Machine Learning Thoughts

Some thoughts about philosophical, theoretical and practical aspects of Machine Learning.

The Failure of AI

In the early days of AI, scientists thought they would be able to build an intelligent computer by the end of the 20th century. This raised various fears about computers eventually taking over the world, and human beings replaced by robots.
Not only this did not happen yet but we are very far from this!
But what is even worse is that we are now following a somewhat opposite trend. There are many tasks at which humans are far better than computers, but instead of trying to build better algorithms for these tasks, people are now trying to find ways to better make use of human intelligence, or rather to automate this usage!

Two examples of this: Luis von Ahn's "Artificial artificial intelligence" and Amazon's "Mechanical Turk".

Luis von Ahn has designed a couple of internet games whose purpose is to make players perform useful tasks such as labeling images. Amazon is taking this to the industrial scale (although not as a game anymore) by allowing people to design programs which include calls to web services which are actually executed by paid humans (e.g. your program calls "translate(text)" and the text is sent to someone who translates the text and return the result)!

Update (21/12/2006): I found a related article in the Boston Globe, citing other examples of the same kind, such as Mozes Mob, a cell phone Q&A service powered by humans.

November 30, 2006 in Artificial Intelligence, Data Mining, Machine Learning | Permalink | Comments (45) | TrackBack (1)

Machine Learning Search Engine

Google custom search engines are a very nice idea.
I have started to create such a custom search engine for Machine Learning research. It is still preliminary, so feel free to suggest URLs or categories to add to it.
Here is the link (Update: I removed the search box from this blog as it did not work, you should use the above link to try it out)

November 30, 2006 in Machine Learning, search engine | Permalink | Comments (29) | TrackBack (0)

Machine Learning Videos

I have just created a page with a selection of cool Machine Learning videos (mainly talks given at Google).
Feel free to suggest some more to add to this page...

November 28, 2006 in Machine Learning | Permalink | Comments (43) | TrackBack (0)

Making Machine Learning More Scientific

The title of this post may seem awkward as most people working on ML would consider themselves as scientists.
However, ML is still a young field, and as most scientific fields in their youth, the methods and practices are still being defined and formalized. Indeed, scientific fields usually start unorganized and, as they mature, begin to have proper definitions for their goals, proper vocabulary and proper methods for verifying their results

My feeling is that Machine Learning is still at an early age of its development. Many things are still lacking and I will try to list some of them:

- Foundations
  - Agreed-upon vocabulary: there are many different fields that concurred to the infancy of the domain, they all have different ways of calling the same object, for example: function, model, hypothesis, concept... It would be nice to speak the same language.
  - Statement and definition of the goals: what is the object under study?
  - Statement of main problems: providing a list of problems in a clear and formalized way (this has already started with the open problems sessions at the COLT conference, but there is yet to be a consensus about which ones are more important for advancing the field).
- Experimental methods
  - Common language for specifying algorithms and experiments so that they can be reproduced. Indeed, too often, the papers describe results that no one can reproduce because some specific ad-hoc tuning of the parameters was used. I recently found this article which describes an attempt to provide a language for automating experiments.
  - Define and agree upon a set of measures or criteria that are relevant to assess the success of an algorithm (there could be many, depending on the field of application, the idea would be to use as many as possible rather than focusing only on a handful)
  - Datasets: go beyond the UCI database, share the data, define a common standard for this data or provide tools to convert from one format into another.
  - An idea could be to run evaluations more or less like a challenge (as the ones that have recently been proposed, for example by the Pascal network), but with a way to share even the code of the algorithm: you could send your code to a server and the server runs it on many databases, and measures many different criteria, generates a report and adds the entry to a database, thus creating a big source of data for meta-learning.
  - The goal is to say which algorithm is better for which problems, but not in general, and especially to avoid the dataset selection issue.
- Knowledge management
  - Write more introductory papers, or textbooks.
  - Collect and maintain relevant bibliographic references.
  - Revisit what has been done or discovered so far with a fresh look: avoid repeating always the same things without trying to understand them or putting them under a new light (e.g. "SVM are great because they are well-founded", "Boosting is maximizing the margin", "Fisher's discriminant is valid only when the classes are distributed as Gaussians", "What if the data is not iid", "Bayes rule is optimal")

Of course, this may take some years and a lot of effort, but hopefully, as more money is poured into Machine Learning research and applications, this will happen...

June 20, 2006 in Machine Learning | Permalink | Comments (39) | TrackBack (0)

Extracting Information from People

It seems natural that the goal of any good Machine Learning algorithm should be to extract information from the available data.
However, when you are faced with practical problems, this is not enough. More precisely, data by itself does not hold the solution. One needs "*prior knowledge*" or "*domain knowledge*".
So far, nothing new.

But what is important is how to actually get and use this knowledge, and this is very rarely addressed or even mentioned!
My point here is that building efficient algorithms should mean building algorithms that can extract and make maximum use of this knowledge!
To achieve this, here are some possible directions:

- A first step is probably to think about what are the natural "*knowledge bits*" one may have about a problem and how to formalize them. For example, it can be knowledge about how the data was collected, what the features mean, what kind of errors can be made in the data collection,...
- A second step is to provide simple but versatile tools to encode prior knowledge: this can be done off-line, for example when using a probabilistic framework one can allow the probability distributions to be customized, or on-line (i.e. interactively) with a trial-and-error procedure (based on cross-validation or on expert validation).
- There is also a possibility to go one level higher: often, knowledge is gained by integration of very diverse sources of information, humans (as learning systems) are never isolated: all problems they can solve have some relationship to their environment. So ideally our systems should be able to integrate several sources and have some sort of meta-learning capability rather than starting from scratch every time a new dataset is to be used, and focusing only on this specific dataset.

All the above explains the title of my post, and to be more precise, I even tend to think that research efforts should be focused on **knowledge extraction from experts rather than from data!!!**

Finally, I would like to give examples of such an extraction (we are not talking about sitting experts in a chair with electrodes connected into their brains! but just about providing software that can interact a bit with them).
Below is a (non-exhaustive) list of what can be learned from the user by a learning system:

- Implicit knowledge (when the data is collected and put in a database)
  - data representation: the way the data is represented (the features that are used to represent the objects) already brings a lot of information and often a problem is solved once the appropriate representation has been found.
  - setting of the problem: the way the problem is set up (i.e. the choice of which variables are the inputs and which are the outputs, the choice of the samples...) also bring information.
- Basic information (when the analysis starts)
  - choice of features: choosing the right features, ignoring those that are irrelevant...
  - choice of samples: choosing a representative subset, filtering...
  - choice of an algorithm
  - choice of parameters for this algorithm
- Structural knowledge (usually incorporated in the algorithm design phase)
  - design of kernels, prior distribution
  - design of the algorithm
  - invariances
  - causal structures
- Interactive knowledge: all the above can be repeated by iteratively trying various options. Each trial can be validated using data (cross-validation) or expertise (judging the plausibility of the built model).

As a final remark, let me just mention that the interactive mode is often used (although not explicitly) by practitioners who try several different algorithms and take the one that seems the best (on a validation set). Of course this gives rise to the risk of overfitting, especially because the information brought by the interaction is very limited. Indeed, it simply amounts to the validation error which cannot be considered as knowledge: this kind of interaction simply brings in more data (the validation data) rather than more knowledge.
It would probably be interesting to formalize a bit better these notions...

April 18, 2006 in Data Mining, Machine Learning, Philosophy | Permalink | Comments (2) | TrackBack (1)

Machine Learning Blogs

Blogging is becoming increasingly popular including among Machine Learning researchers.

Here are some interesting blogs about ML:

- Yet Another Machine Learning Blog (Pierre Dangauthier)
- Machine Learning Devotee (Mahdi Shafiei)
- Business Intelligence, Data Mining & Machine Learning (José Carlos Cortizo Pérez)
- Predict This! (Tilmann Bruckhaus)
- Statistical Modeling, Causal Inference, and Social Science (Andrew Gelman)
- Natural Language Processing (Hal Daumé)
- MaLi @ backprop.net (Robert Wall)

Some other blogs deal with related topics (although less directly connected):

- Decision Science News
- Intelligent Machines (Damien François)
- Enterprise Decision Management (James Taylor)

February 14, 2006 in Machine Learning | Permalink | Comments (2) | TrackBack (0)

Be Rational

Usually, performing inductive inference occurs in two steps. The first one consists in constructing a set of assumptions that summarize the knowledge one has about a phenomenon of interest prior to observing instances of this phenomenon. The second one consists in actually observing these instances and deriving new knowledge from this observation.

A possible question is: what principle may guide each of these steps?
A possible answer is: be as rational as possible. In other words, try to avoid inconsistencies.

Regarding the second one, it is sometimes possible to formulate the problem as a purely deductive one. Indeed, the question is "given such assumptions and given such data, what can I deduce?". For example, in a probabilistic framework, one would have a prior distribution and observations and would aim at obtaining an updated distribution. The rational way of doing this is to apply Bayes rule.
In other settings, when the assumptions are not formulated in a probabilistic language, or when the objective is to optimize some sort of worst-case performance, other rules could be used.
The point is that once the objective is clearly and formally specified, rationality naturally leads to the solution via pure deduction.

Regarding the first one (constructing the assumptions), the situation is less obvious. There are guiding principles though, which again rely on rationality.
One such principle is the one of **symmetry**: if there is no reason to prefer one side of a coin to the other (or to assume that both faces would have different properties), simply consider them equally probable. A more elaborated version of this principle is the principle of **maximum entropy**: when choosing a prior distribution over a set of possibilities, choose, among the ones that are consistent with your prior beliefs, the one with maximum entropy.
Finally, there is also the principle of **simplicity** (Occam's razor) which suggests to give more prior weight to the simple hypotheses than the complex ones.

However, all these principles cannot be justified in a formal way. One can surely construct settings where applying one specific principle is the "best" thing to do, but this is somewhat artificial and does not provide a justification.

Instead of proving things, I guess the best thing to do is to provide recommendations. One such recommendation is "be rational", or in other words, try to take into account every piece of evidence you may have before observing the data and to do this in a way that does not lead to contradictions and does not expose you to more risk than you are willing to accept. So in a way, inferences should take into account both your knowledge and your uncertainty and be calibrated according to what you accept to loose if you fail.
I like the idea that performing an inference is like horse race **gambling**: you try to get as much information

you can about the horses, but you know there will always be some missing piece of information. Even if gambling is somewhat irrational, when you have no choice but to do it, better do it in the most rational way!

January 30, 2006 in Machine Learning, Philosophy | Permalink | Comments (4) | TrackBack (0)

More is Less

When we try to understand the causes of a phenomenon, there is a natural tendency to think that the more variables we measure the more likely we are to identify the real cause.
This is generally true if there is no constraint on the amount of experiments we can realize, but as soon as we work with a limited sample, adding more variables may lead to less accurate models.
More precisely, if we have the ability to perform say 100 independent experiments and for each of these experiments we measure d "input" variables and one output variable and try to build a model to predict the output from the inputs, then the larger d is, the more difficult it might be to build the model. There is some kind of optimal value of d: on the one hand, if one has measured too few variables, one may miss important information, on the other hand, if one has measured too many variables, it becomes impossible to distinguish between "true" correlations and distortions caused by the imperfection of the sampling mechanism (for example, it may happen that on these specific 100 experiments, one input variable that has nothing to do with the output is incidentally correlated to the output).

In order to study this phenomenon more precisely, one can imagine to have a framework where, in addition to obtaining the experiments from sampling iid from an arbitrary distribution, one considers that the variables themselves are obtained from a sampling process (also iid from some distribution). This is similar in spirit to the framework recently proposed by Krupka and Tishby, except that in their framework, only the variables (or features) are sampled, and not the examples. Note that the assumption that one samples the variables in an iid fashion does not mean that the variables are necessarily independent in the classical sense of having independent values. One should imagine an infinite matrix representing all possible measurements on all possible experiments for a given problems: rows would be experiments, columns would be measurements.
The framework consists in assuming that one randomly picks n rows and d columns from this matrix (possibly picking several times the same row or column to comply with the iid assumption).

Now the question is: can you obtain a bound on the error of a given learning algorithm when trained on such an (n,d) sample as a function of n and d?

This is probably still too vague to be answered, and one probably needs to put restrictions on the functions that are allowed. For example, a reasonable first goal would be to study the case where the target function is a (possibly countably infinite) linear combination of the variables (i.e. columns of that infinite matrix). Intuitively, one would expect that the result depends on "how similar" the columns of the matrix are, and "how wide-spread" the coefficients of the linear combination are: you need to collect enough variables with high coefficients in the final linear combination, but not too many variables with low or zero coefficient.

January 25, 2006 in Machine Learning | Permalink | Comments (1) | TrackBack (0)

Bayesian brain

In a recent issue of The Economist, there is a very nice article (see here) about how everyday reasoning can be compared to Bayesian inference.
This article is based on a recent paper by Griffiths and Tenenbaum (see here). What they have done is to ask questions such as "How long do you think a man who is xx years old will live?" to several people. It turns out that the answers matched very well with those which would have been obtained by applying Bayes rule. Even more, they tried this with several different types of questions, for which the implicit priors are very different (Gaussian, Erlang or power-law distributions) and in all cases, the intuitive answers given by people had the right form (in terms of distribution).

What they conclude from this is that the way people intuitively reason about the world is quite similar to applying Bayesian inference.

What is intriguing is that the article in The Economist tries to see there a proof of domination of the Bayesian over the frequentist point of view. Also in the paper of Griffith and Tenenbaum, they use the term "optimal" when they talk about Bayes rule. I think this is very misleading and inaccurate.

Indeed, the only conclusion one should draw from this study is that the way people naturally make inferences about events in the world is very much rational and this confirms the fact that has been observed many times before that the intuitive notion of rationality we have match very well with the rules of the calculus of probabilities.
But this is no surprise because these rules were designed in order to be intuitively rational (what else?). What is interesting is that rationality leads necessarily to these rules and no other, but this has been known for years.

I do not see what this study has to do with the debate between Bayesian vs frequentist. First of all, there is no real opposition between these points of view. Indeed, they lead to the same rules for combining probabilities, the only difference is in the meaning that is associated to these probabilities. So this debate is mostly philosophical and should not interfere with cognitive science studies, nor (even less) with machine learning.

January 24, 2006 in Machine Learning, Philosophy | Permalink | Comments (8) | TrackBack (0)

When does sparsity occur?

Sparsity is a very useful property of some Machine Learning algorithms. Such an algorithm yields a sparse result when, among all the coefficients that describe the model, only a small number are non-zero. This is typically associated with interesting properties such as fast evaluation of the model (see the reduced set methods for obtaining sparse kernel expansions), fast optimization (e.g. in SVM, many algorithmic approaches exploit this fact), statistical robustness (sparsity is usually associated to good statistical performance), or other computational advantages (e.g. ability to compute full regularization paths, for example in LASSO-style regression).

However I have not seen a clear explanation of this phenomenon. My feeling (I have no proof but it seems intuitively reasonable) is that sparsity is related to the regularity of the criterion to be optimized.
More precisely, the less regular the optimization criterion, the more sparse the solution may end up being.

The idea is that, for sparsity to occur, the value 0 has to play a special role, hence something unusual has to happen at the value 0. This something can be a discontinuity of the criterion or of one of its derivatives.

If the criterion is discontinuous at 0 for some variables, the solutions might get "stuck" in this value (provided it is a local miminum of course). If instead, the criterion is continuous but has a derivative which is discontinuous at 0, it means that the criterion is V-shaped at 0, so that solutions might be "trapped" at this point. If we continue the reasoning, we see that the "attraction" of the point 0 is less and less effective as the regularity increases. When the function is twice differentiable everywhere, there is not any reason for the solution to be "trapped" at 0 rather than ending up somewhere else.

This reasoning partly explains the sparsity of SVMs. Indeed, the standard L1-SVM (hinge loss) have a discontinuous criterion, while for L2-SVM (squared hinge loss), the criterion has a discontinuous derivative and finally, for the LS-SVM (squared loss), the criterion is twice differentiable. It turns out that the most sparse is the L1 version and then the L2 version, while for LS-SVM there is no sparsity at all.

The same reasoning applies when one compares penalized least squares regression: when the penalization is the L2-norm of the weights, there is no sparsity, while with the L1-norm, the sparsity occurs, and for the L0-norm there is even more sparsity.

I am wondering whether there is any mathematical treatment of these issues anywhere in the Machine Learning litterature. If anyone has a pointer, please let me know.

November 08, 2005 in Machine Learning, Theory | Permalink | Comments (32) | TrackBack (0)