A Project Report

on

Microstructure Analysis Using Machine Learning

Submitted by

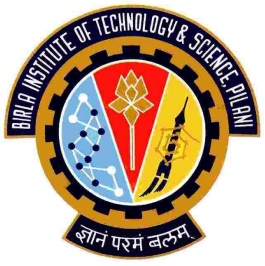
Tejash Singh - 2019ABPS0775P

Abhishek Malav - 2019ABPS0916P

Submitted to

Radha Raman Mishra

Study Oriented Project (MF F266)
First Semester 2021 - 2022



Birla Institute of Technology and Science, Pilani
Pilani-333031 (Rajasthan), India

December 2021

Index

Contents Page No.

List of figures

List of tables

1. Introduction

Traditional engineering alloys are made up of a single major element (for example in steels Fe percentage is 98-99% and Ni concentration in superalloys is near to 50% by weight) plus one or more solute elements present in much lower amounts. Multi-principal element alloys (MPEAs), also known as complex concentrated alloys (CCAs), on the other hand, are a type of alloy when no single factor dominates the mix and three or more major elements play a significant role. MPEAs with 5 or more primary components are commonly referred to as high entropy alloys (HEAs).Machine learning is used to predict the output based on data available so we can predict the better alloy properties based on the available data.

1.1 Background

Machine learning is increasingly being utilized in a variety of fields to forecast various data sets. We looked for previous work regarding the same domain that predicted the properties of high entropy alloys. We also discovered some study papers on the subject, and got some data sets to start our work with.

1.2 Literature review

We researched on various research papers, firstly to finalize the topic to work upon, then to look for one which has an adequate amount of dataset, finally settled at "Microstructure Analysis using Machine Learning" and we found 2 research papers, which are basically the data set of a large number of High Entropy Alloys (HEA) and how that data was collected here is the reference to those papers.

TOPIC	DETAILS
Expanded dataset of mechanical properties and observed phases of multi-principal element alloys	ChristopherK.H.Borg1, Carolina Frey , Jasper Moh, Tresa M. Pollock , StéphaneGorsse, DanielB. Miracle, Oleg N.Senkov, Bryce Meredig & James E.Saal
Database on the mechanical properties of high entropy alloys and complex concentrated alloys	S. Gorsse , M.H. Nguyen, O.N. Senkov, D.B. Miracle

1.3 Research Gap

The dataset present in the paper have been reported from the year 2004 to the year 2016, and as machine learning is a developing domain so such a vast range in data as well as model prediction has given a great hold and ambient ample amount of data and some codes to start with.

1.4 Plan for the present work

We have worked upon the dataset previously mentioned and cleaned data as it contained a lot of NULL points and had a lot of outliers which we had removed, so we came up with a decent accuracy of 87.2% which is a decent accuracy in machine learning, so for the present week we had increased our accuracy by nearly 1%.

2. Methodology

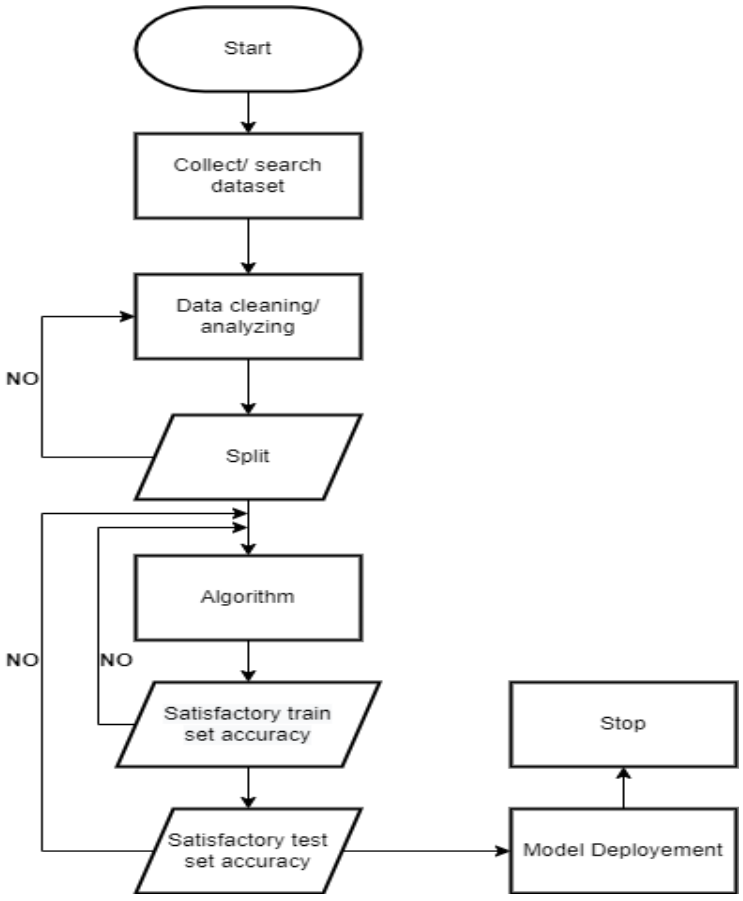
2.1 Materials/ Process/Model

When using a machine learning method, we must first understand the type of data set, after which we may apply the algorithm model. Then we cleaned the data set as it has too many empty spaces, so we tried filling it with the average value for the missing column, but then by filling up the largest value of the column increased the accuracy, so we stuck with that. We tried using many algorithms, but then had the highest accuracy using gradient boosting algorithm and random forest regressor algorithm which gave a decent amount of accuracy.

2.2 Tools/equipment/Software package

For this work we did all of my coding on Jupyter and used the scikit learn library for various algorithms. We have also used numpy, pandas, seasborn, matPlotLib, etc for manipulating and visualizing the data. Jupyter is an open source web-application.

2.3 A brief plan of the work



1) Random forest: Random forest is just a bag of n Decision Trees, each with its own set of hyper parameters and trained on various samples of data. Assume my Random forest contains 100 decision trees. Because each of these decision trees has a unique collection of hyper- 7 parameters and a unique group of training data, the choice or prediction made by these trees might vary greatly. After training each of these 100 trees with its own batch of data.

Now I'll look at the outcome predicted by all of the trees. We simply need to make one decision on one example or one test data now, and we do that with a simple vote. For that case, we go with what the majority of the trees anticipated.

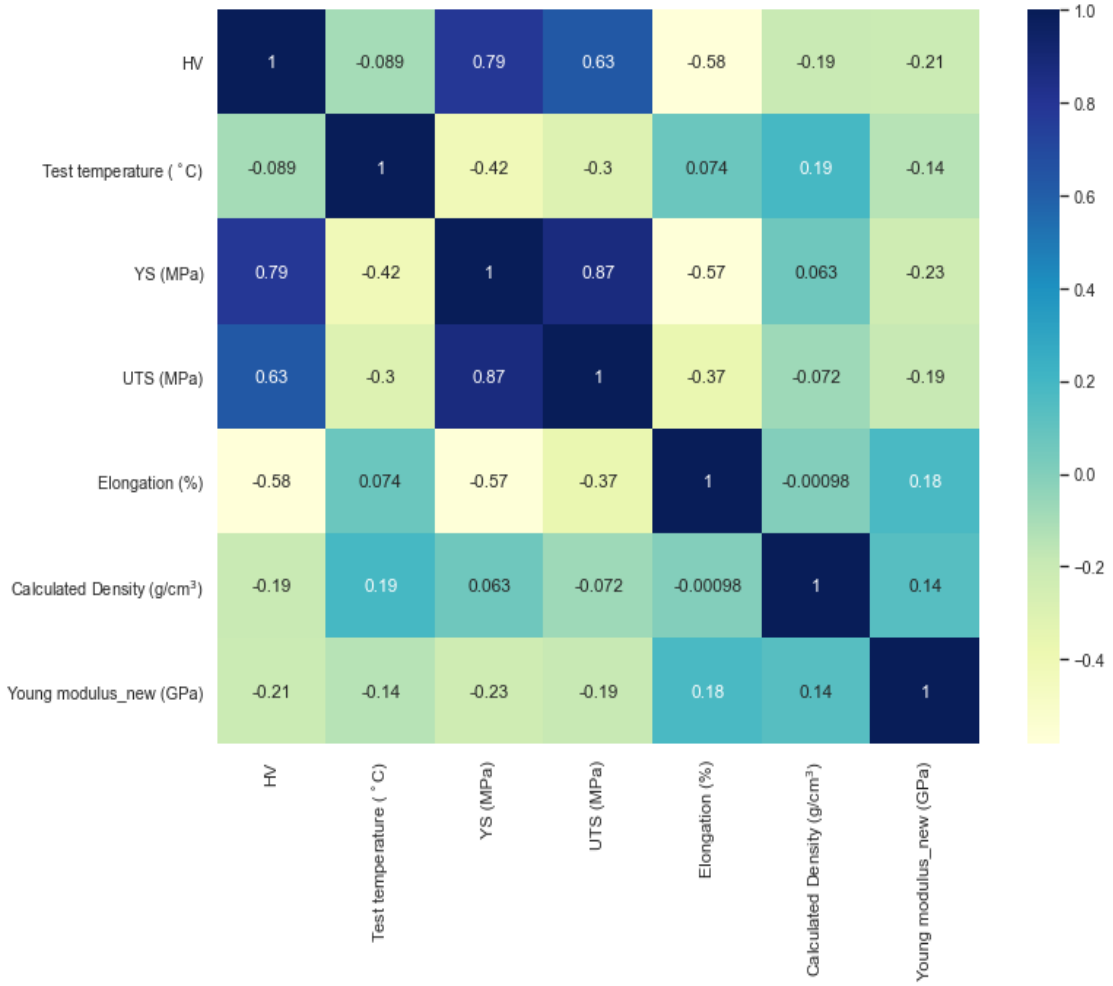
2) Gradient boosting: To create the final predictions, a Gradient Boosting Machine integrates the predictions from numerous decision trees. All of the weak learners in a gradient boosting machine are decision trees in this case. However, if we utilize more than one decision tree to forecast the outcome, it becomes more accurate. will be superior. In GBM, each decision tree's nodes use a distinct collection of characteristics. deciding on the optimum split It signifies that the individual trees are not all the same, and hence they are distinct. capable of extracting various signals from data Furthermore, each new tree takes into account the flaws or blunders committed by earlier trees As a result, each consecutive decision tree is created on the mistake

3) KNN: A k-nearest-neighbor algorithm is a data categorization technique that looks at the data points surrounding it to decide which group a data point belongs to. When an algorithm examines one point on a grid to determine whether it belongs to group A or B, it examines the states of the points nearby. The range is arbitrary, but the goal is to obtain a sample of the data. If the bulk of the points are in group A, the data point in question is likely to be in group A rather than B, and vice versa. Because it does not create a model of the data set beforehand, the k-nearest-neighbor technique is an example of a "lazy learner." It only performs calculations when prompted to poll the data point's neighbors. As a result, k-nn is fairly simple to implement for data mining.

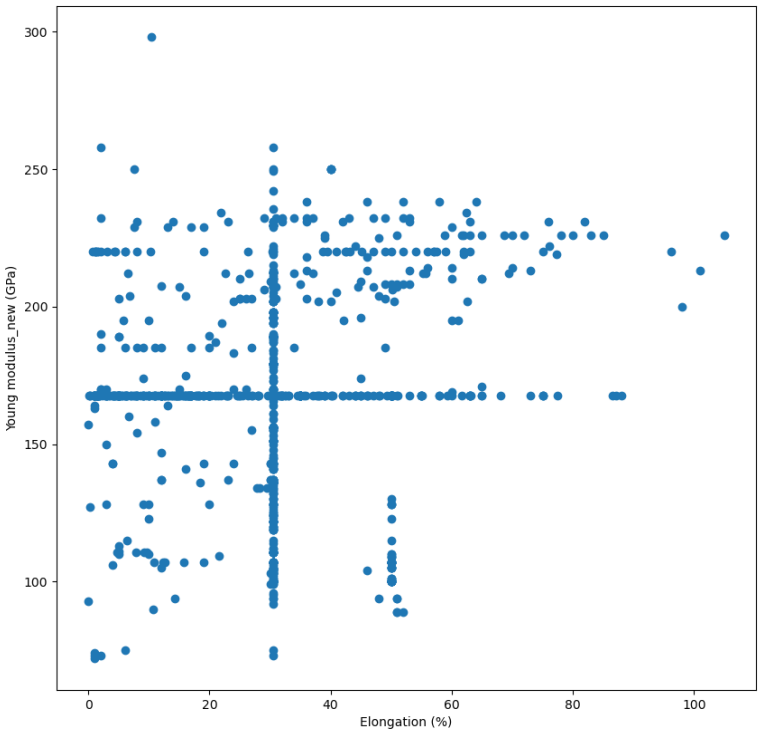
3. Results and Discussion

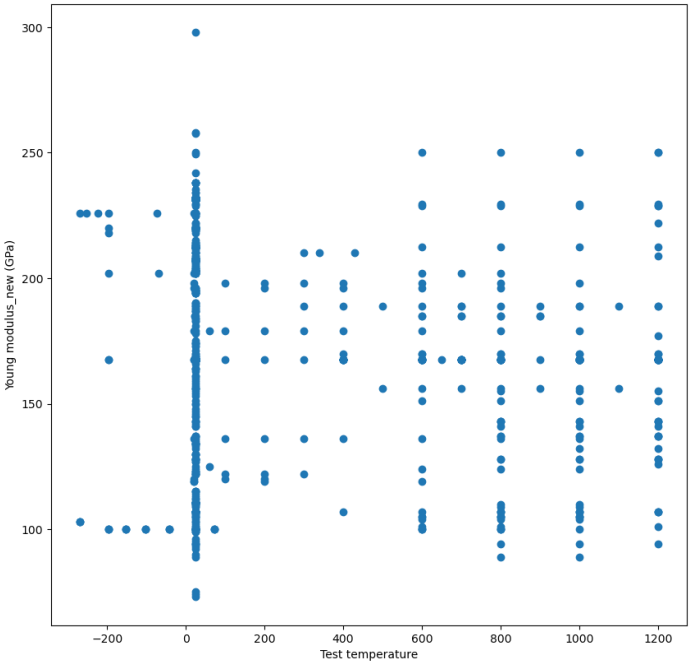
I made a new column in the data named young modulus_new. Data in that column includes that of three columns, namely ROM Young modulus, Exp. Young modulus and calculated young's modulus. I took the maximum value of the three and assigned it to the newly made column.

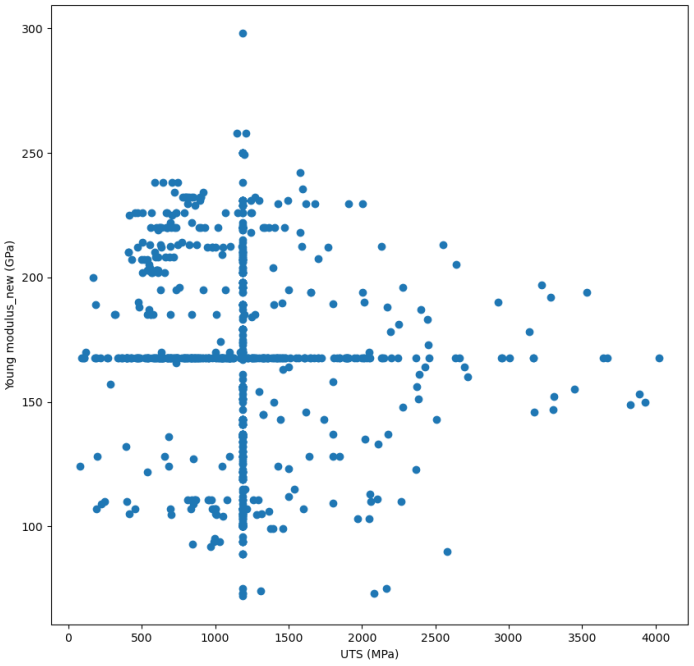
I then plotted a heatmap of the new column 'young modulus_new' in order to understand the interaction between this and the other columns.

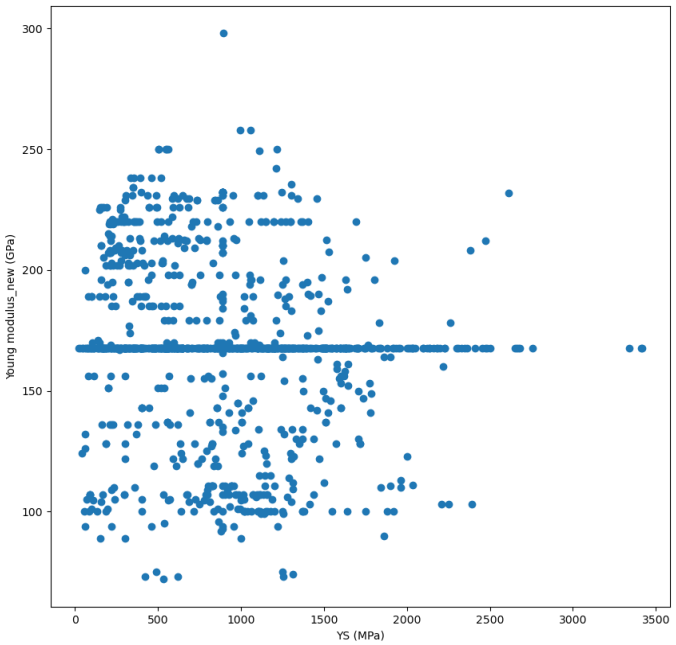


We will plot a few more graphs of this column versus a few other columns. Here are a few of them.









1) Random Forest Regressor

We have used this Regression model to predict the young modulus of these alloys. We have 558 data points. we will divide them in a certain ratio in order to obtain desired accuracy.

case a) when we have replaced the NAN value with the mean of Column.

Train size : test size	Accuracy on test (in %)	RMSE
95 : 5	88.3%	15.4
90:10	79%	20.35
80:20	82.7%	19.6

case b) when we have replaced the NAN with Max of the columns.

Train size : test size	Accuracy on test (in %)	RMSE
95:5	88.2%	15.4
90:10	79%	20.35
80:20	82.6%	19.6

After performing gridSearchCV, we had our parameters for random forest. the following were the results:

```
GridSearchCV(cv=10, estimator=RandomForestRegressor(n_jobs=-1),
             param_grid=[{'bootstrap': [True, False],
              'max_depth': [10, 50, None], 'max_features': [5, 20],
              'n_estimators': [10, 100]}],
             scoring='neg_mean_squared_error')
```

Here we can see that the max accuracy we have achieved is 88.3% which is corresponding to 95:5 ratio of train : test dataset points.

2) gradient boosting regressor

case a) when we have replaced the NAN value with the mean of Column

Train size : test size	Accuracy on test (in %)	RMSE
95:5	85%	17.45
90:10	72.8%	23.18
80:20	83.1%	19.35

case b) when we have replaced the NAN with Max of the columns.

Train size : test size	Accuracy on test (in %)	RMSE
95:5	84.94%	17.46
90:10	72.8%	23.16
80:20	80%	21.07

For gradient boosting regressor, after hyper-parameter tuning we got the following results.

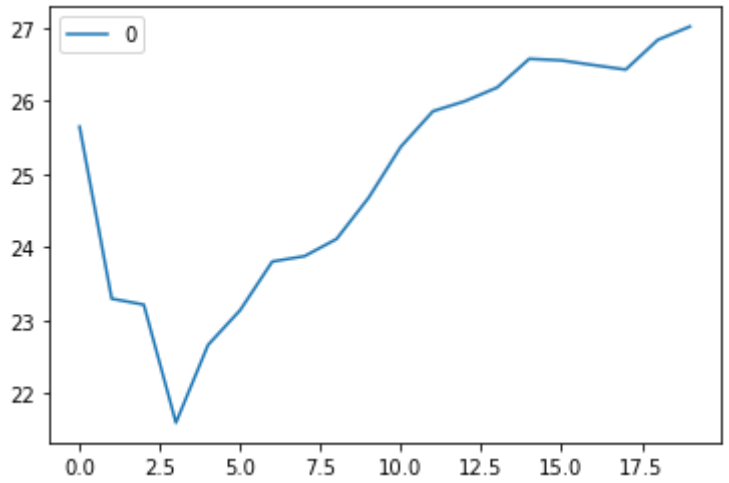
{'learning_rate': 0.01, 'max_depth': 4, 'min_samples_leaf': 3}

3) KNN.

case a) when we have replaced the NAN value with the mean of Column.

Train size : test size	Accuracy on test (in %)	RMSE
95:5	65%	30.5
90:10	76.5%	22.94
80:20	73.6%	22.65

For KNN, the number of neighbors used was 5. We arrived at this number using the elbow method and using optimum RMSE values. Below is the image of the elbow graph. These graphs are for 90:10 ratio of train size : test size.



RMSE value for each neighbor is also given.

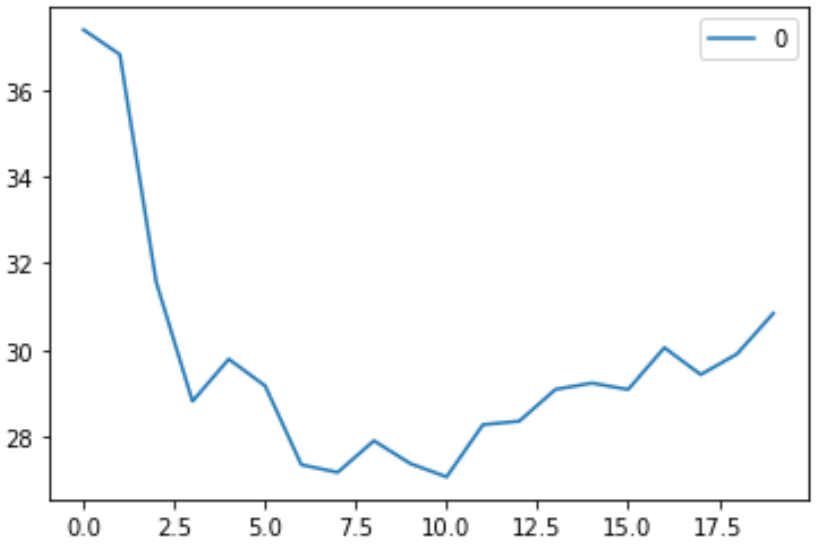
```
RMSE value for k=  1 is: 25.648513602156363
RMSE value for k=  2 is: 23.292875213555877
RMSE value for k=  3 is: 23.21145935175329
RMSE value for k=  4 is: 21.59484065366408
RMSE value for k=  5 is: 22.659944757844656
RMSE value for k=  6 is: 23.1358199697915
RMSE value for k=  7 is: 23.801749742746363
RMSE value for k=  8 is: 23.875074297519024
RMSE value for k=  9 is: 24.1106853429601
RMSE value for k=  10 is: 24.67668146318938
RMSE value for k=  11 is: 25.37412734554266
RMSE value for k=  12 is: 25.860248423723746
RMSE value for k=  13 is: 25.99862730258592
RMSE value for k=  14 is: 26.185635670885254
RMSE value for k=  15 is: 26.5783794906538
RMSE value for k=  16 is: 26.556640740325467
RMSE value for k=  17 is: 26.492164310120156
RMSE value for k=  18 is: 26.43047351600009
RMSE value for k=  19 is: 26.837696458103206
RMSE value for k=  20 is: 27.020029756380993
```

case b) when we have replaced the NAN with Max of the columns.

Train size : test size	Accuracy on test (in %)	RMSE
95:5	63%	31.5
90:10	77.8%	21.92
80:20	68.7%	26.23

For KNN, the number of neighbors used was 3. We arrived at this number using the elbow method and using optimum RMSE values. Below is the image of the elbow graph. These values are for 95:0.05 ratio of train size : test size

```
RMSE value for k=  1 is: 37.39881587124689
RMSE value for k=  2 is: 36.82659534901373
RMSE value for k=  3 is: 31.569542157964406
RMSE value for k=  4 is: 28.808530095268658
RMSE value for k=  5 is: 29.784105540659482
RMSE value for k=  6 is: 29.168450285599544
RMSE value for k=  7 is: 27.34108883135353
RMSE value for k=  8 is: 27.165943527203584
RMSE value for k=  9 is: 27.89657651901899
RMSE value for k=  10 is: 27.369447316827056
RMSE value for k=  11 is: 27.061086228003365
RMSE value for k=  12 is: 28.268136341861368
RMSE value for k=  13 is: 28.34847340920302
RMSE value for k=  14 is: 29.083212931464598
RMSE value for k=  15 is: 29.23063116685472
RMSE value for k=  16 is: 29.080352931460506
RMSE value for k=  17 is: 30.04875649557402
RMSE value for k=  18 is: 29.42620513626146
RMSE value for k=  19 is: 29.90274237893187
RMSE value for k=  20 is: 30.844517905479226
```



4. Conclusions

We have made a table of these models with different data for Young Modulus, one taking mean of NULL values, next taking max of NULL values and calculating accuracy with those sizes. From the tables we can say that, For Mean value gradient boosting classifier gave us an accuracy of 85% with train size of 95% and by using max value gave accuracy of 84.94%. For Mean value Random Forest regressor gave us an accuracy of 88.3% with train size of 95% and by using max value gave accuracy of 88.2% For Mean value KNN gave us an accuracy of 65% with train size of 95%. So the best accuracy was achieved using a Random Forest regressor when the train size was 95%, the accuracy was 88.35%.

5. Scope for future work

We increase accuracy of the model if we use advanced statistics for replacing the null values in the dataset. Furthermore, using computational skills we can find that optimum point between our dataset and outliers where we can still retain most of our dataset but still get a decent amount of accuracy.

References

1) Borg, C. & Saal, J. Expanded dataset of mechanical properties and observed phases of multi-principal element alloys. https://github. com/CitrineInformatics/MPEA_dataset (2020).
2) Research paper provided by Prof. Radha Raman Mishra:- *Expanded dataset of mechanical properties and observed phases of multi-principal element alloys*

Appendix

Working to get the code on github.