Next Materials 9 (2025) 101248

Contents lists available at ScienceDirect

Next Materials

journal homepage: www.sciencedirect.com/journal/next-materials

Research article

Applicability of machine learning approaches for predicting the phase formation in complex concentrated alloys

Kaven A.S., Subhasis Sinha, Surya D. Yadav *

Department of Metallurgical Engineering, Indian Institute of Technology (BHU), Varanasi, Uttar Pradesh 221005, India

ARTICLE INFO

Keywords:
Complex Concentrated Alloys (CCAs)
Histogram gradient Boosting Classifier
Gradient Boost Classifier
Phase Prediction
BCC
FCC
Intermetallic

ABSTRACT

This work deals with the machine learning techniques used to build predictive models to determine the phases in complex concentrated alloys (CCAs). Two different approaches were employed to determine the presence of phases. The one that relies on thermodynamic parameters was insight invoking majorly, and the other helped to predict phases for the specified alloy composition. Predictions were made using ensemble models and neural networks. Five machine learning (ML) algorithms were applied that are (1) K-nearest neighbors (KNN), (2) Support vector machines (SVM), (3) Gradient Boosting, (4) Ada-Boost Classifier, and (5) Histogram gradient Boosting Classifier. These algorithms were utilized for the prediction of Solid Solution (SS) and Intermetallic (IM) phases in CCAs. Among all the models, the Histogram Gradient Boosting Classifier model and Gradient Boosting model demonstrated the highest accuracies of 85.0 % and 83.7 %, respectively, for phase prediction with thermodynamic parameters. For phase prediction considering the alloy composition, the Histogram Gradient Boosting Classifier model obtained an accuracy of 84.8 %, while the Gradient Boosting model resulted an accuracy of 82.3 %. It was deduced that Histogram Gradient Boosting Classifier is the most significant model with respect to phase predictions in CCAs.

1. Introduction

Addition of secondary elements to a primary matrix has long been used to improve the properties of metals. In that direction a new category of alloys that are called complex concentrated alloys (CCAs)/high-entropy alloys (HEAs) have gained popularity in the past decades. This class of metallic materials often referred to as multi-principal element alloys (MPEAs), is distinguished by its unusual composition and very promising properties. CCAs are made up of a number of different elements in about similar atomic proportions as opposed to standard alloys, which often have one or two dominant elements and minimal alloying additives. CCAs have unique properties because of the simultaneous existence of several elements, which results in a high configurational entropy [1]. There are two major approaches to define CCAs, first one is composition-based and the second is entropy-based. Composition-based CCAs consist of five or more principal elements added, with each element's concentration ranging from 5 to 35 at. percent [2]. On the other hand, the entropy-based approach focuses on the configurational entropy, that should exceed 1.61 R (ΔS config $>$ 1.61 R), where R being the universal gas constant [3]. The scientific community has shown great interest in CCAs due to very promising combination of characteristics, including high hardness, good ductility [4], high-temperature strength [5], oxidation resistance [6], and good wear property [7], that are rare in conventional alloys. While some remarkable CCAs have been discovered, the search for new alloys with extraordinary properties continues to captivate researchers. In order to address these challenges, researchers have turned towards machine learning (ML)-based algorithms as an efficient approach to traditional design, analysis and predictions [8]. ML algorithms, which are data-driven and widely used in prediction and classification problems, offer a swift and cost-effective means of discovering new trends and predicting material characteristics. With sufficient data acquired from past experiments, ML models are developed to identify promising alloy candidates, predict microstructures, and estimate physical and chemical properties [9]. While ML models have shown good accuracy for phase prediction of CCAs, problem of computational time, complexity and interpretability still need to be addressed in better ways. Traditional methods based on empirical models [10], calculation of phase diagrams (CALPHAD) [11,12] and density functional theory [13] are advantageous in terms of adoptability, but have the limitations in terms of availability of complete

* Corresponding author.
E-mail address: surya.met@iitbhu.ac.in (S.D. Yadav).

https://doi.org/10.1016/j.nxmate.2025.101248
Received 26 February 2025; Received in revised form 15 September 2025; Accepted 18 September 2025
Available online 23 September 2025

datasets and computational cost [10,14]. On the other hand, ML models require a dataset but have the advantage of computational efficiency [15–17]. Previous studies have utilized ML models to predict the phases that form in CCAs by incorporating physical quantities that affect the phase formation in conventional alloys. These models have provided insights into the relative importance of different variables in determining the resulting phases [18–20]. Solubility was predicted employing ANN and revisiting Hume-Rothery's Rules [21]. It was deduced that Hume-Rothery's Rules work well for certain range of Ag and Cu alloys that too qualitatively, while ANN can also predict the solubility quantitatively. Back propagation ANN was used to predict the crystal structure (BCC or FCC) of cast high entropy alloys employing mixing enthalpy as most influential parameter. Other parameters that affected the crystal structure were difference in atomic size, electronegativity and valence electron concentration [22]. Machine learning approach was also used to predict the crystal structure and stability in quaternary high entropy alloys [23]. Presence of different phases was predicted for TiMnFeNi, MnFeCoNi and TiVMnNb alloys. Synergistic approach that combines gaussian statistical analysis and CALPHAD was used to design the CCAs [24]. NN in combination with ML predicted the phases in MPEAs and it was deduced that valence electron concentration plays the most vital role in determining the phases [25]. ML based approach using ab-initio data in combination with Monte Carlo simulation was used to predict chemical short-range order, phase stability and phase transition in NbMoTaW alloy [26]. Phase stability, phase transition and chemical short-range order was studied and a mechanism responsible for chemical order due atomic relaxation at room temperature was reported. K-NN, SVMs and ANNs in the framework of ML was used to predict the formation of solid solution, intermetallic or mixed solid solution and intermetallic phases in CCAs [27]. ANN performed well compared to SVM in terms of predicting the solid solutions and intermetallic phases, and its combination. NNs in the framework of ML was used to predict the quasi-phase equilibrium in Al-Cu-Mg alloy [28]. SVMs with ML was used to predict the stable BCC and FCC phases in CCAs [29]. Experimental data as well as first principle calculations for 20 quinary alloys were used for the validation. Combination of ML model and binary alloy phase diagrams data was used to predict the phases in CCAs [30]. ML based interatomic potential was used to predict the segregation, short-range order and strengthening mechanism in NbMoTaW alloy [31]. ML based interatomic potential in form of low rank tensor was used to deduce that, Fe and Cr manifest sublattice in the range of 600°C-1230°C for CoCrFeNi alloy [32]. ML in synergy with first principle calculation and neutron diffraction experiments was used to address the elastic properties of Al0.3CoCrFeNi alloy [33]. ML in combination with NN approach was used to predict the composition of AlCoCrFeMnNi alloy that leads to highest hardness [34]. A novel composition for AlCoCrFeNi alloys to get eutectic CCA was found employing ML and data mining [35]. ML with feature engineering was used for the prediction of phases that may form in various CCAs [36]. Insight to phase predictions was provided by ML models for HEAs employing thermodynamic parameters such as density, enthalpy and entropy of mixing, atomic size difference, valence electron concentration, electronegativity and melting temperature. It was deduced that just five parameters that are derived theoretically can helps for designing these alloys [37].

It can be said that all the previous studies have contributed significantly to this topic, but researches with a decent data set (1360) and thermodynamic and composition methods in combinations with ML models are rarely reported. Thus, this study aims to predict the phases using two different methods. The first method involves composition-based prediction of the CCAs, while the second method focuses on predicting the phases employing thermodynamic parameters such as entropy, enthalpy etc. The ultimate goal of our research is to identify the most suitable ML model that can effectively predict the formation of phases and contribute to the development and design of novel CCAs.

2. Dataset and machine learning models

2.1. Dataset

One of the major variable in ML based predictions is dataset [38]. The data used in this paper is closely associated with the previous work and research articles [39,40]. Dataset is derived from a careful extraction of microstructural observations documented in peer reviewed research articles focusing on CCAs. Its foundation lies on the previously published datasets by Miracle et al. [41], Couzinié et al. [42], and Ye et al. [43], incorporating their valuable contributions. The data set was utilized in two different was by splitting them on the basis of thermodynamic / configurational reliant features and composition. In the construction of these dataset, a rigorous selection process was employed to ensure the inclusion of only essential columns. These selected columns were then rearranged in a manner that aligns with the specific requirements of the dataset.

2.1.1. Dataset with thermodynamic parameters

In case of first dataset, the supplementary material accompanying this article contains a total of 1360 experimental CCA samples, with each sample represented by 13 columns, capturing various features and characteristics. These features encompass critical aspects of the multi-component alloy systems, alloy ID, alloy name, number of elements present, density calculated, enthalpy of mixing (dHmix), entropy of mixing(dSmix), Gibb's free energy of mixing (dGmix), melting temperature (Tm), the number of parameters (n.Para), atomic size difference, electronegativity, valence electron concentration (VEC) calculated using the rule of mixtures approach, experimentally observed microstructures (including body-centered cubic solid solutions - BCC_SS, face-centered cubic solid solutions - FCC_SS, dual-phase solid solutions - FCC+BCC_SS and intermetallic - Im). The inclusion of this dataset as supplementary material ensures accessibility, transparency, and reproducibility of the data, facilitating further exploration and analysis.

2.1.2. Dataset with composition of alloys

The second dataset given in the supplementary material accompanying this article comprises of a comprehensive collection of 1360 experimental CCAs samples, each containing 30 columns of valuable information. The first two columns of the dataset, namely Alloy_ID and Alloy, provide essential identification details for each entry, including the dataset entry itself and the corresponding alloy system. Columns 3–27 within the dataset contain information pertaining to the elemental compositions of the alloys. These columns are of utmost significance as they enable a deeper understanding of the alloys' constituent elements. Additionally, Column 28 serves to indicate the precise number of elements that constitute each alloy system (Num_of_Elem). Moving forward, Column 29 plays a vital role in documenting the microstructures or phases observed within the multi-component alloys. These observations, derived from the literature, have been categorized into various classifications. The microstructural observations include single solid solution phases, such as FCC or BCC, dual-phase solid solutions (FCC and BCC), as well as other intermetallic phases denoted as Im. Finally, the last column of the dataset, Column 30, offers an essential reference for each entry.

2.2. Visualization of dataset

In the process of analyzing datasets and extracting meaningful information, Python enabled pandas library offers valuable tools for this purpose. One critical aspect of understanding the dataset involves finding correlations among its features. This correlation analysis helps to establish connections and relationships between the input parameters. Python's seaborn package further enhances this exploration by providing a scatter matrix, which graphically represents the correlations between features. A scatter matrix allows for a comprehensive

visualization of feature correlations, enabling us to discern patterns and dependencies within the data. It provides a matrix of scatter plots, with each plot showcasing the association between a pair of features. As we have discussed, the dataset that is being used for the training is unbalanced with four classes, BCC_SS, FCC_SS, FCC_PLUS_BCC and intermetallic (Im).

Fig. 1 shows the data split into different categories. It can be observed that the amount of database related to intermetallic is less compared to the solid solutions. Fig. 2 represents the Violin plot depicting the correlation between atomic size differences and probable phases. It can be observed that the BCC formation may become increasingly preferable over FCC in CCAs with increasing atomic size difference (which can be most effectively revealed through the Fig. 2). The median and interquartile range gives us an understanding of how the phases are being preferred with increasing atomic size difference.

Fig. 3 shows the variation of entropy of mixing with respect to the valence electron concentration for different phases. Though there is no fixed trend for the same, we could observe the uniqueness in each phase for the given valence electron concentration and entropy. We can observe that the lower valence electron counts results in alloys with BCC_SS structure.

Considering the dataset used for phase selection, Fig. 4 illustrates the relationship between its characteristics, that are different columns of the dataset. Specifically, it shows a diagonal matrix that projects inseparable features, underscoring the significance of each feature in predicting the phase and structure. By examining the dataset as a whole, the dots in the scatter matrix reveal not only the individual values of data points but also patterns that may exist.

Over plotting can be visualized that happens once the data points overlap in such a way where it becomes difficult to distinguish the links between the dots and the associated variables, which can be observed in Fig. 4. Additionally, In Fig. 4 multiple data points are densely grouped together in a compact area, comprehending their proximity or density becomes more challenging. By leveraging Python's pandas library and seaborn package it was possible to extract valuable insights from datasets, understand feature correlations through scatter matrices.

2.3. Machine learning models

In material science domain various ML algorithms [44] have been used and they consist of both, benefits as well as drawbacks. Thus, it is essential to select the right one based on the aim and available dataset size. In case of small size dataset, simple linear regression works well as compared to a deep neural network model due to bias-variance tradeoff. We will only briefly discuss a few representative ML algorithms here because there are already many dedicated research papers on the topic. In this work, predictions were made using ensemble models as they are robust in handling data and help to avoid overfitting. The core idea of ensemble models is that many different models can frequently outperform one another with regard to both knowledge and predictions. This is more correct, especially when the separate models have different

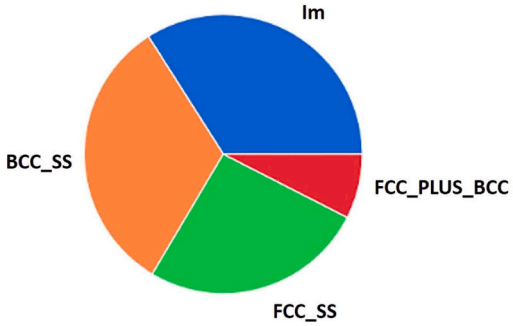

Fig. 1. Presence of different phase class in the dataset.

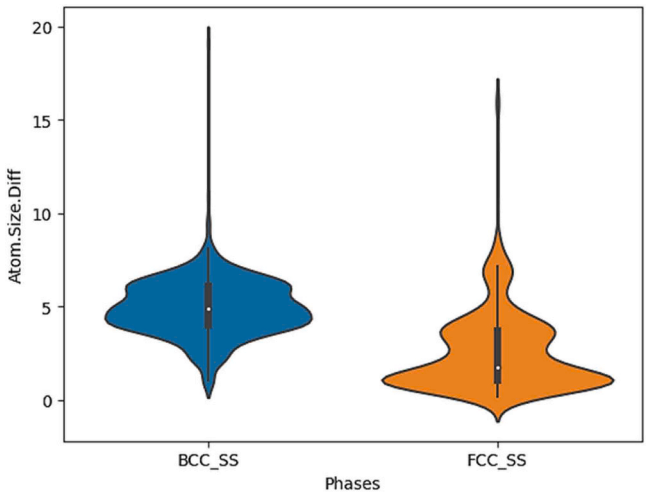

Fig. 2. Violin plot representing the phases on X-axis and atomic size difference on Y-axis.

strengths and shortcomings. Due to their increased ability to tackle complicated issues, enhance prediction accuracy, and minimize overfitting, ensemble models have begun to gain popularity in machine learning. By utilizing the combined knowledge of several models, ensemble models can improve forecast accuracy. They work best when each model has a variety of viewpoints or depth of knowledge regarding various facets of the issue. Due to the fact that they minimize the impact of outliers or data noise, ensemble models are found to be less vulnerable to overfitting. They can reduce the biases and misconceptions of individual models by incorporating numerous models, leading to more reliable predictions. Ensemble models typically perform better on new data because of their improved generalization ability. There are different types of ensemble models that are XGBoost, LightGBM, CatBoost, Bagging, Boosting, random forest, AdaBoost, Bootstrapping and Gradient boosting. In general models such as Random Forest, XGBoost, LightGBM, and CatBoost are indeed powerful and widely used in materials informatics. However, we did experiment with several of these classifiers during the model development phase and out of these ensemble models, the ones producing relatively better results are discussed in the next subsection.

2.3.1. Bagging

Bagging is a form of ensemble learning which includes individual training of several models on various subsets of the training data [45]. Herein, random sampling method is utilized to create subsets with the replacement from the original training dataset to train each model. By combining the predictions of each separate model, through majority voting (for classification problems) or averaging (for regression problems), the final forecast is generated. Fig. 5 depicts the functioning of Bagging models and it can be observed that the training in bagging begins by dividing up the training data into various subsets using replacement samples chosen at random. Each subset has the same number of cases as the original training set but may include duplicates or exclude some examples. Several independent models are trained using each subgroup separately. The models can be trained using the same learning process, but they have distinct training data due to random sampling. As a result, various models are produced that represent various facets of the data. Since, our objective is to classify alloys into groups, the predictions of many models are combined by majority vote. Each model makes a class label prediction for a certain input, and the class label with the most votes is selected as the final prediction. By merging numerous models that have been trained on various subsets of the data, bagging helps to lower the variance of the predictions. The final prediction of the ensemble is less likely to be impacted if one model

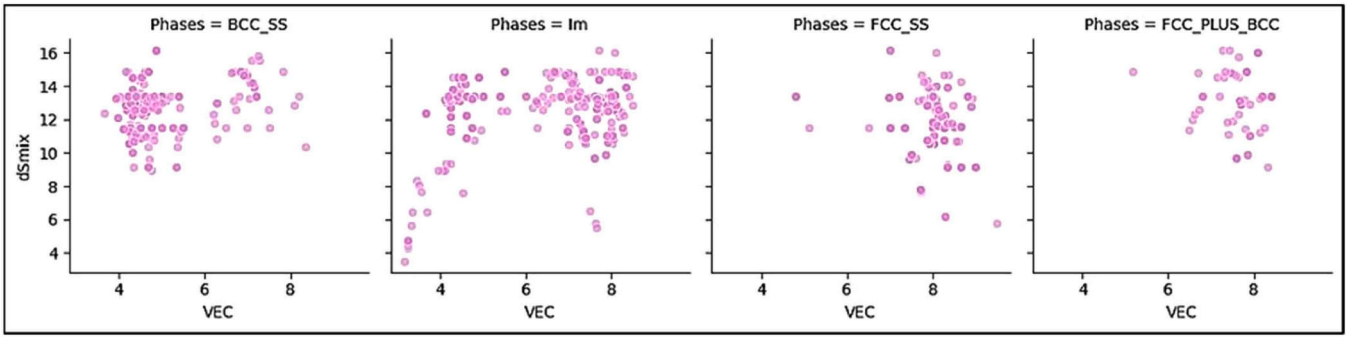

Fig. 3. Entropy of mixing (dSmix) varying with valence electron concentration (VEC) for each class.

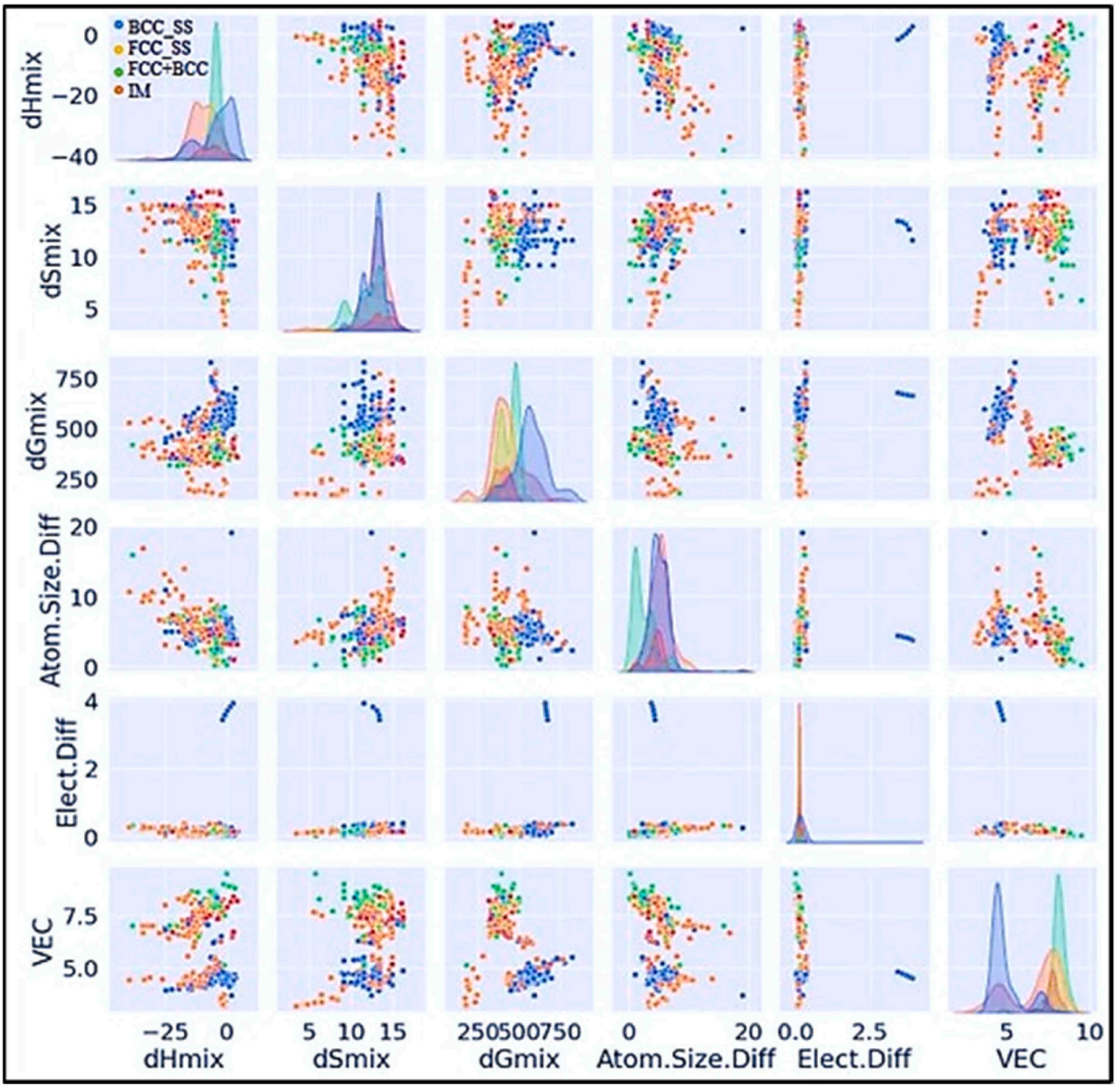

Fig. 4. Relationship between columns of the dataset.

makes inaccurate predictions as a result of model constraints or noise [45].

2.3.2. *Gradient boosting classifier*

Gradient Boosting is a prominent machine learning method for classification and regression applications. In an ensemble technique, numerous weak learners typically decision trees are combined to

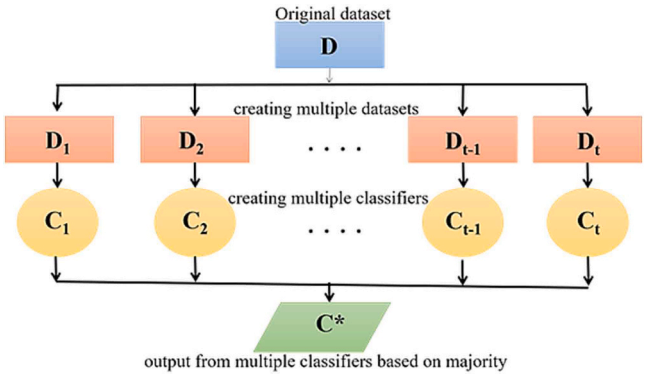

Fig. 5. Depiction of how bagging models work.

produce a powerful prediction model. Gradient boosting's main principle is to iteratively train new models that fix the errors generated by older ones while minimizing the loss function. Initializing a base model, often a decision tree that produces initial predictions based on the input features, is the first step in gradient boosting. The gradient of the error function concerning the current model's predictions needs to be calculated. The amount and direction of the inaccuracy are represented by this gradient, which shows how much the model's predictions need to be modified. By fitting the negative gradient of the loss, a new weak learner (decision tree) is trained to minimize the loss function. The prior model made mistakes that the new model is intended to fix. The contribution of each model to the final ensemble is determined by multiplying the predictions of each new model by a learning rate. The ensemble converges more slowly with a lower learning rate. The new model's predictions are added to the ensemble, and the method is repeated until a convergence criterion is satisfied, for a predetermined number of iterations. By combining the strengths of several weak learners, gradient boosting frequently achieves great prediction accuracy. The mathematical intuition behind the algorithm is discussed below.

We initialize a model $F_0(x)$ as,

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma) \tag{1}$$

where, $L(y, F(x))$ is a loss function, $i = 1, 2, \ldots n$ is the number of points in our training dataset and γ is the regularization parameter.

Further, the pseudo-residuals (r) can be estimated for the given data point as,

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{n-1}(x)}, \tag{2}$$

Herein F is the Hypothesis function. For the above calculated r_{im}, we fit a Machine learning model using (x_i, r_{im}) for all points from our dataset. The model is updated using Eq. (3) as,

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x), \tag{3}$$

whereas γ_m can be defined as,

$$\gamma_m = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)) \tag{4}$$

In Eq. (4), m is the number of models and by iterating it from $m = 1$ *to* M, we obtain our optimal model. It may detect nonlinearities and intricate correlations in the data and can manage various data formats, including categories, text, and even numerical data. It offers a metric for feature relevance, highlighting the attributes that have the most significant influence on the model's predictions. Gradient boosting is less prone to overfitting than a single decision tree as it successively trains models that fix the flaws of the prior models. The implementation of gradient boosting offers a number of hyperparameters that can be tuned to regulate the model's performance and behavior [46,47].

2.3.3. Histogram gradient boosting

A gradient boosting variation called Histogram Gradient Boosting (Histogram GB) uses histogram-based methods to augment the algorithm's effectiveness and scalability. It is made to work with enormous datasets and is especially useful for issues involving a lot of features or high-dimensional data. Histogram GB splits the data into histograms rather than working with the individual data points. It divides the feature range into discrete bins and then creates histograms for each feature. To ensure a balanced depiction, the data distribution is examined and divided into equal-width bins or utilizing quantiles. The calculation of gradients and loss functions is made efficient by the histogram representation. Histogram GB computes the gradients and loss values using histogram statistics instead of computing gradients for each data point. As a result, the training process is significantly sped up, and the computations required are decreased. In order to construct new decision tree nodes, it searches for the optimal split points during the boosting phase. Examining the histogram statistics and analyzing the loss function's progress evaluate probable split sites. The program can investigate potential splits more effectively and choose the best branching decisions by working with histograms rather than individual data points. Histogram GB uses regularization techniques to reduce overfitting and promote generalization. To balance model complexity and performance, regularization parameters, including learning rate, tree depth, minimum samples per leaf, and the maximum number of leaves, can be adjusted. Large-scale datasets can be handled because the histogram-based representation uses substantially less memory than storing individual data points. Histogram GB speeds up training and inference by using effective algorithms for histogram-based operations and acting on histograms, allowing for quicker model building and deployment. A wide variety of data formats, including categorical and numerical features, can be handled by Histogram GB. Missing values are easily accommodated within the histogram structure. Histogram Gradient Boosting is a powerful method that combines the advantages of gradient boosting with computations based on histograms to create machine-learning models that are effective, scalable, and accurate [48].

2.3.4. K-nearest neighbors (KNN) algorithm

The KNN approach is based on the idea that, associated data points have the similar labels or values. When a new data point has to be categorized or forecasted, the algorithm finds the K closest neighbors using a distance metric of choice, commonly Euclidean distance, alternative distance metrics or Manhattan distance. These distance metrics can be utilized to better suit the data characteristics based on problem requirements. The user-specified value of K defines the number of neighbors to examine.

$$\text{Euclidean Distance}(\mathrm{L}, \mathrm{M}) = \sqrt{(L_1 - M_1)^2 + (L_2 - M_2)^2 + \ldots + (L_n - M_n)^2} \tag{5}$$

$$\text{Manhattan Distance}(\mathrm{L}, \mathrm{M}) = |L_1 - M_1| + |L_2 - M_2| \tag{6}$$

$L_1, L_2 \ldots\ldots L_n$ are dimensional components of point L and $M_1, M_2 \ldots\ldots M_n$ are dimensional components of point M.

The method analyzes the labels or values of the K nearest neighbors to create a forecast and assigns the majority class label for classification tasks or calculates the average value for regression tasks. In classification, the new data point is allocated to the class label with the highest count among the K neighbors. The prediction in regression is the average or weighted average of the values of the K neighbors.

Fig. 6 shows the functioning of K-nearest neighbors algorithm. It can be observed that the query data point's distance is calculated from all the points and the query point is labeled by majority class out of the k nearest neighbors. Advantage of KNN algorithm is its simplicity to comprehend and apply, making it excellent for application.

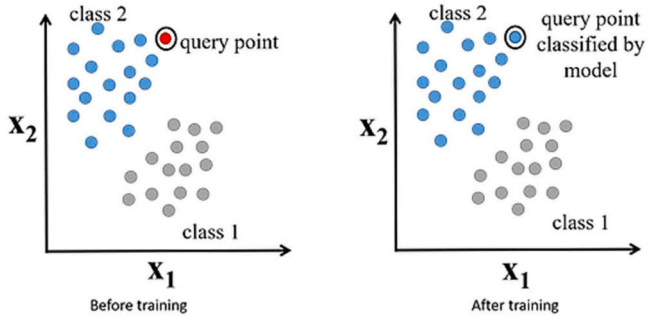

Fig. 6. Depiction of working of K-Nearest Neighbors algorithm, for a K value of 3, it assigns the new data point (red colure) to blue class of data because all 3 of its first 3 neighbouring elements are from blue class. Blue points depict the class 1 and grey points depicts class 2.

Furthermore, because it saves all of the training data in memory, KNN does not require training. This enables incremental learning, in which new data points may be added to an existing dataset without retraining the model. It is critical to choose an adequate distance metric and an ideal K value. Before implementing the KNN method, it is usual practice to preprocess the data by normalizing features and addressing missing values. In addition, assessing the performance of the KNN model often entails dividing the dataset into training and testing sets. The training set is used to construct the model, whereas the testing set is utilized to evaluate its accuracy and generalization skills [47,48].

2.3.5. Support vector machine (SVM)

The basic idea behind SVM is to find the best possible decision boundary that separates different classes of data points. It considers attempting to distinguish two groups of points in a dataset by means of a line. Fig. 7 depicts the important parameters used by SVM algorithm.

It can be seen that margin or the separation between a line and its nearest neighbors in each class, is what the SVM seeks to maximize. Support vectors are used to describe these nearest places. SVM function by upscaling the input data into a higher-dimensional space that allows for a linear separation. A kernel is a type of mathematical function that is used to do this transformation. Complex patterns that are not linearly distinguishable in the original feature space can be handled by SVMs with the help of kernel. After processing the data, SVM identify the best hyperplane, a line in higher dimensions that best divides the classes by the highest margin. Commonly used SVMs are of two types, i.e., linear SVM and non-linear SVM, that are further explained below,

Linear SVM: The linear SVM deals with linearly separable data and the objective is to identify a hyperplane that splits the two classes with the maximum margin. Mathematically, a hyperplane is represented as,

$$w.x + b = 0, \tag{7}$$

where w is the normal vector, x represents feature vector, and b is the bias term. The decision boundary is determined by the sign of the expression $w.\ x + b$, which is shown in Fig. 8.

Non-linear SVM: In cases, where data is not linearly separable, SVM uses the kernel trick to identify the original feature space to a higher-dimensional space so that linear separation becomes possible. The most common kernel functions that used are linear, polynomial and radial basis function (RBF) kernel. The kernel function estimates the inner products between pairs of data points in the high-dimensional space.

The other two important aspects are support vectors and optimization. Support vectors are the data points closest to the decision boundary, which directly influence the placement of the decision boundary, which are shown in Fig. 9. Only support vectors are relevant for defining the decision boundary, making SVM memory-efficient. The SVM formulation involves solving an optimization problem to find the optimal hyperplane.

The objective is to minimize $||w||^2/2$ subject to the constraint,

$$y_i(w.x_i + b) \geq 1, \tag{8}$$

for all training instances (x_i, y_i). This is achieved using quadratic programming techniques [49].

2.3.6. Adaptive boosting classifier

Fig. 10 depicts the working of Adaptive Boosting (AdaBoost) algorithm. The key idea behind AdaBoost is to iteratively train weak classifiers and give more importance to misclassified examples in each iteration, thereby focusing on difficult examples and improving overall accuracy. The algorithm starts by assigning equal weights to each training example. A weak classifier, such as a decision stump (a one-level decision tree), is trained on the dataset. The weak classifier's performance is evaluated, and the misclassified examples are given higher weights for the next iteration. In the subsequent iteration, another weak classifier is trained on the updated dataset. Again, the misclassified examples are assigned higher weights, and the process continues iteratively. The weights of the weak classifiers are adjusted based on their performance, giving more weight to classifiers with higher accuracy as shown in Fig. 10.

Adaboost classifier, $D_1(i)$ is initialized to $1/m$ where $i = 1, 2, \ldots m$ given that there are m training data, which means initialize all weights of your samples to 1 divided by number of training sample. Initially the weak learner is trained using distribution D_1. The weak hypothesis is chosen by looking at the weighted error. It updates the distribution using the Eq. (9), for all the data points in the training data as,

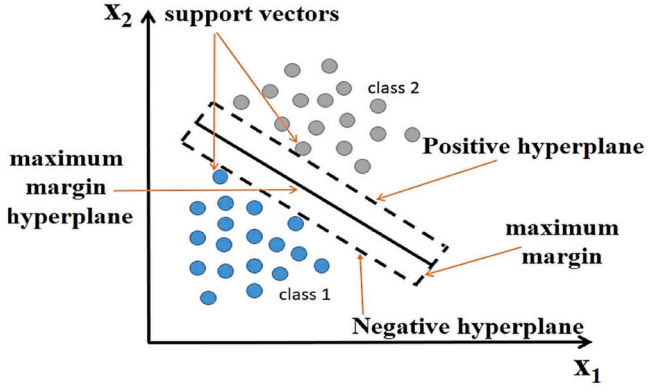

Fig. 7. Important parameters of Support Vector Machine algorithm. Blue points represent the output class 1 and grey points represent the output class 2.

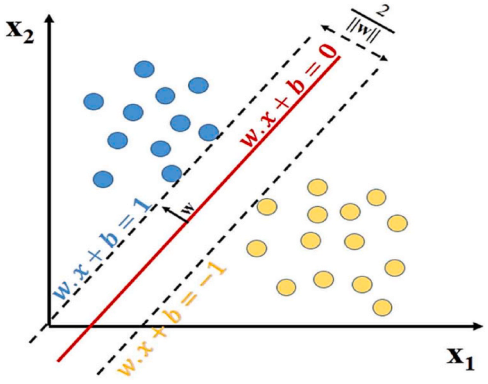

Fig. 8. Equations of hyperplanes and margins distance. Blue points represent the output class 1 and yellow points represent the output class 2.

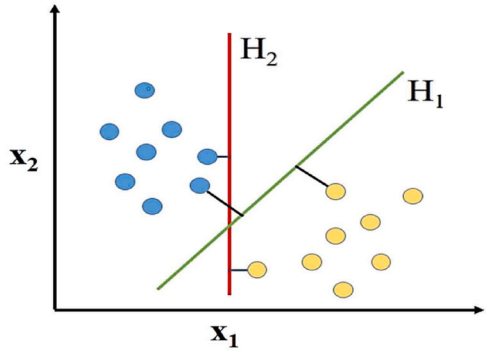

Fig. 9. The closest point of both classes to the hyperplane is the support vectors, which is shown in image. Blue points depict the output class 1 and yellow points depict the output class 2.

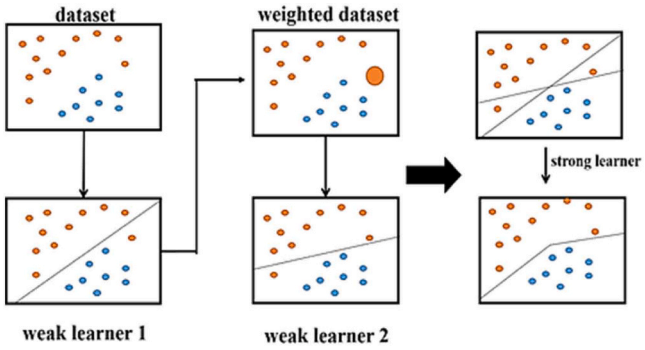

Fig. 10. Depiction of working of Adaptive Boosting algorithm. Blue points represent the output class 1 and orange points represent the output class 2.

$$D_{t+1}(i) = \frac{D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}, \tag{9}$$

where, t = 1,….,T

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{10}$$

$$\varepsilon_t = \Pr_{i\sim D_t}\left[h_t(x_i) \neq y_i\right] \tag{11}$$

Herein, (x_i, y_i) are the training data point, *Pr* is the probability, *D* is weights of each sample, m is the number of training datasets, *T* is the total number of classifiers, h_t is the hypothesis of the weak learner, Z_t is the Normalization factor to make D_{t+1} a distribution, ε is knows as minimum misclassification error, α is the weight for the classifier.

From this we calculate final hypothesis (*H*) for our model using the Eq. (12) as,

$$H(x) = \text{sign}\left(\sum_{t=1}^{T}\alpha_t h_t(x)\right) \tag{12}$$

We repeat the above steps *T* times and iterative over them to get our final model of the above Eq. (12). During the training process, AdaBoost effectively learns to combine the weak classifiers' predictions. Each weak classifier is assigned a weight based on its performance. In the final classifier, the weighted predictions of all the weak classifiers are combined, and the class with the majority vote is selected as the final prediction. By focusing on the misclassified examples in each iteration, AdaBoost gives more attention to difficult instances that are challenging to classify correctly. The final strong classifier produced by AdaBoost is a weighted combination of weak classifiers, where the weights are determined by their individual performance. This ensemble of weak classifiers leads to improved accuracy and robustness compared to using a single classifier [50].

3. Approaches for phase predictions

Fig. 11 depicts the flow diagram of present ML based modelling approach for phase predictions in CCAs. In this work, the ML modelling starts from the scratch by importing essential libraries from python, such as panda, seaborn, matpoltlib, numpy and sklearn. Then the dataset is imported and visualized for insights within the data. Furthermore, the dataset was split into two parts using "train_test_split" from sklearn library, which is (1) training data, having 70 % of original data and (2) test data, which has 30 % of the dataset. This dataset undergoes scaling to make it less computational complex and to improve runtime. In the next step, data is being fitted into various ML model for training. The trained ML models are tested using the test data and evaluated using common evaluation metrics like accuracy, F1 score, recall, and precision. The best model is chosen based on these evaluation metrics. The final block of code allows users to enter input to the code manually. The prediction on that input can be done using the best evaluation model and the output is printed.

3.1. Thermodynamic and configurational parameters reliant predictions

In order to determine the phases and crystal structure in CCAs employing thermodynamic and configurational parameters, it is essential to have prior knowledge of the values associated with those parameters. In the case of the MoNbTiV0.3Zr system, there are ten input parameters that are fed into a machine learning (ML) model, which are the number of elements present, density calculated, enthalpy of mixing (dHmix), entropy of mixing (dSmix), Gibb's free energy of mixing (dGmix), melting temperature (Tm), the number of parameters (n.Para), atomic size difference, electro-negativity and the valence electron concentration (VEC). These values are interpreted by the ML model as an equation by implementing decision-making processes through the ML algorithm and the predicted output is printed. As an example, in the specific case of the MoNbTiV0.3Zr system, the input was "5.0 8.569015–2.125000 12.967078 540.362820 2078.063584 14.347469 5.703788 0.270497 4.375000" and the predicted output is a single-phase SS and BCC crystal structure (BCC_SS). Furthermore, this prediction is validated with the information obtained from the available dataset.

3.2. Prediction done by composition of alloy

In order to determine the phases and crystal structure of CCAs from the composition of the alloy, it is essential to have knowledge of the composition values for each element in the alloy from the beginning. In the case of the MoNbTiVZr system, there are 26 input parameters that represent the compositions of the elements, which are ("Al Co Cr Fe Ni Cu Mn Ti V Nb Mo Zr Hf Ta W C Mg Zn Si Re N Sc Li Sn Be Num_of_Elem"). These parameters are then utilized as input for the machine learning (ML) model, which interprets the values. Subsequently, the ML algorithm applies decision-making processes to generate the predicted output. In the specific case of the MoNbTiVZr system, the user input is "0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.232558 0.069767 0.232558 0.23558 0.23558 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 5.0" the predicted output indicates the presence of a single-phase SS and BCC crystal structure (BCC_SS) and this prediction has been further validated using the available dataset. The ML model considers the composition values of the alloy and utilizes them to make informed predictions regarding the phases and crystal structure.

3.3. Metrics used to evaluate the models

Important metrics that were used for the evaluation of the models are 1. Accuracy 2. Precision 3. Recall and F1 Score. Accuracy deals with the

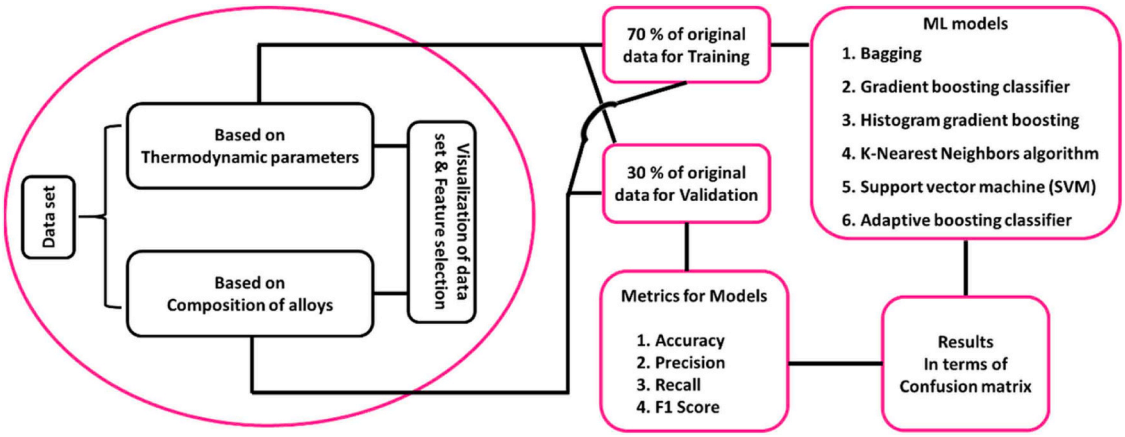

Fig. 11. The work flow diagram of present ML based modelling approach for phase predictions in CCAs.

overall correctness of the model's predictions and estimated as, Number of Correct Predictions / Total Number of Predictions. Similarly, precision focuses on the proportion of correctly predicted positive instances among all instances predicted as positive and estimated as, true positives / sum of true positives and false positives. Recall quantifies the correctly predicted positive instances out of all actual positive instances and estimated as True Positives / Sum of True Positives and False Negatives. Finally, F1 score relies on precision and recall and it is a harmonic mean of both. It provides a balanced measure of both metrics. It is calculated in way to ensure equal contribution from both metrics as, 2 * (Precision multiplied by Recall) / Sum of Precision and Recall.

4. Results

Fig. 12 shows the confusion matrix of our best classification model that is Histogram Gradient Boost Model for the prediction done using the thermodynamic properties of the alloy, while Fig. 13 shows the same for prediction done using the composition of an alloy. Confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data.

In Fig. 12 and Fig. 13, it is observed that confusion matrix of the best performing model helps us in summarizing the performance of the model on the classification problem set that we have trained. By looking at the higher number of true positive and true negative it can be concluded that the model that have been designed is performing well on the unseen data as well. Hence, we could state that this could be used for

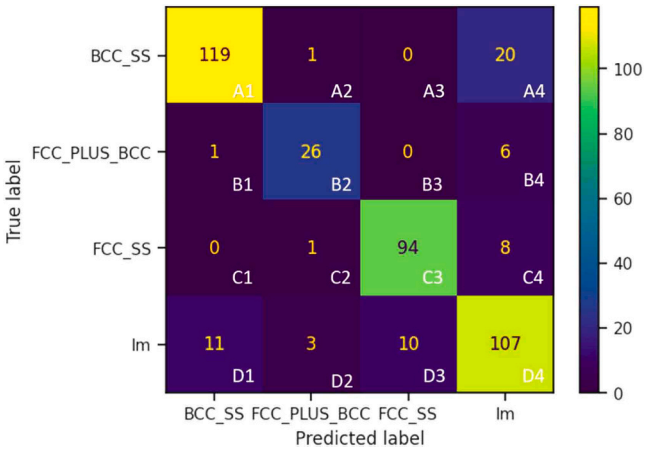

Fig. 12. Confusion matrix of Histogram Gradient Boosting Model which was proven to be the most optimal model for predictions using the thermodynamic parameters.

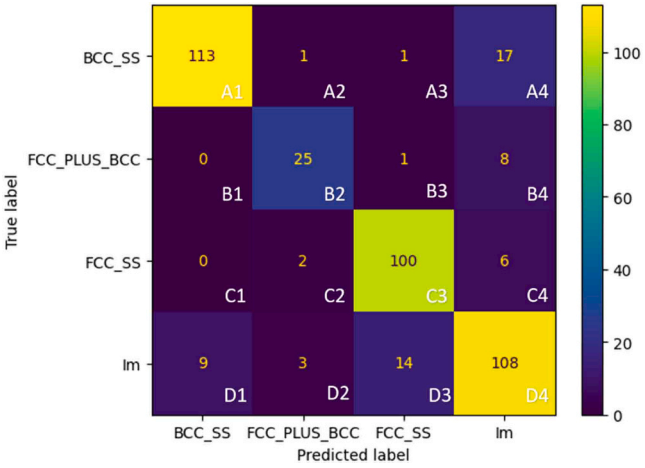

Fig. 13. Confusion matrix of Histogram Gradient Boosting Model which was proven to be the most optimal model for predictions using alloy composition.

predicting the phases which will further help in understanding their properties, behaviour and performance. From Fig. 12 it can be observed that summation of squares A1-A4 represents a total number BCC_SS phases available and it is 140. Further, the value of A1, i.e., 119 suggests that out of 140 BCC_SS phases present in our test data, our model has predicted 119 correctly as BCC_SS. However, by having a look at A2, A3 and A4 squares it shows that there are 21 falsely predicted instances, out of that 1 is misclassified as FCC_PLUS_BCC and 20 are IM. Similarly, summation of squares B1-B4 represents a total number of FCC_PLUS_BCC phases available and it is 33. The value of square B2, i.e., 26 represents that out of 33 FCC_PLUS_BCC phases present in our test data, our model has predicted 26 times correctly as FCC_PLUS_BCC. However, by having a look at B1, B3 and B4 squares it shows that there are 7 falsely predicted instances, out of that 1 is misclassified as BCC_SS and 6 are IM. Now it can be observed that summation of C1-C4 is 103 and that represents the total number of FCC_SS phases present in the dataset. The value of C3 represents a number 94, which means that out of 103 FCC_SS present in our test data, our model has predicted 94 correctly as FCC_SS. However, by having a look at C1, C2 and C4 squares it shows that there are 9 falsely predicted instances, out of that 1 is misclassified as FCC_PLUS_BCC and 8 are IM. Eventually, it can be observed that summation of D1-D4 represents a number 131, and that is the total number of FCC_SS phase present in the dataset. The value of square D4, i.e., 107 means that out of 131 FCC_SS phases present in our test data, our model has predicted 107 correctly as FCC_SS. However, by having a look at D1, D2 and D3 squares it shows that there are 24 falsely

predicted instances, out of that 11 is misclassified as BCC_SS, 3 are FCC_PLUS_BCC and 10 are FCC_SS.

From Fig. 13, it can be observed that it can be observed that summation of squares A1-A4, i.e., 140 represents a total number of instances when BCC_SS is present in the dataset. The value of square A1 represents a number 113, which means that out of 140 BCC_SS present in our test data, our model has predicted 113 correctly as BCC_SS. However, by having a look at A2, A3 and A4 squares, it can be realized that there are 19 falsely predicted instances, out of that 1 is misclassified as FCC_PLUS_BCC, 1 is misclassified as FCC_SS and 17 are IM. Similarly, the summation of squares B1-B4, i.e., 34 represents a total number of instances when FCC_PLUS_BCC is present in the dataset. The value of square B2 represents a number 25, which means that out of 34 FCC_PLUS_BCC present in our test data, our model has predicted 25 correctly as FCC_PLUS_BCC. However, by having a look at B1, B3 and B4 squares it shows that there are 9 falsely predicted instances, out of that 1 is misclassified as FCC_SS and 8 are IM. Further it can be observed that summation of squares C1-C4, i.e., 100 represents a total number of instances when FCC_SS is present in the dataset. The value of square C3 represents a number 94, which means that out of 108 FCC_SS present in our test data, our model has predicted 100 correctly as FCC_SS. However, by having a look at C1, C2 and C4 squares it shows that there are 8 falsely predicted instances, out of that 2 is misclassified as FCC_PLUS_BCC and 6 are IM. Eventually, the summation of squares D1-D4, i.e., 134 represents a total number of instances when IM is present in the dataset. The value of D4 represents a number 108, which means that out of 134 IM present in our test data, our model has predicted 107 correctly as IM. However, by having a look at D1, D2 and D3 squares it shows that there are 26 falsely predicted instances, out of that 9 is misclassified as BCC_SS, 3 are FCC_PLUS_BCC and 14 are FCC_SS.

Furthermore, to get insights, metrics discussed in Section 3.3 have been used for evaluation of ML models and need to be quantitatively discussed. For the predictions using thermodynamic and configurational parameters Bagging with KNN achieved 0.823, 0.822, 0.823 and 0.822 of accuracy, precision, recall and f1 score, respectively. Bagging with SVM achieved 0.717, 0.686, 0.717 and 0.694 of accuracy, precision, recall and f1 score, respectively. Gradient boost achieved 0.837, 0.838, 0.837 and 0.837 accuracy, precision, recall and f1 score, respectively. Adaboost achieved 0.668, 0.659, 0.668 and 0.659 accuracy, precision, recall, and f1 score, respectively. Histogram Gradient Boosting achieved 0.850, 0.853, 0.850 and 0.851 accuracy, precision, recall, and f1 score, respectively. From the above obtained scores, it can be concluded that Histogram Gradient Boosting is the most optimal model for predicting phases based on the thermodynamic and configurational parameters.

Similarly, for predicting the phases based on the composition of alloys the scores of each model are as below. Bagging with KNN obtained 0.791, 0.790, 0.791 and 0.786 accuracy, precision, recall, and f1 score, respectively. Bagging with SVM obtained 0.730, 0.678, 0.730 and 0.702 accuracy, precision, recall, and f1 score, respectively. Gradient boosting obtained 0.823, 0.824, 0.823 and 0.821 accuracy, precision, recall, and f1 score, respectively. Adaboost achieved 0.639, 0.636, 0.639 and 0.635 accuracy, precision, recall, and f1 score, respectively. Histogram Gradient Boosting obtained 0.848, 0.850, 0.848 and 0.848 accuracy, precision, recall, and f1 score, respectively. Here also based on the above obtained scores, it can be concluded that Histogram Gradient Boosting is the most optimal model for predicting phases based on the composition of alloys. Since Histogram Gradient boosting is a very powerful machine learning model it performs exceptionally well on both the datasets and gives promising results which can be used for the studies. We can also observe the same from the confusion matrix that is displayed above. Gradient boost model also performs fairly well for the problem statement but Histogram Gradient Boosting stands out. In terms of computational efficiency Histogram GB reduces memory consumption. When working with high-dimensional data, where there are many features, Histogram GB is quite useful and hence fits our dataset well. By generating histograms for each feature, it eliminates the computational load of analyzing each feature separately, allowing for effective feature selection and splitting decisions. The Fig. 14 shows how the best model (Histogram Gradient Boosting) has performed on unseen data points which are inputted manually. Here, the trained model has given ten inputs manually to the model. The model has predicted all ten new data points correctly into their classes based on the composition of metals and the number of elements present in the alloy. The "Predicted Phases" column is the predicted class. The "phases" shows the actual phase from experiments. Similarly, Fig. 15 shows how the best model has performed on unseen data points which are inputted manually. Here, the trained model has given ten inputs manually to the model. The model has predicted all ten new data points correctly into their classes based on the thermodynamic and configurational parameters of alloy. The "Predicted Phases" column is the predicted class. The "phases" shows the actual phase from the experiments.

As an outlook, it can be said that these obtained results can be more precise and useful if these two feature sets (thermodynamic parameters and the composition) can be combined together. The combination may lead to improve the richness of the input representation and possibly yield better classification accuracy and model performance. Use of SHAP values or feature importance analysis to explain the decision-making process of the models could be introduced, along with demonstrations of their applications in phase prediction of complex concentrated alloys [51]. Another important aspect to be explored is ablation studies, to evaluate the impact of various features or model components on the prediction outcomes. For example, by progressively removing certain features (such as atomic size differences, electronegativity, etc.), the changes in model performance can be observed to identify the key features that have the most significant impact on the prediction results. Furthermore, model's performance under different datasets or noise conditions shall be checked by introducing noisy data or using different datasets to test the stability of the model. As the present work explores the baseline ML models, depicting a pathway towards alloy development and exploration, future work should also be aimed towards introducing custom model, hybrid model or stacking or blending classifier that would significantly contribute towards enhancing the prediction capabilities and enabling better alloy design. Finally, hyperparameter tuning can be integrated to determine hyper parameters that would lead to more improved predictions [52]. Implementation of tree-based classifiers like Random Forest, CatBoost, and XGBoost may provide valuable insights.

5. Summary and conclusions

In this work, we conducted ML based phase prediction using a database comprising 1360 experimental alloy combinations. The database included thermodynamic and configurational parameters along with corresponding phase information. The phases were primarily categorized into two sections: Single Solid (SS) and Intermediate (IM) phases. Additionally, for enhanced microstructure evolution, the crystal structure classification of the SS phase was attempted, focusing on three possibilities: Body-Centered Cubic (BCC_SS), Face-Centered Cubic (FCC_SS), and a combination of BCC and FCC (FCC_PLUS_BCC). Five machine learning (ML) algorithms were implemented for phase and crystal structure prediction. We can draw following conclusions from this work,

- Among the explored models, the Histogram gradient Boosting model and Gradient Boosting model demonstrated the highest accuracies of 85.0 % and 83.7 %, respectively, for phase prediction with thermodynamic parameters.
- Moreover, for phase prediction considering the alloy composition, the Histogram gradient Boosting model achieved an accuracy of 84.8 %, while the Gradient Boosting model achieved an accuracy of 82.3 %.

	Al	Co	Cr	Fe	Ni	Cu	Mn	Ti	V	Nb	...	Si	Re	N	Sc	Li	Sn	Be	Num_of_Elem	Phases	Predicted Phases
51	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.375000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	4	BCC_SS	BCC_SS
1101	0.165289	0.165289	0.165289	0.165289	0.330579	0.00	0.000000	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	6	Im	Im
1313	0.100000	0.250000	0.080000	0.150000	0.360000	0.00	0.000000	0.060000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	6	Im	Im
514	0.000000	0.340000	0.200000	0.060000	0.340000	0.00	0.000000	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	5	FCC_SS	FCC_SS
1075	0.000000	0.200000	0.200000	0.200000	0.200000	0.00	0.200000	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	5	FCC_SS	FCC_SS
430	0.800000	0.000000	0.000000	0.000000	0.000000	0.05	0.000000	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.05	0.0	0	5	Im	Im
855	0.100000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.300000	0.04	0.2	...	0.0	0.0	0.0	0	0.00	0.0	0	6	Im	Im
764	0.000000	0.199601	0.199601	0.199601	0.199601	0.00	0.199601	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	6	FCC_SS	FCC_SS
985	0.000000	0.250000	0.250000	0.250000	0.250000	0.00	0.000000	0.000000	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	4	FCC_SS	FCC_SS
712	0.125000	0.000000	0.000000	0.346154	0.173077	0.00	0.317308	0.038462	0.00	0.0	...	0.0	0.0	0.0	0	0.00	0.0	0	5	Im	Im

10 rows × 28 columns

Fig. 14. Predictions based on alloy composition.

Num_of_Elem	Density_calc	dHmix	dSmix	dGmix	Tm	n.Para	Atom.Size.Diff	Elect.Diff	VEC	Phases	Predicted Phases
4	8.431909	2.360000	12.841516	557.102065	2141.385887	13.138264	4.915964	0.118595	4.300000	BCC_SS	BCC_SS
4	11.060000	3.500000	11.530000	697.736195	2668.500000	8.790000	3.950000	0.120000	4.500000	BCC_SS	BCC_SS
5	9.910916	2.720000	13.381611	592.502252	2282.194003	12.571551	4.994417	0.118254	4.400000	BCC_SS	BCC_SS
7	7.474871	-14.265300	13.468672	366.475597	1412.121864	1.591160	5.828183	0.126941	8.075000	Im	Im
6	7.215294	-8.178175	11.543359	355.574042	1360.004545	2.305171	6.453991	0.165556	7.346192	FCC_SS	FCC_SS
5	8.906166	-3.537415	12.569175	440.666003	1692.079042	6.982870	2.065570	0.148759	8.142857	FCC_SS	FCC_SS
6	7.432030	-14.636667	12.919259	378.881625	1456.840901	1.527001	5.631744	0.129117	8.091667	Im	Im
5	8.240000	-3.900000	12.200000	494.663284	1896.730000	5.930000	1.960000	0.110000	8.200000	FCC_SS	FCC_SS
5	9.538745	-4.477431	7.687645	491.251351	1851.613088	3.648169	3.699123	0.202645	7.723000	FCC_SS	FCC_SS
5	8.202821	-2.335000	10.591006	420.900126	1603.994033	8.514280	2.092874	0.099955	8.025000	FCC_SS	FCC_SS

Fig. 15. Predictions based on thermodynamic parameters.

- To validate the predicted results, a confusion matrix was utilized. The trained models successfully predicted the phases of CCAs when tested with experimentally acquired data. This validation confirmed the predictive capability of the models.
- Overall, this research highlights the significance of accurate phase prediction in alloy design and demonstrates the effectiveness of ML algorithms, specifically the Histogram gradient Boosting Classifier and Gradient Boosting models, in predicting phases and crystal structures in CCAs.
- In future work, the approach can be extended to larger datasets to further enhance the efficiency of phase prediction, facilitating the design process of desired CCAs. Furthermore, datasets related to thermodynamic parameters and the composition of the CCAs may be combined together to yield better results.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at doi:10.1016/j.nxmate.2025.101248.

References

[1] E.P. George, D. Raabe, R.O. Ritchie, High-entropy alloys, vol. 4, issue 8, Nat. Rev. Mater. (2019) 515–534.
[2] M.-H. Tsai, J.-W. Yeh, High-entropy alloys: a critical review, Mater. Res. Lett. 2 (3) (2014) 107–123.
[3] S. Wang, Atomic structure modeling of multi-principal-element alloys by the principle of maximum entropy, Entropy 15 (12) (2013) 5536–5548.
[4] Y. Deng, C.C. Tasan, K.G. Pradeep, H. Springer, A. Kostka, D. Raabe, Design of a twinning-induced plasticity high entropy alloy, Acta Mater. 94 (2015) 124–133.
[5] B. Schuh, F. Mendez-Martin, B. Völker, E. George, H. Clemens, R. Pippan, 404, A. Hohenwarter, Mechanical properties, microstructure and thermal stability 405 of a nanocrystalline CoCrFeMnNi high-entropy alloy after severe plastic defor-406 mation, Acta Mater. 96 (2015) 258–268.
[6] W. Huang, P. Martin, H.L. Zhuang, Machine-learning phase prediction of high-entropy alloys, Acta Mater. 169 (2019) 225–236.
[7] C.-y Hsu, J.-W. Yeh, S.-K. Chen, T.-T. Shun, Wear resistance and high-temperature compression strength of fcc CuCoNiCrAl 0.5 fe alloy with boron addition, Metall. Mater. Trans. A 35 (2004) 1465–1469.
[8] Alireza Valizadeh, Ryoji sahara & maaouia souissi alloys innovation through machine learning: a statistical literature review, Sci. Technol. Adv. Mater. Methods 4 (1) (2024) 2326305.
[9] R. Machaka, Machine learning-based prediction of phases in high-entropy alloys, Comput. Mater. Sci. 188 (2021) 110244.
[10] S. Guo, Phase selection rules for cast high entropy alloys: an overview, Mater. Sci. Technol. 31 (10) (2015) 1223–1230.
[11] Y.-c Liu, S.-y Yen, S.-h Chu, S.-k Lin, M.-H. Tsai, Mechanical and thermodynamic data-driven design of Al-Co-Cr-Fe-Ni multi-principal element alloys, Mater. Today Commun. 26 (2021) 102096.
[12] V.K. Soni, S. Sanyal, K.R. Rao, S.K. SinhaA review on phase prediction in high entropy alloys, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 235(22) (2021) 6268-6286..
[13] P. Singh, A.V. Smirnov, A. Alam, D.D. Johnson, First-principles prediction of incipient order in arbitrary high-entropy alloys: exemplified in Ti0. 25CrFeNiAlx, Acta Mater. 189 (2020) 248–254.
[14] S. Hernandez, Development of methods for reducing the cost of density functional theory and time-dependent density functional theory, UCLA, 2015.

[15] S.Y. Lee, S. Byeon, H.S. Kim, H. Jin, S. Lee, Deep learning-based phase prediction of high-entropy alloys: optimization, generation, and explanation, Mater. Des. 197 (2021) 109260.
[16] Z. Pei, J. Yin, J.A. Hawk, D.E. Alman, M.C. Gao, Machine-learning informed prediction of high-entropy solid solution formation: beyond the Hume-Rothery rules, npj Comput. Mater. 6 (1) (2020) 50.
[17] L. Zhang, H. Chen, X. Tao, H. Cai, J. Liu, Y. Ouyang, Q. Peng, Y. Du, Machine learning reveals the importance of the formation enthalpy and atom-size difference in forming phases of high entropy alloys, Mater. Des. 193 (2020) 108835.
[18] B. Chanda, P.P. Jana, J. Das, A tool to predict the evolution of phase and Young's modulus in high entropy alloys using artificial neural network, Comput. Mater. Sci. 197 (2021) 110619.
[19] Y. Zhang, C. Wen, C. Wang, S. Antonov, D. Xue, Y. Bai, Y. Su, Phase prediction in high entropy alloys with a rational selection of materials descriptors and machine learning models, Acta Mater. 185 (2020) 528–539.
[20] A. Choudhury, T. Konnur, P. Chattopadhyay, S. Pal, Structure prediction of multi-principal element alloys using ensemble learning, Eng. Comput. 37 (3) (2020) 1003–1022.
[21] Y. Zhang, S. Yang, J. Evans, Revisiting Hume-Rothery's rules with artificial neural networks, Acta Mater. 56 (5) (2008) 1094–1105.
[22] Z.-S. Nong, J.-C. Zhu, Y. Cao, X.-W. Yang, Z.-H. Lai, Y. Liu, Stability and structure prediction of cubic phase in as cast high entropy alloys, Mater. Sci. Technol. 30 (3) (2014) 363–369.
[23] L.A. Dominguez, R. Goodall, I. Todd, Prediction and validation of quaternary high entropy alloys using statistical approaches, Mater. Sci. Technol. 31 (10) (2015) 1201–1206.
[24] F. Tancret, I. Toda-Caraballo, E. Menou, P.E.J.R. Díaz-Del, Designing high entropy alloys employing thermodynamics and Gaussian process statistical analysis, Mater. Des. 115 (2017) 486–497.
[25] N. Islam, W. Huang, H.L. Zhuang, Machine learning for phase selection in multi-principal element alloys, Comput. Mater. Sci. 150 (2018) 230–235.
[26] T. Kostiuchenko, F. Körmann, J. Neugebauer, A. Shapeev, Impact of lattice relaxations on phase transitions in a high-entropy alloy studied by machine-learning potentials, npj Comput. Mater. 5 (1) (2019) 55.
[27] W. Huang, P. Martin, H.L. Zhuang, Machine-learning phase prediction of high-entropy alloys, Acta Mater. 169 (2019) 225–236.
[28] X. Jiang, R. Zhang, C. Zhang, H. Yin, X. Qu, Fast prediction of the quasi phase equilibrium in phase field model for multicomponent alloys based on machine learning method, Calphad 66 (2019) 101644.
[29] Y. Li, W. Guo, Machine-learning model for predicting phase formations of high-entropy alloys, Phys. Rev. Mater. 3 (9) (2019) 095005.
[30] J. Qi, A.M. Cheung, S.J. Poon, High entropy alloys mined from binary phase diagrams, Sci. Rep. 9 (1) (2019) 15501.
[31] X.-G. Li, C. Chen, H. Zheng, Y. Zuo, S.P. Ong, Complex strengthening mechanisms in the NbMoTaW multi-principal element alloy, npj Comput. Mater. 6 (1) (2020) 70.
[32] E. Meshkov, I. Novoselov, A. Shapeev, A. Yanilkin, Sublattice formation in CoCrFeNi high-entropy alloy, Intermetallics 112 (2019) 106542.
[33] G. Kim, H. Diao, C. Lee, A. Samaei, T. Phan, M. de Jong, K. An, D. Ma, P.K. Liaw, W. Chen, First-principles and machine learning predictions of elasticity in severely lattice-distorted high-entropy alloys with experimental validation, Acta Mater. 181 (2019) 124–138.
[34] Y.-J. Chang, C.-Y. Jui, W.-J. Lee, A.-C. Yeh, Prediction of the composition and hardness of high-entropy alloys by machine learning, Jom 71 (2019) 3433–3442.
[35] Q. Wu, Z. Wang, X. Hu, T. Zheng, Z. Yang, F. He, J. Li, J. Wang, Uncovering the eutectics design by machine learning in the Al–Co–Cr–Fe–Ni high entropy system, Acta Mater. 182 (2020) 278–286.
[36] D. Dai, T. Xu, X. Wei, G. Ding, Y. Xu, J. Zhang, H. Zhang, Using machine learning and feature engineering to characterize limited material datasets of high-entropy alloys, Comput. Mater. Sci. 175 (2020) 109618.
[37] Reliance Jain, Sandeep Jain, Sheetal Kumar Dewangan, Lokesh Kumar Boriwal, Sumanta Samal, Machine learning-driven insights into phase prediction for high entropy alloys, J. Alloy. Metall. Syst. 8 (2024) 100110.
[38] Ronald Machaka, Glenda T. Motsi, Lerato M. Raganya, Precious M. Radingoana, Silethelwe Chikosha, Machine learning-based prediction of phases in high-entropy alloys: a data article, Data Brief. 38 (2021) 107346.
[39] R. Machaka, G.T. Motsi, L.M. Raganya, P.M. Radingoana, S. Chikosha, Machine learning-based prediction of phases in high-entropy alloys: a data article, Data Brief. 38 (2021).
[40] R. Machaka, Machine learning based prediction of phases in high-entropy alloys, SSRN Electron. J. (2020).
[41] S. Gorsse, M. Nguyen, O.N. Senkov, D.B. Miracle, Database on the mechanical properties of high entropy alloys and complex concentrated alloys, Data Brief. 21 (2018) 2664–2678.
[42] J.-P. Couzinié, O. Senkov, D. Miracle, G. Dirras, Comprehensive data compilation on the mechanical properties of refractory high-entropy alloys, Data Brief. 21 (2018) 1622–1641.
[43] F. YeY, Y. LiuCT andYang, High-entropy alloy: challenges and prospects, Mater. Today 19 (6) (2016) 349–362.
[44] Sijia Liu, Chao Yang, Machine learning design for High-Entropy alloys: models and algorithms, Metals 14 (2024) 235.
[45] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.
[46] A. Natekin, A. Knoll, Gradient boosting machines, a tutorial 7 (December) (2013).
[47] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[48] G. Guo, H. Wang, D. Bell, Y. Bi, K. Greer, KNN model-based approach in classification, on the move to meaningful Internet systems 2003: CoopIS, DOA, and ODBASE: OTM confederated international conferences. CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings, Springer, 2003, pp. 986–996.
[49] D.M.C. Nieto, E.A.P. Quiroz, M.A.C. Lengua, A systematic literature review on support vector machines applied to regression, in: IEEE Sciences and Humanities International Research Conference (SHIRCON), 2021, IEEE, 2021, pp. 1–4.
[50] T. Alaa, Adab. Classif. Overv. (2018).
[51] S. Kumar, V. Jindal, Integrating machine learning and DFT for hardness prediction in high-entropy alloys, MRS Commun. (2025), https://doi.org/10.1557/s43579-025-00736-7.
[52] Ujjawal Kumar Jaiswal, Yegi Vamsi Krishna, M.R. Rahul, Gandham Phanikumar, Machine learning-enabled identification of new medium to high entropy alloys with solid solution phases, Comput. Mater. Sci. 197 (2021) 110623.