
<td>( 6 )</td>
</tr>
</tbody>
</table>

\( P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1 \)

Error \( = 1 - \max (0, 1) = 1 - 1 = 0 \)

<table>
<thead>
<tr>
<th>( C1 )</th>
<th>( \text{1} )</th>
<th>( \text{1} )</th>
<th>( \text{1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{C2} )</td>
<td>( \text{5} )</td>
<td>( \text{5} )</td>
<td>( \text{5} )</td>
</tr>
</tbody>
</table>

\( P(C1) = 1/6 \quad P(C2) = 5/6 \)

Error \( = 1 - \max (1/6, 5/6) = 1 - 5/6 = 1/6 \)

<table>
<thead>
<tr>
<th>( C1 )</th>
<th>( \text{2} )</th>
<th>( \text{2} )</th>
<th>( \text{2} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{C2} )</td>
<td>( \text{4} )</td>
<td>( \text{4} )</td>
<td>( \text{4} )</td>
</tr>
</tbody>
</table>

\( P(C1) = 2/6 \quad P(C2) = 4/6 \)

Error \( = 1 - \max (2/6, 4/6) = 1 - 4/6 = 1/3 \)
Comparison among splitting criteria

For a 2-class problem

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination (to be discussed later)
Decision Tree Based Classification

- **Advantages**
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Accuracy is comparable to other classification techniques for many simple data sets

- **Disadvantages**
  - accuracy may be affected by missing data

---

Practical Issues of Classification

- Underfitting and overfitting
- Missing Values
- Costs of Classification

---

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Underfitting and Overfitting

Underfitting: when model is too simple, both training and test errors are large.

Overfitting due to Noise

Decision boundary is distorted by noise point.

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
How to address overfitting

- **Pre-Pruning (Early Stopping Rule)**
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using $\chi^2$ test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).


How to address overfitting

- **Post-pruning**
  - Grow decision tree to its entirety
  - Trim the nodes of the decision tree in a bottom-up fashion
  - If generalization error improves after trimming, replace sub-tree by a leaf node.
  - Class label of leaf node is determined from majority class of instances in the sub-tree

Handling missing attribute values

- Missing values affect decision tree construction in three different ways
  - Affects how impurity measures are computed
  - Affects how to distribute instance with missing value to child nodes
  - Affects how a test instance with missing value is classified

Other issues

- Data Fragmentation
- Search Strategy
- Expressiveness
- Tree Replication
Data mining: classification

Data fragmentation

- Number of instances gets smaller as you traverse down the tree

- Number of instances at the leaf nodes could be too small to make any statistically significant decision

Search strategy

- Finding an optimal decision tree is NP-hard

- The algorithm presented so far uses a greedy, top-down, recursive partitioning strategy to induce a reasonable solution

- Other strategies?
  - Bottom-up
  - Bi-directional

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Expressiveness

- Decision tree provides expressive representation for learning discrete-valued function
  - But they do not generalize well to certain types of Boolean functions
    - Example: parity function:
      - Class = 1 if there is an even number of Boolean attributes with truth value = True
      - Class = 0 if there is an odd number of Boolean attributes with truth value = True
    - For accurate modeling, must have a complete tree
  - Not expressive enough for modeling continuous variables
    - Particularly when test condition involves only a single attribute at-a-time

Decision boundary

- Border line between two neighboring regions of different classes is known as decision boundary
- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
### Oblique decision trees

- Test condition may involve multiple attributes
- More expressive representation
- Finding optimal test condition is computationally expensive

Class = +

Class = 

\[ x + y < 1 \]

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

### Tree replication

- Same subtree appears in multiple branches

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Rule-based classification

Elena Baralis
Politecnico di Torino

Rule-based classifier

- Classify records by using a collection of “if...then...” rules
- Rule: \((\text{Condition}) \rightarrow y\)
  - where
    - \text{Condition} is a conjunction of attributes
    - \(y\) is the class label
  - \(\text{LHS}:\) rule antecedent or condition
  - \(\text{RHS}:\) rule consequent
- Examples of classification rules
  - \((\text{Blood Type}=\text{Warm}) \land (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}\)
  - \((\text{Taxable Income} < 50K) \land (\text{Refund}=\text{Yes}) \rightarrow \text{Evade}=\text{No}\)

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Rule-based Classifier (Example)

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>python</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>salmon</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>whale</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>frog</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>amphibians</td>
</tr>
<tr>
<td>komodo</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>bat</td>
<td>warm</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>birds</td>
</tr>
<tr>
<td>cat</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>leopard shark</td>
<td>cold</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>turtle</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>reptiles</td>
</tr>
<tr>
<td>porcupine</td>
<td>warm</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>penguin</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>birds</td>
</tr>
<tr>
<td>porcupine</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>pel</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>salamander</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>amphibians</td>
</tr>
<tr>
<td>gila monster</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>platypus</td>
<td>warm</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>penguin</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>birds</td>
</tr>
<tr>
<td>platypus</td>
<td>warm</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
</tbody>
</table>

R1: $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{yes}) \rightarrow \text{Birds}$
R2: $(\text{Give Birth} = \text{no}) \land (\text{Live in Water} = \text{yes}) \rightarrow \text{Fishes}$
R3: $(\text{Give Birth} = \text{yes}) \land (\text{Blood Type} = \text{warm}) \rightarrow \text{Mammals}$
R4: $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{no}) \rightarrow \text{Reptiles}$
R5: $(\text{Live in Water} = \text{sometimes}) \rightarrow \text{Amphibians}$

Rule R1 covers a hawk = Bird
Rule R3 covers the grizzly bear = Mammal

Rule-based classification

- A rule $r$ covers an instance $\mathbf{x}$ if the attributes of the instance satisfy the condition of the rule

R1: $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{yes}) \rightarrow \text{Birds}$
R2: $(\text{Give Birth} = \text{no}) \land (\text{Live in Water} = \text{yes}) \rightarrow \text{Fishes}$
R3: $(\text{Give Birth} = \text{yes}) \land (\text{Blood Type} = \text{warm}) \rightarrow \text{Mammals}$
R4: $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{no}) \rightarrow \text{Reptiles}$
R5: $(\text{Live in Water} = \text{sometimes}) \rightarrow \text{Amphibians}$

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>hawk</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>?</td>
</tr>
<tr>
<td>grizzly bear</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>?</td>
</tr>
</tbody>
</table>

Rule R1 covers a hawk = Bird
Rule R3 covers the grizzly bear = Mammal
Rule-based classification

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds
R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes
R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals
R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles
R5: (Live in Water = sometimes) → Amphibians

A lemur triggers rule R3, so it is classified as a mammal
A turtle triggers both R4 and R5
A dogfish shark triggers none of the rules

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>lemur</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>?</td>
</tr>
<tr>
<td>turtle</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>?</td>
</tr>
<tr>
<td>dogfish shark</td>
<td>cold</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>?</td>
</tr>
</tbody>
</table>

Characteristics of rules

- Mutually exclusive rules
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered by at most one rule

- Exhaustive rules
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered by at least one rule
Data mining: classification

From decision trees to rules

Classification Rules

\[(\text{Refund}=\text{Yes}) \implies \text{No}\]

\[(\text{Refund}=\text{No}, \text{Marital Status} = \{\text{Single, Divorced}\}, \text{Taxable Income} < 80K) \implies \text{No}\]

\[(\text{Refund}=\text{No}, \text{Marital Status} = \{\text{Single, Divorced}\}, \text{Taxable Income} > 80K) \implies \text{Yes}\]

\[(\text{Refund}=\text{No}, \text{Marital Status} = \{\text{Married}\}) \implies \text{No}\]

Rules are mutually exclusive and exhaustive
Rule set contains as much information as the tree

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Rules can be simplified

Initial Rule: \((\text{Refund}=\text{No}) \land (\text{Status}=\text{Married}) \rightarrow \text{No}\)

Simplified Rule: \((\text{Status}=\text{Married}) \rightarrow \text{No}\)

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Effect of rule simplification

- Rules are no longer mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes

- Rules are no longer exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class

Ordered rule set

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a decision list

- When a test record is presented to the classifier
  - It is assigned to the class label of the highest ranked rule it has triggered
  - If none of the rules fired, it is assigned to the default class

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds
R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes
R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals
R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles
R5: (Live in Water = sometimes) → Amphibians
Building classification rules

- Direct Method
  - Extract rules directly from data
  - e.g.: RIPPER, CN2, Holte’s 1R

- Indirect Method
  - Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  - e.g.: C4.5rules

Advantages of rule-based classifiers

- As highly expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Associative classification

Elena Baralis
Politecnico di Torino

The classification model is defined by means of association rules

\[(\text{Condition}) \rightarrow y\]

- rule body is an itemset

Model generation

- Rule selection & sorting
  - based on support, confidence and correlation thresholds
- Rule pruning
  - Database coverage: the training set is covered by selecting topmost rules according to previous sort
Associative classification

- **Strong points**
  - interpretable model
  - higher accuracy than decision trees
    - correlation among attributes is considered
  - efficient classification
  - unaffected by missing data
  - good scalability in the training set size

- **Weak points**
  - rule generation may be slow
    - it depends on support threshold
  - reduced scalability in the number of attributes
    - rule generation may become unfeasible

---

Neural networks

Elena Baralis
Politecnico di Torino
Neural networks

- Inspired to the structure of the human brain
  - Neurons as elaboration units
  - Synapses as connection network

Structure of a neural network

From: Han, Kambe; "Data mining: Concepts and Techniques", Morgan Kaufmann 2006
Data mining: classification

Structure of a neuron

Input vector $x$  Weight vector $w$  Weighted sum  Activation function

$\sum \mu_k$

From: Han, Kamber, "Data mining: Concepts and Techniques", Morgan Kaufmann 2006

Construction of the neural network

- For each node, definition of
  - set of weights
  - offset value
    providing the highest accuracy on the training data
- Iterative approach on training data instances
Construction of the neural network

- **Base algorithm**
  - Initially assign random values to weights and offsets
  - Process instances in the training set one at a time
    - For each neuron, compute the result when applying weights, offset and activation function for the instance
    - Forward propagation until the output is computed
    - Compare the computed output with the expected output, and evaluate error
    - Backpropagation of the error, by updating weights and offset for each neuron
  - The process ends when
    - % of accuracy above a given threshold
    - % of parameter variation (error) below a given threshold
    - The maximum number of epochs is reached

Neural networks

- **Strong points**
  - High accuracy
  - Robust to noise and outliers
  - Supports both discrete and continuous output
  - Efficient during classification

- **Weak points**
  - Long training time
    - weakly scalable in training data size
    - complex configuration
  - Not interpretable model
    - application domain knowledge cannot be exploited in the model
Bayesian Classification

Elena Baralis
Politecnico di Torino

Bayes theorem

- Let C and X be random variables
  \[ P(C,X) = P(C|X) \cdot P(X) \]
  \[ P(C,X) = P(X|C) \cdot P(C) \]
- Hence
  \[ P(C|X) \cdot P(X) = P(X|C) \cdot P(C) \]
- and also
  \[ P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)} \]
Bayesian classification

- Let the class attribute and all data attributes be random variables
  - $C$ = any class label
  - $X = \langle x_1, \ldots, x_k \rangle$ record to be classified
- Bayesian classification
  - compute $P(C|X)$ for all classes
    - probability that record $X$ belongs to $C$
  - assign $X$ to the class with maximal $P(C|X)$
- Applying Bayes theorem
  \[ P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)} \]
  - $P(X)$ constant for all $C$, disregarded for maximum computation
  - $P(C)$ a priori probability of $C$
    \[ P(C) = \frac{N_c}{N} \]

Bayesian classification

- How to estimate $P(X|C)$, i.e. $P(x_1, \ldots, x_k|C)$?
- Naïve hypothesis
  \[ P(x_1, \ldots, x_k|C) = P(x_1|C) \cdot P(x_2|C) \cdot \ldots \cdot P(x_k|C) \]
  - statistical independence of attributes $x_1, \ldots, x_k$
  - not always true
    - model quality may be affected
- Computing $P(x_k|C)$
  - for discrete attributes
    \[ P(x_k|C) = \frac{|x_kC|}{N_c} \]
    - where $|x_kC|$ is number of instances having value $x_k$ for attribute $k$
      and belonging to class $C$
  - for continuous attributes, use probability distribution
- Bayesian networks
  - allow specifying a subset of dependencies among attributes
Bayesian classification: 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>

From: Han, Kamber; "Data mining: Concepts and Techniques", Morgan Kaufmann 2006
**Bayesian classification: Example**

- Data to be labeled
  \[ X = \langle \text{rain}, \text{hot}, \text{high}, \text{false} \rangle \]

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

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

---

**Support Vector Machines**

Elena Baralis

*Politecnico di Torino*
Support Vector Machines

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

From: Tan,Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Support Vector Machines

- One Possible Solution

From: Tan,Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Support Vector Machines

Another possible solution

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Support Vector Machines

Other possible solutions

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
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

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Nonlinear Support Vector Machines

- What if decision boundary is not linear?

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Nonlinear Support Vector Machines

- Transform data into higher dimensional space

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
K-Nearest Neighbor

Instance-Based Classifiers

Set of Stored Cases

<table>
<thead>
<tr>
<th>Atr1</th>
<th>........</th>
<th>AtrN</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Unseen Case

<table>
<thead>
<tr>
<th>Atr1</th>
<th>........</th>
<th>AtrN</th>
</tr>
</thead>
</table>

- Store the training records
- Use training records to predict the class label of unseen cases

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Instance Based Classifiers

- **Examples**
  - **Rote-learner**
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - **Nearest neighbor**
    - Uses $k$ “closest” points (nearest neighbors) for performing classification

Nearest-Neighbor Classifiers

- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of $k$, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify $k$ nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Definition of Nearest Neighbor

(a) 1-nearest neighbor  (b) 2-nearest neighbor  (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

1 nearest-neighbor

Voronoi Diagram

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance
    \[ d(p, q) = \sqrt{\sum_i (p_i - q_i)^2} \]

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, \( w = 1/d^2 \)

Choosing the value of k:
- If k is too small, sensitive to noise points
- If k is too large, neighborhood may include points from other classes
Nearest Neighbor Classification

- Scaling issues
  - Attribute domain should be normalized to prevent distance measures from being dominated by one of the attributes
  - Example: height [1.5m to 2.0m] vs. income [$10K to $1M]

- Problem with distance measures
  - High dimensional data
    - curse of dimensionality

Model evaluation

Elena Baralis
Politecnico di Torino
Model evaluation

- Methods for performance evaluation
  - Partitioning techniques for training and test sets
- Metrics for performance evaluation
  - Accuracy, other measures
- Techniques for model comparison
  - ROC curve

Methods for performance evaluation

- Objective
  - reliable estimate of performance
- Performance of a model may depend on other factors besides the learning algorithm
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006
Learning curve shows how accuracy changes with varying sample size.

- Learning curve requires a sampling schedule for creating the learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)

Effect of small sample size:
- Bias in the estimate
- Variance of estimate

Methods of estimation
- Partitioning labeled data in:
  - training set for model building
  - test set for model evaluation
- Several partitioning techniques
  - holdout
  - cross validation
- Stratified sampling to generate partitions
  - without replacement
- Bootstrap
  - Sampling with replacement
Holdout

- Fixed partitioning
  - reserve 2/3 for training and 1/3 for testing
- Appropriate for large datasets
  - may be repeated several times
    - repeated holdout

Cross validation

- Cross validation
  - partition data into k disjoint subsets (i.e., folds)
  - k-fold: train on k-1 partitions, test on the remaining one
    - repeat for all folds
  - reliable accuracy estimation, not appropriate for very large datasets
- Leave-one-out
  - cross validation for k=n
  - only appropriate for very small datasets
## Metrics for model evaluation

- Evaluate the predictive accuracy of a model
- Confusion matrix
  - binary classifier

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class=Yes</td>
</tr>
<tr>
<td>Class=Yes</td>
<td>a</td>
</tr>
<tr>
<td>Class=No</td>
<td>c</td>
</tr>
</tbody>
</table>

- a: TP (true positive)
- b: FN (false negative)
- c: FP (false positive)
- d: TN (true negative)


## Accuracy

- Most widely-used metric for model evaluation

\[
\text{Accuracy} = \frac{\text{Number of correctly classified objects}}{\text{Number of classified objects}}
\]

- Not always a reliable metric
Accuracy

For a binary classifier

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class=Yes</td>
<td>Class=Yes</td>
</tr>
<tr>
<td></td>
<td>a (TP)</td>
</tr>
<tr>
<td>Class=Yes</td>
<td>Class=No</td>
</tr>
<tr>
<td></td>
<td>b (FN)</td>
</tr>
<tr>
<td>Class=No</td>
<td>Class=No</td>
</tr>
<tr>
<td></td>
<td>c (FP)</td>
</tr>
<tr>
<td>Class=No</td>
<td>Class=No</td>
</tr>
<tr>
<td></td>
<td>d (TN)</td>
</tr>
</tbody>
</table>

Accuracy = \( \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN} \)

Limitations of accuracy

Consider a binary problem

- Cardinality of Class 0 = 9900
- Cardinality of Class 1 = 100

Model

\( () \rightarrow \text{class 0} \)

- Model predicts everything to be class 0
  - accuracy is 9900/10000 = 99.0 %

- Accuracy is misleading because the model does not detect any class 1 object
Limitations of accuracy

- Classes may have different importance
  - Misclassification of objects of a given class is more important
  - e.g., ill patients erroneously assigned to the healthy patients class
- Accuracy is not appropriate for
  - unbalanced class label distribution
  - different class relevance

Class specific measures

- Evaluate separately for each class

Recall \( (r) = \frac{\text{Number of objects correctly assigned to } C}{\text{Number of objects belonging to } C} \)

Precision \( (p) = \frac{\text{Number of objects correctly assigned to } C}{\text{Number of objects assigned to } C} \)

Maximize

\[ F - \text{measure} (F) = \frac{2rp}{r + p} \]
Class specific measures

- For a binary classification problem
  - on the confusion matrix, for the positive class

  \[
  \begin{align*}
  \text{Precision (p)} &= \frac{a}{a + c} \\
  \text{Recall (r)} &= \frac{a}{a + b} \\
  F - \text{measure (F)} &= \frac{2rp}{r + p} = \frac{2a}{2a + b + c}
  \end{align*}
  \]

ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - characterizes the trade-off between positive hits and false alarms

- ROC curve plots
  - TPR, True Positive Rate (on the y-axis)
    \[
    \text{TPR} = \frac{TP}{TP+FN}
    \]
  - against
  - FPR, False Positive Rate (on the x-axis)
    \[
    \text{FPR} = \frac{FP}{FP + TN}
    \]
**ROC curve**

(FPR, TPR)
- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (0,1): ideal

- Diagonal line
  - Random guessing
  - Below diagonal line
    - prediction is opposite of the true class

---

**How to build a ROC curve**

| Instance | P(+|A) | True Class |
|----------|-------|------------|
| 1        | 0.95  | +          |
| 2        | 0.93  | +          |
| 3        | 0.87  | -          |
| 4        | 0.85  | -          |
| 5        | 0.85  | -          |
| 6        | 0.85  | +          |
| 7        | 0.76  | -          |
| 8        | 0.53  | +          |
| 9        | 0.43  | -          |
| 10       | 0.25  | +          |

- Use classifier that produces posterior probability for each test instance P(+|A)
- Sort the instances according to P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Count the number of TP, FP, TN, FN at each threshold
  - TP rate
    - TPR = TP/(TP+FN)
  - FP rate
    - FPR = FP/(FP + TN)
How to build a ROC curve

<table>
<thead>
<tr>
<th>Class</th>
<th>+</th>
<th>-</th>
<th>+</th>
<th>-</th>
<th>+</th>
<th>-</th>
<th>+</th>
<th>-</th>
<th>+</th>
<th>+</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(+</td>
<td>A)</td>
<td>0.25</td>
<td>0.43</td>
<td>0.53</td>
<td>0.76</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.87</td>
<td>0.93</td>
</tr>
<tr>
<td>TP</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>FP</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TN</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>FN</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>TPR</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
<td>0.6</td>
<td>0.6</td>
<td>0.6</td>
<td>0.4</td>
<td>0.4</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>FPR</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

ROC Curve

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006

Using ROC for Model Comparison

- No model consistently outperforms the other
- $M_1$ is better for small FPR
- $M_2$ is better for large FPR
- Area under ROC curve
  - Ideal Area = 1.0
  - Random guess Area = 0.5

From: Tan, Steinbach, Kumar, Introduction to Data Mining, McGraw Hill 2006