Practical Introduction to

Machine Learning and Optimization

Alessio Signorini <alessio.signorini@oneriot.com>

Everyday's Optimizations

Although you may not know, everybody uses daily some sort of optimization technique:

- Timing your walk to catch a bus
- Picking the best road to get somewhere
- Groceries to buy for the week (and where)
- Organizing flights or vacation
- Buying something (especially online)

Nowadays it is a fundamental tool for almost all corporations (e.g., insurance, groceries, banks, ...)

Evolution: Successful Optimization

Evolution is probably the most successful but least famous optimization system.

The strongest species survive while the weakest die. Same goes for reproduction among the same specie. The world is tuning itself.

Why do you think some of us are afraid of heights, speed or animals?

What is Optimization?

Choosing the best element among a set of available alternatives. Sometimes it is sufficient to choose an element that is good enough.

> *Seeking to minimize or maximize a real function by systematically choosing the values of real or integer variables from within an allowed set.*

First technique (Steepest Descent) invented by Gauss. Linear Programming invented in 1940s.

Various Flavors of Optimization

I will mostly talk about heuristic techniques (return approximate results), but optimization has many subfields, as for example:

- Linear/Integer Programming
- Quadratic/Nonlinear Programming
- Stochastic/Robust Programming
- Constraint Satisfaction
- Heuristics Algorithms

Called “programming” due to US Military programs

Machine Learning

Automatically learn to recognize complex patterns and make intelligent decisions based on data.

Today machine learning has lots of uses:

- Search Engines
- Speech and Handwriting Recognition
- Credit Cards Fraud Detection
- Computer Vision and Face Recognition
- Medical Diagnosis

Problems Types

In a search engine, machine learning tasks can be generally divided in three main groups:

- **Classification or Clustering**
  Divide queries or pages in known groups or groups learned from the data. Examples: adult, news, sports, ...
- **Regression**
  Learn to approximate an existing function. Examples: pulse of a page, stock prices, ...
- **Ranking**
  Not interested in function value but to relative importance of items. Examples: pages or images ranking, ...

Algorithms Taxonomy

Algorithms for machine learning can be broadly subdivided between:

- Supervised Learning (e.g., classification)
- Unsupervised Learning (e.g., clustering)
- Reinforcement Learning (e.g., driving)

Other approaches exists (e.g., semi-supervised learning, transduction learning, …) but the ones above are the most practical ones.

Whatever You Do, Get Lots of Data

Whatever is the machine learning task, you need three fundamental things:

- Lots of clean input/example data
- Good selection of meaningful features
- A clear goal function (or good approximation)

If you have those, there is hope for you.

Now you just have to select the appropriate learning method and parameters.

Classification

Divide objects among a set of known classes. You basically want to assign labels.

Simple examples are:

- Categorize News Articles: sports, politics, …
- Identify Adult or Spam pages
- Identify the Language: EN, IT, EL, ...

Features can be: words for text, genes for DNA, time/place/amount for credit cards transactions, ...

Classification: naïve Bayes

Commonly used everywhere, especially in spam filtering.

For text classification it is technically a bad choice because it assumes words independence.

During training it calculates a statistical model for words and categories.

At classification time it uses those statistics to estimate the probability of each category.

Classification: DBACL

Available under GPL at

http://dbacl.sourceforge.net/

To train a category given some text use

dbacl -l sport.bin sport.txt

To classify unknown text use

dbacl -U -c sport.bin -c politic.bin article.txt

OUTPUT: sport.bin 100%

To get negative logarithm of probabilities use

dbacl -n -c sport.bin -c politic.bin article.txt

OUTPUT: sport.bin 0.1234 politic.bin 0.7809

Classification: Hierarchy

When the categories are more than 5 or 6 do not attempt to classify against all of them. Instead, create a hierarchy.

For example, first classify among sports and politics, if sports is chosen, then classify among basketball, soccer or golf.

Pay attention: a logical hierarchy is not always the best for the classifier. For example, Nascar should go with Autos/Trucks and not sports.

Classification: Other Approaches

There are many other approaches:

- Latent Semantic Indexing
- Neural Networks, Decision Trees
- Support Vector Machines

And many other tools/libraries:

- Mallet
- LibSVM
- Classifier4J

To implement, remember: $\log(x*y) = \log(x) + \log(y)$

Clustering

The objective of clustering is similar to classification but the labels are not know and need to be learned from the data.

For example, you may want to cluster together all the news around the same topic, or similar results after a search.

It is very useful in medicine/biology to find non-obvious groups or patterns among items, but also for sites like Pandora or Amazon.

Clustering: K-Means

Probably the simplest and most famous clustering method. Works reasonably well and is usually fast.

Requires to know at priori the number of clusters (i.e., not good for news or results).

Define distance measure among items. Euclidean distance sqrt(sum[(Pi-Qi)^2]) is often a simple option.

Not guaranteed to converge to best solution.

Clustering: Lloyd's Algorithm

Each cluster has a centroid, which is usually the average of its elements.

At startup:

Partition randomly the objects in N clusters.

At each iteration:

Recompute centroid for each cluster.

Assign each item to closest cluster.

Stop after M iterations or when no changes.

Clustering: Lloyd's Algorithm

Desired Cluster: 3 Items: 1,1,1,3,4,5,6,9,11

Random Centroids: 2, 5.4, 6.2

Iteration1:	(2, 5.4, 6.2)	[1,1,1,3] [4,5] [6,9,11]
Iteration2:	(1.5, 4.5, 8.6)	[1,1,1] [3,4,5,6] [9,11]
Iteration3:	(1, 4.5, 10)	[1,1,1] [3,4,5,6] [9,11]

Clustering: Lloyd's Algorithm

Since it is very sensitive to startup assignments, it is sometimes useful to restart multiple times.

When cluster numbers is not known but in a certain range, you can execute the algorithm for different N values and pick best solution.

Software Available:

- Apache Mahout
- Mathlab
- kmeans

Clustering: Min-Hashing

Simple and fast algorithm:

1) Create hash (e.g., MD5) of each word
2) Signature = smallest N hashes

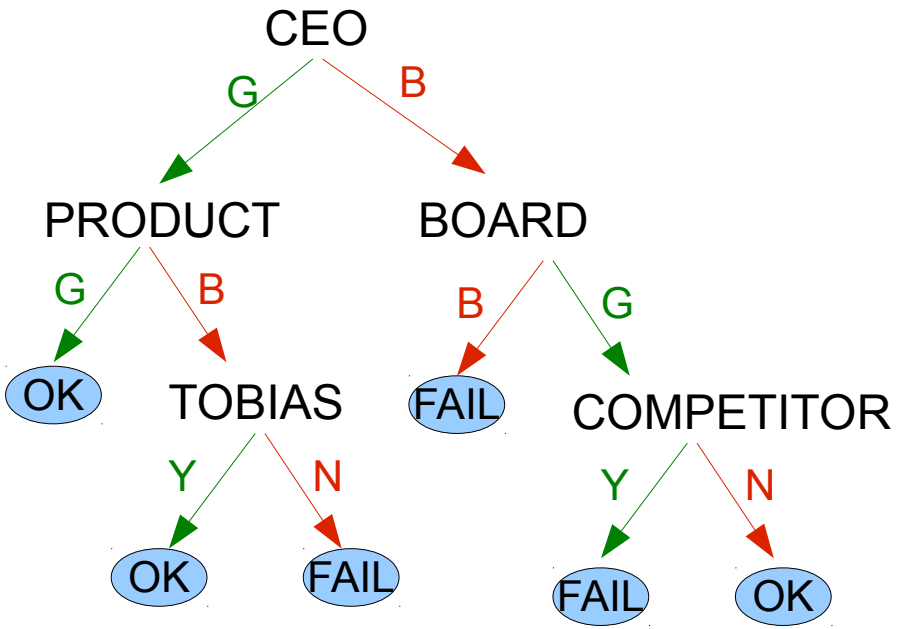
Example:

Similar to what OneRiot has done with its own...
23ce4c4 2492535 0f19042 7562ecb 3ea9550 678e5e0 …
0f19042 23ce4c4 2492535 3ea9550 678e5e0 7562ecb ...

The signature can be used directly as ID of the cluster. Or consider results as similar if there is a good overlap among signatures.

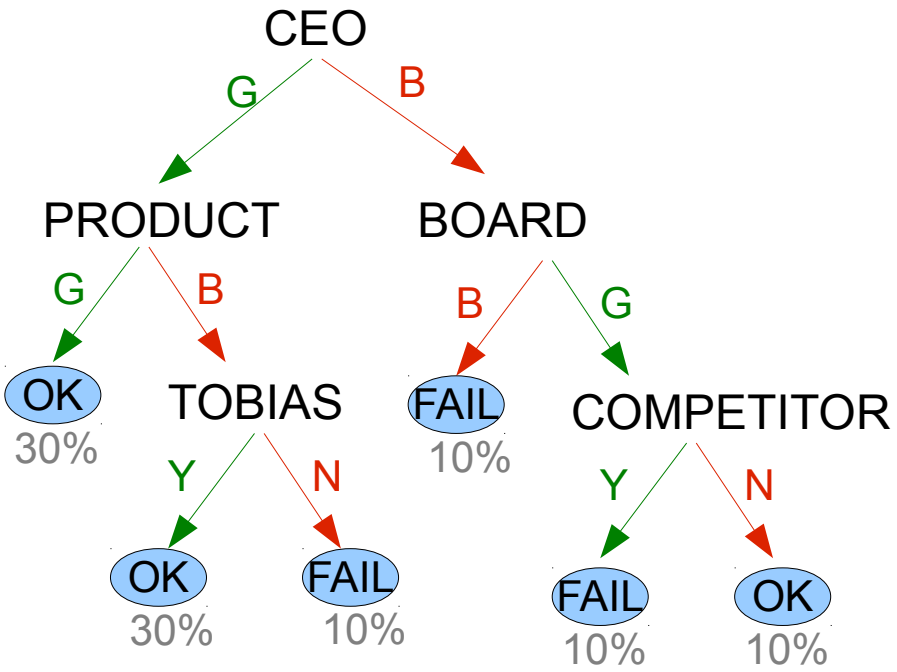
Decision Trees

Decision trees are predictive models that map observations to conclusions on its target output.



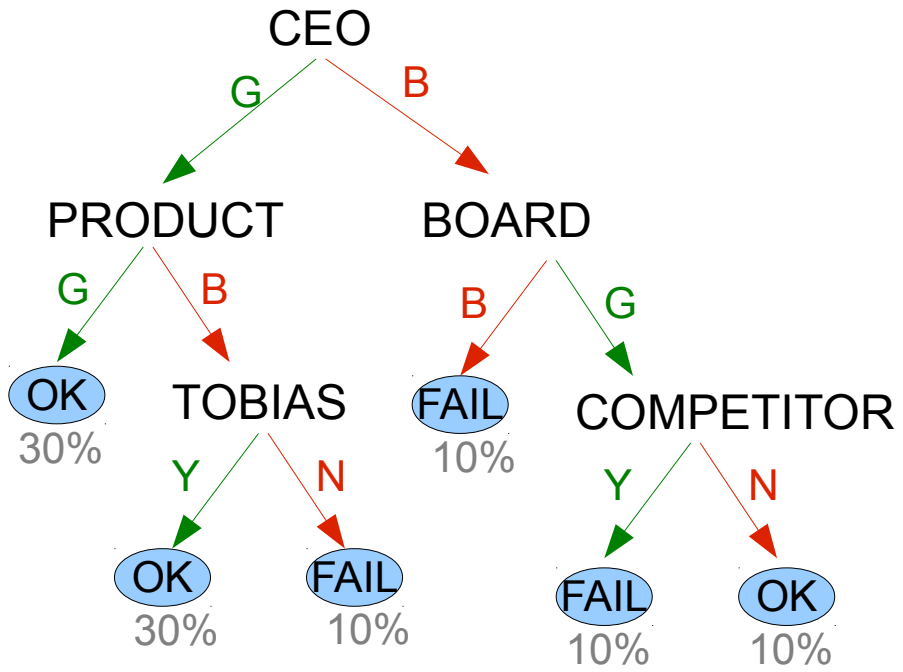
Decision Trees

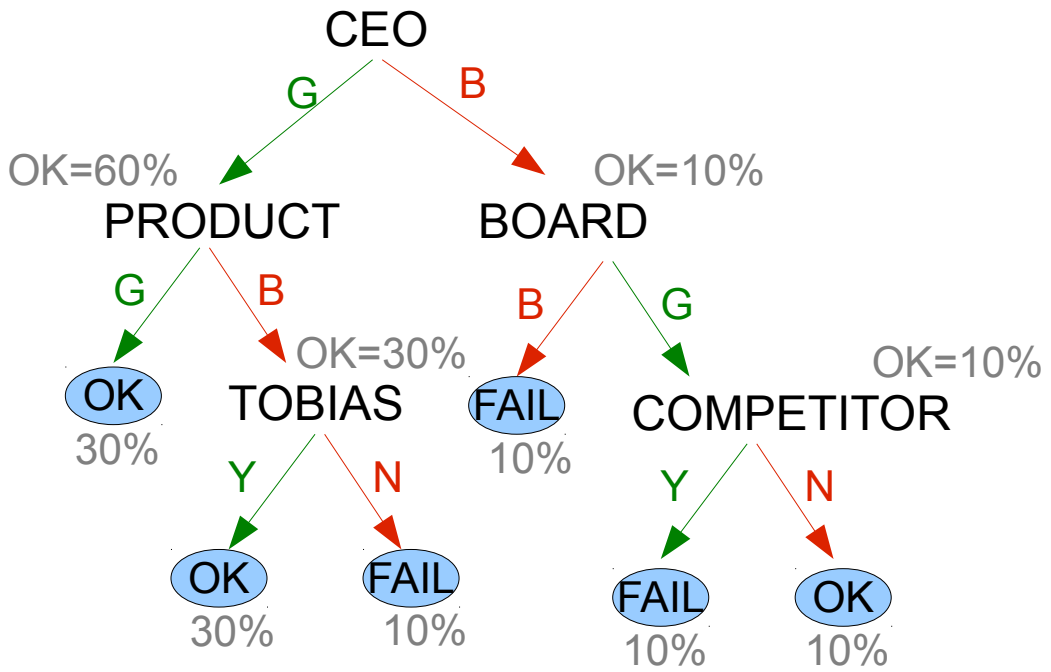
After enough examples, it is possible to calculate the frequency of hitting each leaf.



Decision Trees

From the frequencies, it is possible to extrapolate early results in nodes and make decisions early.





Decision Trees: Information Gain

Most of the algorithms are based on Information Gain, a concept related to the Entropy of Information Theory.

At each step, for each variable V left, compute

$$Vi = (-Pi * \log(Pi)) + (-Ni * \log(Ni))$$

where Pi is the fraction of items labeled positive for variable Vi (e.g., CEO = Good) and Ni is the fraction labeled negative (e.g., CEO = Bad).

Decision Trees: C4.5

Available at

http://www.rulequest.com/Personal/c4.5r8.tar.gz

To train create names and data file

GOLF.names

Play, Don't Play.

outlook:	sunny, overcast, rain.
temperature:	continuous.
Humidity:	continuous.
Windy:	true, false.

GOLF.data

sunny, 85, 85, false, Don't Play
sunny, 80, 90, true, Don't Play
overcast, 83, 78, false, Play
rain, 70, 96, false, Play
rain, 65, 70, true, Don't Play
overcast, 64, 65, true, Play

Then launch

c4.5 -t 4 -f GOLF

Decision Trees: C4.5 Output

Cycle	Tree size	Cases: window	Cases: other	Errors: window	Errors: rate	Errors: other	Errors: rate	Errors: total	Errors: rate
1	3	7	7	1	14.3%	5	71.4%	6	42.9%
2	6	9	5	1	11.1%	1	20.0%	2	14.3%
3	6	10	4	1	10.0%	2	50.0%	3	21.4%
4	8	11	3	0	0.0%	0	0.0%	0	0.0%

```
outlook = overcast: Play
outlook = sunny:
|   humidity <= 80 : Play
|   humidity > 80 : Don't Play
outlook = rain:
|   windy = true: Don't Play
|   windy = false: Play
```

Trial	Before Pruning: Size	Before Pruning: Errors	After Pruning: Size	After Pruning: Errors	After Pruning: Estimate	
0	8	0(0.0%)	8	0(0.0%)	(38.5%)	<<
1	8	0(0.0%)	8	0(0.0%)	(38.5%)	

Support Vector Machine

SVM can be used for classification, regression and ranking optimization. It is flexible and usually fast.

Attempts to construct a set of hyperplanes which have the largest distance from the closest datapoint of each class.

The explanation for regression it is even more complicated. I will skip it here but there are plenty of papers available on the web.

SVM: svm-light

Available at

http://svmlight.joachims.org/

To train an SVM model create data file

RANKING

```
3 qid:1 1:0.53 2:0.12 3:0.12
2 qid:1 1:0.13 2:0.1 3:0.56
1 qid:1 1:0.27 2:0.5 3:0.78
8 qid:2 1:0.12 2.077 3:0.91
7 qid:2 1:0.87 2:0.12 3:0.45
```

REGRESSION

```
1.4 1:0.53 2:0.12 3:0.12
7.2 1:0.13 2:0.1 3:0.56
3.9 1:0.27 2:0.5 3:0.78
1.1 1:0.12 2.077 3:0.91
9.8 1:0.87 2:0.12 3:0.45
```

Then launch

svm_learn pulse.data pulse.model

SVM: Other Tools Available

There are hundreds of libraries for SVM:

- LibSVM
- Algorithm::SVM
- PyML
- TinySVM

There are executables built on top of most of them and they usually accept the same input format of svm-light.

May need a script to extract features importance.

Genetic Algorithms

Genetic Algorithms are flexible, simple to implement and can be quickly adapted to lots of optimization tasks.

They are based on evolutionary biology: strongest species survive, weakest die, offsprings are similar to parents but may have random differences.

This kind of algorithms is used wherever there are lots of variables and values and approximated solutions are acceptable (e.g., protein folding).

GA: The Basic Algorithm

Startup:

Create N random solutions

Algorithm:

1) compute fitness of solutions
2) breed M new solutions
3) kill M weakest solutions

Repeat the algorithm for K iterations or until there is no improvement.

GA: The Basic Algorithm

During breeding (step 2) remember to follow biology rules:

- Better individuals are more likely to find a (good) mate
- Offsprings carry genes from each parent
- There is always the possibility of some random genetic mutations

GA: Relevance Example

Each solution is a set of weights for the various attributes (e.g., title weights, content weight, ...).

The fitness of each solution is given by the delta with editorial judgments (e.g., DCG).

During breeding, you may take title and content weight from parent A, description from parent B, …

When a mutation occurs, the weight of that variable is picked at random.

One of the Problems I Work On

We have a set of users and for each we know the movies they like among a given set.

Extrapolate the set of features (e.g., Julia Roberts, Thriller, Funny, ...) that each movie has so that it is liked by all the users which like it.

Not interested in what the features are or represent: we are fine with just a bunch of IDs.

This could save/improve the life of lots of people.