Predicting the thermodynamic stability of perovskite oxides

Weixi Yao, Seung-Jae Bang, William Yu, Jiaying, Chen, Yiming Huang, Caroline Rutherford

Supervised by Prof. Simon Billinge

Project Background/Motivation

- Aim to use data to find better materials for sustainable energy applications
  - Could be naturally occuring, previously synthesized, or hypothetical compounds
- Motivation: Currently, most energy used is carbon-based from fossil fuels—not sustainable. Need to discover and deploy new materials in order to scale sustainable energy sources, i.e. solar panels
- Perovskites are a promising and increasingly popular material for use in solar cells because of their efficiency, low cost, and scalability
- While perovskites are classically investigated using density functional theory, machine learning has shown promising results in predicting indicators of perovskite stability
- This project focuses on energy above the convex hull

How Does This Relate to ML?

- Traditional method: use **DFT (Density Functional Theory)** to approximate phase stability
    - Built upon electron density functional
    - Comparing the total energy of a compound to other nearby structural arrangements of the same elements
- DFT computational cost is very high
- Approximate phase stability through machine learning
- Use Energy Above Convex Hull (Ehull) as target variable
    - Stable compounds: Ehull < 40 meV/atom
    - Unstable compounds: Ehull > greater than 40 meV/atom

Problem Formulation

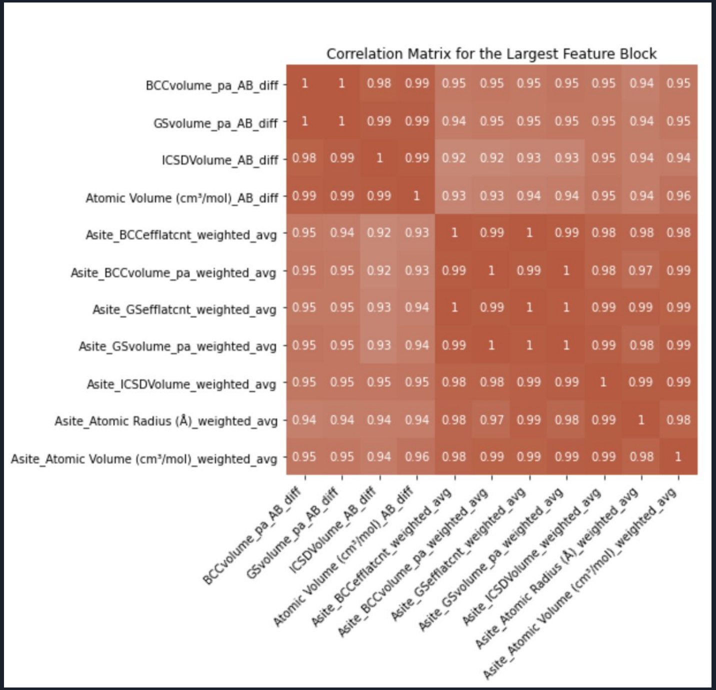
- Classification Task:
  - Encode > 40 meV/atom as unstable compounds, encode < 40 meV/atom as stable compounds
  - Binary classification task
- Regression Task:
  - Predict range of Ehull

Pre-modeling Analysis

- Objective: Perform an exploratory data analysis to reduce feature space and explore informative features
- Variable Selection:
    - Discard features with 0 variance (discard 171 features)
    - Remove redundant features
        - Since the author has created derived features combining the element properties of A site / B site with their min / max / average, etc., many features contain redundant information
        - In EDA stage, we experimented with clustering features into blocks that have very high pairwise correlations (>90%), and then picking one representative feature from each block
- Composition grouping as a feature

EDA - Removing Redundant Features

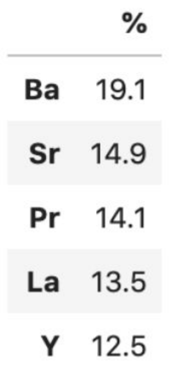
- We experimented with creating blocks of features that are redundant by running a hierarchical clustering on feature correlation matrix with threshold of 90%.
- Once we have determined these cluster blocks, there are many ways to represent the feature block for modeling purposes - such as extracting components of each block through PCA, using average of the features for the block, etc. For EDA purpose we extracted one feature from each block (discard 354 features)


Correlation Matrix for the Largest Feature Block

EDA - Composition Grouping

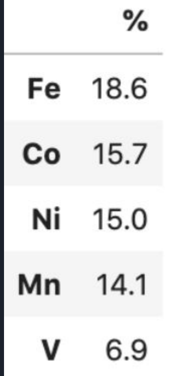
- Objective: Test our intuition that there may exist a relationship between the chemical composition of perovskite oxide and thermodynamic stability
- Focus on major elements that are included on the A-site and B-site of the compositions

Major A-site Elements



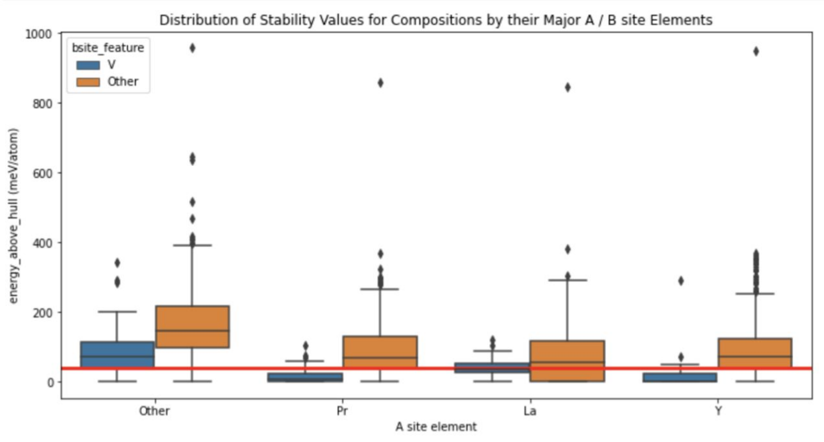
	%
Ba	19.1
Sr	14.9
Pr	14.1
La	13.5
Y	12.5

Major B-site Elements



	%
Fe	18.6
Co	15.7
Ni	15.0
Mn	14.1
V	6.9

EDA - Composition Grouping



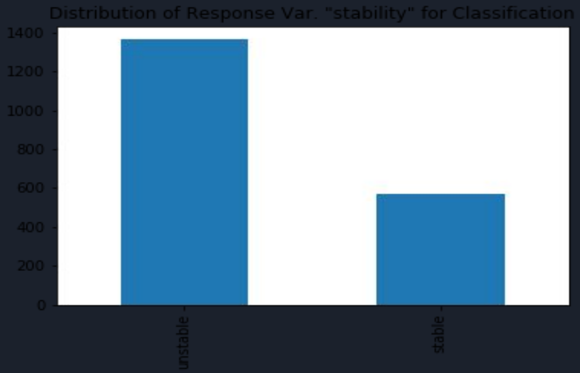
- Compositions that contain ‘Pr’ or ‘Y’ on the A-site and ‘V’ on the B-site are likely to be stable
- Compositions that do not include ‘Pr’, ‘La’ or ‘Y’ on the A-site are likely to be unstable

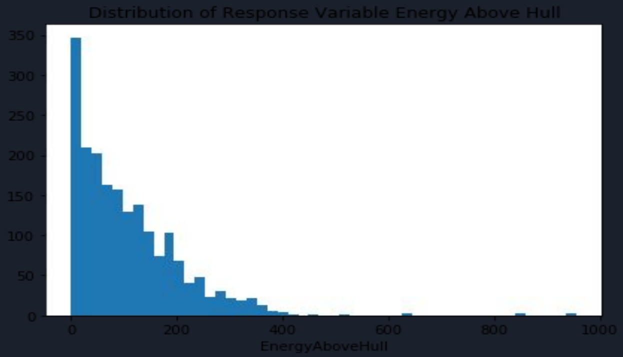
Data augmentation - sample augmentation

- Increase number of materials in our dataset
- Li et al.'s 2018 paper *Predicting the thermodynamic stability of perovskite oxides using machine learning models*: 1,929 simulated perovskite oxide compositions
- Jacobs et al, 2018 paper *Material discovery and design principles for stable, high activity perovskite cathodes for solid oxide fuel cells*: 2,145 simulated compositions
- 7 duplicates → 2,138 compounds in final sample
- In order to generate the 962 features, we extracted the elements in each site and used Li 2018's functions drawing mainly from the elemental property table
- Increase of ~0.01 in the R-squared and the weighted average f1 score

Algorithmic Oversampling

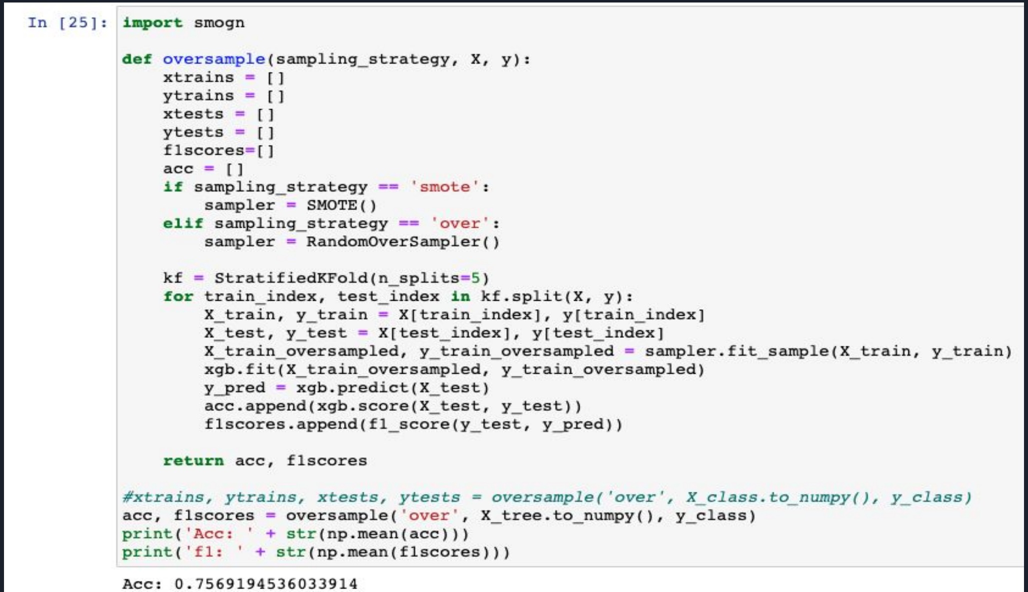
- Class imbalance in the classification task (much more unstable compounds than stable)
- Distribution is heavily skewed right in regression task (graphs below)
- Employ oversampling algorithms to both tasks
  - Incorporate oversampling in each **cross-validation fold**
  - 5 fold cross validation example:
    - For the first fold, oversample on the 4 groups of training data and fit model to the data, and then evaluate on the last group (not oversampled)
    - Continue with next fold





Algorithmic Oversampling

- Classification Task: **SMOTE** (Synthetic Minority Oversampling Technique)
  - f1-score: 0.731
- Regression Task: **SMOGN** (Synthetic Minority Oversampling for Regression with Gaussian Noise)
  - R-squared: 0.727



```
In [25]: import smogn

         def oversample(sampling_strategy, X, y):
             xtrains = []
             ytrains = []
             xtests = []
             ytests = []
             f1scores=[]
             acc = []
             if sampling_strategy == 'smote':
                 sampler = SMOTE()
             elif sampling_strategy == 'over':
                 sampler = RandomOverSampler()

             kf = StratifiedKFold(n_splits=5)
             for train_index, test_index in kf.split(X, y):
                 X_train, y_train = X[train_index], y[train_index]
                 X_test, y_test = X[test_index], y[test_index]
                 X_train_oversampled, y_train_oversampled = sampler.fit_sample(X_train, y_train)
                 xgb.fit(X_train_oversampled, y_train_oversampled)
                 y_pred = xgb.predict(X_test)
                 acc.append(xgb.score(X_test, y_test))
                 f1scores.append(f1_score(y_test, y_pred))

             return acc, f1scores

         #xtrains, ytrains, xtests, ytests = oversample('over', X_class.to_numpy(), y_class)
         acc, f1scores = oversample('over', X_tree.to_numpy(), y_class)
         print('Acc: ' + str(np.mean(acc)))
         print('f1: ' + str(np.mean(f1scores)))

         Acc: 0.7569194536033914
```

Model Optimization
Part I

Baseline Models

GBDT VS SVM

GBDT - a combination model of decision trees

Regression:

MAE - 28.095110093551593
RMSE - 58.6726129559543

Classification:

F1 score -0.8294314381270903

Baseline Models

GBDT VS SVM

SVM - uses classification algorithms for two-group classification problems

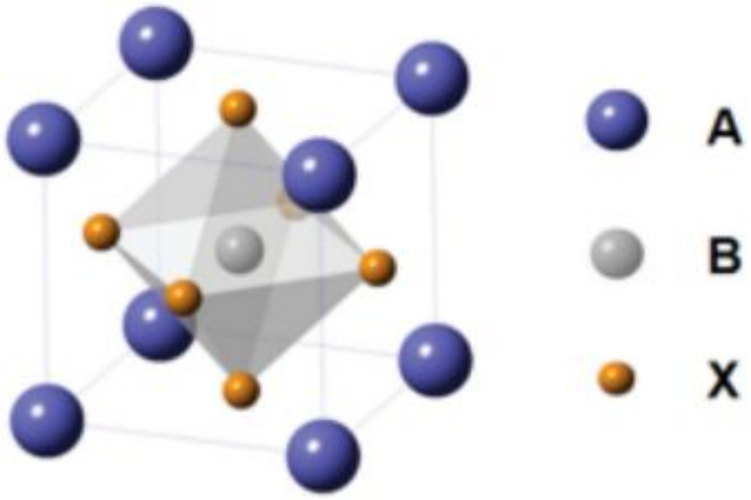
Regression:

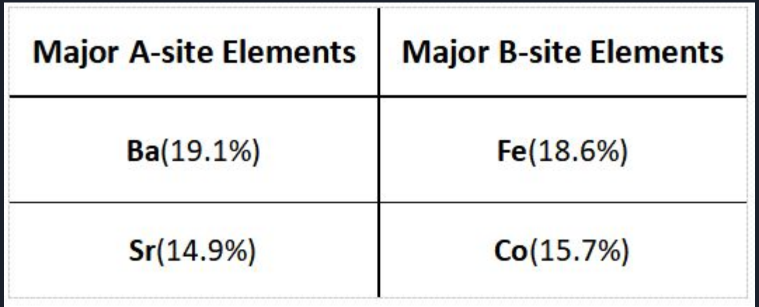
MAE - 20.634
RMSE - 50.59

Classification:

F1 score - 0.904

Intuition of subgrouping



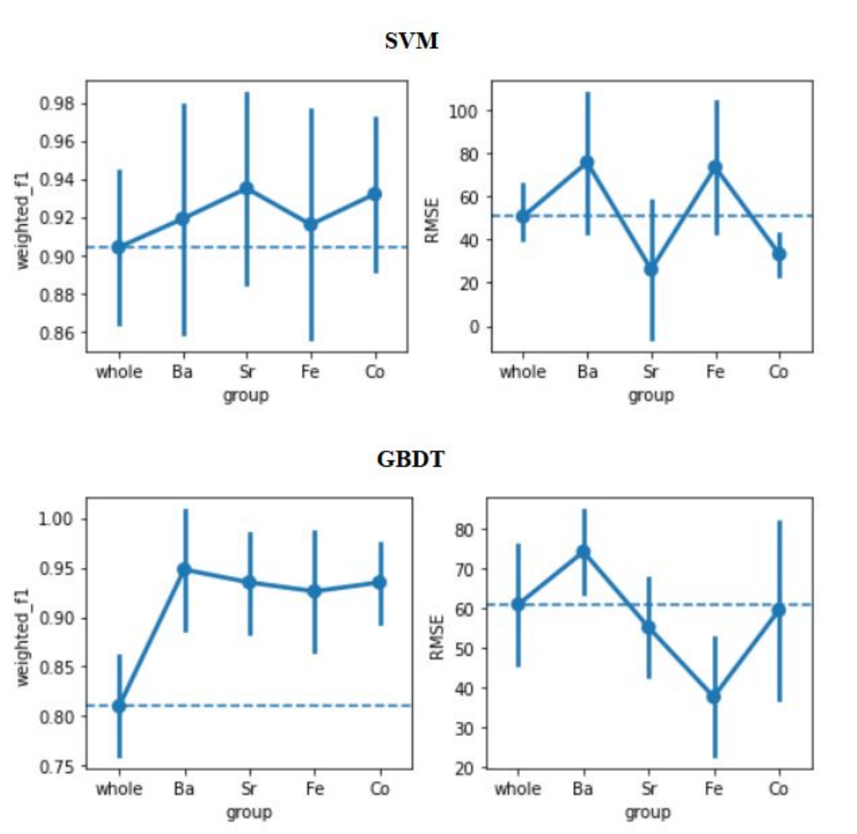


Major A-site Elements	Major B-site Elements
Ba(19.1%)	Fe(18.6%)
Sr(14.9%)	Co(15.7%)

Question:

Is there a relationship between the chemical composition and the thermodynamic stability?

Results of subgroups



Answer:

YES! Major elements play an important role in classification task!

Model Optimization
Part II

Approaches and motivation

Issue:

Initial modeling and data analysis demonstrates that many redundant features are less valuable in helping the models learn

Solution:

Optimize the feature selection pipeline to maximize the expressiveness as well as to minimize the noise of the dataset

Optimized feature selection process overview

- ***Borderline-SMOTE (only for classifier)*** - makes synthetic data along the decision boundary between the two classes.
- ***Variance Threshold*** - eliminates the features with 0 variances.
- ***Select K Best*** - selects features according to the k highest F-value scores
- ***Standard Scaler*** - removes the mean and scales each feature to unit variance.
- ***PCA*** - further reduces the dimensionality of the dataset while still maintaining over 99% of the variance of the test set
- ***Model*** - implements xgb classifier and regressor

Classification results:

- Baseline f1 score: 0.88 ± 0.03
- XGB f1 score: 0.919 ± 0.020

Regression results:

- Baseline MAE/RMSE: 16.7 ± 2.3/28.5 ± 7.5
- XGB MAE/RMSE: 27.484 ± 4.608/47.229 ± 27.223