A Bayesian Method for Guessing the Extreme Values in a Data Set

Mingxi Wu, Chris Jermaine

University of Florida

September 2007

Outline

- Problem definition
- Example applications
- A Bayesian approach
- Experiment and conclusion

Problem definition

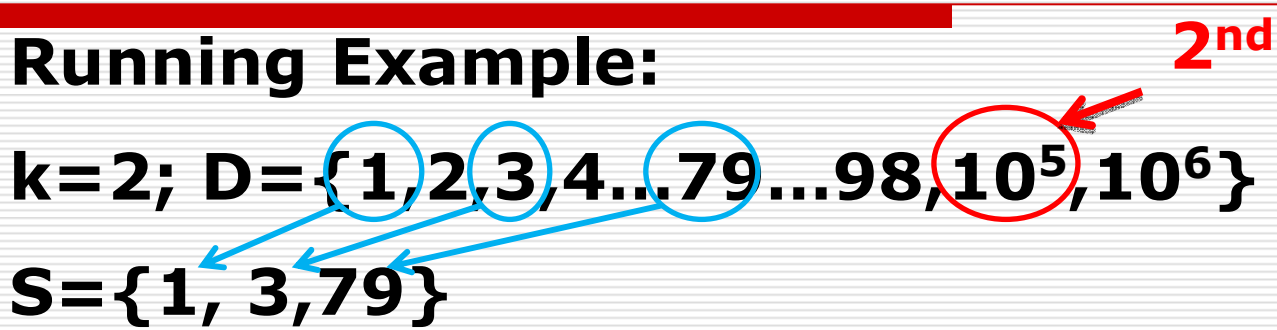
Given a rank k and a finite data set D (each data point in D maps to a real value)

Problem:

Can we use a random sample without replacement from D to predict the ***kth largest/smallest*** *value in the entire data set?*

Problem definition (cont.)

Running Example:



Can we use S to predict 10^5?

- Very hard when D is a query result set, which is produced by applying a selection predicate and an arbitrary scoring function to each record in a database

Outline

- ☑ Problem definition
- ☐ Example applications
- ☐ A Bayesian approach
- ☐ Experiment and conclusion

Example Applications

- Data mining
  - Outlier detection
  - Top-k patterns mining
- Database management
  - Min/max online aggregation
  - Top-k query processing
  - Query optimization
  - Distance join
- Any research with keywords
  - Top/kth/max/min/rank/extreme…

Outline

- ☑ Problem definition
- ☑ Example applications
- ☐ A Bayesian approach
- ☐ Experiment and conclusion

A Natural Estimator

Running Example:

2nd

k=2; D={1,2,3,4…79…98,10^5,10^6}

S={1, 3,79}

Can we use S to predict 10^5?

- Best "guess" with just S.

$$\frac{k}{|D|}=\frac{k'}{|S|}\rightarrow k'=\left\lceil\frac{k}{|D|}\times|S|\right\rceil\rightarrow k'=\left\lceil\frac{2}{100}\times 3\right\rceil=1$$

Our estimator is the (**k')th largest value** in the sample S

A Natural Estimator (cont.)

- In general, $(k')^{th}$ largest in S may be much smaller (or even larger) than k^{th} largest in D
- The relationship between $(k')^{th}$ largest in S and k^{th} largest in D varies from data set to data set
- **Key strategy** is that we want to characterize

the distribution of ratio $k^{th}/ (k')^{th}$

A Natural Estimator (cont.)

- Given distribution on the ratio of $\boldsymbol{k^{th}/\ (k')^{th}}$, deriving bounds on k^{th} largest value in D becomes easy
  - For example, if there is 95% chance the ratio is between *l* and *h*, then there is 95% chance the k^{th} largest in D is between $l\ x\ (k')^{th}$ and $h\ x\ (k')^{th}$

Characterize the Ratio $k^{th}/ (k')^{th}$

Running Example:

2nd

k=2; D={1,2,3,4…79…98,10^5,10^6}

k'=1; S={1, 3,79}

- Imagine D is the query result set obtained by applying an arbitrary function *f()* to a database
- Impossible to predict the ratio of $10^5/79$
- Without any knowledge about *f()*, we can't bias 79 to larger values
- But with some domain (prior) knowledge and a sample of *f()*, we may reasonably guess the behaviors of *f()*

Characterize the Ratio $k^{th}/ (k')^{th}$ (cont.)

Key Question:

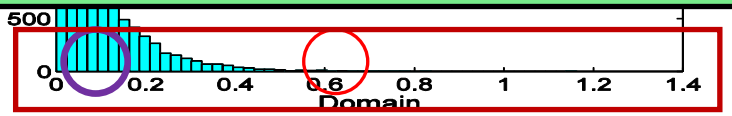
What aspect of the queries we want to model, in order to describe different distributions of $k^{th}/ (k')^{th}$?

tightly distributed around its mean; k^{th} and $(k')^{th}$ are very close

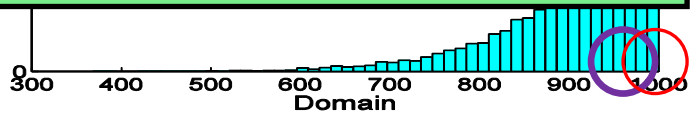
- When a new query is asked, we guess which type of "typical" queries by a few *f() samples*
- A model is needed to classify the queries

Conclusion: We need to model the histogram shape!

- Histogram shape of the query result set affects the ratio of $k^{th}/ (k')^{th}$
- Scale does not matter



(c)



(d)

First Step to Define the Model

- Now, we want a model to capture the histogram shape of a query result set
- Before deriving the math, it is beneficial to think how the model will work
  - Since if we know how the model works, we can find an appropriate probability density function to describe the model

Our Generative Model

- Assume there is a set of histogram shapes; each shape has a weight w_i , where $\sum w_i = 1$
- To generate a data set
  1. A biased die (w_is) is rolled to determine by which shape pattern the new data set will be generated
  2. An arbitrary scale is selected to define the magnitude of the items in the data set
  3. The shape and the scale are used to instantiate a $f(x \mid shape, scale)$ distribution and we repeatedly sample from this distribution to produce the data set

Our Generative Model (cont.)

- The initial set of weights (w_i s) are our ***prior distribution***
  - Indicating our belief that how likely a new query result set's histogram shape will match each shape pattern

Bayesian Approach

- Problem: prior weights cannot give enough info
  - One histogram shape may indicate that our estimator $(k')^{th}$ is close to the extreme value k^{th}
  - One histogram shape may indicate that our estimator $(k')^{th}$ is far from the extreme value k^{th}
  - Question: How do we determine which histogram shape we are experiencing once we have sampled S?

Bayesian Approach (cont.)

- We can rely on a principled Bayesian approach
  - Can combine the informative prior with the sample taken from the new data set
- After a sample of new data set is taken, we use it to update the prior weights, the updated weights are our ***posterior distribution***
  - Incorporate the new evidence to determine the current histogram shape
  - Place more weight on the most probably shape
  - The updated weights w_i's are used for bounding extreme value

Overview of Bayesian Framework

1. Build Prior Model: a set of weighted histogram shapes
2. Update Prior Model with Sample: adjusting prior weight of each shape to better describe the new data set's histogram shape
3. Use the Updated Model to Confidence Bound on the Ratio of k^{th}/ $(k')^{th}$

Bayesian Framework
——Step 1:Prior Shape Model

- Modified mixture of Gamma(x|α,β) distribution
- Treat scale parameter β as a random variable; integrate β, the result:
  - Each mixture component probability density function (pdf) p is indexed by a shape parameter α

$$p(<M,S,N>|\alpha)=\frac{M^{\alpha-1}}{\Gamma^{N}(\alpha)}S^{-N\alpha-1}\Gamma(N\alpha+1)$$

  - The above pdf's input only requires
    - M is the productivity
    - S is the sum
    - N is the cardinality

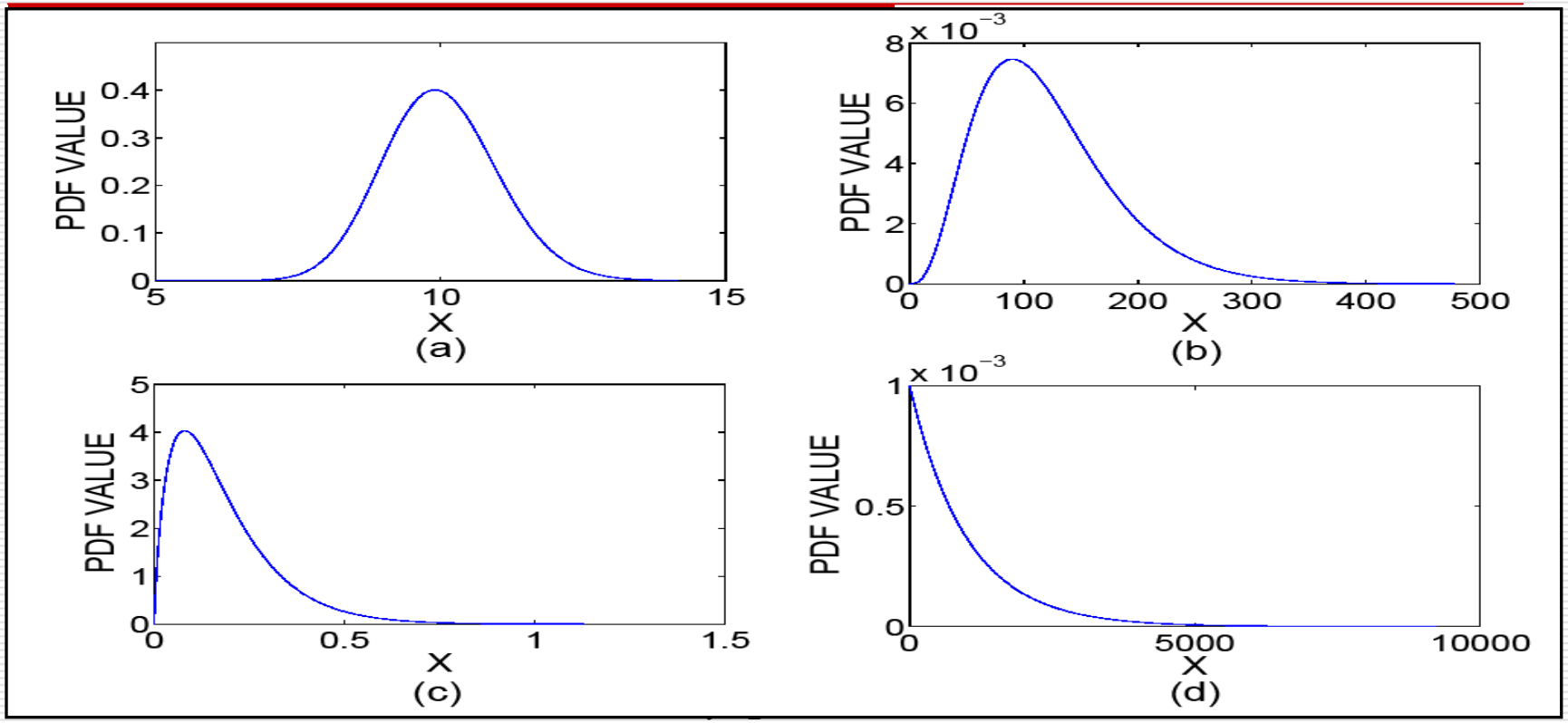
Bayesian Framework
——Step 1:Prior Shape Model

- Use $\vec{x} =< M, S, N >$
- The probability density function of our model becomes

$$f(\vec{x} \mid \Theta) = \sum_{i=1}^{c} w_i \, p(\vec{x} \mid \alpha_i)$$

- We use EM algorithm to learn this model from historical workload

Bayesian Framework Step 1
——Why Gamma distribution?



- Gamma(α,β) distribution can produce shape with arbitrary right leaning skew

Bayesian Framework
—— Step 2:Update Prior Weights

- Aggregate sample to the form $\vec{x} =< M, S, N >$
- Apply Bayes rule to update weights

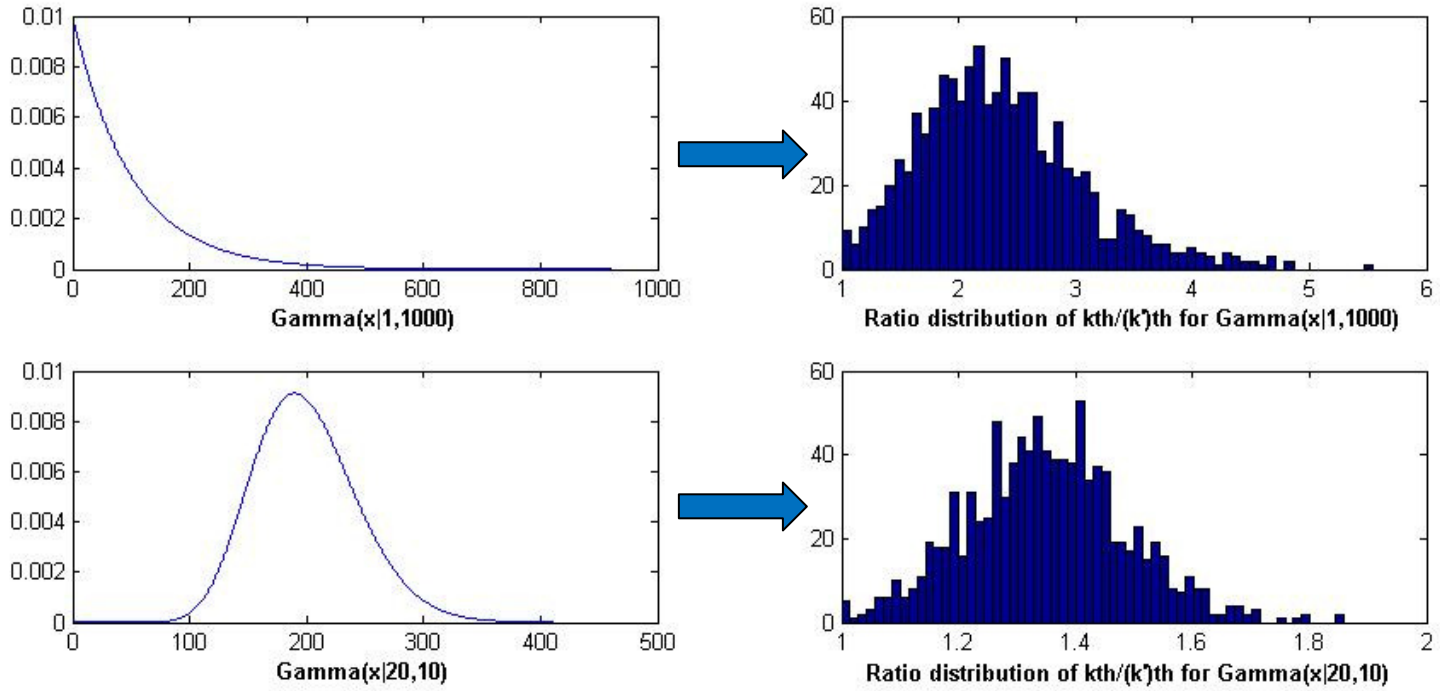
$$w_i' = \frac{w_i\, p(\vec{x} \mid \alpha_i)}{\sum_j w_j\, p(\vec{x} \mid \alpha_j)}$$

- The resulting pdf

$$f^*(\vec{x} \mid \Theta) = \sum_{i=1}^{c} w_i'\, p(\vec{x} \mid \alpha_i)$$

Bayesian Framework

—— Step 3:Confidence Bound Extreme Value



- Recall that each histogram shape characterizes a ratio distribution of $k^{th}/(k')^{th}$

Bayesian Framework

Step 3:Confidence Bound Extreme Value(cont.)

- Each shape has a corresponding ratio distribution
- Step 2 gives the posterior weight w_i' for each shape pattern
- Our model now can be perceived as someone choose a shape with a die roll biased by w_i'
- And we do not know which shape the die roll has chosen
- Then, the resulting ratio distribution is a mixture of each shape pattern's ratio distribution
- Probability of the i^{th} ratio distribution in the mixture is w_i'

Bayesian Framework

Step 3:Confidence Bound Extreme Value(cont.)

- Now we have
  1. The ratio distribution in a mixture form
  2. Our estimator, the $(k')^{th}$ laregest in the sample
- We can confidence bound the k^{th} largest value

Bayesian Framework

Step 3:Confidence Bound Extreme Value(cont.)

- Problem left is that for a given shape, how to ***efficiently*** get its ratio distribution?
- We devised a method called TKD sampling (details in the paper) accomplishes this in O(num*k') time

Outline

- ☑ Problem definition
- ☑ Example applications
- ☑ A Bayesian approach
- ☐ Experiment and conclusion

Experiment Setup

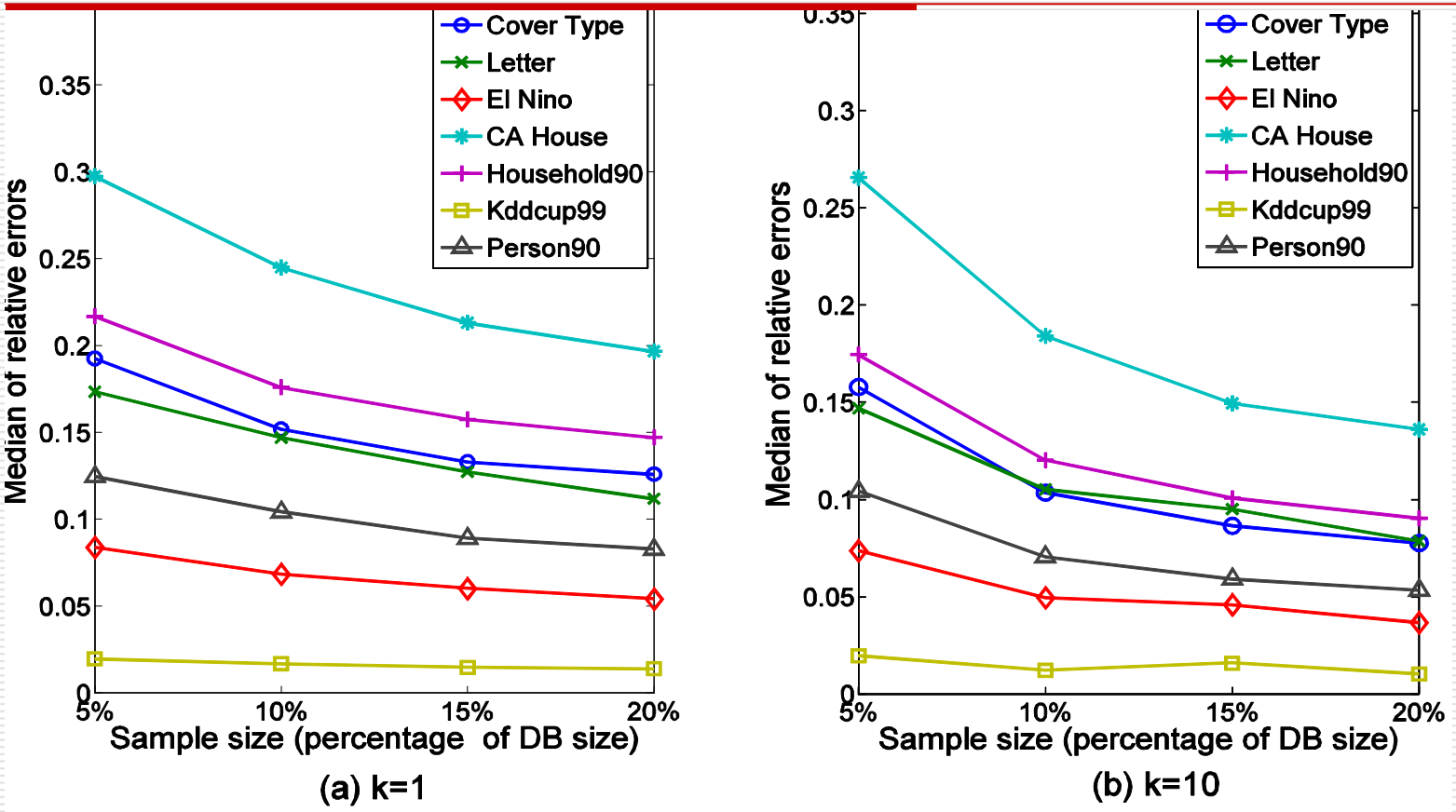
- 7 real multi-dimensional data sets
- A query is created by:
  1. A tuple *t* is randomly selected
  2. A selectivity *s* is randomly picked from 5% to 20%
  3. *s x (DB size)* nearest neighbors of *t* are chosen as the query result set
  4. The scoring function is the randomly-weighted sum of three arbitrary attributes.
- 500 training queries & 500 testing queries
- Confidence level is set to 95%

Experiment Results

Data sets	k=1	k=5	k=10	k=20
Letter	1.00	0.98	0.97	0.94
CAHouse	0.97	0.99	0.97	0.96
El Nino	1.00	1.00	0.99	0.99
Cover Type	1.00	1.00	0.99	0.99
KDDCup 99	0.92	0.91	0.92	0.93
Person 90	0.92	0.94	0.90	0.91
Household 90	0.98	0.97	0.97	0.97

- The Coverage rates for 95% confidence bound, with 10% sample for various k.

Experiment Results



- Increasing sample size for k=1 and k=10

Conclusion

- Defined the problem of estimating the K^{th} extreme value in a data set
- A Bayesian approach is proposed
- Experimental verification on real data

Questions