Data Mining

Practical Machine Learning Tools and Techniques

Slides for Chapter 2 of *Data Mining* by I. H. Witten and E. Frank

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

WEKA
The University of Waikato

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

What's a concept?

- 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

	Sepal length	Sepal width	Petal length	Petal width	Type
1	5.1	3.5	1.4	0.2	Iris setosa
2	4.9	3.0	1.4	0.2	Iris setosa
…					
51	7.0	3.2	4.7	1.4	Iris versicolor
52	6.4	3.2	4.5	1.5	Iris versicolor
…					
101	6.3	3.3	6.0	2.5	Iris virginica
102	5.8	2.7	5.1	1.9	Iris virginica
…					

Numeric prediction

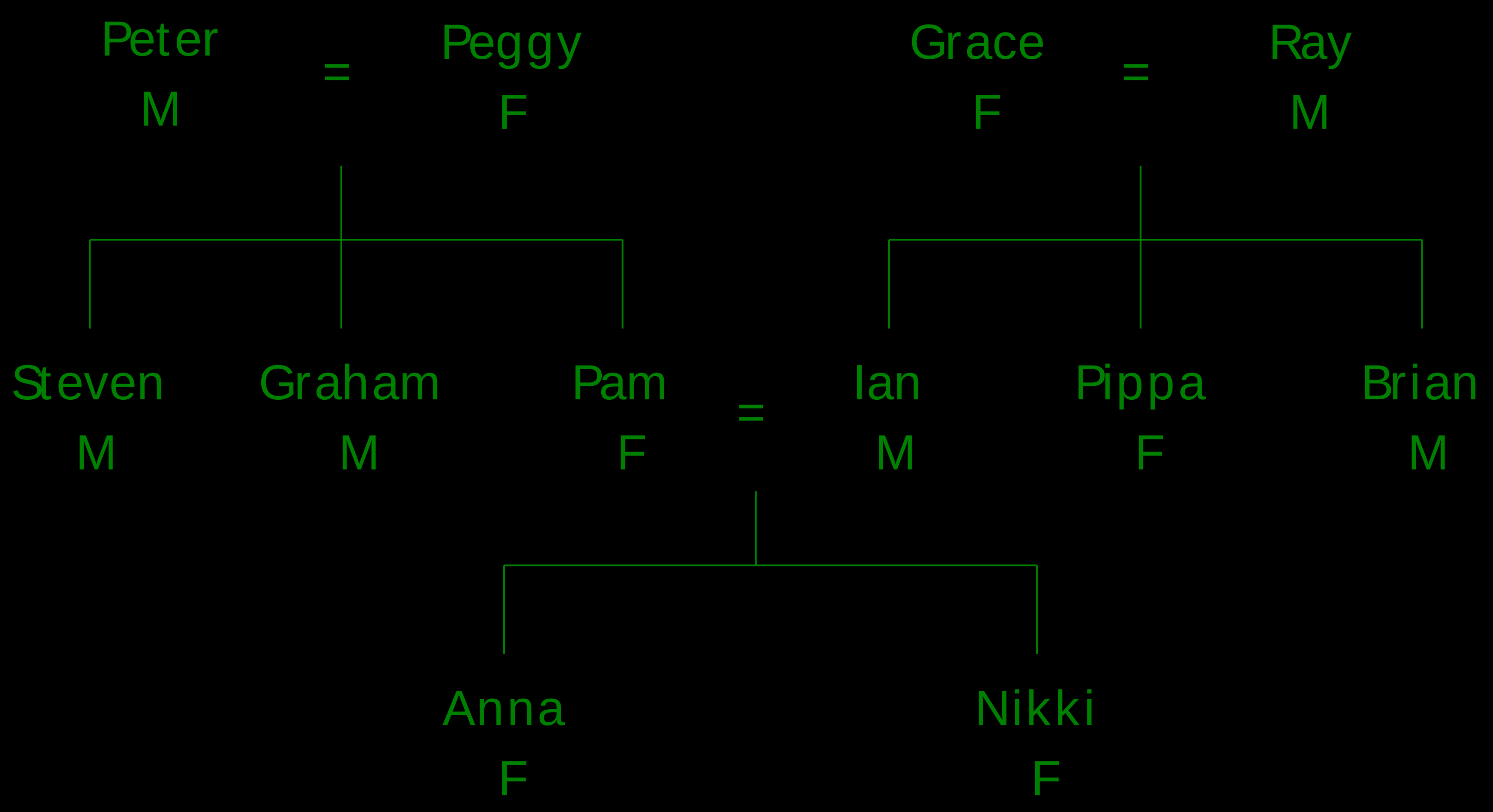
- Variant of classification learning where "class" is numeric (also called "regression")
- Learning is supervised
  - Scheme is being provided with target value
- Measure success on test data

Outlook	Temperature	Humidity	Windy	Play-time
Sunny	Hot	High	False	5
Sunny	Hot	High	True	0
Overcast	Hot	High	False	55
Rainy	Mild	Normal	False	40
…	…	…	…	…

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



Family tree represented as a table

Name	Gender	Parent1	parent2
Peter	Male	?	?
Peggy	Female	?	?
Steven	Male	Peter	Peggy
Graham	Male	Peter	Peggy
Pam	Female	Peter	Peggy
Ian	Male	Grace	Ray
Pippa	Female	Grace	Ray
Brian	Male	Grace	Ray
Anna	Female	Pam	Ian
Nikki	Female	Pam	Ian

The “sister-of” relation

First person	Second person	Sister of?
Peter	Peggy	No
Peter	Steven	No
…	…	…
Steven	Peter	No
Steven	Graham	No
Steven	Pam	Yes
…	…	…
Ian	Pippa	Yes
…	…	…
Anna	Nikki	Yes
…	…	…
Nikki	Anna	yes

First person	Second person	Sister of?
Steven	Pam	Yes
Graham	Pam	Yes
Ian	Pippa	Yes
Brian	Pippa	Yes
Anna	Nikki	Yes
Nikki	Anna	Yes
All the rest		No

Closed-world assumption

A full representation in one table

First person				Second person				Sister of?
Name	Gender	Parent1	Parent2	Name	Gender	Parent1	Parent2	
Steven	Male	Peter	Peggy	Pam	Female	Peter	Peggy	Yes
Graham	Male	Peter	Peggy	Pam	Female	Peter	Peggy	Yes
Ian	Male	Grace	Ray	Pippa	Female	Grace	Ray	Yes
Brian	Male	Grace	Ray	Pippa	Female	Grace	Ray	Yes
Anna	Female	Pam	Ian	Nikki	Female	Pam	Ian	Yes
Nikki	Female	Pam	Ian	Anna	Female	Pam	Ian	Yes
All the rest								No

```
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

First person				Second person				Ancestor of?
Name	Gender	Parent	Parent	Name	Gender	Parent1	Parent2	
Peter	Male	?	?	Steven	Male	Peter	Peggy	Yes
Peter	Male	?	?	Pam	Female	Peter	Peggy	Yes
Peter	Male	?	?	Anna	Female	Pam	Ian	Yes
Peter	Male	?	?	Nikki	Female	Pam	Ian	Yes
Pam	Female	Peter	Peggy	Nikki	Female	Pam	Ian	Yes
Grace	Female	?	?	Ian	Male	Grace	Ray	Yes
Grace	Female	?	?	Nikki	Female	Pam	Ian	Yes
Other positive examples here								Yes
All the rest								No

Recursion

- Infinite relations require recursion

```
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
```

- Appropriate techniques are known as "inductive logic programming"
  - (e.g. Quinlan's FOIL)
  - Problems: (a) noise and (b) computational complexity

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*

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 $\Rightarrow$ play = yes
- Distinction between nominal and ordinal not always clear (e.g. attribute “outlook”)

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

```
%
% ARFF file for weather data with some numeric features
%
@relation weather

@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*

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

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

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

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 $\Rightarrow$ values need to be checked for consistency
- Typographical and measurement errors in numeric attributes $\Rightarrow$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!