A Mill Set-up Model Using a Multi-output Regression Tree for a Tandem Cold Mill Producing Stainless Steel

Hyun-Seok KANG^{1,2)} and Chi-Hyuck JUN^{3)*}

1) Technical Research Laboratories, POSCO, 6261 Donghaean-Ro, Nam-Gu, Pohang, 37859 Republic of Korea.

2) Graduate Institute of Ferrous Technology, Pohang University of Science and Technology (POSTECH), 77 Cheongam-Ro, Nam-Gu, Pohang, Gyeongbuk, 37673 Republic of Korea.

3) Department of Industrial and Management Engineering, Pohang University of Science and Technology (POSTECH), 77 Cheongam-Ro, Nam-Gu, Pohang, Gyeongbuk, 37673 Republic of Korea.

(Received on November 16, 2018; accepted on March 25, 2019; J-STAGE Advance published date: June 14, 2019)

In a tandem cold mill for stainless steel, an optimum reduction rate is necessary for each stand. A conventional mill set-up uses a lookup-table to optimize the rolling schedule. However, to reflecting all the input conditions and manual interventions on a model is difficult. In this paper, we propose a mill set-up model that can efficiently predict the reduction rate for each stand by considering various input conditions. The proposed prediction model has a multi-output tree structure with a smaller time complexity for easy interpretation. The key contribution to the proposed algorithm is variable selection. According to the results of an analysis of the time-complexity, the proposed algorithm is less time consuming and is capable of learning datasets with a large number of variables more efficiently than the single-output CART (classification and regression trees). To evaluate the performance of the proposed algorithm, we applied it to the rolling reduction rate of a tandem cold mill in POSCO. The proposed algorithm achieves a similar level of R-squared in only 18% of the computing time required for an existing single-output CART algorithm.

KEY WORDS: stainless steel; cold rolling; mill set-up; multi-output regression tree.

1. Introduction

To increase productivity in the cold rolling of stainless steel, a tandem cold mill is commonly used. The general trend has been for the number of rolling mill stands to increase, with a few Chinese companies operating mills with more than five stands. Such configurations allow rolling mill companies to produce cold-rolled strips of stainless steel with higher productivity.

It is important to set up an optimum reduction rate for each stand in the tandem cold rolling of stainless steel. If the reduction rate of any one stand is excessive, it will cause surface defects, such as heat streaks.¹⁾ To prevent such defects in stainless steel cold-rolled plates, for which the surface quality of the material is important, the optimum reduction rate should be determined using a mill set-up model. Many studies^{2–7)} have examined the setting up of the reduction rate using optimization techniques. There are several classifications for optimization techniques: 1) nonlinear programming methods,²⁾ 2) GA algorithms,³⁾ or 3) Simplex optimization.^{4–6)} For the objective functions in optimization, previous studies have used uniform rolling load, motor power, shape, and flatness as the cost function.

One feature of tandem rolling is that each stand attains a high reduction rate because it is required to produce the target thickness with a limited number of rolling stands. The tandem rolling of stainless steels attains a relatively high rolling reduction rate, but large amounts of heat caused by friction and plastic deformation are generated. The high temperature causes the film to become thinner,^{8,9)} which increases the probability of incurring heat-streak defects.¹⁰⁾ Also, any roughness of the roll surface can give rise to defects.¹¹⁾ In other words, the optimization of the rolling reduction rate of a tandem cold mill for stainless steel should consider not only the rolling force balance between the stands, but also the temperature of the material, the film thickness, and the roughness of the rolls. All these cost functions are very complicated and difficult to implement in optimization algorithms. In actual rolling, the mill set-up model depends on the operators' experience and their manual intervention. In this paper, we propose a reduction rate set-up model that learns operators' experience from historical data.

The following sections in this paper are as follows. Section 2 describes the problems associated with a conventional mill set-up model and introduces the concept of a mill set-up learning model. Section 3 describes the various approaches to learning and explains the justification of the proposed learning model structure. Section 4 presents the extension of

* Corresponding author: E-mail: chjun@postech.ac.kr

DOI: <https://doi.org/10.2355/isijinternational.ISIJINT-2018-770>

the single-output classification and regression trees (CART) to a multi-output CART and proposes an efficient algorithm to learn reduction rate of mill stand. Section 5 describes the datasets for learning the reduction rate of the actual rolling mill and analyzes the results. Finally, Section 6 presents the main conclusions.

2. Mill Set-up Models

2.1. Conventional Mill Set-up Model

Figure 1 is a schematic representation of the mill set-up model for a four-stand tandem mill. The target rolling reduction rate of each stand is based on the cold-rolling conditions (material type, target exit thickness, total reduction rate, *etc.*). Generally, the entry and exit thicknesses of the rolling mill are known variables. As such, the problem of setting up the reduction rate of each stand reduces to a problem of optimizing the target thickness between the rolling stands. For the cold rolling of stainless steel, we should consider the temperature of the material, the thickness of the oil film, and the surface roughness of the rolls in order to prevent surface defects such as heat streaks during the cold rolling. However, it is very difficult to reflect the above conditions in a mill set-up model for actual rolling.

In actual rolling, the algorithm for setting up the reduction rate is as follows; in this case, we consider a four-stand tandem cold mill.

First, the ratio of the reduction rates between stands is determined either empirically or experimentally, where r_i is the reduction rate for the set-up at the i -th stand and r_i' is the table value (Eq. (1)).

$$r_1 : r_2 : r_3 : r_4 = r_1' : r_2' : r_3' : r_4' \dots\dots\dots (1)$$

The target reduction rate for each stand is defined by Eq. (2), where h_i is the exit thickness at the i -th stand and H_i is the entry thickness at the i -th stand. Equation (3) shows the relationship between the entry and exit thicknesses at the neighboring stands.

$$r_i = \frac{H_i - h_i}{H_i}, i = 1, 2, 3, 4 \dots\dots\dots (2)$$

$$H_{i+1} = h_i, i = 1, 2, 3 \dots\dots\dots (3)$$

Then, we can determine the relationship between the entry and exit thicknesses (Eq. (4)). Finally, the thickness of the rolled steel can be expressed as the reduction ratio and

entry thickness of the rolling mill (Eq. (5)).

$$h_i = (1 - r_i) H_i, i = 1, 2, 3 \dots\dots\dots (4)$$

$$h_4 = (1 + r_1)(1 + r_2)(1 + r_3)(1 + r_4) H_1 \dots\dots\dots (5)$$

There are four unknowns ($r_i, i = 1-4$) in these equations, and we can establish three additional equations using the table values in Eq. (1).

$$r_2 = \frac{r_2'}{r_1'} r_1 \dots\dots\dots (6)$$

$$r_3 = \frac{r_3'}{r_1'} r_1 \dots\dots\dots (7)$$

$$r_4 = \frac{r_4'}{r_1'} r_1 \dots\dots\dots (8)$$

By substituting Eqs. (6), (7), and (8) into Eq. (5), we can form a fourth-order polynomial equation of r_1 , given as Eq. (9), and this can be solved by applying a means of numerical analysis, such as the Newton-Raphson method. Then, the remaining reduction rate can be obtained from Eqs. (6), (7), and (8)

$$h_4 = (1 + r_1) \left(1 + \frac{r_2'}{r_1'} r_1 \right) \left(1 + \frac{r_3'}{r_1'} r_1 \right) \left(1 + \frac{r_4'}{r_1'} r_1 \right) H_1 \dots\dots (9)$$

In the above formula, the most important variable in the reduction rate set-up is the reduction ratio between the rolling stands. These values are usually determined from a lookup-table. However, this incurs several problems. First, when the number of input conditions increases, the lookup-table becomes excessively large. For example, if the input condition is divided into 10 steel types, 10 entry thickness groups, and 10 exit thickness groups, 1 000 ($10 \times 10 \times 10$) lookups are generated, and it would be difficult to determine all these values experimentally. In addition, it is difficult for all the process conditions (such as annealing temperature and time) that affect the material properties to be reflected in the lookup-table. This is not only because the structure of the lookup-table has to be modified, but also because the size of the lookup table becomes proportionally greater due to the inclusion of the new conditions. In addition, the operators' manual intervention is not reflected in the lookup table. Realistically, a lookup table can never be optimal for all actual rolling conditions, so we have to rely on manual intervention by the operators under certain conditions. However, since the lookup-table method is determined only by the rolling conditions, the degree of manual intervention cannot be reduced. This leads to variations among the operators.

2.2. A New Mill Set-up Model

To overcome the disadvantages of the existing mill set-up model, we propose a new alternative. **Figure 2** is a schematic representation of the structure of this new model. A reduction rate prediction model replaces the lookup table. This prediction model can take previous process data as inputs as well as existing rolling data. Then, we predict the actual reduction rate by reflecting the manual intervention. It is possible to calculate the reduction ratio between stands based on the predicted reduction rate at each stand.

Fig. 1. Schematic of conventional mill set-up model. (Online version in color.)

Fig. 2. Schematic of new mill set-up model. (Online version in color.)

This method can reflect the previous process data onto the set-up model, such that we expect that the model would be able to learn the reduction rate from the operator's experience online. As a result, the degree of manual intervention would decrease.

3. Multi-output Prediction Models

3.1. Local and Global Approaches

The newly proposed reduction rate learning model can give rise to multi-output regression problems. There are two main approaches to multi-output regression.¹²⁾ The first is the local method (Fig. 3(a)), which predicts the reduction rate of each stand independently. The second is the global method (Fig. 3(b)), which simultaneously predicts the reduction rates of all the stands. In the former case, a prediction model is constructed for each stand, and the output of each model has a scalar. In the latter case, a single model covers all the stands, and the output from the model has a vector in which the reduction ratio of each stand is a component.

The local method can utilize many existing algorithms, but the computation time can be longer than the global method. In addition, the local method cannot reflect the dependency between stands. However, the global method offers the advantage of being able to consider the dependency of the reduction rates between the stands at the same time.

3.2. Model Types

The different types of multi-output prediction models can be categorized as shown in Table 1. A linear regression model consists of a linear combination of inputs and model coefficients.¹³⁾ Linear models are relatively easy to interpret, but even with a large number of input variables, this is not easy to achieve. In general, the performance is limited because the model does not reflect the nonlinearity. To reflect nonlinearity in the model, therefore, we can consider either a kernel method or a neural network model. In this case, however, the interpretation of the model is difficult because it requires the use of a nonlinear kernel function or activation function. When a black-boxed nonlinear model for the reduction rate prediction, it is difficult to interpret the model in actual implementation. One way of providing a white-boxed model for machine learning is to use a tree-

Fig. 3. Two types of multi-output regression. (a) Local method, (b) Global method.

Table 1. Model characteristics of multi-output regression.

Model	Characteristics
Linear regression models	Easy interpretation
	Low model flexibility and predictive accuracy
Support vector regressions (SVR)	Easy interpretation, low model flexibility (Linear kernel function)
	Hard interpretation, high model flexibility (Non-linear kernel function)
Regression trees	Very easy interpretation
	Fast for learning and prediction
	Low on memory usage
Neural networks	Hard interpretation
	High model flexibility
	Hard to optimize the hyperparameters

Fig. 4. Example of multi-output regression tree.

structured model.¹⁴⁾ The regression tree is a simple tree-like structure such as $x_i < c$, where x_i is a specific variable, and c is a split. This is as easy to interpret as the reduction ratio table. In this paper, we propose the multi-output, tree-structured model that would enable easy interpretation. Figure 4 shows an example of constructing a tree model in a global method. The data in the top node is divided into

two groups, with each node minimizing the variance of the divided data. At the end node, the average value of the output of the divided data is sent to the output.

If we assume that the training data set D consists of N observations of input variable vector and output variable vector, $D = \{(x^1, y^1), \dots, (x^N, y^N)\}$. The l -th observation of the input variable vector is an m -dimensional vector $x^l = (x_1^l, \dots, x_m^l)$ and the l -th observation of the output variable vector is a d -dimensional vector, $y^l = (y_1^l, \dots, y_d^l)$.

4. Proposed Algorithm for New Mill Set-up Model

4.1. Multi-output CART

CART is one of the most widely used algorithms.¹⁵⁾ This method is used for single-output prediction and/or classification. The algorithm searches for a variable and a split to partition the data recursively, so as to reduce the impurity function as the tree grows. CART uses the Gini index for the impurity function in a classification problem, while the mean squared error (MSE) is used in the case of regression. The prediction of the reduction rate in our case is a regression problem, so we use MSE for the impurity function.

Suppose that the data set at node m (Q_m) is partitioned into two sub-nodes (Q_{left} , Q_{right}) by a split, as shown in Fig. 5. Also, Q_m , Q_{left} , Q_{right} have N_m , n_{left} , n_{right} observations, respectively. The impurity function at node m before the split is calculated as:

$$Impurity(Q_m) = \sum_{j=1}^d \frac{1}{N_m} \sum_{l \in Q_m} (y_j^l - \bar{y}_j)^2, \dots (10)$$

where

$$\bar{y}_j = \frac{1}{N_m} \sum_{l \in Q_m} y_j^l \dots (11)$$

The decrease in the impurity function after applying the split s is expressed by

$$\Delta Impurity(s) = Impurity(Q_m) - \left[\frac{n_{left}}{N_m} Impurity(Q_{left}) + \frac{n_{right}}{N_m} Impurity(Q_{right}) \right] \dots (12)$$

Here, we need to find the best split s to maximize $\Delta Impurity(s)$.

Fig. 5. Data partitioning at node m (Q_m) using CART algorithm. (Online version in color.)

The CART algorithm procedures are as follows. First, we sort for all continuous and ordinal x variables. Next, we exhaustively find the variable and split which minimize the impurity function. Finally, we divide the data into the derived variable and split. If the stopping criterion is not satisfied, the above steps repeat for the divided data. In the case of multi-output, all the steps are the same for single-output and only the impurity measure is different.^{16–18)}

The time-complexity of the CART algorithm is $O(mN \log N)$, where m is the number of variables and N is the number of observations.¹⁹⁾ The learning time is proportional to the number of input variables. As described in Section 2.2, the input items of the newly proposed learning model include the data of the pre-cold rolling process. That is, if the number of input items is large, it will take a long time to learn the model when using the CART algorithm. To learn the model in the actual rolling process, a more efficient algorithm than that of the CART is needed.

4.2. Clustering-based Split-variable Selection

The key concept of the proposed algorithm is to use clustering when selecting a split variable. The most inefficient part of the CART algorithm is that all the input variables must be sorted before a split variable can be selected. The Quicksort algorithm has a time-complexity of $O(N \log N)$ and will have total time-complexity of $(mN \log N)$ if it is sorted for every variable. If we could find the variable in which the impurity function is minimum, not be necessary to sort all the variables. Therefore, a tree could be built more efficiently by sorting only a specific variable if we could find the variable giving the minimum impurity.

The CART partitions the data into two groups to minimize the specific impurity measure. This is similar to the process of a clustering algorithm. Among the clustering algorithms, one of the most efficient is the k-means algorithm. In other words, the k-means algorithm could provide an alternative method of building a tree-structured model.

The k-means algorithm clusters the data to minimize the within-cluster variance, where k is the number of clusters. If $k = 2$, the k-means algorithm minimizes the variance sum of two partitions of observations. In this case, upon applying the k-means algorithm to observations consisting of a pair of one arbitrary input variable and one output variable, the observations are clustered into two groups as a function of the input and output variables. The algorithm proposed in this paper attempts to attain the split-variable selection using this property.

To illustrate the principle of the variable selection algorithm based on the k-means algorithm, we used pairs of example data with one arbitrary input variable and one output variable (Data source: Boston housing data from the UCI repository of machine learning database,²⁰⁾ x : average number of rooms per dwelling, y : median value of owner-occupied homes in \$1 000). Figure 6 shows the scatter plot obtained using the example data. The figure shows that the relationship between the input and output is roughly linear. The goal is to divide the data into two clusters so that the output variance is minimized when using the k-means algorithm.

By modifying the within-cluster distance measure, we can construct a modified k-means algorithm such as 1) and 2).

Fig. 6. Scatter plot of x_6 - y for Boston house price dataset. (Online version in color.)

Fig. 8. Clustering result of x_6 - y for Boston house price dataset. (Distance measure: distance along x -axis). (Online version in color.)

Fig. 7. Clustering result of x_6 - y for Boston house price dataset. (Distance measure: distance along y -axis). (Online version in color.)

If we sequentially apply steps 1) and 2) for each iteration of the k-means algorithm, we can construct a tree model that is likely to minimize the output within-cluster variance.

1) Distance measure defined along the y -axis

As shown in **Fig. 7**, the data is divided by the y -axis and the summation of the variance of y is minimized. This has the advantage of minimizing the MSE, which is an impurity measure of the CART, but it cannot be used to construct a model with a tree structure like $x_i < c$.

2) Distance measure defined along the x -axis

As shown in **Fig. 8**, the data is divided by the x -axis, and the summation of the variance of x is minimized. We can construct a general tree-structured model. However, it cannot guarantee that the variance of y is minimum.

In the k-means algorithm, the center of each cluster is updated using the mean value. However, since the mean value is sensitive to outliers, it degrades the clustering performance. In particular, there is the possibility that outliers may exist in the process data in actual rolling, so there is a need for an algorithm that is less sensitive to outliers. According to Reference 21), the selection of the medoids closest to the mean, instead of the mean, when updating the reference point of each cluster, improves the clustering performance. The algorithm proposed in this paper adopts this concept.

Applying the algorithm mentioned above makes it easy to divide the data and obtain the impurity measure based on clustered data. Also, if we apply the algorithm to all the variables, we can determine the impurity measure for each

variable and select the optimum variable with the lowest impurity.

4.3. Details of Proposed Algorithm

Figure 9(a) is a flow chart of the algorithm proposed in this paper. Once the learning data are obtained, a pair consisting of one variable and an output is defined as the input data for the clustering algorithm. Then, the clustering algorithm is applied to the pre-defined data to obtain the degree of impurity of the variable. After applying the above clustering algorithm to all the variables, we can identify which variable has the lowest level of impurity.

Figure 9(b) illustrates the clustering algorithm. First, the two centroids are initialized with the minimum and maximum values of the input variable and the output. Then, the distance between the two centroids and each observation along the output is calculated. All the observations can be assigned to the cluster for which the distance from the centroids is the smallest (as shown in **Fig. 7**). The means of the clustered data are calculated, and the resulting value is used to update the centroid. Once again, the distances between the new centroids and each observation along the input variable are calculated, after which each observation is assigned to the cluster in the same way as described above (as shown in **Fig. 8**). Afterwards, the centroids are updated with the medoids with the nearest means of each cluster. If there is no difference from the centroid in the previous step, the algorithm stops and the impurity measure is obtained.

After the selection of the variable, the subsequent procedure is the same as the existing CART algorithm. It sorts the selected variable and exhaustively searches for the split point at which the impurity is a minimum. Then, the data can be divided into two parts as a result of the obtained split. If the stopping criterion is not satisfied, the above steps are repeated for the divided data.

4.4. Time-complexity Analysis of the Proposed Algorithm

In this section, we analyze the time-complexity in detail in order to determine the time efficiency of the proposed algorithm. The proposed algorithm is divided into two steps. First, the variable selection part executes the clustering algorithm in much the same way as the k-means algorithm, as many times as for the input variable. In this algorithm, the time-complexity of the k-means algorithm is $O(kNdn_{iter})$,

Fig. 9. Flow chart of proposed algorithm. (a) Main flow chart of proposed algorithm for multi-output regression tree, (b) Proposed clustering algorithm for calculating degree of impurity of x_i .

where k , N , n_{iter} , and d are the number of clusters, observations, iterations, and the dimension of the output, respectively.²²⁾ Since k is 2, d is 4 (4 stands), and n_{iter} is smaller than N , the total time-complexity of each clustering and variable selection algorithm is $O(N)$, $O(mN)$, respectively.

The second step involves searching for the split where the impurity is a minimum. Unlike the existing CART algorithm, it only sorts the selected variable, such that the time-complexity of this part is $O(N \log N)$. Since the latter part is more time consuming, the total time-complexity of the proposed algorithm is $O(N \log N)$. This algorithm is less affected by the number of variables than the existing CART algorithm. It can learn datasets with a large number of variables more efficiently.

5. Experiment and Results

5.1. Datasets

To evaluate the performance of the proposed algorithm, we applied it to the set-up of the rolling reduction rate of a tandem cold mill for stainless steel in POSCO. This process uses a four-stand mill and has been producing cold-rolled strips since 2009. **Figure 10** shows the actual plant, and the approximate specifications of the mill are listed in **Table 2**. The entire process for producing cold-rolled strip is as follows. First, slabs are produced by steelmaking and continuous-casting processes, after which hot-rolled strips are produced through a hot-rolling process. Then, white-coils are produced through annealing and cold-rolled strip is produced through continuous cold rolling. Several processes must be completed before the cold-rolling process. The data for these processes are used as the input, and the actual reduction rates of each stand of the rolling mill are used as the outputs. The data for each process are listed in **Table 3**. The data were acquired in 2017, with 2 200 coils being randomly selected.

Fig. 10. Actual plant. (Online version in color.)

Table 2. Mill specification.

Specification		data
Annual production		500 000 t/year
Material grades		Stainless-steel hot and cold strip
Strip data, entry section	Width	600 to 1 350 mm
	Thickness	1.8 to 5.0 mm (Hot strip) 0.8 to 2.0 mm (Cold strip)
	Coil weight	Max. 40 000 kg
	Coil weight	Max. 40 000 kg
Strip data, exit section	Width	600 to 1 350 mm
	Thickness	0.4 to 2.0 mm
	Coil weight	Max. 40 000 kg

5.2. Algorithm Comparison among Single-output Regression

This section compares the performances of several single-

Table 3. Input, output data list.

input/output	process	variable	item
input	steel making	x_1	wt% of C
		x_2	wt% of Si
		x_3	wt% of Mn
		x_4	wt% of P
		x_5	wt% of S
		x_6	wt% of Cu
		x_7	wt% of Al
		x_8	wt% of Nb
		x_9	wt% of Ni
		x_{10}	wt% of Cr
		x_{11}	wt% of Mo
		x_{12}	wt% of Ti
		x_{13}	wt% of Co
		x_{14}	wt% of N
		x_{15}	wt% of B
		x_{16}	slab thickness
		x_{17}	slab width
		x_{18}	slab length
	hot rolling	x_{19}	hot-rolled strip thickness
		x_{20}	hot-rolled strip width
		x_{21}	hot-rolled strip length
		x_{22}	slab temperature at furnace entry
		x_{23}	temperature at preheating zone
		x_{24}	temperature at heating zone
		x_{25}	temperature at soaking zone
		x_{26}	preheating time
		x_{27}	heating time
		x_{28}	soaking time
		x_{29}	pitch time of finishing mill
		x_{30}	roll unit order
		x_{31}	number of pass in #1 roughing mill
		x_{32}	number of pass in #2 roughing mill
		x_{33}	entry temperature of finishing mill
		x_{34}	exit temperature of finishing mill
		x_{35}	coiling temperature
		x_{36}	bar thickness
		x_{37}	bar width
		x_{38}	hot-rolled strip crown
output	annealing	x_{39}	number of process
		x_{40}	number of furnace
		x_{41}	temperature at preheating zone
		x_{42}	temperature at heating zone
		x_{43}	temperature at soaking zone
		x_{44}	reduction rate of skin pass mill
		x_{45}	line speed
		x_{46}	white coil width
	cold rolling	x_{47}	target thickness
		x_{48}	total reduction rate
output	cold rolling	y_1	#1 stand reduction rate
		y_2	#2 stand reduction rate
		y_3	#3 stand reduction rate
		y_4	#4 stand reduction rate

output regression models. By doing this, we can compare the characteristics of each model and identify the performance target of the proposed algorithm. The model types are divided into linear regression, regression tree, support vector regression (SVR), ensemble tree, Gaussian process regression (GPR), and neural network (NN). The forms of each model are shown in **Table 4**. The simulations of all these models were performed using Matlab® applications. (linear regression, regression tree, SVR, ensemble tree, and the GPR model were fitted by the Regression Learner function, while the NN model was fitted by the Neural Network Fitting function). Since these are single-output regressions, the reduction rates of the four stands were determined independently. The learning performances of the different models were compared using R-squared (**Table 5**).

According to the simulation, the best-performing model was ensemble bagged tree. This was followed by GPR, regression tree, and the NN models. Overall, tree-structured models exhibited a relatively high level of performance. In general, a NN model exhibits a high learning performance. It is interesting the performance of the tree-structured model surpasses that of the NN model. This is because the learned data are obtained from the table. A level of approximately 92–95% can be judged as being the highest level of learning performance for all the stands, and can be set as the target

Table 4. Details of local methods for single-output regression.

Model group	Model	Model options
Linear regression	Multiple regression	Constant and linear terms
Regression tree	Fine tree	Max. leaf size is 4
	Medium tree	Max. leaf size is 12
	Simple tree	Max. leaf size is 36
SVR	Linear SVR	Linear kernel function
Ensemble tree	Boosted tree	Least-squares boosting
	Bagged tree	Bootstrap bagging
GPR	Matern 5/2	Matern 5/2 kernel function
	Exponential	Exponential kernel function
NN	Multi-layer perceptron	1 hidden layer : 10 neurons

Table 5. Simulation results for performance of local models.

Model group	Model	R-squared [%]			
		y_1	y_2	y_3	y_4
Linear regression	Multiple regression	89.24	81.41	92.94	94.39
Regression tree	Fine tree	95.11	83.69	95.82	97.36
	Medium tree	94.58	84.87	96.92	96.73
	Simple tree	93.69	85.01	96.29	95.92
SVR	Linear SVR	90.25	79.68	93.81	94.04
Ensemble tree	Boosted tree	80.97	76.13	79.26	79.88
	Bagged tree	95.91	90.09	97.43	96.76
GPR	Matern 5/2	95.66	87.22	96.59	97.01
	Exponential	95.94	87.53	97.35	96.83
NN	Multi-layer perceptron	93.27	84.17	93.41	94.70

level of performance for the proposed algorithm.

Ensemble models can learn and combine several weak learner models to achieve a higher level of performance.^{23,24)} However, they are not recommended for implementation in an actual rolling mill model because the entire model is difficult to interpret and does not outperform the simple regression tree models by a particularly significant margin.

The NN model also exhibits a level of performance in excess of 90% R-squared. However, the model incurs several disadvantages that make it difficult to interpret, because it contains nonlinear activation functions, such as sigmoid and ReLu functions.²⁵⁾ In addition, the structure of this model includes many hyperparameters, making it difficult to find an optimal model.^{26,27)} For example, the types of activation function and the number of hidden layers and neurons are not easy to optimize unless the actual learning performances are compared.

As a result of assessing the performance and interpretational aspects of the model, it is thought that the regression tree-structure models are suitable for learning the rolling reduction rate.

5.3. Results Obtained Using the Proposed Algorithm

In this section, we compare the results obtained with the proposed and existing CART algorithms, and examine the applicability to the actual rolling mill model. To fairly compare the learning performance, we implemented the proposed and CART algorithms in Python[®]. Learning was executed 100 times for each algorithm. In each case, we apportioned the data into training and test sets, with and 80-20 split. As the hyperparameters of the model, the maximum depth of the tree is 1 000 and the minimum number of samples required to be at a node is 5% of the number of the training data set. The average R-squared value and the learning time were obtained. The most important advantage

of the proposed algorithm is its high time efficiency. Therefore, the performance was evaluated based on the R-squared and learning time values.

The proposed algorithm is applicable to both single- and multi-output regression tree. The performances of each model are listed in **Table 6**. The single-output regression tree learns the reduction rate of each stand, such that it has four sub-models. The learning time of this model is, on average, more than four times greater than that of a multi-output regression tree. Also, it is shown that the multi-output regression tree model learned by the proposed algorithm requires only about 18% of the learning time relative to the existing single-output CART algorithm. Therefore, it can be concluded that the proposed algorithm is better than CART in terms of time efficiency. In addition, in terms of R-squared, the multi-output regression tree is slightly inferior to the single-output regression model, but not by a significant margin. The proposed tree-structure model and algorithm can achieve a similar R-squared level of performance to the existing single-output CART algorithm.

Figure 11 shows scatter diagrams of each stand's actual and predicted reduction rate from the proposed model with a test dataset. As the scatter approaches the linear, the R-squared value increases. As such, we can see that the

Table 6. Performance of tree-type models.

Tree structure	Algorithm	R-squared [%]					Time [s]
		y_1	y_2	y_3	y_4	All	
Single-output	CART	93.56	85.48	96.14	96.26	92.86	1 978
	Proposed	93.52	85.3	95.73	96.04	92.65	362
Multi-output (Proposed)	Multi-variate CART	92.93	85.38	95.76	95.78	92.46	468
	Proposed	92.87	85.15	95.36	95.72	92.23	84

Fig. 11. Scatter plots of actual/predicted reduction rate for each stand (Dotted lines: actual reduction rate = predicted reduction rate). (Online version in color.)

Fig. 12. Time series chart for actual, predicted reduction rate of #1 stand. (Online version in color.)

learning performance for the second stand is lower than for the other stands. This is similar to the results obtained with single-output regression and is due to the high rate of manual intervention in the second stand. In general, to prevent heat-streak defects, operators usually adjust the rolling reduction rate of the second stand. This reflects the actual operation of the rolling mill. **Figure 12** is a time series chart of the reduction rate of the first stand. We can see that the difference between the actual and predicted values is negligible. However, there are 3 points (coil number are 14, 30 and 45) which have more than 3.5% difference between the actual and prediction values. We presume that these points are affected by manual interventions. Although the difference is excessively large, the reduction ratio among stands is calculated and applied to the set-up model.

6. Conclusions

In this paper, we proposed a mill set-up model that efficiently predicts the reduction rate of a tandem cold mill for stainless steel. This model can be implemented using the actual reduction rate of each stand with the data obtained from cold rolling and the previous processes. Given that this is not a conventional lookup-table model, it can set up the reduction rate by incorporating operators' experience online. Moreover, it is possible to reflect changes in the major engineering process variables in the set-up model. As such, it should be possible to reduce the variance and the degree of manual intervention by operators.

The proposed model is a multi-output regression tree. Therefore, it offers the advantage of building only a single model for the multiple-output variables and reflects the dependency of the reduction rates among stands. The existing single-output CART searches for a variable and a split to partition the data recursively, such that it is not time efficient. To overcome this issue, we proposed a more time-efficient split-variable selection algorithm based on clustering. This allows us to sort only one variable and search for

a split that minimizes the impurity measure. According to the analysis of time-complexity, the proposed algorithm is less time consuming.

To evaluate the performance of the proposed algorithm, we applied it to the set-up problem of the rolling reduction rate of a tandem cold mill for producing stainless steel in POSCO. According to the simulation results, the proposed algorithm exhibits a similar level of R-squared performance and a higher time efficiency compared with the existing CART algorithm.

Acknowledgement

This work was supported by a grant from the National Research Foundation of Korea (Project Number: 2017R1A2B4005450). We would like to thank Stainless Steel Rolling Dept. in POSCO for providing with rolling data.

REFERENCES

- 1) I. Yarita, K. Kitamura, M. Kitahama, K. Kataoka, K. Nakagawa, S. Aoki, O. Matuda and A. Yosida: *Tetsu-to-Hagané*, **67** (1981), 2152 (in Japanese).
- 2) I. C. Ozsoy, G. E. Ruddle and A. F. Crawley: *Can. Metall. Q.*, **31** (1992), 217.
- 3) D. D. Wang, A. K. Tieu, F. G. de Boer, B. Ma and W. Y. D. Yuen: *Eng. Appl. Artif. Intell.*, **13** (2000), 397.
- 4) C. T. A. Pires, H. C. Ferreira, R. M. Sales and M. A. Silva: *J. Mater. Process. Technol.*, **173** (2006), 368.
- 5) H. C. Ferreira, C. T. A. Pires and R. M. Sales: *ISIJ Int.*, **48** (2008), 1389.
- 6) S. Chen, X. Zhang, L. Peng, D. Zhang, J. Sun and Y. Liu: *J. Cent. South Univ.*, **21** (2014), 1733.
- 7) T. Watanabe, A. Kitamura, H. Narazaki, Y. Takahashi and H. Hasegawa: *ISIJ Int.*, **40** (2000), 771.
- 8) P. Singh, R. K. Pandey and Y. Nath: *Tribol. Online*, **4** (2009), 36.
- 9) N. Fujita and Y. Kimura: *ISIJ Int.*, **52** (2012), 850.
- 10) A. Azushima: *Tribology in Sheet Rolling Technology*, Springer International Publishing, Switzerland, (2016), 213.
- 11) K. Kenmochi, I. Yarita, H. Abe, A. Fukuhara, T. Komatsu, H. Kaito and A. Kishida: *Tetsu-to-Hagané*, **78** (1992), 1546 (in Japanese).
- 12) H. Borchani, G. Varando, C. Bielza and P. Larranaga: *Data Min. Knowl. Discov.*, **5** (2015), 216.
- 13) A. J. Izenman: *J. Multivar. Anal.*, **5** (1975), 248.
- 14) L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone: *Classification and Regression Trees*, Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, (1984), 216.
- 15) L. Rokach and O. Maimon: *Data Mining with Decision Trees: Theory and Applications*, World Scientific Publishing, Singapore, (2008), 77.
- 16) G. De'ath: *Ecology*, **83** (2002), 1105.
- 17) D. R. Larsen and P. L. Speckman: *Biometrics*, **60** (2004), 543.
- 18) A. D'Ambrosio, M. Aria, C. Iorio and R. Siciliano: *Expert Syst. Appl.*, **69** (2017), 21.
- 19) R. O. Duda, P. E. Hart and D. G. Stork: *Pattern Classification*, John Wiley & Sons, New York, (1997), 406.
- 20) D. Dua and E. K. Taniskidou: *UCI Machine Learning Repository*, <http://archive.ics.uci.edu/ml>, (accessed 2018-06-01).
- 21) H. S. Park and C. H. Jun: *Expert Syst. Appl.*, **36** (2009), 3336.
- 22) R. O. Duda, P. E. Hart and D. G. Stork: *Pattern Classification*, John Wiley & Sons, New York, (1997), 526.
- 23) R. Maclin and D. Opitz: *J. Artif. Intell. Res.*, **11** (1999), 169.
- 24) L. Rokach: *Artif. Intell. Rev.*, **33** (2010), 1.
- 25) V. Nair and G. E. Hinton: *Proc. 27th Int. Conf. on Machine Learning (ICML'10)*, Omnipress, Madison, (2010), 807.
- 26) J. Bergstra, R. Bardenet, Y. Bengio and B. Kegl: *Adv. Neural Inf. Process. Syst.*, **24** (2011), 2546.
- 27) J. Bergstra and Y. Bengio: *J. Mach. Learn. Res.*, **13** (2012), 281.