**Journal of Industrial Information Technology and Application** *ISSN(Online): 2586-0852*
Vol.8 No.3

# Design of Automatic Learning System for Regression and Classification Machine learning Algorithm according to Discrete and Continuous Data Attribute

Sehyun Myeong[1)] and Yoosoo Oh[2,*)]
[1, 2)]School of AI, Daegu University, Republic of Korea

**Abstract.** We present a low code-based system that enables automatic application and prediction of learning algorithms regardless of data type. The proposed system automatically converts inputted data with discrete properties into continuous data and continuous properties into discrete data. The proposed system performs well even when discrete and continuous data are cross-applied to classification and regression algorithms. Particularly, the data attribute is converted to fit the selected algorithm properly according to the type of algorithm. Accordingly, the proposed system can predict well even if the user applies inputted data to any algorithm without concern about inputted data properties.

**Keywords;** Discrete Data; Continuous Data; Regressor; Classifier; Machine Learning

## 1. Introduction

Low code is helpful for a novice or another field developer because it makes development easy without expert skills for developing a machine learning system [1]. A beginner in machine learning or a developer in other fields has difficulty distinguishing inputted data attributes like discrete or continuous data and distinguishing classification or regression algorithms. If we apply a low-code approach to machine learning, a beginner in machine learning and a developer in another field can reduce their difficulties when they choose data or algorithms. Thus, a machine learning low code system is required to provide an environment that automatically analyzes data attributes and trains algorithms for machine learning.

* Corresponding author: yoosoo.oh@daegu.ac.kr
Received: Jul 20, 2024; Accepted: Aug 25, 2024; Published: Sep 30, 2024

In this paper, we propose a low-code system that automatically trains data sets to classification and regression machine learning algorithms, whether the data sets are discrete or continuous. The proposed system automatically classifies inputted data according to discrete and continuous attributes. Moreover, the proposed system converts discrete data into continuous data and continuous data into discrete data by dividing, transforming, and recasting data values. So, when applying inputted data with discrete attributes to regression algorithms, the proposed system converts the inputted data into continuous data by using line spacing. Moreover, when applying inputted data with continuous attributes to classification algorithms, the proposed system quantiles inputted data, transforming it into discrete data. Furthermore, the proposed system automatically predicts the data that the user wishes to predict and notifies the user of the prediction results. The proposed system demonstrates high learning performance when applying inputted data with discrete attributes to regression algorithms or continuous attributes to classification algorithms.

## 2. Related Research

Table 1 represents an analysis of related research. To analyze related studies, we investigate and compare research that applies classification data to regression or regression data to classification algorithms.

In Table 1, previous research 1 and 2 use classification data as input. Research 1 utilized a classification algorithm by combining linear regression and a decision tree for data classification. Research 2 classifies data by applying a loss function optimization filter to logistic regression. Both research 1 and 2 integrate regression and classification to achieve high accuracy. However, there is no attempt to apply classification data to independent regression algorithms directly.

Research 3, 4, and 5 in Table 1 utilize regression data as input. Research 3 and 5 preprocess the regression data into a format suitable for classification algorithms. In particular, research 5 demonstrates various preprocessing methods to transform regression data into a format resembling classification data. However, previous research did not show examples of applying regression data to classification algorithms. In particular, previous 4 and 5 primarily applied regression data to decision trees, which are versatile for both regression and classification tasks, rather than simply applying regression data to classification algorithms.

Table I. RELATED RESEARCH ANALYSIS

| Related Research | Characteristic | Advantages | Disadvantages | Data and Algorithm |
|---|---|---|---|---|
| 1. A Decision Tree Algorithm Combined with Linear Regression for Data Classification | Combining Decision Tree with Linear Regression | The highest accuracy<br>Real data usage | Classification only<br>Comparison without considering regression | Discrete data<br>Classification algorithm |
| 2. Application of Logistic Regression with Filter in Data Classification | Logistic Regression with a filter method | The optimal parameters of loss function<br>The extremum of non-convex loss function | Classification only<br>No comparison with other algorithms | Discrete data<br>Classification algorithm |
| 3. Regression as Classification | Transforming a Regression dataset into Classification using a discretization | Performance comparison using double cross-validation<br>Five benchmarking regression tasks | No diverse regression dataset to classification algorithms | Continuous data<br>Classification algorithm |
| 4. Efficient Regression Algorithms for Classification of Social Media Data | Comparison between decision tree and other machine laerning algorithms | Efficient pre-processing<br>Real data usage | Decision tree only | Continuous data<br>Classification algorithm |
| 5. Regression by Classification | Preprocessing mapping for regression dataset to classification<br>Using decision tree(CN2, C4.5) | Various discrete class formation methods | No diverse regression dataset to classification algorithms | Continuous data<br>Classification algorithm |

Upon analyzing previous studies, they performed preprocessing techniques to transform discrete or continuous data, but they did not apply the transformed data interchangeably to regression or classification algorithms. Therefore, a preprocessing method is needed to transform the data, like discrete data to a regression algorithm or continuous data to a classification algorithm. Additionally, a method is needed to enable the learning of such preprocessed data regardless of the type of regression or classification algorithm.

## 3. Proposed System

The proposed system is designed to facilitate access to machine learning by applying a low-code approach for novices who are unfamiliar with machine learning.

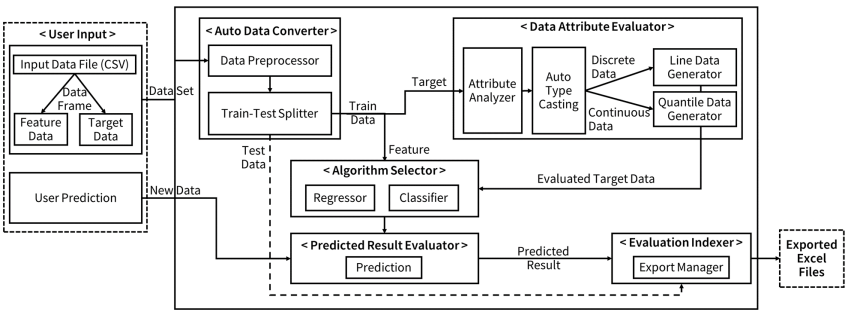



Figure 1. Proposed System Diagram

The proposed system consists of a User Input, Auto Data Converter, Data Attribute Evaluator, Algorithm Selector, Predicted Result Evaluator, and Evaluation Indexer. Figure 1 represents the system diagram.

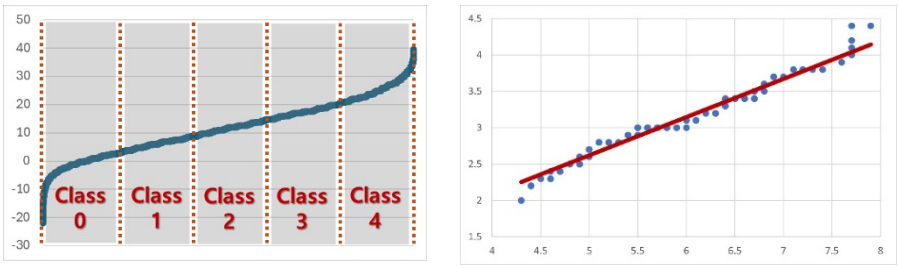
User Input is an input step where a novice user can handle data without concerns about the data attribute. First, the user converts the expected prediction data file into a Data Frame and separates it into feature and target data before the input step. The user does not need to consider whether the target data’ s attributes are discrete or continuous. After generating the learning model, the user can input new values for prediction by the predetermined types and ranges of the feature data.

Auto Data Converter is a step that preprocesses inputted data. Auto Data Converter automatically converts inputted data into an appropriate format for learning before applying it to algorithms. This step fills missing values(e.g., NaN values) with mean or maximum values. Moreover, this step converts strings into integers to facilitate vector operation for the learning algorithm. Furthermore, this step prepares for learning by splitting the preprocessed data into training and test datasets, using a 7:3 ratio.

Data Attribute Evaluator is a step that automatically classifies each data attribute and performs type casting into formats suitable for regression and classification algorithms. Data Attribute Evaluator loads all columns of the Data Frame and analyzes the attributes of each column(by Attribute Analyzer). Moreover, the Auto Type Casting module automatically discerns whether the analyzed data attributes are discrete or continuous.

Line Data Generator converts analyzed discrete data into continuous data suitable for regression algorithms by applying line spacing. The quantile method evenly divides

continuous data into intervals with uniform spacing. Quantile Data Generator divides continuous data into (N+1) quantiles. The number of quantiles depends on the range of the continuous data. In other words, it assigns unique values, from 0 to N(N is a threshold), as labels to the divided quantiles without duplication. Suppose a column is classified as continuous data consisting of integer types and is less than the threshold value (N) we defined. In that case, that column is categorized as discrete data.




(a) Continuous Data to Discrete Data in Quantile Data Generator

(b) Discrete Data to Continuous Data in Line Data Generator

Figure 2. Data Attribute Transformation

Algorithm Selector is a step that applies data to all algorithms, regardless of classification or regression. As shown in Figure 1, the Algorithm Selector generates the learning model with training data and the evaluated target data as inputs. The regression learning algorithms used are SVR, KNeighborsRegressor, and LinearRegression. Also, the used classification learning algorithms are SVC(Linear kernel), SVC(RBF kernel), LinearSVC, KNeighborsClassifier, RadiusNeighborsClassifier, MLPClassifier, VotingClassifier, RandomForestClassifier, GradientBoostingClassifier. Our system demonstrates high learning performance even when applying discrete data to regression or continuous data to classification algorithms. That is because we transform the data to fit the characteristics of the algorithms before applying it. We automatically apply the converted data across different algorithm types without distinction.

Predicted Result Evaluator is a step that automatically creates the classification and regression prediction results. Predicted Result Evaluator produces the prediction results for classification and regression algorithms, regardless of the data attributes. Our system can yield the data attribute whenever the Predicted Result Evaluator produces the learning results for each algorithm. In addition, it can identify whether the algorithm is classification or regression. Accordingly, a user can recognize the algorithm applied to the data by reviewing the output results. The user can also compare and evaluate metrics for each algorithm's prediction results, accuracy, errors, and other evaluation criteria.

Evaluation Indexer is a step that exports metrics for each algorithm, and its Export Manager creates an external Excel file. Exported Excel files are generated depending on whether the data attribute is converted or not and whether or not algorithms are cross-applied. In this step, the Excel file for discrete data consists of three parts: the case in discrete data to classification algorithms, the case in discrete data transformed into a suitable format for regression algorithms, and the original case in discrete data to regression algorithms without transforming. Also, the Excel file concerning regression data consists of three parts: the case in continuous data to regression algorithms, the case in continuous data transformed into a suitable format for classification algorithms, and the original case in continuous data to classification algorithms without transformation. The classifier Excel files store a table summarizing each classification algorithm's predicted values, accuracy, precision, recall, and F1 score. Also, the regressor Excel files store a table summarizing the predicted values, MAE(Mean Absolute Error), MSE(Mean Squared Error), R2 Score, and other metrics for each regression algorithm. The user can briefly compare algorithm performances through the organized tables in the Excel files. Besides, the user can compare the evaluation metrics of each algorithm previously trained and outputted at any time.

## 4. Experiment and Evaluation

In order to evaluate our system, we conducted a performance test of the algorithm with two datasets. The first dataset, ' Laliga player stats,' contains soccer-related information about Spanish professional players. It includes columns such as 'Team,' 'Position,' and 'Goals scored' [6]. We selected the 'Goals scored' column as the target data: the number of goals each soccer player scored. The 'Goals scored' data is considered regression because it goes from 0 to 27. Furthermore, we selected 'Minutes played' and 'Shots on target' as feature data in the 'Laliga player stats' dataset. The second dataset, 'wine quality,' contains information about wine, such as 'pH,' 'alcohol,' and 'quality.' We chose 'type' as the target data, representing a column classifying wine types. In addition, we selected 'pH,' 'residual sugar,' and 'alcohol' as feature data.

We applied two datasets to classification and regression algorithms. The classification algorithms we used consist of SVC(Linear kernel), SVC(RBF kernel), LinearSVC, KNeighborsClassifier, RadiusNeighborsClassifier, MLPClassifier, VotingClassifier, RandomForestClassifier, GradientBoostingClassifier. Moreover, our regression algorithms consist of SVR, KNeighborsRegressor, and LinearRegression. We used Accuracy, Precision Score, Recall Score, and F1 Score as evaluation metrics for classification algorithms. Moreover, we used MAE, MSE, RMSE, and Max Error as evaluation metrics for regression algorithms.

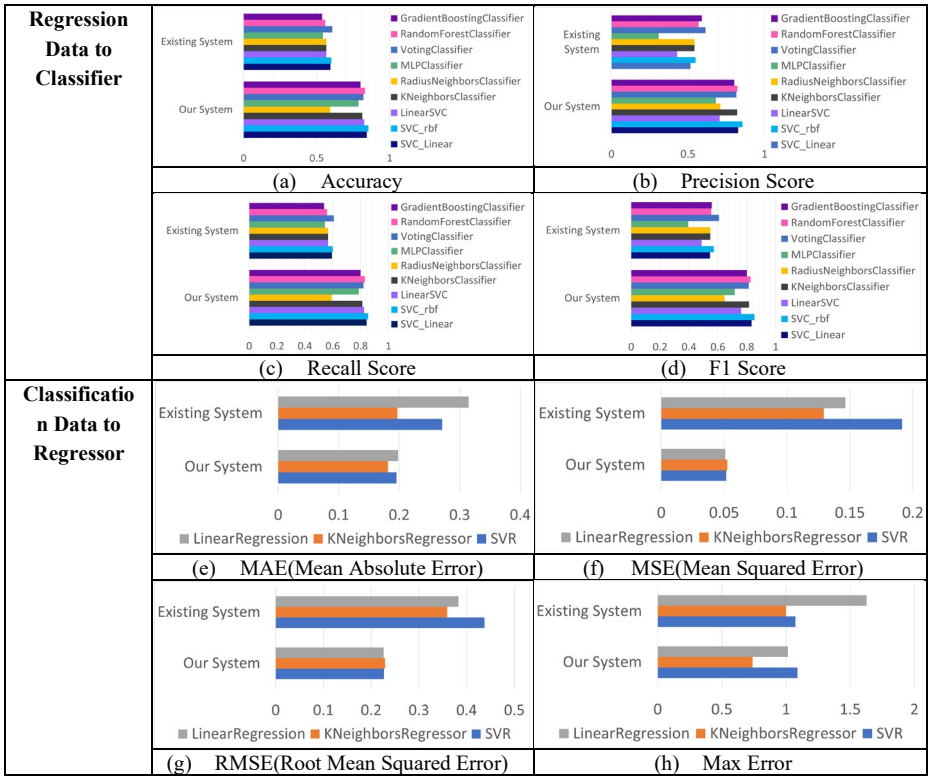



Figure 3. Regression and classification cross automatic transformation experiment results

We evaluated the performance metrics as shown in Figure 3. The proposed system converted inputted data to match the characteristics of the appropriate algorithms. Figure 3 shows the performance comparison between our system, which applied the transformed inputted data to the algorithms, and the existing system, which applied the algorithms directly to the data without any transformation.

Figures 3 (a), (b), (c), and (d) show the results of measuring the classifier evaluation metric when continuous data is applied to classification. Accuracy, Precision Score, Recall Score, and F1 Score are the metrics. Accuracy is the learning accuracy of machine learning. Precision Score, Recall Score, and F1 Score are performance evaluation indicators from the confusion matrix in machine learning classification. If the metric index is close to 100% for these four evaluation indicators, it performs better. As shown in Figure 3, our system appears higher in all performance indicators than the existing system.

Figures 3 (e), (f), (g), and (h) show the results of measuring MAE, MSE, RMSE, and Max Error, which are regression algorithm evaluation metrics when discrete data is applied to regression. Since all four performances are values obtained by calculating error rates, the small value means the better performance. As shown in the graph in Figure 3, all four regression performance indicators are smaller for our system than for the existing system. In other words, our system showed a lower error rate than the existing system, which meant our system had an excellent performance.

Table II. DISCRETE DATA INPUT TO ANY ALGORITHM(PREDICTION 0='WHITE', 1='RED')

| **(a) Discrete Data to Classifier** | | | | | |
|---|---|---|---|---|---|
| ***Algorithm*** | ***Prediction*** | ***Accuracy*** | ***PrecisionScore*** | ***RecallScore*** | ***F1 Score*** |
| SVC_Linear | [0] | 0.78012313 | 0.756214806 | 0.780123131 | 0.748278043 |
| SVC_rbf | [0] | 0.84476693 | 0.837512452 | 0.844766931 | 0.837274958 |
| LinearSVC | [0] | 0.78649956 | 0.765791678 | 0.78649956 | 0.763496069 |
| KNeighbors Classifier | [0] | 0.82277924 | 0.822673852 | 0.822779244 | 0.822726366 |
| RadiusNeighbors Classifier | [0] | 0.82277924 | 0.822673852 | 0.822779244 | 0.822726366 |
| MLPClassifier | [0] | 0.76495162 | 0.73495856 | 0.764951627 | 0.70206632 |
| VotingClassifier | [0] | 0.85180299 | 0.847441305 | 0.85180299 | 0.848856585 |
| RandomForest Classifier | [0] | 0.85422163 | 0.848450872 | 0.854221636 | 0.845507244 |
| GradientBoosting Classifier | [0] | 0.86697449 | 0.864892595 | 0.866974494 | 0.865753535 |
| **(b) Discrete Data to Regressor** | | | | | |
| ***Algorithm*** | ***Prediction*** | ***MAE*** | ***MSE*** | ***RMSE*** | ***MaxError*** |
| SVR | 0 | 0.19560224 | 0.051601031 | 0.227158603 | 1.088117685 |
| KNeighbors Regressor | 0 | 0.181427278 | 0.05259421 | 0.229334276 | 0.740558292 |
| LinearRegression | 0 | 0.197983173 | 0.051000094 | 0.225832004 | 1.012845139 |

Even if discrete data is applied to regression algorithms and continuous data is applied to classification algorithms, the applied algorithms produce high learning performance. Direct Comparing classification and regression algorithms is challenging due to their different performance metrics. In this experiment, we compared the case where the cross-application was in Discrete Data to Regressor and Continuous Data to Classifier and the case in Discrete Data to Classifier and Continuous Data to Regressor.

Table 2 compares the performance of (a) discrete data to the classifier and the performance of (b) discrete data to the regressor. As a result, the proposed system showed high performance (approximately 86%) on the classification evaluation index and a low error rate (MAE 0.18) on the regression evaluation index. Table 3 compares

the results when applying the continuous data to the classifier (a) and the continuous data to the regressor (b). As a result, the proposed system showed high performance (about 85%) on the classification evaluation metric and a low error rate (MAE 0.7) on the regression evaluation metric.

Table III. CONTINUOUS DATA INPUT TO ANY ALGORITHM

| **(a) Continuous Data to Classifier** | | | | | |
|---|---|---|---|---|---|
| ***Algorithm*** | ***Prediction*** | ***Accuracy*** | ***PrecisionScore*** | ***RecallScore*** | ***F1 Score*** |
| SVC_Linear | 3.0~36.0 | 0.84358974 | 0.828866097 | 0.843589744 | 0.83332693 |
| SVC_rbf | 3.0~36.0 | 0.85384615 | 0.85637207 | 0.853846154 | 0.853117543 |
| LinearSVC | 3.0~36.0 | 0.82564102 | 0.707337447 | 0.825641026 | 0.760925516 |
| KNeighborsClassifier | 3.0~36.0 | 0.81282051 | 0.82226637 | 0.812820513 | 0.816473603 |
| RadiusNeighborsClassifier | 3.0~36.0 | 0.59230769 | 0.711614774 | 0.592307692 | 0.64613155 |
| MLPClassifier | 3.0~36.0 | 0.78717948 | 0.681823362 | 0.787179487 | 0.716168557 |
| VotingClassifier | 3.0~36.0 | 0.82051282 | 0.817305675 | 0.820512821 | 0.814360557 |
| RandomForestClassifier | 3.0~36.0 | 0.83076923 | 0.824415484 | 0.830769231 | 0.827320966 |
| GradientBoostingClassifier | 3.0~36.0 | 0.8 | 0.802883889 | 0.8 | 0.801233011 |
| **(b) Continuous Data to Regressor** | | | | | |
| ***Algorithm*** | ***Prediction*** | ***MAE*** | ***MSE*** | ***RMSE*** | ***MaxError*** |
| SVR | [15.16237866] | 0.775964161 | 2.218568828 | 1.489486095 | 10.87730677 |
| KNeighbors Regressor | [17.] | 0.764957265 | 2.212820513 | 1.487555213 | 10.33333333 |
| Linear Regression | [19.13849603] | 0.814848461 | 1.771454975 | 1.33096017 | 8.599455203 |

We compared the predictions in Table 2 and Table 3. The predictions in Table 2 show that both predictions were 0 when applied to the classifier and regressor. Furthermore, in the prediction results in Table 3, it was predicted to be 3.0 to 36.0 when applied to the classifier. Besides, when applied to the regressor, our system predicted around 15 to 19, indicating the correct range of prediction results in both cases.

## 5. Conclusion

In this paper, we proposed a low-code system that automatically learns classification and regression machine learning algorithms regardless of input data properties. The proposed system automatically classifies and cross-transforms the inputted data according to discrete and continuous data properties. Additionally, the proposed system

automatically predicts the data the user wants to predict and informs the user about the prediction results. The experimental results showed that the proposed system automatically converted regression, classification, and high-performance machine learning. The proposed system allows novices to handle models that are learned automatically without worrying about discrete or continuous data inputs and regression or classification algorithms. In future research, we will develop the proposed system to recommend algorithms that produce the most optimal learning when applied to discrete and continuous data.

## References

[1] A. Sahay, A. Indamutsa, D. Di Ruscio and A. Pierantonio, "Supporting the understanding and comparison of low-code development platforms," 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Portoroz, Slovenia, 2020, pp. 171-178, doi: 10.1109/SEAA51224.2020.00036.

[2] A. M. Ahmed, A. Rizaner and A. H. Ulusoy, "A Decision Tree Algorithm Combined with Linear Regression for Data Classification," 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), Khartoum, Sudan, 2018, pp. 1-5, doi: 10.1109/ICCCEEE.2018.8515759.

[3] Z. Yang and D. Li, "Application of Logistic Regression with Filter in Data Classification," 2019 Chinese Control Conference (CCC), Guangzhou, China, 2019, pp. 3755-3759, doi: 10.23919/ChiCC.2019.8865281.

[4] R. Salman and V. Kecman, "Regression as classification," 2012 Proceedings of IEEE Southeastcon, Orlando, FL, USA, 2012, pp. 1-6, doi: 10.1109/SECon.2012.6196887.

[5] S. Desai and S. T. Patil, "Efficient regression algorithms for classification of social media data," 2015 International Conference on Pervasive Computing (ICPC), Pune, India, 2015, pp. 1-5, doi: 10.1109/PERVASIVE.2015.7087040.

[6] Torgo, Luís, and João Gama. "Regression by classification." Advances in Artificial Intelligence: 13th Brazilian Symposium on Artificial Intelligence, SBIA'96 Curitiba, Brazil, October 23–25, 1996 Proceedings 13. Springer Berlin Heidelberg, 1996.

[7] https://www.kaggle.com/datasets/thegreatcoder/laliga-player-stats

[8] https://www.kaggle.com/datasets/sanketsudamjadhav/wine-quality-prediction