Internet Measurement and Data Analysis (11)

Kenjiro Cho

2015-01-14

review of previous class

Class 10 Anomaly detection and machine learning (12/22)

- Anomaly detection
- Machine Learning
- SPAM filtering and Bayes theorem
- exercise: naive Bayesian filter
- **the final report**

today's topics

Class 11 Data Mining

- Pattern extraction
- Classification
- Clustering
- privacy issues
- exercise: clustering

data mining

- huge volume of data
  - difficult to handle with traditional methods
  - need to extract information hidden in data that is not readily evident
- Data Mining
  - huge volume, multi-dimensional diverse data, non-trivial distributions
  - methods often derived from ideas in machine learning, AI, pattern recognition, statistics, database, signal processing
- data processing becomes practical by growing computing power (e.g., cloud computing)

Data Mining methods

definition: non-trivial extraction of implicit, previously unknown and potentially useful information from data

- pattern extraction: find existing models and patterns in data
  - correlation
  - time-series
- classification: automatically create new classes that do not exist in the original data
  - rule-based methods
  - naive Bayesian filter
  - neural networks
  - support vector machine (SVM)
  - dimensionality reduction (e.g., PCA)
- clustering: compute the distance (or similarity) between data points and group them
  - distance based, density based, graph based
  - k-means, DBSCAN
- anomaly detection: find deviation from normal state using statistical methods
  - univariate, multivariate
  - outlier detection

clustering

important technique for classifying data with complex relationship

compute the distance (or similarity) of variables to make them into groups

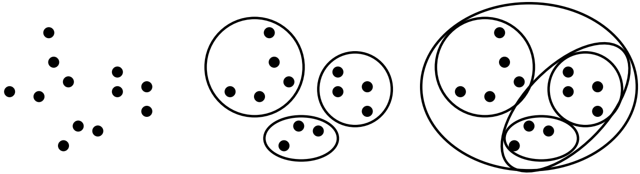
- to classify and understand data
- to summarize data

various applications

- business: grouping customers for marketing purposes
- meteorology: finding patterns in complex weather data
- biology: classifying genes and proteins
- medical science and pharmacy: complex relationship of symptoms and effects

clustering methods

- partitional clustering
  - k-means method
- hierarchical clustering
  - MST method
  - DBSCAN method



original points partitional clustering hierarchical clustering

k-means method

- partitional clustering
- specify the number of cluster, k
- basic algorithm is simple
  - each cluster has centroid (usually mean)
  - assign each object to the closest cluster
  - repeat re-computation of centroids and cluster assignments
- limitations
  - need to specify the number of clusters, k, beforehand
  - sensitive to the selection of initial points
  - clusters are supposed to have similar sizes and densities, and a round shape
  - sensitive to outliers

basic k-means algorithm:

1: select k points randomly as the initial centroids
2: **repeat**
3: form k clusters by assigning all points to the closest centroid
4: recompute the centroid of each cluster
5: **until** the centroids don't change

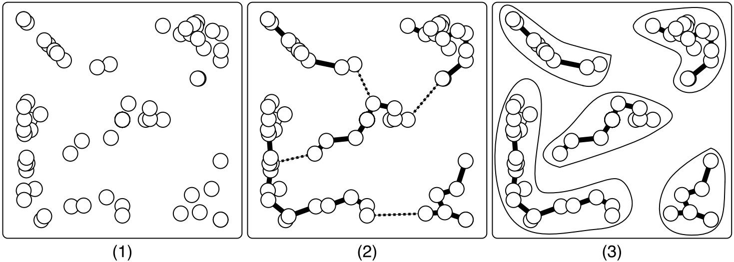
hierarchical clustering

- generate clusters using a tree structure
  - the cluster structure can be explained by the tree
- no need to specify the number of clusters beforehand
- 2 approaches
  - agglomerative: start with data points as individual clusters, and repeat merging the closest clusters
  - divisive: start with one all-inclusive cluster, and repeat splitting clusters

MST clustering

Minimum Spanning Tree clustering

- divisive hierarchical clustering
- start with an arbitrary point, and create MST
- repeat dividing clusters by removing the longest edge



DBSCAN

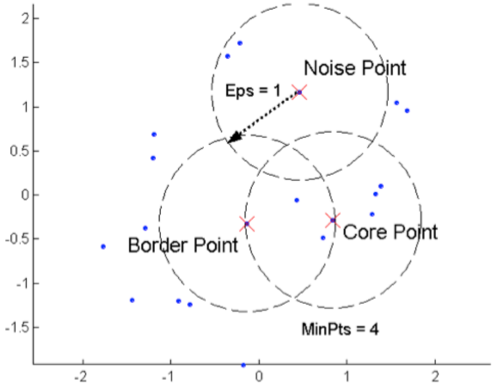
Density-Based Spatial Clustering

- density-based: combine data points within the specified distance
- can extract arbitrary (non-round) shapes of clusters
- robust against noise and outliers
- distance threshold Eps and point threshold $MinPts$
  - Core points: within the distance Eps, more than $MinPts$ neighbors exist
  - Border points: not Core, but have a core within the distance Eps
  - Noise points: have no core within the distance Eps
- limitations: clusters with different densities, or with large number of parameters

DBSCAN algorithm:

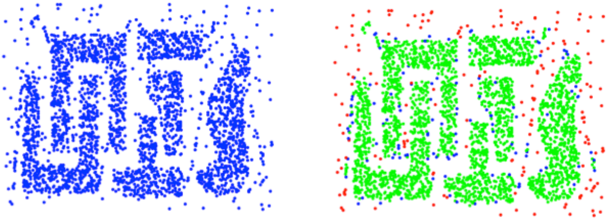
1: label all points as core, border, or noise points
2: eliminate noise points
3: put an edge between all core points that are within Eps of each other
4: make each group of connected core points into a separate cluster
5: assign each border point to one of the clusters of its associated core points

DBSCAN: Core, Border, and Noise Points



source: Tan, Steinbach, Kumer. Introduction to Data Mining

DBSCAN: example of Core, Border, and Noise Points



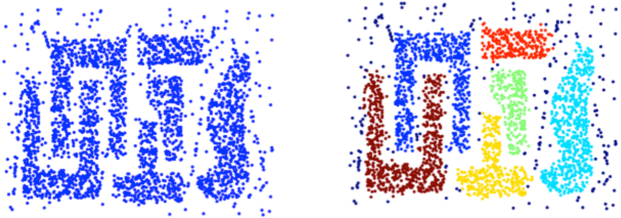
Original Points

Point types: core, border
and noise

Eps = 10, MinPts = 4

source: Tan, Steinbach, Kumer. Introduction to Data Mining

DBSCAN: example clusters

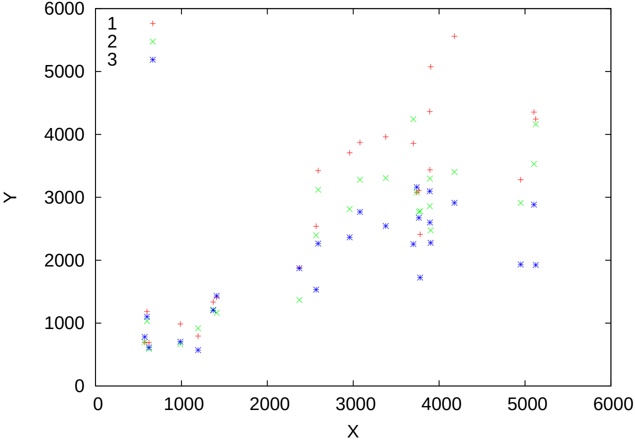


Clusters

source: Tan, Steinbach, Kumer. Introduction to Data Mining

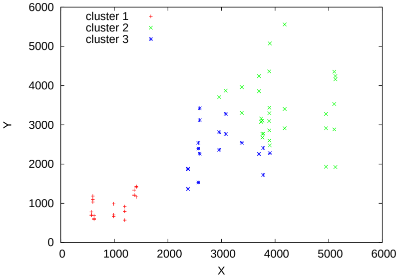
today's exercise: k-means clustering

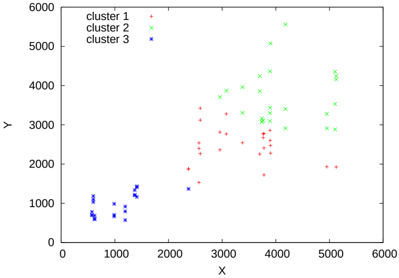
```
% ruby k-means.rb km-data.txt > km-results.txt
```



k-means clustering results

- different results by different initial values





k-means code (1/2)

```
k = 3 # k clusters
re = /^(\d+)\s+(\d+)/
INFINITY = 0x7fffffff

# read data
nodes = Array.new # array of array for data points: [x, y, cluster_index]
centroids = Array.new # array of array for centroids: [x, y]
ARGF.each_line do |line|
  if re.match(line)
    c = rand(k) # randomly assign initial cluster
    nodes.push [$1.to_i, $2.to_i, c]
  end
end

round = 0
begin
  updated = false

  # assignment step: assign each node to the closest centroid
  if round != 0  # skip assignment for the 1st round
    nodes.each do |node|
      dist2 = INFINITY # square of dsistance to the closest centroid
      cluster = 0 # closest cluster index
      for i in (0 .. k - 1)
        d2 = (node[0] - centroids[i][0])**2 + (node[1] - centroids[i][1])**2
        if d2 < dist2
          dist2 = d2
          cluster = i
        end
      end
      node[2] = cluster
    end
  end
```

k-means code (2/2)

```
  # update step: compute new centroids
  sums = Array.new(k)
  clsize = Array.new(k)
  for i in (0 .. k - 1)
    sums[i] = [0, 0]
    clsize[i] = 0
  end
  nodes.each do |node|
    i = node[2]
    sums[i][0] += node[0]
    sums[i][1] += node[1]
    clsize[i] += 1
  end

  for i in (0 .. k - 1)
    newcenter = [Float(sums[i][0]) / clsize[i], Float(sums[i][1]) / clsize[i]]
    if round == 0 || newcenter[0] != centroids[i][0] || newcenter[1] != centroids[i][1]
      centroids[i] = newcenter
      updated = true
    end
  end

  round += 1

end while updated == true

# print the results
nodes.each do |node|
  puts "#{node[0]}\t#{node[1]}\t#{node[2]}"
end
```

gnuplot script

```
set key left
set xrange [0:6000]
set yrange [0:6000]
set xlabel "X"
set ylabel "Y"

plot "km-results.txt" using 1:($3==0?$2:1/0) title "cluster 1" with points, \
"km-results.txt" using 1:($3==1?$2:1/0) title "cluster 2" with points, \
"km-results.txt" using 1:($3==2?$2:1/0) title "cluster 3" with points
```

previous exercise: SPAM filtering

- SPAM filtering using naive bayesian classifier
  - based on the code from "Programming Collective Intelligence" Chapter 6

```
% ruby naivebayes.rb
classifying "quick rabbit" => good
classifying "quick money" => bad
```

naive bayesian classifier for the exercise

compute the propbability of a document to be classified into a specific category by words appearing in the dicument

$$P(C)\prod_{i=1}^{n} P(x_i|C)$$

- $P(C)$: the probability of the category
- $\prod_{i=1}^{n} P(x_i|C)$: product of the conditional probability of each word in the category

select the category with the highest probability

- threshold： the probability of the best category should be $thresh$ times higher than that of the second best category

SPAM classifier script

- training and classifier

```
# create a classifier instance
cl = NaiveBayes.new

# training
cl.train('Nobody owns the water.','good')
cl.train('the quick rabbit jumps fences','good')
cl.train('buy pharmaceuticals now','bad')
cl.train('make quick money at the online casino','bad')
cl.train('the quick brown fox jumps','good')

# classify
sample_data = [ "quick rabbit", "quick money" ]

sample_data.each do |s|
  print "classifying \"#{s}\" => "
  puts cl.classify(s, default="unknown")
end
```

script: Classifier Class (1/2)

```
# feature extraction
def getwords(doc)
  words = doc.split(/\W+/)
  words.map!{|w| w.downcase}
  words.select{|w| w.length < 20 && w.length > 2 }.uniq
end

# base class for classifier
class Classifier
  def initialize
    # initialize arrays for feature counts, category counts
    @fc, @cc = {}, {}
  end

  def getfeatures(doc)
    getwords(doc)
  end

  # increment feature/category count
  def incf(f, cat)
    @fc[f] ||= {}
    @fc[f][cat] ||= 0
    @fc[f][cat] += 1
  end

  # increment category count
  def incc(cat)
    @cc[cat] ||= 0
    @cc[cat] += 1
  end

  ...
```

script: Classifier Class (2/2)

```
  def fprob(f,cat)
    if catcount(cat) == 0
      return 0.0
    end

    # the total number of times this feature appeared in this
    # category divided by the total number of items in this category
    Float(fcount(f, cat)) / catcount(cat)
  end

  def weightedprob(f, cat, weight=1.0, ap=0.5)
    # calculate current probability
    basicprob = fprob(f, cat)
    # count the number of times this feature has appeared in all categories
    totals = 0
    categories.each do |c|
      totals += fcount(f,c)
    end
    # calculate the weighted average
    ((weight * ap) + (totals * basicprob)) / (weight + totals)
  end

  def train(item, cat)
    features = getfeatures(item)
    features.each do |f|
      incf(f, cat)
    end
    incc(cat)
  end
end
```

script: NaiveBayes Class

```
# naive baysian classifier
class NaiveBayes < Classifier
  def initialize
    super
    @thresholds = {}
  end

  def docprob(item, cat)
    features = getfeatures(item)
    # multiply the probabilities of all the features together
    p = 1.0
    features.each do |f|
      p *= weightedprob(f, cat)
    end
    return p
  end

  def prob(item, cat)
    catprob = Float(catcount(cat)) / totalcount
    docprob = docprob(item, cat)
    return docprob * catprob
  end

  def classify(item, default=nil)
    # find the category with the highest probability
    probs, max, best = {}, 0.0, nil
    categories.each do |cat|
      probs[cat] = prob(item, cat)
      if probs[cat] > max
        max = probs[cat]
        best = cat
      end
    end
    # make sure the probability exceeds threshold*next best
```

debug: dumping the feature probabilities

internal states after the training:

```
fprob for "nobody":    good:0.333 bad:0.000
fprob for "owns":      good:0.333 bad:0.000
fprob for "the":       good:1.000 bad:0.500
fprob for "water":     good:0.333 bad:0.000
fprob for "quick":     good:0.667 bad:0.500
fprob for "rabbit":    good:0.333 bad:0.000
fprob for "jumps":     good:0.667 bad:0.000
fprob for "fences":    good:0.333 bad:0.000
fprob for "buy":       good:0.000 bad:0.500
fprob for "pharmaceuticals":   good:0.000 bad:0.500
fprob for "now":       good:0.000 bad:0.500
fprob for "make":      good:0.000 bad:0.500
fprob for "money":     good:0.000 bad:0.500
fprob for "online":    good:0.000 bad:0.500
fprob for "casino":    good:0.000 bad:0.500
fprob for "brown":     good:0.333 bad:0.000
fprob for "fox":       good:0.333 bad:0.000
```

on the final report

- select A or B
  - A. Wikipedia pageview ranking
  - B. free topic
- up to 8 pages in the PDF format
- submission via SFC-SFS by 2015-01-29 (Thu) 23:59

final report topics

A. Wikipedia pageview ranking

- purpose: extracting popular keywords from real datasets and observing temporal changes
- data: pagecount datasets from Wikipedia English version
- items to submit
  - A-1 CCDF plot of the pagecount distribution
  - A-2 list of top 10 titles for each day and for the week
  - A-3 plot the changes of the daily ranking of the top 10 titles
  - A-4 other analysis (optional)
    - optional analysis of your choice
  - A-5 discussion on the results
    - describe what you observe from the data

B. free topic

- select a topic by yourself
- the topic is not necessarily on networking
- but the report should include some form of data analysis and discussion about data and results

more weight on the discussion for the final report

A. Wikipedia pageview ranking

data: pagecount datasets from Wikipedia English version

- original datasets provide by wikimedia
  - http://dumps.wikimedia.org/other/pagecounts-raw/
- pagecount dataset for the report: en-201412.zip (790MB, 2.4GB uncompressed)
  - hourly pagecounts of the week, Dec 1-7, 2014
  - only for English Wikipedia, only 4 hours (00-04 UTC) for each day (to reduce the data size)

data format

- project encoded_pagetitle requests size
  - project: wikimedia project name (all "en" in this dataset)
  - encoded_pagetitle: URI encoded page title
  - requests: the number of requests
  - size: the size of the content

```
$ head -n 10 pagecounts-20141203-030000
en !! 1 9295
en !!! 6 103994
en !!!_(album) 2 23644
en !%20(disambiguation) 1 10393
en !%EF%BF%BD%02 1 6645
en !Adios_Amigos! 1 15951
en !Alabadle! 1 10736
en !Bang! 1 15328
en !Ciauetistico! 2 21038
en !Hero 1 10938
```

a script to decode titles

- titles are percent-encoded
    - can be converted to UTF-8 by ruby's CGI.unescape()

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

require 'cgi'

re = /^([\w\.]+)\s+(\S+)\s+(\d+)\s+(\d+)/

ARGF.each_line do |line|
  if re.match(line)
    project, title, requests, bytes = $~.captures
    decoded_title = CGI.unescape(title)
    print "#{project} \"#{decoded_title}\" #{requests} #{bytes}\n"
  end
end
```

A. more on pagecount ranking

- A-1 CCDF plot of the pagecount distribution
  - aggregate all the datasets, sum up all requests for each title, and plot CCDF of the pagecount distribution
  - a log-log plot with request count on the X-axis, CCDF on Y-axis
- A-2 list of top 10 titles for each day and for the week total
  - create a table similar to the following

```
rank 12/1            12/2        12/3                 ...  12/7                 total
1    "Main_Page"      "Main_Page" "Main_Page"          ... "Main_Page"          "Main_Page"
2    "Ethernet_frame" "Cofferdam" "Special:HideBanners" ... "Special:HideBanners" "Special:HideBanne
         ...
```

- A-3 plot the changes of the daily ranking of the top 10 titles
  - time on X-axis, ranking on Y-axis
  - come up with a good way by yourself to show the changes of ranking over the week

privacy

from webster,

- privacy: ”the quality or state of being apart from company or observation”
- right to privacy: ”freedom from intrusion”

- views on privacy heavily depend on context and culture
  - basic human right
  - a commodity (asset); if infringed, it should be compensated via tort laws (for negligence, liability, etc)

informational privacy

- on collecting, storing and sharing one's personal information

- secrecy of correspondence
  - a fundamental legal principle in the constitutions of many countries
    - against censorship by govenment or any other third party
- naturally extended to communication
  - e.g., telephony, the Internet
- exceptions
  - informed consent (e.g., virus checker/spam filtering services)
  - emergency situations (to protect other services)
  - lawful business practices (e.g., looking at destination in a packet header)
- lawful interception (wire-tapping)
  - suspicion of crime
- communication data retention
  - many countries have laws to require service providers to store communication records for certain periods

traceable records everywhere

- cash cards, credit cards, transportation cards, members cards
- medical records
- device IDs: phone sim, MAC addresses, IP addresses, RFID
- web cookies, geo-location info
- surveillance cameras, fingerprints and machine recognition

your privacy data

- name, date-of-birth, sex, marital status
- address, phone number
- names of family members and pets
- financial data: income, savings, stocks
- educational records, medical records, religion
- purchase data, trip data
- photographs
- online behaviors
- personal preferences
- communication records (when, where, who, how)
- friends

who has your privacy data

authorities

- government bodies
- hospitals
- banks
- universities

commercial services

- stores and other services
- social network services

marketing values

- demographic data, geographic data, other statistics
- condition: individials are anonymous
- existence of black markets (for data thefts)

cybercrime market

- demands
  - permaceuticals: there exit people who need cheap permaceuticals, without prescriptions. most cheap drugs are ok, but some are fatal
  - adult content
- black market commodities
  - credit card numbers
  - email address list, or other personal info
  - online banking accounts or other online payment accounts
  - access to servers, hijacked PCs
- players
  - exploiters (attacking vulnerable hosts, or phishing)
  - spammers (sending spam emails)
  - web designers (for creating malicious web sites)
  - cashiers (withdrawing money from compromised credit cards or service accounts)
  - droppers (receiving marchandise at untraceable drop points)
  - credit card processors
  - hosting providers
  - various types of brokers

technological evolutions

technologies increase risks of privacy breaches

- computing power
- database
- consolidation (smartphones, RFIDs)

anonymized datasets and privacy risks

online behavior records

- location information
- shopping/browsing history
- values for marketing
- in old days, it was ok to provide anonymized data to third parties
- increasing privacy risks with technical advance

personally identifiable information

- directly related to a specific individual: name, member id, phone number, etc

anonymized personal information

- an individual can be distinguished from others
- possibility to identify the person with additional information
    - by time, location, or other non-frequent records

- example: anomymous survey at a university campus
    - Keio University students 33,681 (male 22,499 female 11,182)
    - SFC students 4,851 (male 2,871 female 1,980)

security and safety vs. privacy

- digital copyright (copyright holders/licensing agencies/users)
- war against terrorism, criminals

- convenience vs. privacy
    - consolidation
    - social networks (your privacy relies on your friends)

privacy in the future

- the post privacy era?
  - we may not have privacy as in the current form in the future
  - the concept of privacy is fairly new
    - since around 1890 after the advent of mass media
- complex issues (cultural, legal, economical aspects)
- as a user, you need to protect your privacy by yourself
  - don't need to be too pessimistic
  - awareness and understanding

summary

Class 11 Data Mining

- Pattern extraction
- Classification
- Clustering
- privacy issues
- exercise: clustering

next class

Class 12 Search and Ranking (1/19)

- Search systems
- PageRank
- exercise: PageRank algorithm