FINDING VARIOUS MATERIALS PROPERTIES BY MACHINE LEARNING MODEL

Hannan (Danny) Wang[1], JD Qian[1]

1. Department of Mechanical Engineering, University of Rochester

ABSTRACT

This project applies machine learning (ML) methods to predict three material properties (band gap, shear modulus, and bulk modulus) using data from the Materials Project（ MP database. Four ML models (Random Forest, Gradient Boosting, Linear Regression, and Neural Networks) were trained and evaluated with features extracted by an open-source descriptor tool. The study compares model performance to identify the most accurate approach for each property. Results show that Gradient Boosting often outperforms other ML models, while neural networks show varying accuracy depending on the property. This work demonstrated the potential of ML in the field of material science, offering insights into model selection for property prediction.

1. INTRODUCTION

The discovery and optimization of advanced materials are critical for technological progress in fields such as semiconductors and structural engineering. However, experimental determination of material properties, such as electronic band gap, shear modulus, and bulk modulus, can be very difficult and both cost and time-consuming. Computational approaches, like machine learning (ML), have shown up and serve as powerful tools for accelerating materials research by predicting properties directly from compositional and structural data.

The Materials Project (MP) database provides a massive amount of calculated and experimental material properties. In this project, we use this database to train and evaluate four ML models (Random Forest, Gradient Boosting, Linear Regression, and Neural Networks) for predicting

- **band gap**: an electrical property that determines a material's electrical conductivity by the energy difference between valence and conduction bands.
- **shear modulus**: a mechanical property that determines a material's resistance to deformation under shear stress.
- **bulk modulus**: a mechanical property that determines a material's resistance to uniform compression.

Feature extraction is performed using an open-source descriptor tool, which processes raw material data in the form of crystallographic information file (CIF) that is from MP into meaningful input [1].

The goal of this project is to compare the performance of different ML models in predicting material properties, identify the most suitable ML model for each property, and demonstrate the feasibility of using ML to predict material properties.

By applying ML methods to the field of material science, this work demonstrated its potential and offered insights into model selection for property prediction.

2. METHOD

The material properties we predicted in this project are band gap, shear modulus, and bulk modulus. The data for these material properties were obtained from a website that has a massive database for materials, computed by density function theory (DFT), called Materials Project (MP). Data extraction was performed using the MPRester (provided by the Pymatgen library) and API key (provided by the MP website). Each material entry was identified by its unique MP ID and the corresponding crystal structures were retrieved in the format of a crystallographic information file (CIF). These structures were then parsed using the CifParser class, enabling further feature extraction.

To transform the structural information into numerical inputs suitable for machine learning (ML), an open-source descriptor from an existing study was employed. This tool converts each CIF file into a vector of numerical features. These features include compositional descriptors (elemental fractions, atomic radii, electronegativity), structural descriptors (space group, coordination numbers), and electronic descriptors (valence electron counts, band structure features). The resulting feature vectors were used as inputs (X) for the machine learning models, while the target outputs (y) corresponded to the material properties of interest.

Four different ML models (linear regression, random forest, gradient boosting, and neural network) were implemented to predict these properties.

All models were trained using an 80/20 train-test split. There are, in total, 3760 data for band gap, 12960 data for shear modulus, and 12916 data for bulk modulus. The amount of data for training and testing is shown in the table below:

# data	Band Gap	Shear Modulus	Bulk Modulus
Training	3760	10368	10332
Testing	940	2592	2584

table 1) Number of training and testing data for each material property after the split

Model performance was assessed by mean absolute error (MAE), and coefficient of determination (R^2). Feature scaling was applied, particularly for models sensitive to input magnitudes such as neural networks and linear regression. In addition, hyperparameter tuning was conducted using grid search or randomized search methods to enhance model performance.

2.1. Linear Regression

Linear regression served as a baseline model to establish the minimum predictive performance. The model was trained on the scaled training data (X_train_scaled, y_train) to learn the relationship between the input features and the target variable. Predictions were generated on the scaled test set (X_test_scaled).

2.2. Random Forest

The random forest model, an ensemble of decision trees, was employed for its ability to capture nonlinear patterns and reduce overfitting through bootstrapping. The training process begins by defining a grid of hyperparameters (param_grid) that specify different configurations of the Random Forest model. These hyperparameters include the number of trees in the forest (n_estimators), the maximum depth of each tree (max_depth), the minimum number of samples required to split a node (min_samples_split), the minimum number of samples required at each leaf node (min_samples_leaf), and various options for feature selection (max_features). The grid also includes whether or not to use bootstrap sampling (bootstrap).

A RandomForestRegressor is implemented with the random_state for consistency. To optimize model performance, GridSearchCV is employed to perform an exhaustive search over all combinations of the defined hyperparameter grid.

After completing the grid search, the best combination of hyperparameters is retrieved using grid_search.best_params_, and the optimal Random Forest model is refitted on the entire training set (X_train_scaled, y_train). Once the model is trained, it is used to make predictions (y_predicted_rf) on the scaled test dataset (X_test_scaled).

2.3. Gradient Boosting

Gradient boosting was used as a more sophisticated ensemble technique that builds trees sequentially. The model used was XGBRegressor, a powerful tree-based ensemble method designed for efficient and accurate prediction in regression tasks. Similar to the random forest model, a comprehensive hyperparameter grid was defined, a RandomizedSearchCV was employed and the best-performing model was selected. Once the model is trained, it is used to make predictions (y_predicted_xgb) on the scaled test dataset (X_test_scaled).

2.4. Neural Network

A neural network was developed with multiple hidden layers, dropout regularization, and adaptive learning rate optimization, allowing for the capture of highly nonlinear relationships in the data. The network, defined in the NeuralNetwork class, consisted of an input layer, three hidden layers with sizes 512, 256, and 128 respectively, and an output layer predicting a single continuous value. The hidden layers used the ReLU activation function, which allows a small, non-zero gradient when the unit is not active, thereby helping solve the vanishing gradient problem. The output layer used a linear activation, appropriate for regression tasks. During the forward pass, each layer computed its weighted sum (z) and applied the activation function to produce the output passed to the next layer. In the training phase, mini-batch gradient descent was used, where the training data was shuffled and divided into batches of size 64 to update the weights iteratively over multiple epochs.

The training process also implemented early stopping: after each epoch, the model evaluated its loss on a separate validation set, and if no improvement in validation loss was observed over 20 consecutive epochs, training was terminated to avoid overfitting.

Before training, both the input features and the target values were standardized using StandardScaler. After training, the model was evaluated on both the training and testing datasets.

3. RESULT

The following graphs show the result after training by four machine learning (ML) models (linear regression, random forest, gradient boosting, and neural network). The accuracy would be represented by the coefficient of determination (R^2) and the error would be evaluated by mean absolute error (MAE). The higher the R^2, the higher the accuracy, meaning the better performance of the model.

3.1. Band Gap

The data distribution for the band gap is presented below in the form of a histogram.

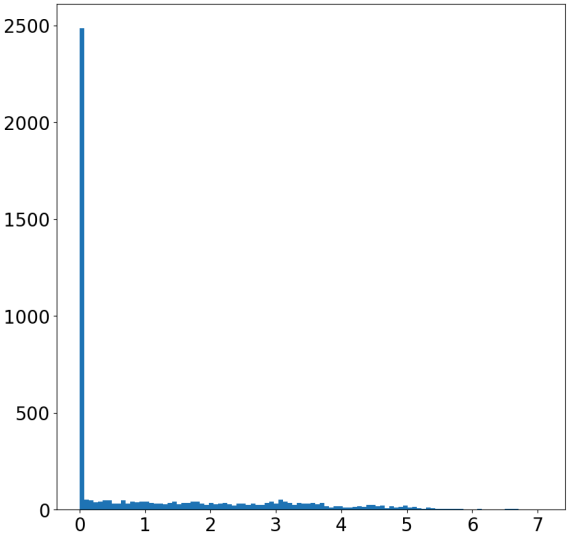


figure 1) The data distribution of the band gap in the histogram

The result of the four ML models is shown in the table below, with their accuracy and MAE.

Band Gap	Linear Regression	Random Forest	Gradient Boosting	Neural Network
R^2	41.80%	69.43%	71.05%	62.86%
MAE	92.83%	51.61%	49.46%	63.23%

table 2) The R^2 and MAE results for band gap in four different ML models.

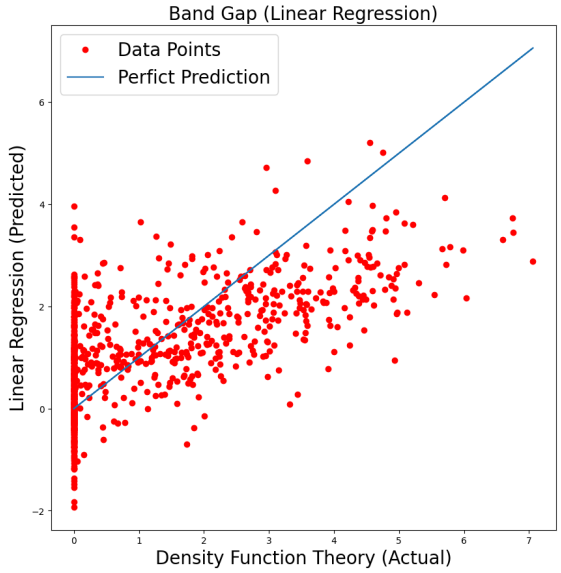


figure 2) the accuracy demonstration of band gap prediction after training by linear regression model.

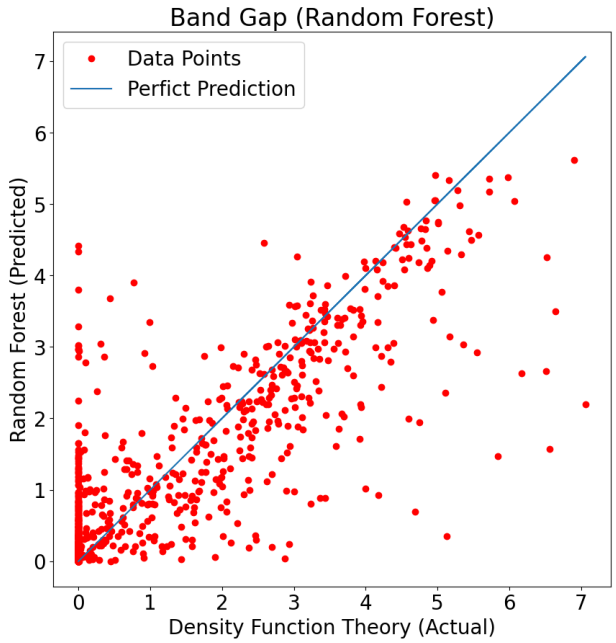


figure 3) The accuracy demonstration of band gap prediction after training by random forest model.

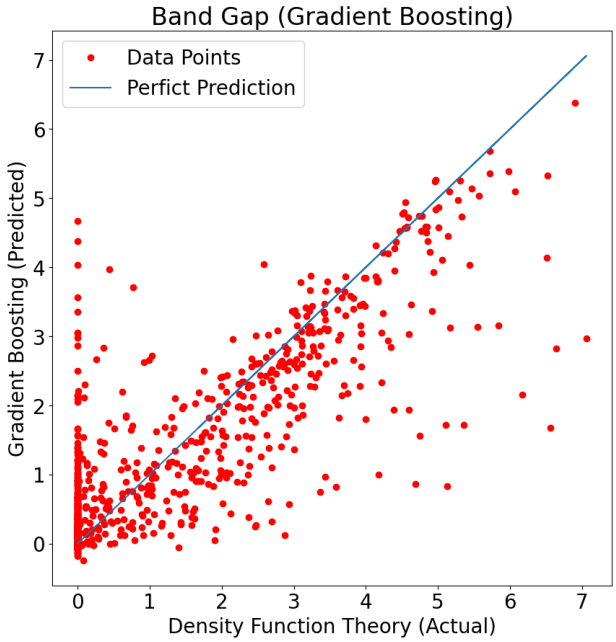


figure 4) the accuracy demonstration of band gap prediction after training by gradient boosting model.

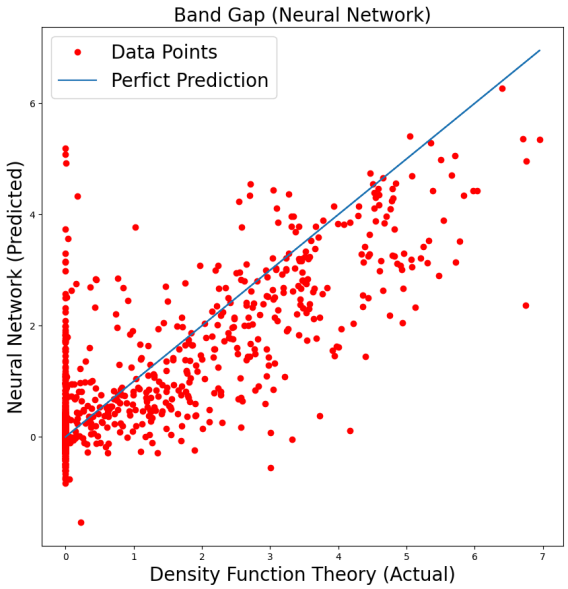


figure 5) the accuracy demonstration of band gap prediction after training by neural network model.

3.2 Shear Modulus

The data distribution for shear modulus is presented below in the form of a histogram.

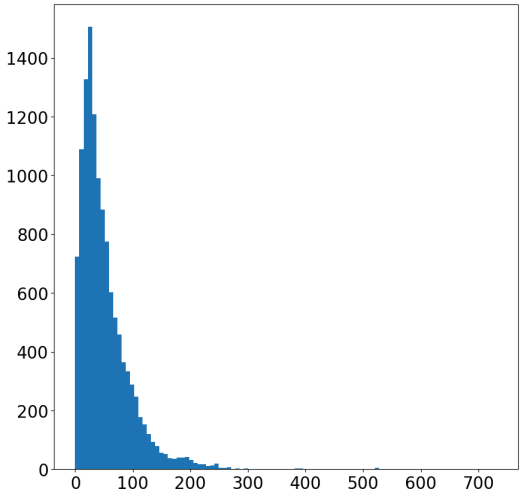


figure 6) The data distribution of the shear modulus in the histogram

The result of the four ML models is shown in the table below, with their accuracy and MAE.

Shear Modulus	Linear Regression	Random Forest	Gradient Boosting	Neural Network
R^2	N.A.	72%	81%	87%
MAE	N.A.	12.73	8.99	9.99

Table 3) The R^2 and MAE results for shear modulus in four different ML models.

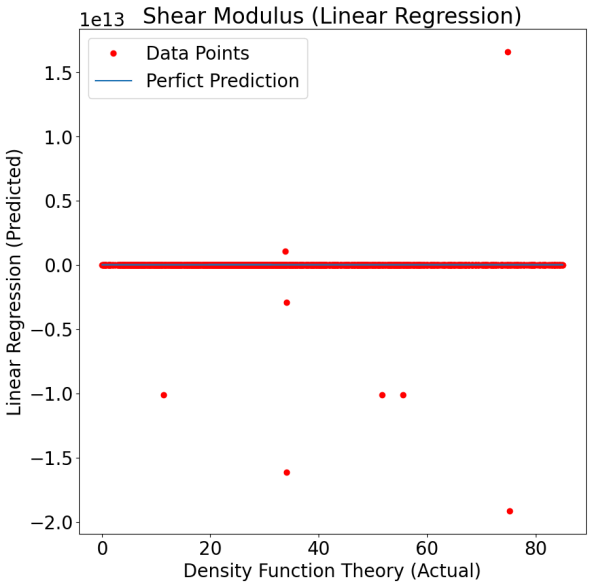


figure7) the accuracy demonstration of shear modulus prediction after training by linear regression model.

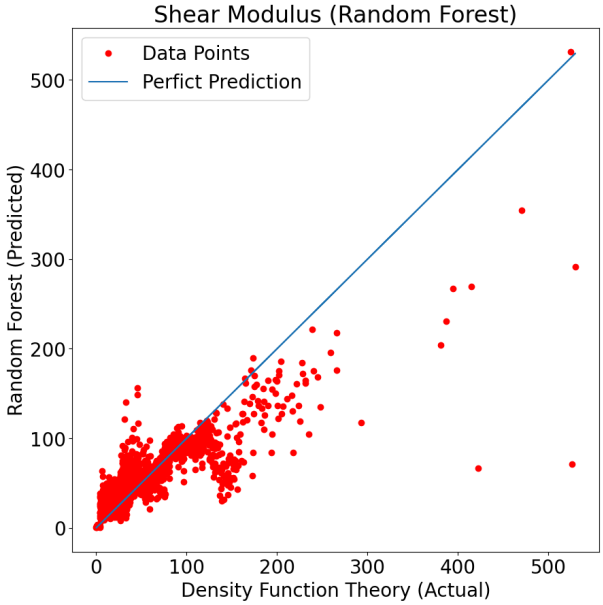


figure 8) The accuracy demonstration of shear modulus prediction after training by random forest model.

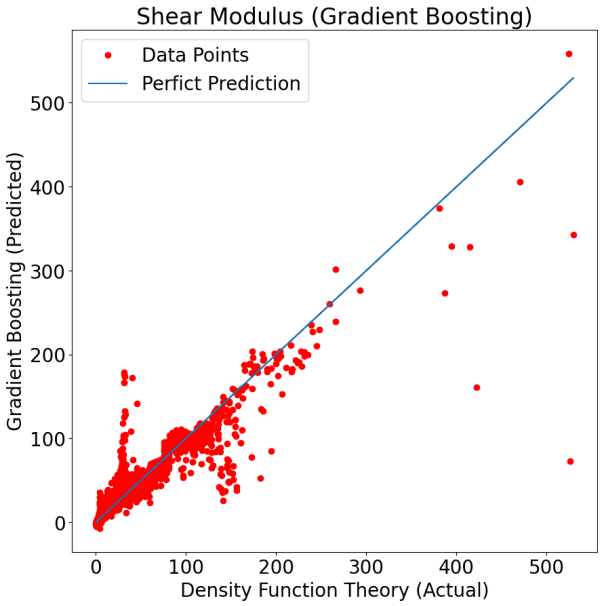


figure 9) the accuracy demonstration of shear modulus prediction after training by gradient boosting model.

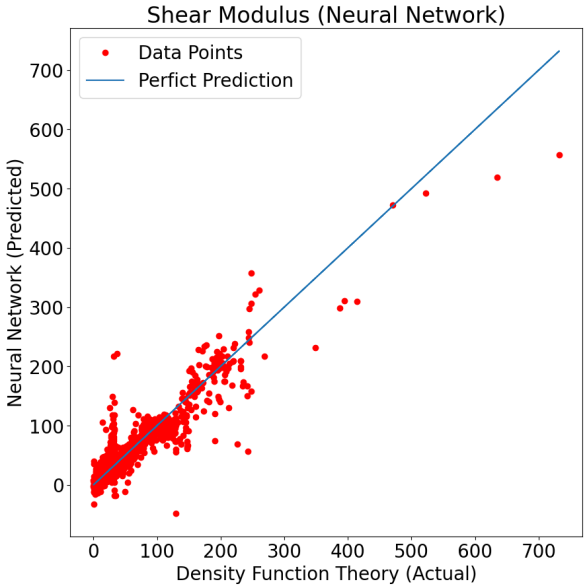


Figure 10) the accuracy demonstration of shear modulus prediction after training by neural network model.

3.3 Bulk Modulus

The data distribution for bulk modulus is presented below in the form of a histogram.

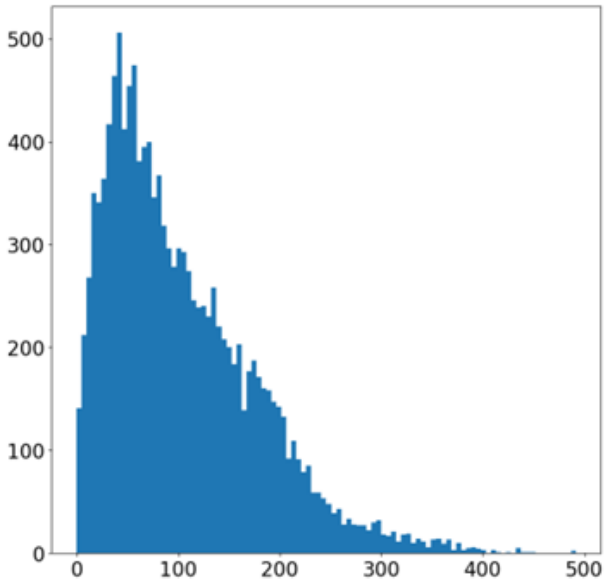


figure 11) The data distribution of the bulk modulus in the histogram

The result of the four ML models is shown in the table below, with their accuracy and MAE.

Bulk Modulus	Linear Regression	Random Forest	Gradient Boosting	Neural Network
R^2	68.10%	75.23%	85.40%	83.51%
MAE	28.59%	22.38%	16.14%	17.46%

Table 4) The R^2 and MAE results for bulk modulus in four different ML models.

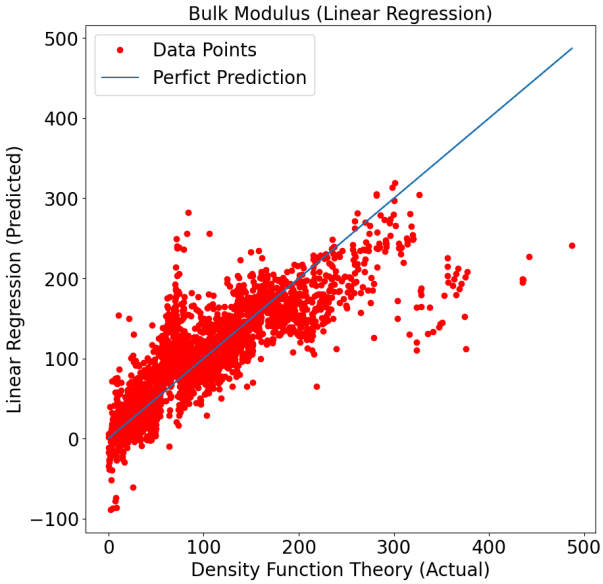


figure 12) The accuracy demonstration of bulk modulus prediction after training by linear regression model.

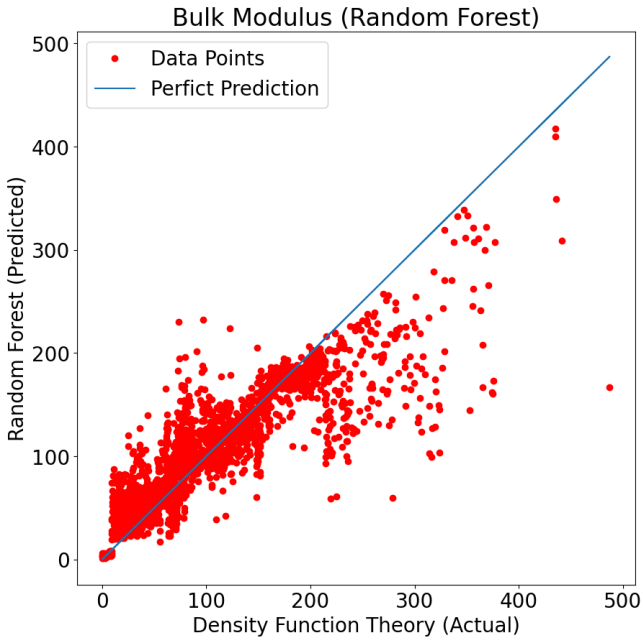


figure 13) The accuracy demonstration of bulk modulus prediction after training by random forest model.

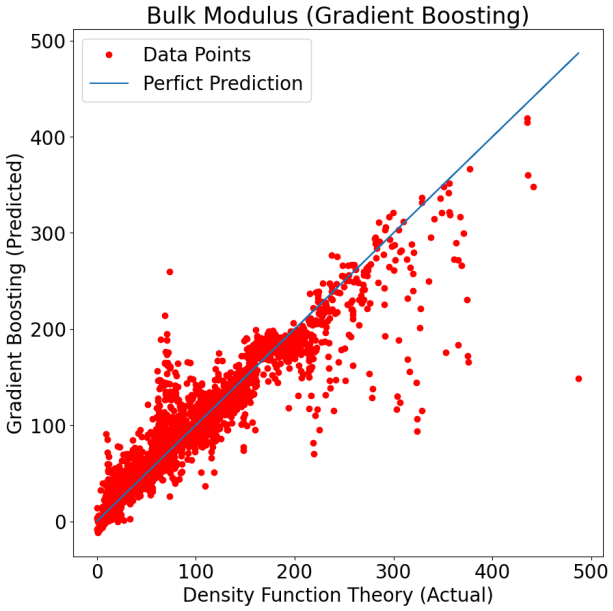


figure 14) The accuracy demonstration of bulk modulus prediction after training by gradient boosting model.

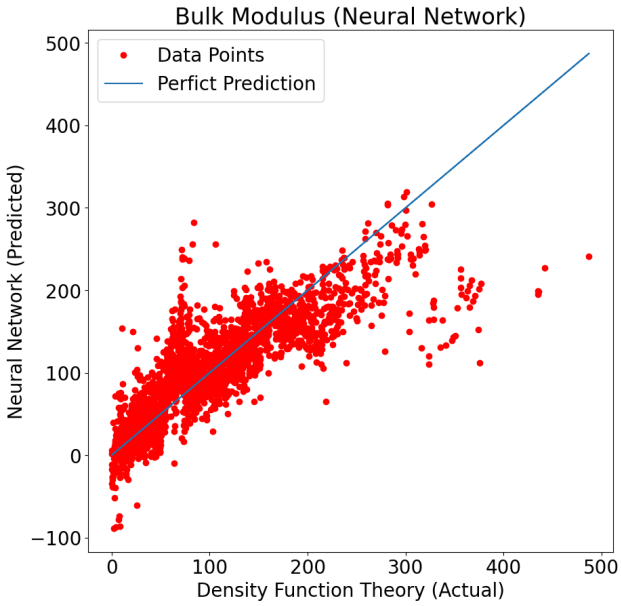


figure 15) The accuracy demonstration of bulk modulus prediction after training by neural network model.

4. DISCUSSION

The performance comparison of the machine learning models used in this study (linear regression, random forest, gradient boosting, and neural networks) shows significant differences in accuracy for material properties such as band gap, shear modulus, and bulk modulus. Among these models, linear regression showed the weakest performance, both in terms of mean absolute error and R^2 score. This was expected because it is limited to modeling only for linear relationships between features and the target variable. The material properties that focus in this study are often nonlinear and complex, involving interactions among multiple physical and chemical features. Linear regression, lacking the flexibility to capture these interactions or higher-order dependencies, tends to underfit the data, leading to poor generalization. This could be shown from the shear modulus prediction (figure 7), as the linear regression model predicts some extreme value from a nonlinear dataset to cause poor performance.

In contrast, models like random forest and gradient boosting performed significantly better. Random forest benefited from its ensemble approach, averaging multiple decision trees trained on random subsets of the data. This helped reduce overfitting and capture some degree of nonlinearity. Additionally, gradient boosting further improved performance by training trees sequentially, each one learning from the residual errors of the previous. This sequential refinement allowed the gradient boosting model

to capture more subtle relationships in the data and adapt better to the complex feature. Gradient boosting also naturally handles interactions between variables. For example, the effect of atomic radius might depend on the material's symmetry or bonding structure, which this model can represent using splits across multiple features. Hyperparameter tuning in both models results in a balanced and accurate model.

After standardizing the data and applying techniques, such as adaptive learning rates and early stopping, the neural network model has the highest predictive accuracy in both shear modulus and bulk modulus. The neural network's deep architecture enabled it to learn abstract, nonlinear feature interactions that are important in determining material properties. Unlike tree-based methods, neural networks can model continuous functions smoothly and scale well with increasing data complexity. Additionally, by employing ReLU activations, the network effectively resolved issues like vanishing gradients and unstable convergence.

Overall, the results demonstrate that models like gradient boosting and neural networks are better suited for predicting material properties because of their capacity to capture nonlinear relationships in material datasets. While linear regression provides a useful baseline, its assumptions are too simplistic for complex predictions in materials science. The improvement in accuracy observed in more complex models addressed the importance of using appropriate algorithms that align with the structure and complexity of data.

While the current models provide reasonable predictions for the target properties, future work should focus on improving predictive accuracy by adjusting hyperparameters. Furthermore, some of the models took 100-200 hours to complete training. Thus improving training efficiency should also be considered.

5. REFERENCES

[1] S. A. Tawfik and S. P. Russo, "Naturally-meaningful and efficient descriptors: machine learning of material properties based on robust one-shot ab initio descriptors," *Journal of Cheminformatics*, vol. 14, no. 1, Nov. 2022, doi: https://doi.org/10.1186/s13321-022-00658-9.