Machine Learning

Course Project Report

(Draft-04, Team No- 2)

Title of the project: Superconductivity Data

Student 1: Kavin Fidel, kavin.j-24@sas.saiuniversity.edu.in
Student 2: Ishaan Reddy, ishaan.r-25@scds.saiuniversity.edu.in

ML Category: Regression

---

1. Introduction

This research paper encloses the detailing of a regression model using Machine Learning concepts on the superconductivity dataset that was taken from the UCI machine learning repository. The objective is to build a robust ML model using regression to make accurate predictions about the critical temperature of the superconductor.

This report contains exploratory data analysis on specific important features. Then we use feature selection to filter out the 50 best features that will help us build a robust linear regression model. We also find the value of R^2(Measure of the goodness of the regression line with respect to the data) to predict the accuracy of our model and also use the cross validation technique to evaluate its performance.

2. Dataset and Features

The superconductivity dataset contains 81 features and each of them has 21263 rows of data. In total there are 17,22,303. Columns such as critical_temp, wtd_gmean_density, gmean_density, wtd_mean_density and mean_density are some of the best features in the dataframe.

Exploratory Data Analysis:

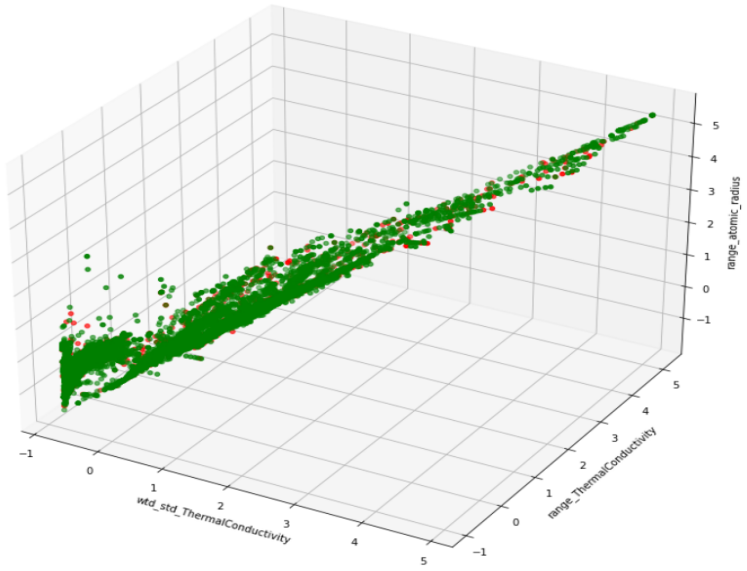
Dataset description using df.describe() function

	number_of_elements	mean_atomic_mass	wtd_mean_atomic_mass	gmean_atomic_mass	wtd_gmean_atomic_mass	entropy_atomic_mass	v
count	21263.000000	21263.000000	21263.000000	21263.000000	21263.000000	21263.000000	
mean	4.115224	87.557631	72.988310	71.290627	58.539916	1.165608	
std	1.439295	29.676497	33.490406	31.030272	36.651067	0.364930	
min	1.000000	6.941000	6.423452	5.320573	1.960849	0.000000	
25%	3.000000	72.458076	52.143839	58.041225	35.248990	0.966676	
50%	4.000000	84.922750	60.696571	66.361592	39.918385	1.199541	
75%	5.000000	100.404410	86.103540	78.116681	73.113234	1.444537	
max	9.000000	208.980400	208.980400	208.980400	208.980400	1.983797	

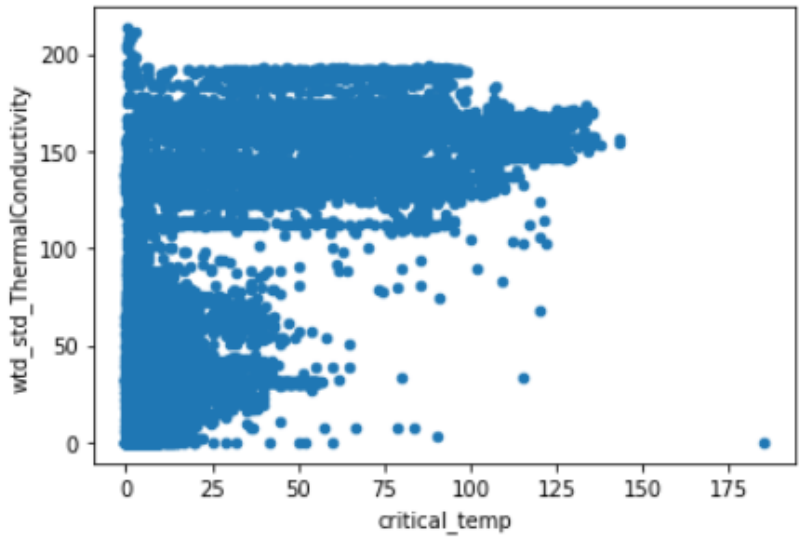
Correlation matrix

	number_of_elements	mean_atomic_mass	wtd_mean_atomic_mass	gmean_atomic_mass	wtd_gmean_atomic_mass
number_of_elements	1.000000	-0.141923	-0.353064	-0.292969	-0.454525
mean_atomic_mass	-0.141923	1.000000	0.815977	0.940298	0.745841
wtd_mean_atomic_mass	-0.353064	0.815977	1.000000	0.848242	0.964085
gmean_atomic_mass	-0.292969	0.940298	0.848242	1.000000	0.856975
wtd_gmean_atomic_mass	-0.454525	0.745841	0.964085	0.856975	1.000000
...	...	...	...	...	...
range_Valence	0.231874	-0.107450	-0.039155	-0.165010	-0.078641
wtd_range_Valence	-0.447770	0.168633	0.330904	0.272303	0.409674
std_Valence	0.105365	-0.080279	-0.003681	-0.124627	-0.033313
wtd_std_Valence	0.035216	-0.081253	0.077323	-0.117336	0.030361
critical_temp	0.601069	-0.113523	-0.312272	-0.230345	-0.369858

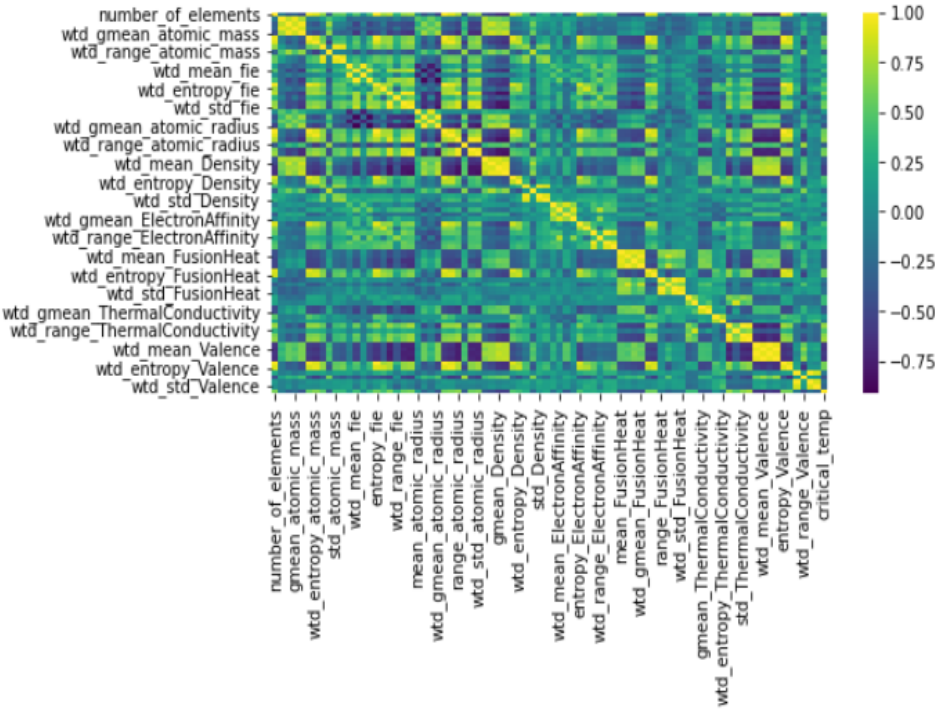
wtd_std_ThermalConductivity Vs **range_ThermalConductivity** Vs **range_atomic_radius**



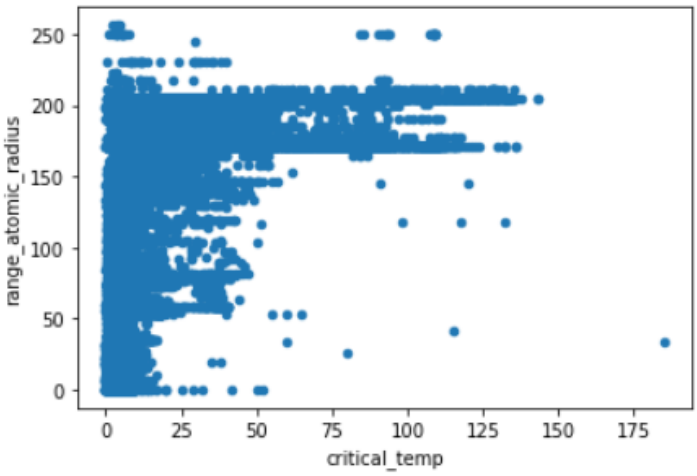
critical_temp Vs **wtd_std_ThermalConductivity**



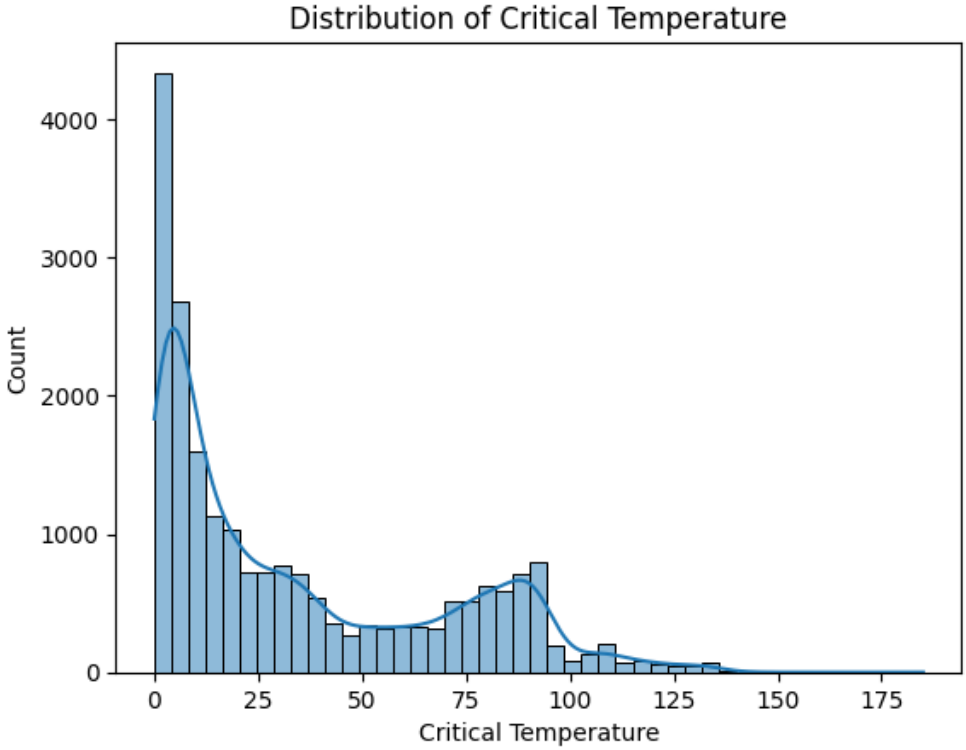
Correlation matrix heatmap

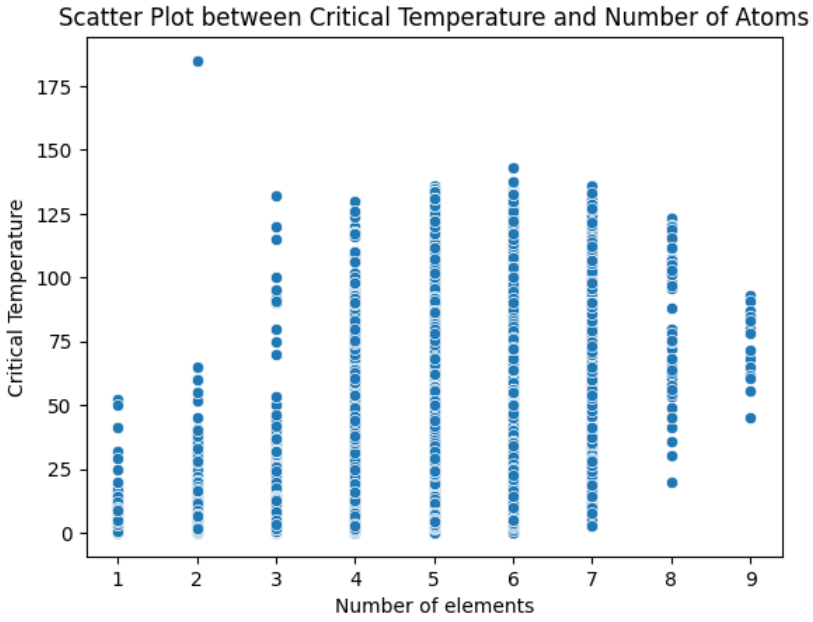


critical_temp Vs range_atomic_radius

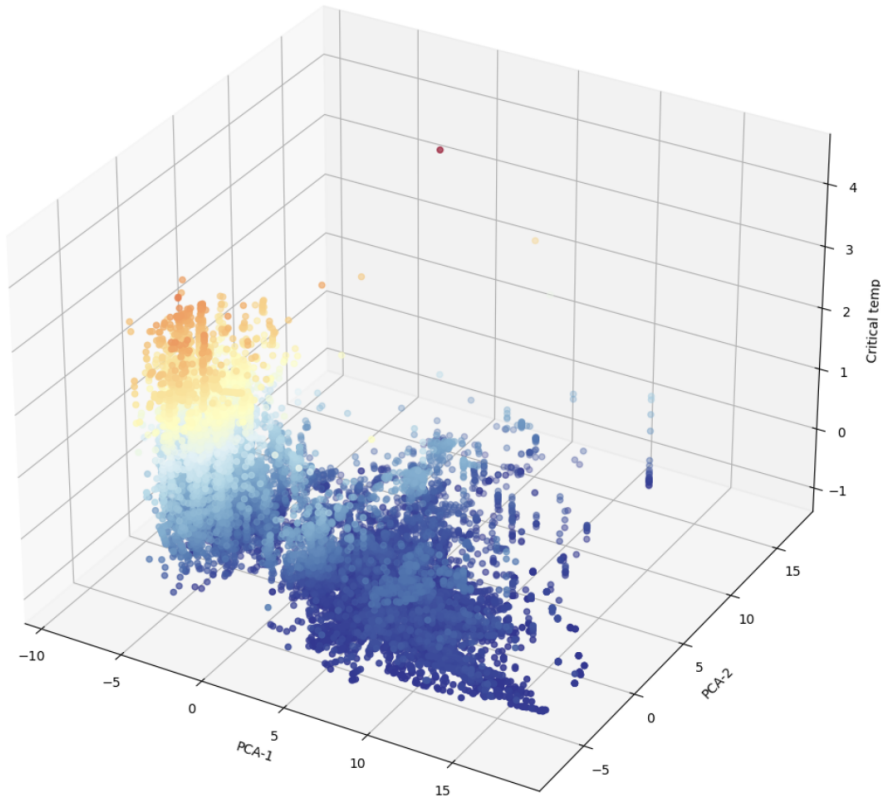


Distribution of Critical Temperature in the dataset:





Principal Component Analysis:



(Visual Representation using 2D-PCA)

3. Methods

3.1 Baseline - Linear Regression

Linear Regression in simple terms essentially means to predict the value of a variable based on the value of another variable(s). The variable that is to be predicted is called the target variable or the dependent variable. The other variables are called independent variables. In the project the “Critical Temperature” is the target variable, which is to be predicted based on the other independent variables present in the data. This type of regression is called Multivariate Linear Regression since it has more than at least 2 independent variables.

3.2 Polynomial regression

We can find the difference between the actual value and the best fitting line we predicted. Polynomial Regression predicts the best-fit line that matches the pattern of the data.

$$y = b_0 + b_1 x_1 + b_2 {x_1}^2 + \ldots + b_n {x_1}^n$$

3.3 Regularization

In order to avoid overfitting of the model over the training data we use regularization models by reducing the number of polynomial degrees. This is done by adding a term(penalty) to the cost function.

LASSO Regression(L1):

Least absolute shrinkage and selection operator regression(LASSO) is one of the regularized versions of linear regression. In LASSO regression the regularization term is the L1 norm of the weight vector. The L1 norm is the sum of the absolute values of the vector.

Here α is the hyperparameter that controls the regularization of the model.

$$||w||_1 = |w_1| + |w_2| + \ldots + |w_n|$$

(L1-norm)

$$J(\ominus) = MSE(\ominus) + 2\alpha \sum |(w)_i|$$

Ridge Regression(L2):

The Ridge Regression uses the L2 norm as the regularization term. The L2 norm is the sum of the squares of the absolute values of a vector.

$$||w||_2 = |{w_1}^2| + |{w_2}^2| + \ldots + |{w_n}^2|$$

(L2-Norm)

$$J(\ominus) = MSE(\ominus) + \alpha/m \sum |(w)^2_i|$$

Elastic Net Regression:

Elastic Net Regression uses a regularization term that is a weighted sum of both ridge and lasso regularization terms along with a "r" factor that controls the mix ratio.

$$J(\ominus) = MSE(\ominus) + 2\alpha \sum|(w)_i| + (1 - r)\,(\alpha/m)\sum|(w)^2_i|$$

3.4 Support Vector Machine Regression

The use of support vectors to draw a straight line(Hyper-Plane) for regression is called Support Vector Machine Regression.

Linear SVM Regression:

Linear SVM Regression uses the linear kernel function to the hyperplane that best fits the data.

Kernel SVM Regression

In kernel SVM regression the input data is transformed into a higher dimensional space using the kernel function. This helps in capturing the non-linear relationship between the input features and the target variable("Critical temperature"). The radial functions could be linear,polynomial or radial basis function kernels.

The Results of the SVM regression models are in section 4.

3.5 Decision Trees:

Decision Tree Regressor:

A decision tree regressor is a type of machine learning algorithm used for regression tasks. It is based on the concept of a decision tree, which is a flowchart-like structure where internal nodes represent feature tests, branches represent possible feature outcomes, and leaf nodes represent the predicted target values.

CART Algorithm:

CART(Classification and Regression Trees) is the underlying algorithm used to construct decision trees. It is a greedy algorithm., it recursively partitions the input data by selecting the best feature and split point at each step. In the context of regression, the algorithm aims to minimize the mean squared error (MSE) as the impurity measure.

3.6 Ensemble Methods:

Random Forest Regressor:

The Random Forest Regressor builds a collection of decision trees, where each tree is trained on a random subset of the training data with replacement (bootstrap sampling), and at each node, only a random subset of features is considered for splitting. This randomization helps to introduce diversity among the individual trees in the forest.

AdaBoost Regressor:

AdaBoost Regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction. As such, subsequent regressors focus more on difficult cases.

GradientBoost Regressor:

GradientBoost Regressor is a machine learning algorithm used for both classification and regression problems. It works on the principle that many weak learners can together make a more accurate predictor. Gradient Boosting Regression is an analytical technique that is designed to explore the relationship between two or more variables (X, and Y). Its analytical output identifies important factors (X i) impacting the dependent variable (y) and the nature of the relationship between each of these factors and the dependent variable.

Hyperparameter Tuning:

In Hyperparameter Tuning, the hyperparameters cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn. Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem. The two best strategies for Hyperparameter tuning are:

- GridSearchCV- In GridSearchCV approach, the machine learning model is evaluated for a range of hyperparameter values. This approach is called GridSearchCV, because it searches for the best set of hyperparameters from a grid of hyperparameters values.
- RandomizedSearchCV- RandomizedSearchCV solves the drawbacks of GridSearchCV, as it goes through only a fixed number of hyperparameter settings. It moves within the grid in a random fashion to find the best set of hyperparameters. This approach reduces unnecessary computation.

4. Experiments & Results

4.1 Protocol

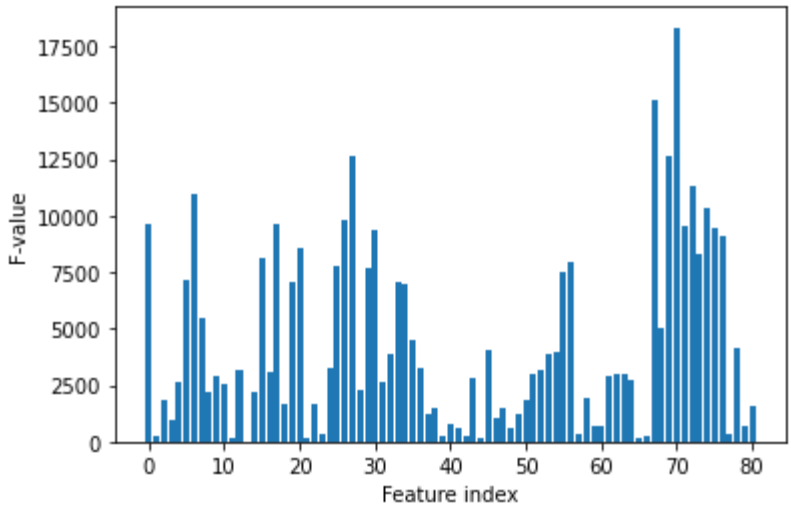
Pre-Processing:

The dataset underwent 2 Preprocessing methods:

- Feature Selection
- Feature Scaling

Feature Selection:

- Feature selection is a process that chooses a subset of features from the original dataset so that in effect the feature space is optimally reduced.
- For this dataframe the Chi-square method was used to select the features.
- Using the feature selection method, the top 50 most relevant features were copied into the ‘dataframe’.



(A plot of F-value vs Feature index)

Feature Scaling:

In order to normalize the range of features in the dataset we use what is called "Feature Scaling". In this model, we make use of Feature standardization.

Feature standardization works by determining the mean and standard deviation for each feature and calculating the new data point by the formula.

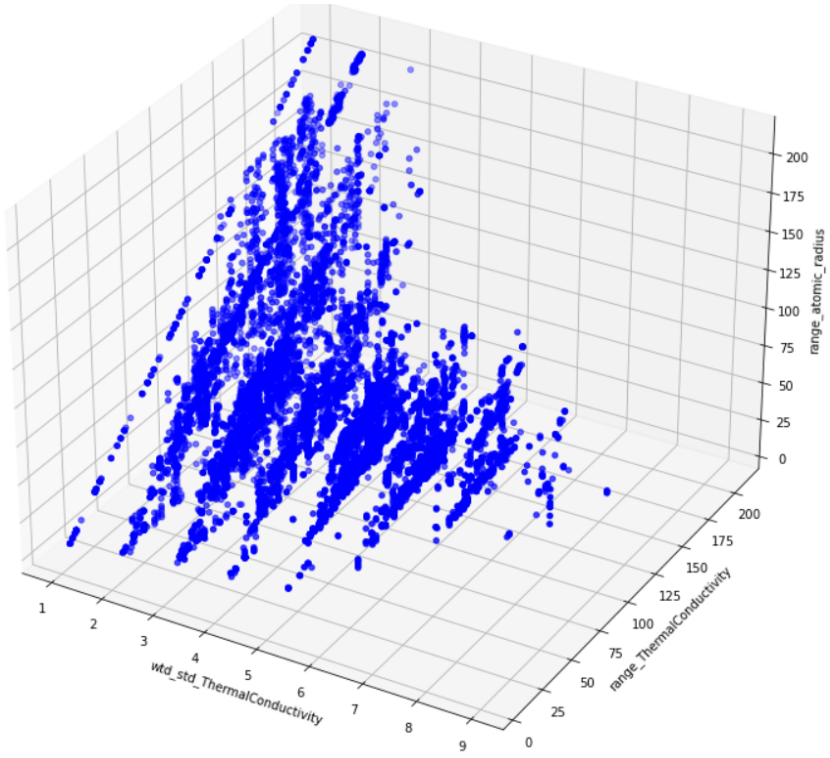
$$\tilde{x} = (x - mean)/S.D$$

$S.D$: Standard Deviation

Dataset before standardization:

	wtd_gmean_Density	gmean_Density	wtd_mean_Density	mean_Density	wtd_range_Density	range_Density
0	53.543811	724.953211	2961.502286	4654.35725	1579.583429	8958.571
1	54.095718	1237.095080	3021.016571	5821.48580	1667.383429	10488.571
2	53.974022	724.953211	2999.159429	4654.35725	1667.383429	8958.571
3	53.758486	724.953211	2980.330857	4654.35725	1623.483429	8958.571
4	53.117029	724.953211	2923.845143	4654.35725	1491.783429	8958.571
...	...	...	...	...	...	...
21258	4082.735787	6404.741690	4963.928889	7341.25000	2449.715556	7511.000
21259	66.286408	962.364248	2827.415190	5174.28580	1705.918143	11848.571
21260	9170.377777	10150.719680	9260.600000	10296.50000	4451.400000	3453.000
21261	9518.329826	10150.719680	9640.430000	10296.50000	2186.170000	3453.000
21262	6830.731801	6186.508901	6914.900000	6311.00000	3455.100000	3055.000

21263 rows × 51 columns

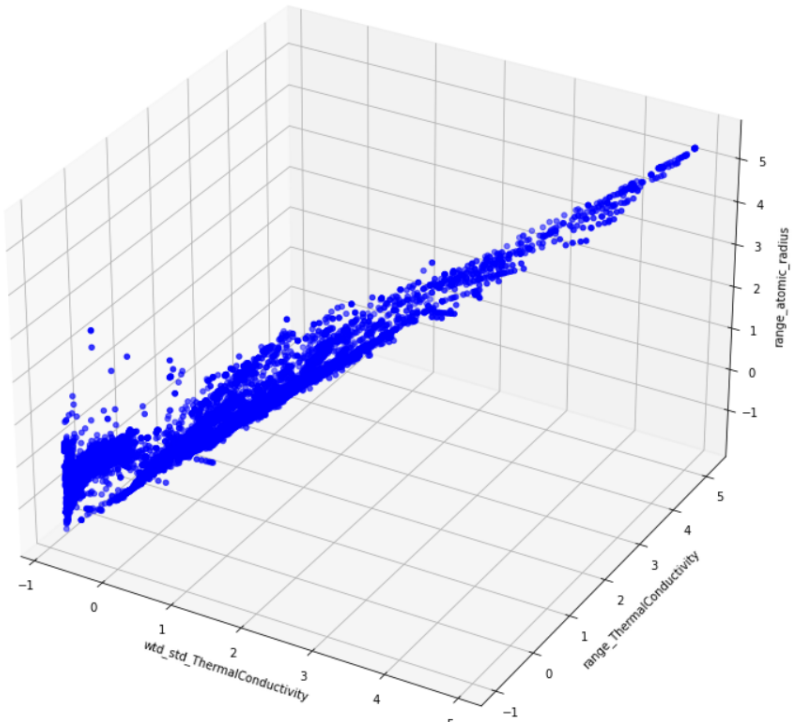


(A scatterplot of 3 variables before standardization)

Dataset after Standardization:

	34	33	32	31	38	37	17
0	-0.770736	-0.738756	-0.715776	-0.511855	-0.551678	0.071547	0.769935
1	-0.770597	-0.600458	-0.697301	-0.101864	-0.515071	0.444989	0.769935
2	-0.770628	-0.738756	-0.704086	-0.511855	-0.515071	0.071547	0.769935
3	-0.770682	-0.738756	-0.709931	-0.511855	-0.533375	0.071547	0.769935
4	-0.770843	-0.738756	-0.727467	-0.511855	-0.588286	0.071547	0.769935
...	...	...	...	...	...	...	...
21258	0.242890	0.795008	-0.094144	0.432001	-0.188884	-0.281775	-0.966459
21259	-0.767530	-0.674646	-0.757402	-0.329214	-0.499004	0.776936	0.769935
21260	1.522791	1.806568	1.239713	1.470126	0.645703	-1.272248	-1.396359
21261	1.610325	1.806568	1.357627	1.470126	-0.298767	-1.272248	-1.396359
21262	0.934205	0.736077	0.511515	0.070093	0.230303	-1.369392	-1.262641

21263 rows × 51 columns



(Scatter plot of 3 variables after standardization)

Splitting the dataset:

After the features in the dataset were filtered and scaled, the dataset was split into a training set and testing set with the 80-20 percent proportion respectively. The random_state parameter was set to 42 to preserve the randomness of the split.

4.2 Results

Linear regression:

A Linear regression model is created followed by the splitting of the dataset. After creating the model, the R^2 value is calculated using the testing dataset.

What is R^2 value?

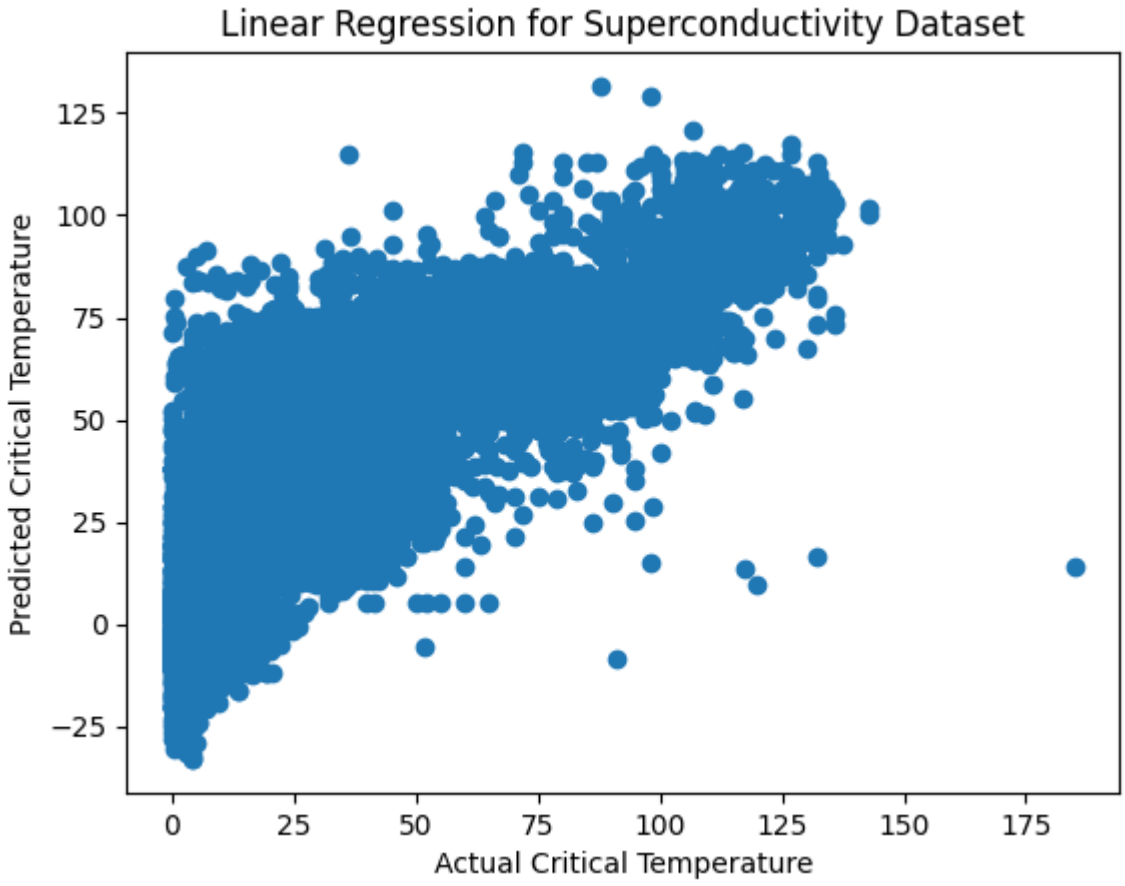
R squared value is also called the coefficient of determination. The goodness of fit of a regression model is judged based on the R squared method. When the value of R^2 is close to 1, the model is said to be perfect.

$R^2 = 1 - (SSres/SStot)$

SSres is the residual sum of squares.

SStot is the total sum of squares.

In this regression model the R^2 value is 0.738.



Cross-Validation:

Cross-Validation is a method used to estimate the accuracy of the regression model. It helps in highlighting whether the predicted model is overfitting the training data. In short, through cross validation we create k number of folds or partitions in the data and run the prediction on each fold and average out the estimate.

The mean and the standard deviation acquired through the cross validation is reported as:
$0.145 + / - 0.525$.

Polynomial Regression:

Polynomial Regression helps us to find the relationship between the independent variable x and the dependent variable y. It is described as the nth degree polynomial in x.

In this regression model the R^2value is 0.609.

L1 / LASSO Regression:

The R^2 value of L1 is: 0.7265134750131724

L2 / Ridge Regression:

The R^2 value of L2 is : 0.7376647316768912

Elastic Net Regression:

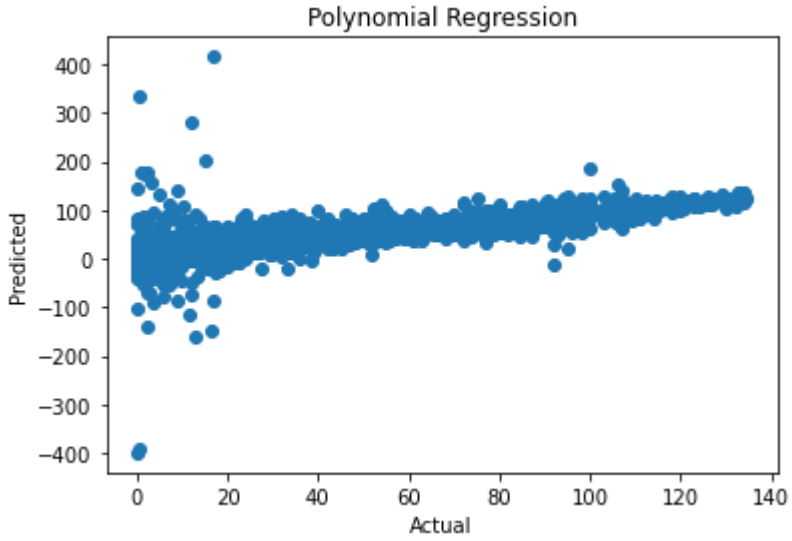
The R^2 value of Elastic Net is: 0.7237062288046392

Linear SVM Regression:

The R^2 value of Linear SVM Regression is: 0.7210594697655597

Polynomial SVM Regression:

The R^2 value of Polynomial SVM Regression is: 0.6771535258322248

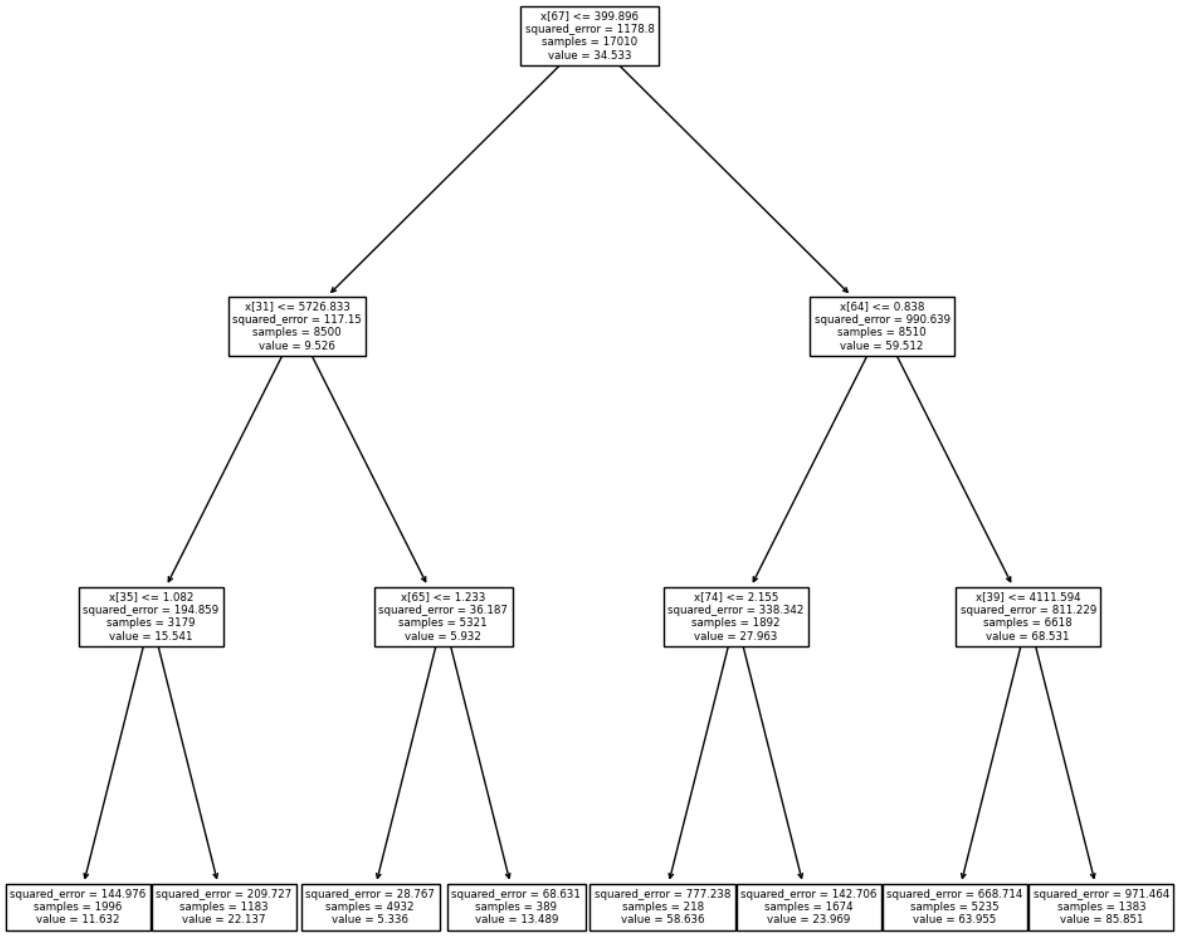


RBF SVM Regression:

The R^2 value of RBF SVM Regression is: 0.7840798753053251

Decision Tree Regressor:

The decision tree regressor yielded a score value of 0.985.



(Visualizing the decision tree)

Random Forest Regressor:

The random forest regressor yielded a score value of 0.814

NOTE: The above 2 methods(Decision Trees, Random Forest regressor) do not require feature scaling.

AdaBoost Regression:

The AdaBoost regression yielded a score value of 0.728.

GradientBoost Regression:

The GradientBoost regression yielded a score value of 0.868.

Hyperparameter Tuning:

- GridSearchCV- This model yielded a score value of 0.758.
- RandomizedSearchCV- This model yielded a score value of 0.758.

7. References

- Hamidieh, Kam, A data-driven statistical model for predicting the critical temperature of a superconductor, Computational Materials Science, Volume 154, November 2018, Pages 346-354
- Serafeim Loukas, P.D. (2023) *How to perform feature selection for regression problems*, *Medium*. MLearning.ai. Available at: https://medium.com/mlearning-ai/how-to-perform-feature-selection-for-regression-problems-cc6ea56c6d48 (Accessed: April 20, 2023).
- Roy, B. (2023) *All about feature scaling*, *Medium*. Towards Data Science. Available at: https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35 (Accessed: April 20, 2023).
- S, Premanand. (2021, October 29). Understanding Polynomial Regression Model. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/10/understanding-polynomial-regression-model/
- "Sklearn.Ensemble.AdaBoostRegressor." *Scikit-learn*, scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html
- Dash, Shruti. "Gradient Boosting – a Concise Introduction From Scratch." *Machine Learning Plus*, Apr. 2022, www.machinelearningplus.com/machine-learning/gradient-boosting.
- GeeksforGeeks. "Hyperparameter Tuning." *GeeksforGeeks*, Aug. 2022, www.geeksforgeeks.org/hyperparameter-tuning.