Predicting the Critical Temperature of Superconductors

Reshma Sri Challa
Computer Science
University of Ottawa
Ottawa,Canada
rchal050@uottawa.ca

Dharna Shukla
Computer Science
University of Ottawa
Ottawa,Canada
dshuk066@uottawa.ca

***Abstract*— A superconductor is an important material used in many practical applications. Depending on their critical temperature they are used for various applications. We used different machine learning regression and classification techniques to determine the critical temperature and its level of superconductivity. Regression is used to predict T_c (Critical Temperature) and classification to predict the level of superconductivity (Low or high).The models used here are Decision Tree, Linear regression, Support vector Machine (SVM), K-Nearest Neighbor (KNN), Gradient Boosting, bagging.**

Keywords—Superconductivity, Critical Temperature, Decision Tree, Linear Regression, Support vector Machine (SVM), Gradient Boosting, K-Nearest Neighbor (KNN).

I. Introduction

This is an application-oriented project. The dataset chosen is based on superconductivity. Superconductivity is a phenomenon of exactly zero electrical resistance and expulsion of magnetic flux fields occurring in certain materials, called superconductors, when cooled below a characteristic critical temperature. The low temperature semiconductors are used for Magnetic Resonance Imaging (MRI), Nuclear Magnetic Resonance (NMR), Particle accelerators, magnetic fusion devices, Electric power transmission etc. [7]. The high temperature superconductors are used in applications like magnetically levitated trains and power transmission. So, predicting the critical temperature of the metal is necessary. Conventional superconductors usually have critical temperature ranging from around 20 K to less than 1 K which is the base of our classification model. Predicting this critical temperature is done in the following steps from figure 1.

A. Feature Extraction

Feature extraction is done in the following ways:

1. Correlation HeatMap:

A correlation heatmap is portrayed in a 2D matrix connecting two discrete dimensions using colored cells. The values of the first dimensions are the rows of the table and the values of the second are the columns of the table. The color of each cell is directly proportional to the number of measurements that match the dimensional values. This helps us to identify the patterns, the correlation between the attributes with the critical temperature and to recognize anomalies.

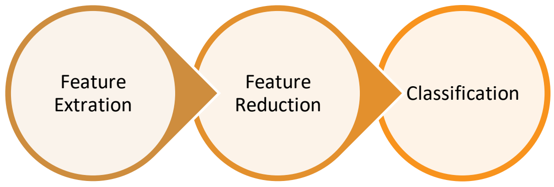


Figure 1: Process to predict the critical temperature of superconductors

2. Univariate Selection

To select the features having the strongest relationship with the critical temperature,statistical tests are used the SelectKBest class in the scikit-learn library is used to select a specific number of features. This assigns the scores for each attribute. The higher the score the more significant it is to the critical temperature in our dataset.

3. Normalization:

The objective of normalization is to scale the numeric column values of the dataset to a common scale, without losing information. We have used two ways here, general normalization between 0 and 1 and the other MinMaxScaler. The techniques MinMaxScaler allows us to define the range in scikit learn, whereas the normal normalization function is between 0 and 1 and this might create anomalies in our dataset.

$$F(x) = \frac{x_i - \min(x)}{\max(x) - \min(x)} \qquad (1)$$

B. *Feature Reduction*

After Feature extraction we perform Feature Reduction using Principal Component Analysis (PCA). We can explain that the aim of PCA is to decide that Y- the unit basis vector on the x-axis - is the most crucial dimension. Determining this reality permits an experimenter to anticipate which dynamics is necessary, which are just excessive and that is just noise. In the data set operated for this project, at a point in instant, there are 80 columns. There are converted into a matrix with 2 components, as in 80 rows and 2 columns.

1. Test Options:

i) Test-Train Split

The test size parameter decides the size of the data that has to be split as the test dataset. The train size parameter simplifies size of the data that has to be split into test dataset.

ii) K-Fold

Cross-validation is a method used to in predictive models to partition the original data into a training set and a test set. Here, the original sample is randomly segmented into k subsamples which are equal sized. Then, a one subsample from the k subsamples is used for testing and the rest of the k-1 subsamples are used for training. This process is then repeated k times. The k results from these k folds are combined to generate a single evaluation.

C. ***Classification***

The machine learning techniques used for this dataset are regression and classification. Regression and classification are categorized under supervised machine learning as shown in the figure 2. In supervised learning, an algorithm is used to learn the mapping function from the input variable (x) to the output variable (y) i,e. $y = f(X)$. The main distinction between these two is that the output variable in regression is numerical or continuous value but for classification it is categorical or discrete. Examples of common regression algorithms include linear regression, Support Vector Regressor (SVR) and regression trees. For this project we are going to use linear regression, KNN, Gradient Boosting and decision tree regressor.

This dataset is a regression problem, but we are converting it into a classification problem by introducing the two classes as discussed below. The classification task is based on the low and high superconductors, i,e, class 1 is T_c <=20K (Low superconductivity) and class 2 is T_c >20K (high superconductivity).

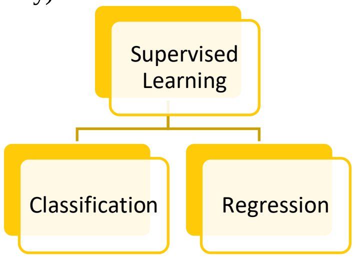


Figure 2: Supervised Learning types

D. *Performance Analysis*

1) Mean Absolute Error (MAE)

The Mean Absolute Error measures the average of the absolute difference between each ground truth and the predictions.

$$MAE = \frac{1}{n}|y_j - \hat{y}_j| \qquad (2)$$

Here,
$\hat{y}$ is the prediction,
y is the actual,
n is the number of instances

2) Root Mean Squared Error (RMSE)

The Root Mean Squared Error measures the square root of the average of the squared difference between the predictions and the ground truth.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(|y_j - \hat{y}_j|)^2}{N}} \qquad (3)$$

Here,
$\hat{y}$ is the prediction,
y is the actual,
N is the number of instances

3) Variance

Variance refers to an algorithm's sensitivity to specific sets of the training set occurs when an it has a limited flexibility to learn the true signal from the original data.

4) Correlation Coefficient

Correlation Coefficient helps you find out the relationship between two quantities. It gives you the measure of the strength of association between two variables. The value of Correlation Coefficient can be between -1 to +1. 1 means that they are highly correlated and 0 means no correlation. -1 means that there is a negative correlation.

5) Accuracy

$$acc = pos * TPR + neg * TNR \qquad (4)$$

Where,

$$True\ Negative\ Rate = TNR = \frac{TN}{Neg} = \frac{TN}{TN+FP} \qquad (5)$$

	Pred(+)	Pred(-)
Actual (POS)	TP	FN
Actual (NEG)	FP	TN

Table 1: Confusion Matrix for classification

6) Precision

$$precision, confidence = \frac{TP}{TP+FP} \qquad (6)$$

7) Recall

$$True\ Positive\ Rate/Recall = TPR = \frac{TP}{pos} = \frac{TP}{TP+FN} \qquad (7)$$

8) F1 Score

$$F1 = \left(\frac{\frac{1}{recall}+\frac{1}{precision}}{2}\right)^{-1} \qquad (8)$$

II. Case Study

We have downloaded the data set from UCI (Machine learning Repository), Superconductivity data dataset [1]. This is a numerical dataset containing 21263 instances and 81 attributes and a column for the critical Temperature. It contains various attributes that describe the properties of the superconductor (Eg: atomic mass, number of elements, entropy, atomic radius, etc.). The Tc ranges from less than 1K to 185K.The dataset had few zero values in between; these were replaced by the mean values of that attribute.

Feature Extraction, Feature reduction, Classification all were implemented in scikit learn, but rule-based algorithm was implemented using WEKA tool. (Weka results can be found in the weka folder in the submission file)We have used many classifiers and regressors but we settled on the ones in table 2 , as they were more suitable to our dataset giving us better results in terms so the performance measures we have chosen.

Type	Regressor	Classifier
Linear classifier	Linear Regression	Logistic Regression, SVM , Naive Bayes
Distance Based	KNN	KNN
Tree Based	Decision Tree	Random Forest
Rule Based	Decision Table	Decision Table
Ensemble	Gradient Boosting	Bagging, Boosting

Table 2: Depicts the regressors and classifiers used for each type of Model

III. Experimental setup and evaluation

Now we will see each of the steps seen in introduction with respect to our case study discussed above.

A. Feature Extraction

1. Correlation HeatMap:

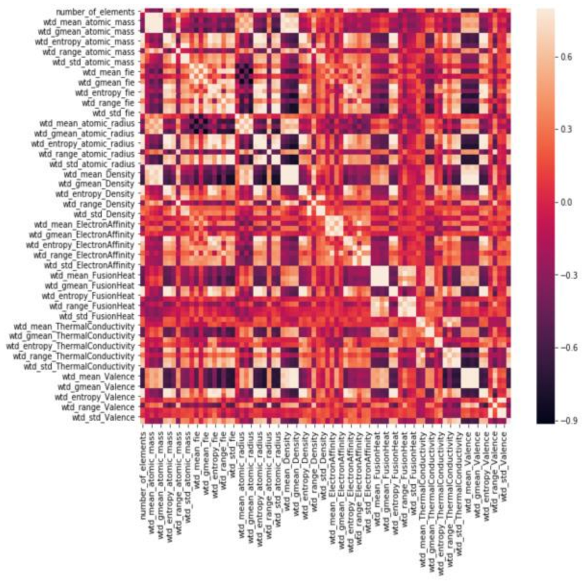


Figure 3:Correlation between all the attributes

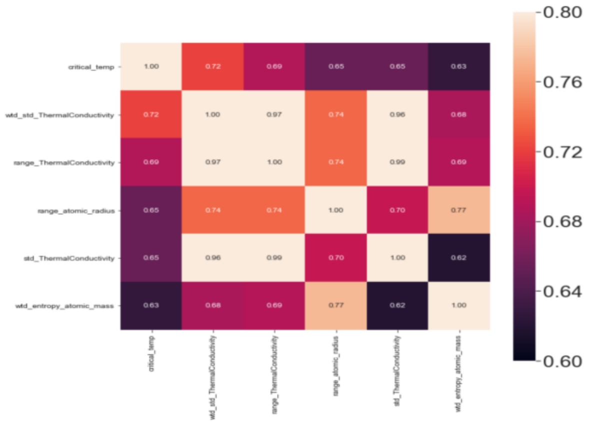


Figure 4:Corelation heatmap number between the important features

number_of_elements	0.601069
entropy_atomic_mass	0.543619
wtd_entropy_atomic_mass	0.626930
entropy_fie	0.567817
range_fie	0.600790
std_fie	0.541804
wtd_std_fie	0.582013
entropy_atomic_radius	0.558937
wtd_entropy_atomic_radius	0.603494
range_atomic_radius	0.653759
std_atomic_radius	0.559629
wtd_std_atomic_radius	0.599199
gmean_Density	0.541684
wtd_gmean_Density	0.540046
entropy_FusionHeat	0.552709
wtd_entropy_FusionHeat	0.563244
range_ThermalConductivity	0.687654
std_ThermalConductivity	0.653632
wtd_std_ThermalConductivity	0.721271
mean_Valence	0.600085

wtd_mean_Valence	0.632401
gmean_Valence	0.573068
wtd_gmean_Valence	0.615653
entropy_Valence	0.598591
wtd_entropy_Valence	0.589664
critical_temp	1.000000

Table 3: List of important features with correlation > 0.5

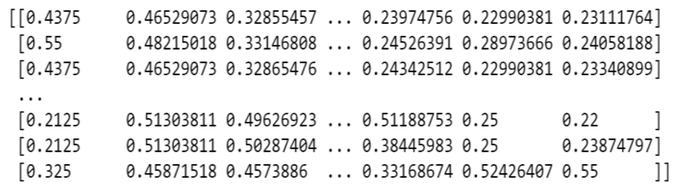
2. Univariate Selection

Specific features	Score
wtd_std_ThermalConductivity	23054.166196
range_ThermalConductivity	19072.343588
range_atomic_radius	15869.689816
std_ThermalConductivity	15858.920670
wtd_mean_Valence	14169.928184
wtd_entropy_atomic_mass	13767.767217
wtd_gmean_Valence	12977.312840
wtd_entropy_atomic_radius	12179.023271
number_of_elements	12026.064871
range_fie	12008.649352

Table 4: Scores of the important features

3. Normalization:

After comprehensive study, we concluded that for this data set MInMaxScaler better suited, as zero normalization between 0 and 1 created a lot of zeros and this degraded the performance of the models.



```
[[0.4375     0.46529073 0.32855457 ... 0.23974756 0.22990381 0.23111764]
 [0.55       0.48215018 0.33146808 ... 0.24526391 0.28973666 0.24058188]
 [0.4375     0.46529073 0.32865476 ... 0.24342512 0.22990381 0.23340899]
 ...
 [0.2125     0.51303811 0.49626923 ... 0.51188753 0.25       0.22      ]
 [0.2125     0.51303811 0.50287404 ... 0.38445983 0.25       0.23874797]
 [0.325      0.45871518 0.4573886  ... 0.33168674 0.52426407 0.55      ]]
```

Figure 5:Scaled Features after using MinMaxScaler

After implementing the above feature extraction methods, it was observed that all the 80 features are equally important to predict T_c and scaling all these features in the range 0.1 to 1 using MinMaxScaler improved the results, which will be further discussed in the performance metrics below

B. *Feature Reduction*

Like mentioned above PCA algorithm was used for Feature Reduction and the results are shown below in Figure 6 and figure 7.

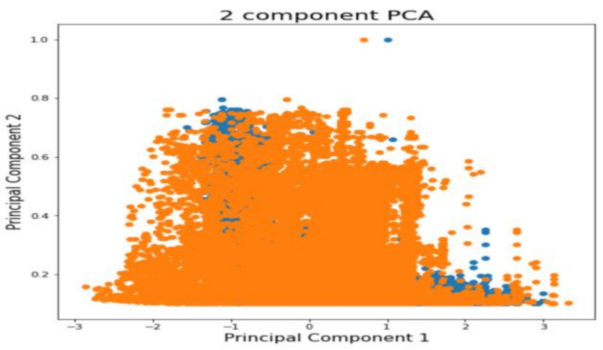
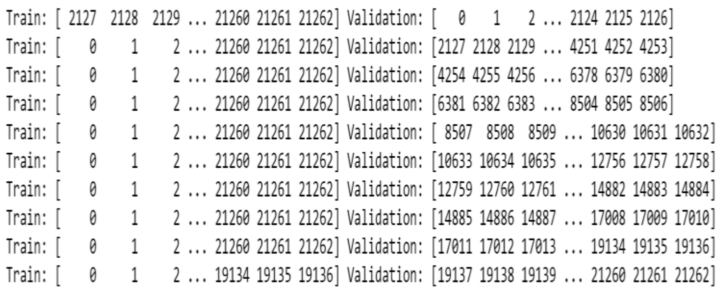


Figure 6: Feature reduction using PCA



```
Train: [ 2127  2128  2129 ... 21260 21261 21262] Validation: [   0    1    2 ... 2124 2125 2126]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [2127 2128 2129 ... 4251 4252 4253]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [4254 4255 4256 ... 6378 6379 6380]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [6381 6382 6383 ... 8504 8505 8506]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [ 8507  8508  8509 ... 10630 10631 10632]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [10633 10634 10635 ... 12756 12757 12758]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [12759 12760 12761 ... 14882 14883 14884]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [14885 14886 14887 ... 17008 17009 17010]
Train: [    0     1     2 ... 21260 21261 21262] Validation: [17011 17012 17013 ... 19134 19135 19136]
Train: [    0     1     2 ... 19134 19135 19136] Validation: [19137 19138 19139 ... 21260 21261 21262]
```

Figure 7:Cross Validation Results after using MinMaxScaler and PCA

C. *Classification*

Have used all the combinations of classification and regression models with all the above feature extraction methods and the results are shown below.

1) Regression

The table 6 contains the results before the feature reduction PCA algorithm was introduced. Once after feature extraction when PCA was introduced,better results were seen for MinMaxScaler and PCA together, shown in Table 5. K-Fold cross validation had better results than Test-Train Split for regression. The results are shown below and the reasons to this are justified in the performance analysis section. The decision table has weka results in table 5.

	MAE	RMSE	Variance	CC
Linear Regression	0.0697	0.0898	0.1932	0.501
KNN	0.037	0.062	0.496	0.711
Decision Tree	0.043	0.075	0.250	0.610
Gradient Boosting	0.0423	0.065	0.448	0.679
Decision Table	0.0759	0.1063	-	0.770

Table 5: Results of MinMaxScaler and PCA with regression various model using K-Fold Cross Validation

Now, lets see the results of each regression model with MinMaxScaler and PCA.

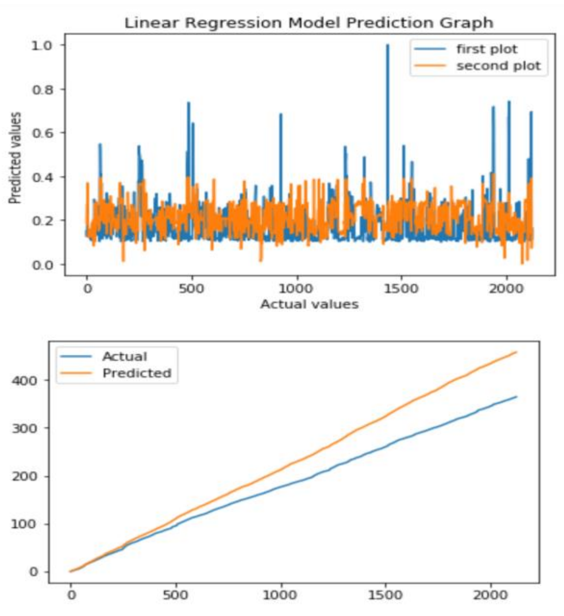


Figure 8,9: Plotting the actual and predicted values of Linear Regression model

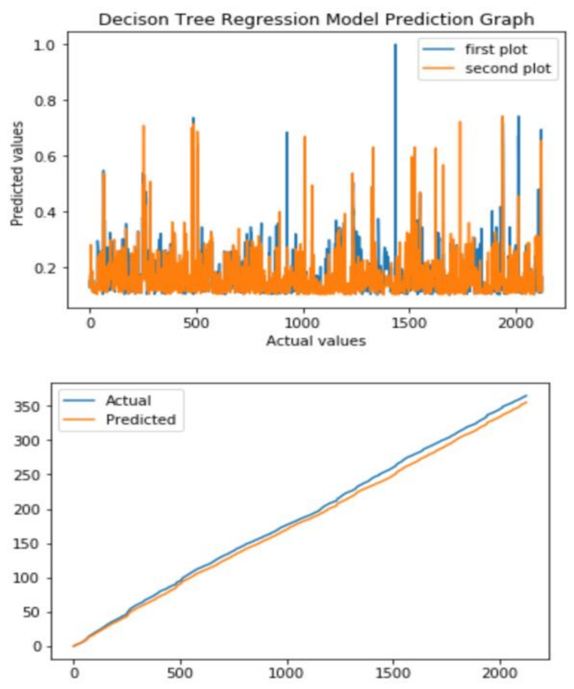


Figure 12,13: Plotting the actual and predicted values of Decision Tree Model

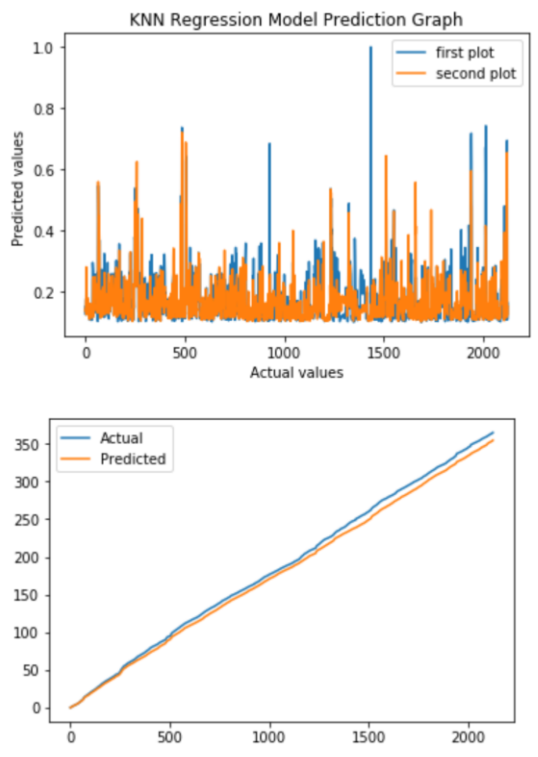


Figure 10,11: Plotting the actual and predicted values of K-Nearest Neighbour Mode

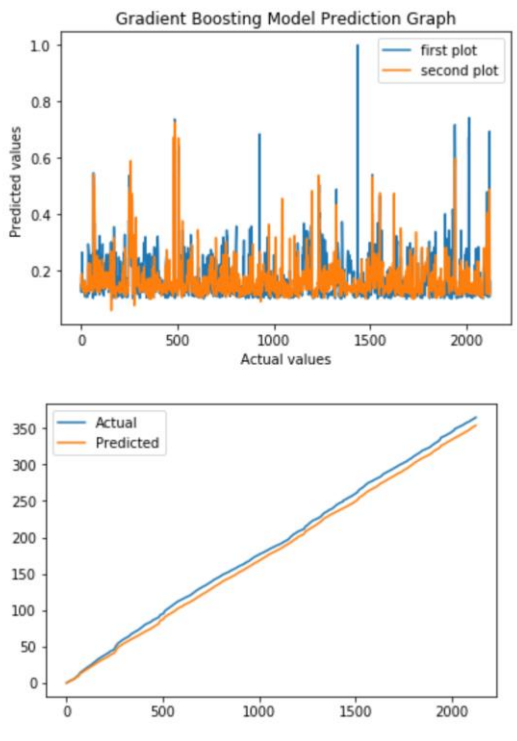


Figure 14,15: Plotting the actual and predicted values of Gradient boosting Model

2) Classification

Once after feature extraction when PCA was introduced,better results were seen for MinMaxScaler and PCA together, shown in Table 7.For Classification Test-Train Validation had better results than K-fold cross validation. The results are shown below and the reasons to this are justified in the performance analysis section.

	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.897	0.928	0.863	0.895
KNN	0.940	0.955	0.926	0.940
Random Forest	0.936	0.946	0.926	0.936
SVM	0.898	0.926	0.869	0.896
Naive Bayes	0.874	0.913	0.832	0.871
Bagging	0.945	0.959	0.932	0.946
Boosting	0.935	0.953	0.917	0.934

Table 7 : Results of MinMaxScaler and PCA with various classification models using Test-Train Split

Now, lets see the results of each classifier model with MinMaxScaler and PCA.

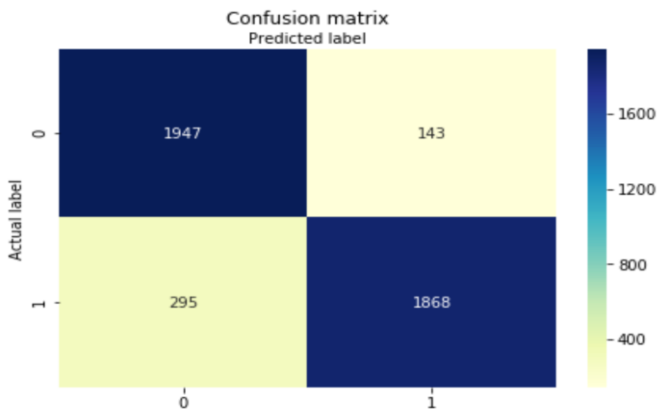


Figure 16 :Confusion matrix for Logistic Regression

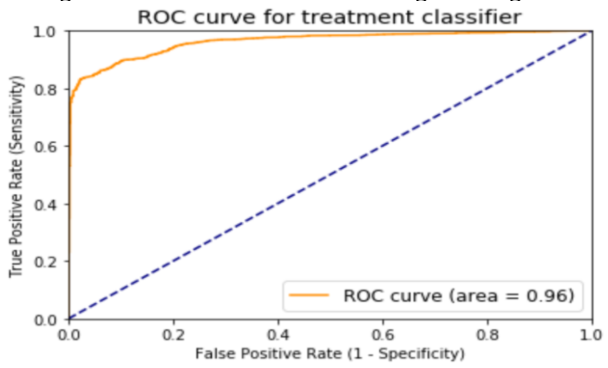


Figure 17 :ROC Curve for Logistic Regression

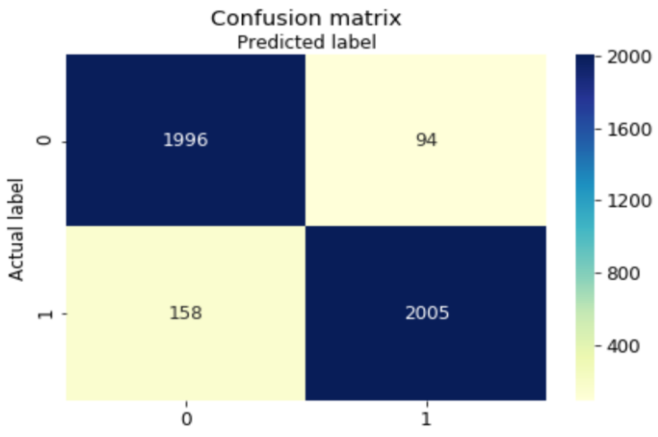


Figure 18 :Confusion matrix for KNN

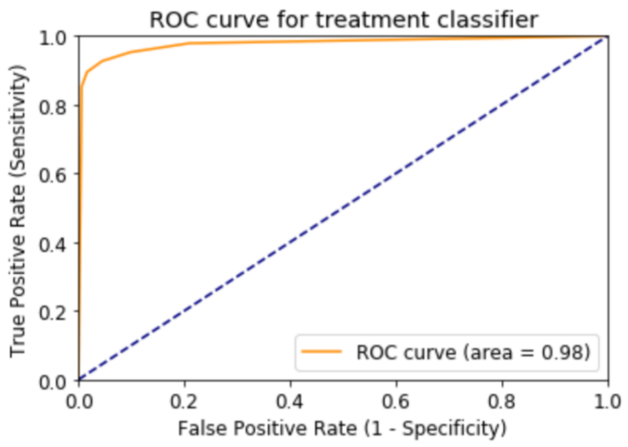


Figure 19 :ROC Curve for KNN

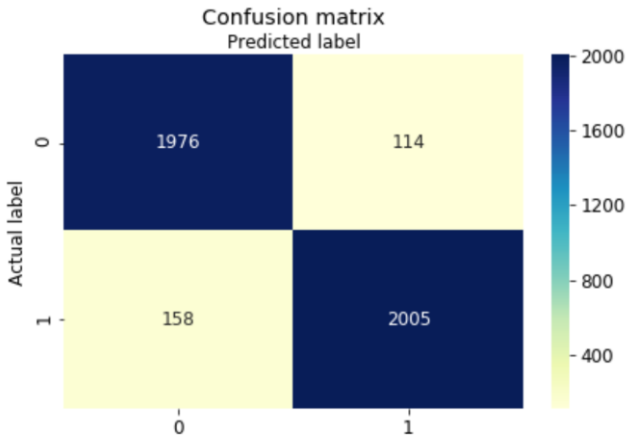


Figure 20 :Confusion matrix for Random Forest

	Test and train split:				K-fold:			
	MAE	RMSE	Variance	CC	MAE	RMSE	Variance	CC
Univariant and linear regression	17.374	22.312	0.571	0.755	17.524	22.484	0.561	0.749
Correlation heatmap and Liner regression	17.442	22.388	0.568	0.753	17.554	22.533	0.559	0.748
Correlation heatmap and K-nearest neighbour	7.498	13.269	0.848	0.921	7.359	13.816	0.8344	0.9144
Correlation heatmap and Gradient-boosting	6.804	11.724	0.881	0.939	6.560	11.099	0.893	0.945
Normalization and Liner regression	0.0010	0.0013	0.729	0.854	0.0010	0.0014	0.7018	0.8385
Normalization and K-nearest neighbour	0.0003	0.0007	0.9187	0.9594	0.00039	0.00081	0.9019	0.9506
Normalization and Decision tree	0.0005	0.0009	0.8609	0.9308	0.00044	0.00082	0.89976	0.9495
Normalization and Gradient-boosting	0.00044	0.00076	0.91459	0.956	0.0004	0.00073	0.920	0.959
MinMaxScaler and Liner regression	5.765	7.789	1.0	0.99	0.065	0.086	0.725	0.851
MinMaxScaler and K-nearest neighbour	0.0217	0.044	0.928	0.963	0.0249	0.0467	0.9199	0.959
MinMaxScaler and Decision tree	0.0217	0.0444	0.9281	0.9639	0.0002	0.0031	0.9996	0.999

Table 6: Results of various combinations of feature extraction and Models.

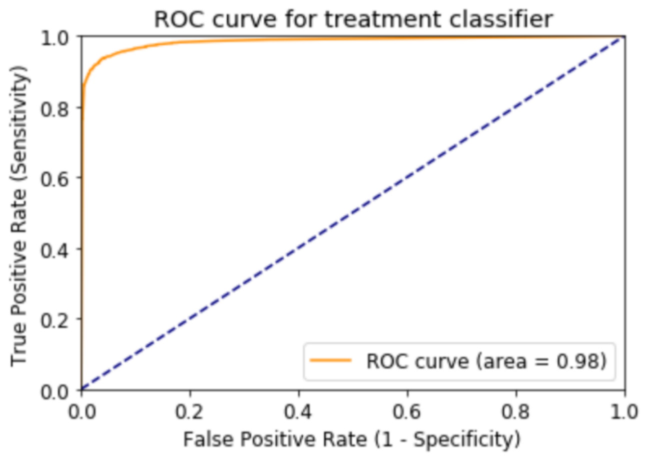


Figure 21: ROC Curve for Random Forest

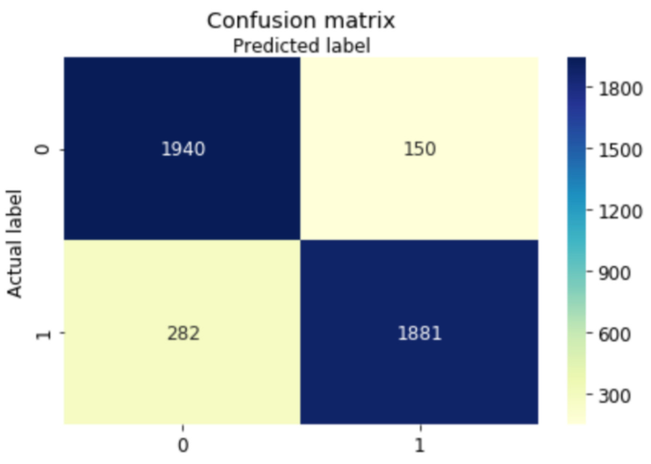


Figure 22 : Confusion matrix for SVM

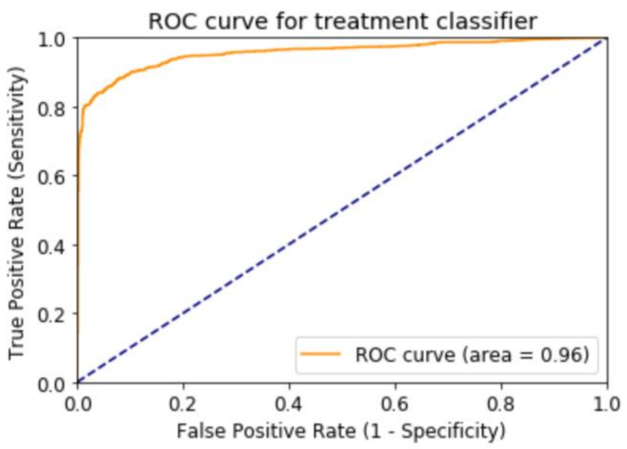


Figure 23:ROC Curve for SVM

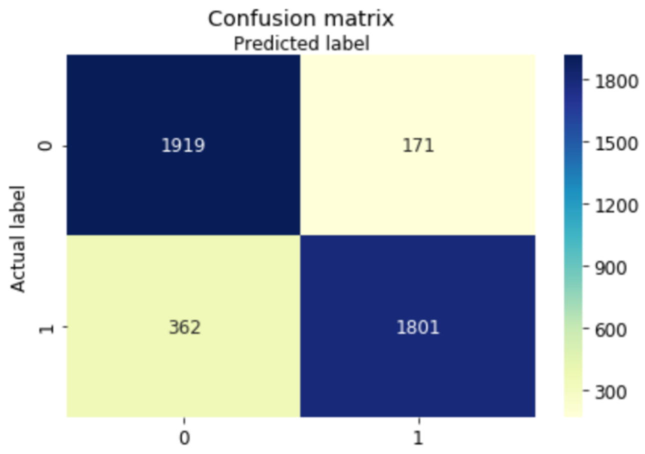


Figure 24 :Confusion matrix for NB

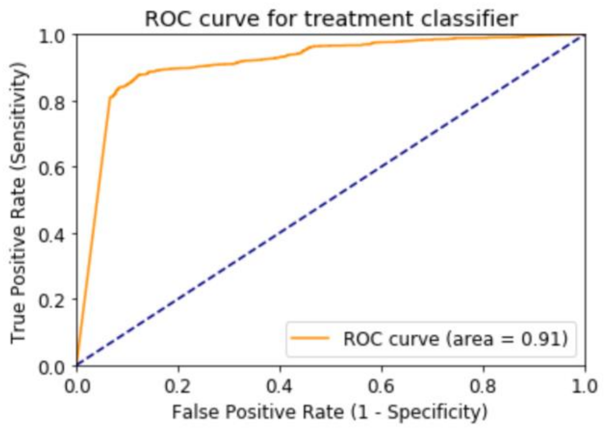


Figure 25:ROC Curve for NB

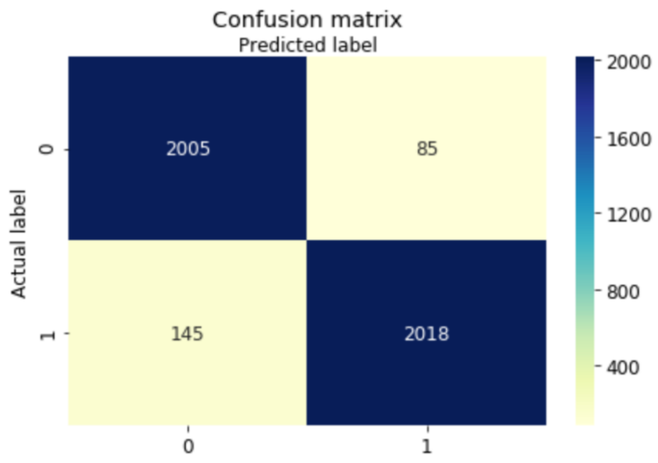


Figure 26 :Confusion matrix for Bagging

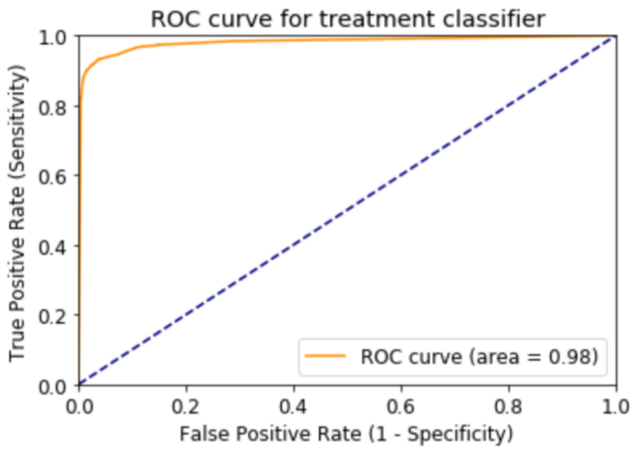


Figure 27:ROC Curve for Random Bagging

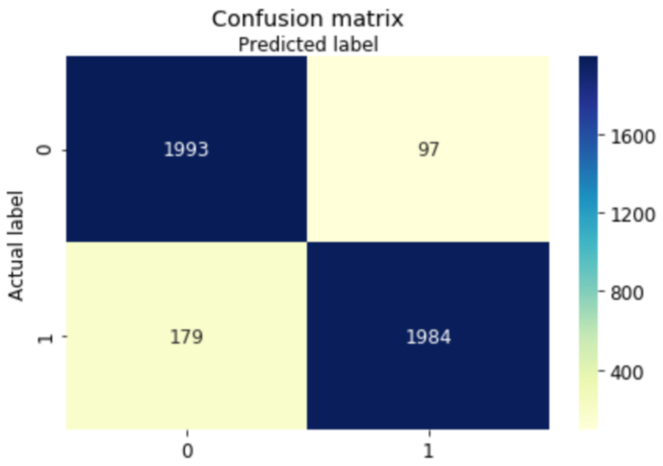


Figure 28 :Confusion matrix for Boosting

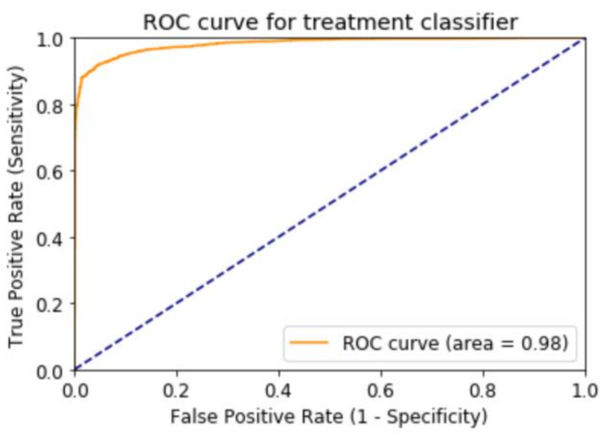
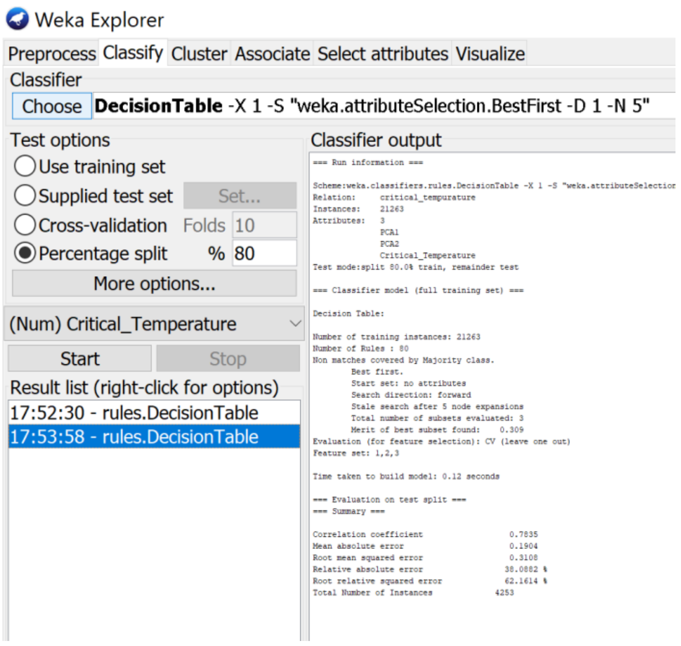


Figure 29:ROC Curve for Boosting



DecisionTable -X 1 -S "weka.attributeSelection.BestFirst -D 1 -N 5"

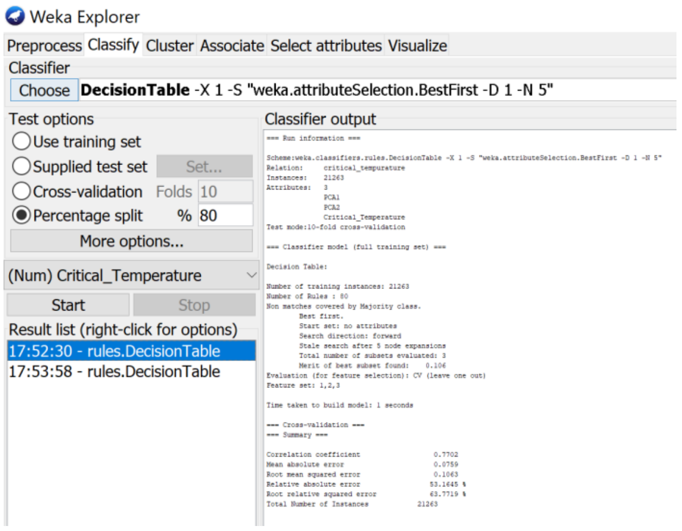
Use training set
Supplied test set
Cross-validation Folds 10
[✓ Percentage split] % 80

(Num) Critical_Temperature

17:52:30 - rules.DecisionTable
17:53:58 - rules.DecisionTable

```
=== Run information ===

Scheme:weka.classifiers.rules.DecisionTable -X 1 -S "weka.attributeSelectior
Relation:     critical_tempurature
Instances:    21263
Attributes:   3
              PCA1
              PCA2
              Critical_Temperature
Test mode:split 80.0% train, remainder test

=== Classifier model (full training set) ===

Decision Table:

Number of training instances: 21263
Number of Rules : 80
Non matches covered by Majority class.
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 3
        Merit of best subset found:    0.309
Evaluation (for feature selection): CV (leave one out)
Feature set: 1,2,3

Time taken to build model: 0.12 seconds

=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.7835
Mean absolute error                      0.1904
Root mean squared error                  0.3108
Relative absolute error                 38.0882 %
Root relative squared error             62.1614 %
Total Number of Instances             4253
```

Figure 30:Weka results for rule based in classification



DecisionTable -X 1 -S "weka.attributeSelection.BestFirst -D 1 -N 5"

Use training set
Supplied test set
Cross-validation Folds 10
[✓ Percentage split] % 80

(Num) Critical_Temperature

17:52:30 - rules.DecisionTable
17:53:58 - rules.DecisionTable

```
=== Run information ===

Scheme:weka.classifiers.rules.DecisionTable -X 1 -S "weka.attributeSelection.BestFirst -D 1 -N 5"
Relation:     critical_tempurature
Instances:    21263
Attributes:   3
              PCA1
              PCA2
              Critical_Temperature
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

Decision Table:

Number of training instances: 21263
Number of Rules : 80
Non matches covered by Majority class.
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 3
        Merit of best subset found:    0.106
Evaluation (for feature selection): CV (leave one out)
Feature set: 1,2,3

Time taken to build model: 1 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.7702
Mean absolute error                      0.0759
Root mean squared error                  0.1063
Relative absolute error                 53.1645 %
Root relative squared error             63.7719 %
Total Number of Instances            21263
```

Figure 31: Weka results for rule based in regressor

D. *Performance Evaluation*

Performance analysis is the one of the important tasks of classification as this analyses the right models for the dataset based on the advisable performance measures for that particular dataset.For this superconductivity dataset, after all the combination of Feature extraction and regression models in Table 6 , we can conclude that there these models have high variance and low bias. We have to consider a tradeoff between the Bias and Variance. Bias occurs when an algorithm has limited flexibility to learn the true signal from the dataset. Bias is the accuracy of our predictions. A high bias implies that the prediction will be inaccurate. If you are highly biased, you are more likely to make wrong assumptions about them. Bias is the same as the mean square error (MSE). Variance is the difference between many model's predictions.

For our dataset, it was found that considering High Variance -Low Bias is creating a lot of overfitting in our dataset. Overfitting occurs when a model learns the noise in the training data to an extent that it negatively impacts the performance of the model on new data. This implies that the noise in the training data is used by the model to learn as concepts. The problem here is that these concepts don't apply to new data and the model lacks the ability to generalize. So, we consider the case of Low variance and high bias, for which the MinMaxScaler combined with PCA was choose and computed with the chosen regression model from table 2.

Weka results for classification and regression are described in Figure 30 and Figure 31 respectively. It can be perceived from figure 8-15, that models Decision tree, Gradient Boosting and KNN have a good fit for our dataset than compared to Linear regression. From the above Figures 16-29, which depict the results of the classifiers and we can distinguish that KNN and Bagging have better accuracy compared to the others.

ROC (Receiver Operating Characteristics**)** is a probability curve and (Area Under the Curve) AUC represents degree or measure of separability. It tells us the extent to which the model is capable of differentiating between classes. Higher the AUC, better the model is at predicting 0s as 0s and 1s as 1s. Higher the AUC, better the model is at differentiating between the classes, in our case it is low and high.

IV. CONCLUSION AND FUTURE WORK

We have successfully learnt and applied most of the techniques learnt in this course in the right path for this project. We have identified the right approach to the superconductivity problem using regression and classification techniques. This helps us predict the critical temperature of the element and classifies into a high or low temperature superconductor. Now these can be used for various applications based on their level of superconductivity. We can conclude form our analysis that normalization using MinMaxScaler was the best feature extraction method for this dataset. We can observe that the RMSE is larger than the MAE. Since the RMSE is squaring the difference between the predictions and the ground truth, any significant difference is made more substantial when it is being squared. There is no significant difference between the regressors- Decision tree, Gradient Boosting and KNN for predicting the critical temperature of the element (this is concluded using the performance measures variance and bias), while the best classifiers are KNN and Bagging for the classification of the level of superconductivity, as they have the highest AUC among all the classifiers used.

Using cluster analysis might divide the level of superconductors into even smaller clusters, making it available to many other applications. Forming the clusters based on the number of elements in attributes, we will be able to judge the superconductor based on the number of elements it contains. This might be a necessary factor to determine the class of superconductors. Performing the statistically analysis of the regression and classification task for this dataset would be challenge.

V. ACKNOWLEDGMENT

The success and the outcome of this project required guidance and assistance from different sources, and we feel extremely fortunate to have got this all along the completion of our project. Whatever we have done is largely due to such guidance and assistance and we would not forget to thank them.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from our teaching assistant who helped us in successfully completing our project work.

I owe my profound gratitude to my professor guide Ms. Herna L Viktor, who took keen interest in my project work and guided me all along in the class lectures and providing the necessary information to proceed with the project.

VI. REFERENCES

[1] https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data

[2] http://docs.arcadiadata.com/4.0.0/pages/topics/visual-correlation.html

[3] https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7

[4] https://www.openml.org/a/estimation-procedures/1

[5] https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/

[6] http://benalexkeen.com/feature-scaling-with-scikit-learn/

[7] Wikipedia