

A REVIEW OF STUDIES ON MACHINE LEARNING TECHNIQUES

Yogesh Singh

Prof & Dean

*Guru Gobind Singh IP University
Delhi, 110006, India*

ys66@rediffmail.com

Pradeep Kumar Bhatia

*Reader, Department of Computer Science & Engineering
Guru Jambheshwar University of Science & Technology
Hisar, Haryana, 125001, India*

pk_bhatia2002@yahoo.com

Omprakash Sangwan

*Head, CISCO Regional Networking Academy
Amity Institute of Information Technology
Amity University, Uttarpradesh, 201303, India*

sangwan_op@aiit.amity.edu

Abstract

This paper provides an extensive review of studies related to expert estimation of software development using Machine-Learning Techniques (MLT). Machine learning in this new era, is demonstrating the promise of producing consistently accurate estimates. Machine learning system effectively “learns” how to estimate from training set of completed projects. The main goal and contribution of the review is to support the research on expert estimation, i.e. to ease other researchers for relevant expert estimation studies using machine-learning techniques. This paper presents the most commonly used machine learning techniques such as neural networks, case based reasoning, classification and regression trees, rule induction, genetic algorithm & genetic programming for expert estimation in the field of software development. In each of our study we found that the results of various machine-learning techniques depends on application areas on which they are applied. Our review of study not only suggests that these techniques are competitive with traditional estimators on one data set, but also illustrate that these methods are sensitive to the data on which they are trained.

Keywords: Machine Learning Techniques (MLT), Neural Networks (NN), Case Based Reasoning (CBR), Classification and Regression Trees (CART), Rule Induction, Genetic Algorithms and Genetic Programming.

1. INTRODUCTION

The poor performance results produced by statistical estimation models have flooded the estimation area for over the last decade. Their inability to handle categorical data, cope with missing data points, spread of data points and most importantly lack of reasoning capabilities has triggered an increase in the number of studies using non-traditional methods like machine learning techniques.

Machine Learning is the study of computational methods for improving performance by mechanizing the acquisition of knowledge from experience [18]. Expert performance requires

much domain specific knowledge, and knowledge engineering has produced hundreds of AI expert systems that are now used regularly in industry. Machine Learning aims to provide increasing levels of automation in the knowledge engineering process, replacing much time-consuming human activity with automatic techniques that improve accuracy or efficiency by discovering and exploiting regularities in training data. The ultimate test of machine learning is its ability to produce systems that are used regularly in industry, education, and elsewhere. Most evaluation in machine learning is experimental in nature, aimed at showing that the learning method leads to performance on a separate test set, in one or more realistic domains, that is better than performance on that test set without learning.

At a general level, there are two types of machine learning: inductive, and deductive. Deductive learning works on existing facts and knowledge and deduces new knowledge from the old. Inductive machine learning methods create computer programs by extracting rules and patterns out of massive data sets. Inductive learning takes examples and generalizes rather than starting with existing knowledge one major subclass of inductive learning is concept learning. This takes examples of a concept and tries to build a general description of the concept. Very often, the examples are described using attribute-value pairs.

Machine learning overlaps heavily with statistics. In fact, many machine-learning algorithms have been found to have direct counterparts with statistics. For example, boosting is now widely thought to be a form of stage wise regression using a specific type of loss function. Machine learning has a wide spectrum of applications including natural language processing, search engines, medical diagnosis, bioinformatics and cheminformatics, detecting credit card fraud, stock market analysis, classifying DNA sequences, speech and handwriting recognition, object recognition in computer vision, game playing and robot locomotion.

In our study we concentrate on the various paradigms, which are used in machine learning. Our review also examines the comparative study of machine learning technique with suitable application area.

This paper is organized as follows: In section 2 we discuss about the use of Neural Network in machine learning. CBR with application area is presented in section 3. CART is another efficient learning method described in section 4. Another paradigm rule induction is highlighted in section 5. In section 6 the impact of genetic algorithm and programming are discussed. Section 7 presents the discussion on various machine-learning techniques and conclusions and future direction are presented in section 8.

2. NEURAL NETWORKS

Neural networks have been established to be an effective tool for pattern classification and clustering [8, 15]. There are broadly two paradigms of neural learning algorithms namely supervised and unsupervised. Unsupervised neural algorithms are best suited for clustering patterns on the basis of their inherent characteristics [8, 14]. There are three major approaches for unsupervised learning: -

- (a) Competitive Learning
- (b) Self Organizing feature Maps
- (c) ART Networks

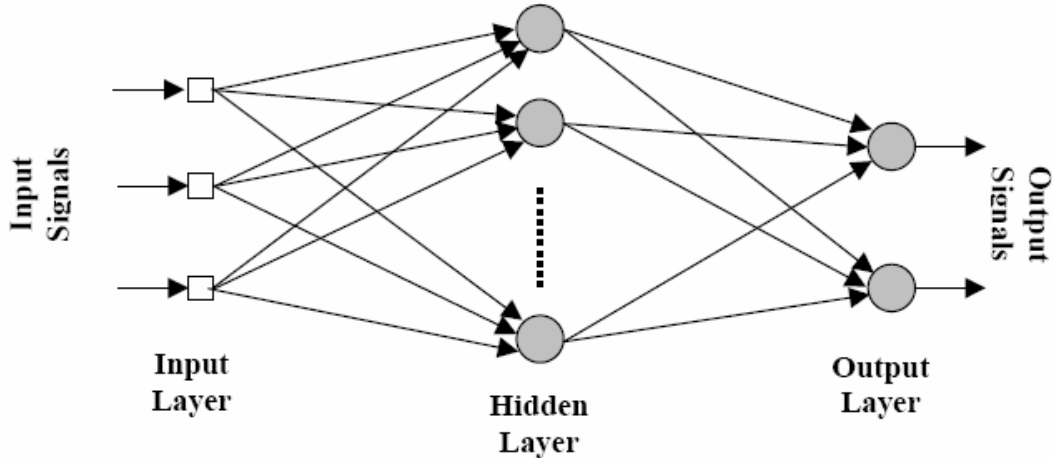


Figure 1: The architecture of the neural network

The other paradigm of neural learning is the so-called supervised learning paradigm. These networks have been established to be universal approximators of continuous/discontinuous functions and therefore they are suitable for usage where we have some information about the input-output map to be approximated. A set of data (Input-Output information) is used for training the network. Once the network has been trained it can be given any input (from the input space of the map to be approximated) and it will produce an output, which would correspond to the expected output from the approximated mapping. The quality of this output has been established to correspond arbitrarily close to the actual output desired owing to the generalization capabilities of these networks.

The activation function used is the log-sigmoid function as given in [9] can be expressed as: -

$$\Phi(a) = \frac{1}{1 + e^{-a}} \quad (1)$$

Where

$$a = \sum_{i=1}^N w_i x_i \quad (2)$$

w 's are the synaptic coefficients and x 's are the outputs of the previous layer. For the hidden layer x 's correspond to the input of the network while for the output layer x 's correspond to the output of the hidden layer. The network is trained using the error back propagation algorithm [9]. The weight update rule as given in [9] can be expressed as: -

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \quad (3)$$

where α is usually a positive number called the momentum constant, η is the learning rate, $\Delta w_{ji}(n)$ is the correction applied to the synaptic weight connecting the output of neuron i to the input of neuron j at iteration n , $\delta_j(n)$ is the local gradient at n th iteration, $y_i(n)$ is the function signal appearing at the output of neuron i at iteration n .

From experimental results we conclude that neural network can be used as test oracle, effort estimation, cost estimation, size estimation & other application areas of software engineering [1,7,

12, 13]. However the percentage error that can be tolerated will depend on the specific application for which test case is being design. The architecture and training algorithm will depend upon the space spanned by the test case parameters. There are some other systems like complex simulation in mechanical design, weather and economic forecasting and geological exploration that are built to solve unsolved problems using neural network for which there is no analytical solution.

The primary advantage of using neural network approach is that they are adaptable and nonparametric; predictive models can be tailored to the data at a particular site.

3. CASE BASED REASONING (CBR)

Case Based Reasoning is a technique by which we solve new problems by adapting the solutions from similarly solved problems. We take the instances of solutions from problems that have happened in the past and try to solve new problems by using these cases. Each such solution available to us can be termed as a case [11].

3.1 CBR Process

A general CBR process includes the following four processes.

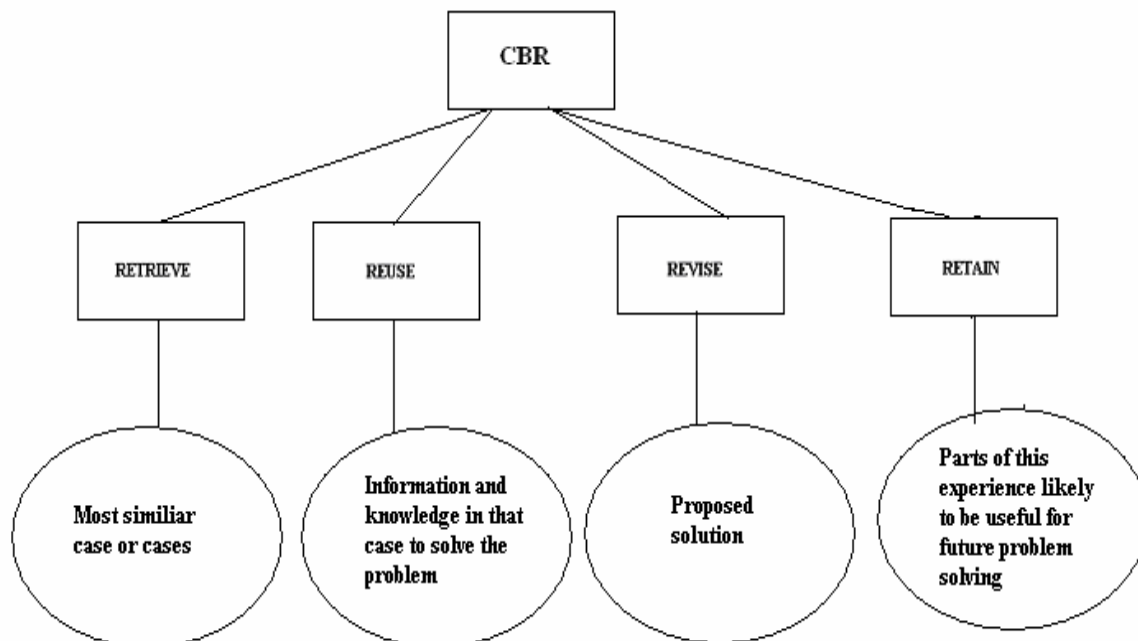


Figure 3 A General CBR Process

A new case is defined by the initial description of any problem. This new case is *retrieved* from a collection of previous cases and this retrieved case is then combined with the new case through reuse into a *solved case*. This solved case is nothing but a proposed solution to the defined problem. Once this solution is identified, applying it practically to the real world tests it. This process of testing is termed as *revision* of the problem. Then comes the process of *retain* where useful experience is retained for future reuse and the case base is updated by a new learned case or by modification of some existing cases.

Thus we can say that CBR is a four-step process:

1. RETRIEVE
2. REUSE

3. REVISE
4. RETAIN

The figure: 4 give a brief illustration of the CBR Cycle:

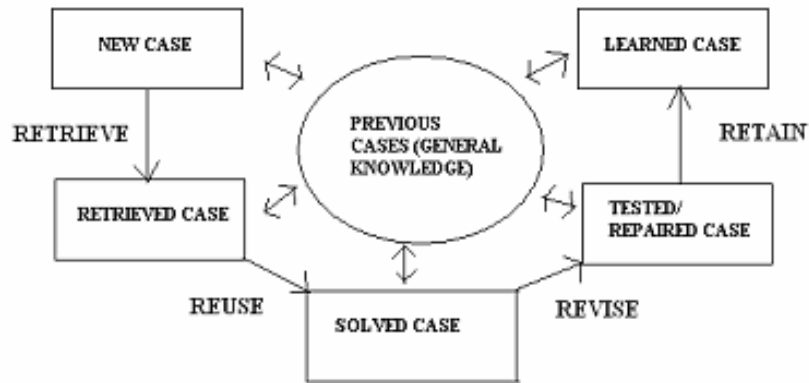


Figure 4 The CBR Cycle

It is clear from the figure that general knowledge plays a crucial in CBR. It supports all the CBR processes. General knowledge here implies domain dependent knowledge as opposed to specific knowledge embodied by cases. For instance in diagnosing a patient by retrieving and reusing the case of a previous patient, a model of anatomy together with casual relationships between pathological states may constitute the general knowledge used by a CBR system.

3.2 Fundamentals of Case Based Reasoning

3.2.1 Case Retrieval

The process of retrieval in CBR cycle begins with the problem description and ends when the best possible case from the set of previous cases has been obtained. The subtasks involved in this particular step include identifying features, matching, searching and selecting the appropriate ones executed in that order. The identification task finds a set of relevant problem descriptors, then the matching task returns those cases that are similar to the new case and finally the selection task chooses the best possible match. Among well-known methods for case retrieval are: nearest neighbor, induction, knowledge guided induction and template retrieval. These methods can be used alone or combined into hybrid retrieval strategies.

1) Nearest Neighbour (NN)

NN approach involves the assessment of similarity between stored cases and the new input case, based on matching a weighted sum of features.

2) Induction

This involves generating a decision tree structure to organize the cases in memory by determining which features do the best job in discriminating cases.

3) Knowledge guided induction

By applying knowledge to the induction process by manually identifying case features that are known or thought to affect the primary case feature we perform case retrieval. This approach is frequently used in conjunction with other techniques, because the explanatory knowledge is not always readily available for large case bases.

4) Template retrieval

Template retrieval returns all cases that fit within certain criteria often used before other techniques, such as nearest neighbour, to limit the search space to a relevant section of the case-base.

3.2.2 Case Reuse

This involves obtaining the solved case from a retrieved case. It analyses the differences between the new case and the past cases and then determines what part of the retrieved case can be transferred to the new case. CBR is essentially based on the concept of analogy wherein by analyzing the previous cases we formulate a solution for the new cases [5].

3.2.3 Copy

In the trivial cases of reuse we generally copy the solution of the previous cases and make it the solution for the new cases. But many systems take into consideration the differences between the two cases and use the adaptation process to formulate a new solution based on these differences.

3.2.4 Adaptation

The adaptation process is of two kinds:

Structural adaptation- Adaptation rules are applied directly to the solution stored in cases i.e. reuse past case solution.

Derivational adaptation- Reuse the method that constructed the solution to a past problem.

In structural adaptation we do not use the past solution directly but apply some transformation parameters to construct the solution for the new case. Thus this kind of adaptation is also referred to as transformational adaptation. In derivational adaptation we use the method or algorithm applied previously to solve the new problem [17].

3.2.5 Case Revision

After reusing the past cases to obtain a solution for the new case we need to test that solution. We must check or test to see if the solution is correct. If the testing is successful then we retain the solution, otherwise we must revise the case solution using domain specific knowledge.

3.2.6 Case Retainment- Learning (CRL)

The solution of the new problem after being tested and repaired may be retained into the existing domain specific knowledge. This process is called Case Retainment Learning or CRL. Retaining information involves selecting what information to retain, in what form to retain it, how to index the case for later retrieval from similar problems, and how to integrate the new case in the memory structure.

3.2.7 Case Based Learning

An important feature of CBR is its coupling to learning [2]. Case-based reasoning is also regarded a sub-field of machine learning. Thus, the notion of case-based reasoning does not only denote a particular reasoning method, irrespective of how the cases are acquired, it also denotes a machine learning paradigm that enables sustained learning by updating the case base after a problem has been solved. Learning in CBR occurs as a natural by-product of problem solving. When a problem is successfully solved, the experience is retained in order to solve similar problems in the future. When an attempt to solve a problem fails, the reason for the failure is identified and remembered in order to avoid the same mistake in the future. CBR can be applied to solve real world problems for instance handling of multiple disorders [16] or for engineering sales support [23].

4. CLASSIFICATION AND REGRESSION TREES (CART)

CART is a very efficient machine learning technique. The difference between this technique and other machine learning technique is that CART requires very little input from the analyst. This is in

contrast to other technique where extensive input from the analyst, the analysis of interim results and modification of method used is needed. Before going into the details of CART we identify the three classes and two kinds of variables, which are important while defining classification and regression problems.

There are three main classes of variables:

1) Target variable -- The “target variable” is the variable whose values are to be modeled and predicted by other variables. It is analogous to the dependent variable in linear regression. There must be one and only one target variable in a decision tree analysis.

2) Predictor variable -- A “predictor variable” is a variable whose values will be used to predict the value of the target variable. It is analogous to the independent in linear regression. There must be at least one predictor variable specified for decision tree analysis; there may be many predictor variables.

3) Weight variable -- You can specify a “weight variable”. If a weight variable is specified, it must be a numeric (continuous) variable whose values are greater than or equal to 0 (zero). The value of the weight variable specifies the weight given to a row in the dataset.

There are 2 main kinds of variables:

1) Continuous variables -- A continuous variable has numeric values such as 1, 2, 3.14, -5, etc. The relative magnitude of the values is significant (e.g., a value of 2 indicates twice the magnitude of 1). Examples of continuous variables are blood pressure, height, weight, income, age, and probability of illness. Some programs call continuous variables “ordered” or “monotonic” variables.

2) Categorical variables -- A categorical variable has values that function as labels rather than as numbers. Some programs call categorical variables “nominal” variables. For example, a categorical variable for gender might use the value 1 for male and 2 for female. The actual magnitude of the value is not significant; coding male as 7 and female as 3 would work just as well

CART builds classification and regression trees for predicting continuous dependent variables (regression) and categorical predictor variables (classification).

Regression-type problems: These are generally those where one attempts to predict the values of a continuous variable from one or more continuous and/or categorical predictor variables.

Classification-type problems: These are generally those where one attempts to predict values of a categorical dependent variable from one or more continuous and/or categorical predictor variables.

CART is a non-parametric statistical methodology developed for analyzing classification issues either from categorical or continuous dependent variables [24, 25]. If the dependent variable is categorical, CART produces a classification tree. When the dependent variable is continuous, it produces a regression tree.

4.2 Binary Recursive Partitioning

Consider the problem of selecting the best size and type of laryngoscope blade for pediatric patients undergoing intubations [20]. The outcome variable, the best blade for each patient (as determined by a consulting pediatric airway specialist), has three possible values: Miller 0, Wis-Hipple 1.5, and Mac 2. The two-predictor variables are measurements of neck length and or pharyngeal height. The smallest patients are best incubated with the Miller 0, medium sized patients with the Wis-Hipple 1.5, and the largest patients with the Mac 2.

CART is basically used to avoid the disadvantage of the regression techniques. CART analysis is a form of binary recursive partitioning [20]. The term “binary” implies that each node in a decision tree can only be split into two groups. Thus, each node can be split into two child nodes, in which case the original node is called a parent node. The term “recursive” refers to the fact that the binary partitioning process can be applied over and over again. Thus, each parent node can give rise to two child nodes and, in turn, each of these child nodes may themselves be split, forming additional children. The term “partitioning” refers to the fact that the dataset is split into sections or partitioned.

The figure:5 illustrates this kind of a partitioning. This tree consists of a root node (Node 1), containing all patients. This node is split based on the value of the neck length variable. If the neck length is < 2.45 centimeters, then those patients are put in the first terminal node, denoted Node -1, and the best blade is predicted to be a Miller 0. All other patients are placed in Node 2. The group of patients in Node 2 is initially assigned a Wis-Hipple 1.5 blade but they are also split based on there or pharyngeal height. Those patients with an or pharyngeal height less than 1.75 are placed in terminal Node -2, and assigned a Wis-Hipple 1.5 blade, while those with an or pharyngeal height 1.75 are placed in terminal Node –3 and assigned a Mac 2 blade.

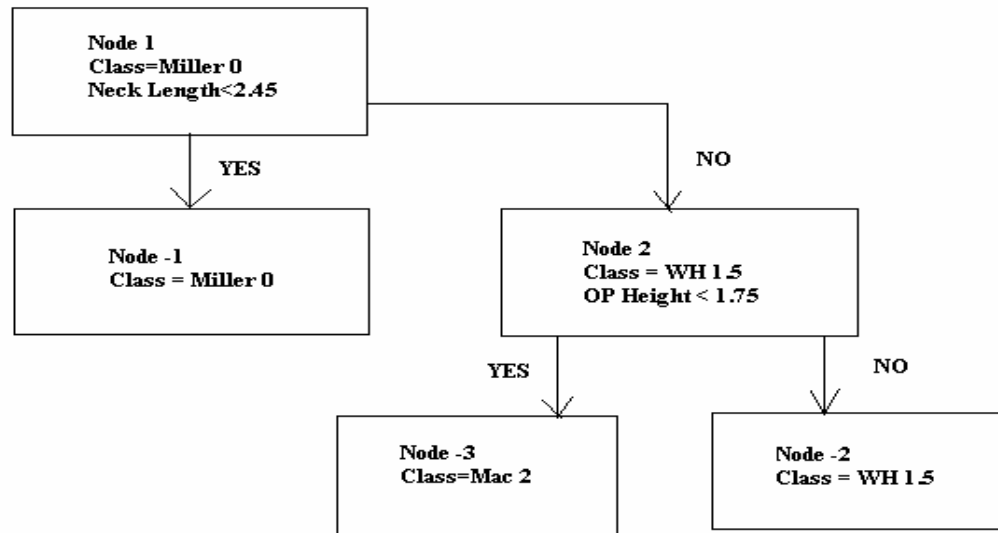


Figure 5 A CART Analysis Tree

4.3 CART Analysis

CART analysis is a tree-building technique, which is unlike traditional data analysis methods. It is ideally suited to the generation of clinical decision rules.

CART Analysis consists of four basic steps: -

1. It consists of tree building, during which a tree is built using recursive splitting of nodes. Each resulting node is assigned a predicted class, based on the distribution of classes in the learning dataset, which would occur in that node and the decision cost matrix. The assignment of a predicted class to each node occurs whether or not that node is remove space subsequently split into child nodes.
2. CART Analysis consists of stopping the tree building process. At this point a “maximal” tree has been produced, which probably greatly over fits the information contained within the learning dataset.
3. It consists of tree “pruning,” which results in the creation of a sequence of simpler and simpler trees, through the cutting off increasingly important nodes.

4. This step consists of optimal tree selection, during which the tree that fits the information in the learning dataset, but does not over fit the information, is selected from among the sequence of pruned trees.

5. RULE INDUCTION

Rule Induction is another very important machine learning method and it is easier because the rules in rule induction are transparent and easy to interpret than a regression model or a trained neural network. This paradigm employs condition-action rules, decision trees, or similar knowledge structures. Here the performance element sorts instances down the branches of the decision tree or finds the first rule whose conditions match the instance, typically using an all-or-none match process [19]. Information about classes or predictions is stored in the action sides of the rules or the leaves of the tree. Learning algorithms in the rule induction framework usually carry out a greedy search through the space of decision trees or rule sets, typically using a statistical evaluation function to select attributes for incorporation into the knowledge structure. Most methods partition the training data recursively into disjoint sets, attempting to summarize each set as a conjunction of logical conditions.

Rule Learning process

If we are given a set of training examples i.e. instances for which classification is known we find a set of classification rules which are used to predict new cases that haven't been presented to the learner before. While deriving these instances the bias imposed by languages must be taken into account such as restrictions imposed while describing data and we must also consider the language used to describe the induced set of rules.

Consider a binary classification problem of classifying instances into classes positive and negative. We are given a data description language, which impose a bias on the data, training examples, a hypothesis language imposing a bias on the induction rules and a coverage function defining when an instance is covered by a rule. Given the above data we need to find a hypothesis defined by a set of rules in a language, which is consistent that it does not cover any negative examples and is complete that it covers all positive examples. Thus in this manner, given the required data and the problem we can determine a set of rules, which classify the instances in that problem. This forms the basis of rule induction.

There are two main approaches to rule induction namely propositional learning and relational rule learning.

Propositional Rule Learning

Propositional rule learning systems are suited for problems in which no substantial relationship between the values of the different attributes needs to be represented. A set of instances with known classifications where each instance is described by values of a fixed collection of attributes is given. The attributes can have either a fixed set of values or take real numbers as values. Given these instances we then construct a set of IF-THEN rules. The output of learning is a hypothesis represented by a set of rules. After the rules have been defined determining the accuracy of such rules and then applying these rules to practical problems analyze their quality. In propositional learning the available data has a standard form with rows being individual records or training examples and columns being properties or attributes to describe the data.

Relational Rule Learning/ Inductive logic Programming (ILP)

When data is stored in several tables then it has a relational database form. In such cases the data has to be transformed into a single table in order to use standard data mining techniques. The most common data transformation approach is to select one table as the main table to be used for learning, and try to incorporate the contents of other tables by summarizing the information contained in the table into some summary attributes, added to the main table. The

problem with such single-table transformations is that some information may be lost while the summarization may also introduce artifacts, possibly leading to inappropriate data mining results. What one would like to do is to leave data conceptually unchanged and rather use data mining tools that can deal with multi-relational data. ILP is intended at solving multi-relational data mining tasks.

Thus ILP is to be used for data mining in multi-relational data mining tasks with data stored in relational databases and tasks with abundant expert knowledge of a relational nature. Another important concept within the realm of relational rule learning is that of boosting. Boosting is a particularly robust and powerful technique to enhance the prediction accuracy of systems that learn from examples [22]. Thus boosting helps to improve the overall efficiency of the results obtained.

An example to illustrate Rule Induction

Case Study (Making Credit Decisions)

Loan companies regularly use questionnaires to collect information about people applying for credit, which they then use in deciding whether to make loans. This process has long been partially automated. For example, American Express UK used a statistical decision process based on discriminated analysis to reject applicants falling below a certain threshold and to accept those exceeding another. The remaining 10 to 15 percent of the applicants fell into a borderline region and were referred to higher authorities giving loan for a decision. However, records showed that these authorities were no more than 50% accurate in predicting whether these borderline applicants would default on their loans. These observations motivated American Express UK to try methods from machine learning to improve the decision process. Starting with 1014 training cases and 18 descriptive attributes (such as age and years with an employer), Michie and his colleagues used an induction method to produce a decision tree, containing around 20 nodes and ten of the original features, that made correct predictions on 70% of the borderline applicants. In addition to achieving improved accuracy, the company found the rules attractive because they could be used to explain the reasons for decisions to applicants. American Express UK was so impressed that they put the resulting knowledge base into use without further development.

6. GENETIC ALGORITHMS AND GENETIC PROGRAMMING

The genetic approach to machine learning is a relatively new concept. Both genetic algorithms and Genetic Programming (GP) are a form of evolutionary computing which is a collective name for problem solving techniques based on the principles of biological evolution like natural selection. Genetic algorithms use a vocabulary borrowed from natural genetics in that they talk about *genes* (or bits), chromosomes (individuals or bit strings), and population (of individuals) [10]. Genetic algorithm approach is centered around three main processes- crossovers, mutation and selection of individuals. Initially many individual solutions are gathered together to make a randomly generated population. Genetic algorithms are based upon the Darwin theory of "The survival of the Fittest" depending upon the fitness function the best possible solutions are selected from the pool of individuals. The fitter individuals have greater chances of its selection and higher the probability that its genetic information will be passed over to future generations. Once selection is over new individuals have to be formed. These new individuals are formed either through crossover or mutation. In the process of crossover, combining the genetic make up of two solution candidates (producing a child out of two parents) creates new individuals. Whereas in mutation, we alter some individuals, which means that some randomly chosen parts of genetic information is changed to obtain a new individual. The process of generation doesn't stop until one of the conditions like minimum criteria is met or the desired fitness level is attained or a specified number of generations are reached or any combination of the above [21].

John Koza popularized GP, an offshoot of Genetic Algorithm in 1992. It aims at optimizing computer programs rather than function parameters.

GP is a supervised machine learning technique where algorithms are modeled after natural selection. These algorithms are represented as function trees where these trees are intended to

perform a given task [6]. In GP the fitter individuals are retained and allowed to develop whereas others are discarded [4].

GP works in a manner similar to genetic algorithm. It also follows the principles of natural evolution to generate a solution that maximizes (or minimizes) some fitness function [3]. GP differs from GA in the sense that GP tends to find the solution of a given problem by representing it as a array of integers while the goal of a GP process is to produce a computer program to solve the optimization problem at hand. GP cycle works as any evolutionary process. New individuals are created; tested and fitter ones succeed in creating their own children. The unfit individuals are removed from the population. The figure:6 illustrates how GP cycle works.

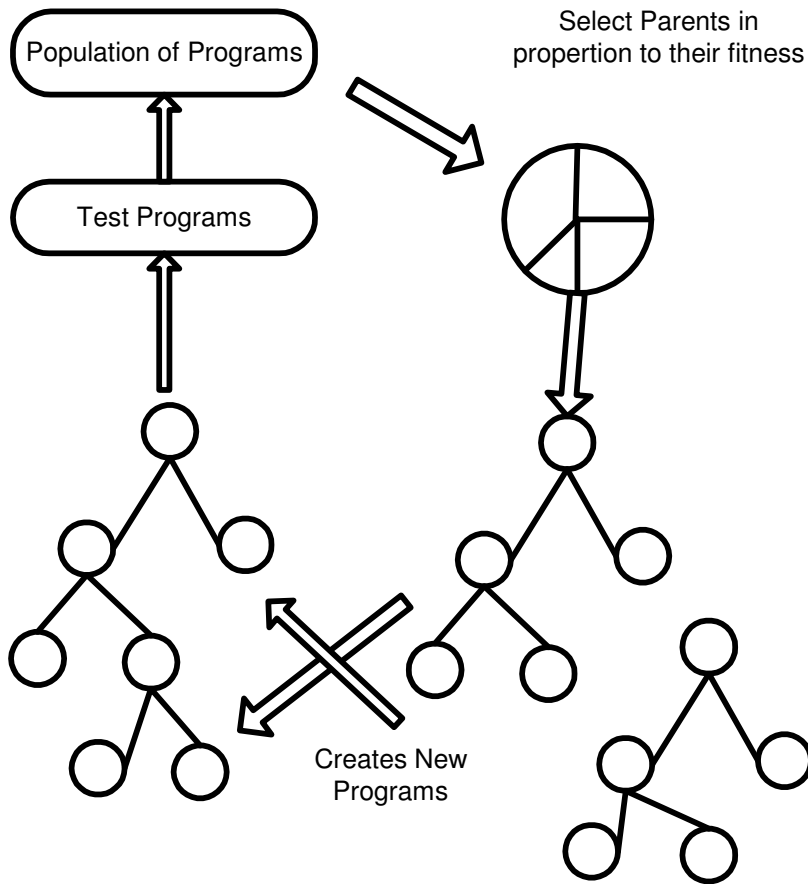


Figure 6: Genetic Programming Cycle

7. Discussion on Various Machine Learning Techniques			
Technique	Application Areas	Potential Benefits	Limitations
Neural Networks (NN)	Testing Effort Estimation Function Point Analysis Risk Management Reliability Metrics Sales Forecasting	Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience. Self-Organization: An ANN can create its own organization or representation of the information it receives during learning time. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.	Minimizing over fitting requires a great deal of computational effort. The individual relations between the input variables and the output variables are not developed by engineering judgment so that the model tends to be a black box or input/output table without analytical basis. The sample size has to be large.
Case Based Reasoning (CBR)	Help-Desk Systems Software Effort Estimation Classification and Prediction Knowledge Based Decision systems.	No Expert is Required The CBR Process is more akin to human thinking. CBR can handle failed cases (i.e. those cases for which accurate prediction cannot be made) No extensive maintenance is required.	Case data can be hard to gather. Predictions are limited to the cases that have been observed.
Classification and Regression Trees (CART)	Financial applications like Customer Relationship Management (CRM)	It is inherently non-parametric in other words no assumptions are made regarding the underlying distribution of values of the	Relatively new and somewhat unknown. Since CART is a new technique it is

	Effort Prediction (used in models like COCOMO)	<p>predictor variables.</p> <p>CART identifies splitting variables based on an exhaustive search of all possibilities. It has methods for dealing with missing variables.</p> <p>It is a relatively automatic machine learning technique.</p> <p>CART trees are easy to interpret even for non-statisticians.</p>	<p>difficult to find statisticians with significant expertise in this technique.</p> <p>CART may have unstable decision trees.</p> <p>CART splits only by one variable.</p>
Rule Induction	<p>Making Credit Decisions (in various loan companies)</p> <p>Diagnosis of Mechanical Devices</p> <p>Classification of Celestial Objects</p> <p>Preventing breakdowns in transformers</p>	<p>Simplicity of input variables.</p> <p>The representation in rule-based technique is easier to depict and understand.</p>	<p>No sufficient background knowledge is available. It is deduced from examples.</p> <p>Hard to maintain a complex rule-base.</p>
Genetic Algorithms (GA) and Genetic Programming (GP)	<p>Optimization</p> <p>Simulation of economic processes</p> <p>Scientific research purposes (Biological Evolution)</p> <p>Computer Games</p>	<p>GA and GP techniques can be applied to a variety of problems.</p> <p>GP is based on the 'Survival of the Fittest Scheme' allowing fitter individuals to develop and discarding unfit ones.</p> <p>GA is easy to grasp and can be easily applied without much difficulty</p>	<p>Resource requirements are large.</p> <p>It can be a time consuming process.</p> <p>GA practitioners often run many copies of the same code with the same inputs to get statistically reliable results.</p>

8. Conclusions and Future Directions

The main contribution of this review is to discuss the various Machine-Learning Techniques employed in effort estimation, cost estimation, size estimation and other field of Software Engineering. The paper also gives a relative comparison of all the techniques based on their applications, advantages and limitations. After analysis of all the techniques, we cannot state as any one technique being the best. Each technique has different application areas and is useful in different domains based on its advantages. Thus, keeping in mind the limitations of each of the techniques and also the prime focus being the improvement in performance and efficiency we should use that technique, which best suits a particular application. For instance GA and GP prove to be useful in the area of scientific research involving biological evolution whereas rule based techniques and CART analysis may be useful in many financial applications. Similarly CBR is being developed for use in Help- Desk Systems, a relatively new application and NN may be employed for Risk Management or Sales Forecasting.

Our study also encourages that no one technique can be classified as being the perfect machine learning technique. For this reason there is a strong need for better insight into the validity and generality of many of the discussed techniques. In particular we plan to continue with research on: -

When to use machine-learning techniques and estimation models.

How to select and combine a set of test cases for effective estimation technique & to get better results?

9. REFERENCES:

- [1] Aggarwal K.K., Yogesh Singh, A.Kaur, O.P.Sangwan "A Neural Net Based Approach to Test Oracle" ACM SIGSOFT Vol. 29 No. 4, May 2004.
- [2]. Agnar Aamodt, Enric Plaza. "Foundational Issues, Methodological Variations, System approaches." *AICom -Artificial Intelligence Communications*, IOS Press Vol. 7: 1, pp. 39-59.
- [3] *AI Globus*. "Towards 100,000 CPU Cycle-Scavenging by Genetic Algorithms." CSC at NASA Ames Research Center, September 2001.
- [4] Chris Bozzuto. "Machine Learning: Genetic Programming." February 2002.
- [5] Dr. Bonnie Morris, West Virginia University "Case Based Reasoning" *AI/ES Update* vol. 5 no. 1 Fall 1995.
- [6] Eleazar Eskin and Eric Siegel. "Genetic Programming Applied to Othello: Introducing Students to Machine Learning Research" available at <http://www.cs.columbia.edu/~evs/papers/sigcse-paper.ps>.
- [7] Gavin R. Finnie and Gerhard E. Wittig, "AI Tools for Software Development Effort Estimation", *IEEE Transaction on Software Engineering*, 1996.
- [8] Haykin S., "Neural Networks, A Comprehensive Foundation," Prentice Hall India, 2003.
- [9]. Howden William E. and Eichhorst Peter. Proving properties of programs from program traces. In *Tutorial: Software Testing and Validation Techniques*: E Miller and W.E.howden(eds.0. New York:IEEE Computer Society Press, 1978.
- [10] Hsinchun Chen. "Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms" available at <http://ai.bpa.arizona.edu/papers/mlir93/mlir93.html#318>.
- [11] Ian Watson & Farhi Marir. "Case-Based Reasoning: A Review " available at <http://www.ai-cbr.org/classroom/cbr-review.html>.
- [12] Juha Hakkarainen, Petteri Laamanen, and Raimo Rask, " Neural Network in Specification Level Software Size Estimation", *IEEE Transaction on Software Engineering*, 1993.
- [13] Krishnamoorthy Srinivasan and Douglas Fisher, "Machine Learning Approaches to Estimating Software Development Effort", *IEEE Transaction on Software Engineering*, 1995.
- [14] Kohonen T., "Self Organizing Maps", 2nd Edition, Berlin: Springer- Verlag, 1997.
- [15]. Mayrhauser A. von, Anderson C. and Mraz R., "Using A Neural Network to Predict Test Case Effectiveness" – *Procs IEEE Aerospace Applications Conference*, Snowmass, CO, Feb.1995.

- [16] Martin Atzmueller, Joachim Baumeister, Frank Puppe, Wenqi Shi, and John A. Barnden " Case Based Approaches for handling multiple disorders" Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society, 2004.
- [17] Nahid Amani, Mahmood Fathi and Mahdi Rehghan. "A Case-Based Reasoning Method for Alarm Filtering and Correlation in Telecommunication Networks" available at <http://ieeexplore.ieee.org/iel5/10384/33117/01557421.pdf?arnumber=1557421>.
- [18] Pat Langley, Stanford and Herbert A. Simon, Pittsburgh. "Application of Machine Learning and Rule Induction." available at <http://cll.stanford.edu/~langley/papers/app.cacm.ps>.
- [19] Peter Flach and Nada Lavrac. "Rule Induction" available at www.cs.bris.ac.uk/Teaching/Resources/COMSM0301/materials/RuleInductionSection.pdf.
- [20] Roger J. Lewis. "An Introduction to Classification and Regression Tree (CART) Analysis" Presented at the 2000 Annual Meeting of the Society for Academic Emergency Medicine in San Francisco, California.
- [21] Stephen M Winkler, Michael Aenzeller and Stefan Wagner. "Advances in Applying Genetic Programming to Machine Learning, Focusing on Classification Problems" available at <http://www.heuristiclab.com/publications/papers/winkler06c.ps>.
- [22] Susanne Hoche. "Active Relational Rule Learning in a Constrained Confidence-Rated Boosting Framework" PhD Thesis, Rheinische Friedrich-Wilhelms-Universitaet Bonn, Germany, December 2004.
- [23] Watson, I. & Gardingen, D. " A Distributed Case-Based Reasoning Application for Engineering Sales Support". In, Proc. 16th Int. Joint Conf. on Artificial Intelligence (IJCAI-99), Vol. 1: pp. 600-605, 1999.
- [24] Yisehac Yohannes, John Hoddinott " Classification and Regression Trees- An Introduction" International Food Policy Research Institute, 1999.
- [25] Yisehac Yohannes, Patrick Webb " Classification and Regression Trees" International Food Policy Research Institute, 1999.