Cases — Data Mining (DM)

Walter Kosters, Universiteit Leiden

Friday September 26, 2008 — ICTinBusiness

http://www.liacs.nl/home/kosters/

Before the fact . . .

Crime novels

Crime novels deal with crime — in many ways. There are several sub-genres:

detective/ion Poe, Doyle, Christie (whodunnit)

psychological Rendell (whydunnit; inverted crime novels)

police procedural Sjöwall and Wahlööö, Simenon

private eye Hammett, Chandler

spy Ambler, Le Carré

thriller legal, historical, medical, literary, Da Vinci, . . .

. . .

Doyle (1893):

'Is there any other point to which you would draw my attention?'

'To the curious incident of the dog in the night-time.'

'The dog did nothing in the night-time.'

'That was the curious incident.'

Hammett (1927):

I first heard Personville called Poisonville by a red-haired mucker named Hickey Dewey in the Big Ship in Butte. He also called his shirt a shoit.

Rendell (1977):

Eunice Parchman killed the Coverdale family because she could not read or write.

From a scientific point of view you can approach crime novels from many angles:

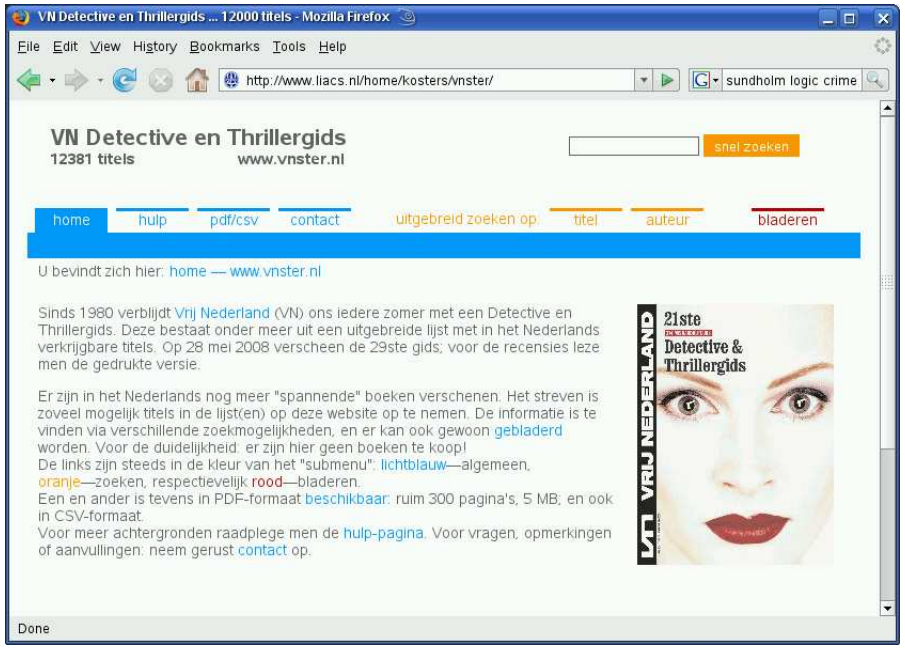
- Why do people like them?
- When is a novel good or bad?
- When is it literature?
- What are the boundaries between (sub-)genres?

But there are also computer science related issues like:

- What is the logic in a novel? (Sundholm)
- How to build recommender systems?

Website

As a sort of escape from science, Hendrik Jan Hoogeboom and I built a website concerning crime novels, starting from the printed "Vrij Nederland Detective & Thrillergids": an extensive ranking system.



`www.vnster.nl`

How to recommend?

Some ICTiB students wrote their Master's Thesis dealing with Data Mining research questions:

- Explanation interfaces in recommender systems
- Artificial immune system for knowledge discovery
- EQPD — A way to improve the accuracy of mining fused data?
- Application of collaborative filtering algorithm in bio-informatics
- Making the right offer to the right customer
- Crime data mining
- Simulation and optimization in analytical and collaborative customer relationship management

Data Mining

Data Mining tries to find interesting and (un)expected patterns in large amounts of (un)ordered data.

Cases (among others):

- Market basket analysis: What and how do we buy?
- Bio-informatics: What is common in DNA?

Problems (selection):

- need for expected results and for unexpected ones
- moving targets
- acquisition of data
- distance notion

The Data Mining process — or rather the KDD process, for Knowledge Discovery in Databases — is usually divided into the following steps:

1. data selection
2. cleaning, e.g., de-duplication and domain consistency
3. enrichment, e.g., data fusion
4. coding
5. Data Mining — the real thing
   this is not intended to be recursive ;-)
6. reporting

Techniques

There are *many* Data Mining techniques, or rather algorithms that can be used to mine data:

- (don't forget:) statistical methods
- all sorts of machine learning techniques from AI: evolutionary computation, neural networks, Bayesian networks, . . .
- all sorts of clustering and classification techniques (decision trees, . . .)
- association rules
- ad-hoc methods

Association rules

We are interested in connections between sets of items: products, or molecules, or visits to websites. Rules should be like: "if you buy bread, you usually buy butter too". And: can you identify a customer by the products he/she buys? (Yes, to a large extent.)

Given a database with n records (customers, transactions), where every record consists of several items (products). The support of such a set of items is the number of records that contain that set, usually divided by n. A set with high support (above some threshold) is called frequent.

For example: 20% of the customers in a supermarket buy both bread, butter and cheese — and possibly more.

Now suppose that out of the α customers that buy a set A, β buy a set B too. Then we can say that the association rule $A \Rightarrow B$ has confidence β/α.

For example: it might be that 80% of the customers that buy butter and cheese, also buy bread.

Now we are interested in rules $A \Rightarrow B$ with both high confidence *and* high support, i.e., high support for the itemset $A \cup B$ (20% in the above case).

Aspects

There is an extensive literature on association rules, in particular on the following aspects:

- efficient algorithms to find them . . .
- how to select the interesting ones . . .
- how to deal with non-Boolean attributes . . .

For this last issue one can use fuzzy logic, where instead of 0/1 (not-buy vs. buy) intermediate values can occur. For example: age can be "young" to an extent of 0.35.

Example

product = item customer = transaction	1	2	3	4	5	6	7	8	9
1	1	1	0	0	1	1	1	1	0
2	1	0	1	0	0	0	0	1	1
3	0	1	1	0	1	0	1	0	0
4	1	0	1	0	1	1	0	1	1
5	0	0	0	0	1	0	0	0	0
6	0	1	0	0	1	0	1	0	0

Even for this small database it is hard to see that the itemset $\{2, 5, 7\}$ is the only set with 3 items that is "bought" by at least 50% (some fixed bound, the so-called support threshold) of the customers.

Frequent itemsets naturally lead to association rules, like $\{2, 7\} \Rightarrow \{5\}$.

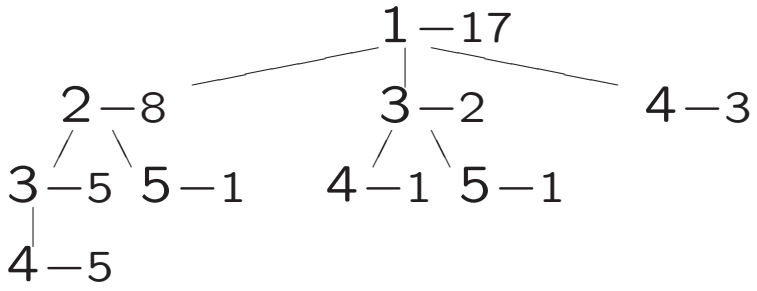
Most fast implementations rely on the following property:

A subset of a frequent set must be frequent too!

This is called the Apriori-property, because it is used in Apriori, one of the most famous Data Mining algorithms. The property is anti-monotone. It is applied like this: small frequent sets are the building blocks for larger ones: first you join them to make candidates, and for these candidates you compute the support.

The current fastest algorithms use FP-trees for condensed representations of the database.

An FP-tree, that contains a database, looks like:



Paths in the tree represent itemsets, including the number of customers that "buy" them. The example tree shows (among other things) that there are $5+1+3 = 9$ customers that "buy" the itemset $\{1, 4\}$ — so its (absolute) support equals 9.

Market basket analysis

Market basket analysis

Many stores try to analyze consumer behavior by doing so-called market basket analysis.
Some possible applications are:

- direct marketing
  "Which customer shall we present some special offer?"

- analysis of current catalog
  "Which products are of interest for a small shop, say in a railway station?"

- clustering and classification
  "Can we group (or recognize) customers by means of their behavior?"

A more general goal might just be a better understanding of the consumers.

Recognizing customers

It turned out that it was quite easy to recognize customers by what they buy. In order to do so, one has to define a symmetric distance measure between customers.

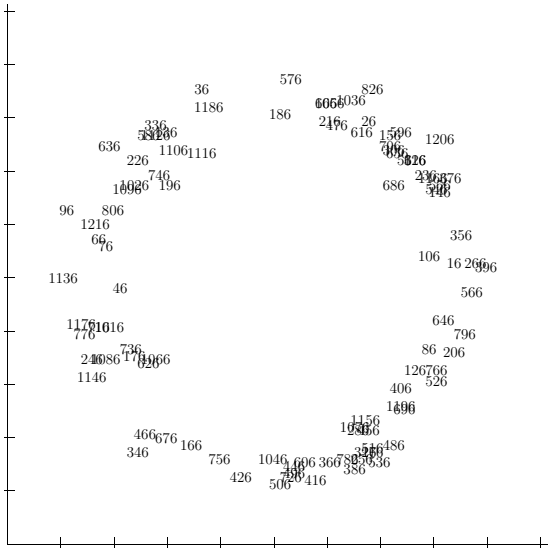
For instance, one can simply let $distance(Cust_1, Cust_2)$ be equal to the number of products that the two customers $Cust_1$ and $Cust_2$ have in common. This can be refined by weights, or by "punishing" differences. We could *for example* define (in set notation; $|Z|$ is number of elements in Z; $U \backslash V$ consists of elements in U that are not in V):

$$distance(X, Y) = |Universe| - |X \cap Y| + |X \backslash Y| + |Y \backslash X| \ .$$

You can "prove" this by the following experiment: if you visit a supermarket, observe the difference between your purchases and those from others, and remember the similarity between your current and previous baskets.

Clustering shops

A large chain of shops wanted to obtain insight in their "positioning", based on accumulated weekly sales. Pictures will look like:



Here we see 100 shops, clustered in the unit square, with respect to their common weekly sales.

In order to cluster, one has to define a distance notion. Now suppose that two shops have the following sales lists for this week:

wine	bread	cheese	sausage	bananas
10	120	34	0	40
7	100	21	0	0

Their distance could be: $3 + 20 + 13 + 0 + 40 = 76$, or $3 + 20 + 13 = 36$, or $3/10 + 20/120 + 13/34$, or . . .
It seems reasonable to normalize for the total sales of a shop.

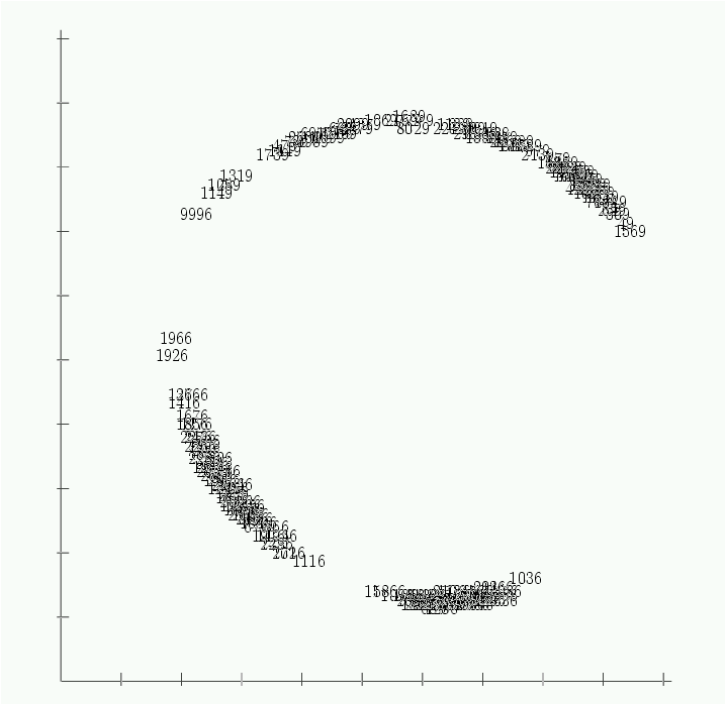
Anyway, there are many possibilities!

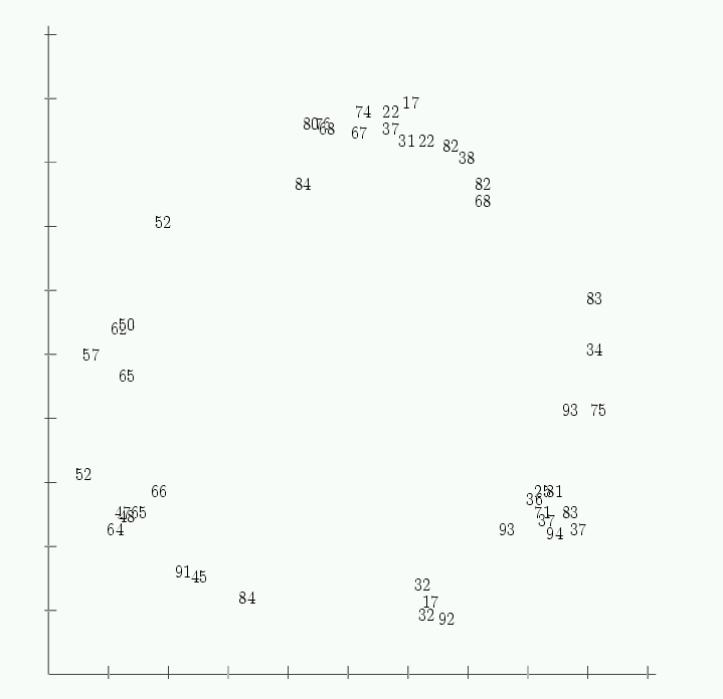
Clustering techniques can be

- hierarchical: repeatedly join smaller clusters to bigger ones
- non-hierarchical: improve existing clustering

We have chosen for a neural network like technique (cf. Self Organizing Maps (SOMs) and Multi Dimensional Scaling (MDS)).
Start from a random positioning of the data points. Then move (in random order) points toward one another if their relative distance is smaller than the Euclidean distance in their current position, and vice versa.

Left we see a clear distinction between shops with and without restaurant. Right we look at zip code; in that case one clearly distinguishes "eating habits".





In a supermarket there are *many* customers who can choose from a *large* stock. Association rules can be easily applied.

"Research" revealed that cigarette paper and tobacco are often bought together (what a surprise!), but also that special types of tobacco required special types of cigarette paper.

There is much interest in hierarchies. E.g., special brands in relation to more general categories.

The diapers—beer connection is a fairy-tale.

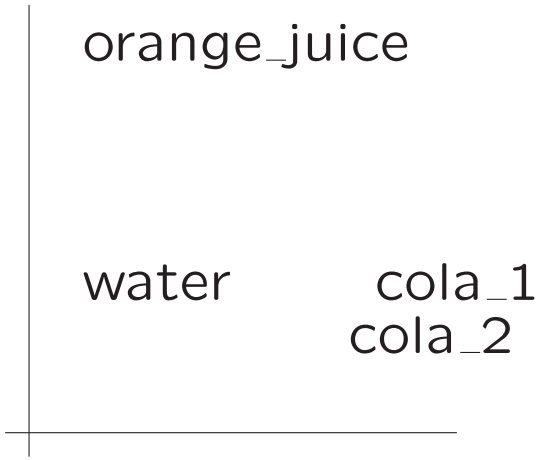
Genetic Algorithms

Yet another interesting problem is the presentation of products in a shop. Is it possible to automate the displays?

A possibility is to use Genetic Algorithms (Evolutionary Computation). We evolve candidate solutions by means of operators like selection, mutation and crossover. A solution (i.e., a possible display of all products) must be properly judged: the so-called fitness function. This is a sum of several properties (spare room, adjacent colors, . . .).

A final application in a supermarket environment is the following. We would like to group products, e.g., beverages, in such a way that products are near if they can "replace" one another.

This can be derived from the shopping carts: such products are hardly ever sold together. Similar techniques as above lead to pictures like:

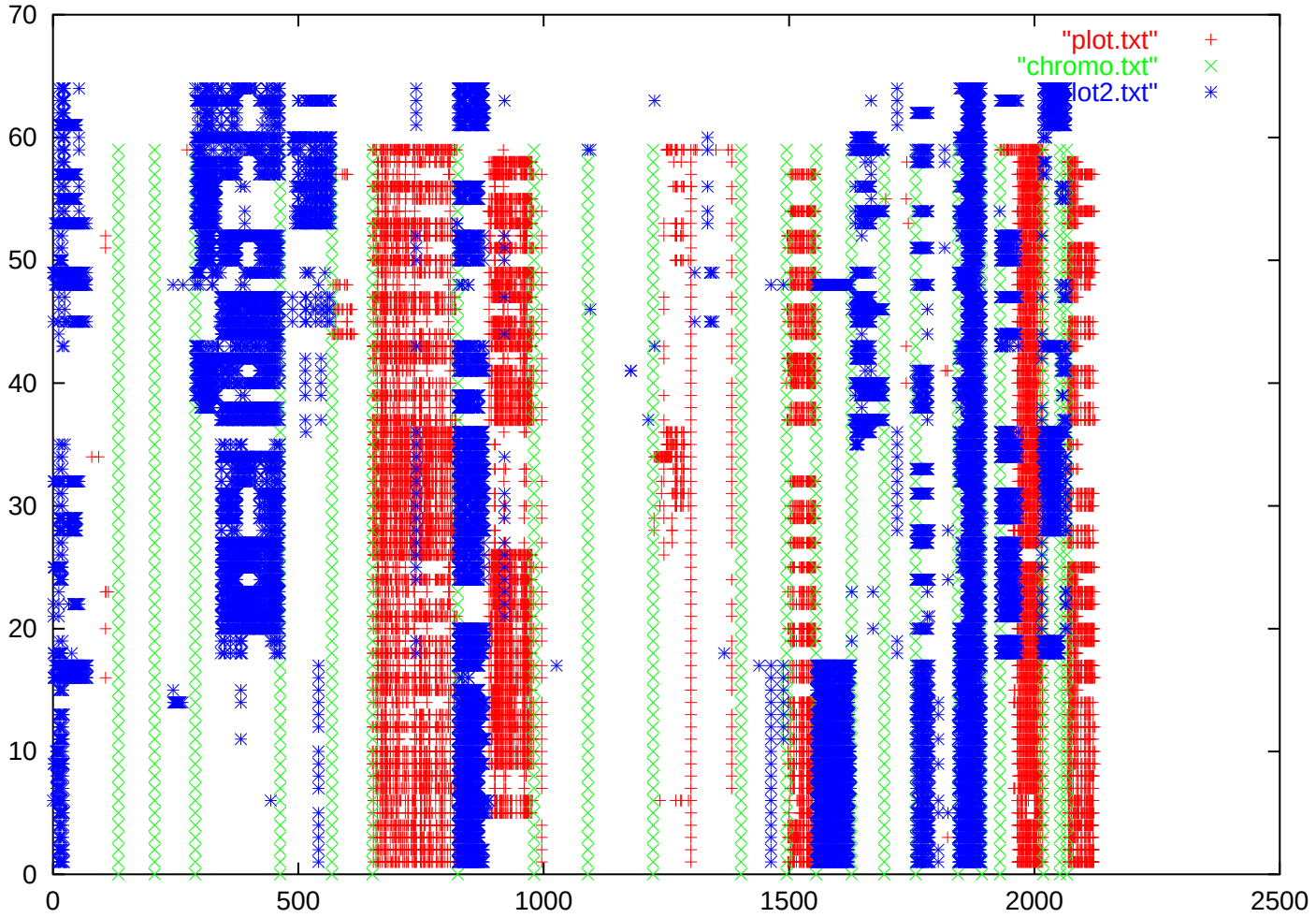


Bio-informatics

Many questions from Bio-informatics can be considered as Data Mining problems. Let us examine a few of them.

In a somewhat simplified way, DNA can be viewed as a string of characters from the alphabet $\{A, C, G, T\}$. Human DNA contains $\approx$ 3,000,000,000 characters — the human genome, divided among some 20+ chromosomes.

Now suppose that for some 2,000+ particular substrings, evenly spread over the human genome, we know how much they are present in 150 patients, say. Here, "how much" is a real number somewhere between -5 ("loss") and $+5$ ("amplification"). How to mine this heap of data?



chromosome boundaries, amplifications, losses

In the plot we see many groups of patients, where each group behaves more or less the same on the 2,000+ given substrings ("clones").

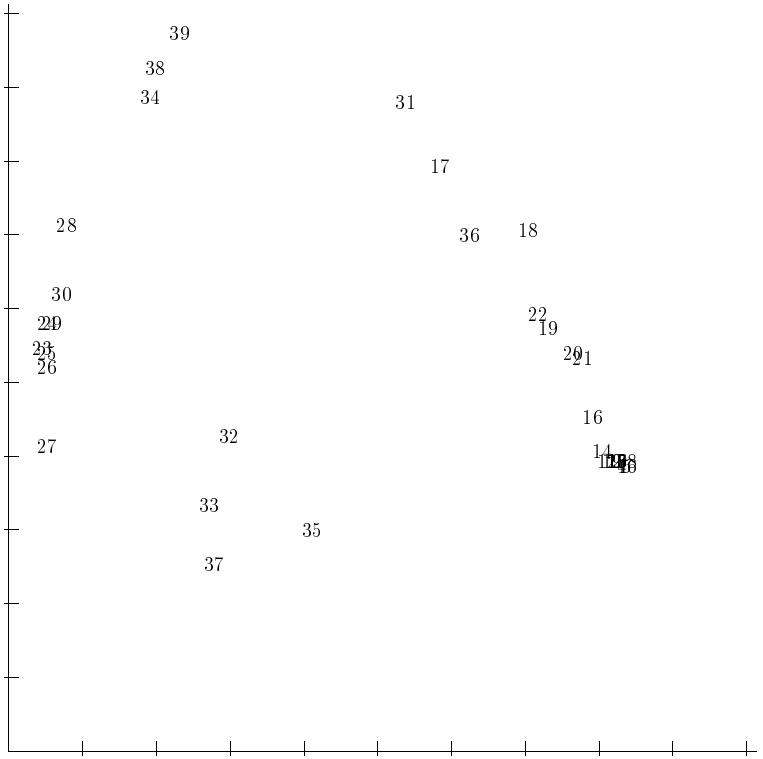
A group of patients is selected (is "frequent") if it satisfies the following property: all patients within the group have > 0.225 (some amplification) on the same c clones, where c is at least some threshold value. This yields the red data points. There are ≈ 60 groups shown.
Analogous for the losses, in blue.

So what we see are — loosely speaking — groups of patients having some (maximal) common behavior.

How to achieve these results — remembering the six KDD steps? And how to interpret them?

The association rule algorithm is easily applied to the data (after writing some Perl conversion script). It then takes some time to tune the parameters, such as the support threshold. After that the presentation deserves attention, see the previous plot.

It appeared that domain experts (biologists from the medical center) would like to see coherence — if any — in the frequent sets. This suggests a clustering. We can use the same technique as before, see the next plot (patients look like products ;-).



We see 39 "frequent sets of patients" being clustered in the unit square with respect to some distance notion: two such sets are near if they behave similar on chromosomes.

Another question from the biologists was: Generate X-thousand substrings of the human genome, such that:

- they are substantially far apart (but not too far)
- they are unique
- they contain "sufficiently many" C's and G's
- they react at the right temperature
- they do not "interfere"
- ...

This requires ad-hoc methods. And suffix trees/arrays ...

So we see successful usage of Data Mining in many areas. Some other application fields:

- insurance companies
- banks (e.g., fraud)

And some more issues:

- privacy
- presentation of results

After the fact . . .

Tetris

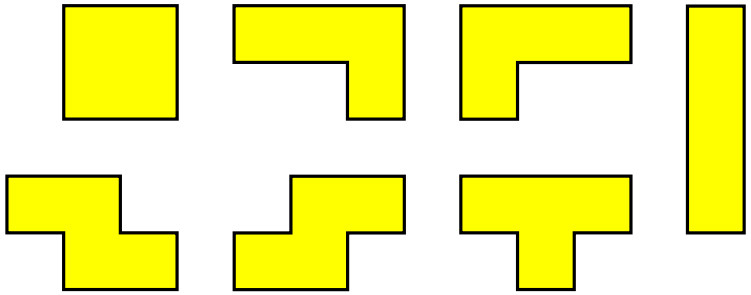
There are several interesting questions attached to a game like Tetris (no connection with Data Mining ;-):

- How to play well? (AI — Artificial Intelligence)
- How hard is it? (complexity)
- What might happen?

It has been shown that certain Tetris-problems are NP-complete (joint work with researchers from MIT), that you can reach almost all configurations, but that not all problems are "decidable".

Tetris: NP-complete?

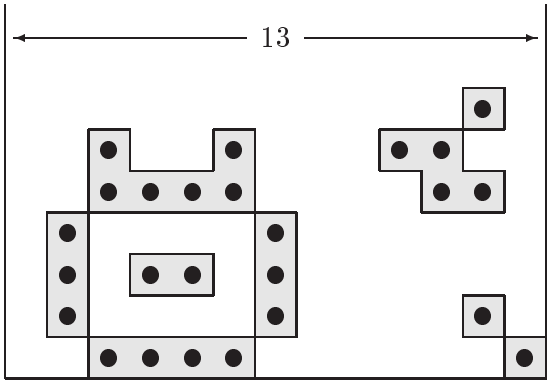
The 7 Tetris-pieces:



Random pieces fall down, and filled lines are cleared.
The question "Is it possible, given a finite ordered series of these pieces, to clear a partially filled game board?" is NP-complete.
If someone clears the board, this is easy to verify. If clearing is *not* possible however, up till now the only thing one can do to prove this is to check all possibilities, one by one!

Tetris: reachable?

An "arbitrary" configuration:



This figure can be made by dropping 276 suitable Tetris-pieces in the appropriate way, see

`http://www.liacs.nl/home/kosters/tetris/`

Claim: on a game board of odd width every configuration is reachable.