UNLOCKING EFFICIENCY AND SUSTAINABILITY IN ENZYME OPTIMIZATION: A MACHINE LEARNING-DRIVEN APPROACH FOR INDUSTRIAL APPLICATIONS

Yijia Zhang[1], Ivan Revilla[2]

[1]Huaai Preparatory Academy, No. 188 Jingyuan West Road, Pidu District, Chengdu, Sichuan, China 610000
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

This paper addresses the challenge of optimizing enzyme activity and production in various industries by leveraging machine learning models [9]. Traditionally, enzyme optimization has been resource-intensive and costly [10]. Our proposed solution involves collecting diverse enzymatic reaction data, generating synthetic data, and using cross-validation and ensemble methods for model selection. Challenges such as data availability and negative value generation in dummy data were addressed creatively. Experimentation revealed that ensemble methods like Random Forest and Decision Tree Regressor outperformed linear models, highlighting the potential of machine learning in enzyme optimization [11]. This research offers a data-driven approach that promises efficiency and resource conservation, with significant implications for biotechnologists, industrial manufacturers, and the scientific community [12]. The application of machine learning in enzyme optimization not only streamlines processes but also paves the way for sustainability and innovation in enzyme-related industries, making it a compelling solution for widespread adoption.

KEYWORDS

Enzyme Optimization, Machine Learning Models, Resource Conservation, Industrial Efficiency

1. INTRODUCTION

Problem: The optimization of enzyme activity and production is a critical challenge in various industries, where enzymes play a pivotal role. Traditionally, this optimization process entails costly and time-consuming repetitive experimentation [1].

Background: Enzymes serve as indispensable catalysts in numerous industrial processes. However, the conventional methods for enhancing their activity and production are resource-intensive, involving a significant investment of time and materials [2].

Importance: The integration of machine learning models into this domain offers a promising avenue to streamline these processes, significantly improving efficiency while conserving valuable resources.

David C. Wyld et al. (Eds): SIPR, NCWC, BIBC, SOFEA, DSML, ARIA, NLP, CSEN -2023
pp. 185-197, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.131915

Stakeholders: This innovative approach directly benefits biotechnologists and researchers seeking to unlock the full potential of enzymes, as well as industrial manufacturers aiming to enhance productivity and competitiveness.

The article titled "A Comprehensive Analysis of Machine Learning-Based Assessment and Prediction of Soil Enzyme Activity" focuses on predicting soil enzyme activity based on various soil characteristics and properties. It employs multiple machine learning models, including Multiple Linear Regression (MLR), Random Forest (RF), and Artificial Neural Networks (ANN), to achieve this.

The paper on "Predicting the effect of silver nanoparticles on soil enzyme activity using the machine learning method" aims to predict the impact of silver nanoparticles (AgNPs) on soil enzyme activities. It utilizes machine learning models such as Artificial Neural Networks (ANN) optimized with Genetic Algorithm (GA), Gradient Boosting Machine (GBM), and Random Forest (RF) for analysis.

Shortcomings of these methodologies include limitations in dataset representativeness and challenges in generalizing results to broader contexts.

Our project builds upon these works by offering a method for generic enzyme activity prediction. We explore various machine learning models and choose the one with the best accuracy. While similar in concept, our project's focus is on a different domain, a broader field of enzyme activity prediction.

Solution: We propose using machine learning models and algorithms to predict generic enzyme activity and production [3].

Steps of solving the problem:

1. Gather data from enzymatic reactions involving different Initial substrate concentration, enzymes addition, and reaction productivity. Dummy data are generated to provide a larger dataset.
2. Randomly divide the dataset into two subsets: a training set and a testing set. The training set would allow models to learn patterns of the dataset, and the testing set would evaluate their performance.
3. Test appropriate machine learning models for the best prediction accuracy. Compare the performance of different models based on evaluation metrics such as accuracy and R-squared value to choose the best-performing model.

In our experiments, we sought to identify potential blind spots in our program's data analysis and machine learning processes. Through Experiment A, we evaluated the generalization capabilities of different regression models by examining their training and testing accuracies. Experiment B focused on the generation of dummy data using a normal distribution, contrasting it with the randomness introduced by a uniform distribution.

We set up Experiment A by employing cross-validation techniques and analyzing model training and testing accuracies. Experiment B involved calculating mean and standard deviation for each parameter, generating dummy data using a normal distribution to mimic original trends.

In Experiment A, the most significant findings were that ensemble methods like Random Forest and Decision Tree Regressor exhibited better performance compared to linear-based models.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Data Availability

In the context of gathering data for the enzymatic reaction, data availability (acquiring a real-world dataset) is a problem faced. Although my dad's pharmaceutical company was able to provide a dataset of their Quercetin conversion with enzymes from Rutin, the data was not large enough.

In order to resolve it, dummy data could be generated from existing data to create additional samples.

2.2. The Generation of Negative Values

When doing the data processing step, one major challenge is handling the generation of negative values within the dummy data due to the utilization of a "normal distribution" for dummy data generation.

To address this issue, we could drop all the negative values, ensuring that the generated data aligns with the characteristics of actual experimental results.

We initially thought that scaling is another issue. If the different variables are scaled separately, it will affect the model's ability to make accurate and reliable predictions.
However, after scaling our data, we discovered that scaling, in fact, doesn't significantly impact the results, proving it to not be a major problem in data processing.

2.3. Machine Learning Models

When dealing with machine learning models, selecting the most suitable one is crucial. The challenge is determining which model will provide the best prediction accuracy for the testing set.

To solve this problem, I could try tons of different models to see which one works the best. I could also review scholarly papers related to this research problem and incorporate algorithms that are effective in those studies.

3. SOLUTION

Three major components: Data Preparation, Model Training, and Web Interface.

1. Data Preparation:
A. Collect enzymatic reaction data
B. Split data into training and testing sets
C. Scale the data & remove all negative values

2. Model Training:
A. Train machine learning models using preprocessed data to predict enzyme activity and production
B. Select the best algorithm for accurate predictions

3. Web Interface:

A. Create a website enabling users to upload CSV files with their data
B. Automatically process the data and make predictions
C. Users have the option to choose from different algorithms to find the one that works best for their data

Flow (of the website):

1. Users upload CSV data on the website.
2. Data is preprocessed and cleaned.
3. Machine learning models are trained for predictions.
4. Users select the best algorithm base on accuracy.
5. Models display the predictions.

What did I use to create this?

- Python (pandas, numpy, scikit-learn…) for data processing and training
- HTML, CSS, JS for website

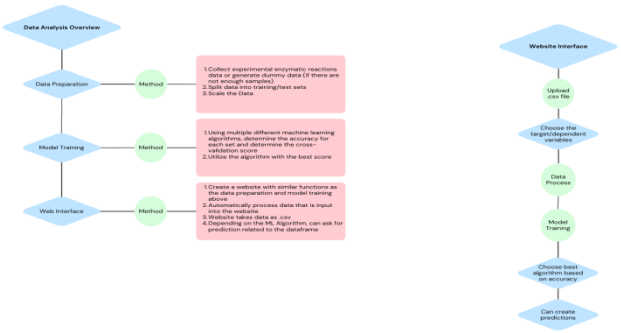


Figure 1. Overview of the solution

The purpose of the first component, Data Preparation, is to prepare and preprocess the enzymatic reaction dataset for machine learning analysis [15]. It ensures that the data is clean, scaled (normalized), and structured. Raw data is also converted to a data frame for easier analysis.

Due to the limited original dataset size, we also created 1000 dummy data based from the mean and the standard deviation from a normal curve. This allowed us to amplify the dataset's size and diversity, providing a broader range for our machine learning models to learn from.

To achieve this, we used Basic Data Analysis libraries with Python to manage and clean the data. Additionally, we employed the scikit-learn library to standardize the data, ensuring that different scales don't affect the results. This step also involved splitting the data into two parts: one for training the machine learning models and another for testing the models, which we used the train_test_split in sklearn.

Function:This component takes the raw enzyme reaction data and transforms it into a structured and normalized format. The data cleaning and scaling phase addresses any irregularities or data quality concerns. By splitting the dataset into training and testing sets, it facilitates the training of machine learning models on one set and their evaluation on another. The inclusion of dummy data expands our analysis scope, enabling the models to learn from more examples/data.

Figure 2. Screenshot of the code 1

The code starts by setting up experimental data for different enzymatic reactions. Each experiment is represented as an array containing parameters like initial substrate concentration, methanol concentration, peak area, conversion rate, productivity, and more. These arrays are different experimental scenarios provided in the original raw dataset that influence enzyme productivity. Each array is converted to a data frame, allowing a more efficient analysis.

Considering the small amount of data points, dummy data is created based from the mean and the standard deviation from a normal curve. A new DataFrame is created from the generated data.

Columns related to target values Conversion_Rate and Productivity are dropped from the dummy data dataframe due to the fact that they were randomly generated in the dummy data. The values will be calculated properly and inserted in the dataframe.

The data is then divided into training and testing subsets using train_test_split function, allowing model assessment. Here, we used test_size=0.3, meaning 30% of the data is reserved for testing.

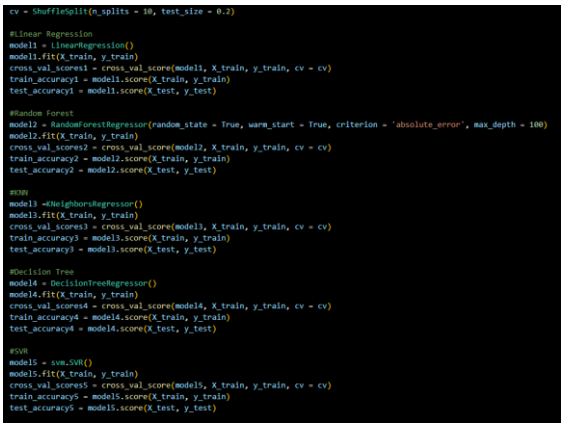
Then, the StandardScaler standardized (scaled) both training and testing data so that they have a mean of 0 and a standard deviation of 1. The transformed data is stored in X_train and X_test dataframes.

The purpose of the second component, model training, is to develop and train machine learning models that can predict outcomes based on the provided data. These trained models can then be used to make predictions on new, unseen data.

This component primarily utilizes machine learning algorithms and techniques provided by the scikit-learn library. These algorithms include Linear Regression, Random Forest, K-Nearest Neighbors, Decision Tree, and Support Vector Regression.

Cross-validation is performed to estimate how well the trained regression models are likely to perform on new data. The code used “Shuffle Split” to randomly shuffle the dataset and then split it into training and testing sets for multiple iterations.

Function: This component functions by creating various regression models, training them on the provided training data, evaluating their performance using cross-validation, and assessing their accuracy on both training and testing data. This allows the trained models to predict outcomes (i.e. conversion rate, productivity) with new input values.



```
cv = ShuffleSplit(n_splits = 10, test_size = 0.2)

#Linear Regression
model1 = LinearRegression()
model1.fit(X_train, y_train)
cross_val_scores1 = cross_val_score(model1, X_train, y_train, cv = cv)
train_accuracy1 = model1.score(X_train, y_train)
test_accuracy1 = model1.score(X_test, y_test)

#Random Forest
model2 = RandomForestRegressor(random_state = True, warm_start = True, criterion = 'absolute_error', max_depth = 100)
model2.fit(X_train, y_train)
cross_val_scores2 = cross_val_score(model2, X_train, y_train, cv = cv)
train_accuracy2 = model2.score(X_train, y_train)
test_accuracy2 = model2.score(X_test, y_test)

#KNN
model3 =KNeighborsRegressor()
model3.fit(X_train, y_train)
cross_val_scores3 = cross_val_score(model3, X_train, y_train, cv = cv)
train_accuracy3 = model3.score(X_train, y_train)
test_accuracy3 = model3.score(X_test, y_test)

#Decision Tree
model4 = DecisionTreeRegressor()
model4.fit(X_train, y_train)
cross_val_scores4 = cross_val_score(model4, X_train, y_train, cv = cv)
train_accuracy4 = model4.score(X_train, y_train)
test_accuracy4 = model4.score(X_test, y_test)

#SVR
model5 = svm.SVR()
model5.fit(X_train, y_train)
cross_val_scores5 = cross_val_score(model5, X_train, y_train, cv = cv)
train_accuracy5 = model5.score(X_train, y_train)
test_accuracy5 = model5.score(X_test, y_test)
```

Figure 3. Screenshot of code 2

The process begins by initializing a cross-validation strategy using the Shuffle Split method [14]. This strategy involves splitting the dataset into multiple subsets for iterations, with a portion reserved for testing and the rest for training. The code then proceeds to create instances of different regression models, including Linear Regression, Random Forest, K-Nearest Neighbors, Decision Tree, and Support Vector Regression.

For each model, it is first trained using the training data, allowing it to learn the relationships within the dataset. Next, cross-validation scores are computed using the training data. The training accuracy of the model, indicating how well it fits the training data, and the testing accuracy, reflecting its performance on testing data, are also calculated.

Throughout the process, variables are generated to hold the trained models, cross-validation scores, and accuracy metrics.

The entire process, from generating 1000 dummy data points to using the machine learning model to calculate accuracy, is iterated 100 times. After completing the 100 iterations, the final step involves calculating the average accuracy across all iterations. This average accuracy provides a robust assessment of the model's overall performance, taking into consideration its consistency across multiple runs with different data splits.

The Web Interface component serves as a front end of the system, providing an interactive platform for users to interact with their own data set and pre-integrated machine learning models.

This component utilizes web development languages such as CSS, JavaScript, php, etc. The website helps create a user-friendly interface that allows users to input data and receive predictions based on their new input (or experimental parameters) data points without requiring them to interact directly with the code or datasets.

Function: The Web Interface component accepts user inputs as a .csv file, allows users to pick on the machine learning model as a regressor or classifier, and then communicates these inputs to the backend of the system, where the machine learning models are deployed. The input data is then processed and used by the trained models to generate predictions. The results are sent back to the interface, which displays them in a user-friendly format as charts.

This component bridges the gap between the technical aspects of data analysis and the ease of use for users who may not have programming expertise.

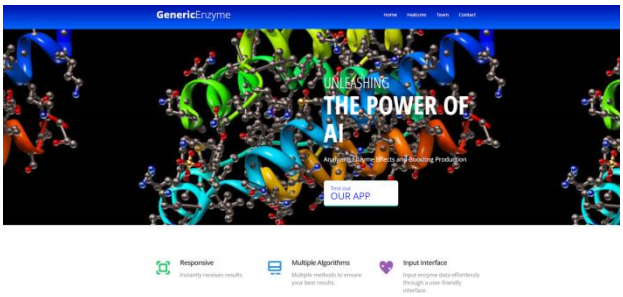


Figure 4. Screenshot of website

```
def training(data_index):
  data = data_store[data_index]
  for c in data.data.columns:
    if data.data[c].dtype == "object":
      data.data[c] = data.data[c].astype("category")
  '''target = data.data.columns[data.target]'''
  y = data.data[data.target.replace("checked", "")]
  ''picked_columns = [data.data.columns[index]for index in data.columns]'''
  X = data.data[data.columns]
  regression_list = [
    SVR(),
    LinearRegression(),
    SGDRegressor(),
    GradientBoostingRegressor(),
    ElasticNet()
  ]
  regression_names = [
    "Support Vector Regression", "Linear Regression",
    "Stochastic Gradient Descent Regression", "SGD with Gradient Boosting",
    "Elastic Net"
  ]
  classifications_list = [
    KNeighborsClassifier(3),
    SVC(kernel="rbf", C=0.025, probability=True),
    NuSVC(probability=True),
    DecisionTreeClassifier(),
    RandomForestClassifier(),
    AdaBoostClassifier(),
    GradientBoostingClassifier(),
    GaussianNB(),
    LinearDiscriminantAnalysis(),
    QuadraticDiscriminantAnalysis()
  ]
  classifications_name = [
    "KNC", "SVC", "DTreeC", "RForestC", "ABC", "GBC", "GNB", "LDA", "QDA"
  ]
  numeric_transformer = Pipeline(
    steps=[("imputer",
      SimpleImputer(strategy="median")), ("scaler", StandardScaler())])
```

Figure 5. Screenshot of code 3

This part of the code focuses on data preprocessing and model training for both regression and classification algorithms of the website.

4. EXPERIMENT

4.1. Experiment 1

A potential blind spot was the model's performance on unseen data. Model accuracy was evaluated using cross-validation and testing accuracy.

While the machine learning models were trained on the provided dataset, their ability to accurately predict outcomes on new data is crucial for generalization to real-world scenarios and practical applications.

The experiment will be set up by first partitioning the dataset into training and testing subsets.

To evaluate model performance accurately, a cross-validation technique called ShuffleSplit will be employed, splitting the data into multiple iterations of training and testing sets.

The training accuracy of machine learning models measures how well they fit the training data. While high training accuracy suggests that the model has learned the training data well, it's important to consider its ability to generalize to new, unseen data (or its testing accuracy). If a model has excessively high training accuracy but performs poorly on the testing data, it might be overfitting the training data.

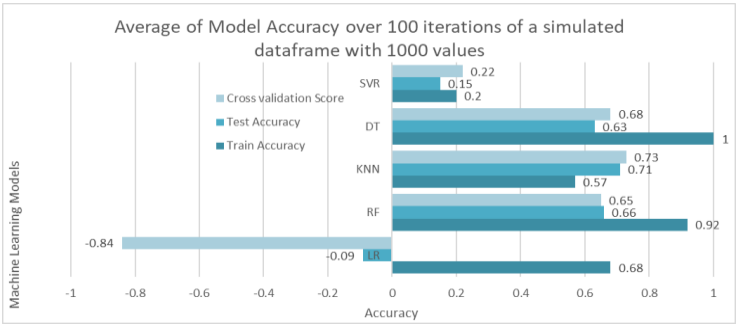


Figure 6. Average of model accuracy

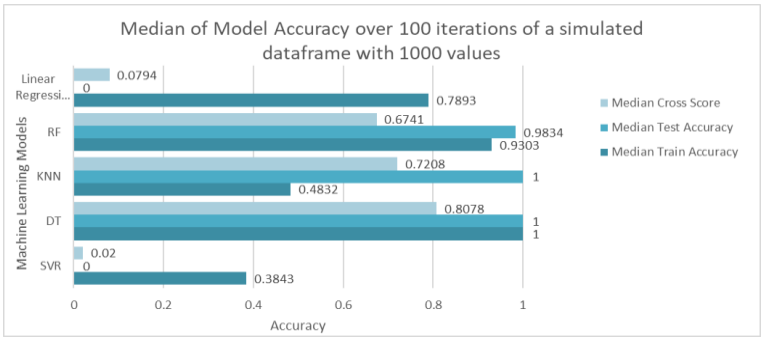


Figure 7. Median of model accuracy

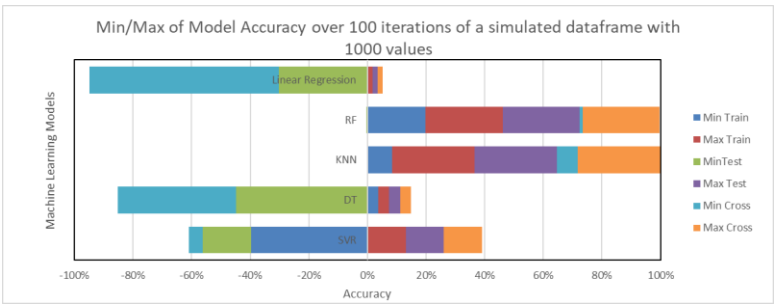


Figure 8. Min/Max of model accuracy

Average of 100 Iterations of Simulated Data

	Average Train Accuracy	Average Test Accuracy	Average Cross Score
SVR	0.2024	0.1539	0.225
DT	1	0.6361	0.6808
KNN	0.5743	0.7127	0.7343
RF	0.9249	0.6618	0.6554
Linear Regression	0.6816	-0.0997	-0.8483

Figure 9. Average of 100 iterations of simulated data

Median of 100 Iterations of Simulated Data

	Median Train Accuracy	Median Test Accuracy	Median Cross Score
SVR	0.3843	0	0.02
DT	1	1	0.8078
KNN	0.4832	1	0.7208
RF	0.9303	0.9834	0.6741
Linear Regression	0.7893	0	0.0794

Figure 10. Median of 100 iterations of simulated data

Min/Max of 100 Iterations of Simulated Data

	Min Train	Max Train	MinTest	Max Test	Min Cross	Max Cross
SVR	-3.0429	1	-1.2728	1	-0.3562	1
DT	1	1	-12.0653	1	-10.8266	1
KNN	0.2982	1	-0.0072	1	0.2532	1
RF	0.7571	1	-0.0132	1	0.0418	1
Linear Regression	0.0183	1	-17.7047	1	-37.9615	1

Figure 11. Min/Max of 100 iterations of simulated data

The highest test accuracy score achieved is 0.71 by the K-Nearest Neighbors (KNN) model, followed by Decision Tree Regressor and Random Forest with approximately 0.65, showcasing its effectiveness in making accurate predictions. However, due to some of the negative accuracy in DT, its consistency might be worse than RF and KNN.

Conversely, the lowest accuracy score is -0.09 by the Linear Regression (LR) model on the testing data and 0.15 by the Support Vector Regression model. This low accuracy could be due to the complex relationships in the enzymatic reaction dataset, which might not be well captured by the linear nature of the LR and SVR model. The small amount of data points might also restrain the effectiveness of LR and SVR.

Models like KNN, Random Forest and Decision Tree exhibit relatively high testing and training accuracy, while LR and Support Vector Regression (SVR) struggle to provide accurate predictions [13].

4.2. Experiment 2

Using a uniform distribution for generating dummy data could lead to a blind spot in the program. This blind spot arises from the fact that uniform distribution would introduce a completely random pattern into the data, potentially not capturing the actual underlying relationships present in the enzymatic reactions.

In contrast, opting for a normal distribution allows the generated data to align more closely with the mean and standard deviation of the original dataset, providing a better representation of the trends observed in real-world scenarios.

Ensuring that the generated dummy data follows a normal distribution is important because it enables the machine learning models to learn from data that mimics the actual experimental conditions.

To generate realistic dummy data that accurately represents the trends in enzymatic reactions, a normal distribution approach was employed. This choice was made to avoid introducing randomness and to ensure that the generated data aligns closely with the statistical characteristics of the original dataset.

The process of generating dummy data involved several steps. First, we analyzed the original dataset to determine the mean and standard deviation of each parameter. Next, we determined the number of dummy data points required for our analysis (which is 1000) and stored the generated values. For each parameter, we employed a normal distribution to generate random values centered around the calculated mean with a spread indicated by the standard deviation.

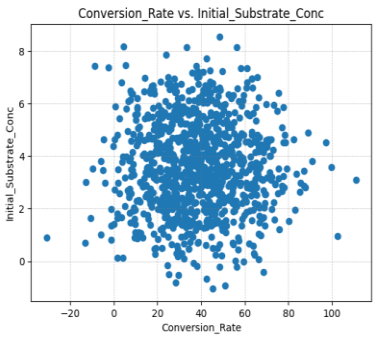


Figure 12. Generated dummy data with normal distribution

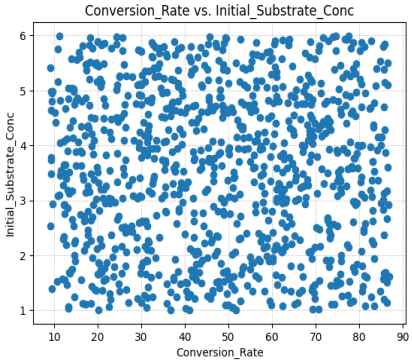


Figure 13. Generated dummy data with uniform distribution

4. RELATED WORK

The scholarly article titled "A Comprehensive Analysis of Machine Learning-Based Assessment and Prediction of Soil Enzyme Activity" aims to use machine learning techniques to predict soil enzyme activity based on various soil characteristics and properties [4]. The study focuses on assessing and predicting the activity of soil enzymes such as amylase and urease, along with evaluating soil physical and chemical properties like sand, clay, silt, etc. The researchers utilize a combination of Multiple Linear Regression (MLR), Random Forest (RF) models, and Artificial Neural Networks (ANN) to achieve their objectives.

The solution involves collecting a substantial dataset comprising soil properties and measurements of nine different soil enzyme activities. Machine learning models, including MLR, RF, Extra Tree classifier, and ANN, are trained on a subset of the data and evaluated on another subset. These models learn the complex relationships between soil properties and enzyme activities, enabling them to make predictions about enzyme activity levels based on input soil characteristics. This is much like how we trained our model using a training set and a separate testing set.

The solution's effectiveness is evaluated using various metrics such as the coefficient of determination (R2), mean absolute error (MAE), and root mean square error (RMSE). The study demonstrates that the Random Forest model is particularly effective in predicting soil enzyme activities accurately. The ANN model also achieves 99% accuracy in predicting cellulase enzyme activity. This scholarly source reinforces the results of this project in predicting and assessing enzyme activity analysis.

The solution's limitations include its dependence on the quality and representativeness of the dataset. If the dataset is biased or lacks diversity, the model's predictions may not generalize well to other regions or soil types. Additionally, the study does not delve into the broader environmental context, such as climate and land use, which can influence enzyme activity and crop productivity.

This scholarly paper titled “Predicting the effect of silver nanoparticles on soil enzyme activity using the machine learning method: type, size, dose and exposure time” addresses the prediction of the impact of silver nanoparticles (AgNPs) on soil enzyme activities using machine learning techniques [5]. The study aims to understand the effects of various factors such as AgNPs type, size, dose, and exposure time on soil enzyme activities. Three machine learning models are employed in the analysis: Artificial Neural Network (ANN) optimized with Genetic Algorithm (GA), Gradient Boosting Machine (GBM), and Random Forest (RF).

The study collects and preprocesses data from 721 sets of literature related to AgNPs properties and soil enzyme activities. The data is then used to train and optimize the machine learning models. The ANN model optimized with GA is found to be most suitable for capturing overall trends, while GBM and RF are more appropriate for analyzing smaller-scale effects. The study employs partial dependency profile (PDP) analysis to understand the influence of AgNPs properties on soil enzyme activity.

The outcomes of this paper further validate the efficacy of the Random Forest model, which aligns with the effectiveness of the RF demonstrated in this project.

5. CONCLUSIONS

While our project offers insights into enzyme productivity prediction, it does have some limitations. Firstly, the dataset size remains relatively small, potentially limiting the generalizability of our machine learning models [6]. To address this, acquiring a larger and more diverse dataset would enhance the models' robustness.

Additionally, while the current approach utilizes a range of machine learning models to capture various relationships, more advanced techniques like neural networks could be explored. Inspired by the two scholar papers mentioned earlier in this paper, enhancing my project's capabilities could involve integrating Artificial Neural Networks (ANN) to capture complex patterns within enzyme prediction analysis [4][5].

Furthermore, our dummy data generation process assumes a normal distribution for each parameter. This might not accurately represent all experimental scenarios, especially those with non-normal distributions. Implementing a more sophisticated data generation method that takes into account various distribution types and correlations would improve the generated data [7].

In conclusion, our project uses machine learning to predict enzyme activity and production. While there are limitations, such as dataset size and data generation assumptions, addressing these challenges can lead to more accurate predictions and better generalization.

REFERENCES

[1] Sarker, Iqbal H. "Machine learning: Algorithms, real-world applications and research directions." SN computer science 2.3 (2021): 160.
[2] Horton, John J., David G. Rand, and Richard J. Zeckhauser. "The online laboratory: Conducting experiments in a real labor market." Experimental economics 14 (2011): 399-425.
[3] Lawson, Christopher E., et al. "Machine learning for metabolic engineering: A review." Metabolic Engineering 63 (2021): 34-60.
[4] Shahare, Yogesh, et al. "A Comprehensive Analysis of Machine Learning-Based Assessment and Prediction of Soil Enzyme Activity." Agriculture 13.7 (2023): 1323.
[5] Zhang, Zhenjun, Jiajiang Lin, and Zuliang Chen. "Predicting the effect of silver nanoparticles on soil enzyme activity using the machine learning method: type, size, dose and exposure time." Journal of Hazardous Materials (2023): 131789.
[6] Rajput, Daniyal, Wei-Jen Wang, and Chun-Chuan Chen. "Evaluation of a decided sample size in machine learning applications." BMC bioinformatics 24.1 (2023): 48.
[7] Rajput, Daniyal, Wei-Jen Wang, and Chun-Chuan Chen. "Evaluation of a decided sample size in machine learning applications." BMC bioinformatics 24.1 (2023): 48.
[8] Field, Martin J. "Simulating enzyme reactions: challenges and perspectives." Journal of computational chemistry 23.1 (2002): 48-58.
[9] Chen, Ji, et al. "Leveraging machine learning for gaining neurobiological and nosological insights in psychiatric research." Biological psychiatry 93.1 (2023): 18-28.
[10] Newton, Matilda S., et al. "Enzyme evolution: innovation is easy, optimization is complicated." Current Opinion in Structural Biology 48 (2018): 110-116.
[11] Balogun, Abdul-Lateef, and Abdulwaheed Tella. "Modelling and investigating the impacts of climatic variables on ozone concentration in Malaysia using correlation analysis with random forest, decision tree regression, linear regression, and support vector regression." Chemosphere 299 (2022): 134250.
[12] Guerriero, Gea, et al. "Production of plant secondary metabolites: Examples, tips and suggestions for biotechnologists." Genes 9.6 (2018): 309.
[13] Smola, Alex J., and Bernhard Schölkopf. "A tutorial on support vector regression." Statistics and computing 14 (2004): 199-222.

[14] Putrada, Aji Gautama, et al. "Shuffle Split-Edited Nearest Neighbor: A Novel Intelligent Control Model Compression for Smart Lighting in Edge Computing Environment." Information Systems for Intelligent Systems: Proceedings of ISBM 2022. Singapore: Springer Nature Singapore, 2023. 219-227.

[15] Zhang, Shichao, Chengqi Zhang, and Qiang Yang. "Data preparation for data mining." Applied artificial intelligence 17.5-6 (2003): 375-381.