**Machine Learning for Molecular Engineering**

**Problem Set 2**

**Date:** March 22, 2021
**Due:** 10 pm ET on Thursday, March 18, 2021

**Instructions** Please submit your solution on Canvas as an IPython (Jupyter) Notebook file, `*.ipynb`. We highly suggest using Google Colab for this course and using this template for the first problem set. If you have not used Google Colab, you might find this example notebook helpful. After opening this template link, you should select "Save a copy in Drive" from the File menu. When you've completed the assignment, you can download the `*.ipynb` file from the File menu to upload to Canvas. Your submission should contain *all* of the code that is needed to fully answer the questions in this problem set when executed from top to bottom and should run without errors (unless intentional). Please ensure that your plots are visible in the output file and that you are printing any additional comments or variables as needed. Commenting your code in-line and adding any additional comments in markdown (as "Text cells") will help the grader understand your approach and award partial credit. You are encouraged to ask for clarification on Piazza if you find any of the question statements unclear or ambiguous; chances are, another student feels the same way!

To get started, open your Google Colab or Jupyter notebook starting from the problem set template file pset2.ipynb (direct Colab link). You will need to use the data files here.

# Background

In this P-set you will learn how to use neural networks to predict the properties of crystalline materials of varying composition. The focus will be on **perovskite** materials.

## Perovskite materials

Natural perovskite is a mineral composed of calcium titanium oxide ($CaTiO_3$). It was discovered in Russia by Gustav Rose and named after mineralogist L. A. Perovski. More generally, we called *a* perovskite to any material whose crystalline structure is similar to the perovskite mineral. You don't *need* to know anything about crystal structure to succeed in this P-Set, but we encourage you to learn about crystal structure if you are not familiar with the topic.

Because their structure can support a large number of combinations of elements, perovskites are one of the most abundant structural families and are found in an enormous number of compounds which have wide-ranging properties, applications and importance, such as solar cells, piezoelectrics or catalysts.

### Structure of perovskites

The general chemical formula for perovskite compounds is $ABX_3$, where 'A' and 'B' are two cations. Typically A is large (alkali metal, rare earth or even organic cation such as methylammonium in lead-halid perovskites) and B is typically a transition metal, and X is an anion, typically oxide but also fluoride, nitride or halides that is bound to both cations. The idealized structure of perovskites is cubic (space group $Pm\bar{3}m$), with the B cation in 6-fold coordination and surround anions forming

an octahedron, and the A cation in 12-fold cuboctahedral coordination. However, this ideal structure is often distorted for real materials and shows have reduced symmetry. These distortions, for instance are responsible for the lack of central symmetry in $BaTiO_3$ and its piezoelectric behavior

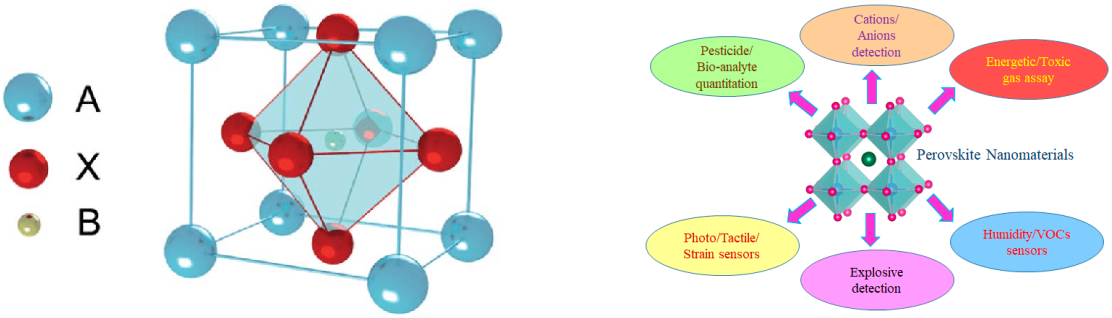
There are many possible A, B and X elements that can be combined in stable perovskite structures. Furthermore, it is possible to combine different choices of elements for A, B and X, which results in an enormous combinatorial space of possible perovskites that can be made and used based on their properties.

### Properties of perovskites

Because their compositions and thus their properties are extremely tunable, pervoskites are exciting materials platforms for many applications.

For example, solar cells based on organic-inorganic halide perovskites (with an organic cation in the A site and lead or a relate elemented in the B site, and Cl, Br or I as X) are the subject of extensive research. Since their discovery about a decade ago, perovskite solar cells have reached efficiency close to that of silicon photovoltaics ( 25%) but are much cheaper to manufacture and can be made into thin films that are flexible and foldable. Although they are a promising platform for widespread affordable photovoltaics, perovskite solar cells suffer from stability issues that prevent mass adoption.

Because of the presence of multiple metal and oxygen sites on their surfaces and the ability of oxygen to participate actively in surface reactions, perovskites are investigated as catalysts that can match the performance of scarce transition metals like ruthenium. Many inorganic perovskites show exciting magnetic and electric phenomena such as photochromic, electrochromic and image storage. Perovskites also have applications for chemical sensing for molecular recognitions.




(a) Perovskite crystal structure

(b) Applications for molecular sensing

Figure 1: Perovskite structure and applications [1]

### Data Generation

The data you will be modeling is generated with high-quality Quantum Chemistry calculations using Density Functional Theory (DFT). Each structure takes hundreds of CPU hours to calculate. They are fairly expensive to obtain. Your task is to train predictive models to help theorists to reduce their computational cost.

**Python packages required for this exercise**

In this p-set, you will asked to use Pandas data-frames. If you have not used Pandas before, please work through this tutorial. You will also be asked to encode chemical formulae into numpy.arrays. If you are not familiar with `numpy`, please also work through this tutorial.

# Part 1: Modeling unalloyed perovskites

In this question, you are asked to predict $E_{hull}$ (Energy above hull). It has the unit of eV/Atom. It describes the thermodynamic stability of the crystal structure. If $E_{hull} = 0.0$, it means the structure is at its most stable form.

**Energy Above Hull** The energy of decomposition of this material into the set of most stable materials at this chemical composition, in eV/atom. Stability is tested against all potential chemical combinations that result in the materials composition. For example, a $Co_2O_3$ structure (cobalt oxide) would be tested for decomposition against other $Co_2O_3$ structures, against Co and $O_2$ mixtures, and against CoO and O2 mixtures. $E_{hull}$ is computed as :

$$E_{\mathrm{Co_2O_3}} - \mathbf{min}(2E_{\mathrm{Co}} + \frac{3}{2}E_{\mathrm{O_2}}, 2E_{\mathrm{CoO}} + E_{\mathrm{O_2}}, E_{\mathrm{Co_2O_3}})$$

Different from pset 1, you will be performing regression tasks. For regression tasks, a metric we can use to evaluate model performance is the coefficient of determination, or the $R^2$ score. It is defined as:

$$R^2 = 1 - \frac{\sum_i^{N_{\mathrm{data}}}(y_i - \hat{f}(\mathbf{x}_i))^2}{\sum_i^{N_{\mathrm{data}}}(y_i - \mathbf{mean}(y_i))} \tag{1}$$

where $i$ is the index for each sample, $y_i$ is the target value, $\hat{f}$ is the model, and $\mathbf{x}_i$ is the feature vector. The larger the $R^2$ score the more accurate the prediction is. If your prediction is perfect $R^2 = 1$. Note that metrics like mean absolute error (MAE) or mean squared error (MSE) can provide more meaningful and interpretable measures of performance.

You will first need to load the dataset using the code provided.

## Part 1.1 (10 points) Encoding chemical formulae into feature vectors

After loading the pandas dataframe, take a moment to inspect your data by looking at the rows (samples) and columns (features) . The important chemical information is stored in columns [‘A’, ‘B’, ‘X’].

First, you will train a model that uses chemical compositions (e.g. $CaTiO_3$) to predict Energy above Hull($E_{hull}$) which is a scalar property. How can you map chemical formula like $CaTiO_3$ to numeric features for training a model? You need to define a dictionary to encode elements into numerical features so that we can apply programs like linear regressions on these input features. One way to do is to use one-hot encoding to transform elements into bit vectors. For example, if we have elements Fe, Ni, and Mn for A and B sites, we can assign bit vectors of size 3 to fully encode this label information. We do not encode the elements for the X site because Oxygen is the only element available in our dataset for X. We present simple examples on how you should encode perovskite chemistries in table 1.

| A site | B site | bit vector representation |
|---|---|---|
| 'Fe' | 'Ni' | $[1, 0, 0, 0, 1, 0]$ |
| 'Mn' | 'Ni' | $[0, 0, 1, 0, 1, 0]$ |
| 'Fe' | 'Mn' | $[1, 0, 0, 0, 0, 1]$ |
| 'Fe' | 'Fe' | $[1, 0, 0, 1, 0, 0]$ |

Table 1: Example bit vector representation of A/B sites in perovskites

Use `preprocessing.LabelBinarizer` to transform features ['A', 'B'] into one-hot encoded arrays. We provided the choices of elements for A/B sites in `elements.npy` which can be loaded using the code we provided to you. These elements are the "vocabularies" you need to use to construct your bit vectors.

Before you start, please read the documentation about `preprocessing.LabelBinarizer()` and understand the example code in the documentation. Your code should output a featurized numpy array.

**Hint:** There are many ways to do it. One way to do it is to loop over rows (perovskites) in your dataframe and encode A and B into bit vectors separately, and then stack the two arrays horizontally using `np.hstack`.

### Part 1.2 (10 points) Modeling with Ridge Regression

After obtaining bit vector representations for the perovskite composition, train a Ridge regressor (linear regression with L2 regularization) to predict $E_{hull}$ (column name: `e_above_hull`). Randomly split your data into train/test with an 80%/20% ratio. Report your test score. Plot your prediction using a scatter plot (`matplotlib.pyplot.scatter`) for both train and test data.

Please first read the documentation on `sklearn.linear_model.Ridge` before training the model. We have also provided a code snippet for creating a scatter plot.

### Part 1.3 (10 points) Modeling with a multi-layer perceptron

Using the same train/test split, train a model to predict $E_{hull}$ using a multi-layer perceptron with the following parameters in table 2. Visualize your prediction for train and test data with a scatter plot. Calculate the number of parameters used in your MLP given the provided '`hidden_layer_sizes`'. Briefly comment on the meaning of (256, 256, 256) which is the input you will use for the `hidden_layers_sizes` optio.

Please first read the documentation on `sklearn.neural_network.MLPRegressor` and understand the meaning of each hyperparameter before training the model.

| | |
|---|---|
| 'hidden_layer_sizes' | (256, 256, 256) |
| 'activation' | 'relu' |
| 'alpha' | 0.16 |
| 'solver' | 'adam' |
| 'early_stopping' | True |

Table 2: hyperparameters to be used in part Part 1.3

### Part 1.4 (5 points) Chemical transferability of one-hot representations

We prepared a special validation set for you to test your model performance. This validation set contains elements that do not exist in the training set. Load the validation set using the code provided. Featurize the data with label encoder defined in part Part 1:. Validate your trained Ridge regressor and MLP by computing the $R^2$ score for both of them. Visualize your prediction with a scatter plot for the two models you trained. Briefly describe explain what you observed. Do either of these models generalize well to this new data?

### Part 1.5 (10 points) Featurize perovskites with physical descriptors

Now load the data for atomic properties stored in `mendeleev.csv` with the code we provided. Use all the numerical features to construct a new feature set for prediction (please exclude atomic symbols which are strings). You need to first scale the data with `preprocessing.StandardScalar` you have used in p-set1 and split the data in to train(80%) and test(20%) set. Train on the training set with a MLP with the same hyperparameters we provided to you in part Part 1.3. Visualize your prediction with scatter plots for both train and test data.

### Part 1.6 (5 points) Chemical transferability of physical descriptors

Report the $R^2$ score on your validation set using the model trained in part Part 1.5 and visualize your prediction with a scatter plot like you did before. Has your prediction on the validation set improved? Briefly explain why.

## Part 2: Modeling alloyed (hybrid) perovskites

In this part, you will be asked to model alloyed perovkites with MLPs for both classification and regression tasks. Alloying simply means mixing two or more types of crystal materials together, this sometimes results superior properties of materials. See figure 2 for a visual understanding. For example, Steel is made by alloying Iron (Fe) with other elements and has superior properties than pure Iron.

As the result of alloying perovskites, the A/B sites will have more than one elements of different fractions. For example, in the dataframe, you will see the element compositions at A/B sites are represented by a dictionary like: `{'Fe': 0.15, 'Mn': 0.05}` . where the keys represent possible elements at the site and the values represent the composition fraction.

Please load a new dataset with the code we provided.

### Part 2.1 (10 points) Encoding fractional compositional information

In this question, you will be asked to write code to encode both chemical and composition information for A/B sites. Similar to what you developed in part Part 1:, you can use the composition fraction to encode the chemical information. For example, if both Fe and Mn present at site A with fraction 0.15, and 0.05 respectively, instead of representing its composition with `[1, 0, 1]`, replace 1 withe the actual compositions which are provided as values in the dictionary. As a result, you will get `[0.15, 0, 0.05]`. Encode A and B site separately and concatenate them into one bit vector (table 3).

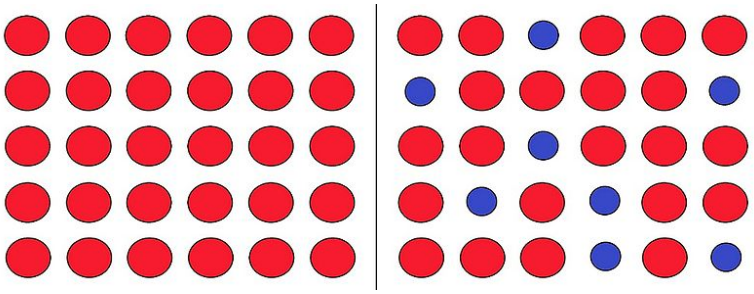


Figure 2: Alloying simply means mixing elements together. By allowing mixed systems, we have more possibilities for different structures which may exhibit more beneficial or tunable properties.

| A site | B site | bit vector representation |
|---|---|---|
| {'Fe': 0.15, 'Ni': 0.05} | {'Mn': 0.2} | $[0.15, 0.05, 0, 0, 0, 0.2]$ |
| {'Fe': 0.15, 'Mn': 0.05} | {'Fe': 0.1, 'Mn': 0.1} | $[0.15, 0, 0.05, 0.1, 0, 0.1]$ |

Table 3: Example bit vector representation of A/B sites with compositional information

Develop the code to encode all the alloyed perovskites into a new feature set `X_alloy` with compositional information. Split your data into train (80%) and test (20%) set. Fit an MLP to predict `e_above_hull` on the training data with the hyperparameters listed in table 4. Report your $R^2$ scores for both train and test data. Visualize your prediction for both train and test data with a scatter plot.

| | |
|---|---|
| 'hidden_layer_sizes' | (50) |
| 'activation' | 'tanh' |
| 'alpha' | 0.0001 |
| 'solver' | 'adam' |
| 'early_stopping' | False |

Table 4: Hyperparameters to be used in part Part 2.1

### Part 2.2 (20 points) Optimize your MLP architecture with HyperOpt

**??**

In Pset 1, you used grid search for model hyperparameters. However, grid search in many cases is not tractable due to combinatorial explosion. In practice, there exist more efficient algorithms to improve exploration efficiency. Tree of Parzen Estimators (TPE) is such a more advanced search method based on sequential model-based optimization. The details of TPE algorithm are beyond the scope of this class, but you are encouraged to read more about it in ref. [2]. TPE has been implemented in the `Hyperopt` package. In this part, you will try to use `Hyperopt` to explore the space of hyperparamters for MLP.

We have provided the skeleton code for you with predefined ranges of hyperparameters in the template file, but you will have to write your own `model_eval()` function which should take a suggested hyperparameter set and return the resulting 5-fold cross-validation (CV) score on the training data you obtained from part Part 2.1.

Report the best CV score and the optimized model hyperparameters. Run the program for 40

evaluations. The hyperopt program might take a while to run, and you can check the progress bar to monitor the program executions. You can use this time to make yourself a cup of coffee or do HIIT. Finally, used the optimized hyperparameter set to train a MLP with all the training data again, and report your validation score on the test data you obtained from part Part 2.1. Have you seen any improvement?

**Caveats** What Hyperopt will do is to optimize your hyperparameters given the range you provide and minimize your loss function using `hyperopt.fmin`. Please note that because a higher $R^2$ indicate better performance, you should minimize the negative $R^2$ score with `fmin` (We have implemented this for you). Also, to constrain the complexity of the search space, we fix the hidden layer width for all the layers during the search. If you have more than one hidden layer, all layers will have the same width. However, you could have different hidden layer widths for each layer in practice.

### Part 2.3 (15 points) Applying MLPs to classifying electronic properties of alloyed perovskites

In this part, you will be making a multi-category classifier to classify different types of perovskites based on their electronic structures which characterizes their energy levels of electrons in crystals.

**Electronic Structure** electronic band structure (or simply band structure) of a solid describes the range of energy levels that electrons may have within it, as well as the ranges of energy that they may not have (called band gaps or forbidden bands).

These types include *direct bandgap materials*, *indirect bandgap materials*, and *metals*. These three types of materials have different electronic properties. Materials that do not have bandgaps are metals and conduct electrons. For materials with bandgaps, they can either be insulating or semi-conducting. A bandgap is the energy difference between the conduction band and valence band. Whether the minima of the conduction bands and the maxima of the covalent band are aligned affects the process of electronic excitation, and therefore affecting its optical properties like absorption. If they are aligned, it is a direct bandgap material; if they do not align, it is a indirect bandgap material. See figure 3 to have a visual understanding. In general, direct bandgap materials have a wider range of electronic and optical application in industry. You are asked to make a MLP-based classifier to classify the three classes of electronic materials mentioned.

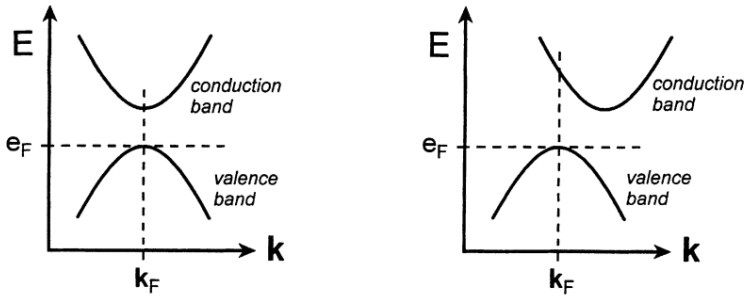



Figure 3: Graphical description for direct/indirect bands (from [3])

The dataset you will use is the same as what you loaded in part Part 2.1. Please also use the same encoded features you developed in in part Part 2.1 and use the same MLP architecture you obtained from hyperopt in Part 2.1. The class labels are in the "class" column in your loaded

dataframe. Randomly split your data into 80% train and 20% test data and train a classifier on your training data.

Please use the `metrics.plot_confusion_matrix` function to visualize the classification result for both train and test data. Different from the binary classification task from p-set 1, there are three classes to predict so your confusion matrix will have a size of 3x3.

Before you start, please read documentations on `sklearn.neural_network.MLPClassifier` to understand the examples.

# Part 3: Conclusion

## Part 3.1 (5 points) Feedback survey

You will get 5 points for completing the survey posted on Canvas. This will help us improve future problem sets.

**Acknowledgement**

We thank Daniel Schawbe-Koda for helping with the data generation for this problem set.

# References

[1] Chen, Y., Zhang, L., Zhang, Y., Gao, H. & Yan, H. Large-area perovskite solar cells–a review of recent progress and issues. *RSC advances* **8**, 10489–10508 (2018).

[2] Bergstra, J., Bardenet, R., Bengio, Y. & Kégl, B. Algorithms for hyper-parameter optimization. In *25th annual conference on neural information processing systems (NeurIPS 2011)*, vol. 24 (Neural Information Processing Systems Foundation, 2011).

[3] Abedin, M. I., Islam, A. & Hossain, Q. D. A self-adjusting lin-log active pixel for wide dynamic range cmos image sensor. In *2015 IEEE International Conference on Telecommunications and Photonics (ICTP)*, 1–4 (IEEE, 2015).