DINCON 2011
10ª Conferência Brasileira de Dinâmica, Controle e Aplicações
28 de agosto a 1º de setembro de 2011

APPLYING MACHINE LEARNING ALGORITHMS TO BRIDGE DESIGN

Aloísio Carlos de Pina [1], *Vitor de Souza Colimodio* [2], *Adriano Armani da Silva* [2], *Armando Carlos de Pina Filho* [3]

[1] UFRJ, COPPE, Civil Engineering Program, Rio de Janeiro, RJ, Brazil, e-mail: aloisiopina@bol.com.br
[2] UFRJ, POLI, Department of Civil Engineering, Rio de Janeiro, RJ, Brazil, e-mails: colimodio@poli.ufrj.br, adrianoarmani@poli.ufrj.br
[3] UFRJ, POLI, Urban Engineering Program, Rio de Janeiro, RJ, Brazil, e-mail: armando@poli.ufrj.br

Abstract: Bridge design is a very complex process. The designers need to exercise their judgment concerning many aspects, such as: aesthetics, cost, load determination, failure modes, etc. This work shows the application of several Machine Learning algorithms to the decision-making process of a bridge design, based on specification properties.

Keywords: Bridge Design, Machine Learning, Artificial Neural Networks.

1. INTRODUCTION

A bridge is a structure which permits the connection at the same level of points that are not accessible, separated by rivers, valleys or other natural or artificial obstacle. Bridges are built in order to permit the passage of people, cars, trucks, piping or water (aqueduct) over the obstacle which must be transposed. When a bridge is built over a water way, it is done in a manner that boats can pass under the bridge in a secure way. When it is built over a dry element, it is called viaduct.

In the achievement of a bridge project, many variables are considered before arriving to the final project. We will quote below some of them:

- The bridge construction aim: if the bridge is built with the aim of being only a human passage, or a trucks passage or even an aqueduct;
- The bridge length;
- The bridge broadness;
- What will be the raw materials used in its construction: steel, iron or wood, for example;
- How the bridge recesses will be;
- The bridge kind, such as: cantilever bridge that is built using cantilevers; arch bridge that is arch-shaped and has abutments at each end; suspension bridge that is suspended from cables; simple truss bridge; and continuous truss bridge that extends without joints among 3 or more supports. Some examples are showed in Fig. 1, 2 and 3.

The proposed question is the following: which one is the best type of bridge to be built, knowing the other variables? Yoram Reich developed a system, called *BRIDGER* [1], in order to help designers and architects, especially in the aesthetic way, in a bridge project. This system was developed based on machine learning.

Fig. 1. Howrah Bridge, in Kolkata, India: a cantilever bridge.

Fig. 2. Aqueduct of Segovia, Spain: an arch bridge.

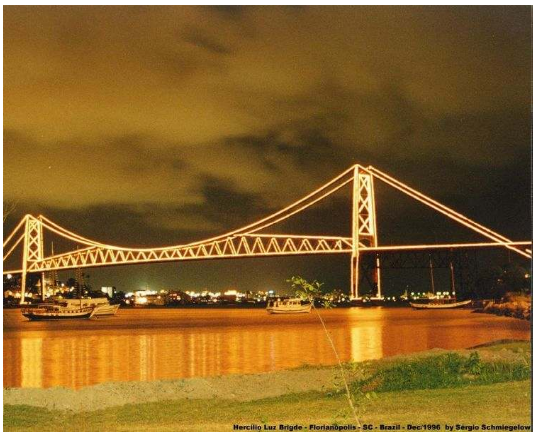

Fig. 3. Hercílio Luz Bridge, Florianopolis, Brazil: a suspension bridge.

Our aim is the search for a solution for this problem, also based on machine learning, by testing a large number of different algorithms and then making a comparison of their results, concluding which of them was the best.

2. MACHINE LEARNING

A machine learning program [2] is a program capable of learning with experience. Until nowadays, it is not known a manner of making a computer learn as well as a person. However, algorithms were developed with this objective and they are efficient on some kinds of learning tasks.

An algorithm learns through examples. An example is a combination of values of variables from the problem. So, in order to build a data set, some steps must be done: definition of the variables; collection of its values experimentally; and build of examples with the collected values.

The following variables must be defined: the target variable that represents the information to be learned; and auxiliary variables whose values can be determined and can help to calculate the value of the target variable.

The variables can be continuous or discrete. If the target variable is continuous, it is called target value and the learning is called regression; if it is discrete, it is called class and the learning is called classification. Thus, each example is a combination of auxiliary variables values to which it is known the class or target value associated.

Continuous values can be used as if they were discrete, passing through a “discretization” process. This process permits: the application of algorithms which only work with discrete variables in problems where there are continuous variables; and the use of classifier to resolve a regression problem, although the results are not always good.

Irrelevant attributes can signify computational waste, or even they can make difficult the learning process. So, they must be identified and removed from the data set before the learning. Some variables are clearly irrelevant and can be identified and removed manually, as for instance a variable that has a different value associated to each example. Nevertheless, some times it is complicated to identify the irrelevant attributes. Noise is another problem which can damage the learning process. It can be caused by wrong measures, atypical phenomena, typing mistakes, etc.

An algorithm receives as input a set of examples that is used in the learning and so it becomes capable of answering the target value associated to any combination of attributes values. A combination to which it is not known the target value associated is called “test-case” and the data set used in the learning is called “training set”.

The answer that is given by the trained program is not always the correct one, depending on the data set and the algorithm which is being used. Thus, part of the examples must be separated in order to evaluate the program’s performance. These examples compose the test set. The process occurs as follows: considering a data set, it is divided in training and test sets; the training set is used to train the algorithm; each test set example is transformed into a test-case which is given to the algorithm; and the answers given by the program are compared to the real target values and then the performance is evaluated.

3. EXPERIMENTAL EVALUATION

In this section, the data set and the experimental settings will be described.

For the experiments, it was used a data set available at the UCI (University of California, Irvine) Machine Learning Repository [3]. The data set was created by Yoram Reich and Steven J. Fenves [4], and it contains design information cataloged from Pittsburgh bridges, USA.

There are two versions of the data set: the original version and the one containing descriptions after discretization of numerical properties. Both data sets consist of 108 examples, each one composed by the values of 12 attributes and the target value. The target variable is the type of bridge to be built and the other attributes determine bridge specifications, as described below:

1. IDENTIF: identifier of the example;
2. RIVER: specifies which river is being crossed (Allegheny, Ihoa, Monongahela or Youghigheny);
3. LOCATION: bridge location;
4. ERECTED: period in time when the bridge was built;
5. PURPOSE: purpose of the construction;
6. LENGTH: total length of the crossing;
7. LANES: number of lanes;
8. CLEAR-G: specifies whether a vertical navigation clearance requirement was enforced in the design or not;
9. T-OR-D: specifies the vertical location of the roadway on the bridge: within the structure (through) or on top of it (deck);
10. MATERIAL: material that the bridge was built;
11. SPAN: length of the main span of the bridge;
12. REL-L: relative length of the main span of the bridge to the total crossing length;
13. TYPE: describes the type of the bridge, which may be one of the following: simple, continuous or cantilever truss; arch; suspension; or wood (target variable).

The possible values for each attribute are presented in Table 1. The Type column states whether a property is continuous/integer (C) or nominal (N). For properties with C,N Type, the range of continuous numbers is given first and possible nominal values follow the semi-colon.

Table 1. Data set attributes and values.

Name	Type	Possible values
1. IDENTIF	--	--
2. RIVER	N	A, M, O, Y
3. LOCATION	N	1 to 52
4. ERECTED	C,N	1818-1986 ; crafts, emerging, mature, modern
5. PURPOSE	N	walk, aqueduct, rr, highway
6. LENGTH	C,N	804-4558 ; short, medium, long
7. LANES	C,N	1, 2, 4, 6 ; 1, 2, 4, 6
8. CLEAR-G	N	N, G
9. T-OR-D	N	through, deck
10. MATERIAL	N	wood, iron, steel
11. SPAN	N	short, medium, long
12. REL-L	N	S, S-F, F
13. TYPE	N	wood, suspen, simple-t, arch, cantilev, cont-t

The attribute IDENTIF was removed before the learning process, since it is obviously an irrelevant attribute, and its presence could damage the learning task.

In order to find out which algorithm provides the best accuracy in the solution of the bridge design problem, we use the machine learning program Weka [5], an open source application that contains various learning algorithms available for test (http://www.cs.waikato.ac.nz/ml/weka/).

Using only a pair of training set and test set does not give a statistically significant performance estimate. If the data set is large, it can be divided into independent pairs of training set and test set. The algorithm's performance is obtained by averaging the performances of all pairs.

If the data set is not large, in general it is used the cross-validation method. Using this method, in order to construct each pair of training set and test set, the data set is divided at random into n sets, all of them with the same size (called folds), $T_1, \ldots, T_n$. So, n rounds are conduced. In each round, T_i is the test set and all T_j, $j \neq i$, compose the training set. The performance is the average of the performances for the n rounds.

Table 2. Paradigms and algorithms.

Paradigm	Algorithm
Bayes learning	*Bayes Net*
	Naive Bayes
	Naive Bayes Simple
	Naive Bayes Updateable
Function learning	*Logistic*
	Multilayer Perceptron
	RBF Network
	Simple Logistic
	SMO
Instance based learning	*IB1*
	IBk
	*K**
	LWL
Rule learning	*Conjunctive Rule*
	Decision Table
	JRip
	NNge
	OneR
	PART
	Ridor
	ZeroR
Tree learning	*Decision Stump*
	J48
	LMT
	NB Tree
	Random Forest
	Random Tree
	REP Tree

If the program adapts too much to the training data (overfitting), there will be a high test error. On the other hand, if the program is not trained enough (underfitting), the performance will also be poor. Between these two extremes, there is a model which gives the minimum test error.

Overfitting can be avoided by using a validation set. Part of the examples is separated with the intention of checking the system's performance during the training. This way, the training stops when there is a performance decline with the validation set.

Twenty eight algorithms were used in this research, from various learning paradigms. Table 2 presents a list of all 28 algorithms and their respective paradigms. Details about these paradigms and algorithms can be seen in [2].

4. RESULTS AND DISCUSSION

Tables 3 and 4 show the predictive accuracy achieved by all 28 algorithms for the original and modified data sets, respectively. One can note that, from the 10 first ranked algorithms for both versions of data set, 9 are the same, changing positions. Tables 5 and 6 contain the average training time of each algorithm for both data sets.

Table 3. Ranking of predictive accuracy for the original data set.

Algorithm	Predictive accuracy	Algorithm	Predictive accuracy
J48	71.4286%	*IBk*	62.8571%
Random Forest	70.4762%	*IB1*	62.8571%
Naive Bayes	70.4762%	*Logistic*	62.8571%
Naive Bayes Updateable	70.4762%	*LWL*	62.8571%
LMT	69.5238%	*RBF Network*	60.9524%
Naive Bayes Simple	69.5238%	*REP Tree*	60.9524%
Multilayer Perceptron	69.5238%	*Decision Table*	60.0000%
Simple Logistic	69.5238%	*JRip*	59.0476%
SMO	67.6190%	*Ridor*	59.0476%
Bayes Net	65.7143%	*Decision Stump*	57.1429%
*K**	64.7619%	*Random Tree*	57.1429%
NB Tree	64.7619%	*Conjunctive Rule*	57.1429%
NNge	63.8095%	*OneR*	55.2381%
PART	62.8571%	*ZeroR*	41.9048%

The results were different for both data set versions. For the original version, the best algorithm was *J48*, the simplest decision tree algorithm. For the discretized version of the data set, 2 algorithms achieved the same predictive accuracy: *LMT* and *Simple Logistic*. In this case, training time was used as decision criterion. Analyzing Table 6, it is possible to verify that *Simple Logistic* is almost 3 times faster than *LMT*, and therefore the former is a better choice.

By comparing the results for both data sets, we see that discretization of numerical attributes is not a good option, since the algorithm *J48* achieved, for the original data set, a higher performance than *Simple Logistic* achieved for the modified data set, and in less average training time. Therefore, we conclude that the best algorithm for this problem is decision tree *J48*, applied to the original data set.

Table 4. Ranking of predictive accuracy for the modified data set.

Algorithm	Predictive accuracy	Algorithm	Predictive accuracy
LMT	69.52%	*JRip*	61.90%
Simple Logistic	69.52%	*PART*	61.90%
NB Tree	68.57%	*REP Tree*	60.95%
SMO	68.57%	*LWL*	60.95%
J48	67.62%	*NNge*	60.00%
Multilayer Perceptron	66.67%	*Random Tree*	60.00%
Naive Bayes	65.71%	*IB1*	59.05%
Naive Bayes Simple	65.71%	*Logistic*	58.10%
Naive Bayes Updateable	65.71%	*Decision Table*	58.10%
Bayes Net	64.76%	*Conjunctive Rule*	57.14%
*K**	63.81%	*Decision Stump*	57.14%
Random Forest	63.81%	*OneR*	53.33%
IBk	62.86%	*Ridor*	49.52%
RBF Network	61.90%	*ZeroR*	41.90%

Table 5. Ranking of training time for the original data set.

Algorithm	Training time (s)	Algorithm	Training time (s)
SMO	6.86	*REP Tree*	0.05
Multilayer Perceptron	6.09	*J48*	0.03
NB Tree	3.69	*Naive Bayes*	0.03
LMT	3.38	*Random Tree*	0.02
Simple Logistic	1.33	*Bayes Net*	0.02
Logistic	1.05	*OneR*	0
RBF Network	0.36	*IBk*	0
PART	0.28	*ZeroR*	0
Random Forest	0.27	*Decision Stump*	0
Ridor	0.17	*K**	0
JRip	0.16	*IB1*	0
Decision Table	0.13	*Naive Bayes Updateable*	0
NNge	0.11	*Naive Bayes Simple*	0
Conjunctive Rule	0.09	*LWL*	0

Table 6. Ranking of training time for the modified data set.

Algorithm	Training time (s)	Algorithm	Training time (s)
Multilayer Perceptron	12.16	*IBk*	0
LMT	6.97	*Random Tree*	0
SMO	5.83	*REP Tree*	0
RBF Network	4.44	*ZeroR*	0
NB Tree	2.70	*OneR*	0
Simple Logistic	2.39	*J48*	0
Logistic	1.25	*Decision Stump*	0
Random Forest	0.09	*K**	0
JRip	0.05	*IB1*	0
Decision Table	0.05	*Naive Bayes Simple*	0
Ridor	0.05	*Naive Bayes*	0
Bayes Net	0.02	*Conjunctive Rule*	0
NNge	0.02	*LWL*	0
PART	0	*Naive Bayes Updateable*	0

5. CONCLUSION

With the use of Weka, we had as result that the best method to obtain the ideal type of bridge for construction is the tree learning algorithm *J48*, using the data set with numeric attributes. Although the predictive accuracy seems to be low (71%), it must be noted that the answer provided by the algorithm should be used as support in the designing process. The low accuracy can be result of underfitting: only 108 examples were available, and only 97 were used for training. By collecting more real examples in other cities, coupled with more detailed attributes, it is possible that one of the algorithms will have a more favorable outcome.

ACKNOWLEDGEMENTS

This research was partially supported by CAPES and FAPERJ.

REFERENCES

[1] Y. Reich, "Converging to 'Ideal' Design Knowledge by Learning", Int. Workshop on Formal Methods in Engineering Design, Colorado, pp. 330-349, 1990.

[2] T. Mitchell, "Machine Learning", McGraw-Hill, 1997.

[3] A. Asuncion, and D. Newman, "UCI Machine Learning Repository", University of California, School of Information and Computer Science, Irvine, USA, 2007.

[4] Y. Reich, and S. J. Fenves, "Inductive learning of synthesis knowledge", Int. Journal of Expert Systems: Research and Applications, v. 5 (4), pp. 275-297, 1992.

[5] I. H. Witten, and E. Frank, "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, 2005.

[6] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning - Data Mining, Inference, and Prediction", Springer Series in Statistics, 2001.