Nei miei dati c'è conoscenza o solo spazzatura?
di Roberto Bello

I DATA quando sono BIG sono di difficile utilizzo.

I BIG DATA dovrebbero servire a scoprire nuove opportunità di business.

PERO'

Ci sono troppe informazioni che già si conoscono a menadito: nel cestino.
Altre informazioni sono incomplete: nel cestino.
Altre informazioni sono assurde: nel cestino.
Altre informazioni sono palesemente false: nel cestino.
Altre informazioni sono fuori tema: nel cestino.
Altre informazioni sono ….: nel cestino.

I DATA rimasti sono ancora troppo BIG.
Facciamoci aiutare da un semplice programmino scritto in un qualsiasi linguaggio per estrarre un campione casuale che potrà considerasi un NORMAL DATA.

Potremmo usare il NORMAL DATA per ottenere delle tabelle incrociando le variabili che a priori riteniamo interessanti.

Però le variabili da incrociare sono troppe e non siamo sicuri di avere individuato quelle giuste.

Ci convincono ad utilizzare un programma di Reti Neurali non supervisionate per catalogare i miei NORMAL DATA in gruppi fra di loro distinti e con caratteristiche omogenee al loro interno.

Così potrò scoprire le interessanti caratteristiche di gruppi che mai e poi mai avrei immaginato esistessero (clienti anziani scontenti che acquistano comunque).

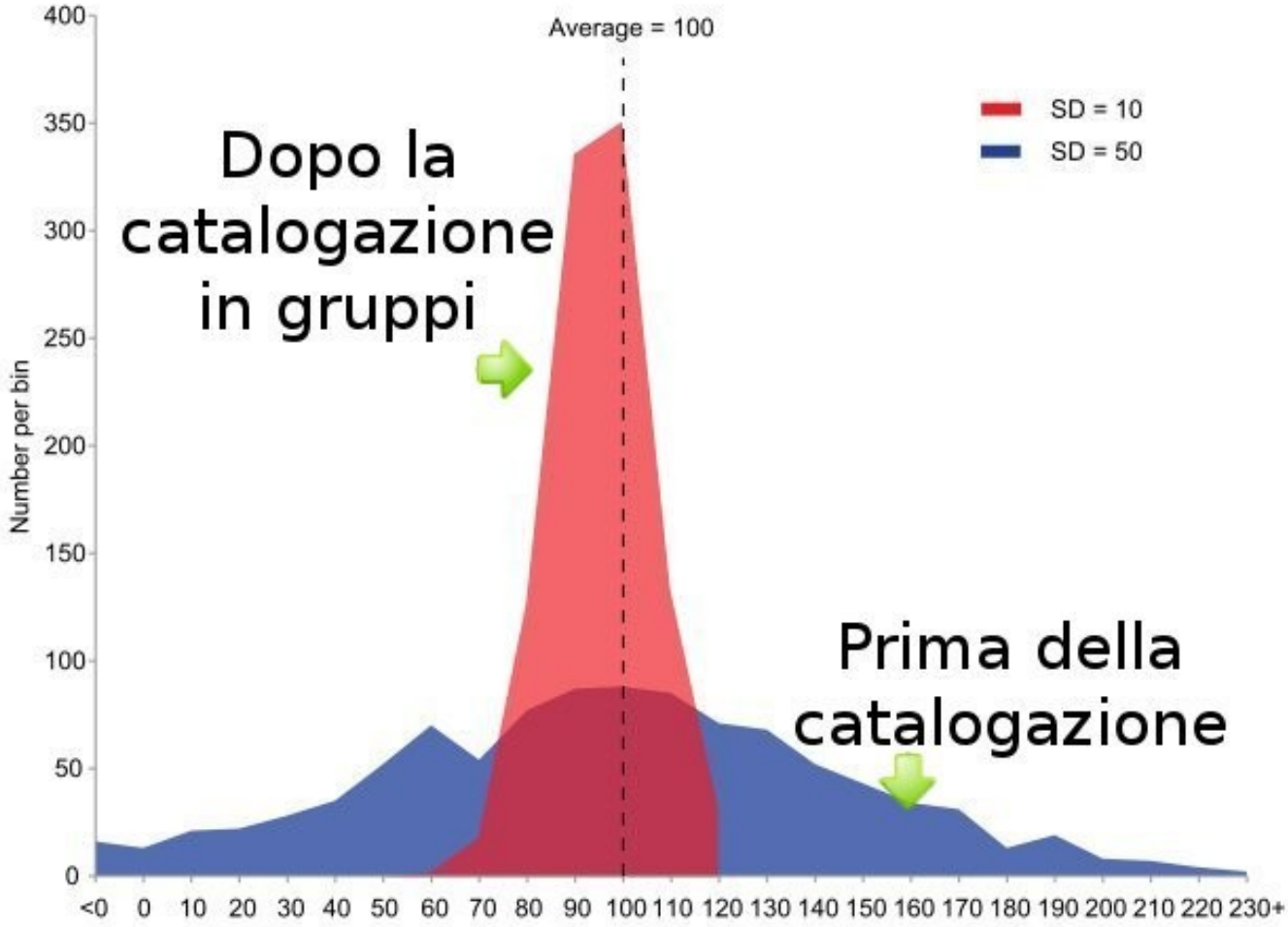
Ma le reti sono neurali e stupide da sempre; loro catalogano in gruppi sempre e comunque, anche se i dati sono stati generati casualmente o sono semplice spazzatura.

Occorre trovare uno strumento che consenta di misurare la qualità della catalogazione in gruppi confrontando il NORMAL DATA di partenza con il NORMAL DATA di arrivo suddiviso in gruppi.

Sono ritornato ai miei lontani studi di statistica.

Ho trovato che il confronto del coefficiente di variazione del NORMAL DATA di partenza con il coefficiente di variazione del NORMAL DATA di arrivo sarebbe stata la soluzione.

Se nel NORMAL DATA di input è contenuta conoscenza, la catalogazione con reti neurali non supervisionate, consente di ottenere dei gruppi con una dispersione dei valori delle variabili rispetto alla loro media significativamente inferiore alla dispersione riscontrabile nelle variabili del NORMAL DATA di input.



La mia soluzione si chiama *pomposamente* KIndex (Knowledge Index).

KIndex (Knowledge Index) è un indice che misura quanta conoscenza sia contenuta nei gruppi catalogati.
Nel caso KIndex raggiunga il valore massimo di 1, ogni gruppo sarebbe composto da record con valori costanti in tutte le variabili / colonne e renderebbe ogni gruppo del tutto distinto dagli altri gruppi.
KIndex è calcolato utilizzando i CV medi-ponderati delle variabili / colonne dei gruppi rapportandoli al CV delle variabili / colonne del file di input prima della catalogazione.

$$KIndex = 1 - \frac{\sum_{i=0}^{n} CV^{(i)} * num^{(i)} \div numtot}{CVpop}$$

Dove il CV è il valore della deviazione standard diviso la media:

$$CV = \frac{\sigma}{\mu}$$

Ecco il programma scritto in Python per il calcolo di KIndex che ha fornito i seguenti risultati:

kIndex input 0.108018012706
KIndex output 0.776356339489

Il file degli animali contiene conoscenza.

```
# -*- coding: utf-8 -*-
# KIndex (Knowledge Index) by Roberto Bello
"""
Achieved cataloging into groups by a SOM neural network,
the question arises whether or not there is knowledge in
the groups, namely whether the groups are between them
distinct and have homogeneous characteristics within each
group.
The use of the coefficient of variation (CV) can be of
help.

KINDEX (Knowledge Index) is an index that measures how much
knowledge is contained in the groups obtained from the SOM
neural network: in the case KINDEX reaches the maximum
value of 1, each group would consist of records with
constant values in all the variables / columns, and each
group would be quite distinct from other groups.
KINDEX is calculated using the weighted-average CV of
variables / columns groups, comparing them to the CV of the
variables / columns of the input file before cataloging.

**********************************************************
Ottenuta la catalogazione in gruppi da una rete neurale di
tipo SOM, sorge il dubbio se nei gruppi esista o meno della
conoscenza, ossia se i gruppi sono fra di loro distinti e
con caratteristiche omogenee all'interno di ogni gruppo.
L'utilizzo del coefficiente di variazione (CV) può essere
di aiuto.
```

```
KIndex (Knowledge Index) è un indice che misura quanta
conoscenza sia contenuta nei gruppi catalogati: nel caso
KIndex raggiunga il valore massimo di 1, ogni gruppo
sarebbe composto da record con valori costanti in tutte le
variabili / colonne e renderebbe ogni gruppo del tutto
distinto dagli altri gruppi.
KIndex è calcolato utilizzando i CV medi-ponderati delle
variabili / colonne dei
gruppi rapportandoli al CV delle variabili / colonne del
file di input prima della
catalogazione.

"""

def mean(x):          # mean
  x = [float(i) for i in x]
  n = len(x)
  mean = sum(x) / n
  if mean == 0.0:
    mean = 0.0000000000001
  return mean

def sd(x):        # standard deviation
  x = [float(i) for i in x]
  n = len(x)
  mean = sum(x) / n
  if mean == 0.0:
    mean = 0.0000000000001
  sd = (sum((x-mean)**2 for x in x) / n) ** 0.5
  return sd

arr0 =
[['*Group*','ANIMAL','FUR','FEATHER','EGGS','MILK','FLYING'
,'AQUATIC','PREDATORY',
'TEETH','VERTEBRATE','POLMONES','POISONOUS','FLIPPERS','LEG
S','TAIL','DOMESTIC'],
['G_00_00','ANTELOPE',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','BUFFALO',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','CALF',1,0,0,1,0,0,0,1,1,1,0,0,4,1,1],
['G_00_00','CAT',1,0,0,1,0,0,1,1,1,1,0,0,4,1,1],
['G_00_00','DEER',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
```

```
['G_00_00','ELEPHANT',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','FIELD_MOUSE',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','GIRAFFE',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','GOAT',1,0,0,1,0,0,0,1,1,1,0,0,4,1,1],
['G_00_00','HAMSTER',1,0,0,1,0,0,0,1,1,1,0,0,4,1,1],
['G_00_00','HARE',1,0,0,1,0,0,0,1,1,1,0,0,4,1,0],
['G_00_00','KANGAROO',1,0,0,1,0,0,0,1,1,1,0,0,2,1,0],
['G_00_00','PONY',1,0,0,1,0,0,0,1,1,1,0,0,4,1,1],
['G_00_00','REINDEER',1,0,0,1,0,0,0,1,1,1,0,0,4,1,1],
['G_00_00','SQUIRREL',1,0,0,1,0,0,0,1,1,1,0,0,2,1,0],
['G_00_00','VAMPIRE',1,0,0,1,1,0,0,1,1,1,0,0,2,1,0],
['G_00_01','CAVY',1,0,0,1,0,0,0,1,1,1,0,0,4,0,1],
['G_00_01','GORILLA',1,0,0,1,0,0,0,1,1,1,0,0,2,0,0],
['G_00_02','BEE',1,0,1,0,1,0,0,0,0,1,1,0,6,0,1],
['G_00_03','CRAB',0,0,1,0,0,1,1,0,0,0,0,0,4,0,0],
['G_00_03','FLY',1,0,1,0,1,0,0,0,0,1,0,0,6,0,0],
['G_00_03','LADYBIRD',0,0,1,0,1,0,1,0,0,1,0,0,6,0,0],
['G_00_03','LOBSTER',0,0,1,0,0,1,1,0,0,0,0,0,6,0,0],
['G_00_03','MIDGE',0,0,1,0,1,0,0,0,0,1,0,0,6,0,0],
['G_00_03','MOLLUSK',0,0,1,0,0,0,1,0,0,0,0,0,0,0,0],
['G_00_03','MOTH',1,0,1,0,1,0,0,0,0,1,0,0,6,0,0],
['G_00_03','POLYP',0,0,1,0,0,1,1,0,0,0,0,0,8,0,0],
['G_00_03','PRAWN',0,0,1,0,0,1,1,0,0,0,0,0,6,0,0],
['G_00_03','STARFISH',0,0,1,0,0,1,1,0,0,0,0,0,5,0,0],
['G_00_03','WASP',1,0,1,0,1,0,0,0,0,1,1,0,6,0,0],
['G_01_00','BEAR',1,0,0,1,0,0,1,1,1,1,0,0,4,0,0],
['G_01_00','BOAR',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','CHEETAH',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','LEOPARD',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','LION',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','LYNX',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','MINK',1,0,0,1,0,1,1,1,1,1,0,0,4,1,0],
['G_01_00','MOLE',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','MONGOOSE',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','OPOSSUM',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','POLECAT',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','PUMA',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_00','WOLF',1,0,0,1,0,0,1,1,1,1,0,0,4,1,0],
['G_01_02','SCORPION',0,0,0,0,0,0,1,0,0,1,1,0,8,1,0],
['G_01_03','FLEA',0,0,1,0,0,0,0,0,0,1,0,0,6,0,0],
['G_01_03','SNAIL',0,0,1,0,0,0,0,0,0,1,0,0,0,0,0],
```

```
['G_01_03','TERMITE',0,0,1,0,0,0,0,0,0,1,0,0,6,0,0],
['G_01_03','WORM',0,0,1,0,0,0,0,0,0,1,0,0,0,0,0],
['G_02_00','DOLPHIN',0,0,0,1,0,1,1,1,1,1,0,1,0,1,0],
['G_02_00','SEAL',1,0,0,1,0,1,1,1,1,1,0,1,0,0,0],
['G_02_00','SEA_LION',1,0,0,1,0,1,1,1,1,1,0,1,2,1,0],
['G_02_01','DUCKBILL',1,0,1,1,0,1,1,0,1,1,0,0,4,1,0],
['G_02_02','TOAD',0,0,1,0,0,1,0,1,1,1,0,0,4,0,0],
['G_02_02','TORTOISE',0,0,1,0,0,0,0,0,1,1,0,0,4,1,0],
['G_03_00','CARP',0,0,1,0,0,1,0,1,1,0,0,1,0,1,1],
['G_03_00','CHUB',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','CODFISH',0,0,1,0,0,1,0,1,1,0,0,1,0,1,0],
['G_03_00','HERRING',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','PERCH',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','PIKE',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','PIRANHA',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','SEAHORSE',0,0,1,0,0,1,0,1,1,0,0,1,0,1,0],
['G_03_00','SEA_SNAKE',0,0,0,0,0,1,1,1,1,0,1,0,0,1,0],
['G_03_00','SHARK',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','SOLE',0,0,1,0,0,1,0,1,1,0,0,1,0,1,0],
['G_03_00','STURGEON',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_00','TUNA',0,0,1,0,0,1,1,1,1,0,0,1,0,1,0],
['G_03_01','FROG',0,0,1,0,0,1,1,1,1,1,0,0,4,0,0],
['G_03_01','TRITON',0,0,1,0,0,1,1,1,1,1,0,0,4,1,0],
['G_03_02','GULL',0,1,1,0,1,1,1,0,1,1,0,0,2,1,0],
['G_03_02','KIWI',0,1,1,0,0,0,1,0,1,1,0,0,2,1,0],
['G_03_02','PENGUIN',0,1,1,0,0,1,1,0,1,1,0,0,2,1,0],
['G_03_03','CHICKEN',0,1,1,0,1,0,0,0,1,1,0,0,2,1,1],
['G_03_03','CROW',0,1,1,0,1,0,1,0,1,1,0,0,2,1,0],
['G_03_03','DOVE',0,1,1,0,1,0,0,0,1,1,0,0,2,1,1],
['G_03_03','DUCK',0,1,1,0,1,1,0,0,1,1,0,0,2,1,0],
['G_03_03','FALCON',0,1,1,0,1,0,1,0,1,1,0,0,2,1,0],
['G_03_03','FLAMINGO',0,1,1,0,1,0,0,0,1,1,0,0,2,1,0],
['G_03_03','HAWK',0,1,1,0,1,0,1,0,1,1,0,0,2,1,0],
['G_03_03','OSTRICH',0,1,1,0,0,0,0,0,1,1,0,0,2,1,0],
['G_03_03','PHEASANT',0,1,1,0,1,0,0,0,1,1,0,0,2,1,0],
['G_03_03','SKYLARK',0,1,1,0,1,0,0,0,1,1,0,0,2,1,0],
['G_03_03','SPARROW',0,1,1,0,1,0,0,0,1,1,0,0,2,1,0],
['G_03_03','SWAN',0,1,1,0,1,1,0,0,1,1,0,0,2,1,0]]

# print "*********input*********"
# print arr0
```

```
# groups are not considered

num_col = len(arr0[0]) - 2
arr0 = arr0[1:]
n = 0
var = []
while n < len(arr0):
  lst1 = arr0[n]
  var.append(lst1[2:])
  n += 1

var         = zip(*var)

means_tot0  = 0.0
sd_tot0     = 0.0
n           = 0

while n < len(var):
  means_tot0 += mean(var[n])
  sd_tot0    += sd(var[n])
  n += 1

cv0 = sd_tot0 / means_tot0

# Groups are considered

recs = len(arr0)-1
groups = []
num_col = len(arr0[0]) - 2

n = 1
while n < len(arr0):
  lst1 = arr0[n]
  groups.append(lst1[0])
  n += 1
groups = list(set(groups))
groups.sort()

group_n     = 0
means_group = 0.0
```

```
sd_group    = 0.0

while group_n < len(groups):
  group = groups[group_n]
  var = []
  r = 0
  while r < len(arr0):
    lst1 = arr0[r]
    if lst1[0] == group:
      var.append(lst1[2:])
    r += 1

  var = zip(*var)
  n = 0
  while n < len(var):
    means_group += mean(var[n])*len(var[n])
    sd_group    += sd(var[n])*len(var[n])
    n += 1
  group_n += 1
means_tot = means_group/recs
sd_tot    = sd_group/recs
cv = sd_tot / means_tot
kindex = 1.0 - cv / cv0

print "kIndex (groups NOT considered)   " + str(1 - cv0)
print "KIndex (groups considered)       " + str(kindex)
```

r.bello@freeopen.org
www.freeopen.org

Laureato in Economia e Commercio con specializzazione in Ricerca Operativa
Data Scientist - Esperto in Knowledge Mining e in linguaggi di programmazione Open Source
ICT Strategist del ClubTI di Milano (www.clubtimilano.net) - Ricercatore dell'Accademia Internazionale di Scienze Forensi (www.accademiascienzeforensi.it) - Perito (CTP) ed ex CTU (Consulente Tecnico di Ufficio) del Tribunale di Milano - Socio fondatore dell'AIPI (Associazione Italiana Professionale di Informatica) - In passato CIO della

Plasmon, della Wrangler in Italia e consulente delle più importanti aziende alimentari italiane
Linkedin: it.linkedin.com/pub/roberto-bello/4/1a5/677
webmaster di www.freeopen.org