Data Mining
Practical Machine Learning Tools and Techniques

Slides for Chapter 2 of Data Mining by I. H. Witten, E. Frank and M. A. Hall

Input: Concepts, instances, attributes

- Terminology
- What’s a concept?
  - Classification, association, clustering, numeric prediction
- What’s in an example?
  - Relations, flat files, recursion
- What’s in an attribute?
  - Nominal, ordinal, interval, ratio
- Preparing the input
  - ARFF, attributes, missing values, getting to know data

Terminology

- Components of the input:
  - Concepts: kinds of things that can be learned
    - Aim: intelligible and operational concept description
  - Instances: the individual, independent examples of a concept
    - Note: more complicated forms of input are possible
  - Attributes: measuring aspects of an instance
    - We will focus on nominal and numeric ones
  - Styles of learning:
    - Classification learning: predicting a discrete class
    - Association learning: detecting associations between features
    - Clustering: grouping similar instances into clusters
    - Numeric prediction: predicting a numeric quantity
  - Concept: thing to be learned
  - Concept description: output of learning scheme
Classification learning

- Example problems: weather data, contact lenses, irises, labor negotiations
- Classification learning is *supervised*
  - Scheme is provided with actual outcome
- Outcome is called the *class* of the example
- Measure success on fresh data for which class labels are known (*test data*)
- In practice success is often measured subjectively

Association learning

- Can be applied if no class is specified and any kind of structure is considered “interesting”
- Difference to classification learning:
  - Can predict any attribute’s value, not just the class, and more than one attribute’s value at a time
  - Hence: far more association rules than classification rules
  - Thus: constraints are necessary
    - Minimum coverage and minimum accuracy

Clustering

- Finding groups of items that are similar
- Clustering is *unsupervised*
  - The class of an example is not known
- Success often measured subjectively

<table>
<thead>
<tr>
<th></th>
<th>Sepal length</th>
<th>Sepal width</th>
<th>Petal length</th>
<th>Petal width</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.1</td>
<td>3.5</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris setosa</td>
</tr>
<tr>
<td>2</td>
<td>4.9</td>
<td>3.0</td>
<td>1.4</td>
<td>0.2</td>
<td>Iris setosa</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>51</td>
<td>7.0</td>
<td>3.2</td>
<td>4.7</td>
<td>1.4</td>
<td>Iris versicolor</td>
</tr>
<tr>
<td>52</td>
<td>6.4</td>
<td>3.2</td>
<td>4.5</td>
<td>1.5</td>
<td>Iris versicolor</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>101</td>
<td>6.3</td>
<td>3.3</td>
<td>6.0</td>
<td>2.5</td>
<td>Iris virginica</td>
</tr>
<tr>
<td>102</td>
<td>5.8</td>
<td>2.7</td>
<td>5.1</td>
<td>1.9</td>
<td>Iris virginica</td>
</tr>
</tbody>
</table>

Numeric prediction

- Variant of classification learning where “class” is numeric (also called “regression”)
- Learning is supervised
  - Scheme is being provided with target value
- Measure success on test data

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>windy</th>
<th>Play-time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>5</td>
</tr>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>True</td>
<td>0</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>55</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>Normal</td>
<td>False</td>
<td>40</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
**What’s in an example?**

- Instance: specific type of example
- Thing to be classified, associated, or clustered
- Individual, independent example of target concept
- Characterized by a predetermined set of attributes
- Input to learning scheme: set of instances/dataset
- Represented as a single relation/flat file
- Rather restricted form of input
- No relationships between objects
- Most common form in practical data mining

---

**A family tree**

Steven  M  =  Peggy  F  

Peter  M  =  Peggy  F  

Grace  F  =  Ray  M  

Graham  M  =  Pippa  F  

Pam  F  =  Ian  M  

Pippa  F  =  Brian  M  

Anna  F  =  Nikki  F  

---

**Family tree represented as a table**

<table>
<thead>
<tr>
<th>Name</th>
<th>Gender</th>
<th>Parent1</th>
<th>Parent2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>Male</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Peggy</td>
<td>Female</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Steven</td>
<td>Male</td>
<td>Peter</td>
<td>Peggy</td>
</tr>
<tr>
<td>Graham</td>
<td>Male</td>
<td>Peter</td>
<td>Peggy</td>
</tr>
<tr>
<td>Pam</td>
<td>Female</td>
<td>Peter</td>
<td>Peggy</td>
</tr>
<tr>
<td>Ian</td>
<td>Male</td>
<td>Grace</td>
<td>Ray</td>
</tr>
<tr>
<td>Pippa</td>
<td>Female</td>
<td>Grace</td>
<td>Ray</td>
</tr>
<tr>
<td>Brian</td>
<td>Male</td>
<td>Grace</td>
<td>Ray</td>
</tr>
<tr>
<td>Anna</td>
<td>Female</td>
<td>Pam</td>
<td>Ian</td>
</tr>
<tr>
<td>Nikki</td>
<td>Female</td>
<td>Pam</td>
<td>Ian</td>
</tr>
</tbody>
</table>

---

**The “sister-of” relation**

<table>
<thead>
<tr>
<th>First person</th>
<th>Second person</th>
<th>Sister of?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>Peggy</td>
<td>No</td>
</tr>
<tr>
<td>Peter</td>
<td>Steven</td>
<td>No</td>
</tr>
<tr>
<td>Steven</td>
<td>Peter</td>
<td>No</td>
</tr>
<tr>
<td>Steven</td>
<td>Graham</td>
<td>No</td>
</tr>
<tr>
<td>Steven</td>
<td>Pam</td>
<td>Yes</td>
</tr>
<tr>
<td>Ian</td>
<td>Pippa</td>
<td>Yes</td>
</tr>
<tr>
<td>Brian</td>
<td>Pippa</td>
<td>Yes</td>
</tr>
<tr>
<td>Anna</td>
<td>Nikki</td>
<td>Yes</td>
</tr>
<tr>
<td>Nikki</td>
<td>Anna</td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Closed-world assumption**
A full representation in one table

<table>
<thead>
<tr>
<th>First person</th>
<th>Second person</th>
<th>Sister of?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Gender</td>
<td>Parent1</td>
</tr>
<tr>
<td>Steven</td>
<td>Male</td>
<td>Peter</td>
</tr>
<tr>
<td>Graham</td>
<td>Male</td>
<td>Peter</td>
</tr>
<tr>
<td>Ian</td>
<td>Male</td>
<td>Grace</td>
</tr>
<tr>
<td>Brian</td>
<td>Male</td>
<td>Grace</td>
</tr>
<tr>
<td>Anna</td>
<td>Female</td>
<td>Pam</td>
</tr>
<tr>
<td>Nikki</td>
<td>Female</td>
<td>Pam</td>
</tr>
</tbody>
</table>

If second person’s gender = female and first person’s parent = second person’s parent then sister-of = yes

Generating a flat file

- Process of flattening called “denormalization”
  - Several relations are joined together to make one
- Possible with any finite set of finite relations
- Problematic: relationships without pre-specified number of objects
  - Example: concept of nuclear-family
- Denormalization may produce spurious regularities that reflect structure of database
  - Example: “supplier” predicts “supplier address”

The “ancestor-of” relation

<table>
<thead>
<tr>
<th>First person</th>
<th>Second person</th>
<th>Ancestor of?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Gender</td>
<td>Parent1</td>
</tr>
<tr>
<td>Peter</td>
<td>Male</td>
<td>?</td>
</tr>
<tr>
<td>Peter</td>
<td>Male</td>
<td>?</td>
</tr>
<tr>
<td>Peter</td>
<td>Male</td>
<td>?</td>
</tr>
<tr>
<td>Peter</td>
<td>Male</td>
<td>?</td>
</tr>
<tr>
<td>Pam</td>
<td>Female</td>
<td>Peter</td>
</tr>
<tr>
<td>Grace</td>
<td>Female</td>
<td>?</td>
</tr>
<tr>
<td>Grace</td>
<td>Female</td>
<td>?</td>
</tr>
</tbody>
</table>

If person1 is a parent of person2 then person1 is an ancestor of person2

If person1 is a parent of person2 and person2 is an ancestor of person3 then person1 is an ancestor of person3

- Infinite relations require recursion

- Appropriate techniques are known as “inductive logic programming”
  - (e.g. Quinlan’s FOIL)
  - Problems: (a) noise and (b) computational complexity
Multi-instance Concepts

- Each individual example comprises a set of instances
  - All instances are described by the same attributes
  - One or more instances within an example may be responsible for its classification
- Goal of learning is still to produce a concept description
- Important real world applications
  - e.g. drug activity prediction

Nominal quantities

- Values are distinct symbols
  - Values themselves serve only as labels or names
  - Nominal comes from the Latin word for name
- Example: attribute “outlook” from weather data
  - Values: “sunny”, ”overcast”, and “rainy”
- No relation is implied among nominal values (no ordering or distance measure)
- Only equality tests can be performed

Ordinal quantities

- Impose order on values
- But: no distance between values defined
- Example: attribute “temperature” in weather data
  - Values: “hot” > “mild” > “cool”
- Note: addition and subtraction don’t make sense
- Example rule:
  temperature < hot ⇒ play = yes
- Distinction between nominal and ordinal not always clear (e.g. attribute “outlook”)

What’s in an attribute?

- Each instance is described by a fixed predefined set of features, its “attributes”
- But: number of attributes may vary in practice
  - Possible solution: “irrelevant value” flag
- Related problem: existence of an attribute may depend of value of another one
- Possible attribute types (“levels of measurement”):
  - Nominal, ordinal, interval and ratio
Interval quantities

- Interval quantities are not only ordered but measured in fixed and equal units
- Example 1: attribute “temperature” expressed in degrees Fahrenheit
- Example 2: attribute “year”
- Difference of two values makes sense
- Sum or product doesn’t make sense
  - Zero point is not defined!

Ratio quantities

- Ratio quantities are ones for which the measurement scheme defines a zero point
- Example: attribute “distance”
  - Distance between an object and itself is zero
- Ratio quantities are treated as real numbers
  - All mathematical operations are allowed
- But: is there an “inherently” defined zero point?
  - Answer depends on scientific knowledge (e.g. Fahrenheit knew no lower limit to temperature)

Attribute types used in practice

- Most schemes accommodate just two levels of measurement: nominal and ordinal
- Nominal attributes are also called “categorical”, “enumerated”, or “discrete”
  - But: “enumerated” and “discrete” imply order
- Special case: dichotomy (“boolean” attribute)
- Ordinal attributes are called “numeric”, or “continuous”
  - But: “continuous” implies mathematical continuity

Metadata

- Information about the data that encodes background knowledge
- Can be used to restrict search space
- Examples:
  - Dimensional considerations (i.e. expressions must be dimensionally correct)
  - Circular orderings (e.g. degrees in compass)
  - Partial orderings (e.g. generalization/specialization relations)
Preparing the input

- Denormalization is not the only issue
- Problem: different data sources (e.g. sales department, customer billing department, …)
  - Differences: styles of record keeping, conventions, time periods, data aggregation, primary keys, errors
  - Data must be assembled, integrated, cleaned up
  - “Data warehouse”: consistent point of access
- External data may be required (“overlay data”)
- Critical: type and level of data aggregation

The ARFF format

@attribute outlook {sunny, overcast, rainy}
@attribute temperature numeric
@attribute humidity numeric
@attribute windy {true, false}
@attribute play? {yes, no}
@data
sunny, 85, 85, false, no
sunny, 80, 90, true, no
overcast, 83, 86, false, yes
...

Additional attribute types

- ARFF supports string attributes:
  ```
  @attribute description string
  ```
  - Similar to nominal attributes but list of values is not pre-specified
- It also supports date attributes:
  ```
  @attribute today date
  ```
  - Uses the ISO-8601 combined date and time format yyyy-MM-dd-THH:mm:ss

Relational attributes

- Allow multi-instance problems to be represented in ARFF format
  - The value of a relational attribute is a separate set of instances
    ```
    @attribute bag relational
    @attribute outlook { sunny, overcast, rainy }
    @attribute temperature numeric
    @attribute humidity numeric
    @attribute windy { true, false }
    @end bag
    ```
  - Nested attribute block gives the structure of the referenced instances
Multi-instance ARFF

% Multiple instance ARFF file for the weather data
% @relation weather
@attribute bag_ID { 1, 2, 3, 4, 5, 6, 7 }
@attribute bag relational
    @attribute outlook {sunny, overcast, rainy}
    @attribute temperature numeric
    @attribute humidity numeric
    @attribute windy {true, false}
@end bag
@attribute play? {yes, no}
@data
1, "sunny, 85, 85, false \n sunny, 80, 90, true", no
2, "overcast, 83, 86, false \n rainy, 70, 96, false", yes

Sparse data

- In some applications most attribute values in a dataset are zero
  
  • E.g.: word counts in a text categorization problem

- ARFF supports sparse data

```
0, 26, 0, 0, 0, 63, 0, 0, 0, “class A”
0, 0, 0, 42, 0, 0, 0, 0, 0, “class B”
```

- This also works for nominal attributes (where the first value corresponds to “zero”)

```
{1 26, 6 63, 10 “class A”}
{3 42, 10 “class B”}
```

Attribute types

- Interpretation of attribute types in ARFF depends on learning scheme
  
  • Numeric attributes are interpreted as
    
    • ordinal scales if less-than and greater-than are used
    
    • ratio scales if distance calculations are performed (normalization/standardization may be required)

  • Instance-based schemes define distance between nominal values (0 if values are equal, 1 otherwise)

  • Integers in some given data file: nominal, ordinal, or ratio scale?

Nominal vs. ordinal

- Attribute “age” nominal

```
If age = young and astigmatic = no
and tear production rate = normal
then recommendation = soft
```
```
If age = pre-presbyopic and astigmatic = no
and tear production rate = normal
then recommendation = soft
```

- Attribute “age” ordinal

  (e.g. “young” < “pre-presbyopic” < “presbyopic”)

```
If age ≤ pre-presbyopic and astigmatic = no
and tear production rate = normal
then recommendation = soft
```
Missing values

• Frequently indicated by out-of-range entries
  • Types: unknown, unrecorded, irrelevant
  • Reasons:
    • malfunctioning equipment
    • changes in experimental design
    • collation of different datasets
    • measurement not possible

• Missing value may have significance in itself (e.g. missing test in a medical examination)
  • Most schemes assume that is not the case: “missing” may need to be coded as additional value

Inaccurate values

• Reason: data has not been collected for mining it
• Result: errors and omissions that don’t affect original purpose of data (e.g. age of customer)
• Typographical errors in nominal attributes ⇒ values need to be checked for consistency
• Typographical and measurement errors in numeric attributes ⇒ outliers need to be identified
• Errors may be deliberate (e.g. wrong zip codes)
• Other problems: duplicates, stale data

Getting to know the data

• Simple visualization tools are very useful
  • Nominal attributes: histograms (Distribution consistent with background knowledge?)
  • Numeric attributes: graphs (Any obvious outliers?)
  • 2-D and 3-D plots show dependencies
  • Need to consult domain experts
  • Too much data to inspect? Take a sample!