Multi-output time series regression analysis of the drug usages in sewage treatment process

HUANG WEIQI
Group10
A0290897H

***Abstract*—The sewage treatment plant currently relies on chemical formulas to calculate the daily usage of various drugs based on water quality data. The objective is to enhance this process by using time series regression analysis to predict the drug usage more accurately, which could improve the existing formulas and reduce unnecessary chemical waste. In this paper, we propose a new system which uses a stacking model based on time series to improve the model performance. Because there are multiple target labels in the data set, we use multi-output to implement simultaneous regression for multiple labels. Considering the large volume of water quality data, feature selection is critical for enhancing model efficiency and performance. Additionally, for specific characteristics of the dataset, we adopt a classification-before-regression strategy.**

***Index Terms*—sewage treatment, time series regression, stacking model, multi-output, feature selection, classification before regression.**

I. Introduction

A. Motivation

In recent years, the sewage treatment industry has relied heavily on chemical formulas to calculate the daily dosage of treatment chemicals based on water quality data. Currently, regression analysis models exist for sewage treatment-related problems [1]. However, traditional methods often lack the flexibility for dynamic adjustments, resulting in either excessive or insufficient chemical usage. This not only increases operational costs but also poses potential environmental risks. With advancements in water quality monitoring technologies, the accumulation of large volumes of time-series data provides new opportunities for accurately predicting chemical usage. Machine learning, particularly time-series regression methods, has shown significant potential in optimizing industrial processes. However, challenges remain when dealing with multi-output targets (e.g., dosages of different chemicals) and high-dimensional water quality datasets. Additionally, the presence of special cases in the dataset can hinder the performance of direct regression approaches.

Building on these observations, the goal of our project is to design a system that enhances existing time series regression techniques to more accurately predict chemical usage in wastewater treatment. By better predicting the amount of drugs used in sewage treatment, we can greatly reduce the waste of drugs, thus achieving the role of cost reduction and efficiency. Such a system could enable smarter and more environmentally friendly wastewater treatment operations.

B. Initial Experiments

At the early stages of the project, a baseline model was chosen to evaluate the feasibility of predicting chemical usage based on water quality data. Initially, a traditional random forest model without incorporating time-series information was tested. However, the results were unsatisfactory, as the model failed to capture the temporal dependencies inherent in the data, leading to poor prediction accuracy. To address this limitation, a time-series-enabled random forest model was employed in the next phase of testing. While this approach considered the sequential nature of the data, the performance improvement was marginal, indicating that the model still struggled to effectively handle the complexity and high dimensionality of the dataset. These initial experiments highlighted the need for more sophisticated methods capable of leveraging both the temporal patterns and the multi-output nature of the problem.

Challenges Encountered

- Temporal Dependency: There could be strong time-dependent relationships between data points. Current water quality observations are often influenced by past values, introducing temporal dependencies [2]. Conventional regression models assume that samples are independent, but this assumption is violated in time series data. As a result, these techniques might not perform well when applied directly to this problem without accounting for the temporal nature of the data. And the existing and commonly used time series random forest model is not good at this problem. There is an urgent need to develop new model frameworks to improve performance.
- Inter-Targets Relationships: Predicting multiple drug usages simultaneously involves capturing interdependencies between the drugs. If these relationships are not properly modeled, important information might be missed, resulting in poor performance. Effectively addressing these correlations is crucial for an accurate multi-output regression model.
- Feature selection The sewage treatment plant collects a large volume of water quality data daily, but not all of these sensors are equally relevant for predicting certain drug usage. Identifying the most important features while excluding redundant data is critical to improving model performance and reducing computation time [3]. Determining which features contribute most to predicting drug

usage, and effectively implementing feature selection, is a significant challenge in this project.

In consideration of the above challenges, we plan to design a time series model to solve the problem of hidden time series information in this project, design a multi-output model to solve the interaction between variables, and design a feature selection method to reduce the impact of irrelevant variables and noise on model prediction.

C. Baseline model

We chose the TimeSeriesForestRegressor model [4] as the baseline model for our time series model. For feature selection, the baseline of not using feature selection and using SelectKBest method [5] as feature selection will be compared with the feature selection method designed by us.

II. Structure of Dataset

There are two datasets, which are input feature dataset and target label dataset. The original data set is Chinese dataset, and the feature names are all Chinese records.

A. input feature dataset

The factory records water quality data every hour, and each time the data is recorded for a total of 20 different indicators. In the data table, each row represents data recorded during a given hour, and each column represents a water quality indicator recorded. In this project, so we have a total of 20 input features, such as PH value, temperature, etc. There are partial namesake characteristics because water quality data is recorded once when the sewage enters the plant and once when it exits the plant.

- **each row**: data recorded during a given hour
- **each column**: data of a water quality indicator

The connections between different input features are shown as a heatmap in the Figure 1.

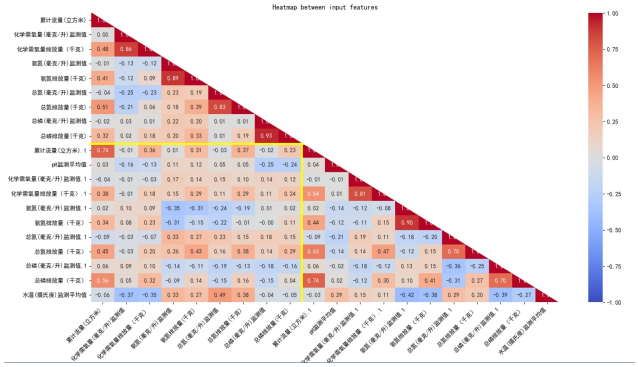

Fig. 1: Heatmap between input features

B. target label dataset

There are five target labels for the project, each indicating the use of five different drugs. The five drugs will be used when the plant treats sewage. Each drug is added to the sewage by the factory only once a day. So the drug usage in the data table is recorded on a daily basis. The target label dataset consists of six lines, the first of which is used to record time information, and the next five lines are used to record the use of five different drugs. Each column of the dataset represents the sample data on a given day.

- **each row**: corresponding to the amount of a particular drug used.
- **each column**: data of a specific day.

The connections between input features and target labels are shown as a heatmap in the Figure 2.

Fig. 2: Heatmap between input features and target labels

Therefore, the input data of our model is all the water quality information data recorded by the factory within 24 hours of each day, and the output data is the amount of five drugs used by the factory on the corresponding day.

III. Data Pre-processing and Workflow pipeline

A. Data Pre-processing

Data matching Water quality data is recorded every hour at the plant, which means that each line in the input characteristic data set represents the sample data within a certain hour. But each drug is only used once a day, and the data in the target label dataset is broken down by day. The first step of preprocessing is to match every 24 rows of data in the input feature data set with the corresponding day's data in the target label data set according to the time information.

Null removal The matched sample data is detected, and the samples containing empty values or outliers are removed.

Test set and Training set partition The data set is divided into a test set and a training set in a ratio of 7 to 3.

Normalized processing We prepared three normalization strategies for the experimental data, which are no-normalization, max-min normalization and standard normalization [6]. After testing with multiple regression models, standard normalization among the three normalization strategies can obtain better experimental results on this problem. Therefore, the experimental results in the following paper are obtained by default based on the use of standard normalization. But we still left interfaces in the code that specify normalization strategies for maintainability and improvement of subsequent projects. Taking MSE valus of label 2 as an example, we used several regression models to test different normalization strategies, and the results are shown in the Figure 3.

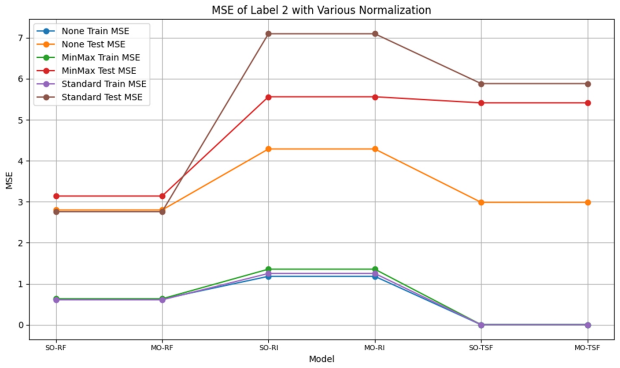

Fig. 3: workflow pipeline of the data process

Each column in the Figure 3 represents the experimental results of a different model. The blue line segment represents the results without normalization in the training process, and the orange line segment represents the results without normalization in the testing process. The green line segment represents the results with max-min normalization in the training process, and the red line segment represents the results with max-min normalization in the testing process. The purple line segments represent the results of standard normalization during training, and the brown line segments represent the results of standard normalization during testing. It can be found that using standard normalization on this dataset can obtain better model performance than using max-min normalization or no normalization.

Special treatment for random forests Tree-based regression models, such as random forests, inherently cannot process multi-dimensional input data due to their reliance on feature splits, which require scalar input values. Therefore, in the process of data preprocessing, we will copy the input feature data set, flatten the two-dimensional input matrix therein, and then use the one-dimensional matrix obtained as the input of the random forest model.

Special treatment for special data When we look at the structure of some of the data, we find that the data distribution of target label one is very special. The data structure of target label one is shown in the EDA Figure 4.

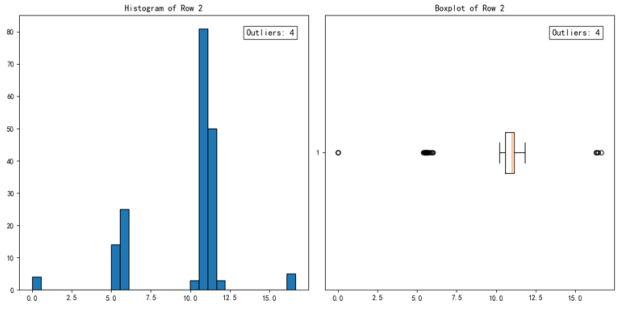

Fig. 4: EDA picture of label 1

As you can see from Figure 4, there are two different distributions for this target label. The first case is that the value of the target label is near 5.5, and the second case is that the value of the target label is near 11. Therefore, we adopt the strategy of classification before regression to improve the prediction accuracy of this label. The random forest classifier is first used to determine which distribution the sample belongs to, and then the judgment results are also used as additional information to control the predictions of subsequent regression models.

B. workflow pipeline

The workflow pipeline of the process is as Figure 5.

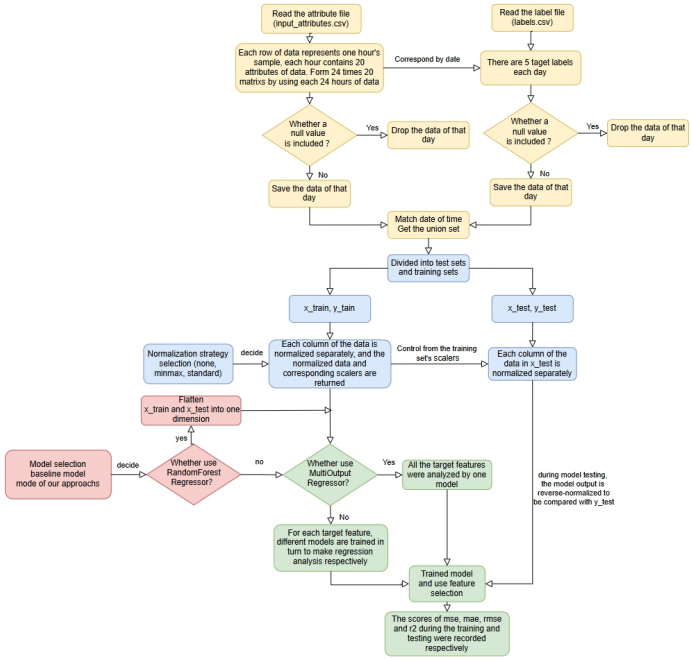

Fig. 5: workflow pipeline of the data process

The yellow blocks in the Figure 5 represent the process of matching data time information and dividing the training set and test set. The blue blocks indicate that the user can specify a normalization strategy (standard normalization is used by default). The red blocks indicate that flattening data will be performed for regression models based on random forest classes. The green blocks represent the multi-output model training process based on time series, in which feature selection is used to optimize the model.

IV. Approaches

A. timeseries regression

The baseline model is the TimeSeriesForestRegressor from the aeon library. To address the shortcomings of the baseline and enhance model performance, I first focused on adapting LightGBM [7] and XGBoost [8] to handle time-series data effectively. It is worth noting that, to date, there are no publicly available libraries that directly implement time-series-specific versions of these two powerful algorithms. Considering that both LightGBM and XGBoost are generally more advanced

and flexible than traditional random forest models, I designed customized versions tailored to time-series regression tasks.

In their standard implementations, LightGBM and XGBoost utilize data from only the current time point when making predictions. However, in time-series tasks, the relationships between past and present observations often hold critical predictive value. To address this, I extended these models to incorporate a sliding window approach, where input features are constructed from a sequence of data spanning a continuous time period in the past. This enables the models to learn and leverage the temporal dependencies and patterns embedded in the data, significantly improving their ability to predict future outcomes.

By using these customized, time-series-enabled versions of LightGBM and XGBoost, the models can extract and utilize the hidden information within the time-series, making them a powerful alternative to traditional random forest approaches for temporal data. Building on this foundation, I then designed a stacking ensemble model to further enhance predictive performance.

The stacking model consists of two layers:

- First layer: It includes three base models: the TimeSeriesForestRegressor, the time-series-enabled LightGBM, and the time-series-enabled XGBoost. These models capture different aspects of the temporal dependencies and patterns in the data.
- Second layer: Two options are explored. The first is a Ridge Regressor, which is used to combine the outputs of the first-layer models in a regularized linear manner, providing robustness to multicollinearity. The second is a GradientBoostingRegressor, which leverages its ensemble nature to further refine predictions by capturing residual errors.

This method effectively integrates the strengths of tree-based models and linear regression techniques, leveraging the temporal structure of the data while addressing the multi-output regression challenge. By stacking these models, the system achieves better generalization and predictive accuracy compared to the baseline.

B. multi-output regression

Many regression models can only support the regression output of a single target variable, random forest is one of them. In order to solve the problem that some of the model frameworks used in the experiment do not support multi-variable output, we designed Algorithm 1 so that the model can fully learn the interaction between various variables.

The proposed two-round regression chain framework offers significant advantages over traditional single-output regression methods by fully leveraging the interdependencies among multiple target variables. In single-output regression, each target variable is predicted independently, ignoring the potential relationships and interactions between them. This approach may fail to capture the underlying structure in the data when the target variables are interrelated.

Algorithm 1 Multi-output Chain Regression Framework

Require: Dataset $D = \{(X_i, \mathbf{Y}_i)\}_{i=1}^{N}$, where:

$X_i \in \mathbb{R}^d$: input feature vector for sample i,

$\mathbf{Y}_i = [Y_{i,1}, Y_{i,2}, \ldots, Y_{i,K}] \in \mathbb{R}^K$: vector of K target labels.

Ensure: Final predictions for all samples:

$\hat{\mathbf{Y}}_i = [\hat{Y}_{i,1}^{(2)}, \hat{Y}_{i,2}^{(2)}, \ldots, \hat{Y}_{i,K}^{(2)}]$.

for $t = 1$ to 2 **do**

  for $k = 1$ to K **do**

    Train regressor $R_k^{(t)}$ using the rule:

$$\hat{Y}_{i,k}^{(t)} = R_k^{(t)} \left(X_i, \{\hat{Y}_{i,j}^{(t)} \mid j < k\}, \{\hat{Y}_{i,j}^{(t-1)} \mid j > k\} \right)$$

    where:

    X_i: original input features for sample i,

    $\{\hat{Y}_{i,j}^{(t)} \mid j < k\}$: predictions for preceding target labels in round t,

    $\{\hat{Y}_{i,j}^{(t-1)} \mid j > k\}$: predictions for subsequent target labels from round $t-1$.

  end for

end for

Return final predictions after $t = 2$:

$$\hat{\mathbf{Y}}_i = [\hat{Y}_{i,1}^{(2)}, \hat{Y}_{i,2}^{(2)}, \ldots, \hat{Y}_{i,K}^{(2)}]$$

In contrast, the regression chain explicitly incorporates the predictions of other target variables as additional features during training, enabling the model to learn the dependencies between targets. By sequentially including predictions from preceding variables and refining these predictions over two rounds, the method ensures a more comprehensive understanding of the relationships among all target variables.

This approach is particularly beneficial in scenarios where target variables exhibit strong correlations or shared patterns. For example, in real-world applications such as time-series forecasting, environmental modeling, or multi-output industrial processes, the proposed framework can significantly enhance prediction accuracy by modeling these interdependencies. Furthermore, the iterative refinement across two rounds provides an additional layer of robustness, allowing the model to correct initial errors and achieve improved overall performance.

C. feature selection

As early as a long time ago, the use of simulated annealing algorithm has been a hot research topic in the field of feature selection [9].

In the next step of our methodology, we propose a feature selection approach that combines an initial feature importance ranking with a simulated annealing optimization framework. The goal of this method is to iteratively refine the feature subset, ensuring optimal model performance while accounting for the contributions of each feature.

The steps of the method are summarized in Algorithm 2.

Algorithm 2 Simulated Annealing with Logarithmic Initial Feature Weights for Feature Selection

Require: Feature set $\mathbf{F} = \{f_1, f_2, \ldots, f_n\}$, training data (X, y), initial temperature T_{init}, cooling rate α, stopping temperature T_{stop}, maximum iterations N_{iter}.

Ensure: Optimal feature subset $\mathbf{F}_{\text{best}}$.

Initial Feature Importance Ranking:

for $j = 1$ to m **do**
  Randomly split $\mathbf{F}$ into $\mathbf{F}_{1,j}$ and $\mathbf{F}_{2,j}$
  Train and evaluate models using $\mathbf{F}_{1,j}$ and $\mathbf{F}_{2,j}$
  Select the better-performing subset as $\mathbf{F}_{\text{well},j}$
end for
Count occurrences of each $f_i \in \mathbf{F}$ in $\mathbf{F}_{\text{well},j}$:

$$C(f_i) = \sum_{j=1}^{m} \delta_{i,j}, \quad \delta_{i,j} = \begin{cases} 1, & \text{if } f_i \text{ is in } \mathbf{F}_{\text{well},j}, \\ 0, & \text{otherwise.} \end{cases}$$

Sort features $\mathbf{F}$ by $C(f_i)$ to get $\mathbf{F}_{\text{sorted}}$.

Initial Feature Weight Assignment:

Assign weights $w_i = \frac{1}{\log(i+1)}$ for $i = 1, 2, \ldots, n$
Normalize weights $\tilde{w}_i = \frac{w_i}{\sum_{j=1}^{n} w_j}$

Initialization:

Set $T \leftarrow T_{\text{init}}$
Select initial $\mathbf{F}_{\text{current}}$ from $\mathbf{F}_{\text{sorted}}$ (e.g., top k)
Compute $E(\mathbf{F}_{\text{current}})$

Annealing Process:

while $T > T_{\text{stop}}$ and iterations $\leq N_{\text{iter}}$ **do**
  Generate new subset $\mathbf{F}_{\text{new}}$:
  Remove a feature f_k from $\mathbf{F}_{\text{current}}$ with probability $\tilde{w}_k$
  Add a feature f_l to $\mathbf{F}_{\text{current}}$ with probability $\tilde{w}_l$
  Evaluate $E(\mathbf{F}_{\text{new}})$
  if $E(\mathbf{F}_{\text{new}}) < E(\mathbf{F}_{\text{current}})$ **then**
    Accept $\mathbf{F}_{\text{new}}$
  else
    Accept $\mathbf{F}_{\text{new}}$ with probability:

$$P(\mathbf{F}_{\text{new}}) = \exp\left(-\frac{E(\mathbf{F}_{\text{new}}) - E(\mathbf{F}_{\text{current}})}{T}\right)$$

  end if
  Update $\mathbf{F}_{\text{current}}$ and $\mathbf{F}_{\text{best}}$ if $E(\mathbf{F}_{\text{current}}) < E(\mathbf{F}_{\text{best}})$
  Update temperature $T \leftarrow \alpha T$
end while
Return $\mathbf{F}_{\text{best}}$

Our approach begins by evaluating the initial importance of features through multiple trials, where features are ranked based on their performance contributions in well-performing subsets. These rankings are then used to assign logarithmic initial weights to features, which guide the simulated annealing process. The annealing framework iteratively adds or removes features, balancing exploration and exploitation through probabilistic acceptance criteria based on performance improvements. This allows the algorithm to escape local minima and converge to an optimal feature subset.

The Simulated Annealing for Feature Selection method with initial feature weights offers several distinct advantages, making it a robust and efficient approach for high-dimensional feature selection tasks. By incorporating initial weights derived from a preliminary feature importance ranking, the algorithm guides the search process toward more promising regions of the solution space from the outset. This reduces the reliance on random exploration and accelerates convergence to an optimal or near-optimal feature subset.

The initial weights prioritize features with higher relevance, ensuring they are more likely to be included in the feature subset during the early stages of the annealing process. This not only improves the efficiency of the search but also enhances the algorithm's ability to capture important features while discarding less relevant ones. Additionally, the weighting mechanism allows the algorithm to balance exploration and exploitation dynamically, maintaining diversity in the search process while focusing computational resources on refining subsets with high potential.

Furthermore, the simulated annealing framework provides a probabilistic mechanism to escape local optima, ensuring that the final selected feature subset is not overly influenced by initial conditions. This is particularly beneficial in scenarios where feature interdependencies or redundancy could otherwise lead to suboptimal selections. Overall, the integration of initial weights with simulated annealing combines the strengths of informed search and stochastic optimization, resulting in a feature selection method that is both computationally efficient and highly effective in identifying relevant features for complex predictive tasks.

V. Experiments

The MSE is used as the model performance evaluation standard.

A. Time series Regression

The results of time series regression are shown in the Figure 6, 7, 8, 9 and 10. In the picture, the point on each column represents a kind of model. From left to right there are TimeseriesForestregressor (baseline), TimeseriesLightGBM, TimeseriesXGboost, Stacked_RidgeRegressor and Stacked_GradientBoostingRegressor. The orange lines represent the test set results and the blue lines represent the training set results.

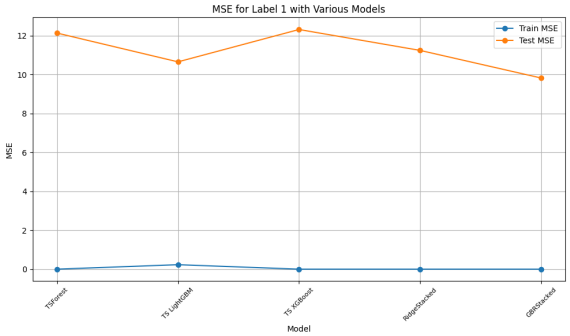

Fig. 6: MSE for label 1 with various models

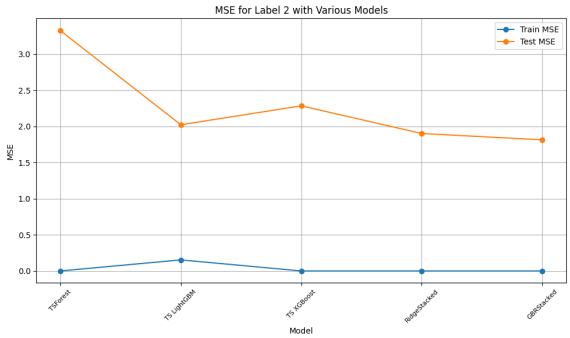

Fig. 7: MSE for label 2 with various models

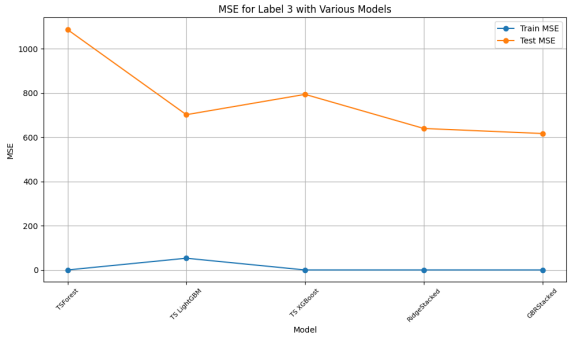

Fig. 8: MSE for label 3 with various models

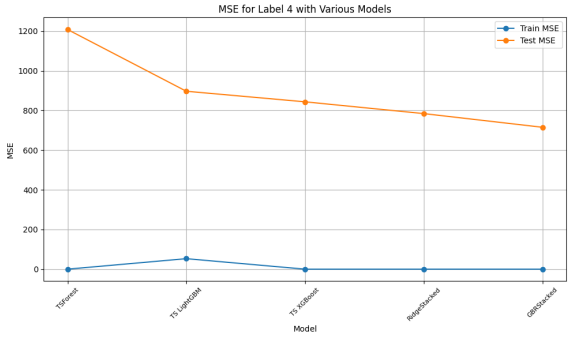

Fig. 9: MSE for label 4 with various models

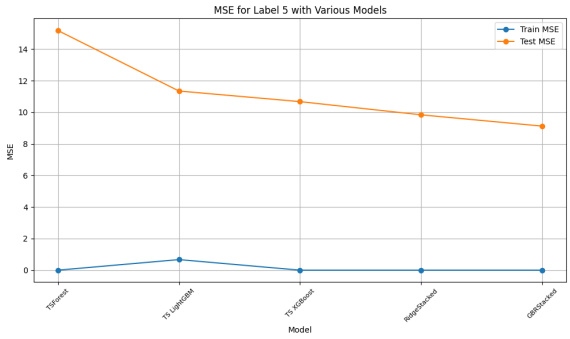

Fig. 10: MSE for label 5 with various models

As you can see from the results, the LightGBM model based time series and the XGBoost model based on time series perform better than the baseline model. And stacking model based on their had the smallest MSE value and achieved the best performance.

B. Feature Selection

The training process of the feature selection model using the improved annealing-based algorithm is shown in the figure below. The model reached the temperature threshold after about 9,500 iterations. The change process of MSE value is shown in the Figure 11, 12, 13, 14 and 15.

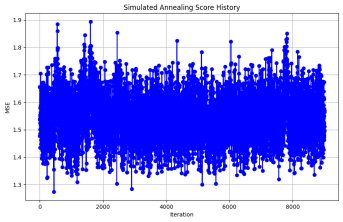

Fig. 11: MSE for label 1 during simulated annealing

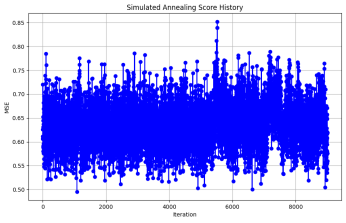

Fig. 12: MSE for label 2 during simulated annealing

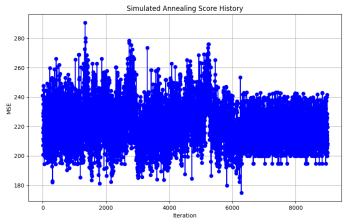

Fig. 13: MSE for label 3 during simulated annealing

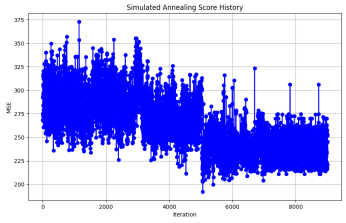

Fig. 14: MSE for label 5 during simulated annealing

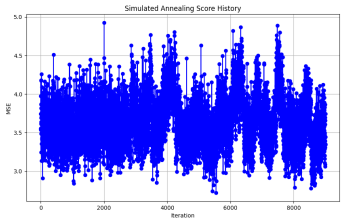

Fig. 15: MSE for label 5 during simulated annealing

We reserve the best result from each iteration as the optimal policy and then compare it with the policy that does not perform feature selection and the baseline strategy that uses the SelectKBest method. The comparison results are shown in the Figure 16, 17, 18, 19 and 20, where the green line represents the model performance without feature selection, the red line represents the baseline model performance using the SelectKBest method, and the blue line represents the results of our method.

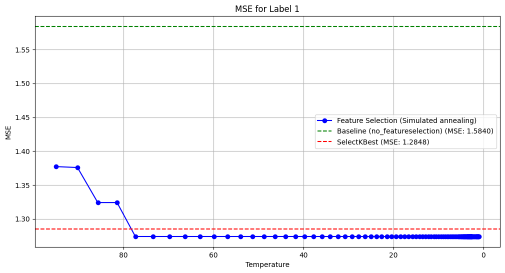

Fig. 16: MSE for label 1 with different feature selections

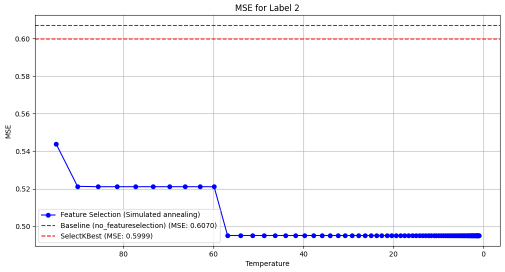

Fig. 17: MSE for label 2 with different feature selections

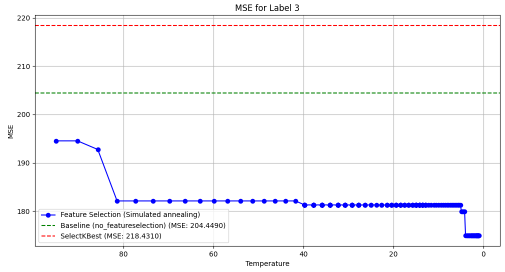

Fig. 18: MSE for label 3 with different feature selections

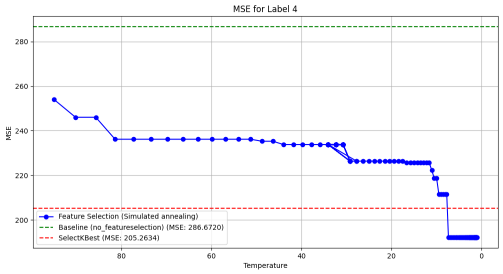

Fig. 19: MSE for label 4 with different feature selections

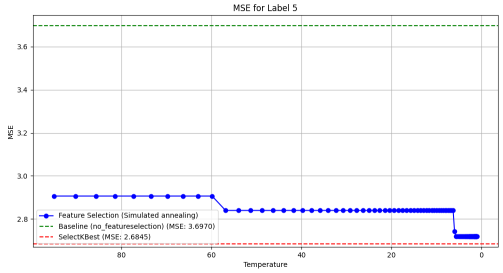

Fig. 20: MSE for label 5 with different feature selections

The proposed feature selection method, based on simulated annealing with initial feature weights, demonstrates significant advantages over the baseline approach. As observed in 16, 17, 18 and 19, our method consistently enhances model performance, outperforming the baseline in most cases. By integrating an initial feature importance ranking with a probabilistic optimization process, the method effectively prioritizes critical features and reduces the influence of noise on model predictions. This results in a more robust feature selection process that better aligns with the underlying data patterns.

Although our approach slightly underperforms the baseline in Figure 20, this outcome is expected and within an acceptable range. The specific conditions depicted in Figure 20 may amplify certain feature interactions or noise patterns that the baseline model inherently favors, giving it a slight edge in this particular case. However, such conditions are uncommon in the broader dataset, and the overall performance across all scenarios still strongly favors our method. The slight underperformance does not detract from the method's effectiveness, as it reflects a trade-off inherent in balancing robustness and generalization.

Overall, our method delivers superior performance in improving model predictions, demonstrating its ability to enhance the original model's accuracy and robustness while maintaining a reliable and practical feature selection process.

VI. Conclusion

A. Conclusion for project

This project presented a comprehensive approach to address the challenges of multi-output time series regression by integrating a custom-stacked model architecture with

an advanced feature selection strategy. The stacked model combines the strengths of multiple base learners, including a TimeSeriesForestRegressor, time-series-adapted LightGBM, and XGBoost, designed to incorporate historical dependencies and capture temporal patterns effectively. These base learners are further refined through a meta-regressor, leveraging their complementary strengths to improve predictive accuracy and robustness.

To complement the model architecture, a simulated annealing-based feature selection method with initial feature weights was developed. This approach prioritizes relevant features based on an initial ranking and uses a probabilistic optimization process to explore promising feature subsets while avoiding local optima. The feature selection strategy not only improves computational efficiency but also enhances the interpretability of the model by focusing on the most impactful predictors.

Together, the combination of the stacked model and feature selection method demonstrated significant improvements in both predictive performance and scalability. This approach addresses key challenges in time series regression, particularly the need to balance complex temporal relationships with efficient feature utilization. The project's innovations provide a robust framework for tackling multi-output regression problems across diverse applications, setting a foundation for further enhancements with advanced meta-learning techniques or deep learning integrations.

B. Project improvement plan

The sample size of the original dataset was insufficient, which could lead to overfitting during training, as observed in some experiments. In the future, we can try to use some data enhancement methods to expand the existing data set and avoid or reduce the risk of overfitting the model.

C. Conclusion for course

In CS5344 course, I learned a lot about data processing knowledge. From the perspective of knowledge, I have learned about clustering, similarity search, frequency analysis, clustering, anomaly detection, classification and regression, graphs mining, PageRank and so on in this course.

However, I think the biggest impact of this course on me is its impact on my attitude towards scientific research. Although I initially found it challenging not to rely on neural networks, this course taught me an important lesson: not all problems can be effectively solved using neural networks. We may sometimes depend on neural networks too much and ignore the structure of the data itself.

Artificial intelligence should not be a show-off technology, and we should stick to the original intention of exploring data. Only by understanding the structure of data can we better use AI-related technologies to solve our challenges.

REFERENCES

[1] Fallahiarezoudar E, Ahmadipourroudposht M, Yakideh K, et al. An eco-environmental efficiency analysis of Malaysia sewage treatment plants: An incorporated window-based data envelopment analysis and ordinary least square regression[J]. Environmental Science and Pollution Research, 2022, 29(25): 38285-38302.

[2] Babii A, Ghysels E, Striaukas J. Machine learning time series regressions with an application to nowcasting[J]. Journal of Business & Economic Statistics, 2022, 40(3): 1094-1106.

[3] Venkatesh B, Anuradha J. A review of feature selection and its methods[J]. Cybernetics and information technologies, 2019, 19(1): 3-26.

[4] Deng H, Runger G, Tuv E, et al. A time series forest for classification and feature extraction[J]. Information Sciences, 2013, 239: 142-153.

[5] Lazaros K, Tasoulis S, Vrahatis A, et al. Feature selection for high dimensional data using supervised machine learning techniques[C]//2022 IEEE International Conference on Big Data (Big Data). IEEE, 2022: 3891-3894.

[6] Jo J M. Effectiveness of normalization pre-processing of big data to the machine learning performance[J]. The Journal of the Korea institute of electronic communication sciences, 2019, 14(3): 547-552.

[7] Ke G, Meng Q, Finley T, et al. Lightgbm: A highly efficient gradient boosting decision tree[J]. Advances in neural information processing systems, 2017, 30.

[8] Ramraj S, Uzir N, Sunil R, et al. Experimenting XGBoost algorithm for prediction and classification of different datasets[J]. International Journal of Control Theory and Applications, 2016, 9(40): 651-662.

[9] Mafarja M M, Mirjalili S. Hybrid whale optimization algorithm with simulated annealing for feature selection[J]. Neurocomputing, 2017, 260: 302-312.

VII. APPENDIX

A. Dataset

You can find the dataset (input_attributes.csv and labels.csv)for this project at https://github.com/gaoyellow369/CS5344_Group10

B. Source Code

You can find the source code for this project on GitHub at https://github.com/gaoyellow369/CS5344_Group10