Learning Rules With Numerical and Categorical Attributes from Linked Data Sources

Andre de Oliveira Melo

Saarland University

andresony@gmail.com

February 5, 2013

Overview

1. Introduction
2. Motivation
3. Inductive Logic Programming
4. Learning Rules With Categorical Attributes
5. Experiments

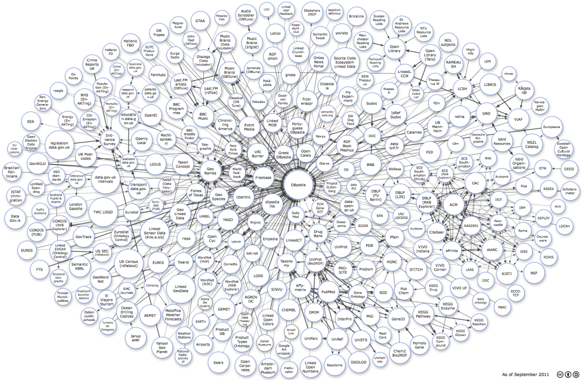
Semantic Web

"provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries"

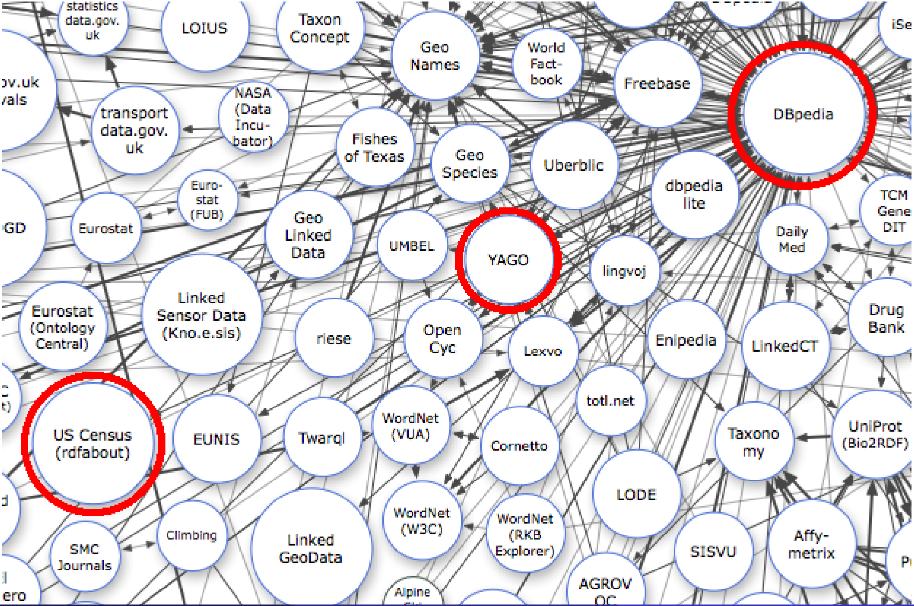
Linked Data

“a term used to describe a recommended best practises for exposing, sharing, and connecting pieces of data, information and knowledge on the Semantic Web using URIs and RDF”

“collection of interrelated datasets on the Web”



Linked Data


statistics data.gov.uk
LOIUS
Taxon Concept
Geo Names
World Fact-book
Freebase
DBpedia
transport data.gov.uk
NASA (Data Incu-bator)
Fishes of Texas
Geo Species
Uberblic
dbpedia lite
Euro-stat (FUB)
Eurostat
Geo Linked Data
UMBEL
YAGO
lingvoj
Daily Med
TCM Gene DIT
Eurostat (Ontology Central)
Linked Sensor Data (Kno.e.sis)
riese
Open Cyc
Lexvo
Enipedia
LinkedCT
Drug Bank
totl.net
US Census (rdfabout)
EUNIS
Twarql
WordNet (VUA)
Cornetto
Taxono my
UniProt (Bio2RDF)
LODE
Climbing
SMC Journals
Linked GeoData
WordNet (W3C)
WordNet (RKB Explorer)
SISVU
Affy-metrix
AGROV OC
Alpine

Motivation

Learn Datalog rules from data:

$$\underbrace{livesIn(X, Y)}_{head} \text{ :- } \underbrace{isMarriedTo(X, Z), livesIn(Z, Y)}_{body}$$

Support and confidence thresholds

- Support: $supp(head \text{ :- } body) = supp(head \wedge body)$
- Confidence: $conf(head \text{ :- } body) = \dfrac{supp(head \wedge body)}{supp(body)}$

Motivation

Refining rules with constants is relevant

- **Base-rule**: Numerical argument with no constant
  $marritalStatus(X, single)$:- $hasIncome(X, Y)$
- **Refined-rule**: Base-rule with numerical variable set to a specific interval
  $marritalStatus(X, single)$:- $hasIncome(X, Y), Y \leq 30k$

Combining with categories is also relevant

$marritalStatus(X, single)$	:-	$livesIn(X, mississipi), hasIncome(X, Y)$
$marritalStatus(X, single)$	:-	$livesIn(X, mississipi), hasIncome(X, Y), Y \leq 15k$
$marritalStatus(X, single)$	:-	$livesIn(X, newYork), hasIncome(X, Y)$
$marritalStatus(X, single)$	:-	$livesIn(X, newYork), hasIncome(X, Y), Y \leq 70k$
$marritalStatus(X, single)$	:-	$livesIn(X, texas), hasIncome(X, Y)$
$marritalStatus(X, single)$	:-	$livesIn(X, texas), hasIncome(X, Y), Y \leq 31k$

Motivation

- Learning for numerical point constants usually doesn't make sense
  $marritalStatus(X, single)$:- $hasIncome(X, 18324)$
- Even if it satisfies all thresholds
  $speaks(X, portuguese)$:- $livesIn(X, Z), hasPopulation(Z, 193946886)$
  $speaks(X, portuguese)$:- $livesIn(X, brazil)$
- Search for intervals instead
  $marritalStatus(X, single)$:- $hasIncome(X, Y), Y \leq 30k$
- Query support and confidence distribution over Y and search for intervals that satisfy the thresholds

Motivation

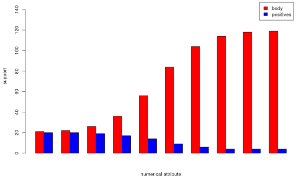
For learning rules with numerical constants, we are more interested in base-rules that:

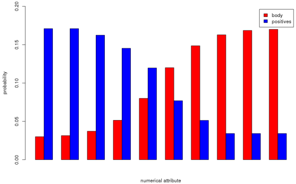
- Satisfy support threshold
- Do not necessarily satisfy confidence threshold
- Potentially has a refined-rule with an interval that satisfies both thresholds

  i.e., has non-uniform confidence distribution along the numerical attribute

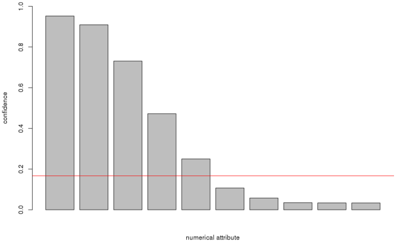
  i.e., has body support and positives (body$\wedge$head) support distributions are different

Example for a rule with positives support 117 and body support 700:





→



Motivation

Problem?

- Search space grows exponentially with the number of predicates and constants
- Querying support and confidence distributions is very expensive

Idea:

- Analyze combinations of numerical and categorical properties
- Measure their level of interestingness
- Use core-ILP algorithm
- Extend it to focus on most interesting combinations

ILP

Inductive Logic Programming: Finds a hypothesis H that covers all positive, and no negative examples

$$positiveExamples + negativeExamples + backgroundKnowledge \rightarrow hypothesis$$

Training Examples	**Background Knowledge**
daughter(mary,ann) +	parent(ann,mary)
daughter(eve,tom) +	parent(ann, tom)
daughter(tom,ann) -	parent(tom,eve)
daughter(eve,ann) -	parent(tom,ian)
	female(ann)
	female(mary)
	female(eve)

$$hypothesis = daughter(X, Y) \text{ :- } female(X), parent(Y, X)$$

ILP

Important concepts:

- Literal: predicate symbol with bracketed n-tuple, e.g:
  $L = \mathit{livesIn}(X, Y)$
- Clause: a disjunction of literals (negated or not), e.g:
  $c = (L_1 \vee L_2 \vee \ldots \vee \neg L_{m-1} \vee \neg L_m)$
- Safe Datalog Rules: Every variable in the head appear in a non-negated literal in the body, negated literal variables in the body should appear in some positive literal in the body, e.g.:
  $\mathit{speaks}(X, Y)$:- $\mathit{wasBornIn}(X, Z), \mathit{hasOfficialLanguage}(Z, Y)$
- Hypothesis: a set of clauses H
- Completeness: H covers all positive examples
- Consistency: H covers no negative examples

ILP

Approaches

- Bottom-up: Start with least general H then perform generalizations
- **Top-down**: Start with most general H then perform specializations
  - Specialization loop: adds literals to a clause and ensures consistency
  - Covering loop: adds clauses to the hypothesis and ensures completeness
  - Anti-monotonic Support: refinements in the clause cannot increase the support
  - Apriori-style pruning

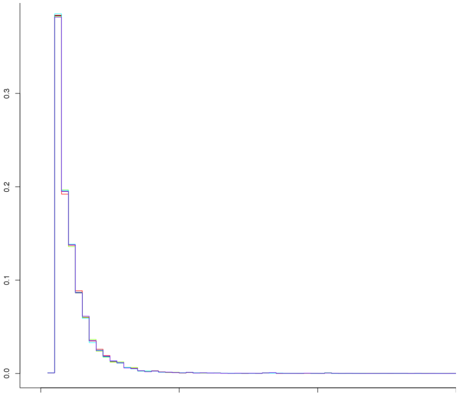
What about large, noisy and incomplete Linked Open Data (LOD) datasets such as YAGO and DBpedia?:

- Sample data to reduce size
- Restrict the number of literals in a clause
- Tolerate a certain level of inconsistency and incompleteness

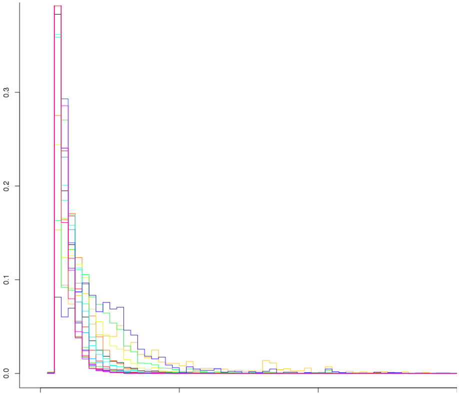
Correlation between Literals

Let's say we want to refine a clause with *hasIncome*(X, Y) with an interval for Y. What property is more interesting to add to the clause body:

hasIncome(X, Y), *quarterOfBirth*(X, Z)



hasIncome(X, Y), *hasEducation*(X, Z)



Correlation between Literals

USCensus constants for Z in $hasEducation(X, Z)$

N/A (less than 3 years old)	High school graduate
No school completed	Some college, less than 1 year
Nursery school to grade 4	One or more years of college, no degree
Grade 5 or grade 6	Associate's degree
Grade 7 or grade 8	Bachelor's degree
Grade 9	Master's degree
Grade 10	Professional school degree
Grade 11	Doctorate degree
Grade 12 no diploma	

Correlation between Literals

How to measure of the interestingness of adding a literal l to a clause c?

- Extract a frequency histogram of the support distribution of c and $(c \wedge l)$
- Normalize both support distributions and measure their divergence (e.g., with Kullback-Leibler)

But, divergence alone isn't a good idea because:

- Lower support histograms are more likely to have a divergent distribution (sampling error)
- It's still interesting to have rules with high support

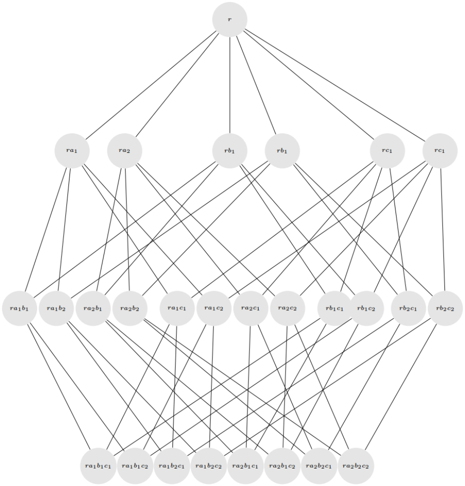
Then combine both measures: divergence*support

Correlation Lattice

- Build a lattice similar to an itemset lattice
- Numerical property r as root
- The "items" are literals that can be joined with the root's non-numerical variable X
- Root's numerical attribute domain is discretized in k buckets $b_1, \ldots, b_k$
- Each node x has a frequency histogram $h(x) =< h_1(x), \ldots, h_k(x) >$ from its clause support distribution

  where $h_i(x) = supp(x|Y \in b_i)$ and $|h(x)|_1 = supp(x)$

Correlation Lattice



$$
\begin{aligned}
r &= hasIncome(X, Y) \\
a_1 &= hasSex(X, male) \\
a_2 &= hasSex(X, female) \\
b_1 &= employmentStatus(X, employed) \\
b_2 &= employmentStatus(X, unemployed) \\
c_1 &= hasDeficiency(X, yes) \\
c_2 &= hasDeficiency(X, no)
\end{aligned}
$$

Correlation Lattice

- Number of nodes in a lattice with ℓ levels n properties and m constants per property:

$$\sum_{i=1}^{\ell} \binom{nm}{i} \tag{1}$$

- Too expensive, we need to reduce size
  - prune by support (safe)
- If not sufficient, we can restrict the literals to be added in the lattice in order to reduce n and m
- As pruning heuristics, we can greedily prune it by interestingness, as well as independence checks (will be discussed later)

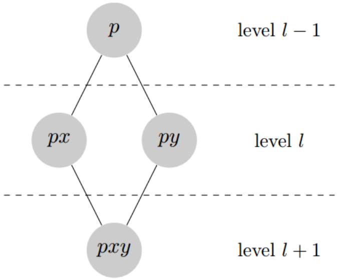
Correlation Lattice

Literal Restrictions

- Lattice literals should directly join with root's join variable X. For example if root is $hasIncome(X, Y)$ we could have:
  $livesIn(X, Z)$
  $hasChild(X, no)$
- Literals that don't directly join with root should be combined with a linking property, e.g.:
  $wasBornIn(X, W), hasOfficialLanguage(W, Z)$
  $votedFor(X, W), isAffiliatedTo(W, labourParty)$
  $isFatherOf(W, X), diedIn(W, Z)$
- This can be used to enable integration with different datasets, e.g.:
  $owl{:}sameAs(X, W), directed(W, Z)$

Independence checks

Checks if a pair of nodes joining nodes are independent given their common parent



p is a clause containing the root literal and $(l-1)$ other literals

x and y are literals, s.t. $x \neq y$ and $x, y \notin p$

- Estimate $\hat{h}(pxy)$ assuming that x and y are independent given p
- Query actual $h(pxy)$ and perform a Pearson's chi-squared test

  $H_0 = x$ and y are independent given p

  $H_1 = x$ and y are dependent given p

Independence checks

$hasIncome(X, Y)$
$h =< 6, 10, 8, 4, 2, 1 >$

$hasIncome(X, Y), marritalStatus(X, single)$
$h =< 4, 5, 4, 3, 1, 0 >$

$hasIncome(X, Y), employmentStatus(X, employed)$
$h =< 3, 6, 6, 4, 2, 1 >$

$hasIncome(X, Y), marritalStatus(X, single), employmentStatus(X, employed)$
$h =< 2, 4, 2, 4, 1, 0 >$
$\hat{h} =< 2, 3, 3, 3, 1, 0 >$

$$\chi^2 = \sum_{i=1}^{k} \frac{(h_i - \hat{h}_i)^2}{\hat{h}_i} = 1 \quad \Rightarrow \quad \text{p-value} = 0.96$$

Independence checks

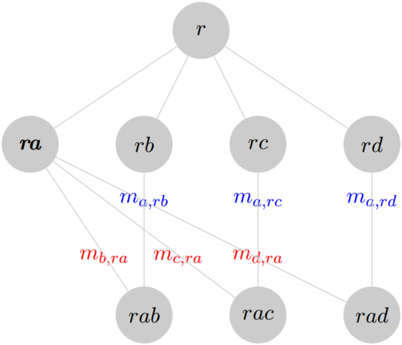
- If there's not enough evidence of dependence, we assume independence, then:

  x :- $p, y \equiv x$:- p

  y :- $p, x \equiv y$:- p
- The greater the χ^2 value, the greater the evidence that x and y are dependent given p, therefore the more interesting it is to join both py and px
- As heuristics, we can prune a node when it's independent for all possible pairs of joining parents. That means its not interesting to any of its parents.

Search in the Lattice

In the refinement loop from the core-ILP, the clauses have a fixed head and literals are added to the body. In the following example, we would have a as head literal, r as root and b, c, d as possible new literals



$$a \;\left|\; \begin{array}{l} b = m_{a,rb} \\ c = m_{a,rc} \\ d = m_{a,rd} \end{array}\right.$$

What has to be done?

- Search the node with body literals
- For each child of such node check head literal can be further added, if so collect the new literal and the interestingness value of adding the head
- Sort the possible new literals by interestingness

Alternative?

- Create mapping in every node with the possible head literals as key and sorted literals to be added to body as value, e.g. for the node ra_1b_1 in the lattice example:

$$\begin{array}{c|l} a_1 & c_1 \; [m_{a_1, rb_1c_1}] \\ & c_2 \; [m_{a_1, rb_1c_2}] \\ \hline b_1 & c_1 \; [m_{b_1, ra_1c_1}] \\ & c_2 \; [m_{b_1, ra_1c_2}] \end{array}$$

- Only add entry if head and new literal not independent given body

Incorporating the Lattice in the Core-ILP

In the refinement step, we detect clauses with body containing a lattice root

- If clause satisfies support threshold and does not satisfy confidence threshold
- Then search in lattice for body literals and head
- Check the interestingness of adding the head to the body and analyze whether to search for numerical intervals
- Query the lattice for suggestions of interesting literals to be added to the clause

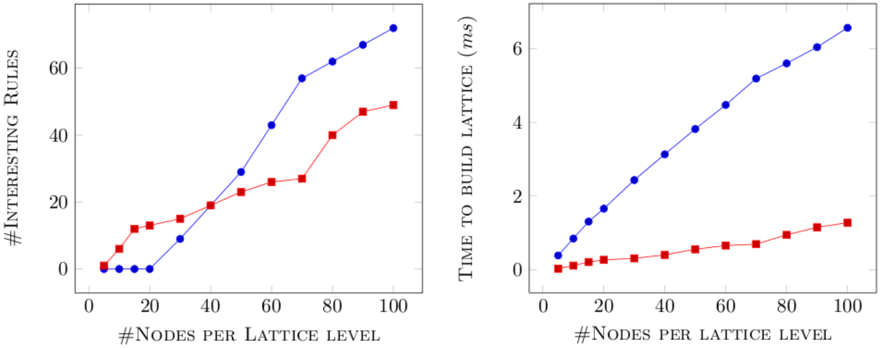
Experiments

Some experiments done so far with USCensus

- All data joined by person only (anonymized)
- All properties categorical (categories as literals)
- Not densely linked to other datasets

Compare interestingness measures:

- divergence*support vs. support only



Thank you