Prediction of Inorganic Crystals' Thermodynamic Stability Using Machine Learning

Sara Nabili

July 31, 2025

Outline

- Intro
  - General Info
  - EBH
  - DFT Limitation
  - Future Direction
- Methodology
  - Energy/Electronic Features
  - Structure and Composition
  - Bond Structure Features
  - Features ROC Curve
  - Correlation Matrix
  - Permutation Feature Importance
- Machine Learning Workflow
  - ML Methods
  - Normalizing Feature
  - Class Weight
  - ML hyperparameter tuning
    - LR Hyperparameters
    - RF Hyper Parameters

Introduction

Purpose: This project uses a *supervised* machine learning (ML) **binary classification** algorithm to predict the thermodynamic stability of inorganic crystals based on various features.

Motivation: In materials science, predicting compound stability is costly and slow. This project aims to accelerate discovery using data-driven models.

Tools/Frameworks: Python, scikit-learn, TensorFlow, XGBoost, pymatgen, data-analysis toolkits, visualization packages, parallel and multiprocessing computing frameworks

Implementation: Feature extracted using pymatgen, engineered descriptors like bond statistics and atomic fractions, and trained ML algorithms such as a deep neural network.

Final Result: The final model achieved 87% validation accuracy, showing promising results for early-stage stability screening.
Application: This tool can be used for material engineering, pharmaceutical, and sustainable energy technologies

General Information

Problem Context: Thermodynamic stability (under a set of conditions) is achieved if the formation energy of the compound can not be lowered by rearranging its atoms [1].

Energy lowering mechanism:

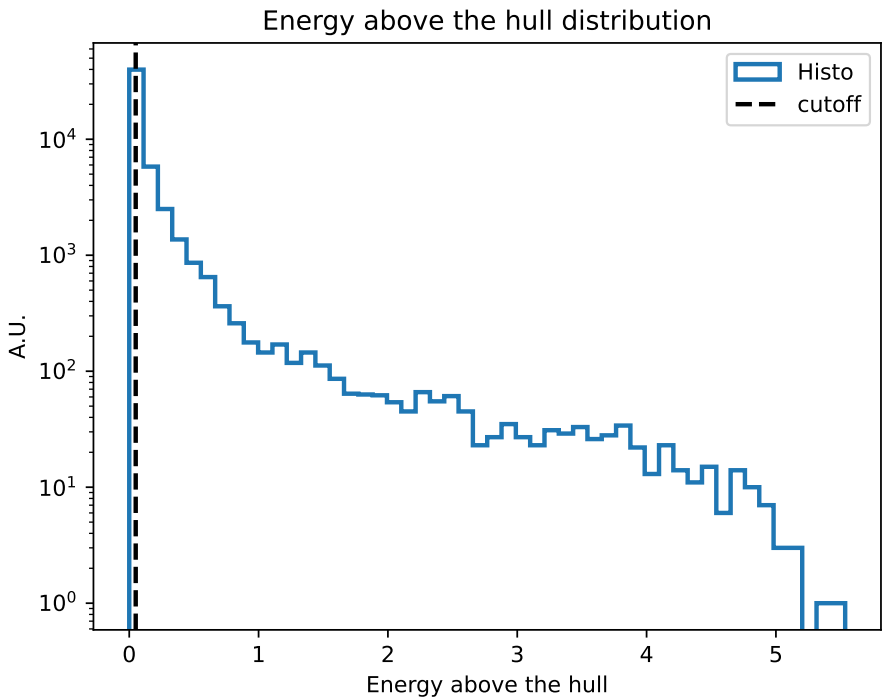
- *Decomposition:* Phase separation into competing materials that share an identical overall composition
- *Polymorphism:* Phase transition into an alternative crystal structure (polymorph) at fixed composition.

Thermodynamic Stability:

- A convex hull represents the lowest formation energy envelope derived from inorganic crystalline materials plotted against their composition.
- *The energy above the hull (EBH):* The energy difference between a compound and the convex hull with the same composition.

Energy above the hull

- The ML algorithms are trained to predict the thermodynamic stability of chemical compounds by the binary classification of the energy above the hull feature (EBH) [1]. The EBH distribution is shown below.
- In this study, thermodynamically stable to slightly metastable materials are classified by applying a threshold of $\mathrm{EBH} \leq 0.05$ as suggested by [4].
- Out of nearly 50,000 materials in the dataset, roughly half meet the criteria for thermodynamic stability, while the other half fall above the defined threshold and are considered unstable.



Limitations of DFT & the Rationale for Machine Learning

Limitations of Density Functional Theory (DFT):

- Highly accurate, but computationally intensive, especially for large or low symmetry structures.
- High-throughput screening of new compounds is limited by cost and runtime.
- Not suited for rapid prototyping or exploration of vast chemical spaces.

How Machine Learning Addresses These Bottlenecks:

- Learns structure property relationships from curated datasets.
- Predicts material stability orders of magnitude faster than DFT
- Enables scalable screening of millions of candidate materials

Study Limitations and Future Directions

Limitations:

- **Dataset Scale:** Access to the Materials Project database was limited to 50,000 compounds, constraining the deep neural network's generalization capacity.
- **Computational Resources:** Experiments were conducted on a single machine with 8 CPU cores, limiting parallelization and throughput.

Future Directions:

- Expanding access to larger datasets can significantly enhance model performance and diversity.
- Deploying high-performance clusters and distributed computing frameworks (e.g., HTCondor) will support advanced feature engineering and more complex ML architectures.

Data Collection

Data source: Data extracted from the Materials Project [1], an open database of DFT-computed material properties.

Feature extraction: The features are classified into four classes:

- **Energy & electronic features:** Helps capture energetic contribution
- **Structural and composition:** Gives access to size, packaging, and complexity
- **Bond features:** Facilitates the detection and quantification of inter-element bonding, while representing the surrounding chemical context
- **Atomic fraction:** Reflects elemental influence on stability

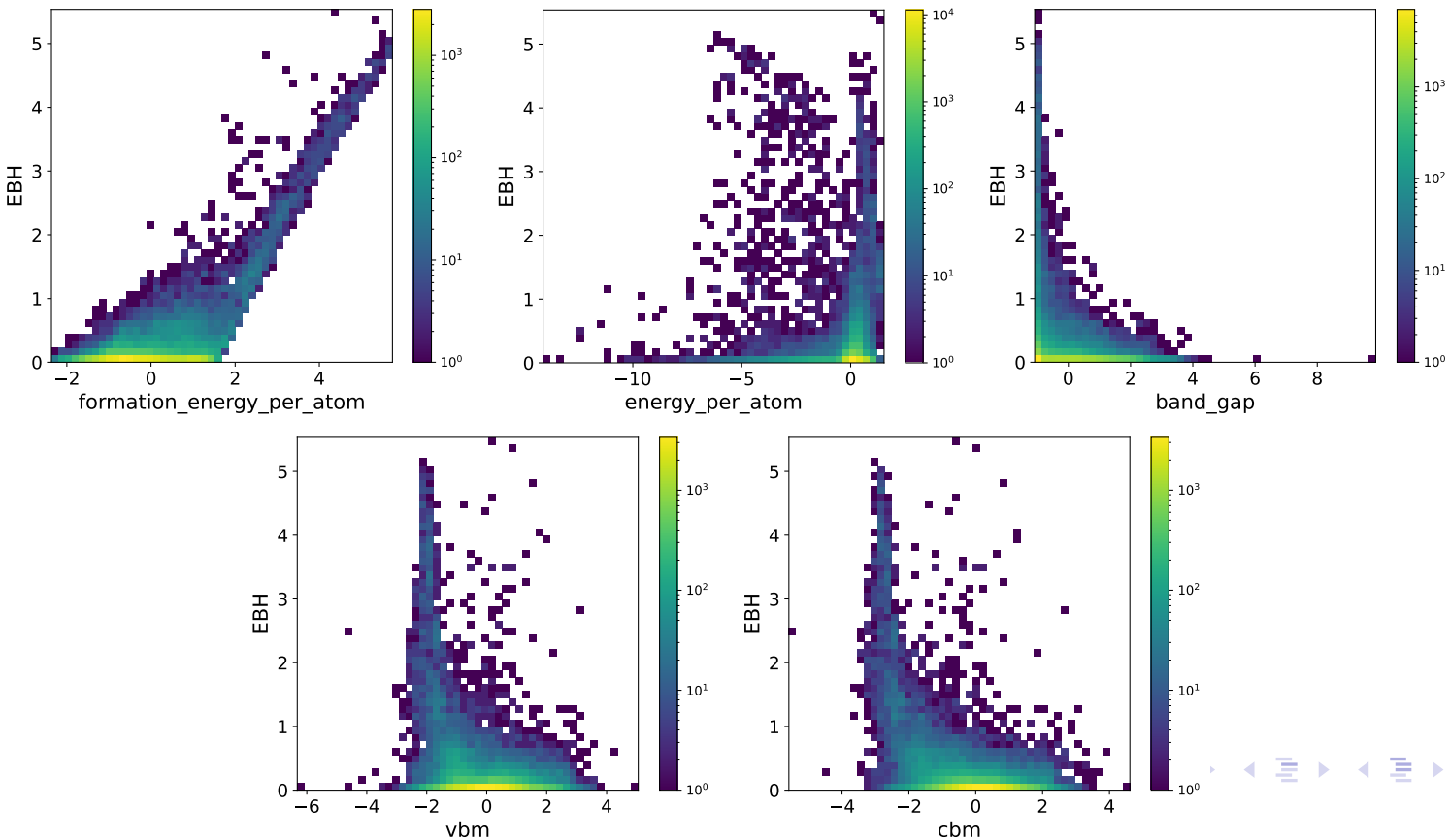
The first two classes are database queries, whereas the rest need feature engineering.

Feature Engineering To build:

- Atomic fractions: Computed element-wise from compositions.
- Bond structure statistics: Using neighborhood-finding algorithms, MinimumDistanceNN [8]:
  - Number of bonds
  - Mean bond length
  - Bond length standard deviation

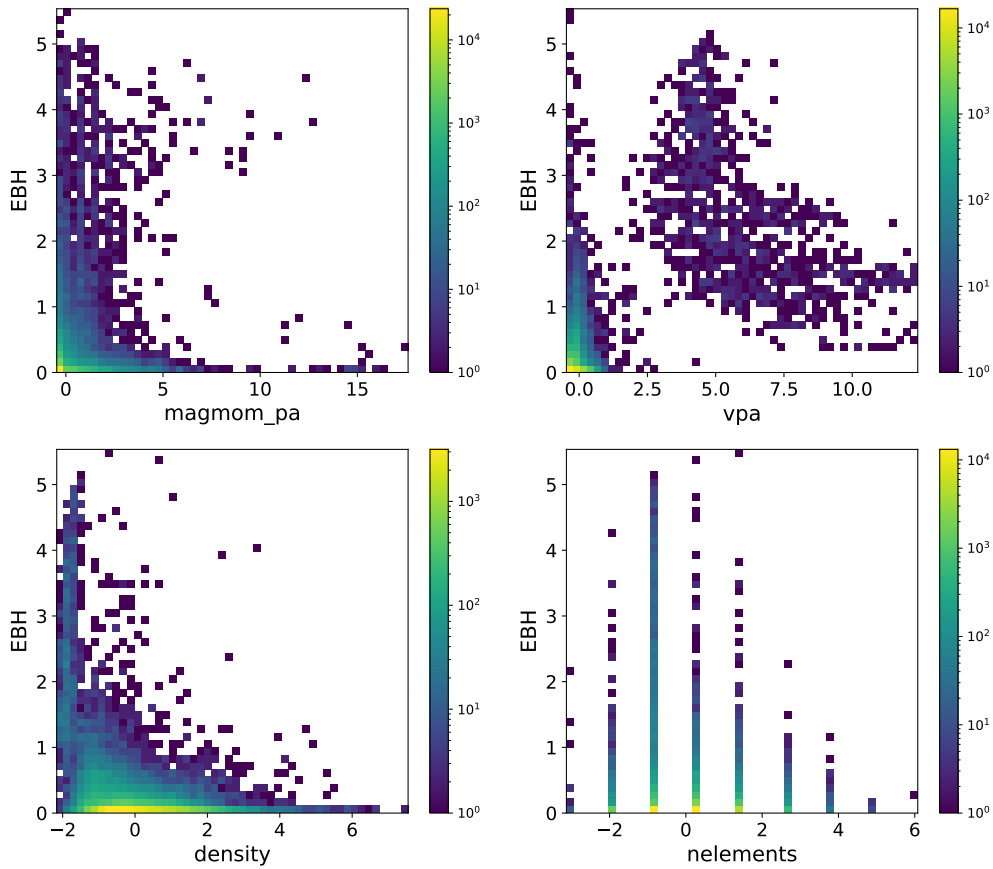
Feature: Energy & Electronic

- **Formation energy per atom:** Energy change as a compound is formed from its constituent elements in their standard states
- **Energy per atom:** include element reference in formation energy per atom
- **Band gap:** related to chemical bonding and electronic stability
- **Valence/conduction band edges:** explores patterns in electronic structure



Feature: Structure and Composition

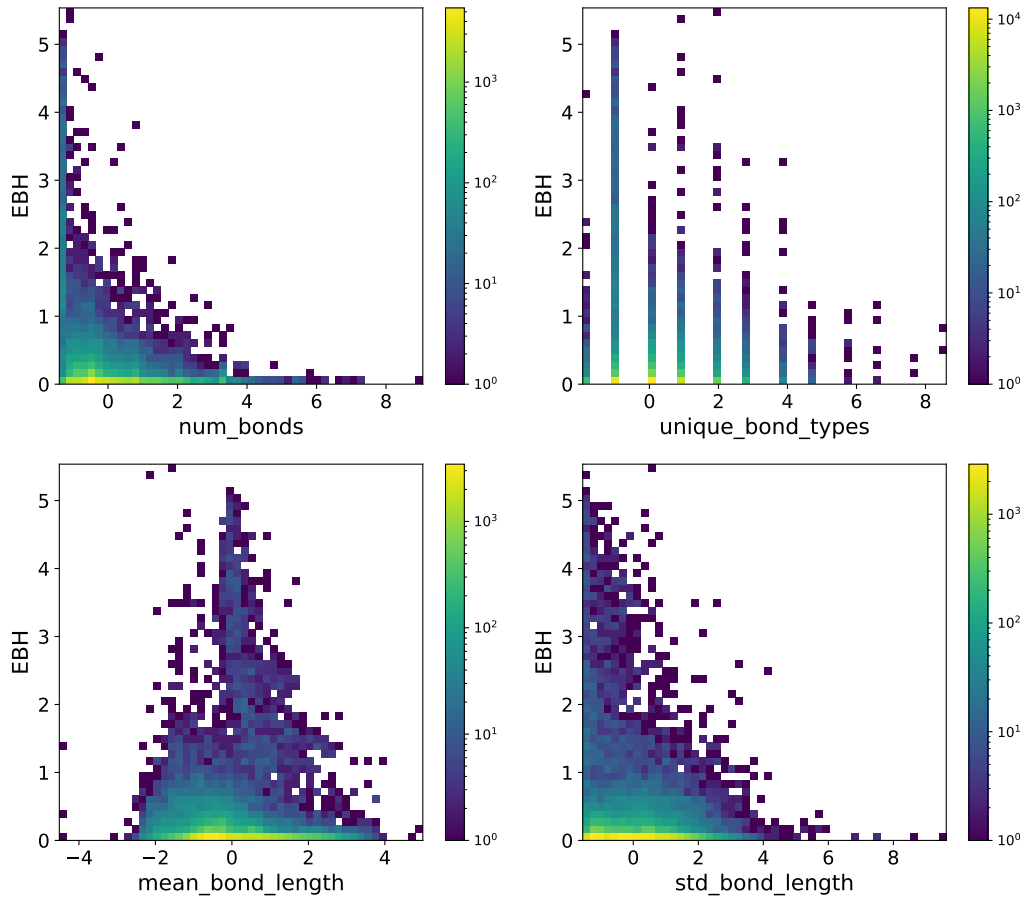
- Magnetic moment per atom: Depending on the material, some stable phases are magnetic
- Volume per atom: global structure feature
- Density: affected by atomic mass and volume
- Number of elements and sites: Capture compound's complexity; too many elements and sites may lead to less stable/metastable phases. Number of sites is considered in magnetic moment and volume per atom computation (see backup slide).



Feature: Bond structure

Captures bond's (min/avg/max) distance and coordination using MinimumDistanceNN [8]. Features are:

- Number of bonds: coordination and connectivity
- Unique bond type: structural diversity and bonding motifs
- Bond length mean and standard deviation: average bonding length and geometrical distortion

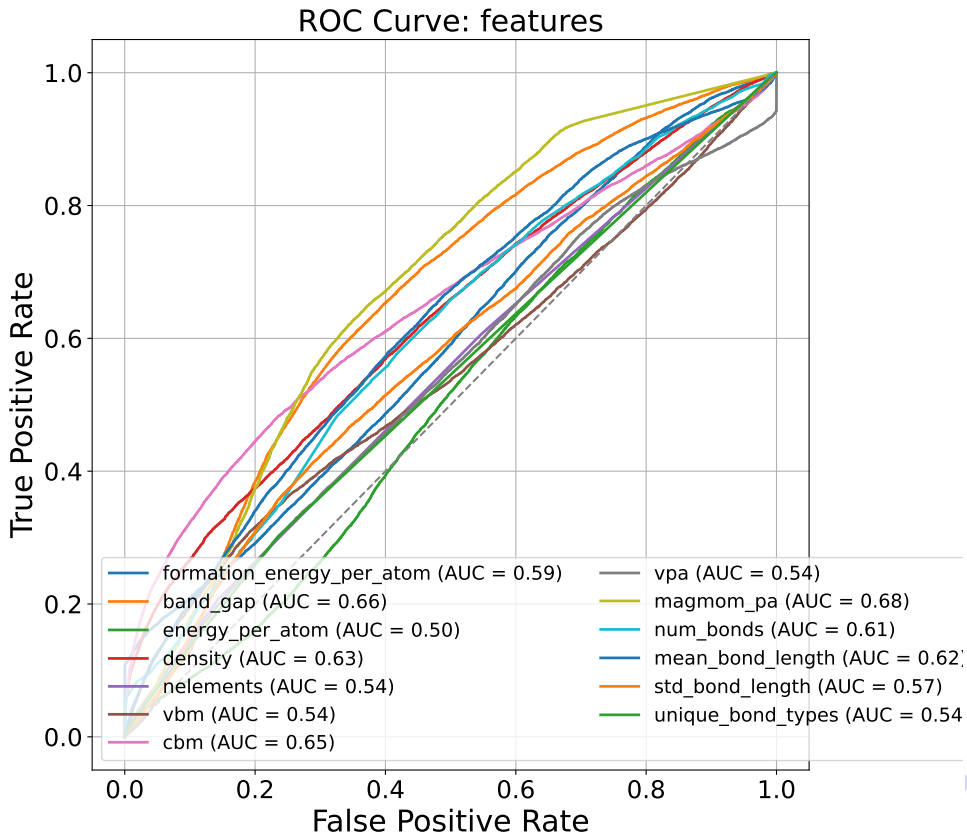


ReceiverOperatingCharacteristic Curve

Receiver Operating Characteristic (ROC) Curve: A graphical tool for evaluating the classification performance of models or features across varying decision thresholds.

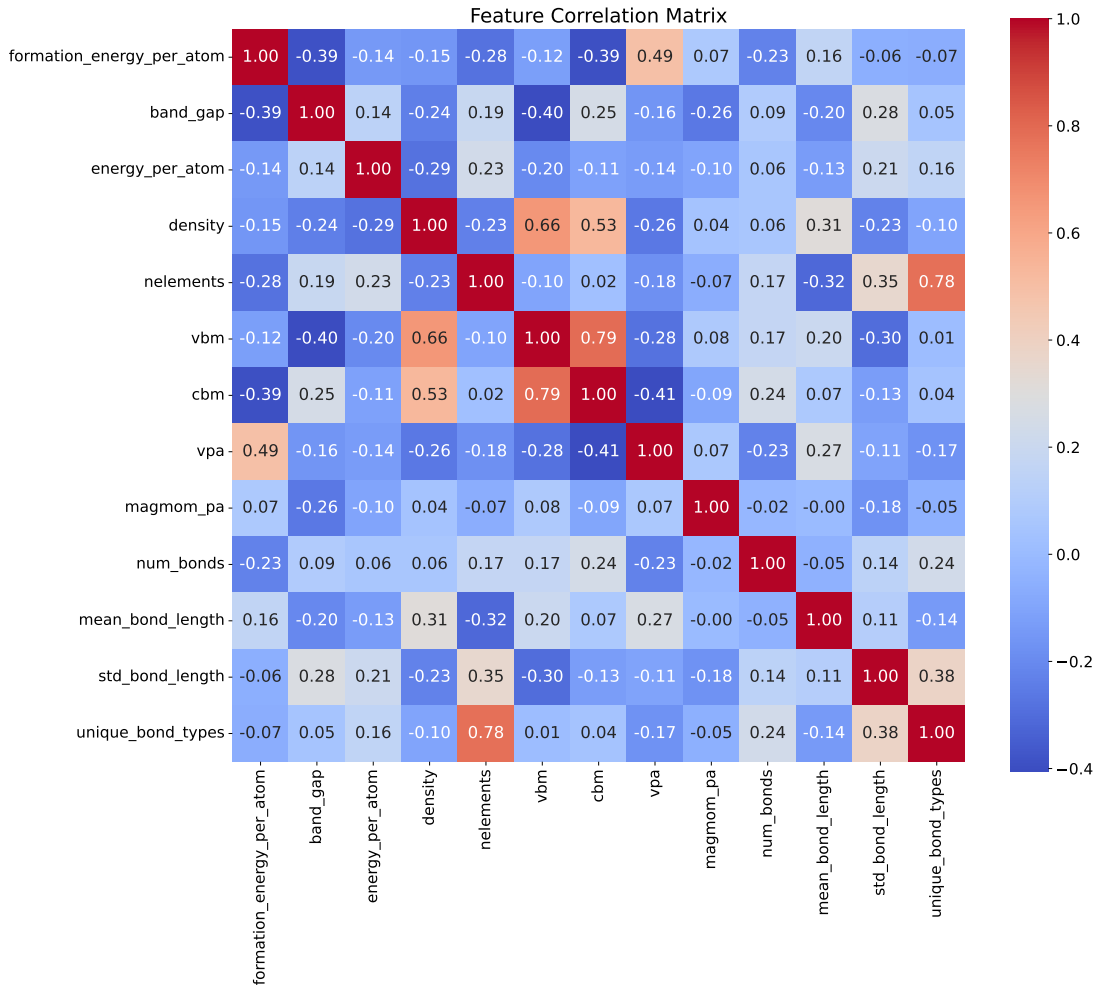
AreaUnderCurve (AUC): A quantitative summary of overall discriminatory power—higher values indicate better separation between classes.

ROC curves: Selected input features, with respective AUC in the legend. The relatively low AUC scores indicate the features' limited ability to differentiate between stable and unstable chemical compounds.



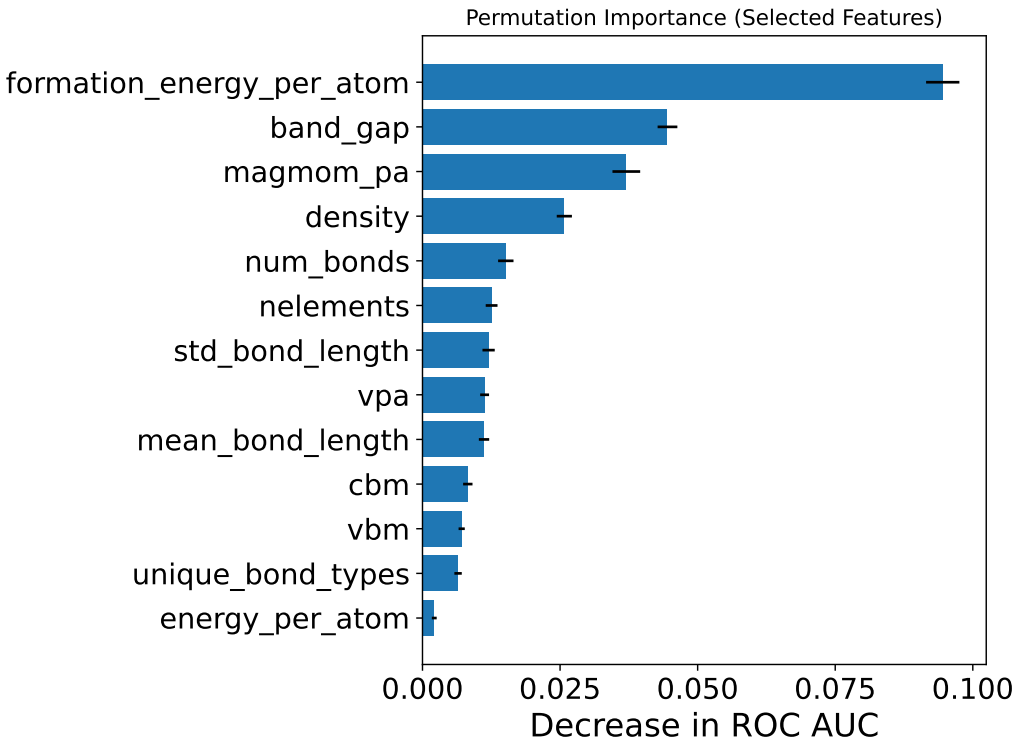
Correlation Matrix

XGBoost [2] was used to analyze relationships among the ML input features. Only the general features' pairwise correlations are visualized as a correlation matrix to avoid confusion in the plot.

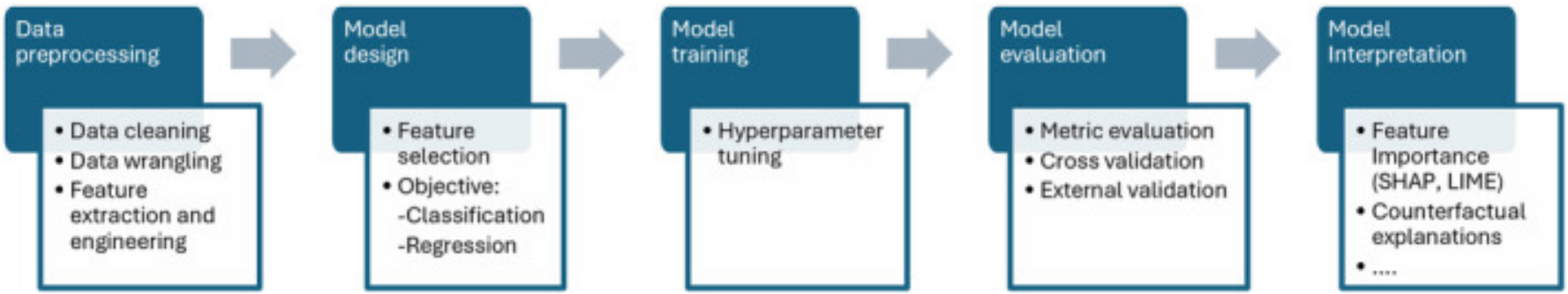


Permutation Feature Importance

- Explains the contribution of a feature to the model accuracy. XGBoost built-in feature importance function represents the overall usage of a feature by the tree nodes
- Used XGBoost to compute, with metric set to area under the ROC curve. Feature importance is assessed 20 times by shuffling its values and checking how the model's performance changes.
- A handful of selection features are used to make the permutation importance plot, the plot shows the mean value of multiple evaluations and the standard deviation of its evaluation as the error bars.



Machine Learning Workflow



Refernce: [9]

Machine Learning Methods

A binary classification was used to predict the thermodynamic stability based on machine learning methods.

- **Deep Neural Network (DNN):** This work focuses on assessing the effectiveness and potential of DNN in predicting materials properties because of its:
  - Flexible architecture
  - Strong capacity for capturing complex, nonlinear relationships in the data
- To validate and benchmark DNN performance, two additional models were employed:
  - **Logistic Regression (LR)**: A fast, interpretable baseline model for comparison with DNN results.
  - **Random Forest (RF)**: A tree-based ensemble model chosen for its robustness on small datasets and its ability to provide insight into feature importance.

Preprossessing Features

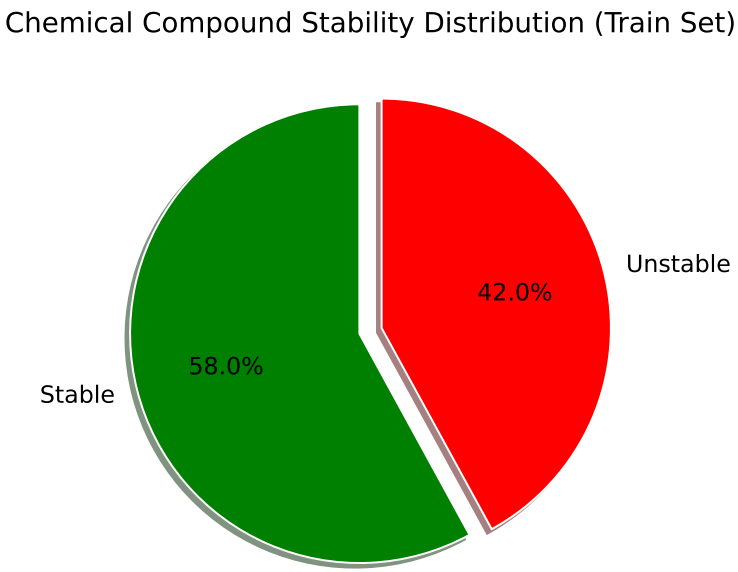
Feature Normalization: Help the model train more efficiently and accurately:

- Used `StandardScaler` from `scikit-learn`.
- Plays a crucial role during pre-processing features by ensuring each input feature has $\mu = 0$ and $\sigma = 1$.
- In **DNN** improves:
  - Gradient descent functionality as features are on similar scales
  - Avoid gradient fluctuation in deep layers
  - Works well with activation functions that are sensitive to input scale like sigmoid
- Helps stabilize DNN, Logistic Regression, and Random Forest model learning.

Class Weight

The dataset is moderately imbalanced, with a higher proportion of stable/metastable compounds. To mitigate this, class-level weight is used:

- To address class imbalance, uniform weights are assigned using `class_weight='balanced'` from `scikit-learn`.
- it automatically adjusts weights inversely proportional to class frequencies, encouraging the model to pay more attention to the minority class.
- Applied to all three ML methods.



ML Methods Hyperparameter Tuning

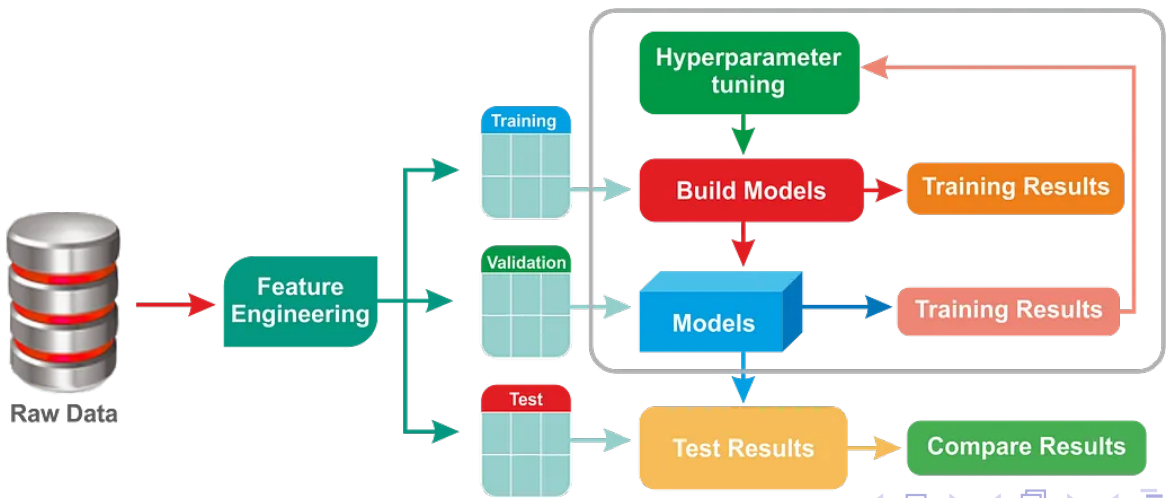
Goal: Find the best combination of ML model parameters that optimizes its performance

Hyperparameters: Settings chosen before training, crucial to maximize model accuracy and effectiveness.

Tuning Method:

- Use `GridSearchCV` from `scikit-learn`
- Performs exhaustive search over predefined parameter values
- 3-fold cross-validation (`cv=3`) used to reduce overfitting

Optimization criteria: *Accuracy* (overall correctness), *F1 score* (balance between precision and recall), and *the area under the ROC curve* (model's power to distinguish among classes).

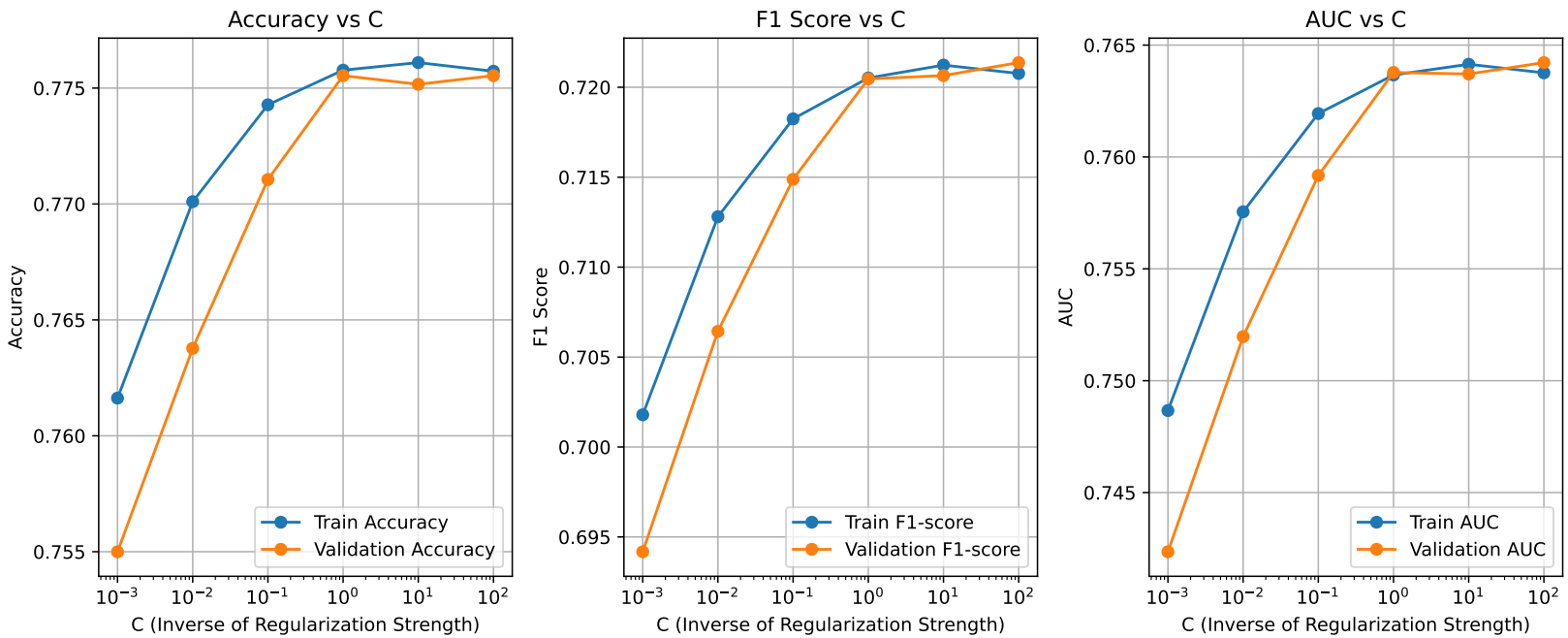


LR Hyperparameters

- **LogisticRegression:** Hyperparameters used in tuning LR:

Hyperparameter	Values	Description
`C`	[0.001, 0.01, 0.1, **1**, 10, 100]	Inverse of regularization strength
`penalty`	l1, **l2**	Type regularization used to prevent overfitting
`max_iter`	[200, **500**, 1000]	Max iterations to converge

(The bold, red values are the hypertuned parameters)

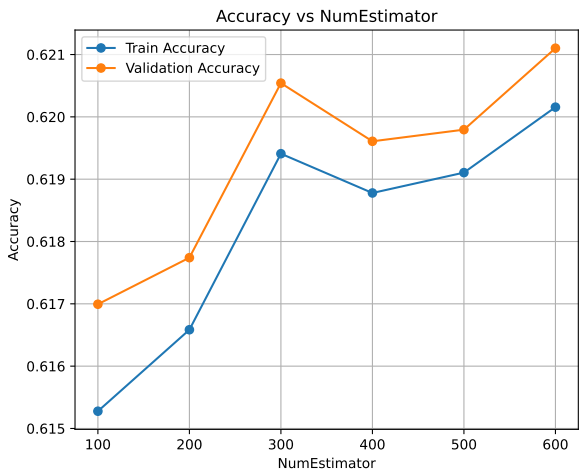


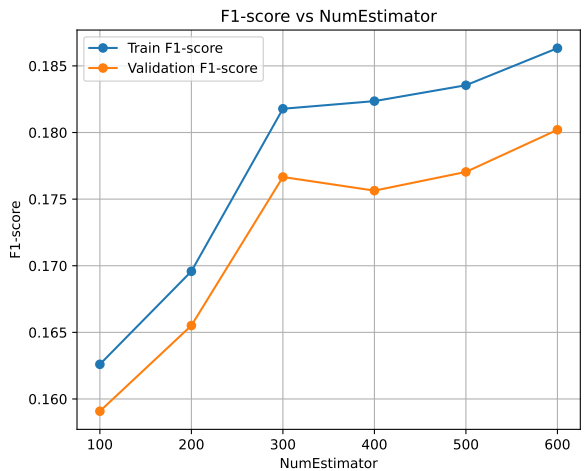
RF Hyper Parameters

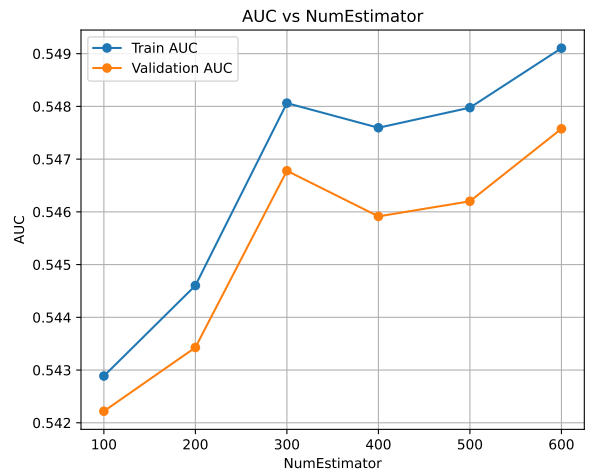
- **RandomForest:** Hyperparameters used in tuning RF:

Hyperparameter	Values	Description
`n_estimator`	[100, **200**, 300, 400]	Number of decision trees
`max_depth`	[**5**, 10, 20]	Number of splits/decision levels from the root node to the deepest leaf node
`min_samplsplit`	[**2**, 5]	Minimum number of samples required to split an internal node

(The bold, red values are the hypertuned parameters.)







DeepNeuralNet

- *Tensorflow* is used to build the neural network
- Features are selected after their importance passes a certain threshold (slide number 14)
- **Activation Function:** Rectified Linear Unit (ReLU) function (ReLu(x) = max(0,x)) for input and hidden layers, and sigmoid function for the output layer.
- Prevent overfitting during training:
  - **Random dropout of neurons:** 10% for the input layer and 30% for the hidden layers;
  - **Batch normalization:** Stabilize learning by normalizing input values per layer;
  - **Early stopping:** Prevent validation loss increment;
- **Learning rate:** Optimizes model weights in response to the estimated error (loss)

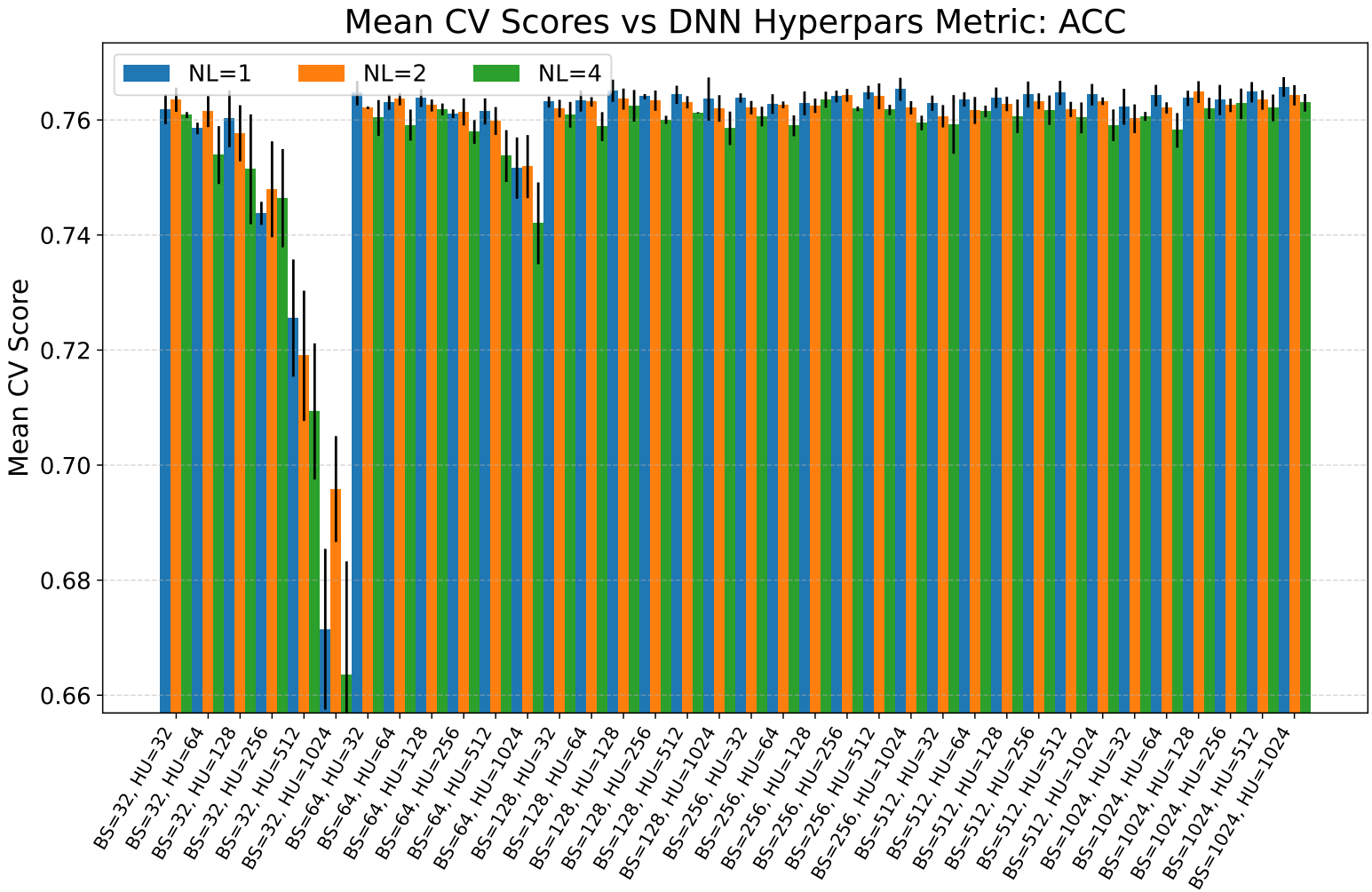
DNN Parameters

DNN Model Parameters:

- Fixed parameters:
  - **Architectural regularization:** Including batch normalization, dropout, and early stopping
  - **Learning rate:** Used with learning rate scheduler to lower the learning rate when the validation loss plateaus
  - **Activation Function**
  - **Epochs**: Number of complete passes of the entire training dataset through a learning algorithm. To avoid overfitting, `EarlyStopping` callback tool from `keras` was used.
- Tune Hyperparameters to optimize DNN performance
  - Number of hidden layers
  - Batch size
  - Number of hidden units (neurons)

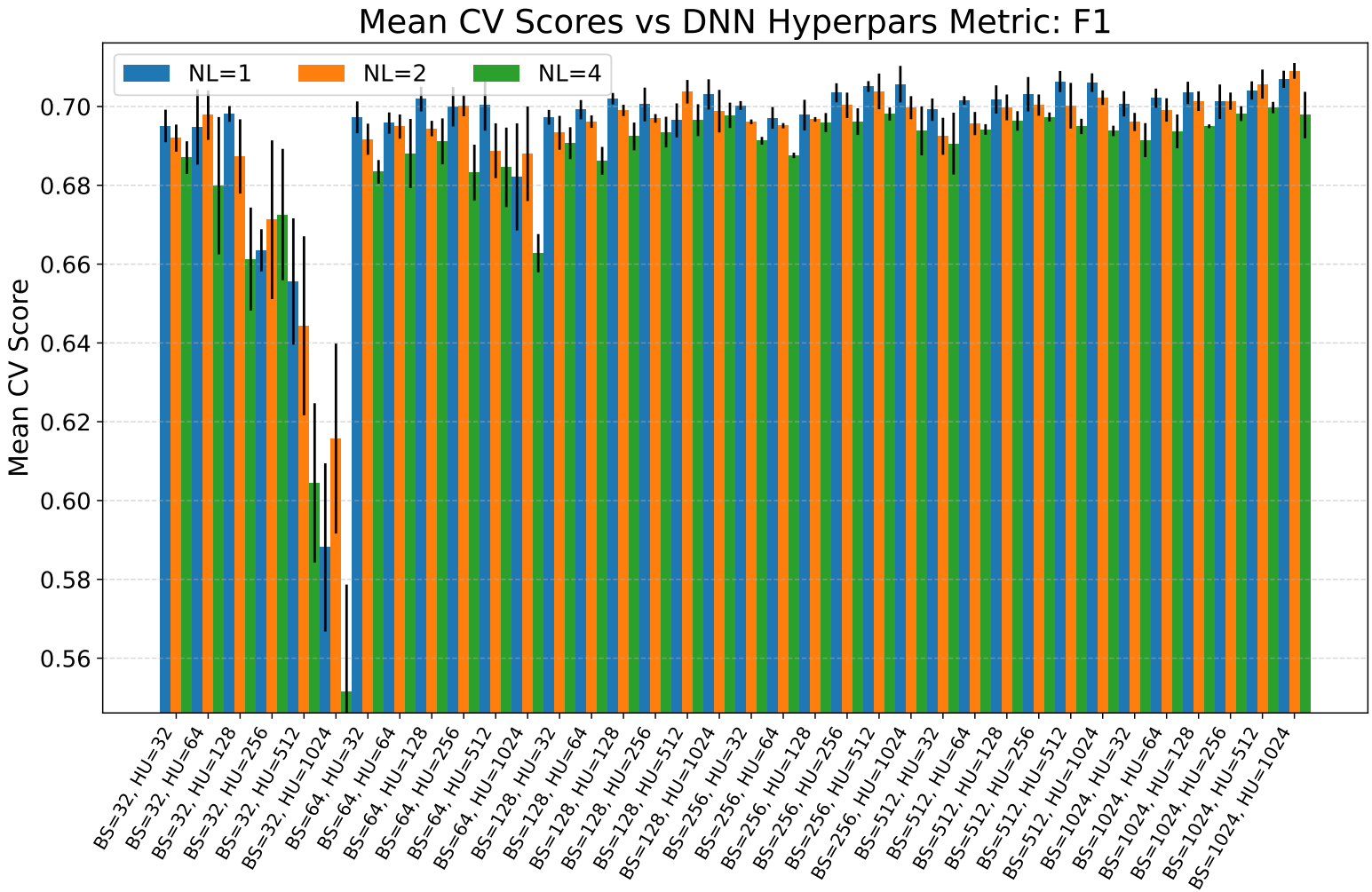
DNN - Hyperparameter Tuning, **Metric: Accuracy**

Hyperparameter	Values	Description
`Batch_size`	[32,64,**128**,256,512]	Number of training samples processed at once before updating weights
`Hidden_units`	[32,64,128,256,512,1024]	Number of neurons within a layer
`Hidden_layers`	[1,**2**,4]	Number of hidden layers



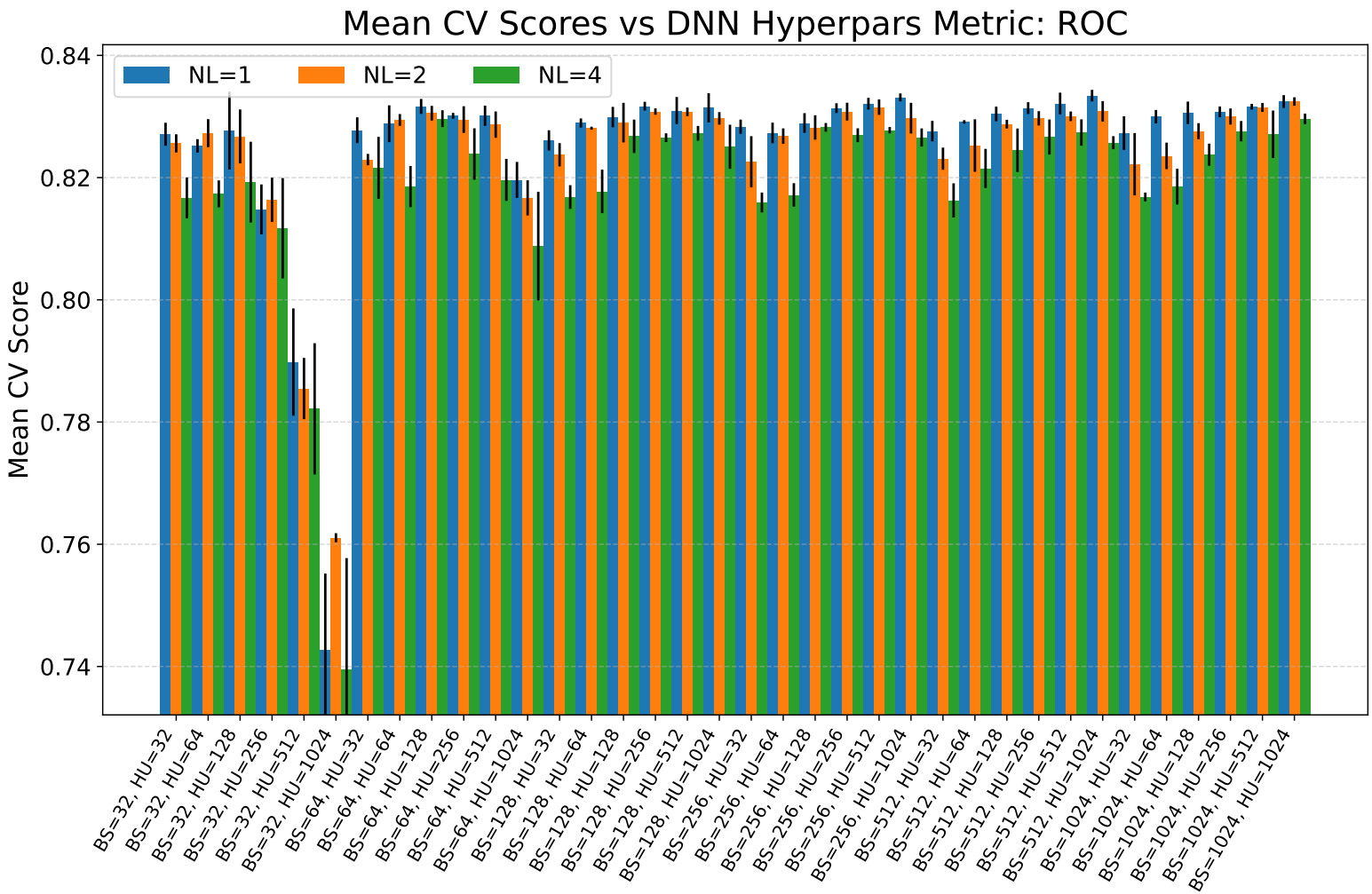
DNN - Hyperparameter Tuning, **Metric: F1 score**

Hyperparameter	Values	Description
`Batch_size`	[32,64,**128**,256,512]	Number of training samples processed at once before updating weights
`Hidden_units`	[32,64,128,256,512,1024]	Number of neurons within a layer
`Hidden_layers`	[1,**2**,4]	Number of hidden layers

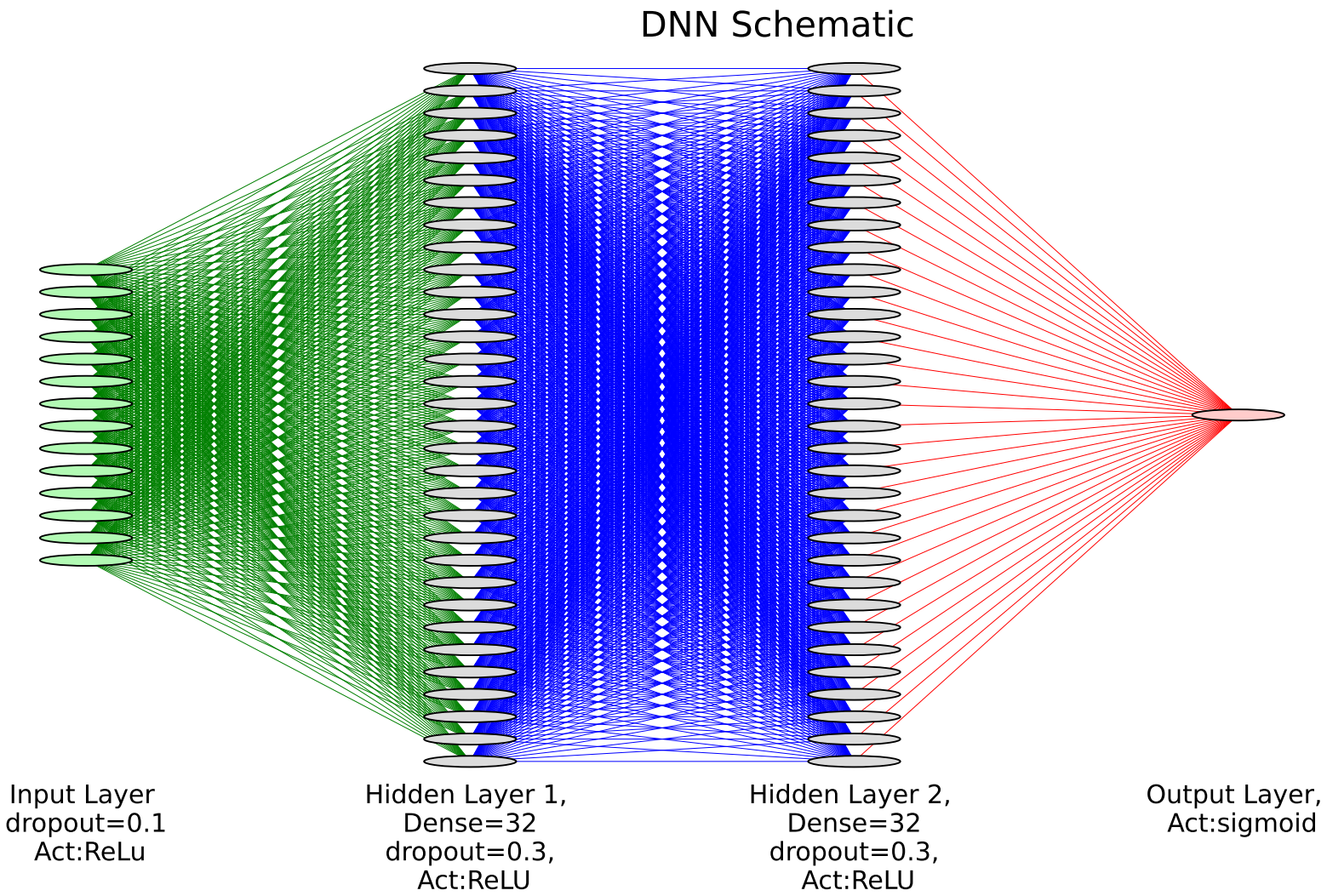


DNN - Hyperparameter Tuning, **Metric: AUC**

Hyperparameter	Values	Description
`Batch_size`	[32,64,**128**,256,512]	Number of training samples processed at once before updating weights
`Hidden_units`	[32,64,128,256,512,1024]	Number of neurons within a layer
`Hidden_layers`	[1,**2**,4]	Number of hidden layers



DNN - Schematic



DNN - Partial Dependence Plots

Partial dependence Plots (PDP):

- Visual technique to understand the relationship between DNN prediction and features
- The effect of the rest of the features is averaged on the model probability
- Helps understand how model prediction varies with feature values by illustrating where the feature has a stronger or weaker influence.

Caveats:

- Works best with uncorrelated features
- As only the average probability is used, the spread in probability is ignored

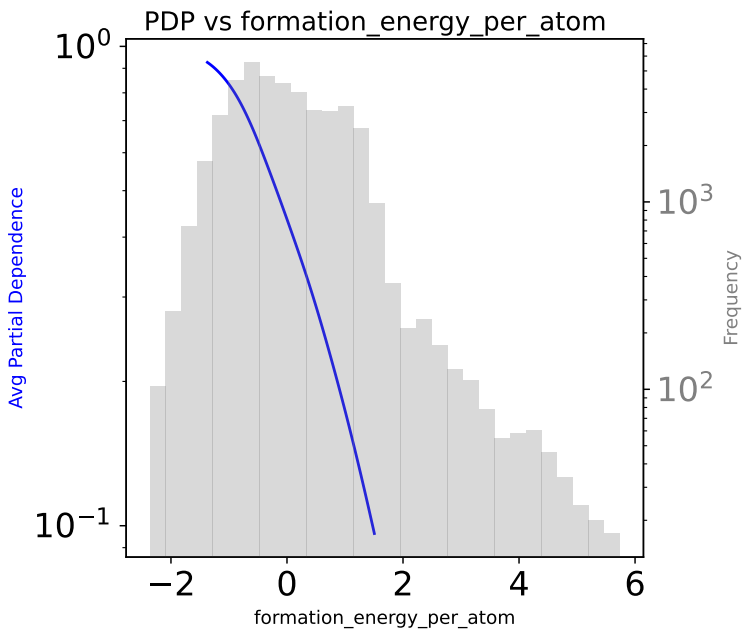
Implementation: Used partial dependence plots from `scikit-learn` to interpret a deep learning model trained with tuned hyperparameters, wrapped for `scikit-learn` compatibility using `KerasClassifier`.

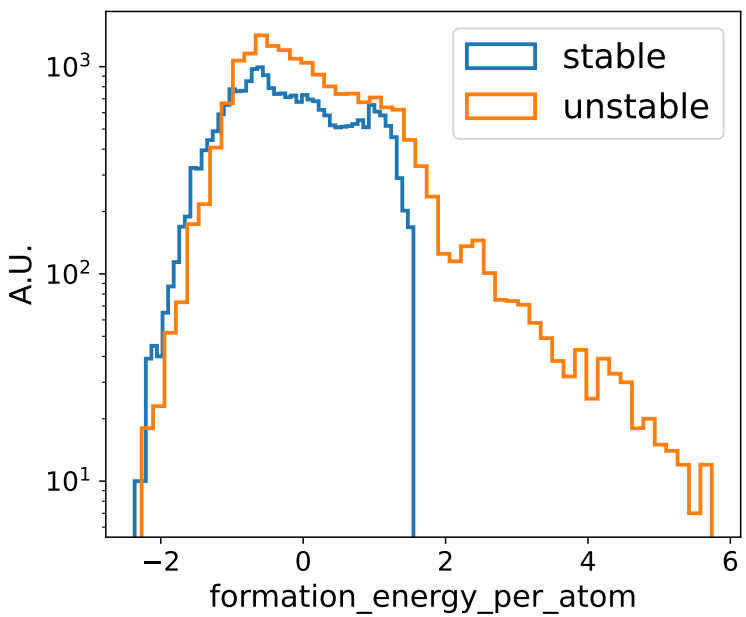
The next two slides show the PDP slides with the feature distribution alongside the stable and unstable feature distribution. The PDP for the rest of the features is shown in the backup slides.

DNN PDP: Energy Feature

Formation Energy per Atom: Energy change when a compound is formed from its constituent elements in their stable energy state:

- Negative formation energy: more thermodynamically stable compound, less likely to undergo spontaneous decomposition
- Positive formation energy: less thermodynamically stable, prone to spontaneous decomposition
- The relationship is nonlinear



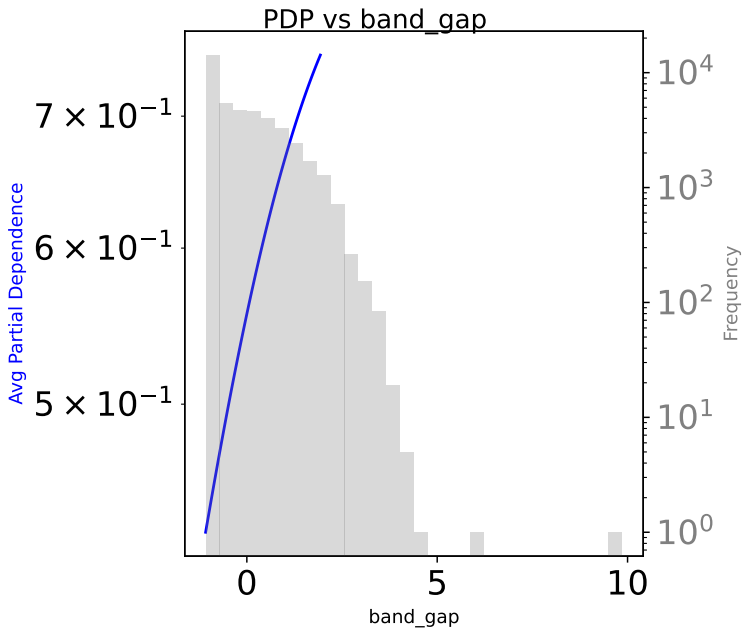


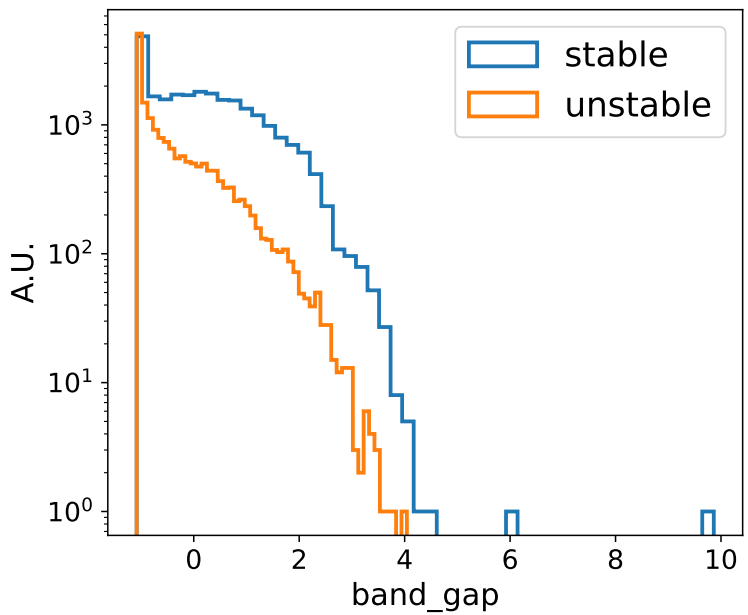
DNN PDP: Electronic Feature

Bandgap: Energy difference between the top of the valence and the bottom of the conduction bands

- **Large bandgap:** Insulator
- **Intermediate bandgap:** Semiconductor
- **Small bandgap:** Conductors

Stable compounds are more semiconductor material than unstable compounds $\Rightarrow$ *PDP increases* in intermediate bandgap values

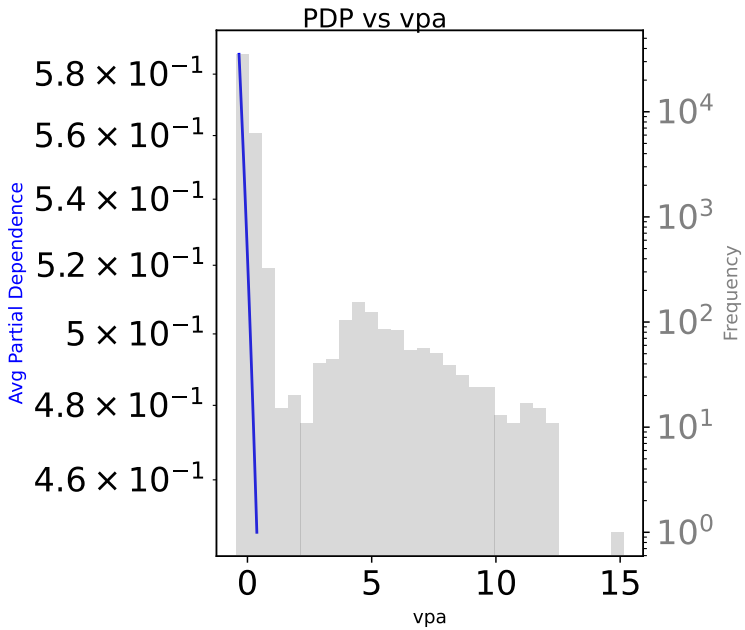


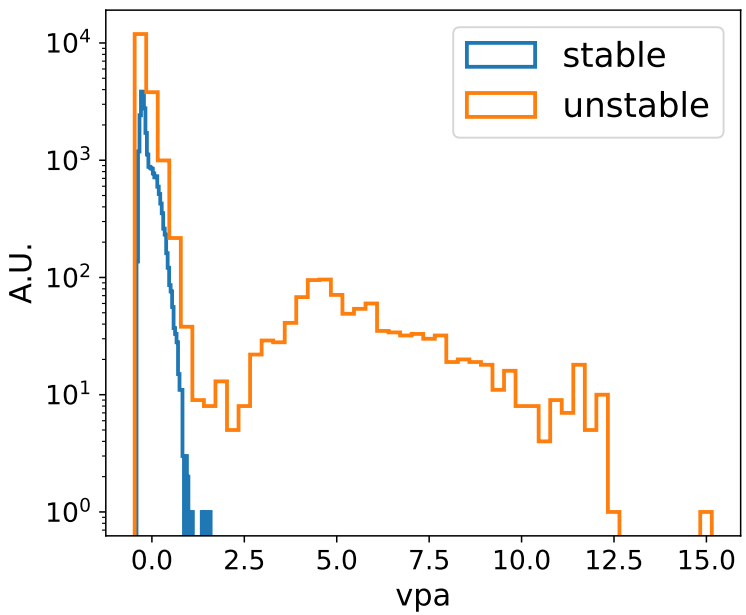


DNN PDP: Structure Feature

Volume per Atom (VPA): Average volume occupied per single atom in the compound

- Inversely related to atomic density
- Directly related to the packing efficiency of atoms in a crystal lattice
- Large VPA: Unstable compound, as highly expanded or porous structures
- Small VPA: More stable compounds



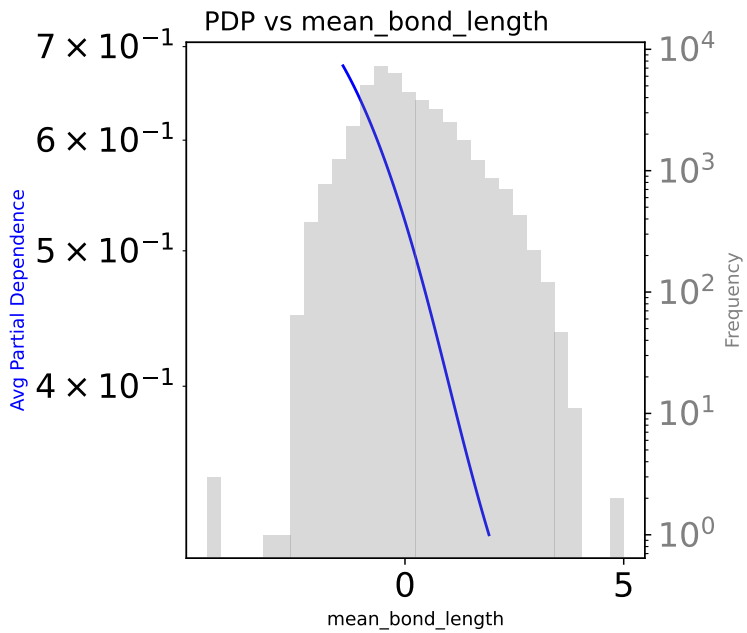


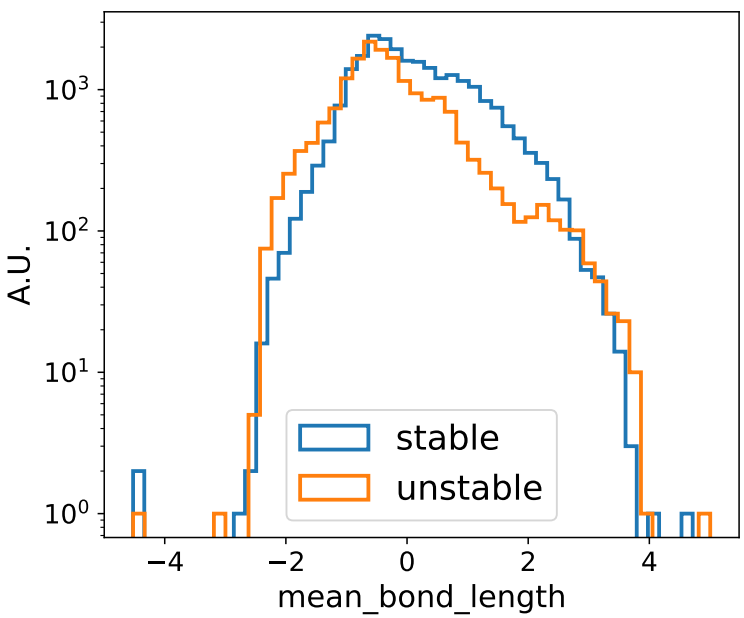
DNN PDP: Bond Structure Feature

Mean Bond Length: Average distance between the nuclei of chemically bonded atoms in a material

- Direct indicator of the strength and nature of interatomic bonds
- Large mean bond length: Larger atoms Less Stable compounds
- Short mean bond length: Higher bond orders $\Rightarrow$ increased electron sharing
- Optimal range: moderate mean bond length

N.B: The horizontal axis shows the scaled normalized mean bond length; henc,e the negative values are datasets with lower than average mean bond length values



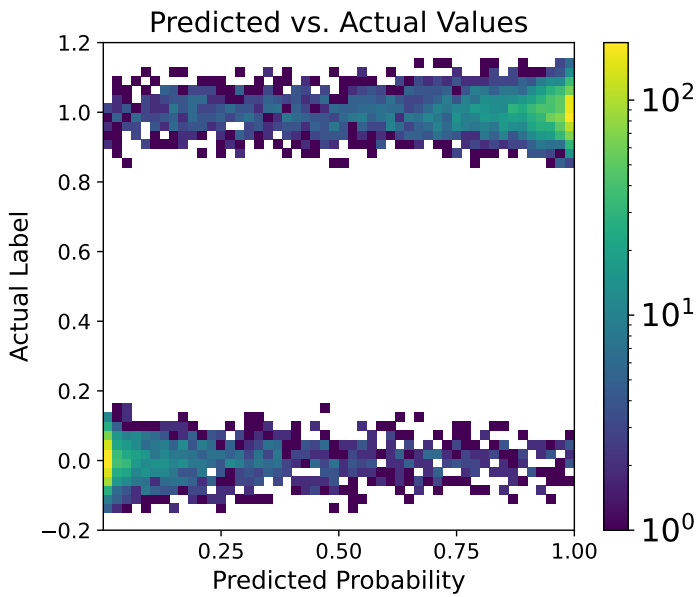


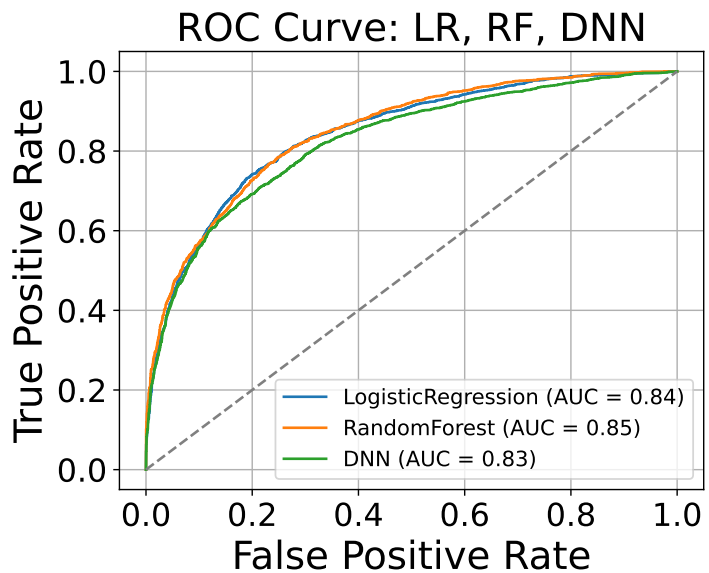
DNN - Performance

- **Predicted probability vs Actual values:** The DNN model on test datasets assesses the DNN's performance. The actual stability is jittered by 5% to help visualize the actual values interpretation:

'Actual_label=0' : unstable; 'Actual_label=1': stable/metastable

- ROC curves
- accuracy
- loss functions





ML Evaluation

ML Methods' Comparison

Introduction

Purpose: Train Machine Learning (ML) algorithms to predict the chemical compounds' thermodynamic stability, following the idea of [3] using material project repository [7] MPR database.

Used methods and algorithms: Binary classification to train, validate, and test ML algorithms:

- Logistic Regression
- Random Forest
- Deep Neural Network (DNN)

Application: This tool can be used for material engineering, pharmaceutical, and sustainable energy technologies

General Info

- ML algorithms are used to train, evaluate, and test three ML algorithms to assess chemical compounds' stability prediction, with the focus on DNN. The binary classification is made using the computed energy_above_the_hull from the MPR dataset.
- Dataset: MPR dataset (computationally derived) is used via a free API_KEY.
- All the codes to make this study are made public and pushed to [6] GitHub.

Methodology

- Publicly available dataset from Material Project (MPR) Database [5]: uses computational material science knowledge and computer science techniques to compute properties of materials
- Data Mining:
  - Feature selection is defined in more detail in feature selection slide
  - Filtering datasets: number of sites and volume cuts
  - Computed arithmetically: volume and magnetic moment per number of sites of the crystal:

$$\text{vpa} = \frac{\text{volume}}{\text{nsites}}$$

$$\text{magmom} = \frac{\text{total magnetization}}{\text{nsites}}$$

  filtering datasets
- feature selections

Challenges and Limitations

- Dataset: MPR dataset (computationally derived) has a limited number of chemical compounds, thus understanding chemical information and creativity in designing neural net are crucial to get a suitable performance
- Inorganic crystal structural database this is not a limitation –¿ should say that in general information
- Computer power: Only one computer was used to process the data and to optimize hyperparameters of the neural network. should be more specific about computer power: how many CPU and GPU were used, how much CPU hour did i use to train and to compute not existed information
- Lack of some MPR data due to incompatibility between computational and experimental data. This was observed for oxidation state for a number of compounds.

Deep Neural Net - Comparison

- expectations

Results

- The ML algorithms are trained to predict the thermodynamic stability by classifying the energy_above_hull feature. The energy_above_hull distribution is shown below.
- A cutoff of 0.05 eV/atom on energy_above_hull is used, as suggested by [4].

Summary & Future Goals

- Summary:
  - Overview to predict stability using ML for inorganic crystalline solid material
  - The role of thermodynamic stability in material design
- Future goals:
  - Moving beyond the prediction of stability to synthesizability

References

[1] Christopher J. Bartel. "Review of computational approaches to predict the thermodynamic stability of inorganic solids". In: *Journal of Materials Science* 57.3 (2022), pp. 10478–10520. DOI: 10.1007/s10853-021-06865-6.

[2] Tianqi Chen and Carlos Guestrin. *XGBoost Documentation: Parameters*. https://xgboost.readthedocs.io/en/latest/parameter.html. Accessed: [Insert date]. 2023.

[3] B. Hao et al. "ElemNet: Deep Learning the Chemistry of Materials From Only Elemental Composition". In: *arXiv preprint arXiv:1812.04153* (2018). URL: https://arxiv.org/abs/1812.04153.

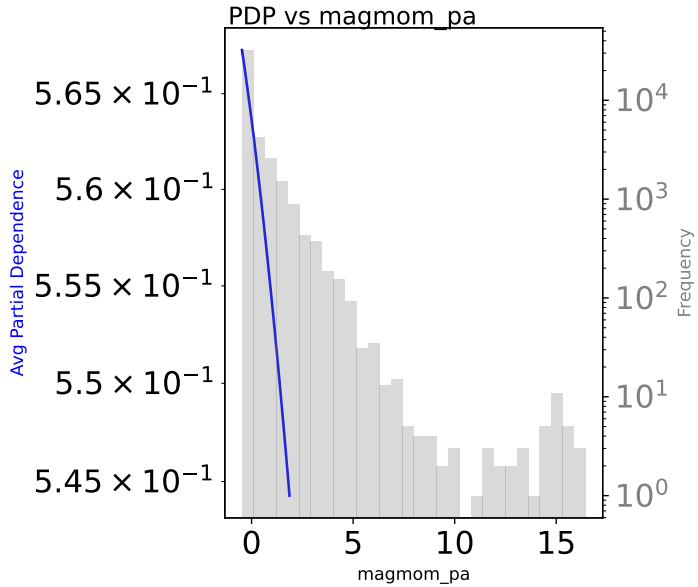
[4] Geoffroy Hautier et al. "Data Mined Ionic Substitutions for the Discovery of New Compounds". In: *Inorganic Chemistry* 50.2 (2011), pp. 656–663. DOI: 10.1021/ic100504j.

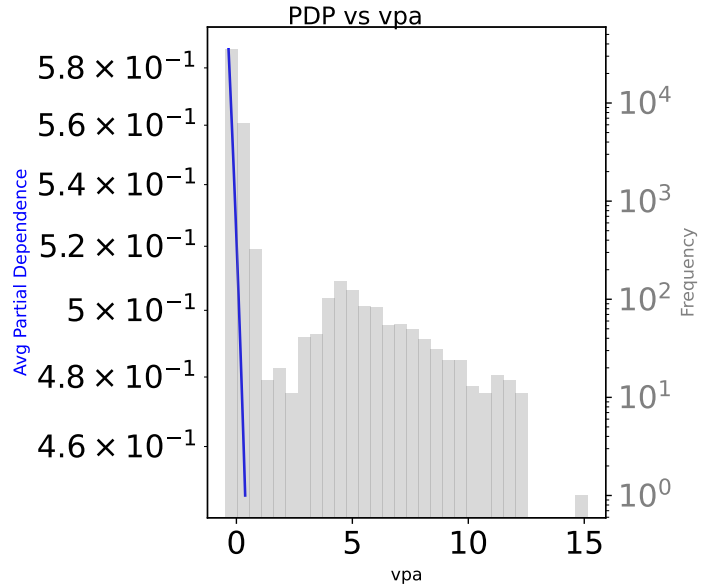
[5] Anubhav Jain et al. "Commentary: The Materials Project: A materials genome approach to accelerating materials innovation". In: *APL Materials* 1.1 (2013), p. 011002. DOI:

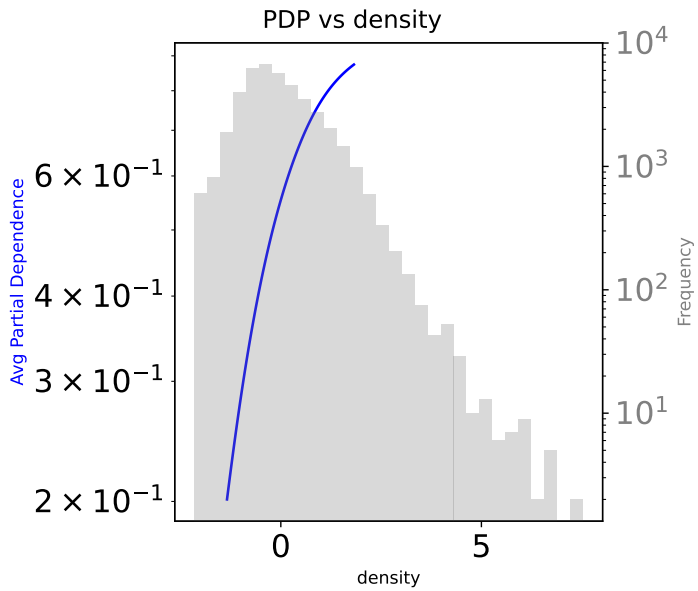
Backup Slides

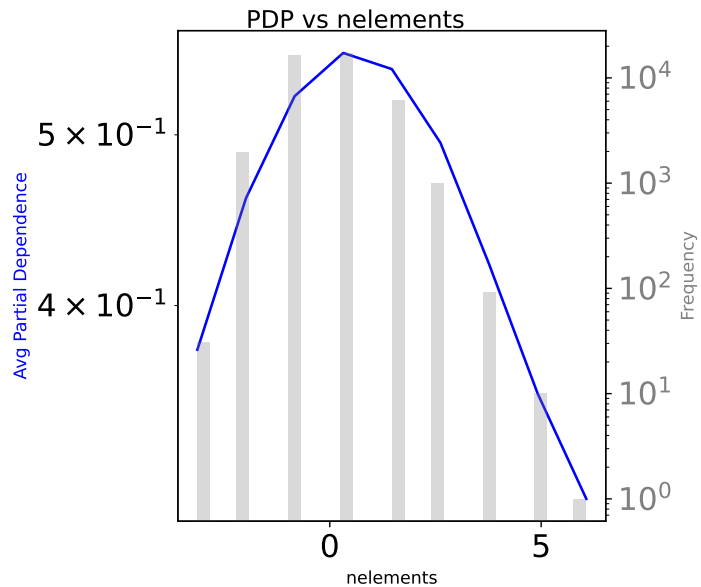
Magnetic moment per atom and volume per atom computation

DNN PDP: Structure & Composition Features









DNN PDP: Bond Structure Features