Lecture 17
Logs and Data Mining

some slides thx to Dan Weld

Administration

- Review:
  - PA7 out yesterday
    - (a day late; due date extended a day, too)
  - FINAL PROJECT now available online. Check it out!
  - Midterm #2 a week from Monday (March 29)

2

Web Logs and Data Mining

- Activity logs, often from web servers, are an absurdly rich source of information
- People have examined logs to learn all sorts of things
  - System failures
  - Security intrusions
  - Buying habits
  - Spelling habits
  - Web surfing behavior
  - Impending disease

3

Web Logs and Data Mining

- Many data mining projects are ethically and politically contentious
  - Credit card offers
  - Financial trades
  - The TIA project
- They also have terrifying and/or hilarious logos
- Many data-mining projects are ethically complicated because of the data used
  - Is the privacy-leaking AOL data OK?
  - "Phishing for the greater good"?

4

Data Mining in 60 Seconds

- Data mining (aka machine learning, aka statistical methods) predates the Web, works fine on non-Web data
- But the Web throws off lots of easily-processible human-activity data
  - A natural target for mining
  - Every successful Web company mines everything all the time
- What kinds of data do you throw off?
- How much have you already generated today?

5

Overview

- Data mining can easily take many, many classes
  - Totally absurd to teach it in one class
  - But data mining endemic in Web endeavors
- We'll give an overview of field, plus one technique in detail
  - Enough to know your known unknowns

6

Types of Learning

- Supervised learning
  - Training data that includes desired outputs
  - "Predict the opening price for GOOG"
  - "Predict likeliest next product purchase"
  - Recommender systems use a form of supervised learning
- Unsupervised learning
  - Training data doesn't have desired outputs
  - "Show the natural clusters in the data"

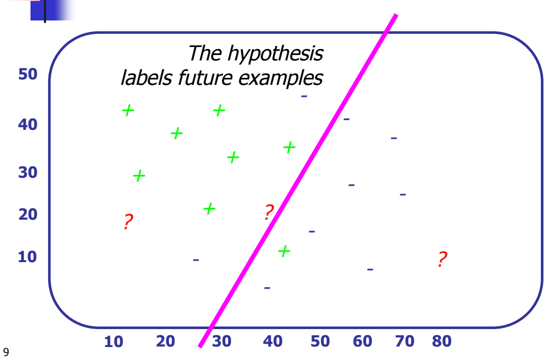
7

Supervised Learning

- Inductive learning, or "prediction"
  - Given examples of a fn (X, F(X)) predict F(X) for a novel value X
- Classification
  - F(X) is discrete; is page relevant or not?
- Regression
  - F(X) is continuous; value of GOOG?
- Probability estimation
  - F(X) is probability of X; will Obama win?

8

Binary classification; 2 inputs



9

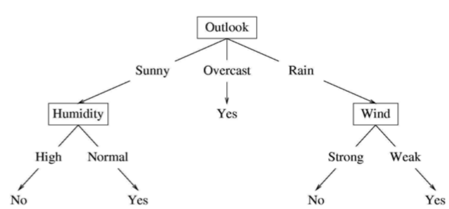
Bias

- Which hypotheses will be considered?
  - Lines?
  - Lines that are perpendicular to axes?
  - Circles?
  - Conic sections?
- Which hypotheses do you prefer?
  - Simple ones or correct ones or …?

10

Classifiers

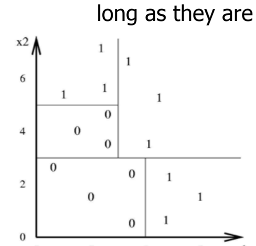
- Lots of classifier algorithms possible
- Decision Trees
  - Build a tree in which input variables are at internal nodes; outputs at leaves

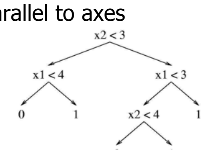


11

Classifiers

- Lots of classifier algorithms possible
- Decision Trees
  - Can express multiple decision regions, as long as they are parallel to axes





12

Classifiers

- Lots of classifier algorithms possible
- Rule Learners build a series of rules that are conjunctions of tests on input variables
  - Overcast => Yes
  - Sunny & Humid => Yes
  - Sunny & Normal => No
  - Rain & Strong-Rain => No
  - Rain & Weak-Rain => Yes
- Trees can always be converted to rules
- Vice-versa, as long as variables can appear multiple times in tree

13

Experimental Evaluation

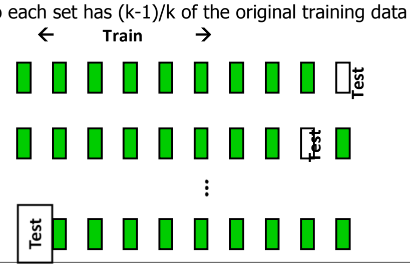
- How do we estimate the performance of classifier on unseen data?
- Can't just look at accuracy on training data – this will yield an over optimistic estimate of performance
- Solution: Cross-validation
  - Sometimes called estimating how well the classifier will generalize

14

14

Evaluation: Cross Validation

- Partition examples into *k* disjoint sets
- Now create *k* training sets
  - Each set is union of all equiv classes ***except one***
  - So each set has (k-1)/k of the original training data



15

Cross-Validation (2)

- Leave-one-out
  - Use if < 100 examples (rough estimate)
  - Hold out one example, train on remaining examples
- 10-fold
  - If have 100-1000's of examples
- M-of-N-fold
  - Repeat M times
  - Divide data into N folds, do N fold cross-validation

16

One Algorithm in Depth

- *Association rules* predict members of a set, given other members of set
  - Given i_0, i_1, … in cart, what else is in cart?
  - E.g., {diapers} => {beer}
  - (Unfortunately, this data mining urban legend appears not to be true)
  - Very popular in Web scenarios
- *Apriori algorithm* is most famous technique

17

Apriori Overview

- Assume database that lists txs and sales info

Tx	Apple	Beer	Chips	DCoke
1	1	1	0	0
2	0	1	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	0	0

- Support of X is % of data that contains X
  - Supp(Beer)= 0.8
- Supp(X=>Y) is the support of union of lhs and rhs
  - Supp(Apple,Beer=>Chips) = 0.2
- Support measures statistical significance in log data

18

Apriori Overview

- OK, easy, right?
- We might have 10ks of products, and millions of txs
- How to efficiently find all rules with at least minimum support in data?

19

Apriori Overview

- An *itemset* is a set of elts in txs
  - A *large itemset* is one with at least *minimum* support
  - Given a *large itemset*, we can fairly easily compute all the rules implied by it
- Problem is finding *large itemsets*
  - Apriori uses a breadth-first approach to limit the amount of necessary work
  - Critical observation: adding an elt to itemset can never increase support

20

Algorithm

- Apriori iterates repeatedly
- For each iteration, it has steps:
  - **Pass**: Going over data, builds up a set of candidate itemsets C. Each itemset adorned with "count"
  - **Consolidate**: Then adds some of those candidates to output set. Also, uses some to build next round of candidates.
- Terminates when consolidate step runs out of starting points for next round

21

Pass step

- **F** init'ed with empty set or prev round
- Forall tuples t do
  forall itemsets f in F do
  C_f = sets which extensions of f & in t
  forall c_f in C_f do
  if $c_f \in C$
  c_f.count++;
  else
  c_f.count=0
  $C_f = C + c_f$
- How do we decide what goes into C_f?

22

Consolidate step

- Clear contents of F
- Forall itemsets c in C do
  if count(c)/dbsize > minsupport then
  // Output itemset c
  if c should be in frontier set F then
  F = F + c
- How do we decide what goes into frontier?

23

Questions Remaining

- **Pass**: How do we decide what goes into C_f?
- **Consolidate**: How do we decide what goes into frontier F?

24

Adding to C_f

- For each database tuple t, we are adding possible itemsets that are present in the tuple
- Could test every possible subset of t
  - Requires 2^m counters, m is #items
  - But note: most of the 2^m will be small
  - Lots of wasted comparison work

25

Adding to C_f

- For each database tuple t, we are adding possible itemsets that are present in the tuple
- Another approach: kth pass, only consider sets that are of size k
  - Frontier set F is always the "large-enough" emitted itemsets
  - New itemsets are just 1-elt extensions to frontier set F
  - Some few large itemsets will get through; this approach leads to very many passes

26

Adding to C_f

- For each database tuple t, we are adding possible itemsets that are present in the tuple
- We want few passes, with few comparisons
- Solution: mixture of above
  - Consider all extensions of frontier itemsets (even multi-elt ones) that are "expected to be large"
  - Also, consider all extensions of frontier itemsets that are single-elt but considered to be small

27

Adding to C_f

- Consider frontier set {AB}, and tuple t=ABCDF
  - ABC, expected large; keep extending
  - ABCD, expected small; stop extending
  - ABCF, expected large; cannot extend further
  - ABD, expected small; stop extending
  - ABF, expected large; cannot extend further
- How do we estimate size?
  - Use 1-elt item frequencies and independence assumption
  - IA: joint prob is product of marginal probs
  - Freq of X+Y=f(I0) x f(I1) x … (x-c)/dbsize

28

Adding to Frontier

- Only have to add itemsets that were expected to be small, but turned out to be large
  - If itemset was expected to be large, all its extensions have already been considered; no need to include
  - If itemset was genuinely small, then no extensions can enjoy > minsupport

29