Rule Induction

Association Rules

Unsupervised Learning

- No goal class (either Y nor G).
- We have N observations of X: $(x_1, \ldots, x_N)$, $x_i \in \{0, 1\}^p$ (some dimensions may be numeric).
- We aim to reason about $P(X)$.
- For $p < 4$ effective nonparametric methods exist.
- For p high **Curse of dimensionality** appears.
- We estimate rough global models
  - mixtures of gaussians (clustering)
  - simple statistics characterizing $P(X)$.

Curse of Dimensionality

Unit ball in p dimensions centered in the origin. Distance from the origin to closest neighbor is roughly:

$$d(p, N) = \left(1 - \frac{1}{2}^{\frac{1}{N}}\right)^{\frac{1}{p}}.$$

Unit cube, fraction of volume (ESL book):

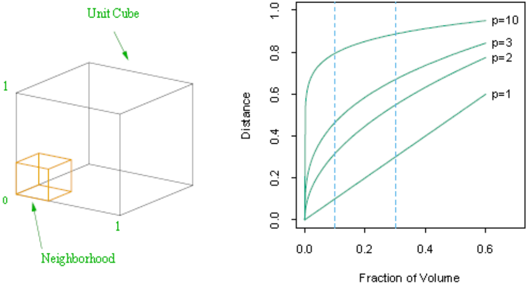


FIGURE 2.6. *The curse of dimensionality is well illustrated by a subcubical neighborhood for uniform data in a unit cube. The figure on the right shows the side-length of the subcube needed to capture a fraction r of the volume of the data, for different dimensions p. In ten dimensions we need to cover 80% of the range of each coordinate to capture 10% of the data.*

High p – Nearest Neighbor Approximation May Fail

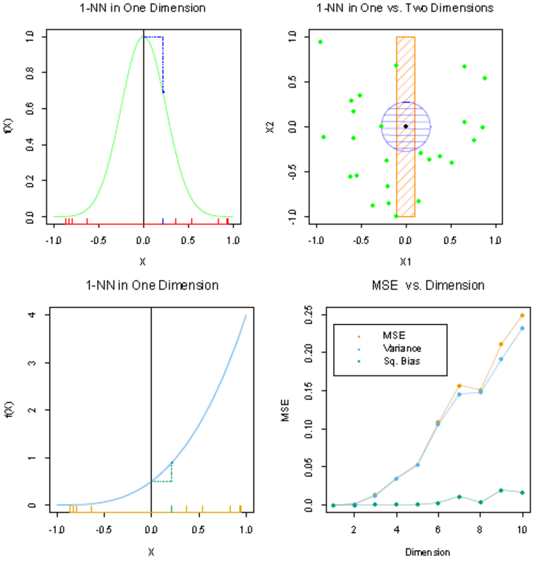


FIGURE 2.8. *A simulation example with the same setup as in Figure 2.7. Here the function is constant in all but one dimension:* $F(X) = \frac{1}{2}(X_1 + 1)^3$. *The variance dominates.*

Places with high $P(X)$

- We search places with high appearance of data samples
- using various languages
  - association rules
    - conjunctive rules
    - really many dimensions p and (usually) binary data
  - clustering *(last week)*
    - means of clusters, list of gaussians
    - usually continuous features.

Association Rules

- Usually binary data $X_{ij} \in \{0, 1\}^{N \times p}$
- Value $= 1$ is our interest; for example purchase.
- p may be very large; for example the size of sortiment of an market.
- Popular application: Market basket analysis.
- Generally: We look for L prototypes $v_1, \ldots, v_L \in X^p$ such that $P(v_\ell)$ is relatively large.
- 'Bump hunting' may be used (gradient search).
- With large p, we do not have enough data to estimate $P(v_\ell)$ since number of observations with $P(X = v_\ell)$ is too small.
- We seek for regions where $P(x)$ is large, that can be written as conjunctive rule on dimension conditions $\bigcap_{j=1}^{p}(X_j \in s_j)$ where s_j are selected values of the feature X_j.

Hypothesis space for Apriori

ESL book Figure:

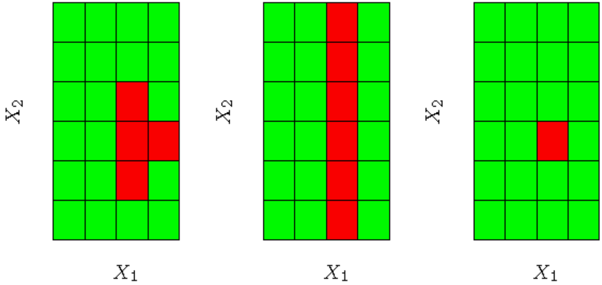


FIGURE 14.1. *Simplifications for association rules. Here there are two inputs X_1 and X_2, taking four and six distinct values, respectively. The red squares indicate areas of high density. To simplify the computations, we assume that the derived subset corresponds to either a single value of an input or all values. With this assumption we could find either the middle or right pattern, but not the left one.*

Market Basket Analysis

- For very large datasets, $p \approx 10^4, N \approx 10^8$; in unit ball is the distance to the nearest neighbour ≈ 0.9981.
- Simplifications: Test on feature X_j either equal to a specific value or no restriction at all,
- I select combinations of items with higher number of occurences (**support**) than predefined threshold t.
- I select all combinations fulfilling conditions above.
- Categorical variables may be codded by dummy variables in advance (if not too many).

Apriori Algorithm

1. Create list of candidates, one–element subsets of the feature space, for example: $\{bread\}$ meaning $X_{bread} = 1$.
2. For each candidate count support in data.
3. Discard candidates with support less than t (predefined threshold).
4. For each length $i = 2, \ldots$
   - 4.1 Generate list of candidates of the length i.
     Join any two candidates from previous step having $i - 2$ elements common. *(More prunning possible.)*
   - 4.2 For each candidate, count support in data.
   - 4.3 Discard candidates with support $< t$.
5. Until empty list of candidates.

Properties of the Apriori Algorithm

- Applicable for very large data (with high treshold t).
- The key idea:
  - Only few of 2^K combinations have high support $> t$,
  - **subset of high–support combination has also high support**.
- The number of passes through the data is equal to the size of the longest supported combination. The data does not to be in memory simultaneously.

Association Rules !

- From each supported itemset $\mathcal{K}$ found by Apriori algorithm we create a list of **association rules**, implications of the form $A \Rightarrow B$ where:
  - A, B are disjoint and $A \cup B = \mathcal{K}$
  - A is called **antecedent**
  - B is called **succedent (consequent)**.
- Support of the rule $T(A \Rightarrow B)$ is defined as support of the itemset $\mathcal{K}$, that is support of the conjunction $A \& B$.

Precision, Lift of a Rule !

There are two important measures for a rule $A \Rightarrow B$:

- **Confidence** (predictability, přesnost)

$$C(A \Rightarrow B) = \frac{T(A \Rightarrow B)}{T(A)}$$

  that is an estimate of $P(B|A)$,
- Expected precision $T(B)$ is an estimate of $P(B)$,
- **Lift** is the ration of confidence and expected precision:

$$L(A \Rightarrow B) = \frac{C(A \Rightarrow B)}{T(B)}$$

  that is an estimate of $\frac{P(A\&B)}{P(A) \cdot P(B)}$.

Association Rule Example

ESL book example:

Association rule 2: Support 13.4%, confidence 80.8%, and lift 2.13.

$$\left[\begin{array}{rcl} \text{language in home} & = & \textit{English} \\ \text{householder status} & = & \textit{own} \\ \text{occupation} & = & \{\textit{professional/managerial}\} \end{array}\right]$$

$$\Downarrow$$

$$\text{income} \geq \textit{\$40,000}$$

- $\mathcal{K} = \{$English, own, pref/man, income$>$\$40000$\}$,
- 13.4% people has all four properties,
- 80.8% of people with $\{$English, own, pref/man$\}$ have income$>$ \$40000,
- $T(\text{income} > \$40000) = 37.94\%$, therefore $\textit{Lift} = 2.13$.

The Goal of Apriori Algorithm !

- Apriori finds all rules with high support.
- Frequently, it finds thousends of rules.
- We usually select lower treshold c on confidence, that is we select rules with $T(A \Rightarrow B) > t$ and $C(A \Rightarrow B) > c$.
- Conversion of itemsets to rules is usually relatively fast compared to seach of itemsets.
- See lispMiner for user interface and a lot of more.

Demographical Data ESL Example

Feature	Demographic	# Values	Type
1	Sex	2	Categorical
2	Marital status	5	Categorical
3	Age	7	Ordinal
4	Education	6	Ordinal
5	Occupation	9	Categorical
6	Income	9	Ordinal
7	Years in Bay Area	5	Ordinal
8	Dual incomes	3	Categorical
9	Number in household	9	Ordinal
10	Number of children	9	Ordinal
11	Householder status	3	Categorical
12	Type of home	5	Categorical
13	Ethnic classification	8	Categorical
14	Language in home	3	Categorical

Demographical Example – Continuing

- $N = 9409$ questionmarks, the ESL authors selected 14 questions.
- Preprocessing:
  - `na.omit() remove records with missing values,`
  - `ordinal features cut by median to binary,`
  - `for categorical create dummy variable for each category.`
- Apriori input was matrix 6876×50.
- Output: 6288 association rules
  - with max. 5 elements
  - with support at least 10%.

Negated Literals – Useful, Problematic

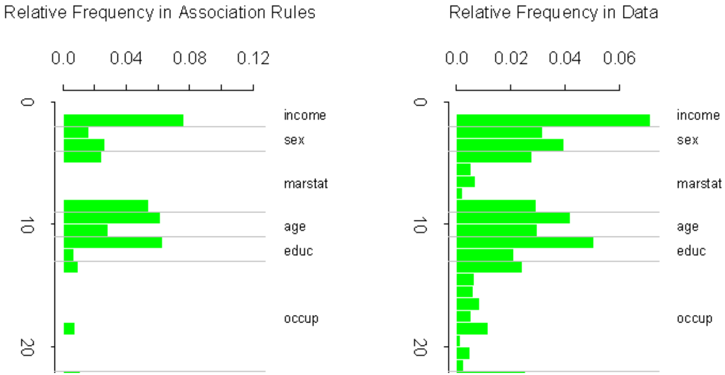
Association rule 3: Support 26.5%, confidence 82.8% and lift 2.15.

$$\left[\begin{array}{rcl} \text{language in home} & = & \textit{English} \\ \text{income} & < & \textit{\$40,000} \\ \text{marital status} & = & \textit{not married} \\ \text{number of children} & = & \textit{0} \end{array}\right]$$

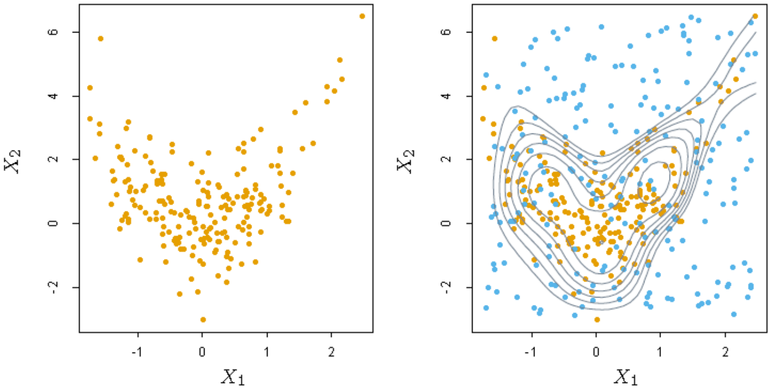
$$\Downarrow$$

$$\text{education} \notin \{\textit{college graduate, graduate study}\}$$

Non–frequent Values Dissapear



Unsupervised Learning as Supervised Learning



- We add additional attribute Y_G.
- $Y_G = 1$ for all our data.
- We generate randomly a dataset of simillar size with uniform distribution, set $Y_G = 0$ for this artificial data.
- The task is to separate $Y_G = 1$ and $Y_G = 0$.

Generalize Association Rules

- We search for high lift, where probability of conjunction is greater than expected.
- Hypothesis is specified by column indexes j and subsets of values s_j corresponding features X_j. We aim:

$$\hat{P}\left(\bigcap_{j\in\mathcal{J}}(X_j \in s_j)\right) >> \frac{1}{N}\sum_{1}^{N} I\left(\bigcap_{j\in\mathcal{J}}(x_{ij} \in s_j)\right)$$

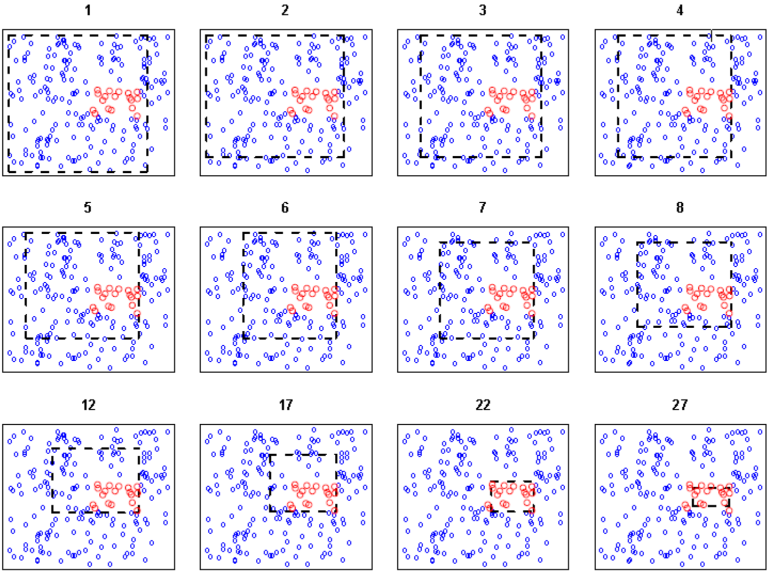
- On the data from previous slide, CART (decision tree alg.) or PRIM ('bump hunting') may be used.
- Figure on previous slide: Logistic regression on tensor product of natural splines.
- Other methods may be used. All are heuristics compared to full evaluation by Apriori.

PRIM = Bump Hunting
Patient Rule Induction Method

- We search iteratively areas with high Y; for each area we create a rule
- CART after approximately $\log_2(N) - 1$ steps looses all data, PRIM can afford approximately $-\frac{\log(N)}{\log(1-\alpha)}$ steps.
  For $N = 128$ and $\alpha = 0.1$ it is 6 and 46 resp. 29, since data counts must be whole numbers.

PRIM Patient Rule Induction Method !

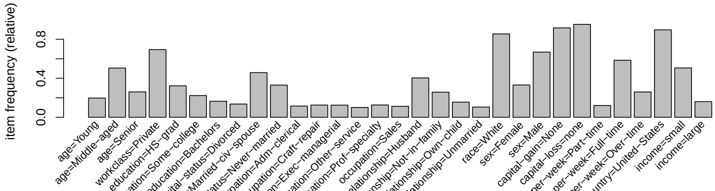
1. Take all data and full feature space, $\alpha = 0.05$ or 0.10
2. Find X_j and its upper or lower boundary such that the removal of $\alpha \cdot 100\%$ observations gives maximum increase of the overall mean of the remaining area.
3. Repeat 2. until at least 10 observations remain.
4. Extend the area in any direction if it increases the mean.
5. From the above list of areas, select the one ($\#$ of observations) best by crossvalidation. We call this area B_1.
6. Remove data in B_1 from the dataset and continue 2 to 5, create B_2, . . . , until a stop criterion.


1
2
3
4
5
6
7
8
12
17
22
27

R Code Example

```
library("arules")
data("AdultUCI")
#48842 rows (elements/itemsets/transactions) and
# 115 columns (items) and a density of 0.1089939
itemFrequencyPlot(Adult, support = 0.1, cex.names=0.8)
rules <- apriori(Adult, parameter = list(support = 0.01,
confidence = 0.6))
#set of 276443 rules
rulesIncomeSmall <- subset(rules, subset = rhs %in%
"income=small"& lift > 1.2)
inspect(head(rulesIncomeSmall, n = 1, by = "confidence"))
#lhs rhs support confidence lift
# 1 marital-status=Married-civ-spouse,
# capital-gain=High,

# native-country=United-States => income=large 0.01562180 0.6849192 4.266398
```



MARS Multivariate Adaptive Regression Splines

- generalization of linear regression and decision trees CART
- for each feature and each data point we create a **mirror pair** of basis functions
- $(x - t)_+$ and $(t - x)_+$ where $+$ denotes non–negative part, minimum is zero.
- we have the set of functions

$$\mathcal{C} = \{(X_j - t)_+, (t - X_j)_+\}_{t \in \{x_{1,j}, x_{2,j}, \ldots, x_{N,j}\}, j=1,2,\ldots,p}$$

- that is $2Np$ functions for non–duplicated data points.

MARS – continuation

- our model is in the form

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X)$$

  where $h_m(X)$ is a function from $\mathcal{C}$ or a product of any amount of functions from $\mathcal{C}$
- for a fixed set of h_m's we calculate coefficients β_m by usual linear regression (minimizing RSS)
- the set of functions h_m is selected itterativelly

MARS – basis selections

- We start with $h_0 = 1$, we put this function into the model $\mathcal{M} = \{h_0\}$.
- We consider the product of any member $\mathcal{M}$ with any pair from $\mathcal{C}$

$$\hat{\beta}_{M+1} h_\ell(X) \cdot (X_j - t)_+ + \hat{\beta}_{M+2} h_\ell(X) \cdot (t - X_j)_+, h_\ell \in \mathcal{M}$$

  we select the one minimizing training error RSS (for any product candidate, we estimate $\hat{\beta}$).
- Repeat until predefined number of functions in $\mathcal{M}$
- The model is usually overfitted. We select (remove) iterativelly the one minimizing the increase of training RSS. We have a sequence of models $\hat{f}_\lambda$ for different numbers of parameters λ.
- (we want to speed–up crossvalidation for computational reasons)
- we select λ (and the model) minimizing **generalized crossvalidation**

$$GCV(\lambda) = \frac{\sum_{i=1}^{N} (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}.$$

- where $M(\lambda)$ is the number of effective parameters, the number of function h_m (denoted r) plus the number of knots K, the authors suggest to multiply K by 3: $M(\lambda) = r + 3K$.

Decision trees from MARS

- The MARS is related with standard decision tree algorithm.
- We replace piecewise linear functions with piecewise constant functions $I(x - t > 0)$ and $I(x - t \leq 0)$
- If a function h_m is used in a product we remove it from the model. Therefore, it is used maximally once: we get binary structure of the tree.
- and we have the standard decision tree algorithm (ID3, CART).