Critical temperature prediction of a Superconducter Using Regression

Priyanka Gopi

Department of Computer Science
University of Central Florida,
Oviedo, Florida
priyankagopi@knights.ucf.edu

Abstract—This study builds a predictive model capable of estimating the critical temperature of a superconductor from experimentally determined physio-chemical properties of the material (input variables): features extracted from the thermal conductivity, atomic radius, valence, electron affinity and atomic mass. Data on superconductors was studied with the goal of predicting the critical temperature at which a material undergoes superconductivity - a regression task. It is crucial to note that our model does not predict whether a material was superconductive, because all data was on superconductive materials. This study was able to get ±10.68 Kelvin RMSE. The data is originally sourced from the UCI Machine learning repository and coded in python 3.6.

Keywords—Linear regression, superconductivity, regression analysis

I. INTRODUCTION

A superconductor is a substance with unlimited conductivity at extremely low temperatures (zero electrical resistance). Essentially, this means that all the electrons and ions will flow freely on the material. The ability of superconductors to produce magnetic levitation at their critical temperature is one of their most intriguing properties. The Meissner Effect is what we refer to it as.

There were attempts to forecast the temperature at which a substance turns superconductive in this investigation. So why does super conductivity depend on temperature? Consider a well-known instance of something that is completely the opposite to comprehend this. A toaster uses electrical resistance to operate. The electrons in a toaster's circuit are compelled to slow down and collide with one another, which results in the buildup of thermal energy. As a result of the temperature being raised by this thermal energy, the toast gets cooked.

There are many materials that can support superconductivity. Each one is a combination of many different elements and has many intrinsic properties. Thus, by creating a table for each super conductor's properties, along with its critical temperature, a regression analysis can be done. The Meissner Effect is what we refer to it as.

The I used two statistical models: A linear regression model which serves as a benchmark model, and the Random Forest model as the main prediction model with improved score.

To find the critical temperature, we need some variables such as atomic mass, first ionization energy, atomic radius, density, electron affinity, fusion heat, thermal conductivity, and valence. Using these variables we calculated the mean, weighted mean, atomic mean, entropy, geometric mean, range, density, etc.

In this project we intend to predict the critical temperature of the superconductors using the Ordinary Least Squares and the Linear Regression.

We used Python, an object-oriented language, for this project. Numerous libraries, including pandas, scikit-learn, NumPy, matplotlib, seaborn, etc., are available for Python. The pandas library, which is needed to retrieve the dataset and obtain the necessary dataset metadata, was imported.

II. METHODOLOGY

The prediction algorithm developed may not provide the best predictions or predict actual critical temperatures but can help us narrow down values closer to them

Steps involved:

1. Data Preparation

Being one of the crucial steps in data analysis, data preparation here involved are removal of columns with very low variance and those having low correlation with response variable. We also dropped duplicate variables as it could mislead the data patterns.

2. Data Exploration

Exploratory data analysis was conducted to visualize and understand the data given. Histograms and heatmaps were plotted to check the pattern of correlation between the variables and response.

3. Filter out irrelevant features

Initially the data shape was 21263 rows by 168 features. Steps like dropping features that varied 5% of less across all rows using Variance Threshold and duplicated features (using. transpose.drop duplicates(keep='first').transpose()). Variance of variables was obtained and the features with lower variance were eliminated as they can be considered as irrelevant due to the nearly similar values that they produce.

4. Selecting the right model fit

In regression prediction models, it is important to find the best model to fit the data decently enough to obtain good testing prediction accuracy. In our case as well, two different models were being used and by comparing the statistical features of both models, one of them is finalized to be the best model for our dataset.

5. Determine feature importance

After selecting the best model here for fitting, features are analyzed and the best feature with highest influence on response values is obtained.

III. ANALYSIS OF THE DATA

Some of the inferences from visualizations are:

The response data (Critical column variable) seems to be highly skewed as supported by the below image.
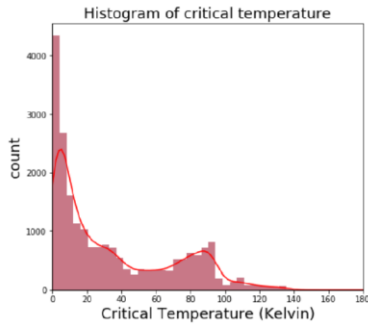

Fig. 1. Histogram of Critical Temperature

Heatmaps were plotted to get a good idea on the correlation between the variables with statistical data like Fie, Atomic mass etc., which helped us understand the mutual dependence between variables. The correlation coefficient between each parameters helps us analyze data better and to better the model prediction by retaining the relevant variables.
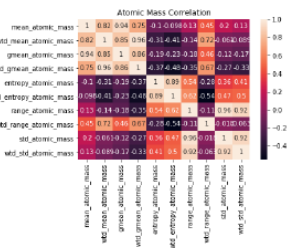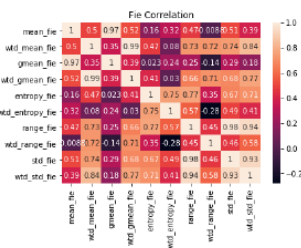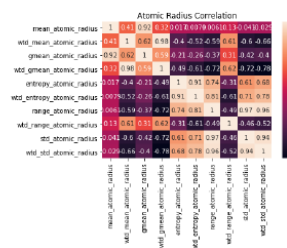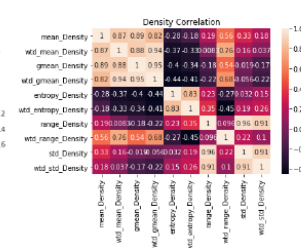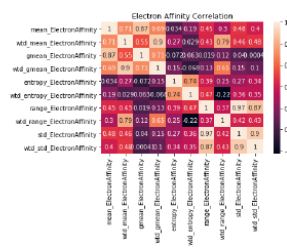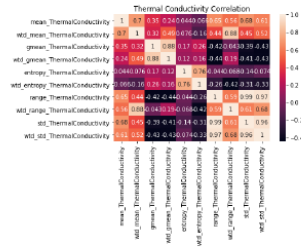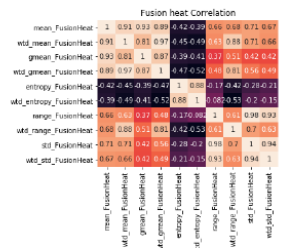
Another way I adopted is to set a threshold for the correlation coefficient and extract or highlight variables with coefficients value above the provided threshold value. We got to see a set of features here below in the image which is a small portion of the table. The whole table image is available in the code file.

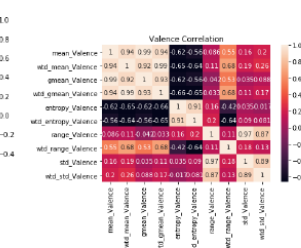|	number_of_elements	wtd_entropy_atomic_mass	range_fie	wtd_entropy_atomic_radius	range_atomic_radius
number_of_elements	1.000000	0.881666	0.780538	0.903869	0.767247
wtd_entropy_atomic_mass	0.881666	1.000000	0.744223	0.961410	0.773918
range_fie	0.780538	0.744223	1.000000	0.797252	0.908514
wtd_entropy_atomic_radius	0.903869	0.961410	0.797252	1.000000	0.812096
range_atomic_radius	0.767247	0.773918	0.908514	0.812096	1.000000
range_ThermalConductivity	0.695240	0.688699	0.682505	0.690006	0.735226
std_ThermalConductivity	0.600972	0.618502	0.640917	0.620366	0.690478
wtd_std_ThermalConductivity	0.664786	0.684302	0.669741	0.675454	0.735364
mean_Valence	-0.608303	-0.583270	-0.740194	-0.622941	-0.758532
wtd_mean_Valence	-0.647462	-0.643826	-0.730255	-0.659561	-0.755878
wtd_gmean_Valence	-0.658208	-0.649781	-0.745692	-0.676740	-0.761020

Fig. 3. Correlation table of variables

Alternatively, a chart representing the correlation between the response and features is as well obtained which clearly indicates the correlation of each feature to the response. Some of the features found relevant here are identified by the values greater than 0.5 like the element O, Cu, Ba, Bi, atomic mass, entropy fie, atomic radius, thermal conductivity, and valence. But this is not the right way to conclude. Further steps have been taken to narrow down the features. This is termed as feature selection.

A. Removal of irrelevant variables

To remove irrelevant features, I dropped features with less than 5% variance and those with duplicated features as these are not going to be of much influence on the response data. Variance less than 5% simply means that there is only 5% chance for the values of those variables to change at various instances. Hence, these variables are of least importance to us for analysis purpose.

```
# remove all columns that have a very small variance

threshold=0.05 #this gets rid of features which are the same value 95% of the time
from sklearn.feature_selection import VarianceThreshold
selector = VarianceThreshold(threshold=threshold).fit(df)
df = pd.DataFrame(selector.transform(df),columns=df.columns[selector.get_support()])
df.shape

(21263, 132)
```

```
df.duplicated().sum()
df.drop_duplicates(inplace=True)
```

Fig. 4. Code snapshot to remove duplicates

IV. DATA MODELLING

A. LINEAR REGRESSION MODEL

Linear Regression is a method for modelling a relationship between a dependent variable and independent variables. Here, we're trying to find the relationship between the response, critical temperature with respect to the predictors like superconductor's atomic radius, density, thermal conductivity, valence and more. These models can be fit with numerous approaches. The most common is *least squares*, where we minimize the mean square error between the predicted values $\hat{y} = X\hat{\beta}$ and actual values.

$$y: (y - \mathbf{X}\beta)^2.$$

Using the Linear regression model, wherein we observed the

intercept and co-efficient outputs (from the LinearRegression() function estimates). Apparently, the R2 score obtained is 0.59 is low in terms of prediction criteria. The evaluation metrics for each one was the R2 score, and the root mean squared error (RMSE) values. The model is rigid.

```
The intercept : 34.44019329985219

The co-efficients are:
[-3.98414184 -2.10738225  2.2278618  -1.41661051  2.50669806  3.30326563
 -0.59923357 -0.66001922 -1.89865311 -0.33795586 -3.26463599 -2.22804297]
The RMSE is : 21.877582693819228
The R2 score: 0.5945268025726214
```

Fig. 5. Code snapshot of linear regression function parameters

B. RANDOM FOREST REGRESSOR MODEL

Next, we used a much flexible Random Forest regressor algorithm that runs works good in case of both regression and classification models. Random forest gave the best results in terms of RMSE and R2 value as it is more flexible than the Linear regression model. Using RMSE, we can easily plot a difference between the estimated and actual values of a parameter of the model.The RMSE helps in clearly distinguishing the efficiency of a model

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\left(Predicted_i - Actual_i\right)^2}{N}}$$

```
#Random Forest regressor

random_forest=RandomForestRegressor(n_estimators=100,random_state=21,verbose=1,max_depth=10)
rfr=random_forest.fit(X_train,Y_train)
RF_predicted=random_forest.predict(X_test)
RF_RMSE=np.sqrt(mean_squared_error(Y_test,RF_predicted))
print("THE RMSE OF RF IS:",RF_RMSE)
print("The R2 score:",r2_score(Y_test,RF_predicted))

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed:  1.2min finished
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done 100 out of 100 | elapsed:   0.1s finished

THE RMSE OF RF IS: 10.689705341949153
The R2 score: 0.9031956324340468
```

Fig. 6. Random Forest regressor model R2 and RMSE values

V. IMPORTANT FEATURE SELECTION

The random forest package's automatic feature importance analysis is one of its more useful features. Here, we can see how each measure stacks up in terms of importance.
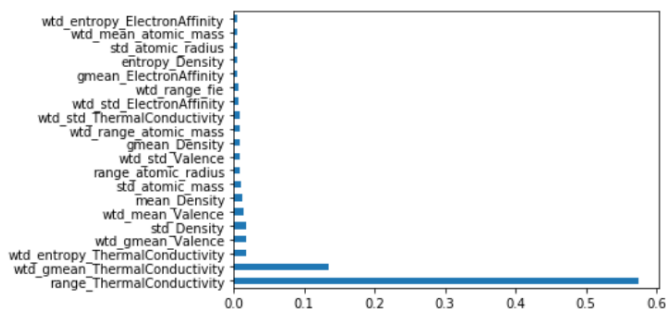

Fig. 7. Important feature selection obtained from Random Forest regressor

VI. OBTAINING A GOOD FITTING MODEL

R2 score and RMSE are not the only statistics to analyse and conclude in regression analysis. Examining the plot of the predicted values versus actual values is important. The plot of the predicted temperatures versus actual temperature is below. It gives us a good idea that there is a linear relationship between the dependent and independent variables.
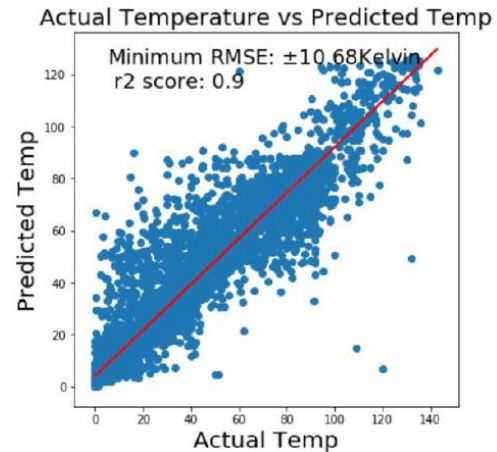

Fig. 8. Scatter plot for actual response by predicted response values

CONCLUSIONS

In the graph above we can see that the important features here is the Thermal Conductivity along with Valence, density, and atomic mass. Which indicates that these parameters play a huge role in predicting response correctly. We got a RMSE of 10.68 Kelvin, which is one of the better results with respect to the given dataset.

ACKNOWLEDGEMENT

I thank Dr. Rui Xie for the support and incorporating this project in the course syllabus.

REFERENCES
[1] Paper 'A data-driven statistical model for predicting the critical temperature of a superconductor'.
[2] Hamidieh, Kam, A data-driven statistical model for predicting the critical temperature of a superconductor, Computational Materials Science, Volume 154, Nov 2018.