A FINE-GRAINED ANALYSIS OF XGBOOST PREDICTING T_c

DANIEL BRISENO

1. Introduction

Superconductors have become increasingly relevant to the development of new technology. Well known applications of superconductors include Magnetic Resonance Imaging (MRI), the Large Hadron Collider at CERN, and proposed superconducing cables which may be able to deliver electricity with no energy loss [3].

Another important application of superconductors is the implementation of quantum computers. Quantum computers have recently been shown to posses speed advantages over classical computers, and a scalable implementation of a quantum computer could revolutionize the fields of computer science, computer engineering, and all applications of these disciplines. Among the most promising implementations of quantum computers are those which utilize superconductors to construct their quantum bits (qubits). Notably, the recent demonstration of quantum supremacy by Google used superconducting qubits to achieve their result [1].

Despite the recent demonstration by Google and the best efforts of a large and active research community, a scalable implementation of a quantum computer (and of superconducting technology in general) is restricted by the fact that superconductors have to be cooled to their critical temperature (T_c) before they exhibit their superconducting properties. The T_c of a superconductor is often below 77°K, the boiling temperature of nitrogen[3]. Thus, cooling a system with more than a few hundred cubits is prohibitively expensive. For a sense of scale, Google's celebrated implementation of a quantum computer has 54 qubits, while a typical laptop has an excess of 64 billion bits of RAM.

To this end, the discovery of a superconductor with a high T_c has become an important area of materials science. However finding a physical model of conductivity which can predict the T_c of an arbitrary superconductor remains an unsolved problem, with the most promising approaches relying on heuristics which are often violated in unpredictable ways [3]. Without a theory-based model to predict T_c, researchers are turning to Machine Learning algorithms to predict the critical temperature of a material given its atomic properties.

In this report, I will be analyzing the performance of the XGBoost gradient-boosted decision tree algorithm [2] applied to predicting T_c of an arbitrary compound. More specifically, I will be analyzing the performance of XGBoost with hyper-parameters specified by Dr. K. Hamidieh in [3].

Previous attempts at predicting T_c have trained on and predicted T_c for Iron-based, Cuprate, low T_c, or MgB_2 based superconductors [4][5]. Dr. Hamidieh's approach extends T_c prediction to the general class of superconductors (or materials suspected of

being superconductors). In his paper, Dr. Hamidieh reports that a properly optimized XGBoost model can predict critical temperature of an arbitrary superconductor with an out-of-sample RMSE of approximately 9.5°K.

The data used to train and test this model is the Superconductivity Dataset found in the UCI Machine Learning Repository[6]. It is a collection of 21263 superconductors with 82 features defined for each superconductor. The features of the superconductors are generated from the atomic properties of the elements in the superconductor summarized in Table 1 (this table can also be found in [3]).

Variable	Units	Description
Atomic Mass	Atomic mass units (AMU)	Total proton and neutron rest masses
First Ionization Energy	Kilo-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 coefficient κ
Valence	No units	Typical number of chemical bonds formed by the element

TABLE 1. Elemental properties used to define features used by XGboost to predict T_c.

For a given superconductor in the data, the atomic properties of the elements which make up that superconductor are combined according to the summary statistics listed below:

(1) Mean
(2) Weighted mean
(3) Geometric mean
(4) Weighted geometric mean
(5) Entropy
(6) Weighted entropy
(7) Range
(8) Weighted range
(9) Standard deviation
(10) Weighted standard deviation

A more in-depth explanation of these summary statistics and their associated formulas can be found in Table 2 of [3]. Apart from these features, the number of elements making up the compound and the T_c of the compound are listed in the data, bringing the total number of features per compound to 82.

While Dr. Hamidieh does specify that XGBoost predicts T_c with an out-of-sample RMSE of 9.5°K, I will be presenting a finer grained analysis of the performance of XGBoost. More specifically, in the following sections I will attempt to answer:

(1) How well does XGBoost preform at predicting T_c of Iron, Cuprate, Mercury and MgB_2 based super conductors? Does this performance improve or worsen when trained only on Iron, Cuprate, or MgB_2 based superconductors?

(2) If we divide the testing data by quartiles of T_c, how well does XGBoost preform at predicting T_c of compounds in each quartile? Is XGBoost reliable when predicting the T_c of a compound with a high T_c?

(3) If we divide the predictions of XGBoost by quartiles of predicted T_c, how accurate and precise is XGBoost's prediction when the prediction falls in a given quartile? Is the predicted value of T_c reliable when this predicted value is high?

The exact metrics used to determine when XGBoost preforms "well" will be elaborated on in the following section.

2. Methods

In this section I will describe the data collection process by first specifying the error statistics collected for predicted T_c values, then describing how these error statistics were collected for any relevant subset of the training and testing data, then finally describing the subsets themselves.

2.1. Error Statistics.

Let P be the predicted value of T_c and A be the actual value. Then I define the **raw error** err as $err := P - A$. Note that a negative raw error indicates an under-prediction, and a positive raw error indicates an over-prediction.

This investigation consisted of collecting summary raw error statistics for T_c predictions by XGBoost on relevant subsets of the testing and training data. The error statistics collected are summarized in Table 2. In the following sections I will call a list of values containing all error statistics in Table 2 an **error vector**.

2.2. Data Collection.

There were essentially two types of analysis done which required slightly different approaches to data collection:

i) Collecting error statistics for predicted T_c of compounds with specific actual attributes.

ii) Collecting error statistics for predicted T_c of compounds in different predicted T_c quartiles.

It is important to remark on the difference between these two types of analysis. The first focuses analysis on subsets of the training data which are defined by *true* values of the superconductors. For example, one attribute used is whether the compound

Statistic	Description
RMSE	The residual mean squared error of predictions
ave_err	Raw average of error
std_err	Standard deviation of raw error values
under_cnt	Number of under-predictions
over_cnt	Number of over-predictions
correct_cnt	Number of exactly correct predictions. Expected to be 0
ave_under	Average value of under-prediction raw error
ave_over	Average value of over-prediction raw error
std_under	Standard deviation of under-prediction raw error
std_over	Standard deviation of over-prediction raw error

TABLE 2. Summary error statistics collected for predicted T_c values.

contains Iron. The second focuses analysis on subsets of the testing data which are defined by the *predicted* value of T_c for a superconductor.

When doing analysis of type i), collecting a single error vector consisted of:

(1) Intersecting the testing and training data with the subset of superconductors under consideration.
    - This creates training and testing subsets consisting only of superconductors in the desired subset.
(2) If necessary, retraining the model on the new subsetted training data.
    - If no retraining is done, then the default model remains in place. The default model is trained on the whole training data partition before subsetting.
(3) Using the XGBoost model to predict T_c values for superconductors in the subsetted testing data.
(4) Subtracting the actual T_c values from the predicted T_c values to obtain a vector of raw errors
(5) Taking the summary statistics in Table 2 of the raw errors to obtain an error vector.

This process was carried out 50 times, each time randomly partitioning the superconductor data into 2/3 training data and 1/3 testing data – and training the XGBoost model on the testing data accordingly. At the end of the 50 trials, this process yielded a matrix of 50 error vectors.

For analysis of type ii), collecting a single error vector consisted of:

(1) Using the XGBoost model to obtain a prediction vector for T_c values of the testing data partition.

(2) Separating prediction vector into predicted T_c quartiles.

(3) For each prediction quartile, subtracting the actual T_c values from the predicted T_c values to obtain a vector of raw errors.

(4) For each vector of raw errors, taking summary statistics in Table 2 to obtain an error vector.

(5) For each error vector, adding a field to the error vector specifying the prediction quartile for which the summary statistics were taken.

This process was carried out 50 times, again, each time randomly partitioning the superconductor data into 2/3 training data and 1/3 testing data, and training the model accordingly. At the end of the 50 trials, this process yielded a matrix of 200 error vectors, 50 vectors per quartile.

2.3. **Subsets Studied.** Previously, I introduced three questions concerning the performance of XGBoost across three categories: superconductors containing certain elements, superconductors in different true T_c quartiles, and superconductors in different predicted T_c quartiles.

To this end, there are three classes of subsets under consideration: the element subsets, the true T_c quartiles, and the predicted T_c quartiles.

2.3.1. *The Element Subsets.* There were four subsets of the data which I call the element subsets. These subsets are: superconductors containing Fe (Iron), containing Cu (Copper), containing MgB_2, and containing Hg (Mercury).

The motivation for the Fe, Cu, and MgB_2 subsets is the body of previous work which applied machine learning to predicting T_c for superconductors containing these compounds. Predicting T_c for superconductors containing these compounds has received priority in the literature due to the practical applications of such superconductors. Thus, it would be useful to characterize how well XGBoost performs on these superconductors.

The motivation for the Hg subset comes from the distribution of T_c values for all superconductors in the dataset, summarized in Figure 1. This distribution is extremely skewed towards low values, and is not particularly clustered around a mean. Thus, it seems possible that the model would be biased towards low T_c values. Superconductors containing Hg offer a "worst case" analysis. Not only do these superconductors have the highest mean T_c of any other elemental class in the dataset, they also have the fourth highest variance [3].

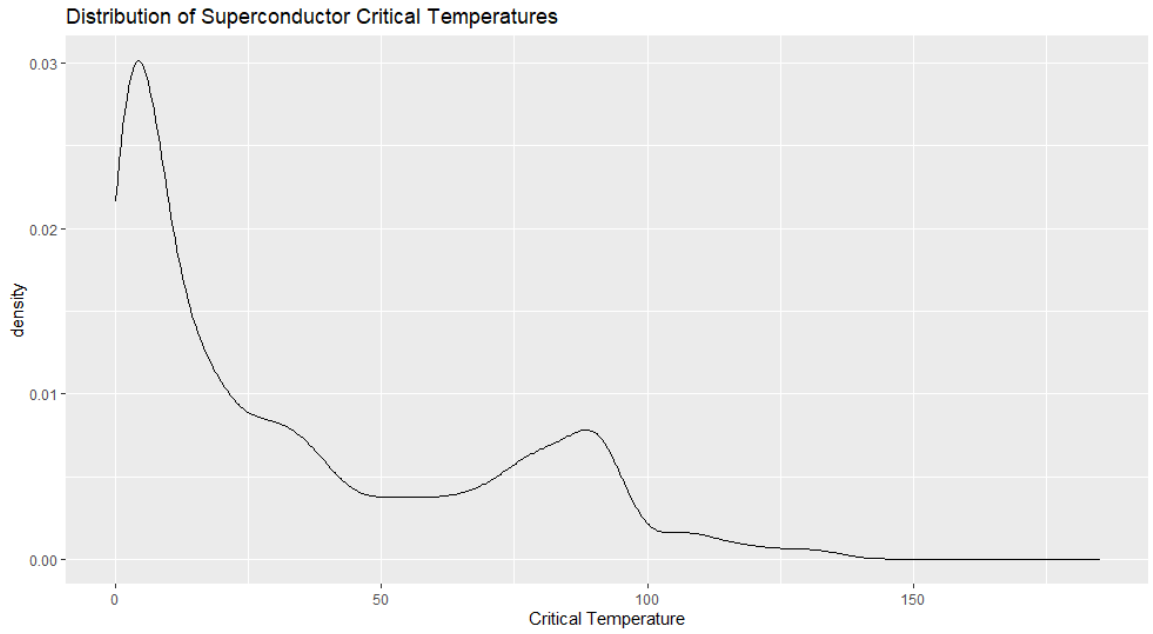

FIGURE 1. Density plot of T_c values for all superconductors in dataset.

Besides determining the performance of XGBoost on these four subsets, we will analyze any potential benefits of training XGBoost only on compounds of a given element subset when trying to predict T_c values for that subset. The motivation for this is to determine if it would be useful to have separate models for these subsets (as has been done in literature previous to [3]), or if the more generalizable XGBoost model given by Dr. Hamidieh provides sufficient predictive power within these subsets.

Since these subsets are determined by true values of the superconductors, they all use the data collection process i.

2.3.2. *The True T_c Quartiles.* As one would expect, these subsets simply the superconductor quartiles as defined by the their true T_c values. The motivation for these subsets is again the distribution of T_c values. Since the distribution is skewed towards low values, it is likely that the algorithm under-predicts values in the upper quartiles.

Identifying this under-prediction (or any other systematic error) may lead to a better optimized algorithm which takes a likely under-prediction of certain superconductors into account.

Since these subsets are determined by the true T_c values of the superconductors, they all use the data collection process i.

2.3.3. *The Predicted T_c Quartiles.* These subsets are the superconductor quartiles as defined by their *predicted* values of T_c. The motivation for this subset is identical to the motivation for the true T_c quartiles, with one distinction. With the predicted T_c quartiles, we will take into account the fact that we do not know the true T_c of an arbitrary conductor. Because of this, implementing a correction term based off of true T_c values would prove difficult, if not impossible.

If a systematic errors can be identified within the predicted T_c quartiles, then implementing a correction term would be easy. We would only need to pick the suitable correction term based off of the observed predicted value of T_c.

Since these subsets are determined by the predicted T_c values, they all use the data collection process ii.

2.3.4. *Control Data.* In the next section, we will also analyze the performance of XGBoost on a simple 2/3 training and 1/3 testing partition of the superconductor data. The purpose of this analysis is to re-create the results in [3] (specifically the out-of-sample RMSE), and to provide a benchmark for the performance of XGBoost in the other subsets.

The control dataset used data collection process i, with no subsets of the data being taken. Thus, 50 error vectors from 50 random test-train data partitions were collected.

3. Results

In this section, I will present the results of the analysis grouped by the data for which XGBoost predicted T_c: the control data, the element subsets, the true T_c quartiles, and the predicted T_c quartiles.

3.1. **Results on the Control Data.** Figure 2 presents average error statistics from the 50 error vectors collected for the control data. Note that the average RMSE is of about 9.5° K, thus we can confirm the out-of-sample RMSE presented in [3].

	RMSE	ave_err	std_err	under_cnt	over_cnt	ave_under	std_under	ave_over	std_over
1	9.518	-4.998	8.099	5611.04	1469.9	-7.008	7.561	2.673	4.843

Figure 2. Average error statistics from the 50 control data error vectors. These error vectors were collected according to data collection process i, with no subsets of the data being taken.

Besides confirming a RMSE of 9.5° K, the data in Figure 2 also suggests that there was both systemic and random error in the T_c predictions by XGBoost, with XGBoost being biased towards low T_c predictions.

The evidence of systemic error comes from the ave_err field. If the prediction error was truly random, then errors should be normally distributed (or at least randomly distributed) about a population mean of 0. Thus, we should expect ave_err to be nearly 0, indicating that XGBoost does not (on average) over nor under predict T_c values. What is observed is that XGBoost, on average, under-predicts T_c by nearly 5° K. Moreover, by comparing the number of over-estimates and under-estimates, we can see that XGBoost under-estimates T_c 79.2% of the time.

Evidence of random error comes from the std_err field. If the prediction error was only due to bias, we should expect the amount of error in each prediction to be closely clustered and directly proportional to the amount of bias in the model. In practice we can see that the distribution of errors has a standard deviation of about 8° K. This suggests that there is some random error present in the model.

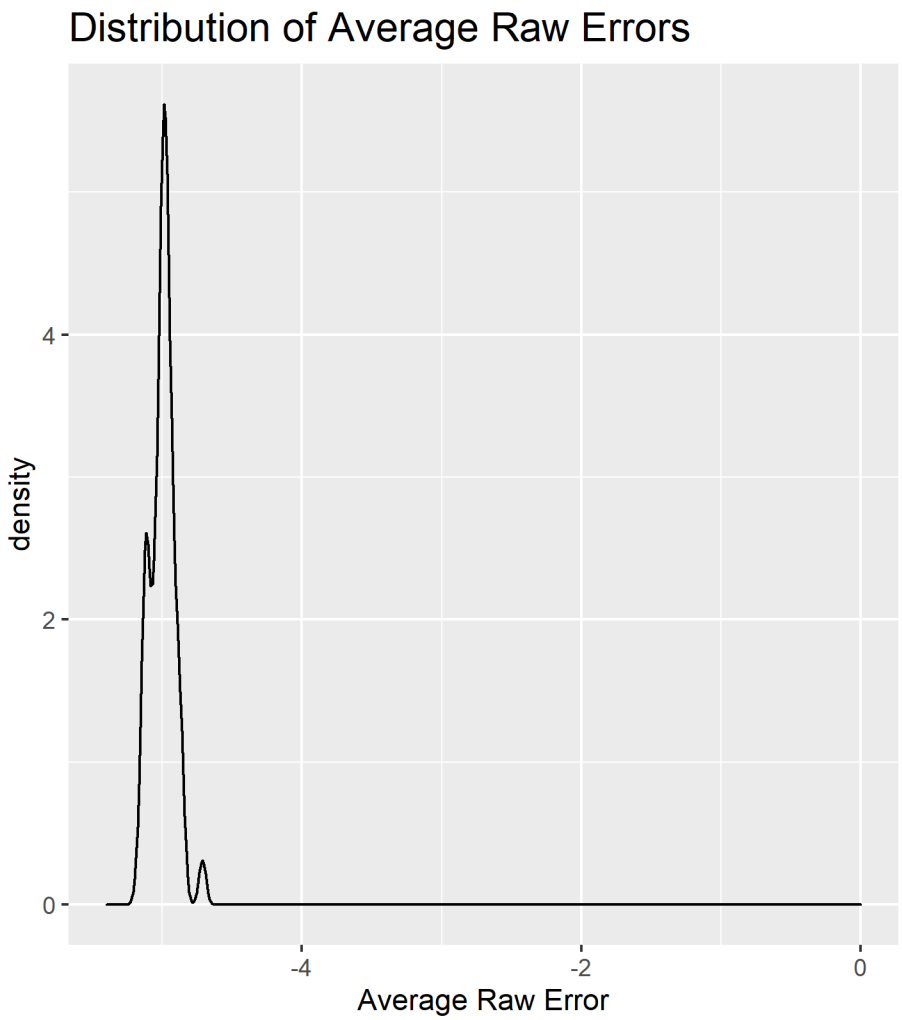

FIGURE 3. Density plot of average raw errors. Raw error averages were taken from the 50 error vectors collected for the control data.

Figure 3 further supports the hypothesis of an under-prediction bias. Again, if prediction inaccuracies were solely due to random errors, we would expect the average raw errors from each error vector to be normally distributed about 0. What is actually observed is an error distribution closely clustered about 5° K, indicating that the errors were not random.

We can further characterize the nature of the error by comparing the distribution of predicted T_c values to that of the actual T_c values, which is done in Figure 4. We can see two trends:

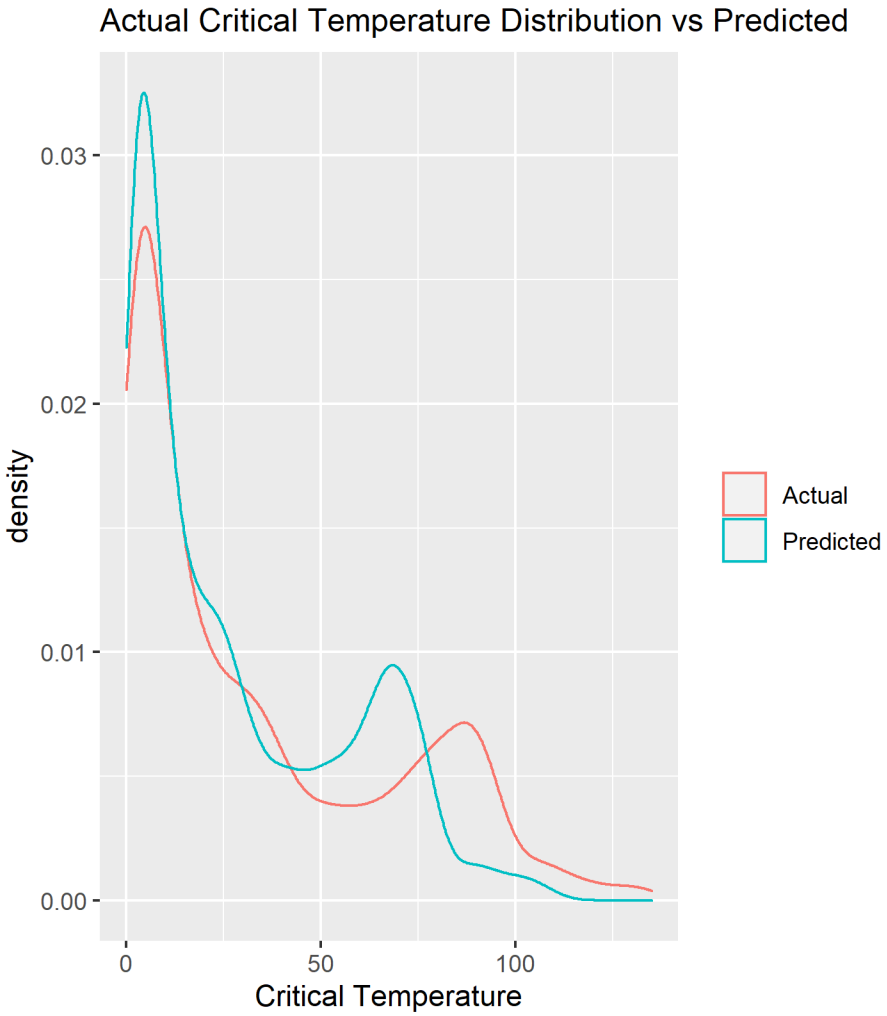

FIGURE 4. Actual test partition T_c distribution plotted alongside the predicted T_c distribution.

(1) XGBoost correctly finds a sharp peak in the distribution of T_c values near a temperature of 0° K. However, XGBoost over-estimates the number of superconductors present in this low T_c cluster.

(2) XGBoost correctly identifies a second small in the distribution with larger T_c values than the low T_c peak. However, XGBoost over-estimates the number of superconductors present in this high T_c cluster, and under-estimates the temperature at which this cluster occurs.

The bias of the model could, in theory, be corrected by applying a naive correction of 5° K to each prediction. However, this may introduce an over-estimation bias in superconductors with low true T_c values. Thus, a finer-grained analysis of the distribution of errors may give a better correction protocol than this naive correction. This is one motivation for the discussion of the T_c quartiles.

	RMSE	ave_err	std_err	under_cnt	over_cnt	ave_under	std_under	ave_over	std_over	Subset
1	7.419	-4.027	6.229	636.44	144.18	-5.572	5.609	2.796	3.723	Fe
2	7.527	-4.257	6.193	147.30	27.86	-5.748	5.097	3.632	5.338	Hg
3	12.819	-8.438	9.649	3046.66	572.56	-11.022	7.745	5.314	6.684	Cu
4	10.046	-4.915	8.351	15.66	2.60	-8.123	2.031	14.474	5.854	B2Mg

FIGURE 5. Average error statistics from 50 error vectors per element subset. The error vectors were collected according to process i.

3.2. **Results on the Element Subsets.** Figure 5 presents average error statistics for the 4 element subsets when XGBoost is trained on a random train partition in the data. Thus, the XGBoost model used to collect these error statistics can be taken to be identical to the model in the control data section.

We can see that XGBoost again has an under-prediction bias across all 4 element subsets, with this bias being most pronounced among the Cu and MgB_2 subsets.

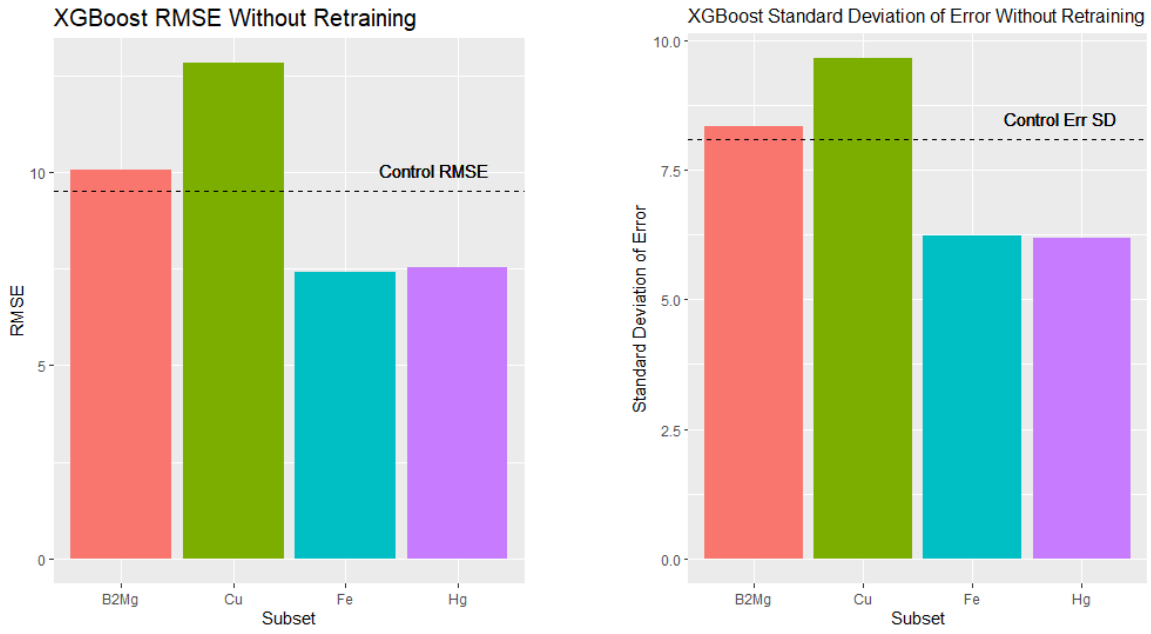

FIGURE 6. RMSE and standard deviation of error on the element subsets without retraining XGBoost on the element subsets. The RMSE and standard deviation of error for the control data are indicated with dashed lines.

Figure 6 allows us to compare the performance of XGBoost across the 4 element subsets and against the control data. The RSME and the standard deviation of error are plotted alongside each other, since the RMSE is a measure of the accuracy of the predictions, while the standard deviation gives a sense of the amount of random error present in each set of predictions.

	RMSE	ave_err	std_err	under_cnt	over_cnt	ave_under	std_under	ave_over	std_over	Subset
1	10.455	-4.458	9.451	572.70	197.42	-7.789	8.020	5.201	5.970	Fe
2	9.142	-4.343	8.024	138.54	35.76	-6.806	6.286	5.228	6.553	Hg
3	16.158	-8.897	13.485	2842.10	770.00	-13.723	9.911	8.917	9.321	Cu
4	9.300	-5.201	7.129	16.30	2.26	-7.713	1.586	13.632	5.899	B2Mg

FIGURE 7. Average error statistics from 50 error vectors per element subset. The error vectors were collected according to process i. In this case, XGBoost was retrained on the elemental subsets.

Both plots tell a similar story. XGBoost preformed with greater accuracy and with less random error in the Fe and Hg subsets than in the control data. In contrast, XGBoost had lower accuracy and more random error among the cuprates and the MgB_2 than in the control data. This comes as a mild surprise, since Hg was supposed to serve as a "worst case" analysis, given the high mean T_c and T_c variance of this subset.

Having established the performance of the control XGBoost model on the element subsets, the element subsets were partitioned into training and testing data, and the XGBoost model was trained and tested accordingly. The results of this process are tabulated in Figure 7 and visualized in Figure 8.

FIGURE 8. Comparison of RMSE and Standard deviation between re-training and no-retraining conditions

	RMSE	ave_err	std_err	under_cnt	over_cnt	ave_under	std_under	ave_over	std_over	control	Quartile
1	1.75	0.23	1.734	933.54	840.52	-0.469	0.379	1.007	2.245	9.515	1
2	3.756	-0.844	3.658	1444.06	317.14	-1.948	1.357	4.183	5.917	9.515	2
3	7.86	-4.398	6.512	1496.38	277.78	-6.32	4.214	5.959	6.921	9.515	3
4	16.829	-14.965	7.695	1738.9	32.02	-15.318	7.289	4.219	3.822	9.515	4

FIGURE 9. Average error statistics from 50 error vectors collected for each true T_c quartile.

Clearly, both the RMSE and the random error (as quantified by the standard deviation) increased with retraining among all subsets but MgB_2 – and the improvement in the performance among the MgB_2 superconductors was negligible. Thus, it does not seem to be advantageous to have separate XGBoost models for the different element subsets.

It is interesting to note that the bias quantified by the average random error does not change much between the retraining and non-retraining conditions. The largest difference in the average random error between training conditions was for Cu, with a difference of 0.459° K. The fact that the average error in the model changed little despite dramatic changes in the training data indicates that the bias might be a result of the XGBoost model itself rather than the training data. However, more analysis is needed before this claim can be made with high confidence.

I end this section by remarking on the low sample size of the MgB_2 superconductors, with an average testing data partition of 20 superconductors. Thus, any conclusions on the performance of XGBoost on this subset should be taken with reservations.

3.3. **Results on the True T_c Quartiles.** Figure 9 presents average error statistics for the 4 true T_c quartiles. As expected from the previously observed under-prediction bias, the table shows that XGBoost preforms with increasingly low accuracy and precision for higher T_c quartiles.

Figure 10 visualizes the bias of the model (as quantified by the average error), the random error (as quantified by the standard deviation of error), and the RMSE of the predicted T_c values among the true T_c quartiles. As expected, the RMSE increases with increasing quartiles. However, we also see that RMSE increases at a rate greater than that of the random error. We also see that the bias increases dramatically with increasing quartiles, indicating that the naive correction term discussed in the control data results is not appropriate.

Figure 10 clearly demonstrates the influence of bias and random error in determining the RMSE. In the lowest quartile, we see that average error (and thus – the bias) is close to zero. Therefore, the errors can be taken to be approximately random and the RMSE is nearly equal to the standard deviation of error, which again quantifies the random error in the predictions. As the average error increases between the quartiles, we see that the RMSE deviates from the standard deviation of error, reflecting an increasing influence of the bias on the RMSE.

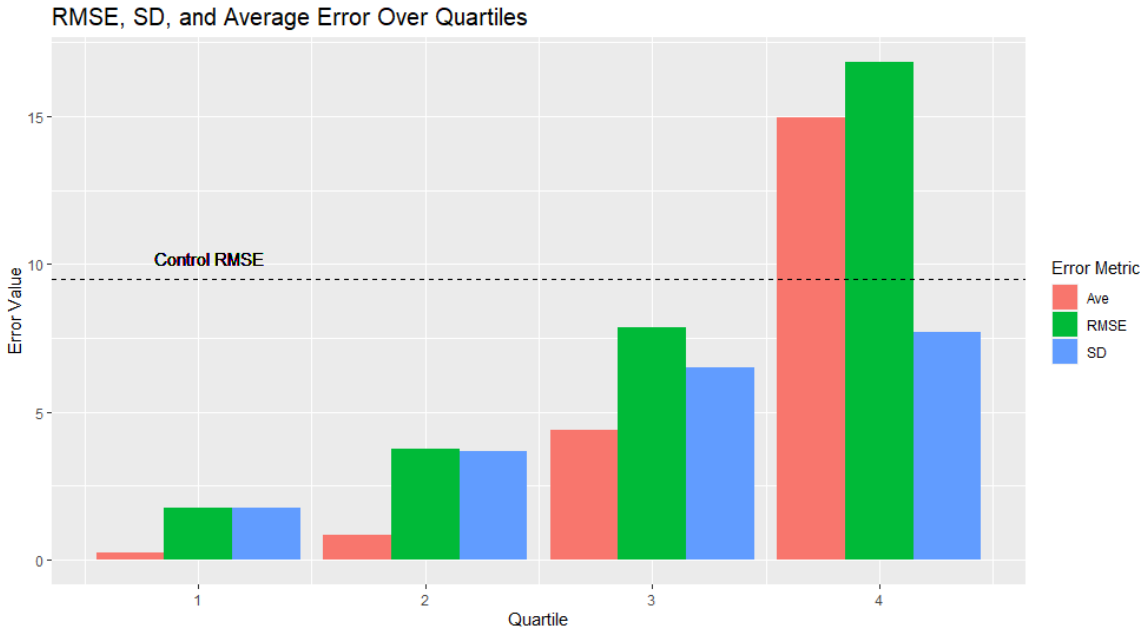

FIGURE 10. RSME, standard deviation of error, and the average of raw errors plotted alongside each other for each true T_c quartile.

This trend culminates in the fourth quartile, where the bias is greater than the random error. Indeed, by looking at the over and under-prediction counts in Figure 9, we can see that the model under-predicts T_c for superconductors in the 4th quartile 98% of the time by an average of 15° K.

The principle application of this model is to find high T_c superconductors. Therefore, the severe bias in the 4th T_c quartile should be corrected. A possible approach would be adding an average error "penalty" term to the RMSE, and using this combined metric as a target for training XGBoost. At the moment, only RMSE is used as a target.

A second, more easily implemented solution would be to add a correction term to each quartile (with the possible exception of the first) to re-center the error distributions about 0, thus eliminating the bias in each quartile. As discussed before, a problem with this approach is that we do not know the true T_c of a superconductor at the time of prediction. Thus, choosing the appropriate correction term based off of the superconductor's true T_c quartile would be impossible, or at least require a classification model to predict this true quartile in addition to the XGBoost regression model – a solution which would be computationally expensive.

To this end, we turn our attention to the *predicted* T_c quartiles. If the same increase in bias can be observed in the predicted T_c quartiles, then we can choose the appropriate correction term without implementing an additional classification model.

3.4. **Results on the Predicted T_c Quartiles.** Figure 11 tabulates the error statistics observed across the predicted T_c quartiles, and these results are visualized in Figure 12.

	RMSE	ave_err	std_err	under_cnt	over_cnt	ave_under	std_under	ave_over	std_over	Quartile
1	0.87	-0.161	0.855	1035.58	736.42	-0.626	0.7	0.493	0.58	1
2	3.497	-1.473	3.17	1445.64	326.36	-2.378	2.506	2.536	2.628	2
3	9.54	-4.99	8.129	1453.36	318.64	-7.374	6.308	5.888	6.446	3
4	16.097	-13.352	8.986	1675.88	96.12	-14.637	7.045	9.064	9.577	4

FIGURE 11. Average error statistics from 50 error vectors collected for each predicted T_c quartile.

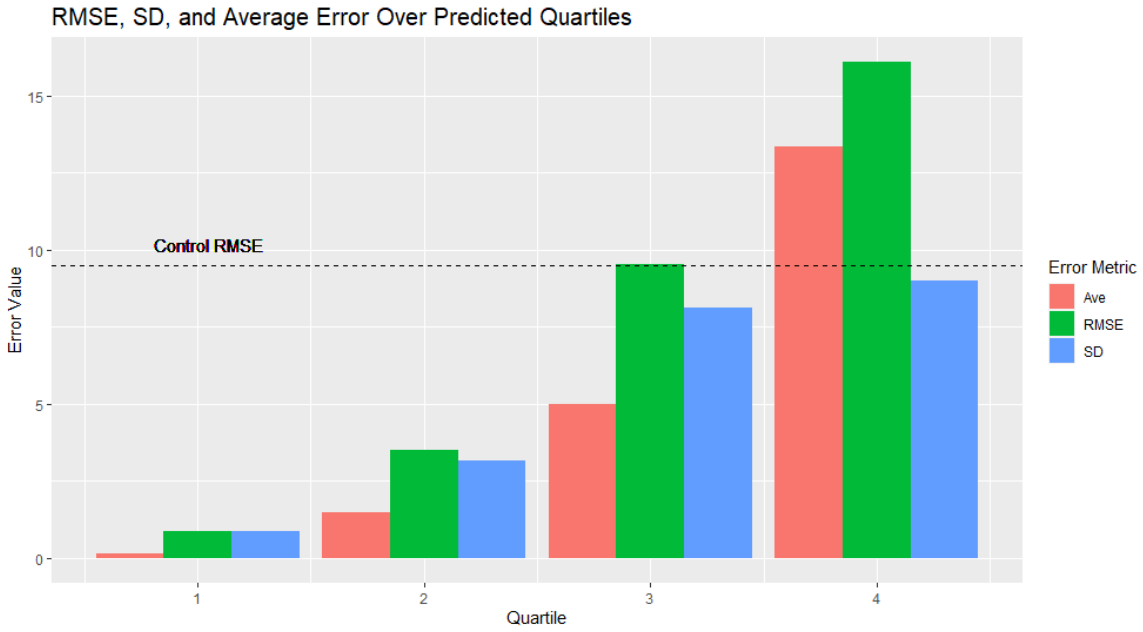

FIGURE 12. RMSE, standard deviation of error, and the average of raw error plotted alongside each other for each predicted T_c quartile.

We can confirm that the same trend of increasing bias observed in the true T_c quartiles is present in the predicted T_c quartiles. Therefore, it is possible to add a correction term to the T_c predictions in each quartile based off of the predicted value of T_c. Doing so would re-center the error distributions of each quartile at 0, and would make the dominating factor in determining RMSE the random error, which is largely unavoidable and as shown in Figure 12, smaller than the bias in the 4th quartile.

4. Conclusion

In this report, we analyzed the performance of XGBoost across three categories of superconductors in order to answer the following three questions:

(1) How well does XGBoost preform at predicting T_c of Iron, Cuprate, Mercury and MgB_2 based super conductors? Does this performance improve or worsen when trained only on Iron, Cuprate, or MgB_2 based superconductors?

(2) If we divide the testing data by quartiles of T_c, how well does XGBoost preform at predicting T_c of compounds in each quartile? Is XGBoost reliable when predicting the T_c of a compound with a high T_c?

(3) If we divide the predictions of XGBoost by quartiles of predicted T_c, how accurate and precise is XGBoost's prediction when the prediction falls in a given quartile? Is the predicted value of T_c reliable when this predicted value is high?

We are now in a position to provide answers to these three questions:

(1) XGBoost predicts T_c for Iron, Cuprate, and Mercury based superconductors reasonably well; in each case XGBoost predicts T_c with an RMSE nearly equal to or below the RMSE of T_c predictions for the control data (9.5° K). The performance of XGBoost at predicting T_c for these subsets worsens with retraining on only training data from these three subsets. Thus, it is does not seem advantageous to have separate models for Fe, Cu, and Hg based superconductors

- The opposite effects were shown for MgB_2 based superconductors. XGBoost preformed considerably worse on this subset, and retraining the model improved performance. However, the low sample size of MgB_2 mean that this should be taken with reservations

(2) XGBoost preforms with rapidly decreasing accuracy for increasing T_c quartiles. Thus, the XGBoost model presented in [3] is not reliable for predicting the T_c of a compound with a high true T_c value. However, this decrease in accuracy is primarily driven by an increase in a low-prediction bias in the model. The bias observed in the true T_c quartiles could be corrected by adding an average error penalty term to the training loss function, which at the moment consists of only the RMSE.

(3) As observed in the true T_c quartiles, XGBoost performs with rapidly decreasing accuracy for increasing T_c quartiles. Thus, the T_c value predicted by XGBoost is not reliable when this value is high. Again, this decreasing accuracy is primarily driven by an increasing low-prediction bias among the quartiles. This bias could be corrected by adding a suitable correction to the predicted value of T_c, wit this correction depending on the quartile in which the prediction lies.

By correcting the bias observed in the predicted and observed T_c quartiles could lead to a dramatically better prediction model for superconductors with high T_c values.

Future work in this direction could include implementing the proposed corrections to XGBoost's predictions, either by adding a average error penalty to the model's loss function, or by adding a correction term to the predicted values of $XGBoost$ which depend on the predicted T_c quartiles.

In addition, the analysis presented in this report relies heavily on summary statistics. The tables presented in the Results section are all averages of 50 or more error vectors. Each error vector is already a set of summary statistics for the raw errors of thousands

of predictions. It may be appropriate to analyze directly analyze the distribution of raw errors rather than the distribution of their averages, as was done in this report. Doing so may yield more accurate data about correction terms and the nature of the bias in this model.

References

[1] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (Oct. 2019). Number: 7779 Publisher: Nature Publishing Group, pp. 505–510. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1666-5. URL: https://www.nature.com/articles/s41586-019-1666-5 (visited on 12/14/2020).

[2] Tianqi Chen et al. *xgboost: Extreme Gradient Boosting*. Version 1.2.0.1. Sept. 2, 2020. URL: https://CRAN.R-project.org/package=xgboost (visited on 12/14/2020).

[3] Kam Hamidieh. "A data-driven statistical model for predicting the critical temperature of a superconductor". In: *Computational Materials Science* 154 (Nov. 2018), pp. 346–354. ISSN: 09270256. DOI: 10.1016/j.commatsci.2018.07.052. URL: https://linkinghub.elsevier.com/retrieve/pii/S0927025618304877 (visited on 12/14/2020).

[4] Taoreed O. Owolabi, Kabiru O. Akande, and Sunday O. Olatunji. "Estimation of Superconducting Transition Temperature TC for Superconductors of the Doped MgB2 System from the Crystal Lattice Parameters Using Support Vector Regression". In: *Journal of Superconductivity and Novel Magnetism* 28.1 (Jan. 1, 2015), pp. 75–81. ISSN: 1557-1947. DOI: 10.1007/s10948-014-2891-7. URL: https://doi.org/10.1007/s10948-014-2891-7 (visited on 12/14/2020).

[5] Valentin Stanev et al. "Machine learning modeling of superconducting critical temperature". In: *npj Computational Materials* 4.1 (Dec. 2018), p. 29. ISSN: 2057-3960. DOI: 10.1038/s41524-018-0085-8. arXiv: 1709.02727. URL: http://arxiv.org/abs/1709.02727 (visited on 12/14/2020).

[6] *Superconductivity Data*. UCI Machine Learning Repository. Nov. 2018. URL: https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data (visited on 12/14/2020).