Classification Algorithms

Covering, Nearest-Neighbour
Outline

- Covering algorithms
- Instance-based (Nearest-neighbour)
Covering algorithms

- Strategy for generating a rule set directly:
  - for each class in turn find rule set that covers all instances in it (excluding instances not in the class)

- This approach is called a covering approach
  - at each stage a rule is identified that covers some of the instances
Example: generating a rule

IF TRUE then class = a

entire dataset
Example: generating a rule

IF TRUE then class = a

IF X > 1.2 then class = a
Example: generating a rule

a single rule describes a convex concept
Example: generating a rule

- ‘b’ is not convex → need extra rules for else part
- possible rule set for class ‘b’
- more rules could be added for a ‘perfect’ rule set
Rules vs. Trees

- Corresponding decision tree:
  (produces exactly the same predictions)
A simple covering algorithm

- Generates a rule by adding tests that maximize a rule’s accuracy
- Each additional test reduces a rule’s coverage:
Selecting a test

- **Goal:** maximize accuracy
  - $t$ total number of instances covered by rule
  - $p$ positive examples of the class covered by rule
  - $t - p$ number of errors made by rule
  - Select test that maximizes the ratio $p/t$

- We are finished when $p/t = 1$ or the set of instances cannot be split any further
## Contact lens data

<table>
<thead>
<tr>
<th>age</th>
<th>spectacle prescription</th>
<th>astigmatism</th>
<th>tear production rate</th>
<th>recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>young</td>
<td>myope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>young</td>
<td>myope</td>
<td>no</td>
<td>normal</td>
<td>soft</td>
</tr>
<tr>
<td>young</td>
<td>myope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>young</td>
<td>myope</td>
<td>yes</td>
<td>normal</td>
<td>hard</td>
</tr>
<tr>
<td>young</td>
<td>hypermetrope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>young</td>
<td>hypermetrope</td>
<td>yes</td>
<td>normal</td>
<td>soft</td>
</tr>
<tr>
<td>young</td>
<td>hypermetrope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>young</td>
<td>hypermetrope</td>
<td>yes</td>
<td>normal</td>
<td>hard</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>myope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>myope</td>
<td>no</td>
<td>normal</td>
<td>soft</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>myope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>myope</td>
<td>yes</td>
<td>normal</td>
<td>hard</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>hypermetrope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>hypermetrope</td>
<td>yes</td>
<td>normal</td>
<td>soft</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>hypermetrope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>pre-presbyopic</td>
<td>hypermetrope</td>
<td>yes</td>
<td>normal</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>myope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>myope</td>
<td>no</td>
<td>normal</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>myope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>myope</td>
<td>yes</td>
<td>normal</td>
<td>hard</td>
</tr>
<tr>
<td>presbyopic</td>
<td>hypermetrope</td>
<td>no</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>hypermetrope</td>
<td>no</td>
<td>normal</td>
<td>soft</td>
</tr>
<tr>
<td>presbyopic</td>
<td>hypermetrope</td>
<td>yes</td>
<td>reduced</td>
<td>none</td>
</tr>
<tr>
<td>presbyopic</td>
<td>hypermetrope</td>
<td>yes</td>
<td>normal</td>
<td>none</td>
</tr>
</tbody>
</table>
Example: contact lens data

Rule we seek:

```latex
\text{if} \ ? \ \text{then} \ \text{recommendation} = \text{hard}
```

Possible tests:

- Age = Young $\quad 2/8$
- Age = Pre-presbyopic $\quad 1/8$
- Age = Presbyopic $\quad 1/8$
- Spectacle prescription = Myope $\quad 3/12$
- Spectacle prescription = Hypermetrope $\quad 1/12$
- Astigmatism = no $\quad 0/12$
- Astigmatism = yes $\quad 4/12$
- Tear production rate = Reduced $\quad 0/12$
- Tear production rate = Normal $\quad 4/12$
Modified rule and resulting data

- Rule with best test added (4/12):
  
  \[
  \text{if astigmatism} = \text{yes} \\
  \text{then recommendation} = \text{hard}
  \]

- Instances covered by modified rule:

<table>
<thead>
<tr>
<th>Age</th>
<th>Spectacle prescription</th>
<th>Astigmatism</th>
<th>Tear production rate</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Reduced</td>
<td>None</td>
</tr>
</tbody>
</table>
Further refinement

- **Current state:**

- **Possible tests:**

  - Age = Young
  - Age = Pre-presbyopic
  - Age = Presbyopic
  - Spectacle prescription = Myope
  - Spectacle prescription = Hypermetrope
  - Tear production rate = Reduced
  - Tear production rate = Normal

```plaintext
if astigmatism = yes and ?
then recommendation = hard
```
Modified rule and resulting data

Rule with best test added:

```
if astigmatism = yes
    and tear production rate = normal
then recommendation = hard
```

Instances covered by modified rule:

<table>
<thead>
<tr>
<th>Age</th>
<th>Spectacle prescription</th>
<th>Astigmatism</th>
<th>Tear production rate</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Young</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Young</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Pre-presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Myope</td>
<td>Yes</td>
<td>Normal</td>
<td>Hard</td>
</tr>
<tr>
<td>Presbyopic</td>
<td>Hypermetrope</td>
<td>Yes</td>
<td>Normal</td>
<td>None</td>
</tr>
</tbody>
</table>
Further refinement

Current state:

if astigmatism = yes
   and tear production rate = normal
   and ?
then recommendation = hard

Possible tests:

- Age = Young 2/2
- Age = Pre-presbyopic 1/2
- Age = Presbyopic 1/2
- Spectacle prescription = Myope 3/3
- Spectacle prescription = Hypermetrope 1/3

Tie between the first and the fourth test

We choose the one with greater coverage
The result

- Final rule:

```
if astigmatism = yes
    and tear production rate = normal
    and spectacle prescription = myope
then recommendation = hard
```

- Second rule for recommending “hard lenses”:
  (built from instances not covered by first rule)

```
if age = young and astigmatism = yes
    and tear production rate = normal
then recommendation = hard
```

- These two rules cover all “hard lenses”
- Process is repeated with other two classes
Pseudo-code for PRISM

For each class C
  Initialize D to the instance set
  While D contains instances in class C
    Create a rule R with an empty left-hand side that predicts class C
    Until R is perfect (or there are no more attributes to use) do
      For each attribute A not mentioned in R, and each value v,
        Consider adding the condition A = v to the left-hand side of R
        Select A and v to maximize the accuracy p/t
          (break ties by choosing the condition with the largest p)
        Add A = v to R
      Remove the instances covered by R from D
Separate and conquer

- Methods like PRISM (for dealing with one class) are *separate-and-conquer* algorithms:
  - First, a rule is identified
  - Then, all instances covered by the rule are separated out
  - Finally, the remaining instances are “conquered”

- Difference to *divide-and-conquer* methods:
  - Subset covered by rule doesn’t need to be explored any further
Instance-based Classification

- k-nearest neighbour
Instance-based representation

- Simplest form of learning: rote learning
  - Training instances are searched for instance that most closely resembles new instance
  - The instances themselves represent the knowledge
  - Also called instance-based learning
- Similarity function defines what’s “learned”
- Instance-based learning is lazy learning
- Methods:
  - nearest neighbor
  - k-nearest neighbor
  - ...
1-NN example
The distance function

- Simplest case: one numeric attribute
  - Distance is the difference between the two attribute values involved (or a function thereof)

- Several numeric attributes: normally, Euclidean distance is used and attributes are normalized

- Nominal attributes: distance is set to 1 if values are different, 0 if they are equal

- Are all attributes equally important?
  - Weighting the attributes might be necessary
Instance-based learning

- Most instance-based schemes use *Euclidean distance*:

\[
\sqrt{(a_1^{(1)} - a_1^{(2)})^2 + (a_2^{(1)} - a_2^{(2)})^2 + \ldots + (a_k^{(1)} - a_k^{(2)})^2}
\]

\(a^{(1)}\) and \(a^{(2)}\): two instances with \(k\) attributes

- Taking the square root is not required when comparing distances

- Other popular metric: city-block (Manhattan) metric
  - Adds differences without squaring them
Nearest Neighbour & Voronoi diagram
Normalization and other issues

- Different attributes are measured on different scales
- Need to be **normalized**:
  \[
  a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i} \quad \text{or} \quad a_i = \frac{v_i - \text{Avg}(v_i)}{\text{StDev}(v_i)}
  \]
  
  \(v_i\): the actual value of attribute \(i\)

- Nominal attributes: distance either 0 or 1
- Common policy for missing values: assumed to be maximally distant (given normalized attributes)
**k-NN example**

- **k-NN approach**: perform a majority vote to derive label
Discussion of 1-NN

- Can be very accurate
  - But: *curse of dimensionality*
    - Every added dimension increases distances
    - Exponentially more training data needed

- ... but slow:
  - Simple version scans entire training data to make prediction
  - Better training set representations: $kD$-tree, ball tree,...

- Assumes all attributes are equally important
  - Remedy: attribute selection or weights

- Possible remedies against noisy instances:
  - Take a majority vote over the $k$ nearest neighbors
  - Removing noisy instances from dataset (difficult!)
Summary

- Simple methods frequently work well
  - robust against noise, errors
- Advanced methods, if properly used, can improve on simple methods
- No method is universally best