Using Decision Trees to Create a Control Mechanism for regulation of the Hot Metal Temperature of the "G" Blast Furnace at Steelcorp

by
Mu'taz M. Qubbaj

Advanced Undergraduate Project (6.199)

Supervised by Dr. Amar Gupta

Abstract

The Hot Metal Temperature (HMT) is an important indicator of the state of the blast furnace, as well as an important determinant of the quality of pig iron produced. Therefore, scientists would like to find a way to control the pig iron's HMT based on the measurements of various other parameters in the furnace. Utilizing See5®, a data-mining tool created by RuleQuest®, decision trees were used to structure a rule-based predictor of the HMT (on an hourly basis) of pig iron produced from a type "G" blast furnace based on the measured values in the furnace. This predictor is subsequently used to structure the HMT control mechanism.

While trying to structure the predictor that provides the best predictive power, upon which the control mechanism is based, various issues become apparent. Initially, the important choice of decision tree structure is carried out based on which structure would optimize and facilitate the control mechanism's functionalities.

The control mechanism is set up in conjunction with a pre-established cost function that dictates the additional cost incurred by taking on a particular path down the decision tree formulated by See5®. The goal, in this context, is to minimize this incurred cost when trying to achieve a certain desired state, i.e., a new specified range of HMT. In addition, the control mechanism is structured to incorporate certain constraints on the involved input parameters. Such constraints would include the maximum amount of a certain element that can be used before reaching a potentially hazardous state or ruining the quality, in terms of composition, of the metal being produced.

Coverage of the method by which a certain path is chosen out of alternative potential paths to reach a desired state is also carried out. This is done within the realm of a feedback procedure that 'guesses' a path to take. Path selection is based on an analysis of the correlation predicted between the relevant input variables and the HMT using Cubist®. This an application also created by RuleQuest® that to produce rule-based models for numerical prediction of the desired variable, HMT in this case, based on large data sets. The method of path selection will rely on this analysis in conjunction with what is defined in the paper as temperature-shift brackets, which categorize the available alternative paths by the respective costs of their

execution. These alternative paths are subsequently run through the structured predictor for all the minimal-cost paths within the temperature-shift brackets up to the higher ones. The predicted HMT resulting from each of the paths is reported until the desired HMT is predicted as an outcome. Precautions are taken to ensure that each proposed path lies within the set constraints of the process, or else the path is by default discarded.

The control also incorporates a continual learning mechanism. This entails taking in the measured results of any particular path implementation and feeding them into the control data set, thus 'upgrading' the data set. This would be followed by reformulating the predicting decision tree, using this updated data set, to be subsequently incorporated into the control mechanism in place of its predecessor.

Table of Contents

The importance of Hot Metal Temperature (HMT) as an indicator of the internal state of a blast furnace as well as of the quality of the pig iron being produced has fueled the desire of scientists and blast furnace operators to attain a measure of control over the HMT. This control would be based on the current conditions of the furnace as well as on the levels of the various input chemicals used. Unfortunately, no precise function for mapping these input variables to a controller of HMT exists. The production of pig iron involves complicated heat and mass transfers and introduces complex relationships between the various raw materials and chemicals used.

In the control mechanism proposed below, decision trees are used as a means of modeling the aforementioned complex inter-variable relationships. The choice of utilizing decision trees and rule sets to structure the control mechanism is a result of the ease with which the decision trees or rule-sets can be easily understood. In turn, these decision trees could easily be manipulated to attain an optimal structure for the desired control. The ease in this case may be seen as a relative measure to the other options within the data-mining realm currently available for predictor formulation.

The purpose of this paper is to examine the feasibility of structuring a control mechanism for the HMT of pig iron utilizing available decision tree and rule-based predictor formulation technology. This examination will entail formulating the best method in which to structure this control effectively utilizing the formulated decision trees and rule sets.

The paper addresses several issues regarding the data used, all of which are important in the context of proper formulation and manipulation of the data set at hand. One of the issues that this paper addresses is exactly how to form the nodes of the decision trees. In other words, what is the optimal decision tree structure that should be used to create the control mechanism? Should the tree structure take on the shape of one large tree in which each path involves entering values for every single variable? Should the tree originate from an intermediary state (initial HMT) and have leaves that pose the final state to be attained? These considerations have a significant effect in determining how the formulated predictor can be most effectively manipulated to structure a control, and how the workings of a cost function can be incorporated within. This is done to the end of enabling the controller to choose the best alternative out of a range of alternatives leading to the same desired result.

The decision tree predictor application, See5®, is first utilized to formulate a predictor of the HMT of pig iron based on seventeen input parameters that reflect the current condition of the blast furnace. The network uses hourly averaged data of variables such as coke content, hot blast temperature, and silicon content to predict the HMT for the current hour.

The different See5® settings used throughout the predictive decision tree formulation were examined to establish the most optimal settings that should be used to minimize error while minimizing the amount of time taken to formulate the predictor to be used to structure the control at hand. The importance of analyzing this is discussed with regards to its relevance in reformulating the decision tree based predictor following the execution of the controller's recommendation.

Further analysis is carried out on establishing a learning mechanism by which the tree or rule set would come to include the results of implementing a recommendation given by the control. These results would subsequently be used as part of the HMT data set, to create the subsequent predictor to be used by the control in the future.

The structure of the paper is as follows. Section 2 provides an overview of the blast furnace and the basics of decision trees and rule-based methodologies. In addition, a survey of previous work is presented. Section 3 describes the methodology of the project. In particular, it describes how the data had to be structured such that it could be manipulated by the application See5®. This section describes the tree structure choices available and why a certain structure was selected. In addition, a description of how the data had to be manipulated to achieve this structure is included. This section also covers the study of the different settings available to the user of See5®, and determining the optimal combination of these settings to use.

Section 4 presents the details of the functioning of the controller, covering information relating to the correlation between certain input variables and the HMT. It also defines and examines the incorporation of both temperature-shift brackets and underlying constraints into the functioning of the controller. The major points of the paper are then summarized in the conclusion.

Section 2: Background

2.1 The blast furnace

The blast furnace is considered one of the most important units in the integrated steel production process. It is there that the raw material for steel, pig iron, is produced. Iron is usually found in ore deposits, in high concentration, where it resides as an oxide. Thus, the main purpose of the blast furnace is to remove the oxygen from iron oxides, creating pig iron as the end product [6]. This process involves massive heat transfers and combustion that raise the temperature of the blast furnace to over one thousand degrees Celsius.

There are three main raw materials used to produce pig iron in a blast furnace. These are iron ore (subsequently reduced to pig iron), coke (a by-product of coal), and limestone. The main purpose of the coke is to raise the blast furnace's internal temperature high enough (between 1450 to 1500 degrees Celsius) to allow the iron ore to be purified. In addition, the coke itself acts as a reducing agent. Limestone is used as a flux to remove impurities from the iron ore.

The process of producing pig iron begins by first producing the coke. This is done through a process called carbonization, whereby blended coal is heated in a coke oven to produce the coke. Once this process is completed, the coke is taken out of the oven and cooled before it is used in the next part of the process.

The iron ore itself also goes through a sort of pre-processing procedure. The ore is mixed with fluxes and coke, and then heated in a sinter plant. This is done by placing the coke on a moving conveyor belt, and heating it to high enough temperatures so that the fluxes and ore particles fuse together. The resulting product is called sinter. This intermediate product is also used by the blast furnace. The reason for doing this is that using sinter makes the overall process of the blast furnace more efficient.

Once processed, iron ore lumps, sinter, coke and limestone are added to the top of the blast furnace in the form of pellets. Then, a blast of hot air containing oxygen is injected into the furnace from the bottom through nozzles called tuyeres. The oxygen in this blast of air combusts with the coke present, causing the temperature of the furnace to rise to around two thousand degrees Celsius. Meanwhile, the iron ore inserted at the top of the furnace makes its way down. The carbon monoxide created from the combustion of the coke and the oxygen then rises up through the furnace. This rising gas removes the oxygen from the iron ore making its way down the furnace. This converts the iron ore to liquid or molten iron. The temperature of this iron ranges from 1450 to 1500 degrees Celsius. The molten iron (or hot metal, as it is commonly called) then collects at the bottom of the furnace and is tapped at regular intervals by opening a hole (a tap-hole) and letting the iron flow out. Impurities that were present in the ore, such as silicon dioxide, react with the residual limestone to produce a substance called slag. This molten slag floats on top of the molten iron because it is lower in density than the pig iron. Thus, the molten slag can be drawn off from the iron and tapped separately.

The produced molten iron (hot metal) is then allowed to flow into torpedo ladles. These ladles are containers by which the liquid iron is transported to the steel plant for steel manufacturing. Once at the steel plant, the pig iron is sent through a unit called the basic oxygen furnace. During this part of the steel making process, the impurities of the molten pig iron are oxidized by blasts of oxygen [5]. This results in the production of carbon steel.

One of the major indicators of the quality of the pig iron produced is its silicon content. Silicon enters the blast furnace through the coke ash as well as through the iron ore. In the high-temperature regions of the furnace, some of the silicates in the coke become dissolved as silicon in the molten iron. The general rule is that the molten iron produced should not contain much silicon. Relatively high silicon content in the molten iron reflects that the blast furnace is producing the pig iron inefficiently.

Another related quality indicator of the molten iron is its HMT. It has been shown that HMT is related to the silicon content of the metal and that these two indicators both change in the same way given the same set of blast furnace conditions.

Thus, the ability to control HMT might indicate the ability to control silicon content. The HMT is also important because it reflects the internal state of the furnace. These facts provide the primary motivations for the desire to control HMT, and, therefore, for the work presented later in this paper.

2.2 Decision Trees

A decision tree is a graph of nodes connected by arcs, with each node corresponding to a non-goal attribute and each arc to a possible value of that attribute. In this case, the non-goal attributes are the values of the various input variables that are seen to have a direct effect on the HMT. Such variables include the level of coke within the furnace, and the amount of gas being pumped into the furnace among many others.

A leaf of the tree specifies the expected value of the goal attribute for the records described by the path from root to leaf. In this case, the goal attribute is the value of the HMT at a certain point in time.

A problem that arises in the event that decision trees are utilized is that they take on a complicated structure that is hard to interpret. As such, rule-sets, taking on an intuitively easier format of antecedents and consequents (if antecedent, then consequent format), are incorporated into decision tree analyses to alleviate some of this difficulty. This is done by means of converting each tree into a set of rules, converting each path from root to leaf into a rule, with the antecedents, the test attributes, and the consequent, in this case, the leaf. An example of a decision tree based on the particular data set taken from Winston's book, Artificial Intelligence [11], and the subsequent conversion of this decision tree into a rule set is presented in the next paragraph.

The data set in Table 1 [4] summarizes data relating to the characteristics of a set of beach-goers, such as their hair color, height, weight, and whether or not they had applied suntan lotion. This is done in an attempt to find the relationships that exist between these characteristics and the susceptibility of being sunburned while at the beach, if any such relationships do exist. These relationships will then establish a prediction measure of potential incidences of sunburn among beach-goers based on whether or not they have the same characteristics, both physical and whether or not they used lotion, as those people found to be more susceptible to sunburn by analyzing Table 1. This table is subsequently converted into the decision tree in Figure 1 to elucidate the relations that may exist between the aforementioned characteristics and the incidence of sunburn.

Sunburn at the beach ?

NAME	HAIR	HEIGHT	WEIGHT	LOTION	RESULT
Sarah	Blonde	Average	Light	No	**sunburn**
Dana	Blonde	Tall	Average	Yes	**None**
Alex	Brown	Short	Average	Yes	**None**
Annie	Blonde	Short	Average	No	**sunburn**
Emily	Red	Average	Heavy	No	**sunburn**
Pete	Brown	Tall	Heavy	No	**None**
John	Brown	Average	Heavy	No	**None**
Katie	Blonde	Short	Light	Yes	**None**

Example from Winston (1996)

Table 1: Tabulation of varying characteristics (hair color, height, weight, and sun-tan lotion use) of beach-goers and the observed incidence of sunburn amongst them.

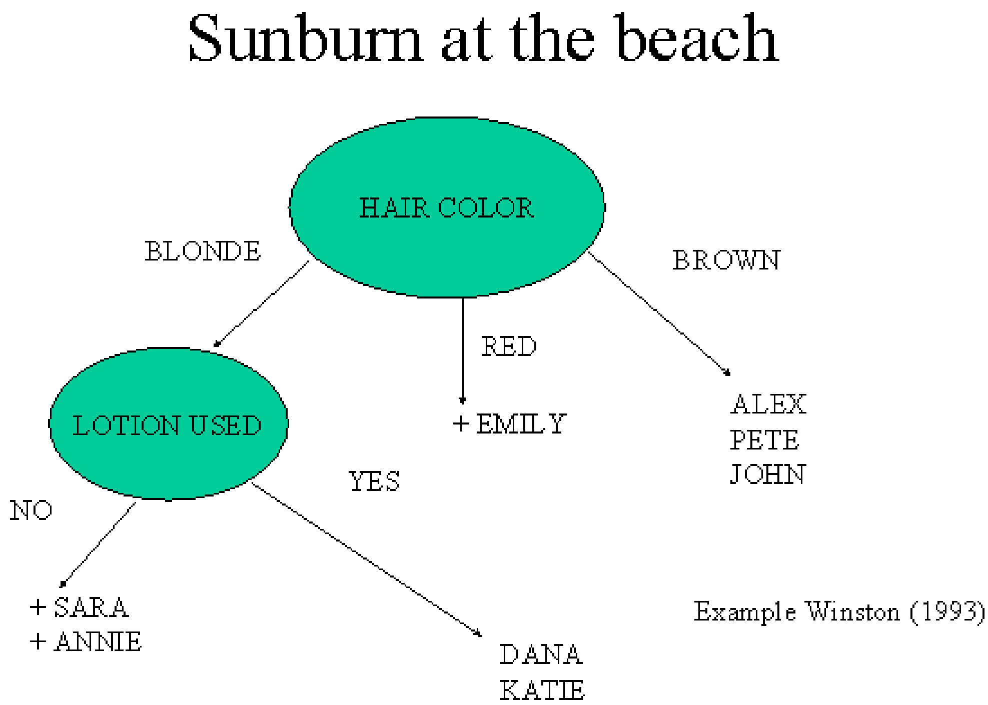

Figure 1: Decision tree created from information within Table 1, classifying subjects on basis of hair-color and subsequently lotion use. This is done to classify incidences of sunburn amongst subjects in a homogenous manner.

This decision tree is subsequently converted into the following rule set by following each path of the decision tree and structuring the rules in antecedent and consequent format:

If   **the person's hair color is blonde**

the person uses lotion

Then **nothing happens**

If   **the person's hair color is blonde**

the person uses no lotion

Then the person turns red

If the person's hair color is red

Then the person turns red

If the person's hair color is brown

Then nothing happens

As for an overview of the currently existing algorithms available for the creation of the necessary decision trees to be utilized, the most commonly used ones are CART®[1], CHAID®[2], and C4.5®[3] [1]. CART® can only build binary trees and it grows the full tree before pruning it, causing problems of over-fitting[4]. C4.5® is similar to CART® except that it can produce varying numbers of branches per node. While CART® and C4.5® can accept both categorical (i.e., the output variable value is expressed in terms of a range of values or a category) and continuous (i.e., the output variable is expressed in terms of a specific value from within a continuous range of values) values, CHAID® is restricted to categorical variables. Continuous variables will have to be broken down into categories when using CHAID®.

---

[1] CART® stands for classification and regression trees. The method was first published in 1984.

[2] CHAID® stands for chi-squared automatic interaction detection. It was first published in 1975.

[3] C4.5® is successor to ID3®.

[4] Over-fitting occurs when the information is too detailed. For example, some road maps are very detailed, including every street in a relatively small area. Other maps are more general, covering the major roads in a larger area. Which is the better map to use? If we need to travel a large distance, it may be difficult to figure out the best path from a patchwork of detailed maps. The detailed map overfits the information.

There are several major advantages as well as disadvantages in using decision trees. Decision trees will generally generate understandable rules. It is easy to follow any one path through the tree, so explaining the decisions along the way is easy. Computation cost for each split is inexpensive. In practice, algorithms tend to produce decision trees with a low branching factor with simple tests at each node. The tree does not grow out of hand and these tests translate into simple Boolean and integer operations that are fast and inexpensive. Using decision trees, the best field at splitting the training records can be singled out for analysis. This enables the user to figure out which variable influences his/her data the most.

More recent entrants on the predictor formulation front, See5® and Cubist, are the next generation follow-ups to C4.5®. These are applications that have been created by the RuleQuest® team surpassing the performance of their predecessor in many ways. See5® works solely on categorical values, whereby Cubist® works solely on data sets of a continuous nature. After undergoing sample case testing comparisons between the two generations of applications, See5® and Cubist® are found to be nearly two hundred times faster than C4.5® on available test data. In addition, these applications use less than 10% of the memory used by their counterpart C4.5®. See5® and Cubist® also outperform C4.5®, producing results with higher predictive accuracy than the latter. This can be noted by a significant decrease in the error levels associated with the more recent applications as opposed to their predecessor.

An additional advantage that See5® and Cubist® have over their counterpart is the added functionality of boosting, a standard part of the more recently created applications. Boosting is a method by which an "ensemble" of classifiers is created that may be more accurate than an individual classifier. Boosting relies on resampling techniques to obtain different training sets for each of the classifiers. Boosting, in effect, is a technique for generating and combining multiple classifiers to give greatly improved predictive accuracy. The predictive error rate subsequent to boosting is reduced substantially on the training data sets when compared to that of C4.5®. This rate is found to be about one-third of the error rate of C4.5®'s single classifiers.

See5® includes full support for boosting with any number of trials. It takes longer to produce boosted classifiers, but it can be worth the additional time in terms of the predictive accuracy to be attained by using boosting. Since, in this case, peak predictive accuracy is required, the following section includes an analysis of decision tree predictive capabilities under various boosting scenarios in the context of resulting error levels and the time taken to create the tree in question.

2.3 Previous work

There have been many attempts by researchers to use AI techniques to control certain state variables within the furnace, such as the HMT, based on structuring effective predictors using the measured conditions within the furnace. However, modeling the relationships between various variables in the blast furnace has been quite difficult using standard statistical techniques [2]. The main reason is that non-linearities exist between the different parameters used in pig iron (hot metal) production.

Production of hot metal in a blast furnace is the result of complex chemical reactions that scientists have not been able to model explicitly using traditional techniques. As such, neural networks have been proposed as a potential solution. This section gives a brief overview of the major work done in this area and provides motivation for the work presented within this paper.

Abhay Bulsary, Henrik and Bjorn Saxen observed promising results when using multi-layered feed-forward Artificial Neural Network (ANN) models to predict the silicon content of hot metal from a blast furnace. Time-dependencies (time lags) between each of the various inputs used and the output (silicon content) were found to be significant. For this reason, each input was lagged from the output by its "optimal" amount (the lag such that, when implemented, produced the highest correlation between the input variable and the output). The input variables used included: blast pressure (as fifteen minute averages, with time lags of thirty minutes and one hour), blast volume (as one hour averages, with a time of five hours), calculated heat loss (as one hour averages, with time lags of one hour and seven hours), oil injection (one hour averages lagged by five hours), and the silicon content of the previous two taps. Feed-forward networks with one, two and three hidden layers were tried and the method used to update the neural network's weights was a non-linear variant of the traditional back-propagation learning algorithm. That paper concludes that the feed-forward ANNs had produced considerably better results than standard linear time-series prediction. Since silicon content is known to be directly related to HMT, the success here in predicting silicon content using ANNs provides an indication that ANNs may also be useful in the prediction of HMT.

Feed-forward neural networks were used again by Bulsari and Saxen [2] when trying to classify the state of a blast furnace based on the measurement of blast furnace temperatures. Horizontal probes within the blast furnace were used to measure the internal temperature. The measurements from these probes are very important because they provide information regarding the distribution of gas flow within the furnace. This, in turn, provides information regarding the state of various critical components of the blast furnace. However, the knowledge relating blast furnace temperature to gas distribution within the furnace is highly complex and non-linear, and, traditionally, had only been inferred by people who had experience operating the blast furnace. Thus, a neural network seemed like a good solution. Based on the measurements of the horizontal temperature probes, the network could classify the state of the gas distribution in the blast furnace into one of six categories. Bulsari and Saxen constructed a feed-forward network using back-propagation to train the network. One major result was that larger networks, with more hidden nodes and hidden layers, seemed to work much better than smaller networks. As the number of hidden nodes decreased, the accuracy of the network also decreased. The worst model, according to Bulsari and Saxen, was the linear regression model. Thus, since more hidden nodes provided better results, Bulsari and Saxen clamed that the larger networks were more capable of capturing the complex relationships between the variables in the system to classify the state of the blast furnace more accurately.

Himanshu Singh, Nallamal Venkata and Bramha Deo [10] tried four different artificial neural network models to predict the silicon content of hot metal based on the following set of variables: coke rate, hot blast temperature, slag rate, top pressure, slag basicity and the logarithm of blast kinetic energy. The simplest learning algorithm used was standard back propagation. The networks consisted of three layers (input, hidden and output) and the number of hidden nodes varied from 6 to 11. Other models tried included using a dynamic learning rate model, a functional link model, and a fuzzy neural network. The best results were obtained from the fuzzy neural network, with the performance of the back-propagating model providing the next best results. The results of this paper clearly show that the use of ANNs increased the predictive ability relating to silicon content as compared to conventional models.

Osamu Lida, Yuichi Ushijima and Toshiro Sawada [7] used neural networks to implement a module of a blast furnace control system. This work was particularly interesting, especially in the context of the current paper, because neural networks were used to predict HMT. The blast furnace control system, within the context of this paper, diagnoses the operating condition of the blast furnace from a large amount of data and gives advice to the operator about the necessary changes to make to stabilize the state of the furnace or to reach some particular condition within the furnace. In particular, neural networks were used to explain the relationship between gas flow distribution and three critical parameters: the charging pattern permeability in the furnace, the HMT and hot metal silicon content. The type of neural network used was self-organized feature mapping. In this neural network model, neurons are located in a 2-dimensional arrangement and all neurons have dimensional connectivity weights whose number is equal to the number of input variables in the data. The weights of these connections are repeatedly updated through learning. The result, again, was that neural networks were good at capturing the non-linearities in the data and provided good prediction and classification of the current conditions of the blast furnace.

It is important to note that the previous work described solely focused on structuring predictors of the state variables within the furnace, with the intention of controlling these state variables. However, previous work in the field of control structuring only sought neural networks as the source of a solution to their predictive needs, leaving an obvious void in the exploration of the capabilities of other data-mining techniques as methods by which to structure predictors of the state-variables within the furnace. Thus arises the motivation of my work, catering to detailing the structuring of the control mechanism of the state variable, HMT, while examining the capabilities of an alternative data-mining tool, decision trees, as the means by which to structure the control mechanism's underlying predictor.

Section 3: Methods

3.1 Choosing Optimal Decision tree structure for Predictor

An analysis of the available data set, as well as the desired output, reveals the importance of studying the available alternative methods of data manipulation to achieve optimal functioning of the control mechanism. The following include the most significant questions that need to be answered:

- What type of decision trees can be structured using the data at hand?
- What are the advantages and disadvantages to each type?
- How must the data be processed to attain the desired tree structure?

As for the types of decision trees that can be structured for predictive purposes, these can be split into two types. The first is that of the structure where the origin node is a node that contains no data. In this case, there is no previously established state prior to the prediction procedure. The state variables, i.e., Input variables, are channeled into the procedure at each subsequent node of the decision tree. In effect, each path down the tree has to pass through a node that determines every input value from the origin of the tree to the tree's respective leaves. Using this type of tree would entail having to enter every input variable value. The proposed alterations in any of these variable values, such as increasing the amount of coke to be added by altering that variable (coke amount) value would be entered at the node in the decision tree at which this value is to be entered. All other variable values would be maintained constant at the levels that they are at while the data-entry process of the control mechanism is underway. After entering the data (input variable values) with necessary alterations into the system, the predictor will give the predicted value of the HMT.

The second type of tree structure that could be utilized to structure the predictor is one in which the tree is actually made up of a great number of sub trees. The origins of these sub trees would depict a specific previously encountered state, i.e., previous HMT values extracted from the available data set. The leaves of these trees would depict final state HMT values that would be reached by going through a specific path of a tree from its origin, a previous HMT value, through the nodes down that path. These nodes would be structured to include only the alterations in input variables incurred on the system while it was in the original state to bring on the change in the HMT to the leaf, i.e., the final state of the system.

The advantage of using the first type of tree is that every possible element affecting the determination of the HMT is taken into account by running through all the nodes of the tree. It is also important to observe the importance of proposing solutions that fall within the given constraints of the system at hand. This can be easily done using this tree structure since all the information that may be necessary to evaluate for consistency with these constraints is available in each particular path of the tree. An example of this is the verification that the level of a certain element within the furnace has to be less than a certain percentage of the additives to the furnace. This can be carried out by examining the current data being provided to the tree and checking to see whether or not the proposed alteration level of the element in question satisfies the underlying constraint. If it does, then the paths involving this alteration of value should be considered as potentially viable methods to achieve the desired change in the HMT within the furnace. However, if the proposed level of the element does not satisfy the constraint in question, then the paths involving such an alteration should not be considered, and are thus ignored.

On the other hand, the first type of tree poses the disadvantage that, for the predictor to function, the value of every variable involved in the formulation of the predictor needs to be known. As such, data that may be irrelevant to the change in the HMT being analyzed would need to be entered in addition to the data relaying the proposed alterations. These data may not be needed to control the HMT, but they are necessary for the functioning of the controller. As such, it may be seen as redundant in cases in which the alteration of only one variable will bring on the desired change in the HMT within the furnace.

Moving onto the advantages and disadvantages of using the second type of tree, the first observed advantage is that of the considerable difference in the amount of information needed to run the predictor, and hence the HMT controller. This is because the only data required by this type of tree are the initial HMT before alterations, the final HMT after alterations, and the alterations that took place in the levels of the input variables that brought on the change in the HMT from the initial to the final HMT.

A major disadvantage of this type of tree is that the origin of every tree only relays a certain HMT, which may have been reached by one of a great number of ways. There is no one specific way to attain a certain HMT. This would depend on the previous levels of certain elements within the furnace dictating the initial HMT. As a result, it would be extremely difficult to generalize the effects of a certain alteration in an input variable's level, within the furnace, on the HMT level. The effects of this alteration would most likely depend on the levels of certain elements within the HMT mixture prior to making these alterations. This is significantly important in that different results may arise in the event of carrying out the same alterations at two different points in time on a blast furnace and its contents, even if the HMT at those two instances is the same.

A related disadvantage is that of the increased difficulty in analyzing whether or not proposed courses of action satisfy the underlying constraints that have been set out. This disadvantage arises since the only data available within the control would be both the initial and final HMTs and the alterations that lead to the transition between the former to the latter. As a result, the information available may not be sufficient to carry out the necessary constraint analyses. This can be seen within the context of the aforementioned example relaying the constraint that the level of a specific element has to be a certain percentage of the total mixture of additives to the furnace. Since the available data only includes alterations in the levels of input variables, it would not include data relevant to the constituents of the whole additive mixture As a result, there would be no way to check whether or not this constraint is satisfied by the proposed path of alterations dictated by the tree.

As for the available data, this has been provided in the form of records of the various input variable levels and the HMT temperature at specific points in time. These data are compatible with the first type of tree structure as every variable value at a specific point in time is available. Hence, the core requirements for structuring a tree of the first type based on the available data are satisfied. A problem exists, however, when examining the case of the adequacy of the available data in the context of structuring the second type of decision tree. The problem is that the data are available in a format that states the values of relevant variables at a specific point in time, whereas the second type of tree requires data that tracks changes in the levels of these variables. An additional problem is the unavailability of the initial temperatures in the data records, thus relaying the lack of a core piece of data necessary for the structuring of the second type of tree.

Upon reviewing the advantages and disadvantages of using the aforementioned types of tree structures to formulate the predictor, as well as the problems arising within the context of preparing the available data set to structure these two types of trees, the decision was made to utilize trees of the first type to create the HMT predictor to be used in the control.

The manner in which the data is manipulated is covered in the next section relating to the application packages in question, See5® and Cubist®.

3.2 Software Details

The software used to create the predictive models were the See5® and Cubist® applications created by RuleQuest® Technologies. See5® is a sophisticated data-mining tool for discovering patterns that delineate categories, assembles them into classifiers, and uses them to make predictions. Some of its more important features include the fact that See5® has been designed to analyze substantial databases containing thousands to hundreds of thousands of records and tens to hundreds of numeric or nominal fields. In addition, to maximizing interpretability, See5® classifiers are expressed as decision trees or sets of if-then rules, forms that are generally easier to understand than neural network based predictors.

In effect, See5®'s job is to find how to predict a case's class from the values of the other attributes. See5® does this by constructing a classifier that makes this prediction. See5® can construct classifiers expressed as decision trees or as sets of rules.

Cubist, on the other hand, is a powerful tool for generating piecewise-linear models that balance the need for accurate prediction against the requirements of intelligibility. Cubist® models generally give better results than those produced by simple techniques such as multivariate linear regression. Some important features of Cubist® include its ability to analyze substantial databases containing thousands of records and tens to hundreds of numeric or nominal fields. In addition, to maximizing interpretability, Cubist® models are expressed as collections of rules, where each rule has an associated multivariate linear model. Whenever a situation matches a rule's conditions, the associated model is used to calculate the predicted value.

To sum up, Cubist® is a tool for generating rule-based predictive models from data. Whereas See5® produces classification models that predict categories, Cubist's models are numeric, i.e., they generate values. For instance, See5® might classify the yield from some process as "high," "medium," or "low," whereas Cubist® would output a number such as 73%. (Statisticians call the first kind of activity "classification" and the second as "regression.")

As for the issue of preparing the data sets to be used by these two applications, it is important to note that the raw data available is in a format in which it relays continuous values for the HMT. However, the application to be utilized to structure the decision tree predictor, See5®, requires that the output variable to be predicted is classifiable within certain categories. As such, the current format of the data would have to be manipulated in such a manner as to convert the continuous values of the output variable available within the data set into a number of ranges that the predictor would classify a specific data entry into. The available HMT temperatures within the training data set to be used by See5® ranged from 1400 degrees Celsius to 1510 degrees Celsius. To satisfy the requirements of See5®'s data characteristics, the output variables were replaced by eleven ten degree band ranges commencing with '1400 up to 1410 degrees Celsius' and ending with '1500 up to 1510 degrees Celsius.' Thus, the data was prepared for use by the See5® application as a training data set upon which the control mechanisms prediction tree and rule sets were to be based.

In the case of Cubist, the data requirements dictate that the output variable should be continuous, as was the HMT value within the data set. Thus, no apparent conflicts were found to exist between the already available raw data set structure and that of the data set structure for the data to be utilized by the Cubist® application.

3.2.1 Analysis of See5® and Cubist® under Various Settings

Inherent to the applications to be used to structure the control mechanism for the HMT of the blast furnace are certain settings that affect both the error levels associated with the predictions of a predictor formulated by See5® and Cubist, and the time taken to structure these predictors. The most important user-controlled setting is that of the boosting level of the application. Boosting is a technique for generating and combining multiple classifiers to give greatly improved predictive accuracy.

In boosting, a number of classifiers are created instead of an individual one. The training set, the set of input and output data upon which the predictor is structured, chosen at any point depends on the performance of earlier classifiers. Cases that are incorrectly predicted are chosen more often than correctly predicted cases. This leads to the structuring of classifiers that cater to these incorrectly predicted cases in particular. These classifiers complement those that correctly predict the other cases in the data set, thus increasing overall reliability and accuracy of the predictor. When a new case is to be classified, each classifier votes for its predicted class and the votes are counted to determine the final class.

See5® includes support for boosting with any number of trials. Naturally, it takes longer to produce boosted classifiers, but boosting can be worth the additional time when peak predictive accuracy is required. The boost setting process determines the number of classifiers that the application will create based on the data set in question. For example, a boost setting of 3 will create three classifiers whereby the subsequent classifiers will tend to be formulated based on cases which are incorrectly predicted in previous classifiers. As the first step, a single decision tree or rule set is constructed as from the training data. This classifier will usually make mistakes on some cases in the data; the first decision tree, for instance, gives the wrong class for 343 out of 2031 test cases. When the second classifier is constructed, more attention is paid to these cases in an attempt to get them right. Consequently, the second classifier will generally be different from the first. It also will make errors on some cases, and these become the focus of attention during construction of the third classifier. This process continues for a pre-determined number of iterations.

As shall be discussed later in the paper, there is a need for an accurate and high-speed mechanism for structuring the underlying predictor upon which the HMT control system is to be structured. Analysis of the trade-off between the time taken to structure the predictor and its predictive accuracy to find the optimal boost setting was carried out on a large sample data set with the following results.

The following table, Table 2, is a summary of the error levels and the necessary run-time of the See5® application associated with various boost settings. The boost settings were chosen to span the range of 0, i.e., no boosting, to 20, i.e., twenty classifiers were made instead of one.

Evaluation Summary for See5® performance

	Settings for See5®	>>> >>>	Evaluation on training data (2031 cases)				
	Boost setting		Decision Tree	DT	Rules	R	Time
Case #			Errors / cases	Errors / %	Errors / cases	Errors / %	/s
1	0		343	16.90%	541	29.10%	10
2	1		343	16.90%	509	29.10%	10.1
3	2		281	13.80%	383	18.90%	19.1
4	3		165	8.10%	250	12.30%	27.3
5	4		111	5.50%	157	7.70%	35.9
6	5		73	3.60%	109	5.40%	43.7
7	6		53	2.60%	64	3.20%	54.5
8	7		35	1.70%	50	2.50%	63.4
9	8		21	1%	33	1.60%	72.7

10	9	20	1%	24	1.20%	81.1
11	10	8	0.40%	18	0.90%	90.3
12	13	5	0.20%	7	0.30%	116.4
13	15	5	0.20%	3	0.10%	134
14	20	4	0.20%	1	0%	177.2

Table 2: Summary of error levels and run-times observed throughout utilization of See5® under varying boost settings.

The following figures depict the above data in graphical format. Figure 2 depicts the variation of the boost setting with the resulting error level. This is equal to the number of wrongly classified cases by the predictor, upon creation of the decision tree and rule set, divided by the total number of cases within the data set the predictor is being trained on, i.e., from which the decision tree is being created.

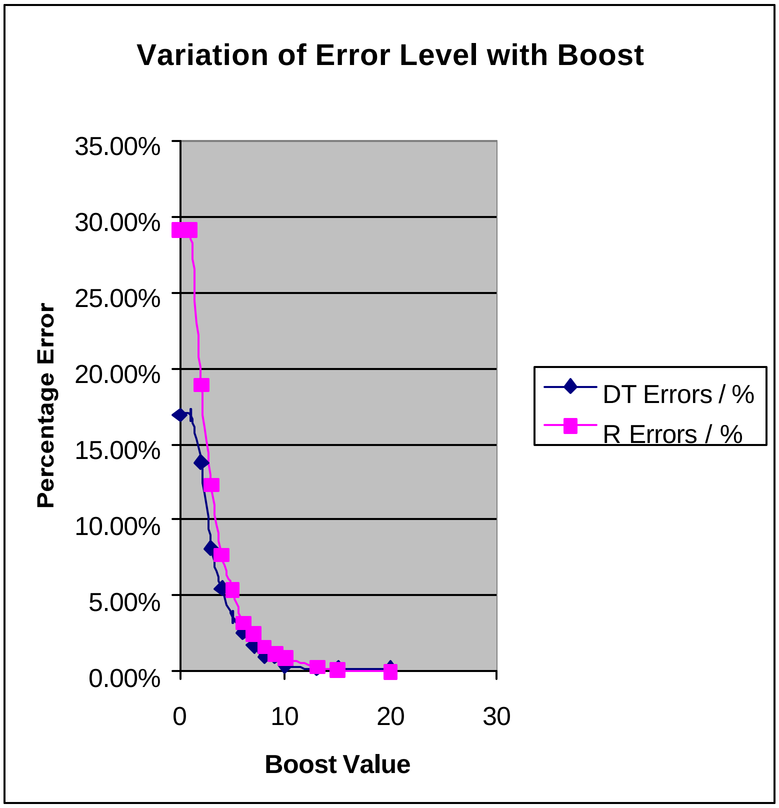

Figure 2: A graph illustrating the error levels associated with running See5® to create a predictor, based on the training data set, under varying boost settings.

Figure 3 depicts the variation of the time taken to structure the decision trees and rule sets based on the available data set associated with the different boost settings.

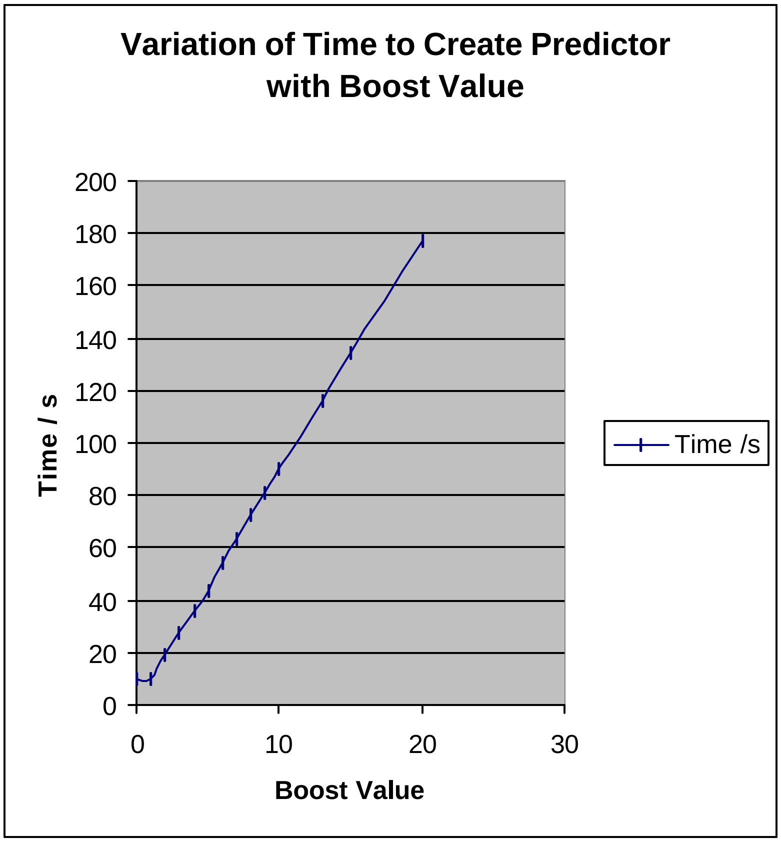

Figure 3: A graph illustrating the run-time observed when running See5® to create a predictor, based on the training data set, under varying boost settings.

Upon observation of the graphs, it is clear to see that the error level of the resulting decision trees and predictors decreases greatly throughout the first few boost settings. The error level decreases from 29.10 % at a boost setting of 0 to .90% at a booster setting of 10 for rule set creation, and from 16.9% to .40% within the same booster setting range for decision tree creation. The error level subsequently tapers off towards 0% as the boost settings increase.

However, accompanying this decrease in error level is an increase in the time taken to form the aforementioned decision trees and rule sets. The time taken to formulate these two predictive models increases nine-fold within the range of 0 to 10 analyzed in the error level domain in the paragraph above.

Finding the optimal 'down-time' (time to formulate a predictor) and error level tolerance combination based on this apparent trade-off depends on the desired predictive accuracy and whether there is an acceptable margin of error tolerated by the users of the control mechanism. In addition, one may question whether it is worth attaining increasingly accurate predictions at minute increments at a cost of increasing the additional 'down-time' taken on by the controller at a greater rate. This can be seen in the case of comparing the changes in error levels of the formulated decision trees and rule sets and the time taken to formulate these predictors at different stages on the above graphs. For instance, there is an observed drop in the error level from 29.10% to 5.40% in the formulated rule set when increasing the boost setting from 0 to 5. This decrease is much larger than the decrease in the error level in the formulated rule set from 5.40% to .90% when the boost setting is increased from 5 to 10. The error level decreased by close to a factor of 5 times more in the first range of boost settings when compared to the decrease over the second range of boost settings. However, the time increase over these ranges, 33.7 s (10 s to 43.7 s) and 46.6 s (43.7 s to 90.3 s) respectively has increased at an increasing rate over the span of five boost settings. This reveals that, at higher boost settings, in a greater amount of time, the error level has decreased by a smaller amount than at lower boost settings. This matter would have to be taken into account by the supervisors of the blast furnaces functionality to determine the optimal level according to their respective time constraints.

3.3 Alternative path proposal mechanism

This section deals with the procedure by which alternative alterations to the current input variables proposed to the HMT controller are generated for consideration as potential courses of action to reach the desired HMT from the current HMT. This procedure is covered in two basic steps, the first of which is to calculate correlations between the varying input variables under observation and the HMT. The second relays the temperature-shift brackets, which are structured, based on these correlations within the context of minimizing the cost of the path to be taken to attain the desired HMT. These are discussed in detail in the following two sub sections.

3.3.1 Finding Correlations between input variables and the HMT using Cubist

The goal behind finding the correlations existent between the various input variables and the HMT using Cubist® is that of being able to predict the relationship that exists between the two utilizing a predictive model rather than a simple linear regression model. In addition, Cubist® was designed to function effectively on substantial data sets, thus resulting in greater predictive accuracy when structuring the predictive model based on the training set at hand.

To obtain the necessary correlations, a method had to be found to ensure that the structured rules encompassed as many rules as possible, if not all of them. This had to be done to guarantee the peak generality of these correlations to all the training data channeled into Cubist®.

To gauge the number of cases covered by the predictor generated by Cubist, one can alter, through the Cubist® interface, the minimum case cover for any rule as a percentage of the number of training cases. That is, the conditions associated with any rule should be satisfied by at least the specified percentage of all training cases. The minimum case cover is not treated as a rigid constraint; in the process of fine-tuning models, Cubist® sometimes allows rules to cover fewer cases than the specified percentage would require. The default value is 1%; if this option is set to a higher value, Cubist® is encouraged to produce models with fewer, more general rules.

To ensure maximum rule coverage, this variable was set to 100%. The following is a summary of the results obtained upon running Cubist® on the available training set of 2031 cases:

Cubist® [Release 1.07]

Options:

Each rule must cover >=100% of cases

Target attribute `v_out'

Read 2031 cases (18 attributes).

Model:

Rule 1: [2031 cases, mean 1457.2202, range 1406.805 to 1507.014]

predicted_hmt = -110690 - 23.3 v_3 + 5.94 v_12 + 3.81 v_13 - 12.4 v_2 + 13.1 v_1 - 0.327 v_7 + 2.52 v_15 + 5170 v_6 - 5.1 v_16 - 1.16 v_5 - 18.6 v_11 - 0.88 v_14

Average |error| 6.2840 %

Relative |error| 0.34 %

Correlation coefficient 0.92

The final part of the output describes the performance of the model on the training data, and on the new cases in the test data (if present). The average error magnitude is the error incurred by using the predictor on the test cases within the training set itself. This is found to be 6.284 % in the case of this predictor indicating that this predictor is accurate in forecasting a large proportion of the cases within the data set upon running the predictor on the same data set after it was formulated.

The relative error magnitude is the ratio of the average error magnitude to the error magnitude that would result from always predicting the mean value. It is indicated within the Cubist® application package that this should be less than 1.0 if the model is useful. The value of the relative error of this predictor is .34 % in this case, relaying the reliability of this model on the measure of usefulness dictated by the previous statement. The correlation coefficient measures the agreement between the cases' actual values of the output variable, the HMT in this case, and those values predicted by the model. The correlation coefficient of .92 indicates a high level of agreement between predicted values of the HMT based on the prediction model and the actual values of the HMT included within the training set.

The rule generated by the predictor is:

predicted_hmt = -110690 - 23.3 v_3 + 5.94 v_12 + 3.81 v_13 - 12.4 v_2 + 13.1 v_1 - 0.327 v_7 + 2.52 v_15 + 5170 v_6 - 5.1 v_16 - 1.16 v_5 - 18.6 v_11 - 0.88 v_14

where predicted_hmt is the HMT value in degrees Celsius to be predicted by the model, and the variables v_1 through v_16 are the values of the input variables that dictate the HMT within the blast furnace.

In this particular case, the correlation between the HMT within the furnace and the input variables can be summarized as follows on the basis that a change in an input variable will result in a change in the HMT in accordance with the prediction model:

Since the prediction model to be created by See5® classifies the predicted output within ranges of 10 degrees Celsius, it would be important to study the available options to increase the HMT within the furnace by 10 degrees Celsius, i.e., alter input variables to shift the HMT from one classification (10-degree band) to another:

1. Decrease v_3 by .43 (10 divided by 23.3) units of v_3,
2. increase v_12 by 1.68 (10 divided by 5.94) units of v_12,
3. increase v_13 by 2.62 (10 divided by 3.81) units of v_13,
4. decrease v_2 by .81 (10 divided by 12.4) units of v_2,
5. increase v_1 by .76 (10 divided by 13.1) units of v_1,
6. decrease v_7 by 30.6 (10 divided by .327) units of v_7,
7. increase v_15 by 3.97 (10 divided by 2.52) units of v_15,
8. increase v_6 by 1.93 * 10^3 (10 divided by 5170) units of v_6,
9. decrease v_16 by 1.96 (10 divided by 5.1) units of v_16,
10. decrease v_5 by 8.62 (10 divided by 1.16)units of v_5,
11. decrease v_11 by .538 (10 divided by 18.6) units of v_11,
12. decrease v_14 by 11.36 (10 divided by .88) units of v_14.

However, some of these options may be extremely difficult to carry out or may be simply infeasible within the context of the furnace. The degrees to which one will be able to carry out such variations in the input variables is a subjective matter that falls in the hands of the furnace supervisor. A supervisor's judgment will evaluate the feasibility of these options singling out the few that are executable to attain the end goal of altering the HMT within the furnace. For the sake of analysis, the variables v_12, v_13 and v_16 have been singled out as the most easily altered variables. This could be determined in practice by determining which of the variables is characterized by easily separable and divisible units of the substances or factors governing the values of these variables.

3.3.2 Defining and selecting appropriate temperature-shift brackets

The second path-generating element that functions in conjunction with the cost-minimizing goal of the HMT controller is that of temperature-shift brackets. A temperature-shift bracket is a number of potential alterations to the HMT input variables grouped within a single category based on the temperature-shift expected as the outcome of such alterations. The alternatives encompassed within a specific temperature-shift bracket are arranged in the order of the cost that would be incurred by following through with that alteration. The alterations undergoing this classification encompass all those alterations that would bring on shifts in the HMT of ten degrees Celsius. The span of these shifts is determined by the width of the classification bands of the records within the data set. Since the bands chosen earlier span ten degrees Celsius, the only observable changes in the HMT with the predictor to be structured by See5® would be those spanning ten or more degrees Celsius. In the case that smaller changes need to be observed, the output variable within the training set passed into See5® could be categorized into band ranges of a smaller range such as five degrees or any other appropriate band width.

An example of a temperature-shift bracket structure using the three aforementioned variables, v_12, v_13 and v_16 follows:

To increase the HMT within the blast furnace by ten degrees Celsius one could:

1. increase v_12 by 1.68 (10 / 5.94) units of v_12,

2. increase v_13 by 2.62 (10 / 3.81) units of v_13,

3. decrease v_16 by 1.96 (10 / 5.1) units of v_16.

Assume the cost of a unit of:

1. v_12 = $5,
2. v_13 = $10,
3. v_16 = $15.

Temperature-shift bracket structure:

Rank of Alternative	**V_12**	**V_13**	**V_16**	**Cost / $**
2	Increase by 1.68			Increase of 8
3		Increase by 2.62		Increase of 26.2
1			Decrease by 1.96	Decrease of 29.4

Table 3: Summary of the various input variable alterations that are predicted to result in an increase in the HMT within a blast furnace of ten degrees Celsius.

Rank of alternative	**V_12**	**V_13**	**V_16**	**Cost / $**

4	Increase by 3.36			Increase of 16
6		Increase by 5.24		Increase of 52.4
1			Decrease by 3.92	Decrease of 58.8
5	Increase by 1.68	Increase by 2.62		Increase of 34.2
2	Increase by 1.68		Decrease by 1.96	Decrease of 21.4
3		Increase by 2.62	Decrease by 1.96	Decrease of 3.2

Table 4: Summary of the various input variable alterations that are predicted to result in an increase in the HMT within a blast furnace of twenty degrees Celsius.

The two temperature-shift brackets above relay the available alternative alterations in input variables that can be carried out to achieve ten degree (Table 3) and twenty degree (Table 4) increases in the HMT within a blast furnace (in accordance with the prediction of the model generated by Cubist® on the training data set) respectively. All permutations resulting in the desired effect of shifting the HMT are included within the temperature-shift brackets.

The question arises as to why these brackets are necessary instead of simply taking the lowest cost method or conversely, the highest ranked method, as the default course of action to be taken in that this course will minimize the cost incurred while achieving the desired result of shifting the HMT. The need for these temperature-shift brackets is a result of constraints that may be placed on the system at hand. For instance, if a constraint stated that the amount to be removed of the substance governing the value of variable v_16 should not exceed 3 units, this would dictate that removing 3.92 units from an additive mixture to the furnace would be out of the question or inadvisable. As such, this alternative would not be considered as a viable alternative to achieve the desired goal of raising the HMT by 20 degrees Celsius. Thus, it is necessary to include as many alternatives as possible with the variables to be utilized to incur HMT shifts. This will ensure that some viable alternatives do exist under the constraints imposed on the blast furnace's inputs.

Section 4: Control functioning – from details to an overview of the whole process

This section deals with the actual functioning of the HMT controller, bringing together the previously covered specifics within the same scope. This section, in effect, elaborates on how the discussed pieces of the puzzle fall into place to create a controller that can be used to effectively propose least-cost courses of action to alter the HMT within a blast furnace to attain the goal of reaching another desired HMT. The first sub-section will cover the step by step mechanism by which the controller functions while the second will cover the constant learning mechanism inherent to this mechanism to further develop and improve its functioning with every subsequent run of the controller.

4.1 Creation of controller & Step by step functioning methodology of control mechanism

Step 1: Run Cubist® on the training data set and deduce the correlations that exist between the input variables and the HMT based on rules generated by the application.

Step 2: Structure temperature-shift brackets based on the correlations determined in the previous step and the cost of the various input variables.

Step 3: Enter constraints governing the input variables into the controller.

Step 4: Reformat data set to convert continuous values of output variable, i.e., HMT, into ranges (HMT bands) to enable See5® classification of records within data set.

Step 5: Run See5® on the reformatted data set to generate rule-based and decision tree predictors.

Step 6: Take in both current HMT and desired HMT and evaluate desired HMT shift. This will dictate which temperature-shift band the controller will look to for potential alternative proposals in input variables to achieve desired HMT shift. For example, if an increase in the HMT of 30 degrees Celsius is desired and the classification (HMT) bands within the data set are 10 degrees in width, then the 30-degree temperature-shift bracket will be examined to locate the optimal course of action to be taken on to attain the target HMT.

Step 7: Locate lowest-cost alternative (lowest rank alternative within temperature-shift bracket) within the previously specified temperature-shift bracket and check to see whether or not this alternative, when carried out on the current values of input variables, satisfies the underlying constraints entered in step 3.

Step 8.1: If these constraints are satisfied, then run the newly formed data record through the predictor and check to see whether or not predicted HMT falls within the desired HMT range.

Step 8.2: If these constraints are not satisfied, then discard the alternative chosen from the temperature-shift bracket for this instance of the controller's functioning and return to step 7.

Step 9.1: If the predicted HMT falls within the desired HMT range, then propose the current set of alterations under examination as the least-cost, constraint-compliant course of action that will lead to achieving the target HMT in the blast furnace.

Step 9.2: If the predicted HMT does not fall within the desired HMT range, then discard the alternative chosen from the temperature-shift bracket for this instance of the controller's functioning and return to step 7.

It is also important to understand the functioning of the controller within the context of the learning mechanism outlined below to better comprehend the overall methodology of the control process and how it is continually improved with time.

4.2 Ongoing data set updating and control learning mechanism

The controller's learning mechanism is utilized to further enhance the predictive accuracy of the underlying prediction models by appending recently generated data records to the training data set is to be structured. This is carried out in the following manner:

Step 1: Run through the control process using the currently available training data set.

Step 2: Record the current values of the input variables used to predict the HMT in a new data record.

Step 3: Once the optimal set of input variable alterations is chosen, execute these alterations on the factors governing the values of the altered variables.

Step 4: Alter the data record created in step 2 to reflect the alterations executed on the input variables.

Step 5: Append the resultant HMT as the output variable within the data record created in Step 2; add this new complete data record to the currently existent training set, and return to step 1.

Conclusion

This paper examined a proposed method by which to construct a control mechanism for the HMT within a "G" blast furnace at Steelcorp using decision tree based predictors. While searching for the optimal control structure, analysis of several important issues was undertaken, resulting in proposed solutions to the various data issues and structural problems that arose. The first goal of the paper was to deduce the control structure that would prove to entail the most effective utilization of the data set at hand. This encompassed analyzing the various decision tree structures that the prediction-formulating application could create and selecting the structure that best suited the control mechanism's purposes. This was found to be a tree structure of which the origin would equal a zero-state containing no data relaying the state of the blast furnace. Each relevant input variable would be taken in by the tree at every subsequent tree level. The leaves of the tree would relay the predicted HMT resulting from following a specific path down the tree.

This was followed by an analysis of the application to be used to structure the decision tree based predictor, See5®. This elucidated the trade off between the boosting level used to structure the predictor and the time taken to do so. Conclusions from the analysis indicated that the greater the boosting level, i.e., the greater the predictive accuracy of the predictor to be formulated by See5®, the more time would be needed to structure the predictor, especially if minimal tolerance for error is a prerequisite for the functioning of the control.

The next issue was that of the methodology by which potential alternatives would be chosen for analysis by the control mechanism, from which to propose the optimal path of alterations to be followed to attain the desired HMT within the furnace. The need to structure such paths to enable for the governance of constraints on any alterations to be carried out within the blast furnace, in addition to maintaining cost-levels of proposed alterations at a minimum, resulted in defining temperature-shift brackets. These brackets entailed the grouping of potential alteration paths to the current blast furnace state according to the proposed shift in HMT that each set of alterations would result in, as concluded by performing correlation analysis between input variables and the HMT using Cubist®. The alteration paths within each of these brackets are ranked according to the cost of each set of alterations to ensure minimizing the cost of proposed alterations. This is carried out by means of proposing paths within a bracket to the control in order of cost, after maintaining that the path in question falls within the constraints set on potential alterations to be carried out within the furnace.

The final topic covered within the realm of structuring an HMT control is that of a continual learning mechanism. This mechanism is set up such that the decision tree based predictor to be utilized by the control is continually updated as the control is put to use. The proposed method to carry this out entailed a feedback system by which the results of implementing alteration paths proposed by the control would be appended to the then existing data set, upon which the predictor would be reformulated using this updated data set.

In conclusion, this paper entails a proposal, in theory, for the structure of an HMT control within a steel blast furnace. The analyses incorporated within this paper would be best suited for review, as guidelines, in the effort to implement the control.

References

[1] Berry, Michael and Linoff, Gordon. Data-mining Techniques, Wiley Computer Publishing, New York. 1997.

[2] Bulsari, Abhay and Saxen, Henrik. "Classification of blast furnace probe temperatures using neural networks." Steel Research. Vol. 66. 1995.

[3] Bulsari, Abhay and Saxen, Henrik and Saxen, Bjorn. "Time-series prediction of silicon in pig iron using neural networks." International Conference on Engineering Applications of Neural Networks. 1992.

[4] Decision Trees, site: http://dsg.harvard.edu/courses/hst951/ClassificationTree.

[5] Elvers, Barbara. Ullman's Encyclopedia of Industrial Chemistry. John Wiley and Sons, New York. 1996.

[6] Kirk, Othmer. Encyclopedia of Chemical Technology. Volume 13. John Wiley and Sons, New York. 1981.

[7] Lida, Osamu and Ushijima, Yuichi and Toshiro, Sawada. "Application of AI techniques to blast furnace operations." Iron and Steel Engineer. October 1992.

[8] Lu, Yong-Zai. "Meeting the challenge of intelligent system technologies in the iron and steel industry." Iron and Steel Engineer. September 1996.

[9] RuleQuest® Research Data-mining Tools, site: http://www.RuleQuest.com.

[10] Singh, Himanshu and Sridhar, Nallamali and Deo, Brahma. "Artificial neural nets for prediction of silicon content of blast furnace hot metal." Steel Research, Vol. 67. 1996.

[11] Winston, Patrick. Artificial Intelligence. Addison-Wesley Publishing Company, New York. 1993.