A comparison of three learning methods to predict N_2O fluxes and N leaching

Nathalie Villa-Vialaneix

http://www.nathalievilla.org

Institut de Mathématiques de Toulouse &
IUT de Carcassonne (Université de Perpignan)

with **Marco Follador** and **Adrian Leip**, Climate Change Unit,
European Commission, Ispra, Italy

June 20*th*, 2010

MASHS 2010, Lille

Outline

1. Data and problem
2. Presentation of three learning methods
3. Methodology and results

Outline

1 Data and problem

2 Presentation of three learning methods

3 Methodology and results

Measure of agricultural impact

- **Agriculture** back on the top of agenda of policy makers to fight against the food crisis and to alleviate poverty **but** Intensive farming practices cause a **degradation of the other ecosystem services** essential for human survival;

Measure of agricultural impact

- **Agriculture** back on the top of agenda of policy makers to fight against the food crisis and to alleviate poverty **but** Intensive farming practices cause a **degradation of the other ecosystem services** essential for human survival;
- European Commission is interested in **quantifying the environmental impact of various policies** by means of effective indicators;

Measure of agricultural impact

- **Agriculture** back on the top of agenda of policy makers to fight against the food crisis and to alleviate poverty **but** Intensive farming practices cause a **degradation of the other ecosystem services** essential for human survival;
- European Commission is interested in **quantifying the environmental impact of various policies** by means of effective indicators;
- To that aim, a platform has been developed in the EC-project CCAT (Cross-Compliance Assessment Tool) that **calculated the impact of cross-compliance legislations on various indicators (GHG, soil pollution, economic indicators, ...)**;

Measure of agricultural impact

- **Agriculture** back on the top of agenda of policy makers to fight against the food crisis and to alleviate poverty **but** Intensive farming practices cause a **degradation of the other ecosystem services** essential for human survival;
- European Commission is interested in **quantifying the environmental impact of various policies** by means of effective indicators;
- To that aim, a platform has been developed in the EC-project CCAT (Cross-Compliance Assessment Tool) that **calculated the impact of cross-compliance legislations on various indicators (GHG, soil pollution, economic indicators, ...)**;
- **But** platform needs **simplified models which make fast simulations** because, in this case, the biogeochemical model DNDC could not be used directly.

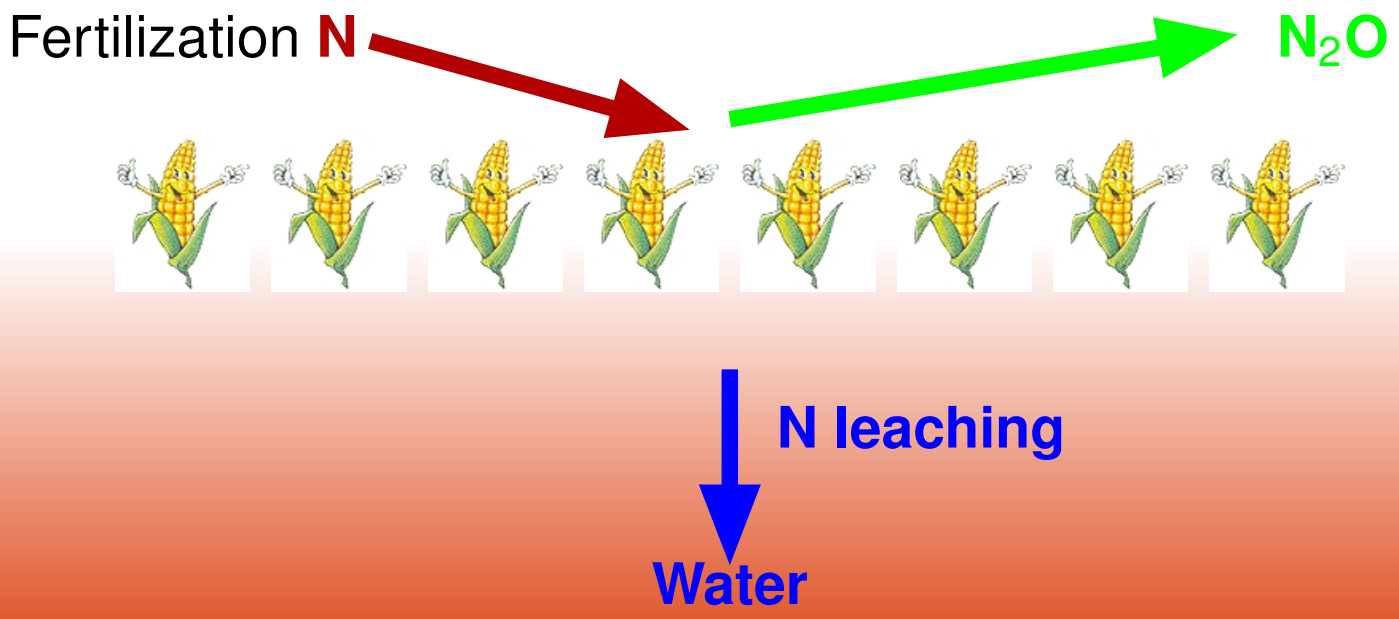
Framework of this study

Selection of HSMU related to **corn culture** (about 20 000 spatial units).
Fertilization is done by adding N which produces N_2O (greenhouse gazes) and N leaching (water pollution).

Fertilization **N**

Framework of this study

Selection of HSMU related to **corn culture** (about 20 000 spatial units).
Fertilization is done by adding N which produces N_2O (greenhouse gazes) and N leaching (water pollution).



Purpose: Estimation of N_20 and N leaching.

Data

5 data sets corresponding to different scenarios coming from the biogeochemical simulator DNDC-EUROPE.

- S1: Baseline scenario (conventional corn cropping system): 18 794 HSMU
- S2: like S1 without tillage: 18 830 HSMU
- S3: like S1 with a limit in N input through manure spreading at 170 kg/ha y: 18 800 HSMU
- S4: rotation between corn (2y) and catch crop (3y): 40 536 HSMU
- S5: like S1 with an additional application of fertilizer in winter: 18 658 HSMU

Data

11 input variables

- N FR (N input through fertilization; kg/ha y);
- N MR (N input through manure spreading; kg/ha y);
- Nfix (N input from biological fixation; kg/ha y);
- Nres (N input from root residue; kg/ha y);
- BD (Bulk Density; g/cm^3);
- SOC (Soil organic carbon in topsoil; mass fraction);
- PH (Soil PH);
- Clay (Ratio of soil clay content);
- Rain (Annual precipitation; mm/y);
- Tmean (Annual mean temperature; C);
- Nr (Concentration of N in rain; ppm).

Data

2 outputs to estimate from the inputs:

- N_2O fluxes;
- N leaching.

Data

2 outputs to estimate from the inputs:

- N_2O fluxes;
- N leaching.

Comparison of 3 methods:

1. **multi-layer perceptrons** (neural networks): [Bishop, 1995]
2. **Support Vector Machines** (SVM): [Boser et al., 1992]
3. **random forests**: [Breiman, 2001]

Outline

1 Data and problem

2 Presentation of three learning methods

3 Methodology and results

Multilayer perceptrons (MLP)

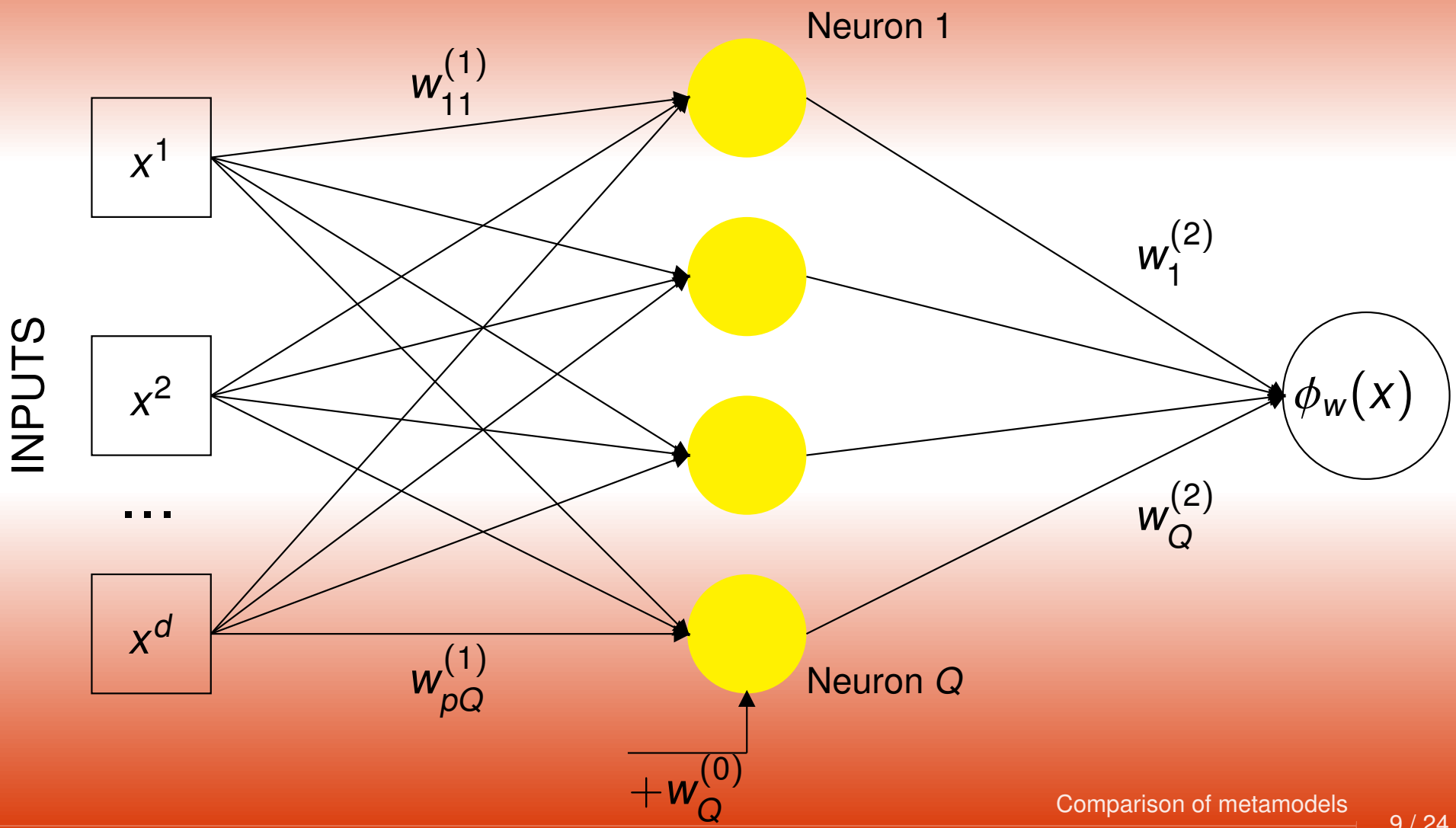
A "one-hidden-layer perceptron" is a function of the form:

$$\Phi_w : x \in \mathbb{R}^d \to \sum_{i=1}^{Q} w_i^{(2)} G\left(x^T w_i^{(1)} + w_i^{(0)}\right) + w_0^{(2)}$$

where:

- the w are the **weights** of the MLP that have to be learned from the learning set;
- G is a given **activation function**: typically, $G(z) = \frac{1-e^{-z}}{1+e^{-z}}$;
- Q is the number of neurons on the hidden layer. It controls the flexibility of the machine. Q is a **hyper-parameter** that is usually tuned during the learning process.

Symbolic representation of MLP



Learning MLP

- **Learning the weights**: w are learned by a **mean squared error minimization** scheme :

$$w^* = \arg\min_w \sum_{i=1}^{N} L(y_i, \Phi_w(x_i)).$$

Learning MLP

- **Learning the weights**: w are learned by a **mean squared error minimization** scheme penalized by a **weight decay** to avoid overfitting (ensure a better generalization ability):

$$w^* = \arg\min_w \sum_{i=1}^{N} L(y_i, \Phi_w(x_i)) + C\|w\|^2.$$

Learning MLP

- **Learning the weights**: w are learned by a **mean squared error minimization** scheme penalized by a **weight decay** to avoid overfitting (ensure a better generalization ability):

$$w^* = \arg\min_w \sum_{i=1}^{N} L(y_i, \Phi_w(x_i)) + C\|w\|^2.$$

Problem: MSE is not quadratic in w and thus the optimization can not be made perfectly right. Some solutions can be local minima.

Learning MLP

- **Learning the weights**: w are learned by a **mean squared error minimization** scheme penalized by a **weight decay** to avoid overfitting (ensure a better generalization ability):

$$w^* = \arg\min_w \sum_{i=1}^{N} L(y_i, \Phi_w(x_i)) + C\|w\|^2.$$

  Problem: MSE is not quadratic in w and thus the optimization can not be made perfectly right. Some solutions can be local minima.

- **Tuning of the hyper-parameters, C and Q**: **simple validation** has been used to tune first C and Q.

SVM

SVM is also a error loss with penalization minimization:

1. **Basic linear SVM for regression**: $\Phi_{(w,b)}$ is of the form $x \to w^T x + b$ with (w, b) solution of

$$\arg\min \sum_{i=1}^{N} L_\epsilon(y_i, \Phi_{(w,b)}(x_i)) + \lambda \|w\|^2$$

where

- λ is a regularization (hyper) parameter (to be tuned during the learning process);
- $L_\epsilon(y, \hat{y}) = \max\{|y - \hat{y}| - \epsilon, 0\}$ is an **ϵ-insensitive loss function**

▸ See ϵ-insensitive loss function

SVM

SVM is also a error loss with penalization minimization:

1. **Basic linear SVM for regression**
2. **Non linear SVM for regression** are the same except that a non linear (fixed) transformation of the inputs is previously made: $\varphi(x) \in \mathcal{H}$ **is used instead of** x**.**

SVM

SVM is also a error loss with penalization minimization:

1. **Basic linear SVM for regression**
2. **Non linear SVM for regression** are the same except that a non linear (fixed) transformation of the inputs is previously made: $\varphi(x) \in \mathcal{H}$ **is used instead of** x**.**
   Kernel trick: in fact, φ is never explicit but used through a **kernel**, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. This kernel is used for $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$.

SVM

SVM is also a error loss with penalization minimization:

1. **Basic linear SVM for regression**
2. **Non linear SVM for regression** are the same except that a non linear (fixed) transformation of the inputs is previously made: $\varphi(x) \in \mathcal{H}$ **is used instead of** x**.**
   Kernel trick: in fact, φ is never explicit but used through a **kernel**, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. This kernel is used for $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$.

Common kernel: Gaussian kernel

$$K_\gamma(u, v) = e^{-\gamma \|u-v\|^2}$$

is known to have good theoretical properties both for accuracy and generalization.

Learning SVM

- **Learning** (w, b): $w = \sum_{i=1}^{N} \alpha_i K(x_i, .)$ and b are calculated by an exact optimization scheme (quadratic programming). The only step that can be time consumming is the calculation of the kernel matrix:

$$K(x_i, x_j) \qquad \text{for } i,\ j = 1, \ldots, N.$$

Learning SVM

- **Learning** (w, b): $w = \sum_{i=1}^{N} \alpha_i K(x_i, .)$ and b are calculated by an exact optimization scheme (quadratic programming). The only step that can be time consumming is the calculation of the kernel matrix:

$$K(x_i, x_j) \qquad \text{for } i,\ j = 1, \ldots, N.$$

The resulting $\hat{\Phi}^N$ is known to be of the form:

$$\hat{\Phi}^N(x) = \sum_{i=1}^{N} \alpha_i K(x_i, x) + b$$

where only a few α_i are non zero. The corresponding x_i are called **support vectors**.

Learning SVM

- **Learning** (w, b): $w = \sum_{i=1}^{N} \alpha_i K(x_i, .)$ and b are calculated by an exact optimization scheme (quadratic programming). The only step that can be time consumming is the calculation of the kernel matrix:

$$K(x_i, x_j) \qquad \text{for } i,\ j = 1, \ldots, N.$$

  The resulting $\hat{\Phi}^N$ is known to be of the form:

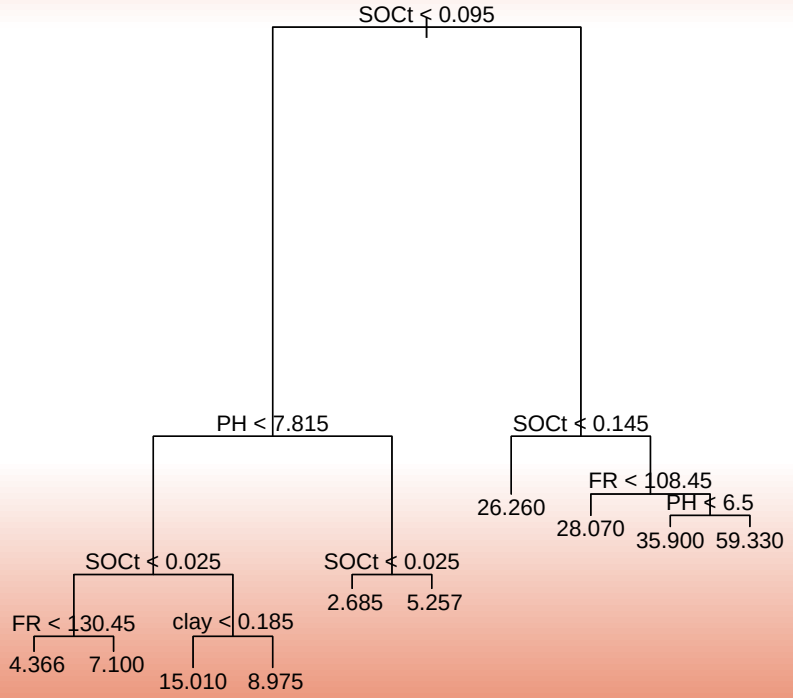
$$\hat{\Phi}^N(x) = \sum_{i=1}^{N} \alpha_i K(x_i, x) + b$$

  where only a few α_i are non zero. The corresponding x_i are called **support vectors**.

- **Tuning of the hyper-parameters,** $C = 1/\lambda$**,** ϵ **and** γ**: simple validation** has been used. To limit waste of time, ϵ has not been tuned in our experiments but set to the default value (1) which ensured $0.5N$ support vectors at most.

From regression tree to random forest

Example of a regression tree



Each **split** is made such that the two induced subsets have the greatest homogeneity possible.

The **prediction of a final node** is the mean of the Y value of the observations belonging to this node.

Random forest

Basic principle: combination of a large number of under-efficient regression trees (the prediction is the mean prediction of all trees).

Random forest

Basic principle: combination of a large number of under-efficient regression trees (the prediction is the mean prediction of all trees). For each tree, **two simplifications** of the original method are performed:

1. A given number of **observations are randomly chosen** among the training set: this subset of the training data set is called in-bag sample whereas the other observations are called out-of-bag and are used to control the error of the tree;
2. For each node of the tree, a given number of **variables are randomly chosen** among all possible explanatory variables.

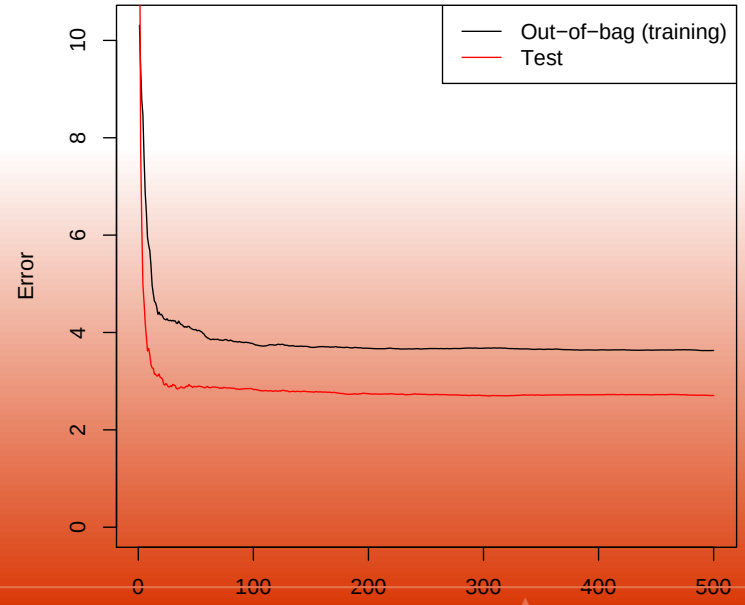
The best split is then calculated on the basis of these variables and of the chosen observations. The chosen observations are the same for a given tree whereas the variables taken into account change for each split.

Learning a random forest

Random forest are **not very sensitive to hyper-parameters** (number of observations for each tree, number of variables for each split): the default values have been used.

Learning a random forest

Random forest are **not very sensitive to hyper-parameters** (number of observations for each tree, number of variables for each split): the default values have been used.
The number of trees should be large enough for the mean squared error based on out-of-sample observations to stabilize:



Outline

1 Data and problem

2 Presentation of three learning methods

3 Methodology and results

Methodology

For **every data set**, **every output** and **every method**,

1. The data set has been **split into a training set and a test set** (on a 80%/20% basis);

Methodology

For **every data set**, **every output** and **every method**,

1. The data set has been **split into a training set and a test set** (on a 80%/20% basis);
2. The machine has been **learned from the training set** (with a full validation process for the hyperparameter tuning);

Methodology

For **every data set**, **every output** and **every method**,

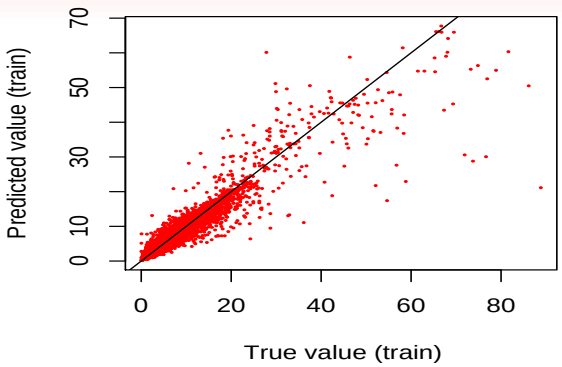
1. The data set has been **split into a training set and a test set** (on a 80%/20% basis);
2. The machine has been **learned from the training set** (with a full validation process for the hyperparameter tuning);
3. The **performances have been calculated** on the basis of the test set: for the test set, predictions have been made from the inputs and compared to the true outputs.

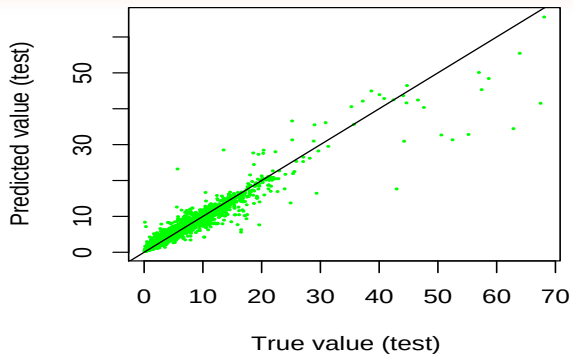
Numerical results

		MLP	RF	SVM	LM
N_2O	Scenario 1	90.7%	**92.3%**	91.0%	77.1%
	Scenario 2	80.3%	**85.0%**	82.3%	62.2%
	Scenario 3	85.1%	**88.0%**	86.7%	74.6%
	Scenario 4	88.6%	**90.5%**	86.3%	78.3%
	Scenario 5	80.6%	**84.9%**	82.7%	42.5%
N leaching	Scenario 1	89.7%	93.5%	**96.6%**	70.3%
	Scenario 2	91.4%	93.1%	**97.0%**	67.7%
	Scenario 3	90.6%	90.4%	**96.0%**	68.6%
	Scenario 4	80.5%	86.9%	**90.6%**	42.5%
	Scenario 5	86.5%	92.9%	**94.5%**	71.1%

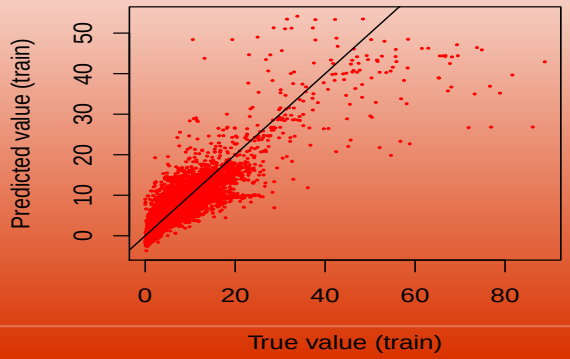
Graphical results

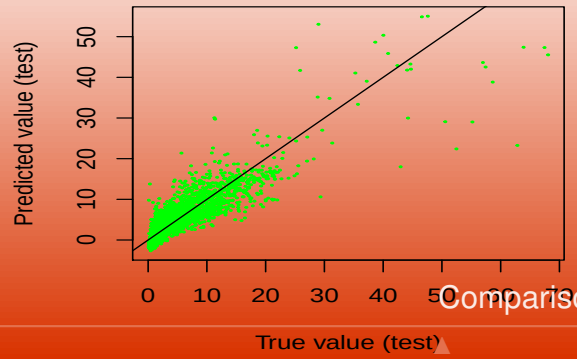
Random forest (S1, N_2O)





Linear model (S1, N_2O)





Further comparisons

	Time for training	Time for using once the model is trained
RF	+ (~ 15 minutes)	= (~ 5 seconds)
SVM	=	- (~ 20 seconds) sensitive to N
MLP	-	+ (~ 1 second) sensitive to d

Times correspond to the prediction for about 20 000 HSMU on a Dell Latitude D830, Intel Core 2DUO 2.2GHz, 2GO RAM, OS Ubuntu Linux 9.10 (and, for RF, to the training of about 15 000 HSMU) with R.

Time for DNDC: around 200 hours by using a desktop computer and around 2 days by using a cluster of computers.

Understanding which inputs are important

Importance: A measure to estimate the importance of the input variables can be defined by:

- for a given input variable randomly permute the input values and calculate the prediction from this new randomly permutated inputs;
- compare the accuracy of these predictions to accuracy of the predictions obtained with the true inputs: the increase of mean squared error is called the importance.

Understanding which inputs are important

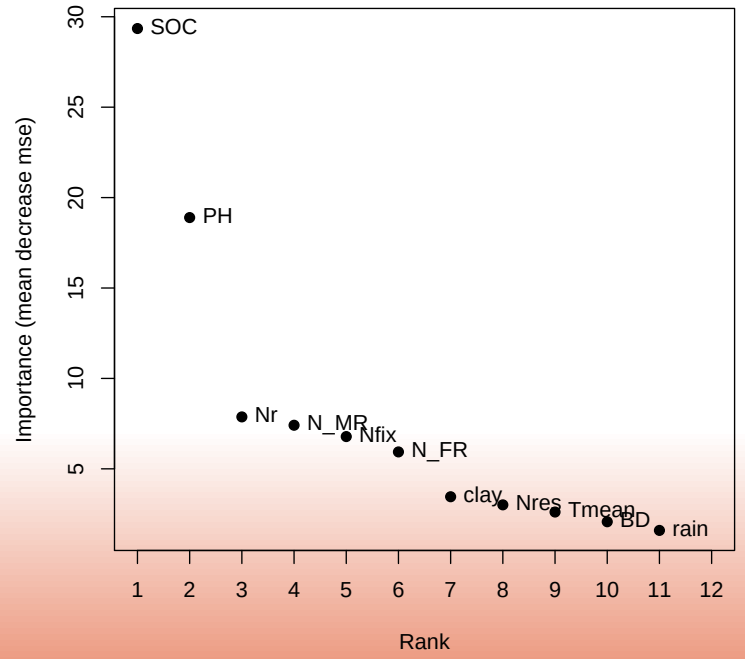
Importance: A measure to estimate the importance of the input variables can be defined by:

- for a given input variable randomly permute the input values and calculate the prediction from this new randomly permutated inputs;
- compare the accuracy of these predictions to accuracy of the predictions obtained with the true inputs: the increase of mean squared error is called the importance.

This comparison is made on the basis of data that are not used to define the machine, either the validation set or the out-of-bag observations.

Understanding which inputs are important

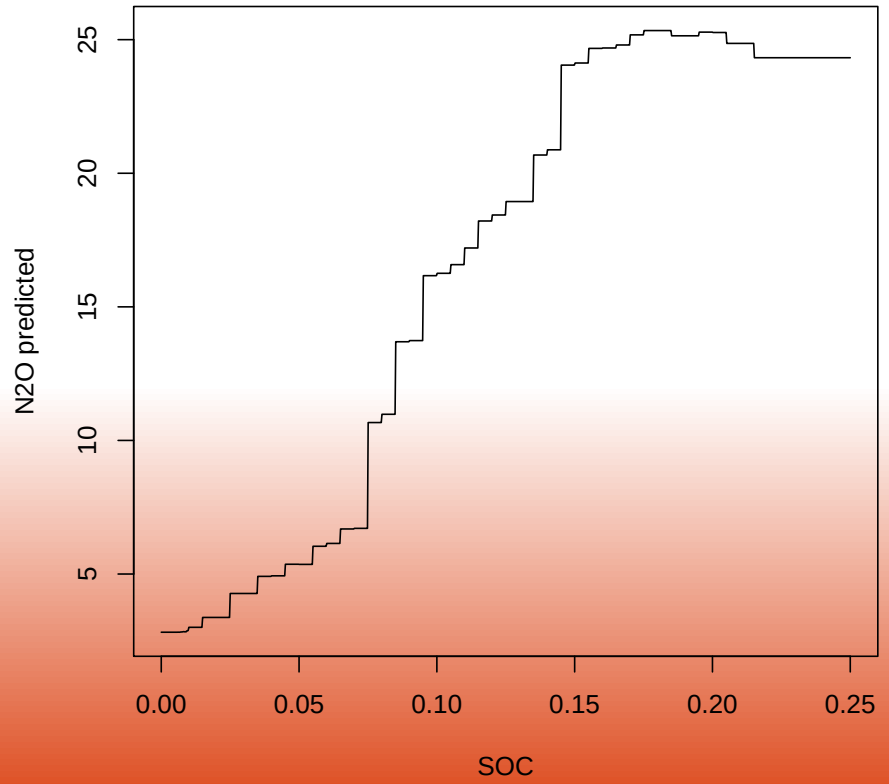
Example (S1, N_2O, RF):



The variables **SOC** and **PH** are the most important for accurate predictions.

Understanding the relation between a given input and the output

Evolution of N_2O in function of the value of SOC



Conclusion and further work

- Learning approaches can provide accurate estimations of the variable of interest much faster than the biogeochemical simulator;
- The random forest approach is now implemented in DNDC to allow fast test of policies impacts.

Conclusion and further work

- Learning approaches can provide accurate estimations of the variable of interest much faster than the biogeochemical simulator;
- The random forest approach is now implemented in DNDC to allow fast test of policies impacts.

- Other statistical nonparametric approaches (based on splines) are currently under test.

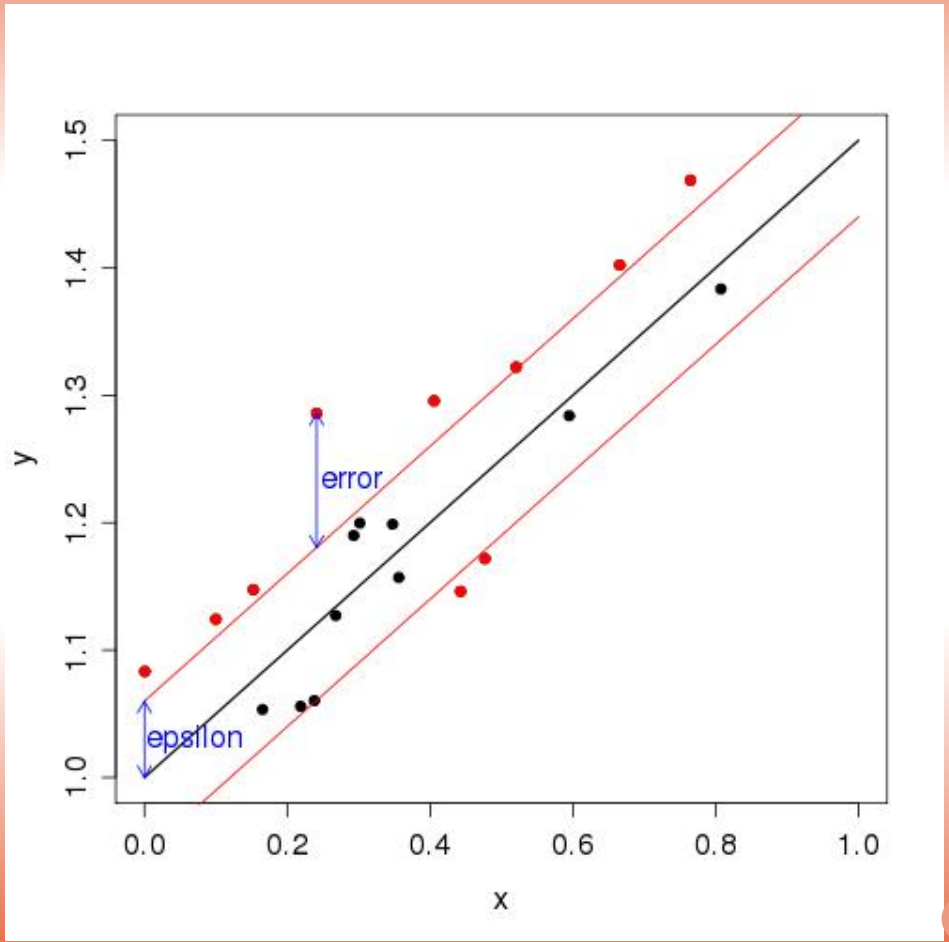
Bishop, C. (1995).
Neural Networks for Pattern Recognition.
Oxford University Press, New York.

Boser, B., Guyon, I., and Vapnik, V. (1992).
A training algorithm for optimal margin classifiers.
In *5th annual ACM Workshop on COLT*, pages 144–152. D. Haussler Editor, ACM Press.

Breiman, L. (2001).
Random forests.
Machine Learning, 45(1):5–32.

Thank you for your attention.
Questions? Comments?

ϵ-insensitive loss function



Go back