Machine Learning for Automatic Environmental Mapping: When and How

Nicolas Gilardi
Particle Physics Experiment Group
University of Edinburgh
Edinburgh EH9 3JZ, UK
ngilardi@ph.ed.ac.uk
www.ph.ed.ac.uk/~ngilardi
and
Samy Bengio
IDIAP Research Institute
CP 592, rue du Simplon 4,1920 Martigny, Switzerland
bengio@idiap.ch
www.idiap.ch/~bengio

30 June 2005

Abstract

This paper discusses the opportunity of using Machine Learning techniques in an automatic environmental mapping context, as was the case for the SIC2004 exercise. First, the Machine Learning methodology is quickly described and compared to Geostatistics. From there, some clues about when to apply Machine Learning are proposed, and what outcomes can be expected from this choice. Finally, three well known regression algorithms: K-Nearest Neighbors, Multi Layer Perceptron and Support Vector Regression, are used on SIC2004 data in a Machine Learning context, and compared to Ordinary Kriging. This illustrates some potential drawbacks of SVR and MLP for applications such as SIC2004.

1 Introduction

It is a common mistake in some fields (such as Geostatistics) to confuse *Machine Learning* and *Artificial Neural Networks*[1] (ANNs). Machine Learning can be seen as a methodology (similarly to Geostatistics), while ANNs are a family of modeling functions (like Kriging).

Algorithms such as ANNs are very powerful estimators, and without a very careful methodology to control them, they can end up predicting something very different from the expectations. This is partly the origin of the bad reputation ANNs have in some scientific communities.

Machine Learning allows to control efficiently the level of information an ANN is modeling. But it is a very general methodology and can also be used with other types of modeling functions, from the very simple and general *K-Nearest Neighbors* [5], to the more complex *Support Vector Machines* [17] (for static data), or *Hidden Markov Models* [13] (for sequential data). In fact, one can use almost any family of statistical modeling functions in the context of Machine Learning.

However, now that the Machine Learning methodology is becoming more and more understood, a new tendency seems to be the application of complex algorithms on problems which do not always need them. Environmental mapping may be one of the fields concerned by such a tendency.

Thanks to the increasing concern about environmental issues, there is now a huge amount of data produced by nearly real time monitoring of pollution problems such as radioactivity or exhaust gas emissions. Fast and efficient estimators are thus necessary to produce high quality decision maps, and the objective of SIC2004 [3] was to make a review of the state-of-the-art in the domain.

In such a situation it is natural to think about applying Machine Learning methodology. But before doing that, it is necessary to verify whether there is a *need* for it, and if so, which algorithm to choose for this specific application ?

In this paper, we try to give some hints about these two questions. First, we give an overview of the ideas behind the Machine Learning methodology. Then, we describe some advantages and drawbacks of the method by comparing it to Geostatistics, and give some advice on when it might be useful to apply Machine Learning instead of other modeling methods. Finally, we evaluate the relative efficiency of three algorithms applied to SIC2004 data in a Machine Learning context and conclude on the opportunity to use each of them

[1] The question of the difference between Machine Learning and Artificial Intelligence is yet another source of eternal debate.

for such application.

2 Machine Learning

Machine Learning can be seen as a branch of both statistics and computer science, which tries to develop computing methods with the aim to solve the so-called *learning problem*.

2.1 Learning from Data

As defined by Vapnik [17], the learning problem consists of extracting meaningful information from a finite number of data, called the *training* set. This very general definition contains the underlying idea that data are the central source of available information. Any prior knowledge about the data and the problem at hand is of course useful but not mandatory[2] .

Unfortunately, in general the training set not only contains the target meaningful information, but also contains less interesting information (such as various kinds of noise) which thus needs to be filtered out. A good model should then be able to extract only the *general* information from data, and filter out the specific one.

Vapnik showed [17] that it is possible to control the *generalization performance* of a model (how good it is expected to perform on unseen data, also known as *test* data) by controlling the *capacity* of the family of functions it is constructed from. This capacity roughly corresponds to the number of degrees of freedom of the family of functions (for instance the degree of a polynomial).

2.2 The Bias/Variance Dilemma

The main principle behind capacity control is to be able to solve the famous *bias/variance* dilemma [6]. A family of functions with a high capacity should allow to fit almost any data set, i.e. it will have a low **bias**, defined as the error on the training set, which the learning procedure tries to minimize. Unfortunately, a new model based on a slightly different data set (with some examples that have been changed, but still coming from the same distribution) may end up with a solution quite different from the previous one. This reflects that this family of functions has a high **variance**, corresponding to the number of solutions in the family of functions that are compatible with the training set, and this is also to minimize.

On the other hand, a family of functions with low capacity will behave exactly the converse: it will produce almost always the same model whatever the exact content of the data set, but many of its predictions will be heavily biased.

It is then easy to understand that finding a family of functions with the optimal capacity (the one that simultaneously minimize the bias and the variance terms) will increase the chance to find a model with optimal generalization performance. Several techniques have been proposed in the literature to search for this optimal capacity, the most generic one, also widely used, being any flavor of the general cross-validation algorithm. In addition, some specific algorithms, such as large margin classifiers [17], contain an "automatic tuning" of their capacity during the construction of the model.

3 Machine Learning and Geostatistics

Machine Learning and Geostatistics have many things in common. First, both are methods of statistical data analysis and modeling. They are also mostly "data driven" approaches, as opposed to physical modeling. One of their main differences is that Geostatistics has a very specific scope of application while Machine Learning is much more general.

During the last decade, many people tried, with more or less success, to use so called "Machine Learning algorithms" to model spatially distributed data [4]. The many contributions of SIC2004 applying such algorithms show that there is still a great interest in this domain. However, if Machine Learning has some advantages over Geostatistics, it may also have some drawbacks.

3.1 What are the Differences ?

Machine Learning and Geostatistics have many differences, but some have very strong consequences on the usefulness of the methods in various situations.

Above all, there is a great conceptual difference in the way of interpreting data. In Machine Learning, the data samples are assumed to be independently and identically drawn from a single unknown random process. The training procedure tries to model, directly or indirectly, this distribution. In Geostatistics, each

[2]Apart from some generic *prior* on the problem, such as *smoothness*, which assumes that a small perturbation on the input yields only a small perturbation on the corresponding output.

data sample is considered as a separate random variable, often assumed to follow a Gaussian distribution, which parameters depend on the rest of the data samples, following some unknown correlation parameters. A great confidence is given to the training data[3] , which are used to find these correlation parameters.

The other big difference, as described before, is that Geostatistics has been developed specifically for spatial data analysis, while Machine Learning is a much more general method, often dealing with thousands of input dimensions, and up to millions of examples, with most of the research focused on classification tasks while Geostatistics is more interested in the estimation of continuous data.

These differences have various consequences on the usage that can be done of Machine Learning algorithms for spatial data analysis.

3.2 The Drawbacks of the Machine Learning Approach

For the "new-comer", the most disturbing aspect of a model created using a Machine Learning approach is that its parameters are often very difficult (or even sometimes impossible) to interpret physically. Machine Learning has been designed to retrieve information, not to interpret it. It is however sometimes possible to produce data using a good Machine Learning model and to fit a physical model on them.

A consequence of the data concept of Machine Learning is that there is no particular confidence about training measures. The main drawback is that the training procedure needs generally more data than a Geostatistical approach would. There exist methods which try to reduce the number of data needed [11] by maximizing the information contained in the training set, but they are only applicable in situations where modeling and sampling are closely related. Another important way to reduce the need for data is to constrain the family of functions with more prior knowledge about the problem to solve, as it is actually done in Geostatistics for spatial data.

Some other consequences of this are a strong smoothing effect in regression applications, and a total lack of robustness when trying to predict data drawn from a phenomenon with only a few or no examples in the training data.

3.3 The Advantages of the Learning Approach

There are three main advantages to Machine Learning. First, it is a very generic approach and can be applied on any kind of data (keeping in mind the limitations described earlier), and to a very large panel of algorithms.

The second advantage of Machine Learning models is their robustness to noise. This is a direct consequence of the data concept and the real strength of the method.

Last but not least, Machine Learning has no constraint on data behavior, apart from the training set to contain the requested information and the expected function to being "smooth". In particular, there is no hypothesis on the shape of the data distribution, which is a very nice property for risk mapping.

3.4 Summary: When to Use Machine Learning ?

Given the various advantages and drawbacks of Machine Learning, it is better not to use it blindly in all situations.

As presented qualitatively in Figure 1, Machine Learning should not be used with small datasets (i.e. less than 100), and would probably not give better results than other statistical methods with medium datasets (a few hundred data) unless some prior knowledge is available. However, with large datasets (a few thousands and more), Machine Learning is probably the best approach whatever the prior knowledge available.

[3]This is the famous "respect of data" principle.

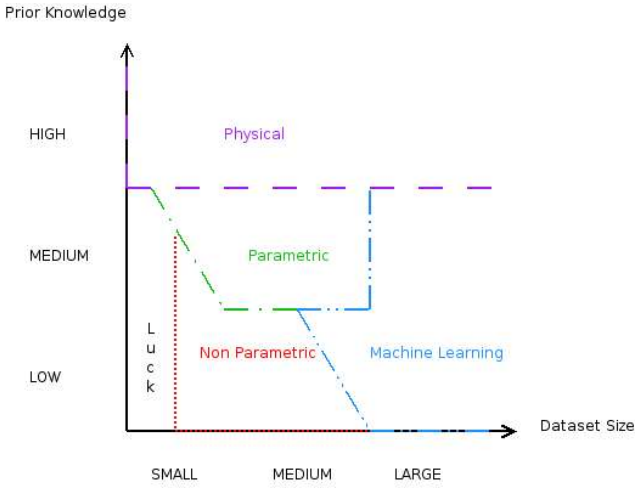


Figure 1: Qualitative representation of the optimal usage of various modeling methods with respect to the size of the data set and the prior knowledge about the studied phenomenon.

At the level of applications, Machine Learning is a good approach for all what concerns probabilities (e.g. risk mapping). On such problems, and still given a sufficient number of data, the absence of *a priori* on data distribution allows dedicated Machine Learning algorithms to be very reliable [8] [7].

Another domain where Machine Learning is a good choice is multi-variate analysis. Dimensionality reduction techniques (such as principal component analysis) have benefited a lot from Machine Learning research, allowing complex non-linear transformations of the input space [15].

Finally, another application of interest for environmental mapping, and where Machine Learning research is very active, is time series prediction [1].

Machine Learning is thus applicable in many fields of interest for environmental mapping, provided enough data are available. However, as we will see in the next section, the fact that Machine Learning can be applied does not mean that *any* Machine Learning algorithm is suited for the job.

4 Comparison of Three Machine Learning Algorithms

Even when Machine Learning can be applied, it is very important to use an algorithm that is suited for the application. In this section, using SIC2004 data, we apply Machine Learning methodology to construct models from three different algorithms, and compare their results to Ordinary Kriging.

The three algorithms chosen for this comparison are well known and often used in Machine Learning. Some contributions to SIC97 [4] and SIC2004 used them in different flavors. In the present paper, we only used their simplest versions.

4.1 K-Nearest Neighbors

It might sound odd to classify K-Nearest Neighbors (KNN) as a Machine Learning algorithm. Actually, it is the way the capacity parameter k is chosen that allows this label.

The principle of KNN is very simple: Let X be a set of l input data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l\}$, $\boldsymbol{x} \in \mathbb{R}^n$ and Y be their corresponding output values, $\{y_1, \ldots, y_l\}$, $y \in \mathbb{R}$. Let $X_{\boldsymbol{x}}$ and $Y_{\boldsymbol{x}}$ be sorted list versions of X and Y, according to their Euclidean distance, in the X space, to $\boldsymbol{x}$. Thus, let $X^i_{\boldsymbol{x}}$ be the i^{th} nearest component of X to $\boldsymbol{x}$, and $Y^i_{\boldsymbol{x}}$ be its corresponding output. Given some $\boldsymbol{x}$, the corresponding estimated output is then given by:

$$\hat{y} = \frac{1}{k} \sum_{i=1}^{k} Y^i_{\boldsymbol{x}}$$

The problem is thus to find the value of k which minimizes the error $(y - \hat{y})^2$ on some unseen data. Hence, k is chosen by cross-validation.

4.2 Support Vector Regression

Support Vector Machines (SVM) [17] for classification problems were developed a decade ago. Later, the algorithm was extended to the regression case [16] and named Support Vector Regression (SVR).

For a given set of data $(\boldsymbol{x_i}, y_i)_{1 \leqslant i \leqslant l}$, $\boldsymbol{x} \in \mathbb{R}^n$ and $y \in \mathbb{R}$, the simplest linear SVR algorithm tries to find the function

$$f(\boldsymbol{x}) = w \cdot \boldsymbol{x} + b$$

minimizing the quadratic optimization problem

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} Q(y_i - f(\boldsymbol{x_i}))$$

where *C* controls the trade-off between optimizing the criterion *Q* and the capacity of the resulting function. In general, $Q(\boldsymbol{x}) = \max\{0, |x| - \epsilon\}$, which corresponds to Vapnik's ε-insensitive loss function, and does not penalize errors less than $\epsilon \geq 0$. After some reformulation and taking into account the case of non-linear regression, the optimization problem is then transformed into the minimization of

$$\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\boldsymbol{x_i}, \boldsymbol{x_j}) + \epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{l} y_i(\alpha_i - \alpha_i^*)$$

subject to

$$\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0$$
$$0 \leq \alpha_i, \alpha_i^* \leq C, \text{ for } 1 \leq i \leq l$$

where the α_i, α_i^* are Lagrange multipliers, solutions of the optimization problem, *C* is the *soft margin* parameter, weighting the influence of the loss function against the regularization term, and $k(\boldsymbol{x_i}, \boldsymbol{x_j})$ is a *kernel function*, defining the feature space in which the optimal solution of the problem will be computed in order to handle non-linear problems. In our experiments, we used the popular Gaussian Radial Basis Function (RBF) kernel:

$$k(\boldsymbol{x_1}, \boldsymbol{x_2}) = \exp\left(-\frac{\|\boldsymbol{x_1} - \boldsymbol{x_2}\|^2}{2\sigma^2}\right)$$

To estimate a new point, we then use the following function f:

$$f(\boldsymbol{x}) = \sum_{i=1}^{N}(\alpha_{s_i} - \alpha_{s_i}^*)k(\boldsymbol{x}, \boldsymbol{x_{s_i}}) + b$$

where the $s_i, 1 \leq i \leq N$ are the indices of the data points for which either α_{s_i} or $\alpha_{s_i}^*$ is non zero. Those points are called *support vectors*.

In order to control the capacity of SVRs, one can tune the following hyper-parameters: *C*, the trade-off between large capacity and small error, ϵ the error we are ready to accept for each estimate, and σ, the bandwidth of the RBF kernel.

4.3 Multilayer Perceptron

A multilayer perceptron (MLP) is a particular architecture of artificial neural networks, composed of layers of non-linear but differentiable parametric functions. An MLP for regression can be written mathematically as follows:

$$f(\boldsymbol{x}; \boldsymbol{\theta}) = b + \sum_{n=1}^{N} w_n \cdot \tanh\left(b_n + \sum_{m=1}^{M} x_m \cdot w_{nm}\right)$$

where the estimated output $f(\boldsymbol{x};\boldsymbol{\theta})$ is a function of the input vector $\boldsymbol{x}$ (indexed by its *M* values x_m), and the parameters $\{\boldsymbol{\theta}: w_n, w_{nm}, b_n, b; \text{ with } n \in [1, N], m \in [1, M]\}$. This MLP is thus a weighted combination of N hyperbolic tangents[4] of weighted combinations of the input vector. Given a criterion *Q* to minimize, such as the mean squared error,

[4]Other non-linear but differentiable functions could replace hyperbolic tangents, including the so-called $sigmoid(x) = 1/(1 + \exp(-x))$.

$$Q = \sum_{i=1}^{l} (y_i - f(\boldsymbol{x_i}; \boldsymbol{\theta}))^2$$

between the desired output y_i and the estimated output $f(\boldsymbol{x_i}; \boldsymbol{\theta})$, for a given training set of size *l*, one can minimize such criterion using a gradient descent algorithm [14]. This algorithm is based on the computation of the partial derivative of the criterion *Q* with respect to all the parameters $\boldsymbol{\theta}$ of $f(\boldsymbol{x};\boldsymbol{\theta})$. Gradient descent can then be performed using

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \lambda \cdot \frac{\partial Q}{\partial \boldsymbol{\theta}}$$

for each parameter $\boldsymbol{\theta}$ where λ is the *learning rate*. It has been shown that given a number of hyperbolic tangents *N* sufficiently large, one can approximate any continuous function using such MLPs [9].

The capacity of MLPs can be tuned in several ways, including the number of hidden units *M*, the learning rate λ, the number of training iterations, and the *weight decay* γ, a factor that can added to the criterion in order to push the parameters $\boldsymbol{\theta}$ to 0 in order to linearize the resulting function[5] .

4.4 Application to SIC2004

4.4.1 The Datasets

We used the SIC2004 data sets to evaluate the adequacy of these algorithms to environmental mapping application. Two problems are to be solved. The first one is called the "Natural" one: data are real measurements of daily gamma radioactivity measured all across Germany. The second is called the "Joker": it is a similar set of measurements as in the Natural set to which a simulated radioactive accident was added.

For each of these two data sets, two case studies have been created. This will allow to evaluate the influence of the training set size on Machine Learning algorithms performances:

- the "Small" case study: 80 measurements are available to estimate the radioactivity at 808 locations;
- the "Large" case study: the training set contains 808 measurements, and the test set only 200.

For the Large case, we took exactly the same data sets as proposed for SIC2004, but inverted the train and test sets. For the Small case, the 80 points were randomly extracted from the 200 training data of SIC2004. For the Small Joker set, we just made sure that the hot spot data were among the 80 training points.

4.4.2 Experimental Protocol

The context of these experiments is a bit different from the one of the SIC2004 competition as we are not using the spare data sets provided as prior information. For the rest, we tried to stay in the spirit of "automatic mapping", human intervention being limited to choosing the variation ranges of the hyper-parameters of the algorithms.

In practice, this means that the choice of the optimal hyper-parameters was computed via cross validation over the predefined list of values. In the case of Ordinary Kriging (OK), an omni directional variogram model was constructed by least square fitting of a spherical model.

The quality of the models is evaluated using the Root Mean Squared Error (RMSE), the Mean Absolute Error (MAE), the Mean Error (ME), and the correlation coefficient of predictions versus real values (R). Furthermore, using the fact that the Mean Squared Error (MSE) and the MAE correspond to averages of some real values, we used a statistical test, the *Student's t-test*, in order to assess the statistical significance of the difference between all pairs of results, with a confidence of 95%.

In addition, as we are dealing with environmental mapping and a simulated emergency situation, we expect the models to be able to detect the "Joker spot" and *locate* it correctly, i.e. accurately identify an area of high radioactivity levels. This is why a qualitative observation of the output maps is also part of the quality evaluation process.

4.4.3 The OK Models

As explained before, the OK models are based only on multi-directional spherical variograms models which parameters are presented in Table 1. The fitting was straight-forward in the case of the Natural sets, but for the Joker, it was necessary to remove the outliers from the Joker spot in order to be able to construct the variogram model. The outliers were chosen based on the fact they were lying very far from the main data

[5]The new criterion then becomes $Q = \sum_{i=1}^{l} (y_i - f(\boldsymbol{x_i}; \boldsymbol{\theta}))^2 + \gamma \sum_j \boldsymbol{\theta}_j^2$.

distribution. Such a procedure is easy to automatize. This ended up with 2 points removed from the Small set and 6 from the Large set. Of course, these points were put back into the dataset for the estimation.

OK	Nugget	Sill	Range
Natural & Small	41.49	377.8	545926
Natural & Large	65.73	474.0	461211
Joker & Small	117.8	469.4	412513
Joker & Large	0.0	1128.6	52653

Table 1: Parameters of the spherical variogram models for the different data sets.

4.4.4 The KNN Models

The number of neighbors was chosen between 1 and 20. The optimal value was obtain by cross-validation over the training data and are presented in Table 2.

KNN	*K*
Natural & Small	4
Natural & Large	8
Joker & Small	11
Joker & Large	3

Table 2: Hyper-parameters of the KNN models for the different data sets.

4.4.5 The SVR Models

The three hyper-parameters of the SVR models were chosen by cross-validation. The input and output data of each data set were normalized, with zero mean and unit standard deviation, estimated on the training sets. The range of values for the hyper-parameters were: $C \in \{1, \ldots, 1000\}$, $\sigma \in [0.1, 2.0]$, $\epsilon \in [0.01, 1.0]$. The optimal values are presented in Table 3.

SVR	C	s	ϵ
Natural & Small	1	0.5	0.03
Natural & Large	1	0.3	0.03
Joker & Small	10	1.1	0.5
Joker & Large	10	0.1	0.01

Table 3: Hyper-parameters of the SVR models for the different data sets.

4.4.6 The MLP Models

The three hyper-parameters of the MLP models were chosen by cross-validation. The input and output data of each data set were normalized, with zero mean and unit standard deviation, estimated on the training sets. The range of values for the hyper-parameters were: number of hidden units (nhu) $\in [1,40]$, learning rate $(\lambda) \in [10^{-5}, 10^{-3}]$, weight decay $(\gamma) \in [0, 0.001]$. The optimal values are presented in Table 4.

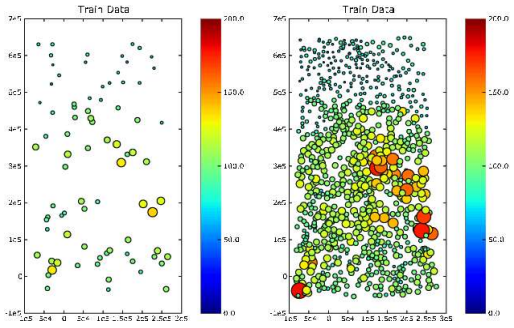
MLP	nhu	λ	γ
Natural & Small	3	0.001	0.001
Natural & Large	35	0.001	0.0
Joker & Small	35	0.001	0.0
Joker & Large	30	0.001	0.001

Table 4: Parameters of the MLP models for the different data sets.

4.5 Analysis of the Results

4.5.1 The Natural Datasets

Figure 2 shows the two training sets for the Natural data. Both sets are relatively smooth with small variance, as can be observed in Table 5. The highest values are located in the East and South-West of the area, while the smallest are located in the North.



(a) (b)

Figure 2: The Natural Small (left) and Large (right) training data sets. Colors are proportional to radioactivity level at the central location. Point sizes are proportional to the fourth power of the radioactivity level.

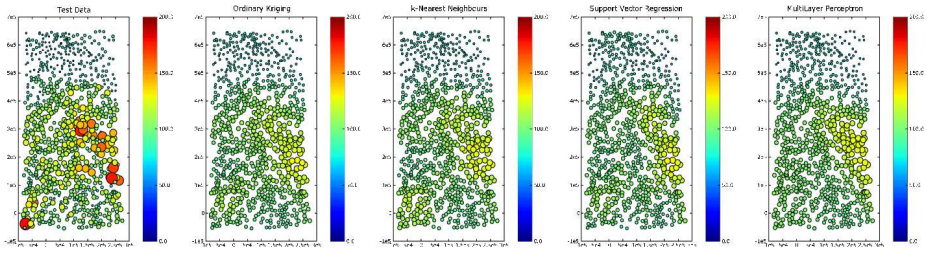
Natural	Min	Median	Max	Mean	StdDev	Skewness	Kurtosis
Small	66.0	96.3	138.0	95.4	17.1	0.3	-0.5
Large	57.0	98.8	180.0	98.0	20.0	0.5	0.7

Table 5: Natural training data sets summary statistics.

The data sampling of the Small dataset is not very uniform, especially in the Center-South and Center-East. However, this should not influence too much the prediction results given the smoothness of data.

Results on the Small Set:

Figure 3 presents the prediction maps of the 4 algorithms (Figures 3b to 3d) with respect to the real data (figure 3a). Visually, it is almost impossible to distinguish the 4 prediction maps from one another. It seems that all the algorithms managed to extract almost exactly the same information from the training set. They are all unable to reproduce exactly the high values from the East and South-West, but the general trend is very well reproduced.



(a) (b) (c) (d) (e)

Figure 3: Small Natural data set estimation maps. Figure a) is the real map. The others are the estimation from: b) Ordinary Kriging, c) K-Nearest-Neighbors, d) Support Vector Regression, and e) Multilayer Perceptron. Color and size scales are the same as in figure 2.

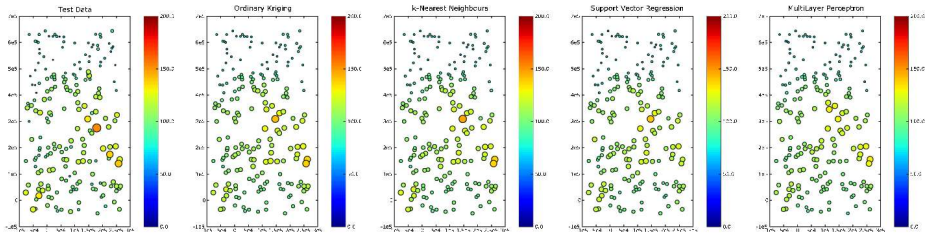
Natural & Small	OK	KNN	SVR	MLP
RMSE	12.82	13.16	14.36	13.26
MAE	9.41	9.68	10.78	10.09
ME	1.52	1.25	2.63	1.5
R	0.77	0.76	0.71	0.75

Table 6: Small Natural data set estimation errors.

The prediction errors given in Table 6 lead to exactly the same conclusion: all the 4 models are performing equivalently, with a RMSE significantly smaller than the standard deviation of the data set. The relatively bad correlation coefficient is the sign that some high values were not correctly estimated. These error are probably the highest contribution to the RMSE as well. In any case, the Student test did not show any significant difference between the performances.

Results on the Large Set:

The prediction maps of figure 4 look again very similar, although OK and KNN seem to give slightly better predictions in the Eastern area than SVR and MLP.



(a) (b) (c) (d) (e)

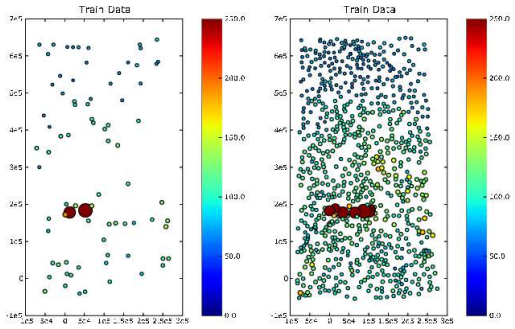
Figure 4: Large Natural data set estimation maps. Figure a) is the real map. The others are the estimation from: b) Ordinary Kriging, c) K-Nearest Neighbors, d) Support Vector Regression, and e) Multilayer Perceptron. Color and size scales are the same as in Figure 2.

Natural & Large	OK	KNN	SVR	MLP
RMSE	10.11	10.07	10.32	10.33
MAE	7.65	7.64	7.63	7.81
ME	-1.75	-1.70	-0.9	-0.63
R	0.83	0.83	0.82	0.81

Table 7: Large Natural data set estimation errors.

Table 7 confirms this similarity between the predictions. The Student test also confirms that there is no significant difference between the models performances.

4.5.2 The Joker Datasets



(a) (b)

Figure 5: The Joker Small (left) and Large (right) training data sets. Point sizes and colors are proportional to radioactivity level at the central location.

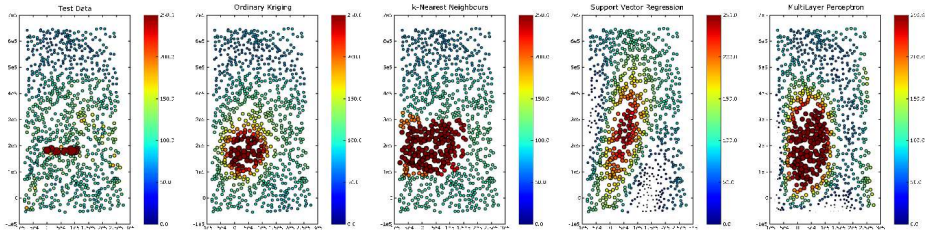
Joker	Min	Median	Max	Mean	StdDev	Skewness	Kurtosis
Small	58.2	99.5	1499.0	127.0	190.9	6.4	41.5
Large	57.0	99.0	1528.2	105.4	83.7	11.5	153.8

Table 8: Joker training data sets summary statistics.

The Joker data sets look far more complex than the Natural one. Basically, we have a very large flat area, with a very tiny hot spot. More precisely, it is a "hot trail", extending in a West-East direction, as we can see on figure 5b. As a consequence, the general statistics of the data sets are strongly perturbed (cf. table 8).

Results on the Small Set:

Unlike the results for the Natural data set, Figure 6 shows very different prediction maps depending on the algorithm involved. Apart for SVR, all the other models manage to locate correctly the center of the hot spot, OK giving the tightest one. However, none is able to actually reproduce the real shape of the phenomenon, due to a lack of information. Interestingly, MLP and SVR show that they are able to build anisotropic models. Unfortunately, they completely missed the West-East shape of the hot spot.



(a) (b) (c) (d) (e)

Figure 6: Small Joker data set estimation maps. Figure a) is the real map. The others are the estimation from: b) Ordinary Kriging, c) K-Nearest Neighbors, d) Support Vector Regression, and e) Multilayer Perceptron. Color and size scales are the same as in Figure 5.

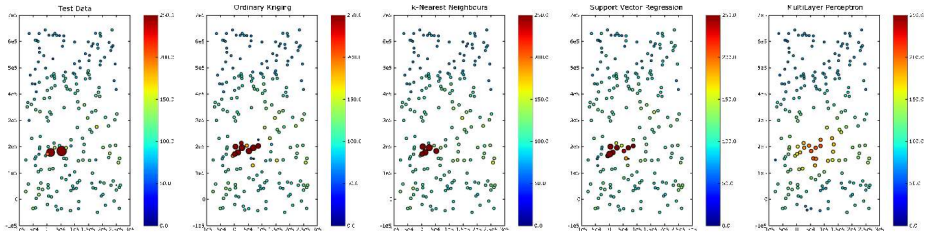
Joker & Small	OK	KNN	SVR	MLP
RMSE	82.60	109.72	91.32	114.64
MAE	33.20	52.99	49.96	65.89
ME	-14.84	-34.63	-3.85	-27.12
R	0.43	0.27	0.18	0.27

Table 9: Small Joker data set estimation errors.

Table 9 gives more details on the prediction values themselves. KNN and MLP are the worst predictors on this case study, while OK is significantly better than the rest. SVR is in between, but its very small

correlation coefficient is the sign that something has going really wrong in the prediction of the hot spot.

Results on the Large Set:



(a) (b) (c) (d) (e)

Figure 7: Large Joker data set estimation maps. Figure a) is the real map. The others are the estimation from: b) Ordinary Kriging, c) K-Nearest Neighbors, d) Support Vector Regression, and e) Multilayer Perceptron. Color and size scales are the same as in Figure 5.

Figure 7 presents the prediction maps for the Large Joker set. The performance of KNN and OK and SVR in localizing the hot spot are very good. MLP manages to locate the hot spot correctly, but its predictions are too smooth.

Joker & Large	OK	KNN	SVR	MLP
RMSE	97.79	96.74	113.69	114.14
MAE	27.16	22.12	28.93	28.29
ME	-3.02	0.27	-1.27	5.61
R	0.60	0.61	0.38	0.36

Table 10: Large Joker data set estimation errors.

In the prediction errors of Table 10, OK and KNN are performing equivalently, a little bit better than SVR and MLP. SVR high RMSE is a bit astonishing given its apparently good performance in localizing the hot spot. However, a very careful look at the data shows that it actually predict a little bit off target, which is enough to produce very high errors and destroy the correlation coefficient. Performances of MLP are easier to understand given the strong underestimation of the hot spot values.

4.5.3 Discussion on the Experiments

The introduction of the Joker set in SIC2004 was a very good idea. As the experiments on the Natural sets showed, it is almost impossible to distinguish prediction algorithms dealing with smooth data, even with a small data set. However, when an event such as the one simulated by the Joker arises, algorithms are under stress and some discrepancy appears.

First of all, Ordinary Kriging is very well adapted to such applications. This is of course not a scoop, but it is still interesting to notice for non Geo-statisticians. In the context of the Joker, the hot spot was small enough with respect to the studied area to allow its exclusion of the variogram modeling process. This might not always be the case however, and if no variogram is available, then OK is useless. Still, OK was very fast to compute (given the small number of data), and very efficient, both in terms of prediction error and of localization of the hot spot.

For what concerns Machine Learning, we have shown that it can be used for such applications, provided the data set size is somehow proportional to the complexity of the problem. On the smooth Natural data sets, only a few data were necessary to extract some relevant information. On the other hand, Machine Learning did not handle very well the Small Joker set, where the various algorithms were all giving too much importance to the hot spot extension. In such context, the lack of information from data is clearly to blame, while the prior information about the importance of spatial correlation is giving a great advantage to OK.

KNN gave very good performance despite (or thanks to) its simplicity. The poor results it gave on the Small Joker were clearly due to the size of the data set, as its good performance on the Large Joker attests. Most of all, it was never outperformed by SVR or MLP, while being a lot faster to train and use.

SVR relative performances were good on the Natural sets, but much worse on the Small Joker where it had some difficulties to locate the hot spot. On the Large Joker, its performances were a bit worse than KNN and OK, but the localization of the hot spot was good, showing that with enough data, this algorithm is able to handle complex phenomena. Its main drawback of the algorithm is the search for the optimal hyper parameters, which is very slow and thus difficult to do in a time critical situation.

Finally, MLP showed also its dependency to the data set size, but it showed its strong smoothing effect as well. This was particularly obvious for the Large Joker where localization was good, but the hot spot values were strongly under estimated. Algorithms such as MLP are more at ease on noisy datasets or large scale tendencies. Unfortunately, the context of the Joker data set is exactly the reverse. In addition, MLP has the strong drawback of being initialized randomly. The consequence is that two models trained on the same data, with the same hyper-parameters can give different results. This is a cause of high uncertainty on the optimality of the "best" hyper-parameters found.

As a conclusion, we can say that SVR and MLP are probably not the best choice for the type of application proposed for SIC2004. If the predictions on smooth data set were good, they were not better than the ones from OK or KNN, which are much simpler and faster to tune. And when a situation like the Joker arises, even given enough data, SVR was still not better than KNN, and MLP predictions were too smooth.

5 Conclusion

In this paper, we presented an overview of Machine Learning, making a strong distinction between the method itself and the applications and algorithms it can be applied on.

By comparing it to Geostatistics, we have shown that it was based on a very different philosophy, which is the source of many interesting properties, but also of some non negligible drawbacks. We have then proposed a qualitative representation of when Machine Learning should be preferred to other methods, with respect to the quantity and type of information available.

In the experiments presented in this paper, we rose the issue of applying “classical” Machine Learning algorithms on problems they were not designed to solve. Hence, we have shown that algorithms such as MLP and SVR were usually not a good choice for environmental mapping applications such as the one proposed for SIC2004. Simpler and faster algorithms such as KNN can give similar or better results for the same amount of data. And if the available measurements allow the construction of a variogram, OK is often much more reliable, whatever the number of data available.

Of course, this does not mean that Machine Learning has nothing to offer to the Environmental Mapping community. Time series analysis and risk mapping are examples of applications where the advantages of Machine Learning can have a significant impact on the quality of the results.

Acknowledgment

The authors thank Grégoire Dubois and Stefano Galmarini for the organization of SIC2004 and their invitation to participate. Ordinary Kriging and variogram modeling were performed using Vesper [12]. The Machine Learning algorithms were constructed using the Torch library [2]. The pictures were produced using the Matplotlib package [10].

References

[1] A. Aussem, F. Murtagh, and M. Sarazin. Dynamical recurrent neural networks: towards environmental time series prediction. *International Journal of Neural Systems*, 6 2:145–170, 1995.

[2] R. Collobert, S. Bengio, and J. Mariéthoz. Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP, 2002.

[3] G. Dubois and S. Galmarini. *Spatial Interpolation Comparison 2004*. To be published, 2005.

[4] G. Dubois, J. Malczewski, and M. DeCort. *Mapping radioactivity in the environment - Spatial Interpolation Comparison 97*. Office for Official Publications of the European Communities, Luxembourg, 2003.

[5] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, 1973.

[6] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

[7] N. Gilardi, S. Bengio, and M. Kanevski. Conditional Gaussian mixture models for environmental risk mapping. *Neural Network for Signal Processing*, 2002.

[8] N. Gilardi and T. Melluish. Confidence evaluation for risk prediction. *Conference of the International Association for Mathematical Geology*, 2001.

[9] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[10] John Hunter. Matplotlib, http://matplotlib.sourceforge.net, 2005.

[11] A. Krogh and J. Vedelsby. Neural networks ensembles, cross validation, and active learning. *Advances in Neural Information Processing Systems*, 7, 1995.

[12] B. Minasny, A. B. McBratney, and B. M. Whelan. Vesper version 1.62. Australian Centre for Precision Agriculture, McMillan Building A05, The University of Sydney, NSW 2006, http://www.usyd.edu.au/su/agric/acpa, 2005.

[13] Laurence R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing*, volume 1. MIT Press, Cambridge, MA., 1986.

[15] B. Schoelkopf, A.J. Smola, and K.-R. Mueller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[16] A. J. Smola and B. Schlkopf. A tutorial on support vector regression. Technical Report 30, NeuroCOLT2, October 1998.

[17] V. N. Vapnik. *The nature of statistical learning theory*. Springer, second edition, 1995.