Data Science with R

Association Rules Analysis

Graham.Williams@togaware.com

Senior Director and Data Scientist, Analytics
Australian Taxation Office

Adjunct Professor, Australian National University
Fellow, Institute of Analytics Professionals of Australia

Graham.Williams@togaware.com
http://datamining.togaware.com

Overview

Association Rule Mining

- An unsupervised learning algorithm—descriptive data mining.
- *Identify items (patterns) that occur frequently together in a given set of data.*
- Patterns = associations, correlations, causal structures (Rules).
- Data = sets of items in ...
  - transactional database
  - relational database
  - complex information repositories
- Rule: $Body \rightarrow Head\ [support, confidence]$

Association Rule Mining

- An unsupervised learning algorithm—descriptive data mining.
- *Identify items (patterns) that occur frequently together in a given set of data.*
- Patterns = associations, correlations, causal structures (Rules).
- Data = sets of items in ...
  - transactional database
  - relational database
  - complex information repositories
- Rule: *Body* $\rightarrow$ *Head* [*support*, *confidence*]

Association Rule Mining

- An unsupervised learning algorithm—descriptive data mining.
- *Identify items (patterns) that occur frequently together in a given set of data.*
- Patterns = associations, correlations, causal structures (Rules).
- Data = sets of items in . . .
    - transactional database
    - relational database
    - complex information repositories
- Rule: $Body \rightarrow Head\ [support, confidence]$

Association Rule Mining

- An unsupervised learning algorithm—descriptive data mining.
- *Identify items (patterns) that occur frequently together in a given set of data.*
- Patterns = associations, correlations, causal structures (Rules).
- Data = sets of items in . . .
  - transactional database
  - relational database
  - complex information repositories
- Rule: $\mathit{Body} \rightarrow \mathit{Head}\ [\mathit{support}, \mathit{confidence}]$

Typical Applications

- Link analysis
- Market basket analysis
- Cross marketing
- Customers who purchase . . .

Examples

- *Friday* $\cap$ *Nappies* $\rightarrow$ *Beer*
  $[0.5\%, 60\%]$
- *Age* $\in [20, 30] \cap$ *Income* $\in [20K, 30K] \rightarrow$ *MP3Player*
  $[2\%, 60\%]$
- *Maths* $\cap$ *CS* $\rightarrow$ *HDinCS*
  $[1\%, 75\%]$
- *Gladiator* $\cap$ *Patriot* $\rightarrow$ *Sixth Sense*
  $[0.1\%, 90\%]$
- *Statins* $\cap$ *Peritonitis* $\rightarrow$ *Chronic Renal Failure*
  $[0.1\%, 32\%]$

Examples

- $Friday \cap Nappies \rightarrow Beer$
  $[0.5\%, 60\%]$
- $Age \in [20, 30] \cap Income \in [20K, 30K] \rightarrow MP3Player$
  $[2\%, 60\%]$
- $Maths \cap CS \rightarrow HDinCS$
  $[1\%, 75\%]$
- $Gladiator \cap Patriot \rightarrow Sixth\ Sense$
  $[0.1\%, 90\%]$
- $Statins \cap Peritonitis \rightarrow Chronic\ Renal\ Failure$
  $[0.1\%, 32\%]$

Examples

- $Friday \cap Nappies \rightarrow Beer$
  $[0.5\%, 60\%]$
- $Age \in [20, 30] \cap Income \in [20K, 30K] \rightarrow MP3Player$
  $[2\%, 60\%]$
- $Maths \cap CS \rightarrow HDinCS$
  $[1\%, 75\%]$
- $Gladiator \cap Patriot \rightarrow Sixth\ Sense$
  $[0.1\%, 90\%]$
- $Statins \cap Peritonitis \rightarrow Chronic\ Renal\ Failure$
  $[0.1\%, 32\%]$

Examples

- $Friday \cap Nappies \rightarrow Beer$
  $[0.5\%, 60\%]$
- $Age \in [20, 30] \cap Income \in [20K, 30K] \rightarrow MP3Player$
  $[2\%, 60\%]$
- $Maths \cap CS \rightarrow HDinCS$
  $[1\%, 75\%]$
- $Gladiator \cap Patriot \rightarrow Sixth\ Sense$
  $[0.1\%, 90\%]$
- $Statins \cap Peritonitis \rightarrow Chronic\ Renal\ Failure$
  $[0.1\%, 32\%]$

Examples

- $Friday \cap Nappies \rightarrow Beer$
  $[0.5\%, 60\%]$
- $Age \in [20, 30] \cap Income \in [20K, 30K] \rightarrow MP3Player$
  $[2\%, 60\%]$
- $Maths \cap CS \rightarrow HDinCS$
  $[1\%, 75\%]$
- $Gladiator \cap Patriot \rightarrow Sixth\ Sense$
  $[0.1\%, 90\%]$
- $Statins \cap Peritonitis \rightarrow Chronic\ Renal\ Failure$
  $[0.1\%, 32\%]$

Framework

- Given
  - Database of transactions
  - Each transaction is a list of items
    E.g. Contents of customer's shopping basket
- Search for all rules that associate one set of items with another set.
- *Every possible association?*

Framework

- Given
  - Database of transactions
  - Each transaction is a list of items
    E.g. Contents of customer's shopping basket
- Search for all rules that associate one set of items with another set.
- *Every possible association?*

Framework

- Given
  - Database of transactions
  - Each transaction is a list of items
    E.g. Contents of customer's shopping basket
- Search for all rules that associate one set of items with another set.
- *Every possible association?*

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$
  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$
  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$
  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$ $support(X \rightarrow Y) - support(X) * support(Y)$

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$
  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$
  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$
  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$
    $support(X \rightarrow Y) - support(X) * support(Y)$

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$

  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$

  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$

  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$
    $support(X \rightarrow Y) - support(X) * support(Y)$

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$
  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$
  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$
  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$
    $support(X \rightarrow Y) - support(X) * support(Y)$

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$
  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$
  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$
  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$ $support(X \rightarrow Y) - support(X) * support(Y)$

Interesting Association Rules

- Measure interestingness of a rule in terms of
  - support: how frequently items appear together
  - confidence: how frequently they conditionally appear
  - lift: increased likelihood of Y if X included
- $X \rightarrow Y[s\%, c\%]$
  - the rule holds in s% of all transactions
  - $support(X \rightarrow Y) = P(X \cup Y)$
  - if X is in the basket, then so is Y in c% of the cases
  - $confidence(X \rightarrow Y) = P(Y|X) = P(X \cup Y)/P(X)$
  - if X is in the basket, then Y more likely in the basket
  - $lift(X \rightarrow Y) = confidence(X \rightarrow Y)/support(Y)$
  - higher frequency of X and Y with lower lift may be interesting
  - $leverage(X \rightarrow Y) =$
    $support(X \rightarrow Y) - support(X) * support(Y)$

Example

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$A \to B$, $A \to B, C$
$C \to A, B$
$A \to C$, $C \to A$
...

- Parameters:
  support $= 50\%$
  confidence $= 50\%$
- $A \to C[50\%, 66.6\%]$
- $C \to A[50\%, 100\%]$

Example

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$A \to B,\ A \to B, C$

$C \to A, B$

$A \to C,\ C \to A$

...

- Parameters:
  support = 50%
  confidence = 50%
- $A \to C[50\%, 66.6\%]$
- $C \to A[50\%, 100\%]$

Example

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$A \rightarrow B$, $A \rightarrow B, C$
$C \rightarrow A, B$
$A \rightarrow C$, $C \rightarrow A$
...

- Parameters:
  support $= 50\%$
  confidence $= 50\%$
- $A \rightarrow C[50\%, 66.6\%]$
- $C \rightarrow A[50\%, 100\%]$

Example

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$A \rightarrow B,\ A \rightarrow B, C$
$C \rightarrow A, B$
$A \rightarrow C,\ C \rightarrow A$
...

- Parameters:
  support $= 50\%$
  confidence $= 50\%$
- $A \rightarrow C[50\%, 66.6\%]$
- $C \rightarrow A[50\%, 100\%]$

Itemsets: Basis of Algorithm

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$\Rightarrow$

Itemset	Support
A	75%
B	50%
C	50%
A, C	50%

Parameters:
support = 50%
confidence = 50%

Rule $A \rightarrow C$

- support$(A, C) = 50\%$
- confidence$(A \rightarrow C)$
  $=$ support(A, C)/support(A)
  $= 66.6\%$

Itemsets: Basis of Algorithm

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$\Rightarrow$

Itemset	Support
A	75%
B	50%
C	50%
A, C	50%

Parameters:
support $=$ 50%
confidence $=$ 50%

Rule $A \rightarrow C$

- support$(A, C) = 50\%$
- confidence$(A \rightarrow C)$
  $=$ support(A, C)/support(A)
  $= 66.6\%$

Itemsets: Basis of Algorithm

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$\Rightarrow$

Itemset	Support
A	75%
B	50%
C	50%
A, C	50%

Parameters:
support $=$ 50%
confidence $=$ 50%

Rule $A \rightarrow C$

- support$(A, C) = 50\%$
- confidence$(A \rightarrow C)$
  $=$ support(A, C)/support(A)
  $= 66.6\%$

Itemsets: Basis of Algorithm

Transaction	Items
12345	A B C
12346	A C
12347	A D
12348	B E F

$\Rightarrow$

Itemset	Support
A	75%
B	50%
C	50%
A, C	50%

Parameters:
support $=$ 50%
confidence $=$ 50%

Rule $A \rightarrow C$

- support(A, C) $=$ 50%
- confidence$(A \rightarrow C)$
  $=$ support(A, C)/support(A)
  $=$ 66.6%

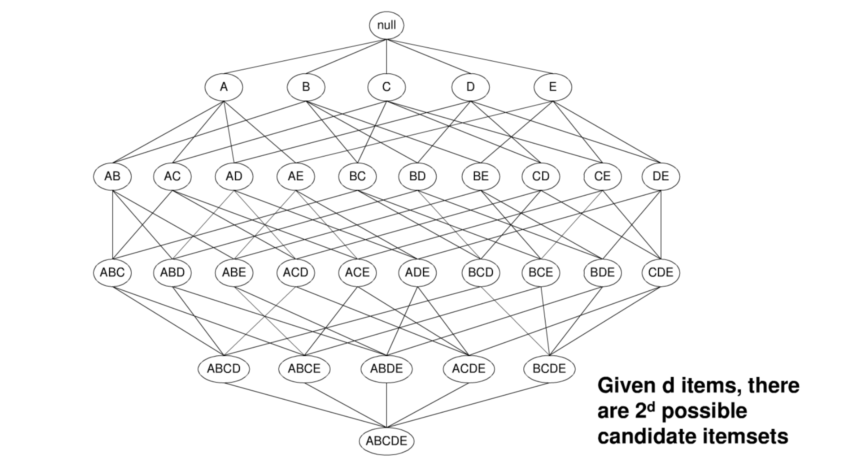
Algorithm Outline

- Find all frequent itemsets
  - sets of items with at least minimum support
  - support is the frequency of occurrence of the itemset
  - *k-itemset* contains *k* items
  - Computationally expensive: Apriori algorithm
- Generate strong association rules from the frequent itemsets
  - For ABCD and AB in frequent itemset the rule AB $\Rightarrow$ CD holds if ratio s(ABCD)/s(AB) is large enough
  - This ratio is the confidence of the rule

Algorithm Outline

- Find all frequent itemsets
  - sets of items with at least minimum support
  - support is the frequency of occurrence of the itemset
  - *k-itemset* contains k items
  - Computationally expensive: Apriori algorithm
- Generate strong association rules from the frequent itemsets
  - For ABCD and AB in frequent itemset the rule AB $\Rightarrow$ CD holds if ratio s(ABCD)/s(AB) is large enough
  - This ratio is the confidence of the rule

Large Search Space



http://www.slideshare.net/pierluca.lanzi/dmtm-04-association-rules-basics

Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

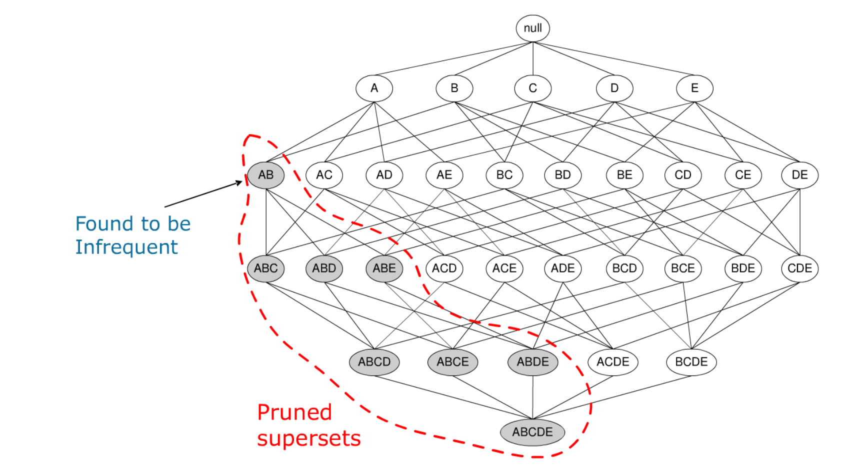
Apriori Algorithm

Basic principle:

Any subset of a frequent itemset must be frequent

- Find the *frequent itemsets*: the sets of items that have minimum support
  - A subset of a frequent itemset must also be a frequent itemset.
  - If AB is a frequent itemset, both A and B should be a frequent itemsets.
  - Iteratively find frequent itemsets with cardinality from 1 to k.
- Use the frequent itemsets to generate association rules.

Pruned Search Space



http://www.slideshare.net/pierluca.lanzi/dmtm-04-association-rules-basics

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k++)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
  - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k++)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
    - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k++)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
    - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k++)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
  - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k++)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
    - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k + +)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
  - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Apriori Algorithm

C_k: Candidate itemset of size k
L_k: Frequent itemset of size k
L_1 ={frequent items}
For $(k = 2; L_k \neq 0; k + +)$

- C_k = candidates generated from L_{k-1}
- For each transaction $t \in D$
  - increment count of candidates in C_k contained in t
- L_k = candidates in C_k with at least min support.

Generate the Rules

Generate the **strong** association rules: having both minimum support and minimum confidence.

- For each frequent itemset I generate all non-empty subsets of I
- Subset s of I rule $s \rightarrow (I - s)$ if confidence $>$ min confidence.

Overview

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

1-Itemset	Sup
A	2
B	3
C	3
D	1
E	3

2-Itemsets	Sup
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

3-Itemsets	Sup
BCE	2

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$\longrightarrow$

1-Itemset	Sup
A	2
B	3
C	3
D	1
E	3

2-Itemsets	Sup
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

$\longrightarrow$

3-Itemsets	Sup
BCE	2

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

→

1-Itemset	Sup
A	2
B	3
C	3
D	1
E	3

2-Itemsets	Sup
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

→

3-Itemsets	Sup
BCE	2

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$\longrightarrow$

1-Itemset	Sup
A	2
B	3
C	3
D	1
E	3

2-Itemsets	Sup
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

$\longrightarrow$

3-Itemsets	Sup
BCE	2

Candidate Rules

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$BC \rightarrow E\ [50\%, 100\%]$
$BE \rightarrow C\ [50\%, 66\%]$
$C \rightarrow A\ [50\%, 66\%]$

Candidate Rules

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$BC \rightarrow E$ [50%, 100%]

$BE \rightarrow C$ [50%, 66%]

$C \rightarrow A$ [50%, 66%]

Candidate Rules

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$BC \rightarrow E$ [50%, 100%]
$BE \rightarrow C$ [50%, 66%]
$C \rightarrow A$ [50%, 66%]

Candidate Rules

Transaction	Items
12345	A C D
12346	B C E
12347	A B C E
12348	B E

$BC \rightarrow E$ [50%, 100%]
$BE \rightarrow C$ [50%, 66%]
$C \rightarrow A$ [50%, 66%]

Health Insurance Commision

- Associations on episode database for pathology services
  - 6.8 million records X 120 attributes (3.5GB)
  - 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
  - cmin = 50% and smin = 1%, 0.5%, 0.25%
    (1% of 6.8 million = 68,000)
  - Unexpected/unnecessary combination of services
- Refuse cover saves $550,000 per year

Health Insurance Commision

- Associations on episode database for pathology services
  - 6.8 million records X 120 attributes (3.5GB)
  - 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
  - cmin = 50% and smin = 1%, 0.5%, 0.25%
    (1% of 6.8 million = 68,000)
  - Unexpected/unnecessary combination of services
- Refuse cover saves $550,000 per year

Health Insurance Commision

- Associations on episode database for pathology services
  - 6.8 million records X 120 attributes (3.5GB)
  - 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
  - cmin = 50% and smin = 1%, 0.5%, 0.25% (1% of 6.8 million = 68,000)
  - Unexpected/unnecessary combination of services
- Refuse cover saves $550,000 per year

Health Insurance Commision

- Associations on episode database for pathology services
  - 6.8 million records X 120 attributes (3.5GB)
  - 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
  - cmin = 50% and smin = 1%, 0.5%, 0.25%
    (1% of 6.8 million = 68,000)
  - Unexpected/unnecessary combination of services
- Refuse cover saves $550,000 per year

Health Insurance Commision

- Associations on episode database for pathology services
  - 6.8 million records X 120 attributes (3.5GB)
  - 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
  - cmin = 50% and smin = 1%, 0.5%, 0.25%
    (1% of 6.8 million = 68,000)
  - Unexpected/unnecessary combination of services
- Refuse cover saves $550,000 per year

Associations in R: Apriori

Sample data—DVD purchases:

```
Sixth Sense,LOTR1,Harry Potter1,Green Mile,LOTR2
Gladiator,Patriot,Braveheart
LOTR1,LOTR2
Gladiator,Patriot,Sixth Sense
Gladiator,Patriot,Sixth Sense
Gladiator,Patriot,Sixth Sense
Harry Potter1,Harry Potter2
Gladiator,Patriot
Gladiator,Patriot,Sixth Sense
Sixth Sense,LOTR,Galdiator,Green Mile
```

Associations in R: Apriori

Using Borgelt's open source apriori C code:

```
library(arules)
tname <- file.path("data", "dvdtrans.csv")
head(read.csv(tname))

##   ID          Item
## 1  1   Sixth Sense
## 2  1         LOTR1
## 3  1 Harry Potter1
## 4  1    Green Mile
## 5  1         LOTR2
## 6  2     Gladiator

dvds <- read.transactions(tname, sep = ",", format = "single", cols = c("ID",
    "Item"))
dvds

## transactions in sparse format with
##  10 transactions (rows) and
##  10 items (columns)
```

Associations in R: Apriori

Build the model:

```
dvds.apriori <- apriori(dvds, parameter = list(support = 0.2, confidence = 0.1))


##
## parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.1    0.1    1 none FALSE            TRUE     0.2      1     10
##  target   ext
##   rules FALSE
##
## algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## apriori - find association rules with the apriori algorithm
## version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[10 item(s), 10 transaction(s)] done [0.00s].
## sorting and recoding items ... [7 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [20 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Associations in R: Apriori

View the resulting rule set.

```
inspect(sort(dvds.apriori, by = "lift"))

##    lhs              rhs               support confidence  lift
## 1  {LOTR1}       => {LOTR2}               0.2     1.0000 5.000
## 2  {LOTR2}       => {LOTR1}               0.2     1.0000 5.000
## 3  {Green Mile}  => {Sixth Sense}         0.2     1.0000 1.667
## 4  {Sixth Sense} => {Green Mile}          0.2     0.3333 1.667
## 5  {Patriot}     => {Gladiator}           0.6     1.0000 1.429
## 6  {Gladiator}   => {Patriot}             0.6     0.8571 1.429
## 7  {Patriot,
##     Sixth Sense} => {Gladiator}           0.4     1.0000 1.429
## 8  {Gladiator,
##     Sixth Sense} => {Patriot}             0.4     0.8000 1.333
## 9  {Sixth Sense} => {Gladiator}           0.5     0.8333 1.190
## 10 {Gladiator}   => {Sixth Sense}         0.5     0.7143 1.190
## 11 {Patriot}     => {Sixth Sense}         0.4     0.6667 1.111
## 12 {Sixth Sense} => {Patriot}             0.4     0.6667 1.111
## 13 {Gladiator,
##     Patriot}     => {Sixth Sense}         0.4     0.6667 1.111
## 14 {}            => {Harry Potter1}       0.2     0.2000 1.000
## 15 {}            => {LOTR1}               0.2     0.2000 1.000
## 16 {}            => {LOTR2}               0.2     0.2000 1.000
```

Overview

Predictive Models

- Predictive Models: predict an outcome based on other variables
- Association rules: associate variable values with target variable
- Basket: collection of variable values
- Target: Rain Tomorrow? Yes/No

```
{Pressure3pm < 1012}
{Sunshine < 8.85}
=> {RainTomorrow = Yes}
```

- For R, all inputs must be categoric.

Example in R

```
library(rattle)
cats <- c("WindGustDir", "WindDir9am", "WindDir3pm", "RainTomorrow")
trans <- as(weather[cats], "transactions")
mymodel <- apriori(trans, parameter = list(support = 0.1, confidence = 0.5))
```

```
##
## parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.5    0.1    1 none FALSE            TRUE     0.1      1     10
##  target   ext
##   rules FALSE
##
## algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## apriori - find association rules with the apriori algorithm
## version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[50 item(s), 366 transaction(s)] done [0.00s].
## sorting and recoding items ... [10 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [6 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(sort(mymodel, by = "confidence"))
```

```
##   lhs                 rhs                support confidence   lift
## 1 {WindDir9am=SE}  => {RainTomorrow=No}  0.1120     0.8723 1.0643
## 2 {WindDir3pm=WNW} => {RainTomorrow=No}  0.1393     0.8361 1.0200
## 3 {WindDir3pm=NNW} => {RainTomorrow=No}  0.1066     0.8298 1.0123
## 4 {}               => {RainTomorrow=No}  0.8197     0.8197 1.0000
## 5 {WindGustDir=NW} => {RainTomorrow=No}  0.1557     0.7808 0.9526
## 6 {WindDir3pm=NW}  => {RainTomorrow=No}  0.1284     0.7705 0.9400
```

Overview

Modelling Framework

Language Set of *Antecedent* $\rightarrow$ *Consequent* rules

Measure Support, confidence, lift, leverage

Search Apriori

Summary

- The “original” data mining algorithm!
- Effective in finding linkages in large customer databases.
- Considerable attention from data mining researchers.
- Available in the R package `arules` as `apriori`.

Summary

- The "original" data mining algorithm!
- Effective in finding linkages in large customer databases.
- Considerable attention from data mining researchers.
- Available in the R package `arules` as `apriori`.

~

Summary

- The “original” data mining algorithm!
- Effective in finding linkages in large customer databases.
- Considerable attention from data mining researchers.
- Available in the R package `arules` as `apriori`.

Summary

- The "original" data mining algorithm!
- Effective in finding linkages in large customer databases.
- Considerable attention from data mining researchers.
- Available in the R package `arules` as `apriori`.

Summary

- The "original" data mining algorithm!
- Effective in finding linkages in large customer databases.
- Considerable attention from data mining researchers.
- Available in the R package `arules` as `apriori`.

∼