Developing Materials Informatics Systems for Alloy Design

A report submitted

In partial fulfillment of the requirements

For the degree of

Bachelor of Technology

By

1. Mansi Srivastava **(R11321XXXX)**

2.Bhavy Mutreja **(R11321XXXX)**

Department of Mechanical Engineering

University of Petroleum and Energy Studies

Dehradun, India-248001

MAY 2021

Department of Mechanical Engineering

University of Petroleum and Energy Studies

Certificate

It is certified that the work contained in the project titled **"DEVELOPING MATERIAL INFORMATICS SYSTEM FOR ALLOY DESIGN"** by following students has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree.

Student Name	***Roll Number***	***Signature***
1.MANSI SRIVASTAVA	***R11321XXXX***	
2.BAHVY MUTREJA	***R11321XXXX***	

Signature

Mr. Dishant Beniwal

Department of Mechanical Engineering,

School of Engineering, U.P.E.S.

Dehradun, Uttarakhand

India-248001

Signature

Dr. Ajay Srivastava

Department of Mechanical Engineering,

School of Engineering, U.P.E.S.

Dehradun, Uttarakhand

India- 248001

Abstract

To be able to leverage technology to improve everyday life the ability to design new material that can undergo shape lifting. 3d printing materials and arbitrary shapes that can change the shape upon stimuli such as electrical current , or photons, lights sources etc. So we can ask the question what drives the need for accelerated material

Discovery and development

Big data and data science impact the idea of accelerating the material discovery and development.

Well it turn out that in various areas of consumer products, of national lead international lead, mobility, security, energy, aging infrastructure , health care , its communication etc. there's always a demand to push the capabilities of these devices, these products.

To be able to improve their performance to make use of latest advances, new advances in material capabilities to do just that. And it typically takes a lot longer than we'd like from the time where we discover a material ,in a laboratory to the time where it's converted into product takes a lot longer , typically 20 to 25 years .we' like to reduce that down to a five to 10 year .

Presently the state was to emerge inquiry to understand this with further depth, there about advances of the past few decades that give us great promises for realizing this imagination. If we go back 50 to 60 years ago quantum mechanics was developed which is essentially and enabling technology for controlling material at atomic scales. With engineering mainly involved in use of much larger scales types of modeling, capabilities and approaches such as finite element methods. But today there 'a great overlap today the engineering and design committees also have at their disposal these tools of quantum.in addition to design theories design methodologies, enhanced by computation and big data.

Acknowledgements

We take this opportunity to express my sincere thanks to my supervisor **Mr. Dishant Beniwal** for their guidance and support. I would also like to express my gratitude towards them for showing confidence in me. It was a privilege to have a great experience working under him in a cordial environment.

We are very much thankful to the University of Petroleum and Energy Studies, for providing me the opportunity of pursuing B.Tech. Mechanical in a peaceful environment with ample resources.

We are thankful to **Mrs. Jhalak Beniwal** for providing basic knowledge of machine learning.

In the end, I would like to acknowledge my parents, family members. Without their support, this work would not have been possible.

Name of students

Mansi Srivastava(5000XXXX)

Bhavy Mutreja(5000XXXX)

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

1.1 Basic Introduction

Data-pushed strategies are essential in materials technological know-how. Applying supervised gadget learning to a fabric's database can expect the properties of unknown materials from the compositions, systems, and processes without synthesizing the fabric. Alternatively, its miles essential to recognize which compositions, structures, and techniques in large part have an effect on properties. Subsequently, one of the predominant troubles in substances science is to extract family members inside the databases. This could be tackled via making use of correlation evaluation, comparing the contributions of a prediction version, and appearing a characteristic choice technique, and applying information-driven techniques to improve the understanding of materials. a few databases include over masses of heaps of statistics. For that reason, many relations were elucidated via information-pushed strategies implemented to substances simulation databases. Then again, the results considerably trade when specializing in experimental databases. In comparison to simulations, experiments in substances technological know-how require a big amount of time and money, and it is difficult to generate a massive amount of experimental statistics. Moreover, current public databases comprise little experimental facts if the targeted property is confined. In lots of cases, it's miles hard to extract members of the family between properties, compositions, systems, and methods from materials databases which might be based totally on experimental effects. Alloy improvement can be increased, if such members of the family are clarified the usage of pure information-pushed techniques. The usage of statistics-pushed strategies it's used to layout an experimental plan to optimize the properties at the same time as promoting the expertise of target materials. We need to speak about the riding forces for coming across new substances and how we need to boost up the fee of their development in incorporation to products.

To summarize modeling have exploded over closing 15 to 20 years , and it leads us to the idea that we can lessen the time for discovery of fabric to deployment of merchandise by means of taking blessings of existing facts**.**

1.2 Motivation

looking forward closer to tomorrow the science fiction of the day past can end up the fact of tomorrow. If we examine the twentieth century most of the advances that you have witnessed is material enabled. simply taking place this simples listing air tour ,water deliver and distribution , computing , net , telephones highways space journey , nuclear technology , renewable energies simply arose in the final century a lot of which might be enabled in part or completely through material trends. These advancements took ages to take place, more than one failed tries and yet we stand at the verge of challenges, these demanding situations range from fitness care, better drugs to capacity to nuclear fusions which could be

a superb step forward. The capability to adopt clinical discovery under consideration in a distributive and collaborative environment can lead us to our favored consequences slightly beforehand. These are 21st century that we will look ahead to imagine and adaptive surroundings which are situational aware and Adaptive so that were can accommodate our wishes on the time. therefore to do so all wonderful advances of century that we are able to exploit , that we will pursue are taken into consideration, in which these advances come from engineered answers ,in which computing and materials will play key roles enabled by using the explosion of facts and data sciences..

1.3 Objective

To develop an informatics device to predict the critical temperature of superconductor. Information-driven strategies are vital in materials technological know-how. Making use of supervised device studying to a fabric's database can is expecting the residences of unknown substances from the compositions, structures, and procedures without synthesizing the material. On the other hand, it is essential to know which compositions, structures, and procedures in large part affect residences. Consequently, one of the principal troubles in substances science is to extract relations within the databases. This may be tackled by way of making use of correlation evaluation, evaluating the contributions of a prediction version, and acting a feature choice method, and applying data-pushed strategies to enhance the understanding of substances. Some databases comprise over loads of lots of information. For this reason, many relations were elucidated thru facts-pushed techniques carried out to substances simulation databases. However, the effects significantly change while focusing on experimental databases. As compared to simulations, experiments in materials technological know-how require a substantial quantity of time and money, and it is tough to generate a big amount of experimental statistics. Moreover, current public databases incorporate little experimental statistics if the focused property is confined. In lots of cases, it's far hard to extract members of the family among homes, compositions, systems, and approaches from materials databases which are based on experimental consequences.

2.LITERATURE SURVEY

Author	Data analysis Method	Informatics system Purpose
Kam Hamidieh.	• The Multiple Regression Model. • The XG-Boost Model.	A Data-Driven Statistical Model for Predicting the Critical Temperature of a Superconductor.
Shaobo Li 1, Yabo Dan 1,*, Xiang Li 1, Tiantian Hu 2, Rongzhi Dong 1, Zhuo Cao 1 and Jianjun Hu.	• A hybrid neural network (HNN) that combines a convolutional neural network (CNN) and long short-term memory neural network (LSTM).	Critical Temperature Prediction of Superconductors Based on Atomic Vectors and Deep Learning.
J.M. Rickman1,2, H.M. Chan2, M.P. Harmer2, J.A. Smeltzer2, C.J. Marvel2, A. Roy3 & G. Balasubramanian3.	• A multiple regression analysis and its generalization. • A canonical-correlation analysis (CCA).	The screening of multi-principal elements and high-entropy alloys.
Zhong-Li Liu,1,2,a) Peng Kang,2 Yu Zhu,3 Lei Liu,3 and Hong Guo.	• Multi-algorithm cross-validation. • Multi-step learning approach.	Layered high-TC of superconductors

Table 1: Literature review [1, 2, 3, 4]

2.1 Research Gap

The family members among the mechanical properties, warmth remedy, and compositions of factors in aluminum alloys are extracted with the aid of a substances informatics method. Model is first educated by using an organized database to be expecting the properties of materials. The dependence of the anticipated houses on explanatory variables, that is, the kind of warmness treatment and detail composition. From the dependencies, a component is to obtained, the preferred residences is investigated. Extracted relations which might be hard to locate through easy correlation analysis. Also used to design an experimental plan to optimize the materials residences while selling the understanding of target substances. It is important to recognize which compositions, systems, and processes largely affect homes. But, for the reason that dependences of the mechanical properties on procedures and element compositions are outstanding, extracting the members of the family is vital inside the development of latest alloys. to date, this mission has trusted enjoy and emotions of professional researchers, but alloy development ought to be increased, if such family members are clarified the usage of pure records-driven strategies. The usage of records-driven techniques, herein we propose a new approach to extract members of the family from alloy experimental databases with small statistics sizes.

- First, we train a machine studying prediction version for the mechanical residences whilst procedure parameters and compositions of elements are inputted. Here, the prediction model with the best prediction performance is selected from a few supervised systems getting to know fashions.
- 2nd, the distributions of the process parameters and detail compositions primarily based at the prediction version are investigated to acquire the preferred mechanical homes. Our method employs generator of latest records via the prediction model.
- 0.33, we extract factors for the preferred mechanical properties, together with the affect degree at a look, to sell the information of alloys.

3. THEORY

3.1 Developing materials informatics systems for alloy design

3.1.1 Informatics system: Predicting the critical Temperature of Superconductors

- Informatics machine: Informatics is the take a look at of the structure, conduct, and interactions of natural and engineered computational structures. Knowledge informational phenomena - together with computation, cognition, and verbal exchange - enable technological advances.
- It targets to expand and follow firm theoretical and mathematical foundations for the features which are not unusual to all computational systems.
- Informatics has many aspects, and encompasses a number of current academic disciplines - artificial Intelligence, Cognitive science and pc technological know-how.
- thus Informatics presents a hyperlink among disciplines with their personal methodologies and perspectives, bringing together a commonplace scientific paradigm, common engineering techniques and a pervasive stimulus from technological improvement and sensible utility.
- In our informatics system it's far customized to predicting the important temperature of superconductors.

on this take a look at, we take a completely records-driven technique to create a statistical version that predicts.

3.1.2 What is Superconductivity?

Superconductivity is a phenomenon whereby a charge moves through a material without resistance. In general words Superconductivity, is a complete disappearance of electrical resistance in various solids when they are cooled below a characteristic temperature. This temperature, called the transition temperature, varies for different materials but generally is below 20 K (−253 °C).

3.1.3 What is Critical Temp?

The essential temperature for superconductors is the temperature at which the electrical resistivity of a metallic drops to 0. The transition is so surprising and complete that it seems to be transitions to a distinct section of depend; this superconducting segment is described with the aid of the BCS theory.

3.1.3.1 How can we predict critical temp of superconductor?

The traditional way to are expecting the crucial temp is to use machine learning algorithms in it .there are various ways of making database and apply by way of the use of distinct ML strategies.

But we will Use CNN (Convolutional Neural Networks) to perceive the critical temp with the aid of the usage of minimum mistakes. CNN is a superb technique because it involves thousands of variables in a unmarried algorithm and provide the approximate precise value.

3.2 Regression modeling

Regression modeling is a shape of predictive modeling strategies which investigates the relationship among dependent variables (goal) and unbiased variables (parameters). This approach is used for forecasting, time series modeling and finding the causal effect relationship between the variables.

There is various types of regression techniques to be had to make predictions, usually those are driven by way of 3 metric -

1. Range of independent variables.
2. Shape of the regression line.
3. Type of based variables.

3.2.1 Linear regression:

One of the most broadly used modeling techniques . in this techniques the structured variable is continuous ,independent can be non-stop or discrete while the character of regression line is directly line.

Linear Regression establishes a dating among established variable (Y) and one or more independent variables (X) the use of a high-quality in shape instantly line (also known as regression line).

It is represented by an equation Y=a+b*X + e, where a is intercept, b is slope of the line and e is an error term. This equation can be used to predict the value of the target variable based on a given predictor variable(s).

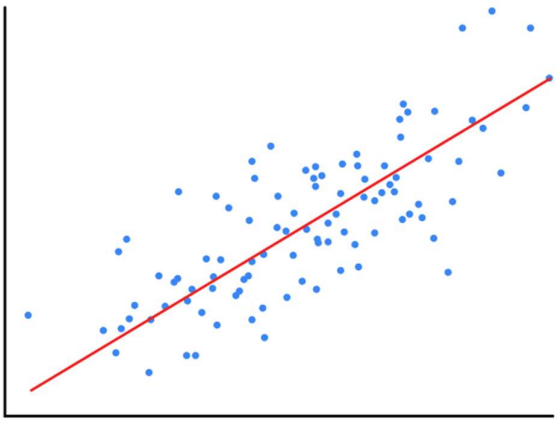


Fig1: Graph of linear Regression [1]

3.2.2 Polynomial Regression:

A regression equation is a polynomial regression equation if the power of independent variable is more than 1. The equation below represents a polynomial equation:

y=a+b*x^2

In this regression technique, the best fit line is not a straight line. It is rather a curve that fits into the data points. Higher polynomials can end up producing weird results on extrapolation.

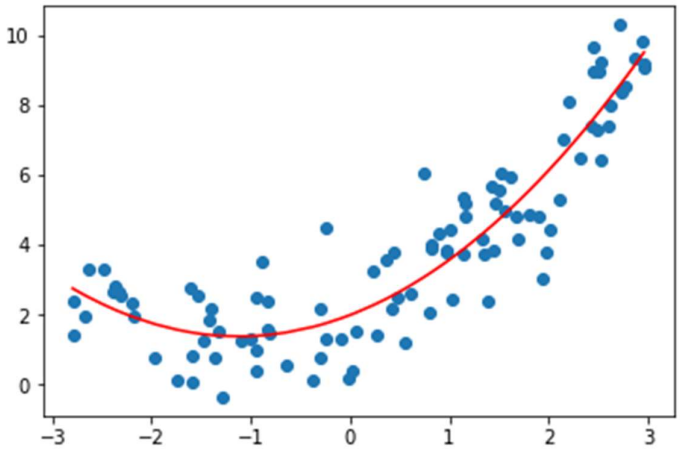


Fig 2: Graph of Polynomial Regression [2]

3.2.3 Decision tree:

Decision trees (DTs) are a non-parametric supervised gaining knowledge of method used for category and regression. The aim is to create a version that predicts the cost of a target variable via gaining knowledge of easy selection rules inferred from the records features. A tree may be visible as a piecewise consistent approximation. Calls for little records instruction. Different strategies often require statistics normalisation, dummy variables want to be created and blank values to be eliminated. Notice however that this module does no longer help missing values. The cost of the use of the tree (i.e., predicting information) is logarithmic inside the wide variety of statistics.

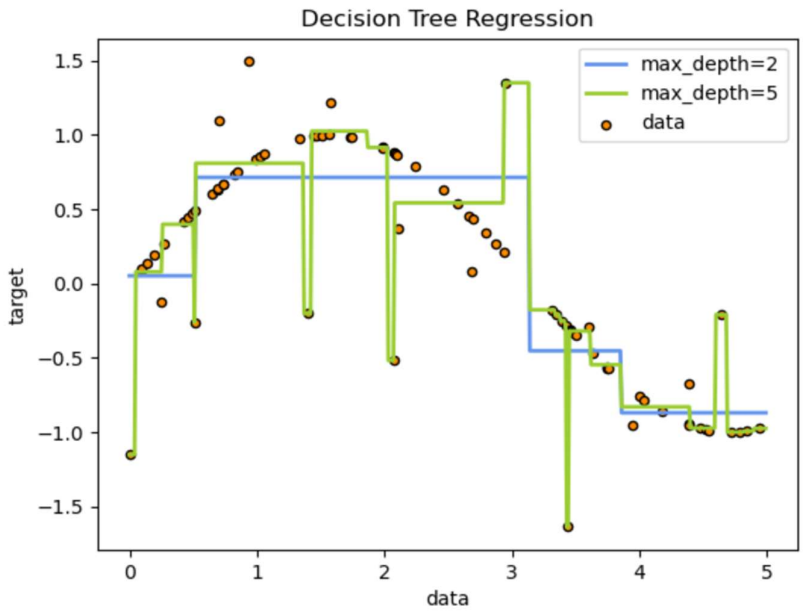


Fig 3: Graph of Decision tree Regression [3]

3.2.4 Neural Network

Neural Networks are a category of models inside the fashionable device getting to know literature. Neural networks are a selected set of algorithms that have revolutionized gadget learning. They're stimulated via organic neural networks and the cutting-edge so-known as deep neural networks have confirmed to work pretty well. Neural Networks are themselves standard function approximations, which is why they may be applied to nearly any system studying trouble about studying a complex mapping from the input to the output area. To simplify a neural community is a chain of algorithms that endeavours to apprehend underlying relationships in a set of data via a method that mimics the way the human brain operates.

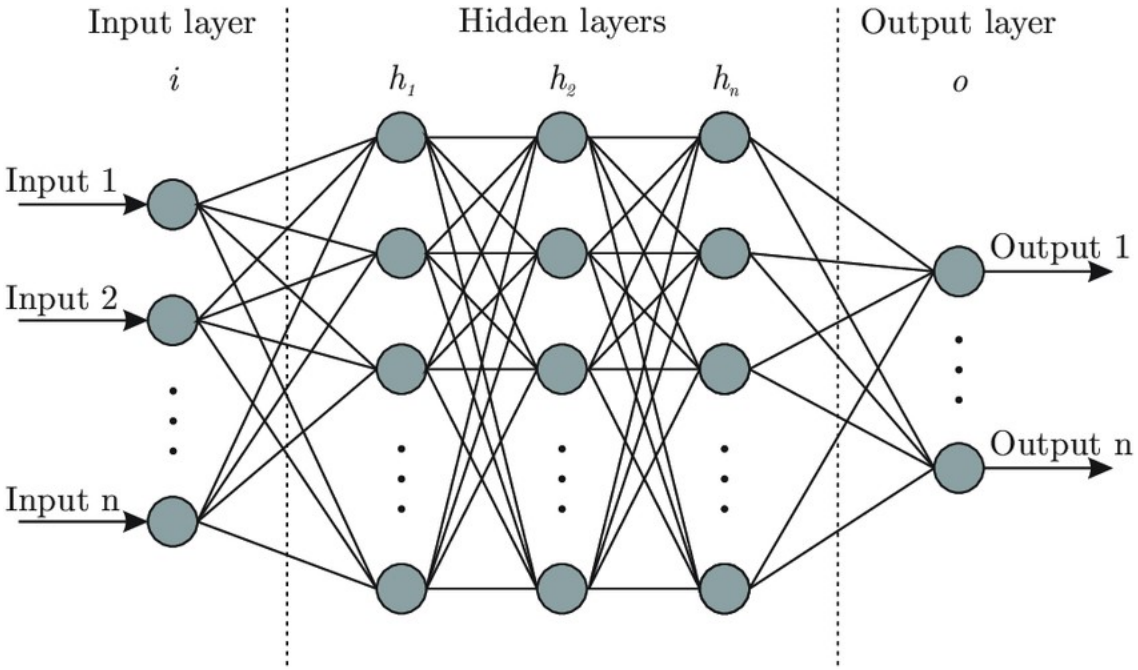


Fig 4: Working of Neural Networks [4]

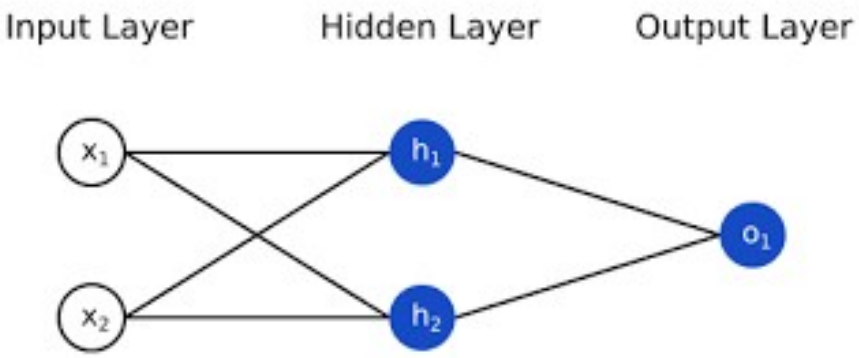


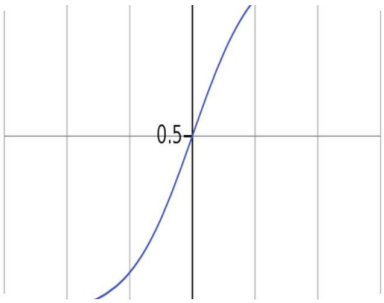
Fig 5: Layered working of Neural Networks [4]

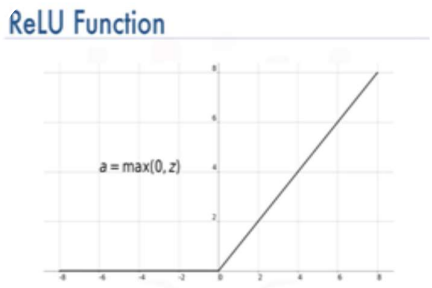
3.2.3.1 Functions

3.2.3.1.1 Sigmoid Function: A Sigmoid function is a mathematical function having a characteristic S-shaped curve or sigmoid curve .The formula for this function is written below

$$S(x) = \frac{1}{1 + e^{-x}}$$

3.2.3.1.2 RELU Function: The Rectified Linear Function or ReLU for short is a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero. It has become the default activation for many types of neural networks because a model that uses it is easier to train and often achieves better performance.





6(A)Sigmoid Curve

6(B) ReLU Curve

Fig 6: Functions

3.3 Random Forest : A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.
Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model.

4. METHODLOGY

Methodology to be followed to achieve the defined objectives is given below

- **Literature Analysis:** After studying the research papers we will be able to create an extensive alloy database from existing literature and can modify it in our project by analyzing them.
- **Database Preparation:** The database is the most essential part of any informatics system.
- **Data Analysis:** Now we will analyze the database and extract the information required in our project and this data can be implemented in Python.
- **Python Programming:** Now the main task is to learn the Python programming from the data we have extracted from papers and implement machine learning algorithms for structure and property prediction.
- **Validation of Model:** The final task is to validate our model and compare it with previous experiments and results.

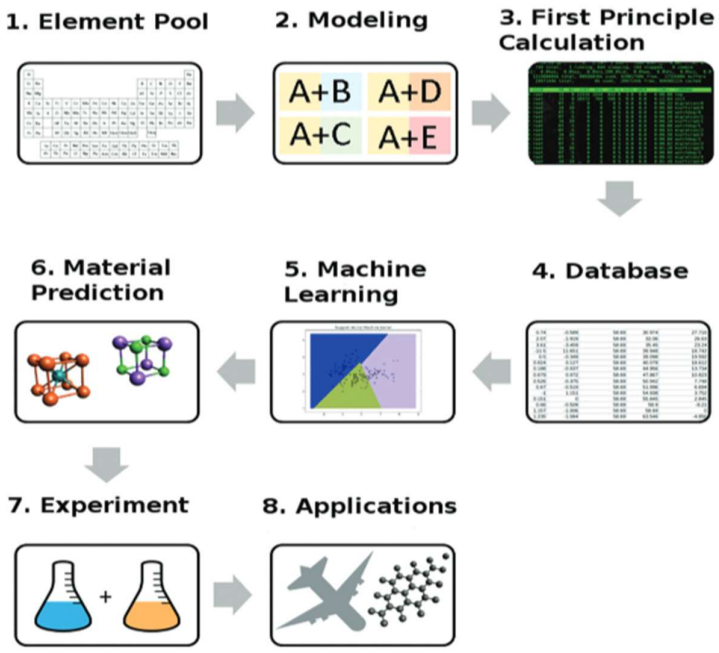


Figure 7: Methodology of project

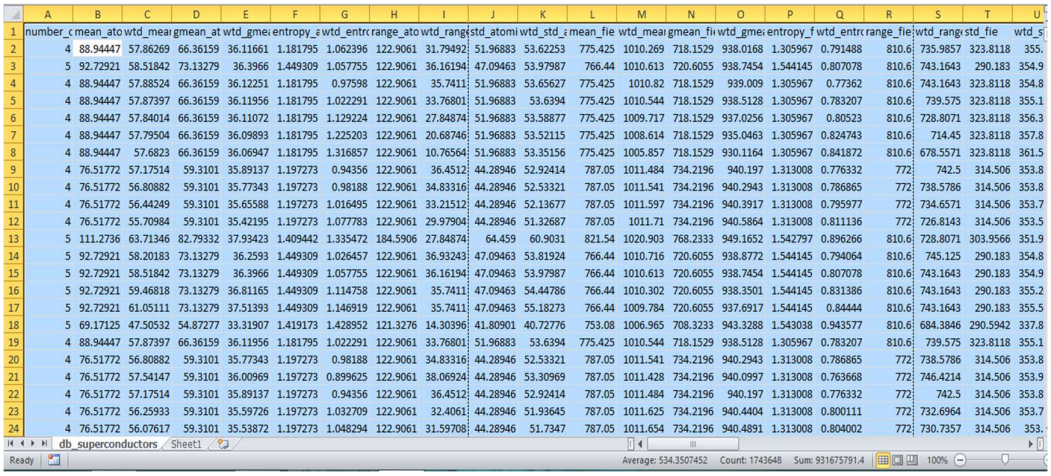
5. SIMULATION RESULTS

5.1 Database Creation:

Materials informatics is strongly depending on huge collections of records known as big statistics. However, database management together with amassing and organizing statistics may be pretty complicated. A center part of substances informatics is the improvement of the database. Right here it is accomplished in two components

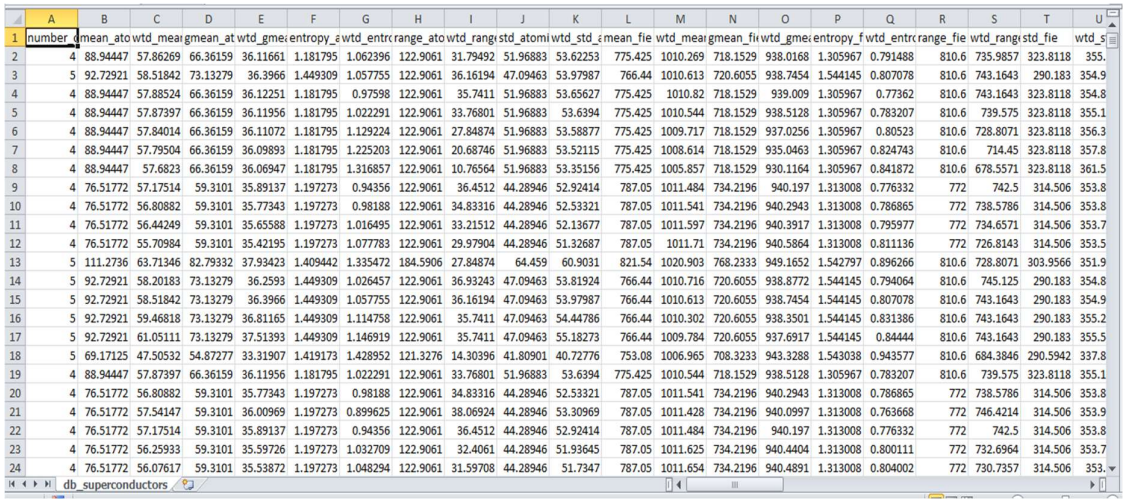
1. There are many databases being advanced which vary in accessibility, topics, and depth. The superconductor records comes from the Superconducting cloth Database maintained by using Japan's National Institute for materials Science (NIMS) at http: //supercon.nims.cross.jp/index_en.html. Database is supported by using the NIMS, a public institution primarily based in Japan. The database contains a large list of superconductors, their critical temperatures, and the supply references frequently from magazine articles.

2. element information preparation The detail information are obtained with the aid of the use of the Element Data function from Mathematical .the primary ionization energy data came from http://www.ptable.com/ and is merged with the Mathematical facts.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	number_c	mean_ato	wtd_mea	gmean_at	wtd_gme	entropy_a	wtd_entr	range_ato	wtd_rang	std_atomi	wtd_std_a	mean_fie	wtd_mea	gmean_fi	wtd_gme	entropy_f	wtd_entr	range_fie	wtd_rang	std_fie	wtd_s
2	4	88.94447	57.86269	66.36159	36.11661	1.181795	1.062396	122.9061	31.79492	51.96883	53.62253	775.425	1010.269	718.1529	938.0168	1.305967	0.791488	810.6	735.9857	323.8118	355.
3	5	92.72921	58.51842	73.13279	36.3966	1.449309	1.057755	122.9061	36.16194	47.09463	53.97987	766.44	1010.613	720.6055	938.7454	1.544145	0.807078	810.6	743.1643	290.183	354.9
4	4	88.94447	57.88524	66.36159	36.12251	1.181795	0.97598	122.9061	35.7411	51.96883	53.65627	775.425	1010.82	718.1529	939.009	1.305967	0.77362	810.6	743.1643	323.8118	354.8
5	4	88.94447	57.87397	66.36159	36.11956	1.181795	1.022291	122.9061	33.76801	51.96883	53.6394	775.425	1010.544	718.1529	938.5128	1.305967	0.783207	810.6	739.575	323.8118	355.1
6	4	88.94447	57.84014	66.36159	36.11072	1.181795	1.129224	122.9061	27.84874	51.96883	53.58877	775.425	1009.717	718.1529	937.0256	1.305967	0.80523	810.6	728.8071	323.8118	356.3
7	4	88.94447	57.79504	66.36159	36.09893	1.181795	1.225203	122.9061	20.68746	51.96883	53.52115	775.425	1008.614	718.1529	935.0463	1.305967	0.824743	810.6	714.45	323.8118	357.8
8	4	88.94447	57.6823	66.36159	36.06947	1.181795	1.316857	122.9061	10.76564	51.96883	53.35156	775.425	1005.857	718.1529	930.1164	1.305967	0.841872	810.6	678.5571	323.8118	361.5
9	4	76.51772	57.17514	59.3101	35.89137	1.197273	0.94356	122.9061	36.4512	44.28946	52.92414	787.05	1011.484	734.2196	940.197	1.313008	0.776332	772	742.5	314.506	353.8
10	4	76.51772	56.80882	59.3101	35.77343	1.197273	0.98188	122.9061	34.83316	44.28946	52.53321	787.05	1011.541	734.2196	940.2943	1.313008	0.786865	772	738.5786	314.506	353.8
11	4	76.51772	56.44249	59.3101	35.65588	1.197273	1.016495	122.9061	33.21512	44.28946	52.13677	787.05	1011.597	734.2196	940.3917	1.313008	0.795977	772	734.6571	314.506	353.7
12	4	76.51772	55.70984	59.3101	35.42195	1.197273	1.077783	122.9061	29.97904	44.28946	51.32687	787.05	1011.71	734.2196	940.5864	1.313008	0.811136	772	726.8143	314.506	353.5
13	5	111.2736	63.71346	82.79332	37.93423	1.409442	1.335472	184.5906	27.84874	64.459	60.9031	821.54	1020.903	768.2333	949.1652	1.542797	0.896266	810.6	728.8071	303.9566	351.9
14	5	92.72921	58.20183	73.13279	36.2593	1.449309	1.026457	122.9061	36.93243	47.09463	53.81924	766.44	1010.716	720.6055	938.8772	1.544145	0.794064	810.6	745.125	290.183	354.8
15	5	92.72921	58.51842	73.13279	36.3966	1.449309	1.057755	122.9061	36.16194	47.09463	53.97987	766.44	1010.613	720.6055	938.7454	1.544145	0.807078	810.6	743.1643	290.183	354.9
16	5	92.72921	59.46818	73.13279	36.81165	1.449309	1.114758	122.9061	35.7411	47.09463	54.44786	766.44	1010.302	720.6055	938.3501	1.544145	0.831386	810.6	743.1643	290.183	355.2
17	5	92.72921	61.05111	73.13279	37.51393	1.449309	1.146919	122.9061	35.7411	47.09463	55.18273	766.44	1009.784	720.6055	937.6917	1.544145	0.84444	810.6	743.1643	290.183	355.5
18	5	69.17125	47.50532	54.87277	33.31907	1.419173	1.428952	121.3276	14.30396	41.80901	40.72776	753.08	1006.965	708.3233	943.3288	1.543038	0.943577	810.6	684.3846	290.5942	337.8
19	4	88.94447	57.87397	66.36159	36.11956	1.181795	1.022291	122.9061	33.76801	51.96883	53.6394	775.425	1010.544	718.1529	938.5128	1.305967	0.783207	810.6	739.575	323.8118	355.1
20	4	76.51772	56.80882	59.3101	35.77343	1.197273	0.98188	122.9061	34.83316	44.28946	52.53321	787.05	1011.541	734.2196	940.2943	1.313008	0.786865	772	738.5786	314.506	353.8
21	4	76.51772	57.54147	59.3101	36.00969	1.197273	0.899625	122.9061	38.06924	44.28946	53.30969	787.05	1011.428	734.2196	940.0997	1.313008	0.763668	772	746.4214	314.506	353.9
22	4	76.51772	57.17514	59.3101	35.89137	1.197273	0.94356	122.9061	36.4512	44.28946	52.92414	787.05	1011.484	734.2196	940.197	1.313008	0.776332	772	742.5	314.506	353.8
23	4	76.51772	56.25933	59.3101	35.59726	1.197273	1.032709	122.9061	32.4061	44.28946	51.93645	787.05	1011.625	734.2196	940.4404	1.313008	0.800111	772	732.6964	314.506	353.7
24	4	76.51772	56.07617	59.3101	35.53872	1.197273	1.048294	122.9061	31.59708	44.28946	51.7347	787.05	1011.654	734.2196	940.4891	1.313008	0.804002	772	730.7357	314.506	353.

Fig8: Database with count and sum



number_	mean_ato	wtd_mea	gmean_at	wtd_gmea	entropy_a	wtd_entro	range_ato	wtd_range	std_atomi	wtd_std_a	mean_fie	wtd_mean	gmean_fie	wtd_gmea	entropy_f	wtd_entro	range_fie	wtd_range	std_fie	wtd_s
4	88.94447	57.86269	66.36159	36.11661	1.181795	1.062396	122.9061	31.79492	51.96883	53.62253	775.425	1010.269	718.1529	938.0168	1.305967	0.791488	810.6	735.9857	323.8118	355.
5	92.72921	58.51842	73.13279	36.3966	1.449309	1.057755	122.9061	36.16194	47.09463	53.97987	766.44	1010.613	720.6055	938.7454	1.544145	0.807078	810.6	743.1643	290.183	354.9
4	88.94447	57.88524	66.36159	36.12251	1.181795	0.97598	122.9061	35.7411	51.96883	53.65627	775.425	1010.82	718.1529	939.009	1.305967	0.77362	810.6	743.1643	323.8118	354.8
4	88.94447	57.87397	66.36159	36.11956	1.181795	1.022291	122.9061	33.76801	51.96883	53.6394	775.425	1010.544	718.1529	938.5128	1.305967	0.783207	810.6	739.575	323.8118	355.1
4	88.94447	57.84014	66.36159	36.11072	1.181795	1.129224	122.9061	27.84874	51.96883	53.58877	775.425	1009.717	718.1529	937.0256	1.305967	0.80523	810.6	728.8071	323.8118	356.3
4	88.94447	57.79504	66.36159	36.09893	1.181795	1.225203	122.9061	20.68746	51.96883	53.52115	775.425	1008.614	718.1529	935.0463	1.305967	0.824743	810.6	714.45	323.8118	357.8
4	88.94447	57.6823	66.36159	36.06947	1.181795	1.316857	122.9061	10.76564	51.96883	53.35156	775.425	1005.857	718.1529	930.1164	1.305967	0.841872	810.6	678.5571	323.8118	361.5
4	76.51772	57.17514	59.3101	35.89137	1.197273	0.94356	122.9061	36.4512	44.28946	52.92414	787.05	1011.484	734.2196	940.197	1.313008	0.776332	772	742.5	314.506	353.8
4	76.51772	56.80882	59.3101	35.77343	1.197273	0.98188	122.9061	34.83316	44.28946	52.53321	787.05	1011.541	734.2196	940.2943	1.313008	0.786865	772	738.5786	314.506	353.8
4	76.51772	56.44249	59.3101	35.65588	1.197273	1.016495	122.9061	33.21512	44.28946	52.13677	787.05	1011.597	734.2196	940.3917	1.313008	0.795977	772	734.6571	314.506	353.7
4	76.51772	55.70984	59.3101	35.42195	1.197273	1.077783	122.9061	29.97904	44.28946	51.32687	787.05	1011.71	734.2196	940.5864	1.313008	0.811136	772	726.8143	314.506	353.5
5	111.2736	63.71346	82.79332	37.93423	1.409442	1.335472	184.5906	27.84874	64.459	60.9031	821.54	1020.903	768.2333	949.1652	1.542797	0.896266	810.6	728.8071	303.9566	351.9
5	92.72921	58.20183	73.13279	36.2593	1.449309	1.026457	122.9061	36.93243	47.09463	53.81924	766.44	1010.716	720.6055	938.8772	1.544145	0.794064	810.6	745.125	290.183	354.8
5	92.72921	58.51842	73.13279	36.3966	1.449309	1.057755	122.9061	36.16194	47.09463	53.97987	766.44	1010.613	720.6055	938.7454	1.544145	0.807078	810.6	743.1643	290.183	354.9
5	92.72921	59.46818	73.13279	36.81165	1.449309	1.114758	122.9061	35.7411	47.09463	54.44786	766.44	1010.302	720.6055	938.3501	1.544145	0.831386	810.6	743.1643	290.183	355.2
5	92.72921	61.05111	73.13279	37.51393	1.449309	1.146919	122.9061	35.7411	47.09463	55.18273	766.44	1009.784	720.6055	937.6917	1.544145	0.84444	810.6	743.1643	290.183	355.5
5	69.17125	47.50532	54.87277	33.31907	1.419173	1.428952	121.3276	14.30396	41.80901	40.72776	753.08	1006.965	708.3233	943.3288	1.543038	0.943577	810.6	684.3846	290.5942	337.8
4	88.94447	57.87397	66.36159	36.11956	1.181795	1.022291	122.9061	33.76801	51.96883	53.6394	775.425	1010.544	718.1529	938.5128	1.305967	0.783207	810.6	739.575	323.8118	355.1
4	76.51772	56.80882	59.3101	35.77343	1.197273	0.98188	122.9061	34.83316	44.28946	52.53321	787.05	1011.541	734.2196	940.2943	1.313008	0.786865	772	738.5786	314.506	353.8
4	76.51772	57.54147	59.3101	36.00969	1.197273	0.899625	122.9061	38.06924	44.28946	53.30969	787.05	1011.428	734.2196	940.0997	1.313008	0.763668	772	746.4214	314.506	353.9
4	76.51772	57.17514	59.3101	35.89137	1.197273	0.94356	122.9061	36.4512	44.28946	52.92414	787.05	1011.484	734.2196	940.197	1.313008	0.776332	772	742.5	314.506	353.8
4	76.51772	56.25933	59.3101	35.59726	1.197273	1.032709	122.9061	32.4061	44.28946	51.93645	787.05	1011.625	734.2196	940.4404	1.313008	0.800111	772	732.6964	314.506	353.7
4	76.51772	56.07617	59.3101	35.53872	1.197273	1.048294	122.9061	31.59708	44.28946	51.7347	787.05	1011.654	734.2196	940.4891	1.313008	0.804002	772	730.7357	314.506	353.

db_superconductors

Fig 9: Database

Size of database-Excel denotation
Count-1743648
Sum-931675791.4

5.2 Techniques Used:

5.2.1 Data set creation: After visual analysis of variation of critical temperatures along with different variable we narrowed down to 8 variable from 80 parameters were derived

Variable	Units	Description
Atomic mass	atomic mass units (AMU)	total proton and neutron rest masses
Frist ionization energy	Joules per mole (kJ/mol)	energy required to remove a valence electron
Atomic radius	picometer (pm)	calculated atomic radius
Density	kilograms per meters cubed (kg/m3)	density at standard temperature and pressure
Electron affinity	kilo-Joules per mole (kJ/mol)	energy required to add an electron to a neutral atom
Fusion heat	kilo-Joules per mole (kJ/mol)	energy to change from solid to liquid without temperature change
Thermal conductivity	watts per meter-Kelvin (W/(m × K))	Thermal conductivity co-efficient
Valence	No unit	Typical number of chemical bonds formed by the element.

Table 2: This table shows the properties of an element which are used for creating features of every element present in periodic table.

Since we are considering for alloy we can convert elemental property to some kind of alloy property.
For each of these 8 elemental properties we have extended each elemental properties into 10 alloy properties.

Feature and description	Formula	Sample values
Mean	= μ = (t1 + t2)/2	35.5
Weighted mean	= ν = (p1t1) + (p2t2)	44.43
Geometric mean	= (t1t2) 1/2	33.23
Weighted geometric mean	= (t1) p1 (t2) p2	43.21
Entropy	= −w1 ln(w1) − w2 ln(w2)	0.63
Weighted entropy	= −A ln(A) − B ln(B) 0.26	0.26
Range	= t1 − t2 (t1 > t2)	25
Weighted range	= p1t1 − p2t2	37.86
Standard deviation	= [(1/2)((t1 − μ) 2 + (t2 − μ) 2)]1/2	12.5
Weighted standard deviation	= [p1(t1 − ν) 2 + p2(t2 − ν) 2)]1/2	8.75

Table 3: This table summarizes the procedure for feature extraction from material's chemical formula.

5.2.2 Feature Extraction:

5.2.3 Data Analysis

The final column serves as an example; features based on thermal conductivities for Re7Zr1are derived and mentioned to two decimal locations. Rhenium and Zirconium's thermal conductivity coefficients are t1 = 48.02 and t2 = 22.05 W/(m×k) respectively. right here: p1 = 67.05 , p2 = 1 7.01 , w1 = 48 seventy one.09 , w2 = 23 seventy one.0008 , A = p1w1 p1w1+p2w2 ≈ 0.926, B = p2w2 p1w1+p2w2 ≈ 0.07455. the feature extraction manner via an in depth instance: take into account Re7Zr1 with Tc = 6.7 k, and awareness at the features extracted based totally on thermal conductivity. Rhenium and Zirconium's thermal conductivity coefficients are t1 = forty eight and t2 = 23 W/(m×okay) respectively. The ratios of the elements within the cloth are used to outline capabilities: p1 = 6 6.001 + 1.0001 = 6 7.0011 ,

p2 = 1 6 + 1 = 1 7 . The fractions of general thermal conductivities are used as nicely: w1 = t1 t1 + t2 = 48 *forty eight + 23 = forty eight 71 , w2 = t2 t1 + t2 = 23 forty eight + 23 = 23 71.

We need a couple of intermediate values based totally on equations eq(1) and eqn(2):
A = p1w1 p1w1 + p2w2 ≈ zero.9258
B = p2w2 p1w1 + p2w2 ≈ 0.0739.

once we've received the values p1, p2, w1, w2, A, and B, we will extract 10 functions from Rhenium and Zirconium's thermal conductivities . We repeat the equal technique above with the 8 variables for example, for features primarily based on atomic mass, just update t1 and t2 with the atomic masses of Rhenium and Zirconium respectively, then carry on with the calculations of p1, p2, w1, w2, A, B, and sooner or later calculate the ten functions defined in desk (2). This gives us 8 × 10 =80 elementals. One extra capability, a numeric variable counting the range of factors in the superconductor, is also extracted. We end up with eighty one capabilities in general.
Now with the help of 80 alloy function and elemental composition table the training set is narrowed down. That is transformed into CSV (comma separated report) which can be immediately feed into this system.

5.2.3.1 Model Analysis: We have just started to investigate the database that we have. In this section we will be discussing the results of the Linear, poly regression model. Neural network tends to be the most accurate one therefore will be performing that for the most accurate system. For the start we tried plotting few graphs models to visualize the data dependencies.

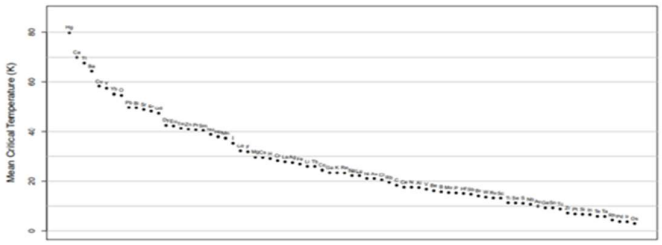
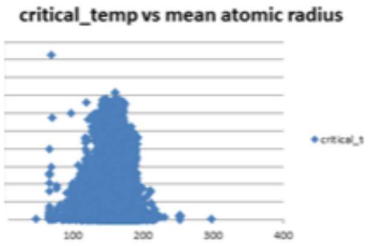


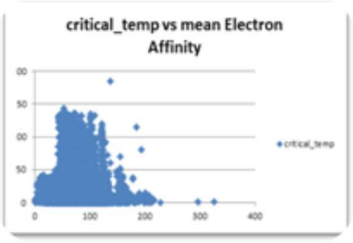
Fig 10: Graph of critical temp

5.2.4 Data Visualization

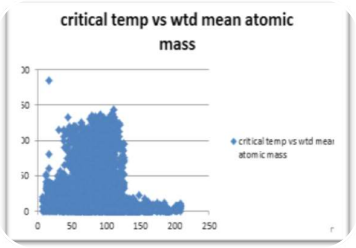
5.2.4.1 Graphs Plotted: For statistical understanding of Data.



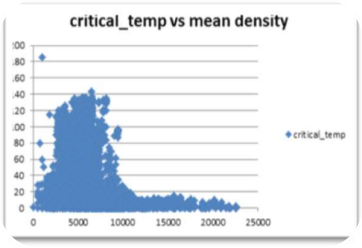
11(A)



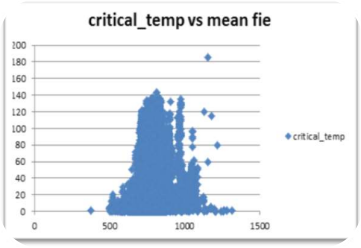
11(B)



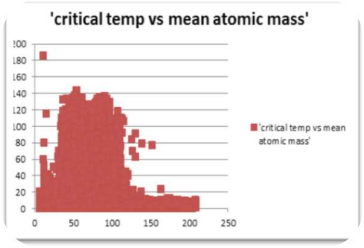
11(C)



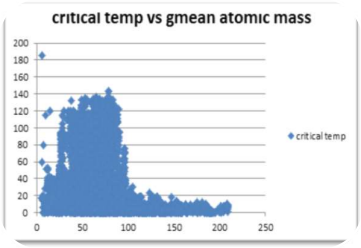
11(D)



11(E)



11(F)



11(G)

Fig 11: Statistical Data

5.2.5 Linear Regression: The training dataset include 80 alloy features along with the elemental data composition table on which the linear model is trained, which further divide into ratio of 80:20 as training set and testing sets.

Platform used for linear regression coding is Python.
CODE can be obtained from Appendix.

5.2.5.1 Parity plot: Plot to compare predicted vs actual experimental data.
R2 score for training set obtained .73 which is decent but when testing set was feed into the system R2 score decreased to .49 which is not satisfactory to predict the value. After looking at the graph we had a rough idea that the result was not close to accurate. Percentage error became quite high for testing sets.at high temperature (critical) results were close to the actual experimental values but with low temperature the %error high which is not acceptable.

5.5.2.2 Result: Not a good fit was obtained, at high temperature linear model gave moderate value but at low temperature % error became high. Therefore the predicted values were too off range that is could be considered on**.**

5.5.3 Decision Tree: Since the linear model results were not very satisfactory to identify and the variable's effect on the critical temperature decision tree is now modeled. Same dataset of 80 alloy elemental database is feed to the system, coding performed on python.
Code can be obtained from Appendix

5.5.3.1 Results:

- Score Training: 98.37 %
- Score Testing: 43.60 %

5.5.3.2 Parity Plot: Plot to compare predicted vs actual experimental data .

- .For training set : the result were more précised than linear ,a considerable improvement which can be taken into consideration score for training set was =**98.37 %**
- But when the model was feed with testing set the score dropped down to **43.60** which are better than linear but it not a significant growth.

5.5.3.3 Observation from the parity plot for decision tree:

- Over fitting observed : the model 's result for training set had much better accuracy but for testing set the results were not considerable .
- For training decision tree model fits well but for the testing is doesn't fit well.

The model cannot make a significant good prediction for not unseen experimental data.
Clustering of prediction in testing set: many points have same predicted values but the actual value is very different which form horizontal lines in parity plot.

5.5.4 Random Forest:

5.5.4.1 Result:

Score for training set =97.13 %

Score for testing set = 56.82 %

For training set the results were more precise than decision tree , as considerable improvement was seen which can be taken into consideration ,but when the model was fed with testing set the score dropped drastically ,which is better than decision but not a significant growth which can be considered.

Also a lot of features were traced which gave almost negligible impact on the final results, since they were not of much use therefore elimination of these factors were done.

6. UPDATED RESULT

6.1 Linear Regression

6.1.1 Result for linear regression:

RMSE VALUES FOR TRAINING SET linear regression: 18.175
RMSE VALUE FOR TESTING SET linear regression: 16.447
Difference is 1.728 which is considerable
The RMSE value for the linear regression is high which signifies the model does not have a good fit thus; this model cannot be considered for predicting the critical temperature.

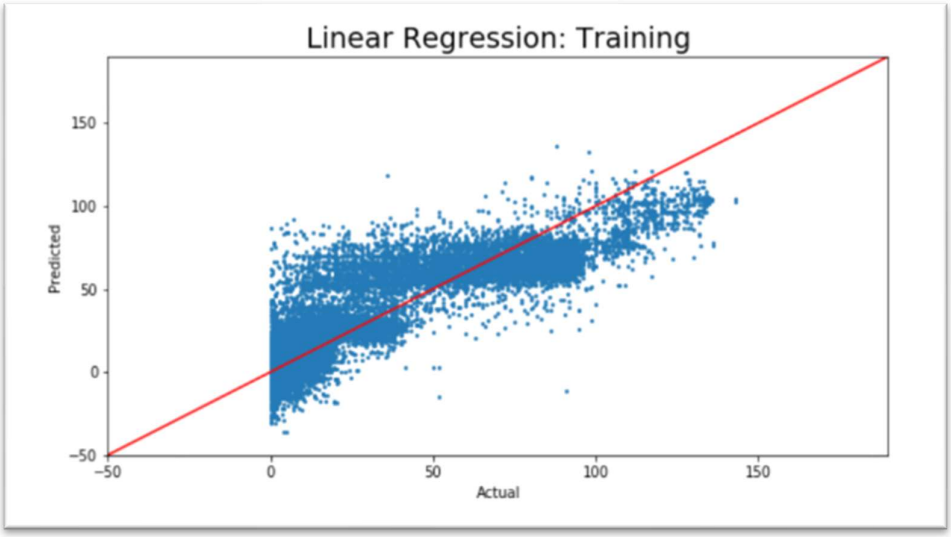


Fig 12: GRAPH of LINEAR REGREGRESSION: TRAINING

Fig 13: GRAPH of LINEAR REGRESSION: TESTING

6.2 Decision Tree

6.2.1 Result for Decision tree:

RMSE VALUES FOR TRAINING SET decision tree: 4.53

RMSE VALUE FOR TESTING SET decision tree: 22.01

The difference between the RMS value of training and testing set is 17.48 which is significantly high which reflects overfitting of the data, another reason why this model cannot be considered for predicting the critical temperature.

Fig 14: GRAPH of DECISION TREE: TRAINING

Fig 15: GRAPH of DECISION TREE: TRAINING

After plotting the result for both regression model (i.e. linear and decision) , it can be found that the decision tree has better performance than linear model but not enough , such that

predicted value can be considered as the training set seem to have high value of accuracy but not testing sets.

The results improved after applying decision tree but not to such extent that it can have significant considerations.

6.3 Random Forest

6.3.1 Result for Random forest:

RMSE VALUES FOR TRAINING SET Random forest: 5.43

RMSE VALUE FOR TESTING SET Random forest: 14.86

The difference between the rms value of training and testing set is 9.43

RMSE values for training set random forest selected feature: 5.82

RMSE value for testing set random forest selected features: 14.56

The difference between the rms value of training and testing set is 8.74

The difference decreases as compared to previous models but not satisfactory still The difference between reflects overfiting however the graphs reflected some chances for accuracy therefore parameters were selected according to the weightage and rmse value were with selected features but not much difference could be spotted , hence this model could not be considered**.**

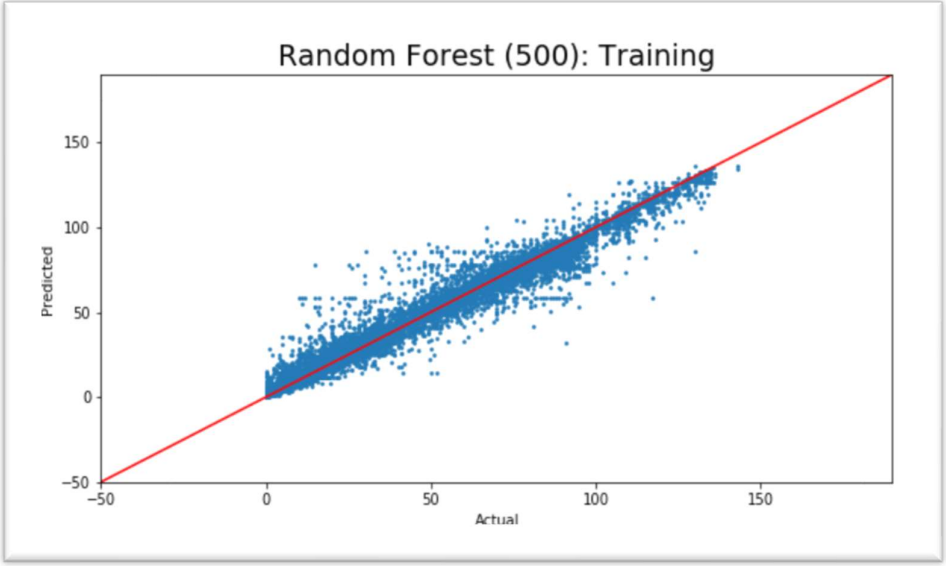


Fig 16: GRAPH of RANDOM FOREST: TRAINING

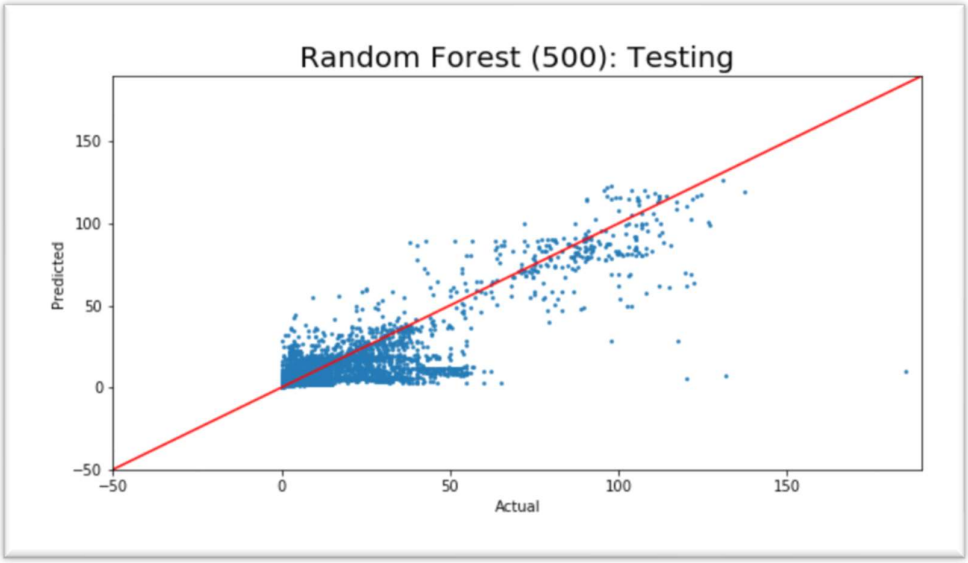


Fig 17: GRAPH of RANDOM FOREST: TESTING

6.4 Neural Network

6.4.1 Result for Neural network:

The RMSE of NN Test is: 3.065734804

The RMSE of NN Train is: 2.76290170

RMSE value decreases significantly as compared to the other models , the value of rmse also depend upon the range of the dataset values, since our dataset has significant large weightage thus the rmse value can be considered optimally low..

The difference between the training sets and testing set is 0.303 which is is small enough to be neglected. The plot of rmse value training vs testing shows linear graphs, hence this model could be considered a good fit.

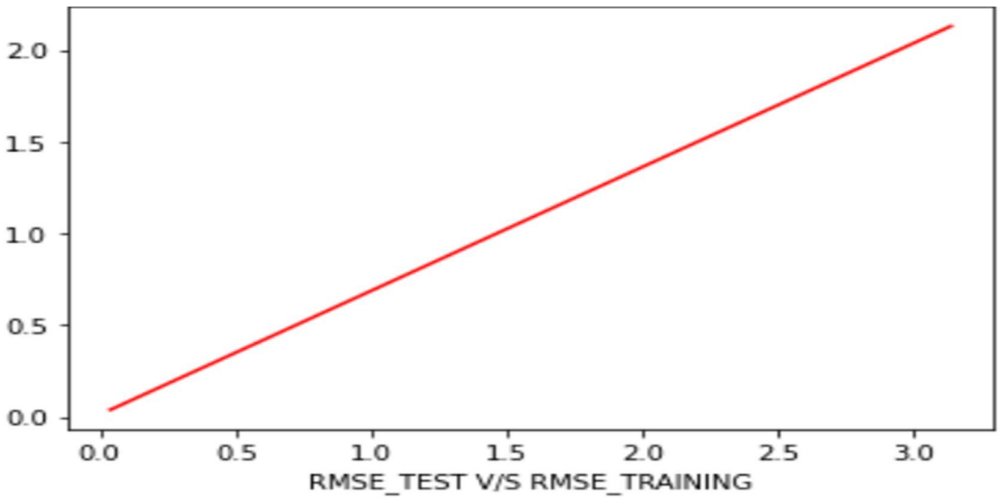


Fig 18: GRAPH of Neural Network

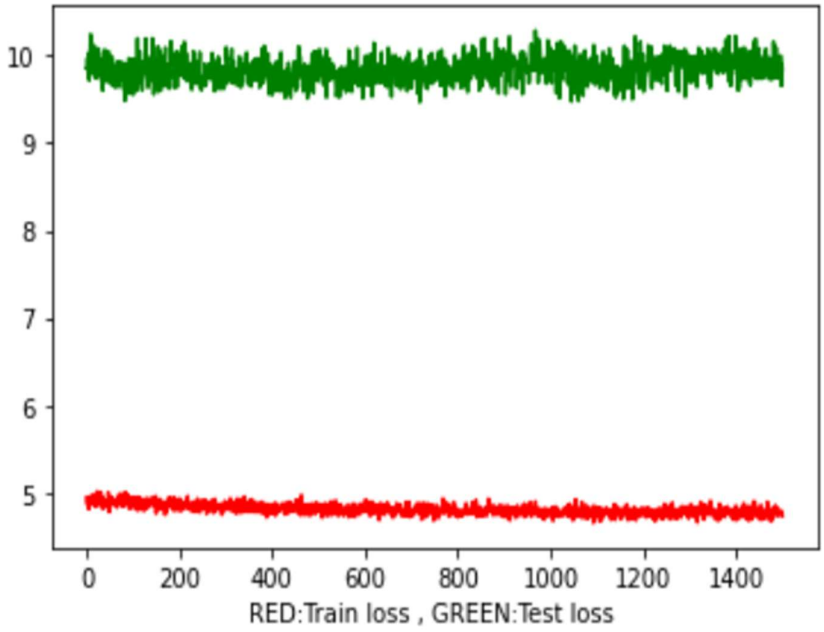


Fig 19: GRAPH of Neural Network (Training loss, Testing loss)

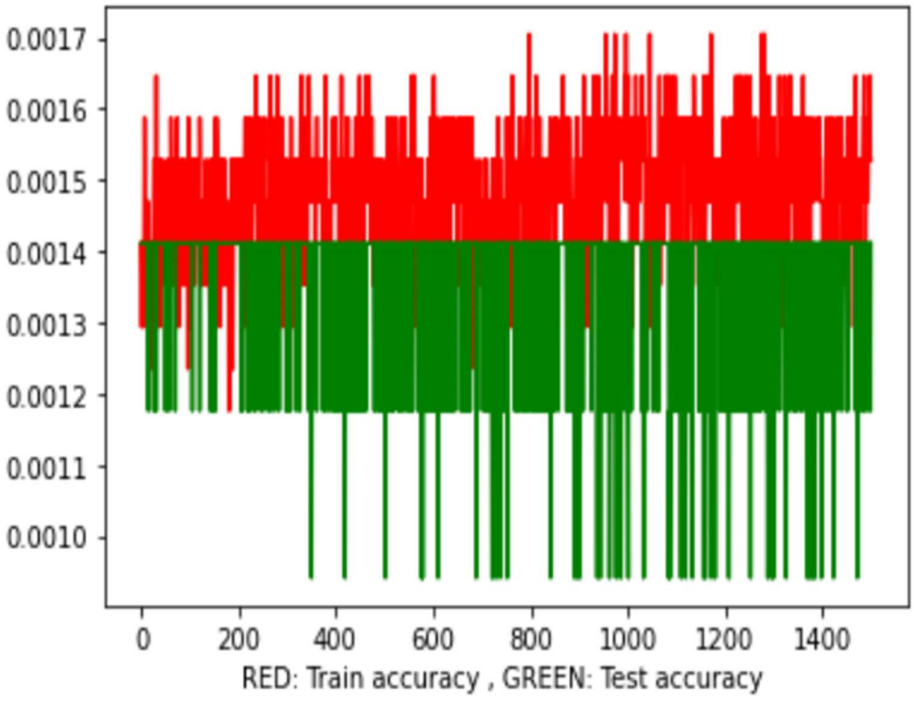


Fig 20: GRAPH of Neural Network (Training accuracy, Testing accuracy)

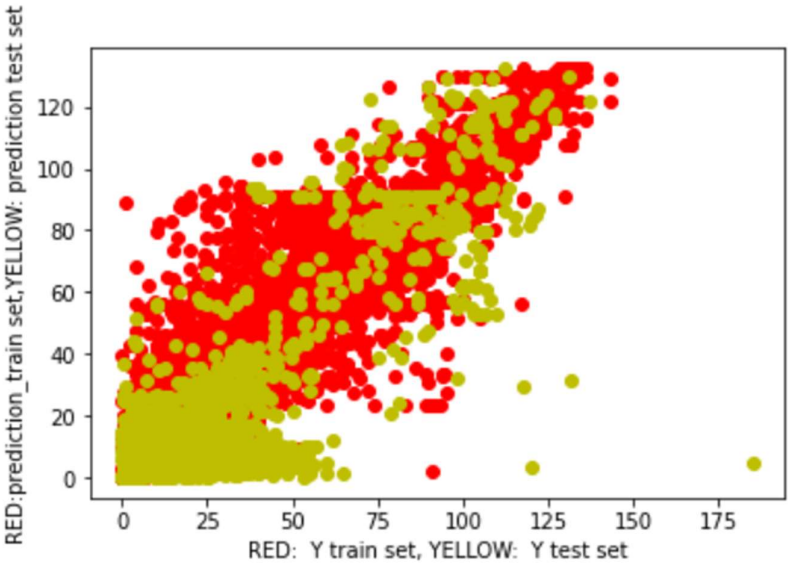


Fig 21: GRAPH of Neural Network (prediction (train set, test set))

7. CONCLUSION

s.no	Models	RMSE VALUE	
		TRAINING SET	TESTING SET
1.	Linear regression	18.17	16.44
2.	Decision tree	4.53	22.01
3.	Random forest	5.43	14.86
4.	Random forest (selected parameters)	5.43	14.56
5.	Neural network	2.76	3.06

Table 4: Comparison table for RMSE values

7.1 TRAINING: RMSE value in descending order:

LINEAR > RANDOM FOREST > RMSE RANDOM FOREST (SELECTED PARAMETERS)>decision tree > neural network

7.2 Testing: MSE value in descending order:

Decision tree> linear regression>random forest selected parameters > random forest >neural network

The value of the **RMSE** decreased significantly using neural networks, the lower the value the better the model with this dataset the value least that could be attained is by neural network, "**RMSE** value for training and testing set is close then model is considered to be a good fit "as calculated the difference in **RMSE** value for neural network is 0.3 which is significantly low and can be neglected.

With all the models, looking at **RMSE** values neural network tend to have the smallest value out of all, at the same time the difference between training and testing set is low which signifies that neural network can be considered to be a good fit for predicting the critical temperatures.

7. REFERENCES

[1]. Kam HamidiehUniversity of Pennsylvania, Wharton, Statistics DepartmentScientific Reports 3, 1–6 (2013).
[2]. Shaobo Li 1, Yabo Dan 1,*, Xiang Li 1, Tiantian Hu 2, Rongzhi Dong 1, Zhuo Cao 1 and Jianjun Hu 1,3,*Published: 8 February 2020
[3]. Materials informatics for the screening of multi-principal elements and high-entropy alloys J.M. Rickman1,2, H.M. Chan2, M.P. Harmer2, J.A. Smeltzer2, C.J. Marvel2, A. Roy3 & G. Balasubramanian3 (2019) 10:2618 |
[4]. Material informatics for layered high-TC Superconductors Zhong-Li Liu , Peng Kang , Yu Zhu, Lei Liu, and Hong Guo ,APL Mater. 8, 061104 (2020).

8. APPENDIX

Code:

Libraries used:

```
import pandas as pd
import numpy as np
import sklearn as sklearn
from sklearn import linear_model
from sklearn.preprocessing import PolynomialFeatures as poly
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from matplotlib import pyplot as plt
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
from collections import defaultdict
from scipy.stats import spearmanr
from scipy.cluster import hierarchy

from sklearn.ensemble import RandomForestClassifier
from sklearn.inspection import permutation_importance
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import load_model
from tensorflow.keras.layers import Dense
from sklearn.metrics import accuracy_score
```

Importing Data

```
db_SC = pd.read_csv("db_superconductors.csv"); #dataframe structure
db_Comp = pd.read_csv("db_composition.csv");
db_SC.shape
feats = db_SC.columns[1:-1];
feats.shape
X = db_SC[feats];
Y = db_SC["critical_temp"];
print(X.shape)
print(Y.shape)
train_size = 0.8;

X_train_unshuffled = np.array(X[:int(X.shape[0]*train_size)]);
X_test_unshuffled = np.array(X[int(X.shape[0]*train_size):]);

Y_train_unshuffled = np.array(Y[:int(Y.shape[0]*train_size)]);
Y_test_unshuffled = np.array(Y[int(Y.shape[0]*train_size):]);

shuffler_train = np.random.permutation(len(Y_train_unshuffled));
```

```
X_train = X_train_unshuffled[shuffler_train];
Y_train = Y_train_unshuffled[shuffler_train].reshape((X_train.shape[0],
1))

shuffler_test = np.random.permutation(len(Y_test_unshuffled));
X_test = X_test_unshuffled[shuffler_test];
Y_test = Y_test_unshuffled[shuffler_test].reshape((X_test.shape[0],1));

xmin = np.amin(X_train, axis=0);
xmax = np.amax(X_train, axis=0);

X_train = (X_train-xmin)/(xmax-xmin);
X_test = (X_test-xmin)/(xmax-xmin);

print("X_train set shape:",X_train.shape)
print("X_test set shape:",X_test.shape)
print("Y_train set shape:",Y_train.shape)
print("Y_test set shape:",Y_test.shape)
```

Linear modelling

Assumption : Critical temperature is function of 80 feature .

Y= critical temperature.

X1,x2,x3: alloy features.

Y=f(x1,x2,x3,……..,x80).

```
LS_reg = linear_model.LinearRegression();   # create linear_reg method
LS_reg.fit(X_train, Y_train);               # perform lin_regression
print('Coeff:', LS_reg.coef_);              # print coefficients i.e. w
eights
print('\nIntercept:',LS_reg.intercept_);    # print intercept i.e. bias


# Training Performance:
pred_train = LS_reg.predict(X_train);
 # training set predictions
R2_LS_train = r2_score(Y_train, pred_train);
 # trainig R2 score
MAE_LS_train = mean_absolute_error(Y_train, pred_train);
 # training mean absolute error
RMSE_LS_train = mean_squared_error(Y_train, pred_train)**0.5;
 # training root mean square error

avg_per_err_train = np.around(np.mean(
                    np.absolute((pred_train -
 Y_train)/Y_train)*100),decimals=3);

# printing training set results
print('\nR2 score(Training set) = %.3f'%R2_LS_train);
print('MAE_Least Square(Training set) = %.3f'%MAE_LS_train);
print('RMSE_Least Square(Training set) = %.3f'%RMSE_LS_train);
```

```
print('Avg. % Error (Training set) = ',avg_per_err_train);

# Testing Performance
pred_test = LS_reg.predict(X_test);
 # testing set predictions
R2_LS_test = r2_score(Y_test, pred_test);
 # testing R2 score
MAE_LS_test= mean_absolute_error(Y_test, pred_test);
 # testing mean absolute error
RMSE_LS_test= mean_squared_error(Y_test, pred_test)**0.5;
 # testing root mean square error

avg_per_err_test = np.around(np.mean(
        np.absolute((pred_test - Y_test)/Y_test)*100),decimals=3
        );
 # testing: average percentage error

# Printing testing set results
print('\nR2 score(Testing set) = %.3f'%R2_LS_test);
print('MAE_Least Square(Testing set) = %.3f'%MAE_LS_test);
print('RMSE_Least Square(Testing set) = %.3f'%RMSE_LS_test);
print('Avg. % Error (Testing set) = ', avg_per_err_test);
plt.figure(figsize=(10,5));
plt.scatter(Y_train, pred_train, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Linear Regression: Training",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
plt.figure(figsize=(10,5));
plt.scatter(Y_test, pred_test,s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Linear Regression: Testing",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
```

Decision Tree modelling

```
dt_function = tree.DecisionTreeRegressor();
dt_mod = dt_function.fit(X_train, Y_train);

train_score = dt_mod.score(X_train,Y_train)*100;
test_score = dt_mod.score(X_test,Y_test)*100;

print("\nScore Training: %.2f "%train_score,"%");
print("\nScore Testing: %.2f "%test_score,"%");
```

```
print("\nFeature Importance: ");
print(dt_mod.feature_importances_);
pred_train_DT = dt_mod.predict(X_train);
pred_test_DT = dt_mod.predict(X_test);
RMSE :
mse_Train = mean_squared_error(Y_train,pred_train_DT)
rmse_Train = mse_Train ** (1/2)

# Print rmse_dt
print("Train set RMSE of dt: {:.2f}".format(rmse_Train))
mse_Test = mean_squared_error(Y_test,pred_test_DT)
rmse_Test = mse_dt ** (1/2)

# Print rmse_dt
print("Test set RMSE of dt: {:.2f}".format(rmse_Test))


plt.figure(figsize=(10,5));
plt.scatter(Y_train, pred_train_DT, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Decision Tree: Training",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
plt.figure(figsize=(10,5));
plt.scatter(Y_test, pred_test_DT,s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Decision Tree : Testing",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
```

Random forest modelling

```
n_trees = 10;
RF_function = RandomForestRegressor(n_estimators=int(n_trees))
RF_mod = RF_function.fit(X_train, Y_train)

train_score = RF_mod.score(X_train,Y_train)*100;
test_score = RF_mod.score(X_test,Y_test)*100;

print("\nScore Training: %.2f "%train_score,"%");
print("\nScore Testing: %.2f "%test_score,"%");

print("\nFeature Importance: ");
print(RF_mod.feature_importances_);
pred_train_RF = RF_mod.predict(X_train);
pred_test_RF = RF_mod.predict(X_test);
```

```
RMSE VALUES:
mse_TrainRF = mean_squared_error(Y_train,pred_train_RF)
rmse_TrainRF = np.sqrt(mse_TrainRF)

# Print rmse_RF
print("Train set RMSE of Random forest: {:.2f}".format(rmse_TrainRF))
mse_TestRF = mean_squared_error(Y_test,pred_test_RF)
rmse_TestRF = np.sqrt(mse_TestRF)

# Print rmse_RF
print("Test set RMSE of Random forest: {:.2f}".format(rmse_TestRF))

plt.figure(figsize=(10,5));
plt.scatter(Y_train, pred_train_RF, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Random Forest (" + str(n_trees) + "): Training",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
plt.figure(figsize=(10,5));
plt.scatter(Y_test, pred_test_RF, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Random Forest (" + str(n_trees) + "): Testing",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
def f_plot_tree(clf):
    fig = plt.figure(figsize=(100,100))
    _ = tree.plot_tree(clf,
                       feature_names=feats,
                        class_names=["F","B","F+B","F+I","B+I","F+B+I","I"]
,
                       filled=True)


def f_plot_importance(mod, X, Y, feats, n):

    tree_MDI_importance = mod.feature_importances_;
    tree_importance_sorted_idx = np.argsort(tree_MDI_importance);
    tree_indices = np.arange(0, len(mod.feature_importances_)) + 0.5;

    feats = np.array(feats)


    p_imp_result = permutation_importance(mod, X, Y, n_repeats=int(n));
    perm_sorted_idx = p_imp_result.importances_mean.argsort();
```

```
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 11));

    ax1.barh(tree_indices,
             tree_MDI_importance[tree_importance_sorted_idx], height=0.
7);
    ax1.set_yticklabels(feats[tree_importance_sorted_idx], fontsize=14)
;
    ax1.set_yticks(tree_indices);
    ax1.set_ylim((0, len(mod.feature_importances_)));
    ax1.set_title("MDI Feature Importances", fontsize=20) # Mean Decrea
se in Impurity (MDI)

    ax2.boxplot(p_imp_result.importances[perm_sorted_idx].T, vert=False
);
    ax2.set_yticklabels(feats[perm_sorted_idx],fontsize=14);
    ax2.set_title("Permutation Feature Importances", fontsize=20)


    fig.tight_layout();
    plt.show();

    return (perm_sorted_idx)
sort_feat_index = f_plot_importance(dt_mod, X, Y, feats, 10);
sorted_feats = feats[sort_feat_index];
X_train_imp = X_train[:,sort_feat_index[0:10]];
X_test_imp = X_test[:,sort_feat_index[0:10]];

n_trees = 10;
RF_function = RandomForestRegressor(n_estimators=int(n_trees))
RF_mod = RF_function.fit(X_train_imp, Y_train)

train_score = RF_mod.score(X_train_imp,Y_train)*100;
test_score = RF_mod.score(X_test_imp,Y_test)*100;

print("\nScore Training: %.2f "%train_score,"%");
print("\nScore Testing: %.2f "%test_score,"%");

print("\nFeature Importance: ");
print(RF_mod.feature_importances_);

pred_test_RF = RF_mod.predict(X_test_imp);

plt.figure(figsize=(10,5));
plt.scatter(Y_test, pred_test_RF, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Random Forest (" + str(n_trees) + "): Testing",size=20);
plt.xlabel("Actual",size=10);
```

```
plt.ylabel("Predicted",size=10);
sort_feat_index[0:10]
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 11))

corr = spearmanr(X).correlation
corr_linkage = hierarchy.ward(corr)
dendro = hierarchy.dendrogram(
    corr_linkage, labels=feats, ax=ax1, leaf_rotation=90)
dendro_idx = np.arange(0, len(dendro['ivl']))

ax2.imshow(corr[dendro['leaves'], :][:, dendro['leaves']])
ax2.set_xticks(dendro_idx)
ax2.set_yticks(dendro_idx)
ax2.set_xticklabels(dendro['ivl'], rotation='vertical')
ax2.set_yticklabels(dendro['ivl'])
fig.tight_layout()
plt.show()
luster_ids = hierarchy.fcluster(corr_linkage, 1, criterion='distance')
cluster_id_to_feature_ids = defaultdict(list)

for idx, cluster_id in enumerate(cluster_ids):
    cluster_id_to_feature_ids[cluster_id].append(idx)

selected_features = [v[0] for v in cluster_id_to_feature_ids.values()]

X_train_sel = X_train[:, selected_features]
X_test_sel = X_test[:, selected_features]

n_trees = 10;
RF_function = RandomForestRegressor(n_estimators=int(n_trees));
RF_mod_sel = RF_function.fit(X_train_sel, Y_train);

train_score = RF_mod_sel.score(X_train_sel,Y_train)*100;
test_score = RF_mod_sel.score(X_test_sel,Y_test)*100;

print("\nScore Training: %.2f "%train_score,"%");
print("\nScore Testing: %.2f "%test_score,"%");

pred_test_RF_sel = RF_mod_sel.predict(X_test_sel);
pred_test_RF_sel = RF_mod_sel.predict(X_test_sel);
pred_train_RF_sel = RF_mod_sel.predict(X_train_sel);
RMSE VALUE:
mse_TestRF_SE = mean_squared_error(Y_test,pred_test_RF_sel)
rmse_TestRF_SE = np.sqrt(mse_TestRF_SE)
mse_TrainRF_SE = mean_squared_error(Y_train,pred_train_RF_sel)
rmse_TrainRF_SE = np.sqrt(mse_TrainRF_SE)

# Print rmse_RF
```

```
print("Test set RMSE of Random forestwith selected features : {:.2f}".f
ormat(rmse_TrainRF_SE))


# Print rmse_RF
print("Test set RMSE of Random forestwith selected features : {:.2f}".f
ormat(rmse_TestRF_SE))


plt.figure(figsize=(10,5));
plt.scatter(Y_test, pred_test_RF_sel, s=3);
plt.plot([-50,190],[-50,190],color="r");
plt.xlim(-50,190);
plt.ylim(-50,190);
plt.title("Random Forest Selected Features (" + str(n_trees) + "): Test
ing",size=20);
plt.xlabel("Actual",size=10);
plt.ylabel("Predicted",size=10);
```

Neural Network

```
NN=Sequential()

NN.add(Dense(80,activation="relu"))
NN.add(Dense(64,activation="relu"))
NN.add(Dense(30,activation="relu"))
NN.add(Dense(30,activation="sigmoid"))
NN.add(Dense(1,activation="relu"))
NN.compile(optimizer='adam',loss='mean_absolute_error',metrics='accurac
y')
NN.fit(X_train,Y_train,epochs=1500,validation_data=(X_test,Y_test))
print("Evaluation started.....................")
NN_fit =NN.fit(X_train,Y_train,epochs=1500,validation_data=(X_test,Y_te
st))
RMSE VALUES:
NN_predicted=NN.evaluate(x=X_test,y=Y_test)
RMSE_Test=np.sqrt(NN_predicted )
print("The RMSE of NN Test is:",RMSE_Test)
NN_predicted_train=NN.evaluate(x=X_train,y=Y_train)
RMSE_Train=np.sqrt(NN_predicted_train )
print("The RMSE of NN Train is:",RMSE_Train)
from sklearn.metrics import mean_squared_error
from matplotlib import pyplot as plt
train_loss=NN_fit.history['loss']
test_loss=NN_fit.history['val_loss']
plt.plot(train_loss,c='r')
plt.plot(test_loss,c='g')
Prediction_test=NN.predict(X_test)
Prediction_train=NN.predict(X_train)
```

```
plt.scatter(Y_train,Prediction_train)
plt.scatter(Y_train,Prediction_train, c='r')
plt.scatter(Y_test,Prediction_test,c='y')
```