Internet Measurement and Data Analysis (6)

Kenjiro Cho

2012-11-07

review of previous class

Class 5 Diversity and complexity (10/31)

- Long tail
- Web access and content distribution
- Power-law and complex systems
- exercise: power-law analysis

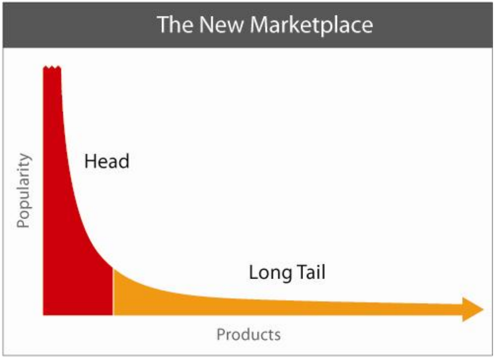
today's topics

Class 6 Correlation

- Online recommendation systems
- Distance
- Correlation coefficient
- exercise: correlation analysis

online recommender systems

- finding potential needs for long-tail users at EC sites
  - by recommending products which fit each user's taste
- widely used as the cost goes down by recomender package software



source: `http://longtail.com/`

recommender systems

- from user online behavior, infer useful information for users automatically
- EC sites: recommend products automatically from purchase or view records
- other applications: music, movies, search engine, etc

different approaches for database structure

- item based: compile data for each item
- user based: compile data for each user
- most systems combine both

prediction methods of recommender systems

- content based:
  - recommend items similar to the items the user used in the past
    - (manual) classifications of items
    - clustering items by machine learning methods
    - building rules from know-how
  - tend to recommend items in the same group, less surprising
- collaborative filtering: employed by amazon and others
  - e.g., "users who bought X also bought Y"
  - compute similarities among users from their online activities
  - recommend items bought by similar users
  - main feature: it does not use the information about items
  - could lead to surprising findings for user (serendipity)
- naive bayesian filter: often used for spam filtering
  - machine-learning technique to compute probabilities from a large number of item and user attributes

collaborative filtering

- several well-known algorithms
- example: simple correlion analysis between users
  - compute correlation between users to find similar users
  - rate item as a sum of others' scores weighted by the similarity

example: purchase history

user	item a	b	c	d	e	f	...
A	1		1		1		...
B			1	1			...
C	1	1					...
D	1		1		1		...
...							...

compute the scores of items that A does not have but A's similar users have

user	similarity σ	item a	b	c	d	e	f	...
A	1	1		1		1		...
S	0.88		0.88		-		0.88	...
C	0.81		0.81		-		-	...
K	0.75		-		-		-	...
F	0.73		0.73		0.73		0.73	...
score			2.50		0.73		1.61	...

Example: Netflix Prize

- an open annual competition for collaborative filtering algorithms to predict user ratings for movies
- sponsored by Netflix, an online DVD-rental/download service company
- competition: data set $< user\_id, movie\_id, date\_of\_grade, grade >$
  - training data set (100 million ratings)
  - qualifying data set (2.8 million ratings)
    - quiz data set (1.4 million)
    - test data set (1.4 million)
  - results are scored by root mean squared error
- competition started in 2006 and ended in 2009
  - criticized by privacy advocates

distances

various distances

- Euclidean distance
- standardized Euclidean distance
- Minkowski distance
- Mahalanobis distance

similarities

- binary vector similarities
- n-dimensional vector similarities

properties of distance

a metric of distance $d(x, y)$ between 2 points (x, y) in space
positivity

$$d(x, y) \geq 0$$

$$d(x, y) = 0 \Leftrightarrow x = y$$

symmetry

$$d(x, y) = d(y, x)$$

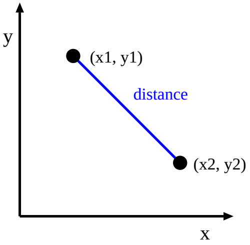
triangle inequality

$$d(x, z) \leq d(x, y) + d(y, z)$$

Euclidean distance

word "distance" usually means "Euclidean distance"
a distance of 2 points (x, y) in a n-dimensional space

$$d(x, y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

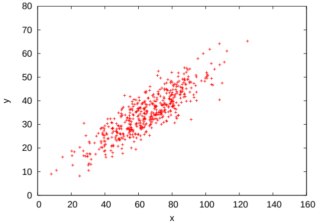


euclidean distance in 2-dimensional space

standardized Euclidean distance

- when variances are different among variables, distances are affected.
- standard Euclidean distance: normalized by dividing the Euclidean distance by the variance of each variable

$$d(x, y) = \sqrt{\sum_{k=1}^{n} (\frac{x_k}{s_k} - \frac{y_k}{s_k})^2} = \sqrt{\sum_{k=1}^{n} \frac{(x_k - y_k)^2}{s_k^2}}$$

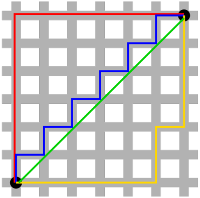


Minkowski distance

generalization of Euclidean distance: as parameter r grows, a short cut crossing different axes is preferred more

$$d(x, y) = (\sum_{k=1}^{n} |x_k - y_k|^r)^{\frac{1}{r}}$$

- $r = 1$: Manhattan distance
  - Hamming distance: for 2 strings of equal length, the number of positions at which the corresponding symbols are different.
  - example: the hamming distance of 111111 and 101010 is 3
- $r = 2$: Euclidean distance



Manhattan distance vs. Euclidean distance

Mahalanobis distance

a distance that takes correlations into account, when correlation exists between variables

$$mahalanobis(x, y) = (x - y)\Sigma^{-1}(x - y)^T$$

here, Σ^{-1} is the inverse matrix of its covariance matrix

similarities

similarity

- numerical measure of how alike 2 data objects are

properties of similarity
positivity

$$0 \leq s(x, y) \leq 1$$

$$s(x, y) = 1 \Leftrightarrow x = y$$

symmetry

$$s(x, y) = s(y, x)$$

in general, triangle inequality does not apply to similarities

similarity between binary vectors

Jaccard coefficient

- used for similarity between binary vectors in which the occurrences of 1 is much smaller than the occurrences of 0
- example: as a metric of similarity by occurrences of words in documents
- many words do not appear in both documents $\Rightarrow$ not considered
- the following table shows the relationship of each item

		vector y	
		1	0
vector x	1	n_{11}	n_{10}
	0	n_{01}	n_{00}

Jaccard coefficient:

$$J = \frac{n_{11}}{n_{11} + n_{10} + n_{01}}$$

similarity between vectors

similarity between (non-binary) vectors

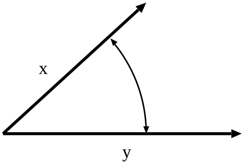
- example: similarity of documents where frequencies of words are also taken into consideration

cosine similarity

- take the angle (cosine) of (x, y) of vectors
- normalized by the length of the vector $\Rightarrow$ length is not considered

$$cos(x, y) = \frac{x \cdot y}{\|x\| \|y\|}$$

$x \cdot y = \sum_{k=1}^{n} x_k y_k$: product of vectors
$\|x\| = \sqrt{\sum_{k=1}^{n} x_k^2} = \sqrt{x \cdot x}$: length of the vector



example: cosine similarity

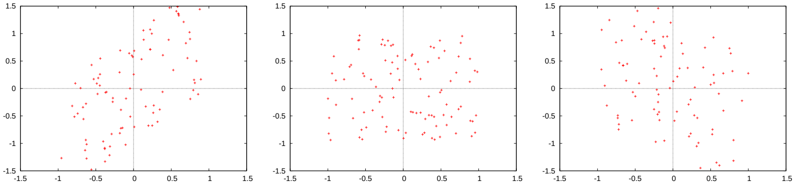
$$x = 3 \quad 2 \quad 0 \quad 5 \quad 0 \quad 0 \quad 0 \quad 2 \quad 0 \quad 0$$
$$y = 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 2$$

$$x \cdot y = 3 * 1 + 2 * 1 = 5$$
$$\|x\| = \sqrt{3 * 3 + 2 * 2 + 5 * 5 + 2 * 2} = \sqrt{42} = 6.481$$
$$\|y\| = \sqrt{1 * 1 + 1 * 1 + 2 * 2} = \sqrt{6} = 2.449$$

$$cos(x, y) = \frac{5}{6.481 * 2.449} = 0.315$$

scatter plots and correlation

- explores relationships between 2 variables
  - X-axis: variable X
  - Y-axis: corresponding value of variable Y
- you can identify
  - whether variables X and Y related
    - no relation, positive correlation, negative correlation
- correlation coefficient: a measure of the strength and direction of correlation



examples: positive correlation 0.7 (left), no correlation 0.0 (middle), negative correlation -0.5 (right)

correlation

- covariance:
$$\sigma_{xy}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$
- correlation coefficient:
$$\rho_{xy} = \frac{\sigma_{xy}^2}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$
- correlation coefficient: the covariance of 2 variables normalized by their product of their standard deviations, a value between -1 and $+1$ inclusive.
- sensitive to outliers. so, you should use a scatter plot to observe outliers.
- correlation and causality
  - correlation does not imply causal relationship
    - third factor C causes both A and B
    - coincidence

computing correlation coefficient (1)

sum of squares

$$\begin{aligned}
\sum_{i=1}^{n}(x_i - \bar{x})^2 &= \sum_{i=1}^{n}(x_i^2 - 2x_i\bar{x} + \bar{x}^2) \\
&= \sum_{i=1}^{n} x_i^2 - 2\bar{x}\sum_{i=1}^{n} x_i + n\,\bar{x}^2 \\
&= \sum_{i=1}^{n} x_i^2 - 2\bar{x}\cdot n\,\bar{x} + n\,\bar{x}^2 \\
&= \sum_{i=1}^{n} x_i^2 - n\,\bar{x}^2 = \sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n}
\end{aligned}$$

sum of products

$$\begin{aligned}
\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) &= \sum_{i=1}^{n}(x_i y_i - x_i\bar{y} - \bar{x}y_i + \bar{x}\bar{y}) \\
&= \sum_{i=1}^{n} x_i y_i - \bar{x}\sum_{i=1}^{n} y_i - \bar{y}\sum_{i=1}^{n} x_i + n\,\bar{x}\bar{y} \\
&= \sum_{i=1}^{n} x_i y_i - \bar{x}\cdot n\,\bar{y} - \bar{y}\cdot n\,\bar{x} + n\,\bar{x}\bar{y} \\
&= \sum_{i=1}^{n} x_i y_i - n\,\bar{x}\bar{y} = \sum_{i=1}^{n} x_i y_i - \frac{(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n}
\end{aligned}$$

computing correlation coefficient (2)

correlation coefficient

$$
\begin{aligned}
\rho_{xy} &= \frac{\sigma_{xy}^2}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}} \\
&= \frac{\sum_{i=1}^{n} x_i y_i - n\,\bar{x}\bar{y}}{\sqrt{(\sum_{i=1}^{n} x_i^2 - n\,\bar{x}^2)(\sum_{i=1}^{n} y_i^2 - n\,\bar{y}^2)}} \\
&= \frac{\sum_{i=1}^{n} x_i y_i - \frac{(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n}}{\sqrt{(\sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n})(\sum_{i=1}^{n} y_i^2 - \frac{(\sum_{i=1}^{n} y_i)^2}{n})}}
\end{aligned}
$$

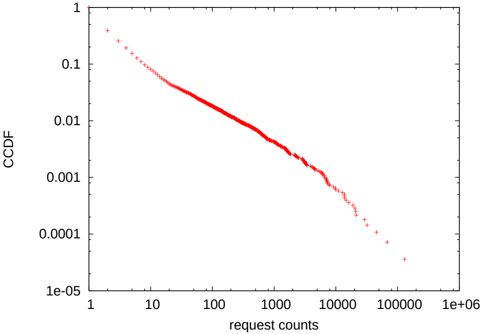
other correlation coefficients

- Pearson's product-moment correlation coefficient
  - or simply "correlation coefficient" (what we have learned)
- rank correlation coefficient: relationships between different rankings on the same set of items
  - Spearman's rank correlation coefficient
  - Kendall's rank correlation coefficient
- others

previous exercise: CCDF plots

extract the access count of each unique content from the JAIST server access log, plot the access count distribution in CCDF

```
% ./count_contents.rb sample_access_log > contents.txt
% ./make_ccdf.rb contents.txt > ccdf.txt
```



extracting the access count of each unique content

```
# output: URL req_count byte_count
# regular expression for apache combined log format
#   host ident user time request status bytes referer agent
re = /^(\S+) (\S+) (\S+) \[(.*?)\] "(.*?)" (\d+) (\d+|-) "(.*?)" "(.*?)"/
# regular expression for request: method url proto
req_re = /(\w+) (\S+) (\S+)/
contents = Hash.new([0, 0])
count = parsed = 0
ARGF.each_line do |line|
  count += 1
  if re.match(line)
    host, ident, user, time, request, status, bytes, referer, agent = $~.captures
    # ignore if the status is not success (2xx)
    next unless /2\d{2}/.match(status)
    if req_re.match(request)
      method, url, proto = $~.captures
      # ignore if the method is not GET
      next unless /GET/.match(method)
      parsed += 1
      # count contents by request and bytes
      contents[url] = [contents[url][0] + 1, contents[url][1] + bytes.to_i]
    else
      # match failed.  print a warning msg
      $stderr.puts("request match failed at line #{count}: #{line.dump}")
    end
  else
    $stderr.puts("match failed at line #{count}: #{line.dump}") # match failed.
  end
end
contents.sort_by{|key, value| -value[0]}.each do |key, value|
  puts "#{key} #{value[0]} #{value[1]}"
end
$stderr.puts "# #{contents.size} unique contents in #{parsed} successful GET requests"
$stderr.puts "# parsed:#{parsed} ignored:#{count - parsed}"
```

script to convert the access count to CCDF

```
#!/usr/bin/env ruby

re = /^\S+\s+(\d+)\s+\d+/

n = 0
counts = Hash.new(0)
ARGF.each_line do |line|
  if re.match(line)
    counts[$1.to_i] += 1
    n += 1
  end
end

cum = 0
counts.sort.each do |key, value|
  comp = 1.0 - Float(cum) / n
  puts "#{key} #{value} #{comp}"
  cum += value
end
```

gnuplot script for plotting the content access count in CCDF

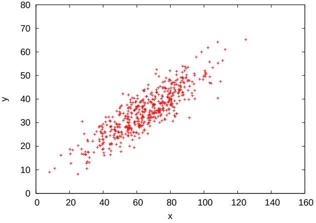
```
set logscale
set xlabel "request counts"
set ylabel "CCDF"

plot "ccdf.txt" using 1:3 notitle with points
```

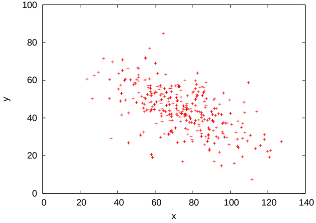
today's exercise: computing correlation coefficient

- compute correlation coefficient using the sample data sets
  - correlation-data-1.txt, correlation-data-2.txt

correlation coefficient

$$\rho_{xy} = \frac{\sigma_{xy}^2}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{n} x_i y_i - \frac{(\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n}}{\sqrt{(\sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n})(\sum_{i=1}^{n} y_i^2 - \frac{(\sum_{i=1}^{n} y_i)^2}{n})}}$$





data-1:r=0.87 (left), data-2:r=-0.60 (right)

script to compute correlation coefficient

```
#!/usr/bin/env ruby

# regular expression for matching 2 floating numbers
re = /([-+]?\d+(?:\.\d+)?)\s+([-+]?\d+(?:\.\d+)?)/

sum_x = 0.0 # sum of x
sum_y = 0.0 # sum of y
sum_xx = 0.0 # sum of x^2
sum_yy = 0.0 # sum of y^2
sum_xy = 0.0 # sum of xy
n = 0 # the number of data

ARGF.each_line do |line|
    if re.match(line)
      x = $1.to_f
      y = $2.to_f
      sum_x += x
      sum_y += y
      sum_xx += x**2
      sum_yy += y**2
      sum_xy += x * y
      n += 1
    end
end

r = (sum_xy - sum_x * sum_y / n) /
  Math.sqrt((sum_xx - sum_x**2 / n) * (sum_yy - sum_y**2 / n))

printf "n:%d r:%.3f\n", n, r
```

summary

Class 6 Correlation

- Online recommendation systems
- Distance
- Correlation coefficient
- exercise: correlation analysis

next class

Class 7 Multivariate analysis (11/14)

- Data sensing
- Linear regression
- Principal Component Analysis
- exercise: linear regression

Class 8 Time-series analysis (11/20) ***makeup class

- Nov 20 (Tue) 11:10-12:40 ϵ11