Carnegie
Mellon
University

Independent Study

for

Data Analytics in Material Science & Engineering

Prof. A.D. Rollett
Student: Kyle Farmer
Date: March 4, 2022
Submission Repository : Github
Data from: Balachandran et al. [1]
Data repository: Github

1 Introduction

Theoretical insight and experimental findings have been the backbone of scientific research for centuries. The digital revolution brought a third paradigm centered on using computational tools to approximate difficult to solve (for humans) theorems and relationships such as Density Functional Theory (DFT) based on the Hohenberg and Kohn equations [2]. As semiconductor research continues to quickly increase our computational power [3], we find ourselves in a data deluge which has necessitated the creation of tools to store, analyze, and query data within the scientific community. This new data revolution has given rise to a field coined “data science”, or “big data”. Data science is a set of fundamental principles that support and guide the principled extraction of information and knowledge from data [4]. The application of data science techniques has brought a new fourth paradigm in scientific discovery centered on intensive computing to exploit high dimensional relationships in datasets that would otherwise be difficult for humans [5].

Data science has seen success in many different fields such as self driving cars, robotics, speech recognition [6], and policy [7] thanks to the to booming field of artificial intelligence and machine learning. While data science is an all encompassing term, machine learning is focused on automatically improving computer systems, through experience, by integrating statistics, computer science, information theory, and domain expertise [6]. Machine learning techniques have recently gained popularity in the materials community thanks to government policies such as the Materials Genome Initiative (MGI) [8]. The MGI’s focus on “discovering and manufacturing new materials in half the time and at a fraction of the cost” has led to an increase in data science proficiency amongst material scientists and provided stable infrastructure to continue to harness the potential of data-driven science[9].

Material discovery is a primary concern of the MGI and an exemplary materials problem that data science, and particularly machine learning, can help solve due to it’s vast search space. Until recently, materials discovery has been limited by experimental procedures such as procurement, availability, and fabrication expertise. Methods such as DFT have shown promising results for high throughput workflows to explore stability of compounds and alloys. However, even the current state of the art super computers limit DFT methods to exploring a limited functional space and unable to search complex solid solutions [10]. Machine learning methods, such as active learning, provide a more efficient pathway to materials discovery. Active learning is an interactive learning algorithm which continuously tries to output a recommended experiment that will try to maximize the performance of the underlying machine learning model. Raccuglia et al. have demonstrated using active learning techniques to discover complex inorganic-organic hybrid materials by gathering data of failed hydrothermal synthesis reactions [11].

In this paper I will explore four machine learning algorithms: Support vector machine for classification (SVC), Principal Component Analysis (PCA), T-distributed Stochastic Neighbor Embedding (t-SNE), Random Forest classification (RF), and K Nearest Neighbor(KNN) to search for high temperature ferroelectric perovskites in a dataset from Balachandran et al. [1]. A perovskite crystal structure is one with an ABO_3 stoichiometry with a symmetry in the set of 15 identified by Lufaso & Woodward [12]. From their original paper, Balachandran used a two-step machine learning model in an active learning loop to iteratively search for potential candidates. I will be focusing on the first step, a classification task of whether a particular compound is a potential candidate or not. I will conclude my discussion with remarks in response to Himanen et al.'s paper on the current status and challenges in materials informatics [9].

2 Methods

2.1 Support Vector Machine

A support vector machine (SVM or SVC for classification tasks) is a training algorithm that maximizes the margin between the training patterns and the decision boundary [13]. A primitive support vector machine could be thought of as a perceptron algorithm, where the model also incurs cost on correct predictions that are near the margin. In practice, the SVM usually takes on a slightly more complex form such as introducing a regularization term (C) to add bias to the model as well as using a kernel (f) to project the data to a higher dimensional space in hopes that the data becomes separable. For completeness I have provided a typical cost function in equation 1 where $h_\theta(x)$ is defined in equation 2. The kernel function f is given in equation 3 and typically transforms the feature vector from $\mathbf{x} \in \mathbb{R}^M$ to $\mathbf{x} \in \mathbb{R}^N$ where N is the number of samples and M is the number of original features.

$$J(\theta) = C\sum_{i=1}^{M} y^{(i)} h_\theta(\theta^T(f^{(i)}) + (1-y^{(i)})h_\theta(\theta^T(f^{(i)}) + \frac{1}{2}\sum_{j=1}^{m}\theta_j^2 \tag{1}$$

$$h_\theta(x) = \begin{cases} 1, & \text{if } \theta^T f >= 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$f(x, l^{(m)}) = exp(-\gamma||x - l^{(m)}||^2) \tag{3}$$

I chose to use this algorithm to enforce the idea of reproducible data [14]. If the results from Balachandran et al. [1] can be matched to a certain extent, then there will be a certain level of confidence in providing further analysis with a given dataset.

2.2 Principal Component Analysis

Principal Component Analysis is a dimensionality reduction technique that creates a new orthonormal basis where the principal components are sorted in order of explained variance. The idea in using this algorithm is to potentially use only the first few principal components to explain almost all of the variance within the dataset. The procedure to compute the newly projected coordinates is by an eigenanlalysis on the covariance matrix to rotate and diagonalize it. In essence, PCA fits a p-dimensional ellipsoid to the data where each axis is a principal component (eigenvector) of the covariance matrix (equation 4), $\mathbf{C}$, and whose length(eigvenvalue) is proportional to the variance for the data in that direction.

$$C_{ij} = \frac{1}{N-1}\sum_{k=1}^{N} X'_{ki}X'_{kj} = \frac{1}{N-1}\sum_{k=1}^{N}(X_{ki} - \bar{X}_i)(X_{kj} - \bar{X}_j) \tag{4}$$

$$X'_{ki} = X - \bar{X}_i \text{ with } \bar{X}_i = 0 \tag{5}$$

I chose PCA because I find it to be an incredibly useful tool to get a feel for feature importance, simplify my data by reducing the number of dimensions, and potentially find clusters in the newly projected data that were not apparent prior.

2.3 T-distributed Stochastic Neighbor Embedding

T-distributed Stochastic Nieghbor Embedding (t-SNE) is a visualization technique for mapping high dimensional data to two dimensions while retaining the local structure and revealing some of the global structure of the data. t-SNE builds off of regular SNE by using an easier to optimize cost function and using a Student-t distribution rather than a Gaussian to compute the similarity between two points in the low-dimensional space. For the sake of brevity in this report I suggest reading the original paper by van der Maaten for details regarding the statistics and implementation [15]. In essence, t-SNE is a simply a visualization tool and the transformed two-dimensional data should not be used for further analysis as this non-deterministic transformation is non-linear and subject to local variations. There is a single hyperparameter within the t-SNE model represented as perplexity. This parameter controls the trade-off of attention given to local vs global trends (suggested value range: 5-50). Another important note is that t-SNE is not deterministic. When applying t-SNE it is recommended to try many different perplexity values, and many iterations using the same perplexity value as well [15].

2.4 Random Forest

Random Forests (RF) are a combination of tree predictors such that each tree depends on the value of a random vector sampled independently and with the same distribution for all trees in the forest [16]. To understand random forests, it is first imperative to understand a decision tree. A decision tree is a simple and natural learning algorithm and is closely related to the computer science notion of "divide and conquer". Put simply, the goal is to come up with a question, regarding the features in the data set, that will maximize the amount of information gained or minimize the loss of the prediction from the results of the conjured question. Going back to the computer science analogy, coming up with a question to split the data is the divide step, then recursively applying this idea throughout the data is the conquer step, until you have reached a leaf node, representing no more data left to split [17].

The primitive decision tree will clearly have issues, as it possesses the capability to perfectly classify the data that trains it. Therefore, generalization methods are necessary to ensure the model can predict unseen data. Firstly, I will mention that decision trees are constructed greedily, meaning the tree makes the locally optimal choice, thus they do not take into account any global trends. To combat the lack of generalization for a single tree, there are pruning techniques which aim to remove unnecessary nodes. Mingers presents a comprehensive overview of five different pruning methods [18]. There are also generalization methods that can be implemented during the construction of the tree such as setting a threshold for the minimum number of data points contained in a leaf node or a threshold on the feature splitting criterion (such as minimum mutual information gain) [17].

A more complex and statistically sound method of generalizing decision trees is to combine many of them through a technique known as "bagging" or bootstrap aggregation to form a Random Forest. Bagging refers to the statistical technique of random sampling with replacement and aggregation refers to returning some average metric. When applied specifically to random forests this means many trees are constructed using only a random subset of both the data points and the features for each tree. Then during prediction, a feature vector is passed through each tree and the average output (or majority vote) across all trees is returned as the prediction. This enforces lower variance in the model without explicitly applying a regularization term to the cost function, thus not increasing the bias. Although random forests eliminate the interpretability that is attractive in a single decision tree, they produce strong models that are not prone to overfitting.

2.5 K Nearest Neighbor

K-nearest neighbors (KNN) is a non-parametric model which simply stores the model during the training phase. Then during prediction, KNN for classification returns the majority vote of it's K closest neighbors using Euclidean distance (equation 6). The intuition behind this is fairly simple, similar data points are expected to return similar outputs. This intuition can be applied to higher dimensional spaces where geometric concepts such as distances can still be applied to assess similarity [17].

$$d(a,b) = \left[\sum_{d-1}^{D} (a_d - b_d)^2 \right]^{\frac{1}{2}} \tag{6}$$

Although this model is non-parametric, KNN does possess a single hyperparameter K which represents the number of neighbors to consider. K, just like other hyperparameters, can, and should, be optimized through various techniques such as grid search.

2.6 Hyperparameter Optimization

Within the machine learning lexicon there is an important distinction between a model parameter and a hyperparameter. A model parameter is estimated directly as a result of the model's loss function whereas the hyperparameters are manually set and are generally related to the structure of the model. For example, in a support vector machine (equation 1) C, the regularization coefficient, and f, the kernel, are hyperparameters which must be set prior to training whereas θ are the weights, or model parameters, learned during training.

While these hyperparameters cannot be learned directly from training, they are encouraged to be optimized. There are plentiful optimization routines but I will highlight one that can be used in conjunction with cross validation.

2.6.1 Grid Search

Grid search is a uniform approach to exhaustively iterate over a search space with uniform increments to find the optimal hyperparameters. Suppose we have hyperparameters $\alpha \in \{a_1, a_2, ..., a_n\}$ and $\beta \in \{b_1, b_2, ..., b_n\}$, run an exhaustive grid search in α and β. This method suffers if the number of hyperparameters is high due to it's exhaustive search characteristic.

2.6.2 Cross Validation

Cross validation is a method of estimating loss on held out data. This is important to report a validation score which can be compared across different models trained during hyperparameter optimization while keeping the test data totally held out. Then a final model is trained on the full training set, based on the best set of hyperparameters optimized through something such as grid search. The unseen test data is then passed through the model and the accuracy is reported.

As for the cross validation algorithm, let's suppose we have N folds. Firstly, shuffle the data randomly, then split the *train* data into N partitions. Train the model on $N-1$ partitions then predict on the left out dataset. Concatenate all the predictions and evaluate loss. Though this computation is slower, the error tends to be more stable and result in a more generalized model.

3 Results

The dataset to be described in this section was originally used in search for high-temperature ferroelectric perovskites. To accomplish this goal, a two step active learning loop was created. This loop can be seen in Figure 1. The first step consists of a classification step to screen for potential compositions with a perovskite structure. The second step uses a regression technique to predict the Curie temperature of the candidate compositions. Only those compositions with a high predicted Curie temperature were fabricated but outputs from both successful and failed experiments were fed back into the classification and regression models to complete the active learning loop. I will be focusing exclusively on the classification model and dataset in my analysis.

3.1 Data

The dataset consists of 15 columns and 193 rows. I will refer to the rows as samples, the subset of columns representing the independent variables as features, and the dependent variable column as the label. There are only 5 features in this dataset: Tolerance Factor, Valence Electron number, Martunov Batsanoc Electronegativty, Ideal Bond Length sum, and Mendeleev number [19][20][21][22]. Pilania et al. [23] showed promising results when using these features in classifying similar perovskite crystal structures and were thus used in Balachandran's work as well. The tolerance factor (t_f) is an engineered feature which assigns a continuous numeric value to the chemical composition. t_f is shown in equation 7 where t_f is calculated as $(r_A + r_O/)\sqrt{2}(r_B + r_O)$ and r_A, r_B, and r_O are the weighted-average Shannon's ionic radii of the A-site, B-site, and O atom, respectively, in the perovskite lattice [24].

$$t_{f,solidsolution} = \left\{x \times t_{f,Bi(Me'Me'')O_3}\right\} + \left\{(1-x) \times t_{f,PT}\right\} \tag{7}$$

There is only one label in this dataset, the formability of the candidate composition into a perovskite structure, represented in the binary pair: 1 or -1 (formality of working with certain classifiers).

The analysis of this data was performed exclusively with Python [25]. Notable packages include: 1) Pandas, for spreadsheet-like manipulation and visualization [26] 2) Numpy, for fast numerical computation [27] and 3) Sci-kit learn, for all model implementations [28].

3.2 Clustering/PCA

The goal of classification is to build a model that predicts a label, or class, for each sample. However before building the classification model, I started my analysis with a couple of unsurpersived learning methods, namely PCA and t-SNE, in hopes of gaining more insight about the individual features. Figure 2 shows a few outputs from performing PCA on the five features in the dataset. The scree plot, shown on the left, shows the explained variance ratio per principal component. As shown, about 75% of the variance is explained with only the first 2 principal components, while the last two principal components make up just over 10% explained variance. The 2 biplots, shown center, pack information regarding the impact that each feature has on the individual principal components, along with the projection of the individual samples onto the respective principal components. The top biplot shows the first two principal components (PC1 & PC2) and the bottom biplot shows the second two principal component (PC2 & PC3). The data remains fairly compact with the change of basis and there seems to be no real clustering. The biplot also shows the relative importance of the features. Vectors with large magnitudes indicate a large impact on the corresponding principal component. For example, the tolerance factor lies on the lowest point in the first biplot and almost exactly on the y-axis. This means tolerance factor has the greatest impact on the first principal component and almost no influence on the second principal component. Remember the first component explains the largest amount of variance within our data, thus indicating tolerance factor may have a large influence on how our models perform.

To further investigate potential clustering in the data, t-SNE plots were generated using perplexity values in the range of 5-50. Three of the best plots were included in Figure 3. Different color scales were used in search of the clustering criteria. The color scales used were by: formability, unique compound identifier, and normalized tolerance factor.

3.3 Classification

For classification, three different classifiers were trained with 10-fold grid search cross-validation. Table 1 lists their performance. The first model trained was a support vector machine. A radial basis function kernel was used as the feature transformer. The two hyperparameters, C, the regularization coefficient and γ, the kernel coefficient found optimal results at values of 1 and 0.1, respectively. The data was broken into 100 pairs of 152 training examples and 40 test samples. Then, 100 SVC classifiers were built from the respective training sets and used to predict on the corresponding test set. The overall accuracy was then calculated by taking the average of all 100 classifiers. This resulted in an accuracy of $77.9\% \pm 0.061\%$ which closely compares to the work of Balachandran. This score is not unreasonable from heterogenous datasets and performs better than a random vote classifier (50%) and a majority vote classifier (65%). [1]

Secondly, a K-Nearest Neighbors classifier was trained using the same split of data detailed in the previous section. Grid search cross validation was used to find the optimal number of neighbors to include and also whether to weight the distances as well. The optimal parameters came to be $K = 1$ and to leave the distances unweighted (although this does not matter for a single nearest neighbor). 1-NN was able to classify 82.5% of unseen samples accurately.

Lastly, a Random Forest classifier was trained, again using the same split of data mentioned in the previous sections. Gridsearch cross validation was used and found the optimal values for max depth, number of trees, and minimum number of samples to split to be 10, 2, and 141 respectively. The random forest classifier was only able to predict 53% of unseen data accurately, despite the rigorous search for parameters to ensure a generalizable model. Due to the complexity of Random Forests, a feature importance score can be estimated. Figure 4 shows the associated importance score using permutation feature importance. This technique can be loosely defined as the decrease in the model's performance when a single feature value is randomly shuffled. Essentially, if the model's performance suffers greatly when a single feature value is randomly changed, a conclusion can be drawn suggesting that particular feature is important. This technique grades tolerance factor as the overwhelming most important feature, which agrees well with what was shown in the PCA and t-SNE analysis.

4 Discussion

4.1 Clustering and classification

Finding which particular ABO_3 compounds form stable crystal structures in the set of perovskite space groups has been a long standing study in materials science. The search space for these compounds is in the order of 61000 unique compounds. Pilania et al. helped narrow down the best features to describe the perovskite compounds but unfortunately the available experimental data of successful and failed perovskite structures synthesis is scarce and heterogeneous [23]. I believe both of these factors plays a major role in the accuracy scores in the reported classification models. An accuracy of 77.9% with SVM and 82.5% with k-NN has been expected for this type of data. The fact that the best performing k-NN model was with a $K = 1$ is interesting. This is saying that the information that leads to the best prediction of whether a compound will form is held in the single closest data point. This highlights the scarcity of the dataset as well as having a unique structure which a euclidean distance function is unable to capture.

The strange structure of the dataset is highlighted in the t-SNE plots in Figure 3. Each of these three t-SNE plots lends a small clue into the structure of the data. Firstly, when colored by formability, the data does not have any defining clusters, however, the formable compounds are found on the ends of the projected data. When colored by compound, each compound forms about a straight line to the center of the data space. Then thirdly, when colored by a normalized tolerance factor, the data looks almost like a firework, with energy concentrated in the center and dissipating outward. t-SNE is a fairly complex procedure and can lead to ambiguous interpretation, but I believe these results warrant further investigation.

The Principal component analysis further confirms my suspicions of feature importance. The tolerance factor dominates the first PC which makes up about 50% of the explained variance. This agrees with the random forest permutation estimation of feature importance. The scree plot shows that each Principal Component holds an appreciable amount of variance (all >5%). The biplots also confirms my suspicion that the data is not structured in a very straightforward way. There is no real distinction between what is formable vs what is not formable using any pair of the first three PCs.

Although these features are well studied to result in the best prediction results, they are purely geometric and chemical features which may have some correlation. Since the data was compiled from various experimental findings, it may be beneficial to add processing and fabrication features to help mitigate the noise common in heterogeneous datasets.

4.2 Data-Driven Materials Science Review

I will wrap up the my discussion by answering five questions in response to the review of Data-Driven Materials Science by Himanen et al [9].

4.2.1 What Does "FAIR" mean in the context of databases?

The FAIR guiding principles for scientific data management and stewardship were published in Scientific Data in 2016 [29]. These virtues were identified as the key principles to enable agile scientific data aggregation and manipulation. *Findable* represents the data-to-metadata relationship which should become uniform to allow humans and computers to quickly query similar data. *Accessible* represents the ease to which a user can access the data, once they have found what they are looking for. For example, is there an API they can access, provided authentication, to download large files or is there a browser based option to directly download a file? *Interoperable* represents the ability for data to be easily integrated into different analysis pipelines. For example, if I am looking at processing-structure-property relationships, I may compile a heterogenous dataset for each aspect in the relationship. The data from each source should seamlessly integrate into a single pipeline for analysis. *Reusable* represents the completeness and accuracy of the metadata within a dataset. The goal here is for scientists to come across a dataset and be able to reproduce this data based on the knowledge contained within the metadata. This ensures accurate and descriptive datasets.

4.2.2 Following up on the "Open Data" discussion, give an example of a published paper in materials science that offers the data behind the paper

Luckily, with the emergence of Journals such as Data in Brief over the last few years, finding data relating to a subject has become increasingly easier, but still scarce. Frankly, I started my search for data by searching the Data in Brief archive. With that being said, it is unusual for me to find the associated data when reviewing literature, and if there are additional resources included, it is normally not in an easily accessible format, such as a table in a PDF. The paper used in this report from Balachandran et al. offers the accompanying data. [1]

4.2.3 Table 1 lists the materials databases known to the authors at the time of writing. Comment on how you think the situation with respect to data has changed since Himenan et al. publishes their paper.

Just as Himanen mentioned in their respective article, the Open Access and Open Source communities have continued their rise in popularity, and with that, data availability. The overall scientific community has become more adept with data science and informatic techniques and infrastructures. For example, Nanohub is an opensource platform for computational research, education, and collaboration in materials science and related fields [30]. Their site hosts a plethora of lecture series and computational tools that utilize cloud computing. Platforms like this create a collaborative environment that is necessary in the open-source community to continue producing large amounts of useful data. This trend in proficiency has likely made the databases Himanen lists, more popular and also better maintained. With that there is probably a larger number of materials databases, as Himanen mentions there were databses emerging between the times the paper was written and published.

4.2.4 Investigate the data resource centers listed in Table 2 and report on any cases where the website did not, in fact, have an advertised service. Also report an example that either corresponds to a technique you have learned or something that is clearly a data analytics method.

The Computational Materials Repository Provides links to many different computational materials databases and provides code snippets for how to produce various results using the data within a particular database [31]. I suppose Weinheim interpreted this as a "data analysis tool" though there is no interaction between user and machine, simply documentation, therefore I would not count this as a tool. There are also the cases of commercial level software such as Materials Design and Citrine, that may vary well have data analysis tools, but they are not freely available on their website [32][33]. AFLOW (Automatic FLOW for Materials Discovery) is a one-stop-shop for workflows and databases related to materials discovery [34]. They offer 8 different material databases and a suite of data analysis tools from data visualization, comparison, and ML models. Within the AFLOW-ML module, there are 3 different models utilizing gradient boost decision trees and Random Forests, which we covered in this course.

4.2.5 Discuss how your results correspond to the advice given in section 7 of the paper

Himenan et al. emphasize the importance of feature engineering in materials science data as it is commonly supplied in a form unusable by most machine learning models. Machine learning models expect numeric features whereas chemical compositions are often an important description of data within materials science and are normally not represented numerically. Therefore it is imperative to come up with a method to encode our composition into a numeric value. This was illustrated in the data I worked with in the tolerance factor t_F feature. Secondly, Himenan et al emphasizes the importance of selecting the proper model for the given data. Although my objective was a rather simple binary classification problem, I was limited to a relatively small number of examples, therefore a high feature space model such as a neural network would not perform well. A support vector machine, or nearest neighbor classifier, however should perform just fine. Lastly, Himanen et al. highlights the importance of performance assessment and points out two major problems: overfitting and underfitting. Overfitting results in a model that does not generalize by incorporating too much of the variance from the training set. Underfitting results from not including enough variance and introducing too much bias to not truly capture the trends within the training data. My attempt at controlling this trade-off was using grid search cross validation to estimate the optimal hyperparameters using a training and validation set. Once the hyperparameters were found I trained the final model with both the training and validation set combined then tested on a partition that the model has never seen before in attempt to best estimate the true error of the data.

5 Conclusions

In this report I shed light on the growing field of materials informatics and gave examples of how machine learning models can integrate with materials science to benefit material discovery. I used the data provided by Balachandran et al. to classify whether a candidate compound would form into a perovskite crystal structure [1]. I used three different classification methods: support vector machine, K-nearest neighbors, and random forest. A 1-nearest neighbor classifier provided the best prediction with an accuracy of 82.5%. Furthermore I investigated the relationships amongst the five features within the dataset with principal component analysis and t-SNE. Though a clear clustering scheme could not be drawn, the t-SNE plot suggests that there is a complex structure associated with the tolerance factor that is worth investigating. Lastly, I responded to a review of the current landscape of data driven materials science by answering five questions pertaining to a paper by Himanen et al.

Classifier	Accuracy(%)
Balachandran et al. SVM	77.5
SVM	77.9
KNN	82.5
RF	53.1

Table 1: Classifier performances

6 Figures

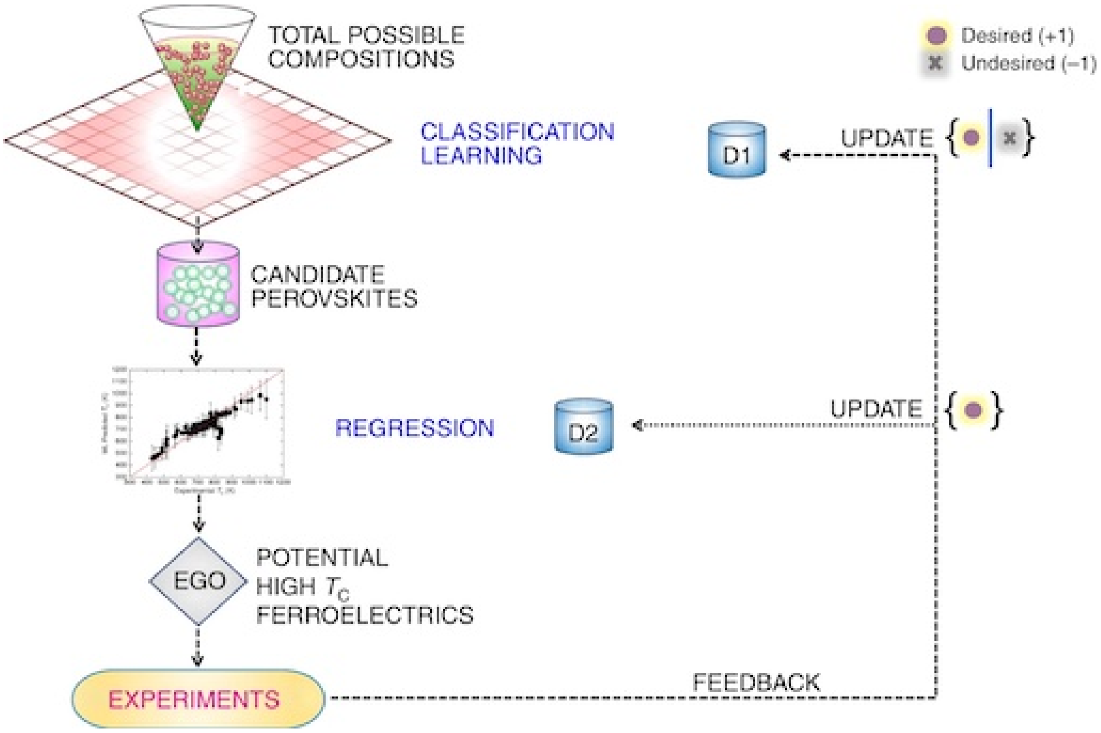


Figure 1: Two step machine learning strategy for sequentially guiding experiments from Balachandran's original paper

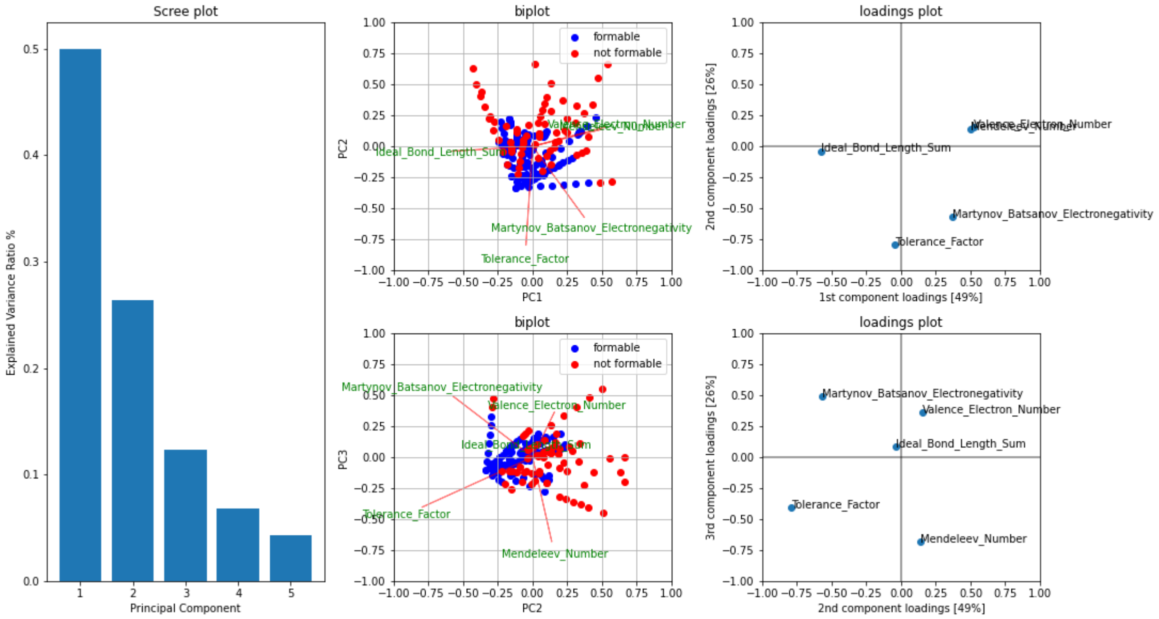
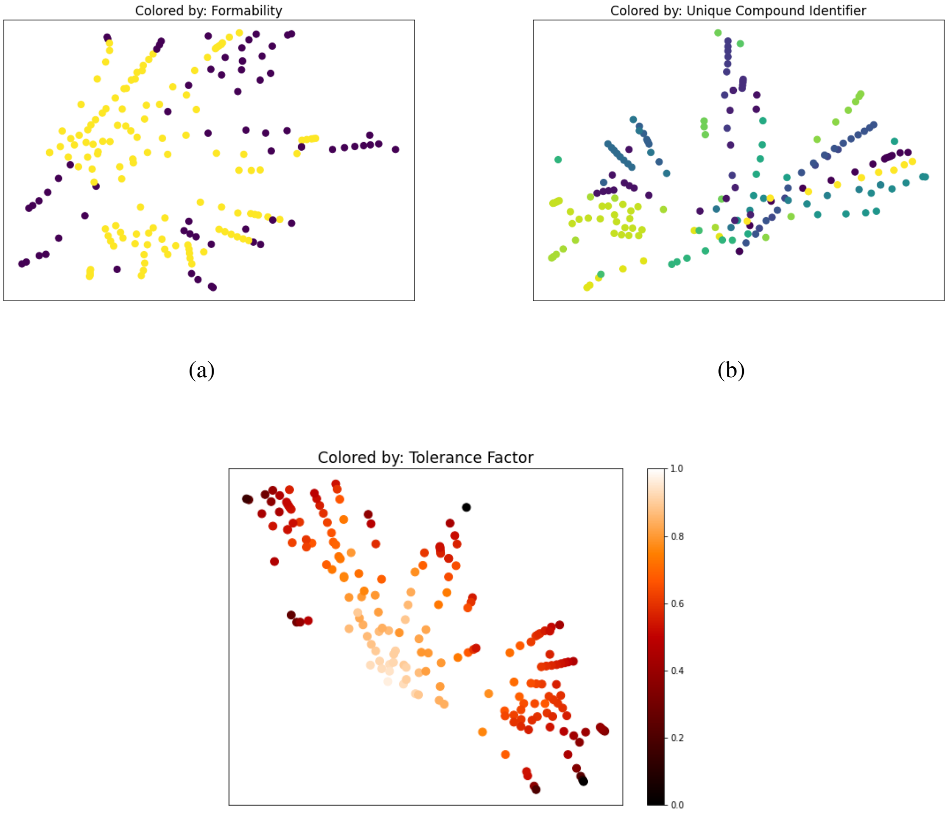


Figure 2: Scree Plot, Biplot, & Loadings plot from Principal Component Analysis



(a)

(b)

(c)

Figure 3: t-SNE

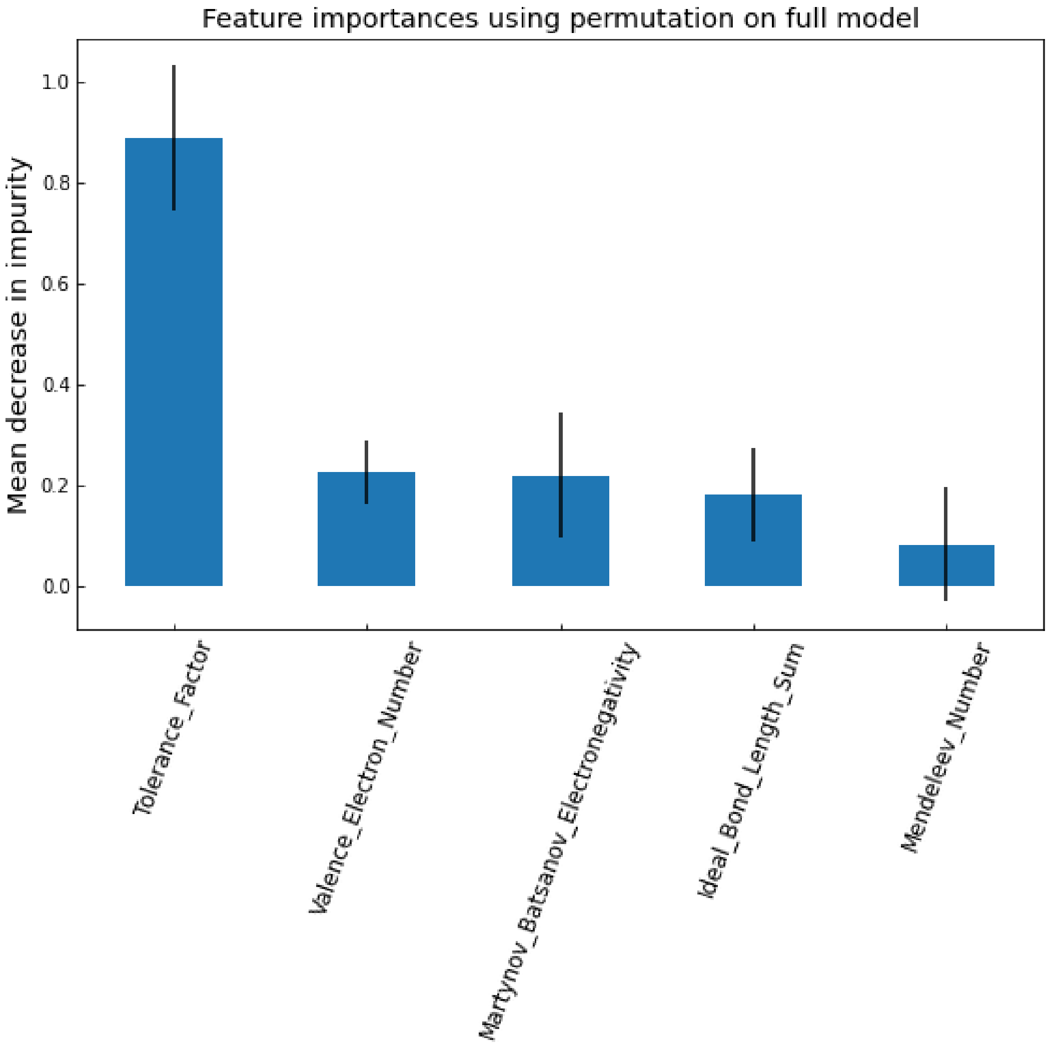


Figure 4: Random Forest feature importance description

References

[1] Prasanna V Balachandran, Benjamin Kowalski, Alp Sehirlioglu, and Turab Lookman. Experimental search for high-temperature ferroelectric perovskites guided by two-step machine learning. *Nature communications*, 9(1):1–9, 2018.

[2] Hohenberg P. and Kohn W. *Phys. Rev.*, 136, 1964. Cited by: 4.

[3] Tanya Nigam, Kok-Yong Yiang, and Amit Marathe. Moores law: Technology scaling and reliability challenges. *Microelectronics to Nanoelectronics: Materials, Devices & Manufacturability*, page 1, 2012.

[4] Foster Provost and Tom Fawcett. Data science and its relationship to big data and data-driven decision making. *Big data*, 1(1):51–59, 2013.

[5] Anthony JG Hey, Stewart Tansley, Kristin Michele Tolle, et al. *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA, 2009.

[6] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[7] Mario Molina and Filiz Garip. Machine learning for sociology. *Annual Review of Sociology*, 45:27–45, 2019.

[8] National Science and Technology Council (US). *Materials genome initiative for global competitiveness*. Executive Office of the President, National Science and Technology Council, 2011.

[9] Lauri Himanen, Amber Geurts, Adam Stuart Foster, and Patrick Rinke. Data-driven materials science: status, challenges, and perspectives. *Advanced Science*, 6(21):1900808, 2019.

[10] Romain Gautier, Xiuwen Zhang, Linhua Hu, Liping Yu, Yuyuan Lin, Tor OL Sunde, Danbee Chon, Kenneth R Poeppelmeier, and Alex Zunger. Prediction and accelerated laboratory discovery of previously unknown 18-electron abx compounds. *Nature chemistry*, 7(4):308–316, 2015.

[11] Paul Raccuglia, Katherine C Elbert, Philip DF Adler, Casey Falk, Malia B Wenny, Aurelio Mollo, Matthias Zeller, Sorelle A Friedler, Joshua Schrier, and Alexander J Norquist. Machine-learning-assisted materials discovery using failed experiments. *Nature*, 533(7601):73–76, 2016.

[12] Michael W Lufaso and Patrick M Woodward. Jahn–teller distortions, cation ordering and octahedral tilting in perovskites. *Acta Crystallographica Section B: Structural Science*, 60(1):10–20, 2004.

[13] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[14] National Academies of Sciences Engineering Medicine et al. Reproducibility and replicability in science. 2019.

[15] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[16] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[17] Hal Daumé. *A course in machine learning*. Hal Daumé III, 2017.

[18] John Mingers. An empirical comparison of pruning methods for decision tree induction. *Machine learning*, 4(2):227–243, 1989.

[19] VM Goldschmidt, T Barth, G Lunde, and WH Zachariasen. Vii: Die gesetze der krystallochemie. *Geochemische Verteilungsgesetze der Elemente (Oslo, 1926)*, 1926.

[20] KM Rabe, JC Phillips, P Villars, and ID Brown. Global multinary structural chemistry of stable quasicrystals, high-t c ferroelectrics, and high-t c superconductors. *Physical Review B*, 45(14):7650, 1992.

[21] P Villars, K Cenzual, J Daams, Y Chen, and S Iwata. Data-driven atomic environment prediction for binaries using the mendeleev number: Part 1. composition ab. *Journal of alloys and compounds*, 367(1-2):167–175, 2004.

[22] Ian David Brown. Recent developments in the methods and applications of the bond valence model. *Chemical reviews*, 109(12):6858–6919, 2009.

[23] Ghanshyam Pilania, PV Balachandran, James E Gubernatis, and Turab Lookman. Classification of abo3 perovskite solids: a machine learning study. *Acta Crystallographica Section B: Structural Science, Crystal Engineering and Materials*, 71(5):507–513, 2015.

[24] Robert D Shannon. Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides. *Acta crystallographica section A: crystal physics, diffraction, theoretical and general crystallography*, 32(5):751–767, 1976.

[25] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[26] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[27] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[28] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[29] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9, 2016.

[30] Gerhard Klimeck, Michael McLennan, Sean P Brophy, George B Adams III, and Mark S Lundstrom. nanohub. org: Advancing education and research in nanotechnology. *Computing in Science & Engineering*, 10(5):17–23, 2008.

[31] David D Landis, Jens S Hummelshøj, Svetlozar Nestorov, Jeff Greeley, Marcin Dułak, Thomas Bligaard, Jens K Nørskov, and Karsten W Jacobsen. The computational materials repository. *Computing in Science & Engineering*, 14(6):51–57, 2012.

[32] Materials design inc. `https://www.materialsdesign.com/`. Accessed: 2022-03-01.

[33] Citrine informatics. `https://citrine.io/`. Accessed: 2022-03-01.

[34] Stefano Curtarolo, Wahyu Setyawan, Gus LW Hart, Michal Jahnatek, Roman V Chepulskii, Richard H Taylor, Shidong Wang, Junkai Xue, Kesong Yang, Ohad Levy, et al. Aflow: An automatic framework for high-throughput materials discovery. *Computational Materials Science*, 58:218–226, 2012.