ME4101A

Bachelor of Engineering Dissertation

Project 318: Exploring two-dimensional ferromagnetic materials by machine learning

Final Report

Sheth Riya Nimish

A0176880R

e0235287@u.nus.edu

Supervisor: Prof. Shen Lei

Department of Mechanical Engineering

National University of Singapore

Semester 2, March 2021

1. Summary

In this project, two-dimensional materials are classified into the categories two-dimensional magnetic materials and two-dimensional non-magnetic materials. The two-dimensional magnetic materials are further classified into two categories, namely two-dimensional ferromagnetic materials and two-dimensional anti-ferromagnetic materials.

It is important to identify two-dimensional ferromagnetic materials from two-dimensional material databases because they are used in various important applications such as data storage mechanisms and spintronic devices.

Raw data from the two-dimensional databases, 2DMatpedia and Computational 2-Dimensional Materials Database was used in the project. Classical machine learning involving algorithms such as decision trees and support vector machines were used to classify the data obtained from the above databases into the respective categories anti-ferromagnetic, ferromagnetic and non-magnetic materials. Techniques such as deep learning, a subset of machine learning and convolutional neural networks, a subset of deep learning were also employed to classify the data. The best results for classification of the two-dimensional data into magnetic and non-magnetic materials were obtained for the classical machine learning algorithm, gradient boost algorithm with an accuracy of 93.37%. The most important features of the materials for determining the magnetic properties were band gap, number of magnetic atoms, maximum ionic character, presence of chromium and presence of manganese in that order. The magnetic materials classified were further classified into ferromagnetic and anti-ferromagnetic materials. The best accuracy for this process was obtained to be 94.44% using the classical machine learning algorithm, random forest classifier. The average ionic character, maximum average ionic character, yang

omega, and band-center were the key features for determining whether a material is ferromagnetic or anti-ferromagnetic.

A web interface was deployed for the models described above so that the end-users could predict the magnetic properties of a material using the models developed through this project. The link to the web application can be found here:
`https://twodferromagnetism-model.herokuapp.com`

The code used for obtaining the results can be found in this github repository:
`https://github.com/RiyaBOT/ME4101A-ShethRiyaNimish`

While the results obtained from this project are promising, continued efforts should be made to obtain datasets of higher quality, conduct an even more thorough process of data mining and featurizing, optimise the models to improve the accuracy and incorporate the use of domain specific knowledge to draw scientific inferences from the results obtained.

2. Acknowledgements

The author wishes to express sincere appreciation for the assistance, support, encouragement and guidance given by the supervisor of this project, Prof. Shen Lei. The author wishes to express gratitude to Zhou Jun and Mohnish Pandey for their help in acquiring the two-dimensional databases used in the project.

Contents

1 Summary **ii**

2 Acknowledgements **iv**

List of Figures **vii**

List of Tables **x**

List of Symbols and Acronyms **xi**

3 Introduction **1**

4 Historical Background **2**

4.1 Nanomaterials 2

4.2 Two-Dimensional Materials 3

4.3 Two-Dimensional Ferromagnetic Materials 3

4.4 Applications of Two-Dimensional Ferromagnetic Materials 5

4.5 Standard Procedure of Discovery of two-Dimensional Ferromagnetic Materials 5

4.6 Machine Learning for Chemical Discovery 6

4.7 Systematic Search for Two-Dimensional Ferromagnetic Materials . . 7

5 Motivation and Objective **9**

5.0.1 Motivation of the Project 9

5.0.2 Objective of the Project 9

5.0.3 General Workflow to Achieve the Objective 10

6 Methodology **11**

6.1 Data Preparation 11

6.1.1 Raw Data Collection 11

6.1.2 Data Featurizing 12
6.1.3 Data Pre-processing 13
6.2 Machine Learning 17
6.2.1 Classical Machine Learning Algorithms Used 17
6.2.2 Explainable Machine Learning 19
6.2.3 Deep Learning 20
6.2.4 Crystal Graph Convolutional Neural Networks (CGCNN) . . 21
6.2.5 Materials Graph Networks (MEGNET) 22
6.3 Reasons for Execution of Three Predictive Techniques 22
6.4 Code 23

7 Results and Discussion **25**
7.1 Results for Data Preparation 25
7.1.1 2DMatpedia 25
7.1.2 Computational 2D Database 27
7.1.3 Discussion on the Reason for Use of the Above Databases . . 27
7.2 Results for Machine Learning 28
7.3 Results for Classical Machine learning 29
7.4 Results for Deep Learning 37
7.5 Results Obtained Using Published Models 41
7.5.1 Crystal Graph Convolutional Neural Networks 41
7.5.2 Materials Graph Networks 43
7.6 Web Application 43
7.7 Discussion 44
7.7.1 Possible Reasons for Better Classical Machine Learning Performance 44
7.7.2 Possible Reasons for Better Gradient Boost Algorithm Performance 44

7.7.3 Discussion on Feature Importances Obtained for the Models 45
7.7.4 Domain Specific Explanation of the Results Obtained 45

8 Recommendations for Further Improvements 46

9 Conclusions 48

References 48

10 Appendix 55

10.1 Code Availability: 55
10.2 Data Availability: 55

List of Figures

4.1 From left to right: A 2D graphene sheet, a 1D carbon nanotube and a 0D quantum dot. 2

4.2 The categorisation of two-dimensional materials based on their magnetic properties. 4

4.3 The structure of two-dimensional ferromagnetic and anti-ferromagnetic materials [1]. 4

4.4 An illustration of the usage of machine learning in the process of chemical discovery[2]. 6

5.1 The concise workflow for the systematic search of 2D ferromagnetic materials. 10

5.2 The workflow to predict the magnetic properties of a material. . . . 10

6.1 The concise workflow for data preparation. 11

6.2 An example of the process of data featurizing. 12

6.3 The steps for data preprocessing. 14

6.4 The relation between the two attributes, Van der Wall's energy per atom and energy per atom, circled in red is linear which shows the dependent relation between the two attributes. Such relations should be investigated and deleted. 16

6.5 Clustering between the features, Van der Wall's energy per atom and band gap. 17

6.6 The different types of classification algorithms used. 18

6.7 The trade-off between accuracy and interpretability for different algorithms [3]. 19

6.8 The different ways of explaining a machine learning model. 20

6.9 The deep learning process depicting the neural networks as a black box with W1 being the weights for the first layer, W2 the weights for the second layer and so forth. 21

7.1 The heat matrix of the data obtained from 2DMatpedia. 26

7.2 The workflow to predict the magnetic properties of a material. Here, M/NM means classification into magnetic and non-magnetic materials, M:FM/AFM, implies that the magnetic material was classified into ferromagnetic and anti-ferromagnetic materials. FM/AFM/NM means classification into ferromagnetic, anti-ferromagnetic and non-magnetic material. 28

7.3 The confusion matrix of the results obtained after machine learning using the gradient boost classifier [4]. 29

7.4 The ROC curve. 29

7.5 The precision recall curve. 29

7.6 The five important features in the prediction of the magnetic properties of a material. 30

7.7 The SHAP values of some of the most important features of this model. 32

7.8 The predictive power of each of the features along with the side they push the prediction, i.e the magnetic or the non-magnetic side. . . . 33

7.9 Example of how the band gap affects the SHAP values. 33

7.10 The explainability of the choice of prediction for the compound MoS_2 using LIME. 34

7.11 The feature importance for classification of a magnetic material into ferromagnetic and anti-ferromagnetic materials. 35

7.12 Accuracy vs number of neurons in each layer. 38

7.13 Accuracy vs batch size. 38

7.14 Accuracy vs number of layers. 38

7.15 Optimisation of batch size, the number of neurons and the number of layers. 39
7.16 The best model accuracy. 39
7.17 The best model loss. 39
7.18 The predictive power of each of the feature and to which side they push the prediction, i.e towards the magnetic side or the non-magnetic. 39
7.19 Force plot explanations of the entire dataset. 40
7.20 The epoch accuracy of the model 40
7.21 The epoch loss of the model 40
7.22 The change in the model accuracy with a change in the learning rate. 41
7.23 The change in the model accuracy with a change in the convolutional neural networks. 41
7.24 The average model accuracy before optimisation. 42
7.25 The average model accuracy after optimisation. 42
7.26 Time taken per episode vs number of epochs. 42

List of Tables

7.1 Best metrics obtained in the machine learning model for 2DMatpedia. 31

7.2 Results with different thresholds of magnetic moment for classification. 35

7.3 Best metrics obtained in the machine learning model for C2DB . . . 37

List of Symbols and Acroynms

AFM	Anti-Ferromagnetic Material.
C2DB	Computational 2-Dimensional Material Database.
CGCNN	Crystal Graph Convolutional Neural Networks.
FM	Ferromagnetic Material.
MEGNET	Material Graph Networks.

3. Introduction

The purpose of this project is to classify two-dimensional materials based on their magnetic properties. Using machine learning, the materials would be classified into three categories: non-magnetic, ferromagnetic and anti-ferromagnetic materials.

The problem that exists in conventional methods of classification is that the experimental procedure is time-intensive, prone to errors and relies greatly on serendipity. The use of machine learning in the classification of materials based on the magnetic properties specifically has not been thoroughly explored. Furthermore, the existing models have not yielded as accurate results as are required to limit the need for experimental testing of two-dimensional materials to determine its magnetic orientation. The models do not provide explainable results which could help researchers discern the relation of various material properties with the magnetic orientation.

In this report, the historical background of the project is detailed and the methodologies used to achieve the objective are elucidated. The methodologies include the discussion of the specific implementations of data preparation, followed by classical machine learning, deep learning and graph networks. After describing the algorithmic procedure employed in the project, the results obtained using the above methods are discussed in great detail and scientific inferences and conclusions are drawn from them. The code used for obtaining the results can be found in this github repository:

`https://github.com/RiyaBOT/ME4101A-ShethRiyaNimish`

4. Historical Background

4.1 Nanomaterials

Nanomaterials are defined as materials having at least one dimension in the nanoscale (a range of 1-100nm) [5]. Nanomaterials can be classified into three categories:

- If all the three dimensions of a material are nano-sized, it is a zero-dimensional material or a nanoparticle.
- If two out of the three dimensions of a material are nano-sized, it is a one-dimensional material or a nanotube.
- If one of the three dimensions of a material is nano-sized, it is a two-dimensional material [6].

Figure 4.1 shows the structures of materials in reducing dimensionality.

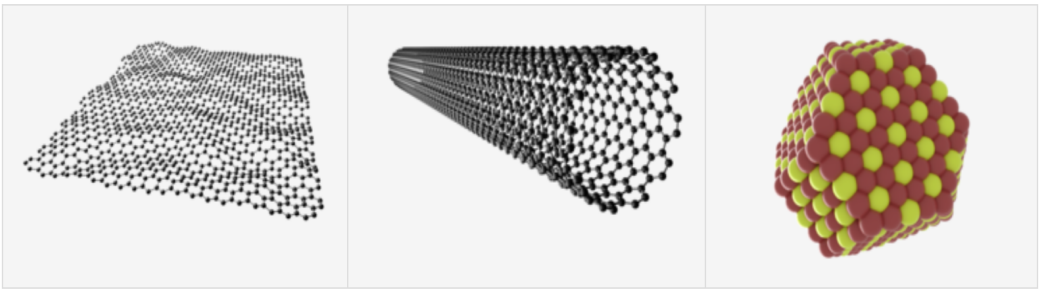

Figure 4.1: From left to right: A 2D graphene sheet, a 1D carbon nanotube and a 0D quantum dot.

This project focuses on one category of nanomaterials, namely the **two-dimensional materials**.

4.2 Two-Dimensional Materials

The first two-dimensional material was discovered in 2004 along with the discovery of graphene. Geim and Novoselov, received the Nobel Prize in Physics for this groundbreaking discovery. This discovery generated tremendous excitement and interest in research as well as high-tech applications due to the unique properties of two-dimensional materials. There are three primary reasons why two-dimensional materials are significantly different from their bulk counterparts:

- Absence of Van der Walls interaction,
- Increased surface area to volume ratio,
- Confinement of electrons in a plane [6].

For the above reasons, two-dimensional materials, which have extremely different mechanical, optical, electrical, thermal, chemical and magnetic properties have revolutionised device applications. One such category of two-dimensional materials that is particularly useful in spintronics, optical communication and quantum computing is the category of **two-dimensional ferromagnetic materials** [7].

4.3 Two-Dimensional Ferromagnetic Materials

The two-dimensional materials can be divided into three categories based on the ordered magnetic properties as show in the flow chart in Figure 4.2. The categories are namely non-magnetic, ferromagnetic and anti-ferromagnetic.

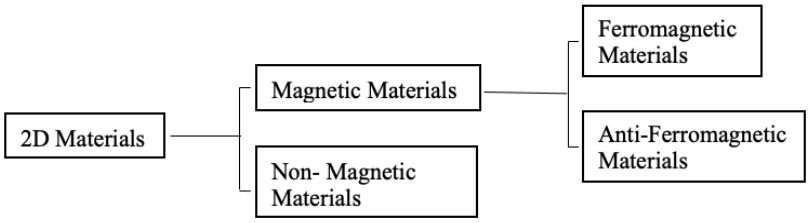

Figure 4.2: The categorisation of two-dimensional materials based on their magnetic properties.

- Magnetic Materials: Materials which have non-zero magnetic moment.
  - Ferromagnetic Materials: Materials which have the magnet moments in parallel alignment.
  - Anti-ferromagnetic Materials: Materials which have the 50-50 anti parallel alignment.
- Non-magnetic Materials: Materials which have zero magnetic moment.

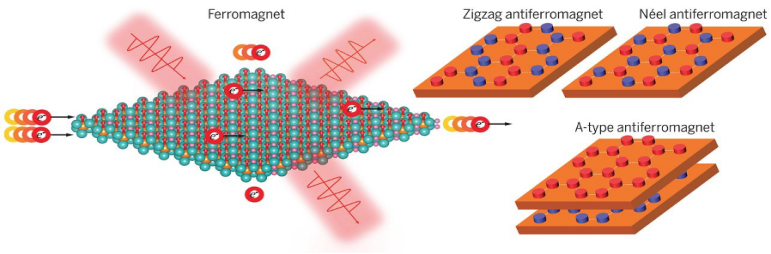

Figure 4.3: The structure of two-dimensional ferromagnetic and anti-ferromagnetic materials [1].

Two-dimensional ferromagnetic materials, being a special subset of two-dimensional materials due to the parallel alignment of the magnetic moment, are extremely useful in many device applications.

4.4 Applications of Two-Dimensional Ferromagnetic Materials

Some of the uses of two-dimensional ferromagnetic materials are listed down below:

- **Spintronic Devices:** These devices use the properties of electrons to transmit, process and store information. Some spintronic devices are spin-valves, spin-orbit torques, spin field-effect transistors and spin-filters [8].
- **Magnetic Tunnel Junctions (MTJ):** These are used in modern disk drives and magnetoresistive random-access memory, a new non-volatile memory [9].
- **Quantum Technologies:** Technology based on the physics of sub-atomic particles which can be used in sensing, secure communications and computing [10].

Two-dimensional ferromagnets at room-temperature are highly desirable for advanced spintronic devices [11]. Hence, a lot studies have been conducted for the high-throughput discovery of high Curie point two-dimensional ferromagnetic materials using the Heisenberg model-based Monte Carlo simulations to propel the practical applications of two-dimensional ferromagnets [12].
Due to the myriad of applications of two-dimensional ferromagnetic materials, it is important to have a systematic search and discovery process for new two-dimensional magnetic materials.

4.5 Standard Procedure of Discovery of two-Dimensional Ferromagnetic Materials

Two-dimensional ferromagnetic materials have been conventionally discovered using a trial and error experimental method. For instance, the intrinsic long range

ferromagnetic order of the material $Cr_2Ge_2Te_6$ was discovered by scanning magneto-optic Kerr microscopy. The classification of a material into ferromagnetic and anti-ferromagnetic is generally done by observing the effect of small applied fields and explaining the resultant phenomenon using spin-wave theory. This is one of the general experimentation procedures that takes place in the determination of whether a material is ferromagnetic or not [13]. These experimental methods, apart from being time-consuming, are also very expensive.

4.6 Machine Learning for Chemical Discovery

The discovery of new molecules and materials is based on the exploration of constantly expanding chemical compound spaces. A particular chemical discovery concerns various aspects such as degrees of freedom, catalytic conditions, different energies and so forth. Hence due to the wide range of possibilities, it is pivotal to have a statistical view on chemical discovery. This is the very reason that there has been a rise in the use of machine learning techniques in the field of molecular and material science [2]. The general workflow for chemical discovery using machine learning has been delineated in Figure 4.4.

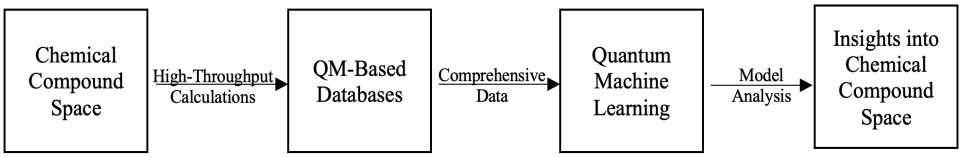

Figure 4.4: An illustration of the usage of machine learning in the process of chemical discovery[2].

First, datasets of molecular structures are generated from subsets of the pertinent chemical compound spaces. High-throughput quantum mechanical calculations are then carried out on the datasets to create databases based on the quantum mechanical molecular properties. Once an extensive amount of quality data is obtained from

the datasets, machine learning algorithms are used to analyse the different quantum mechanic properties. The study of machine learning models is employed to extract information from the respective chemicals which assists in chemical discovery. Apart from classical or non-deep machine learning, techniques like deep learning and graph networks have also been used to achieve the same objective [2].

Machine learning in chemical discovery, specifically drug discovery, has been very successful. In addition to that chemical properties of the drugs such as toxicity, metabolism, drug-drug interactions, carcinogenesis and so forth have been effectively modelled using machine learning as well [14]. De Jong *et al.* in 2016 used machine learning to accurately predict the elastic properties of a large set of inorganic compounds [15]. The above are just two of the numerous successful applications of machine learning in the field of material prediction and chemical discovery. Hence, a similar concept was applied in the systematic search for two-dimensional ferromagnetic materials as well.

4.7 Systematic Search for Two-Dimensional Ferromagnetic Materials

Experimental discovery of two-dimensional ferromagnetic materials is very complex and involves serendipity. Hence, to simplify and speed up the process, novel approaches like (i) high-throughput electronic structure and (ii) classical machine learning plus deep learning models have been used for the discovery, classification and design of new magnetic materials. This project focuses on the second approach mentioned above which is using **classical machine learning** and **deep learning models** for the search of two-dimensional ferromagnetic materials. To create the machine learning and deep learning models, the following databases have been used by researchers:

- Inorganic Crystal Structure Database [16]
- 2DMatpedia Database [17]
- Computational 2D Materials Database [18]

Numerous researchers have used the second approach as mentioned above before, some of the pioneering published models have been listed below:

- ElemNet- A deep neural network model that takes only the elemental compositions as inputs and leverages artificial intelligence to automatically capture the essential chemistry to predict materials properties [19].
- SchNetPack- A toolbox for the development and application of deep neural networks that predict potential energy surfaces and other quantum-chemical properties of molecules and materials [20].
- Crystal Graph Convolutional Neural Networks (CGCNN)- A software that takes an arbitrary crystal structure to predict material properties. [21]
- Materials Graph Networks (MEGNET)- An implementation of DeepMind's graph networks for universal machine learning in materials science [22]

However, while there are machine learning and deep learning models available (such as those mentioned above), some of them are not specifically catered to the search of two-dimensional ferromagnetic materials. Furthermore, the accuracy of these models are not as accurate and precise as it needs to be. Moreover, the models are like black boxes; that is, researchers are unaware of the exact implementations of the model. The models are not interpretable and explainable which increases ambiguity. This brings the author to the objective of the current project.

5. Motivation and Objective

5.0.1 Motivation of the Project

Two-dimensional ferromagnetic materials are immensely crucial for technological acceleration in a plethora of fields such optics and spintronic devices, among others. However, the discovery of two-dimensional ferromagnetic materials is done by a tedious experimental trial and error procedure, which is labour and time intensive, prone to errors, reliant on serendipity and cost-inefficient. Hence, an alternate method for the discovery and classification of two-dimensional ferromagnetic materials must be found. They are numerous applications of machine learning techniques in the recent past which have successfully demonstrated material prediction and classification abilities. Hence, similar techniques in the prediction of whether a two-dimensional material is ferromagnetic or anti-ferromagnetic are aimed to be employed. This process using the principles of data-mining and a machine learning model would result in an extremely fast computational time for prediction.

5.0.2 Objective of the Project

The objective of this project is to conduct a systematic materials informatics search for two-dimensional ferromagnetic materials from the 2DMatpedia Database [17] and Computational 2D Materials Database (C2DB) [18] using classical machine learning and deep learning algorithms and to explain the results obtained through these techniques.

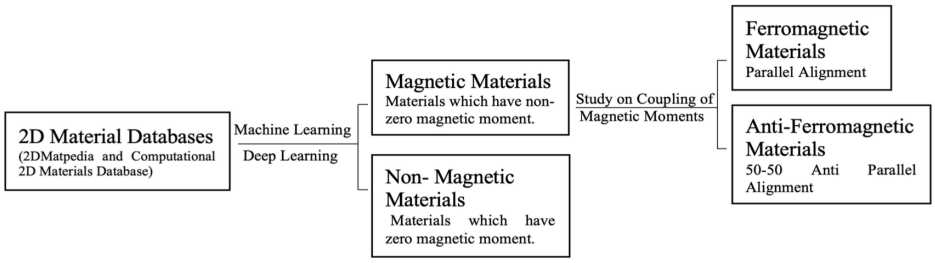

Figure 5.1: The concise workflow for the systematic search of 2D ferromagnetic materials.

5.0.3 General Workflow to Achieve the Objective

To achieve the objective outlined above, a systematic workflow was created. The raw data from the two-dimensional databases would be treated to prepare it for classical machine learning algorithms, deep learning algorithms and graph network algorithms. The results obtained from these algorithms would then be explained and interpreted and tried to be further improved.

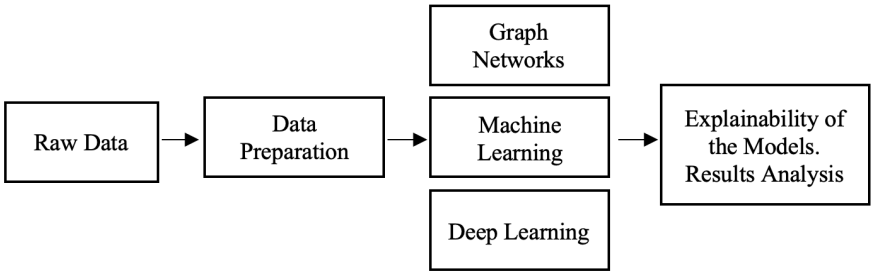

Figure 5.2: The workflow to predict the magnetic properties of a material.

6. Methodology

This chapter provides an outline of the research methodology of this dissertation. The author outlines the research approach, the methods of data preparation and the types of predictive models.

6.1 Data Preparation

Raw data is often not enough to create an accurate predictive model and poor data can ruin the efficacy of a good predictive model. Hence, data preparation is pivotal for the process of data mining. Data preparation has three important sub-steps as described in Figure 6.1. The first sub-step is the collection or raw data, followed by data featurizing and concluding with data pre-processing.

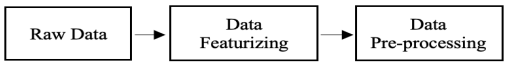

Figure 6.1: The concise workflow for data preparation.

6.1.1 Raw Data Collection

The data collected here was labelled data, meaning that the data was composed of data entries where each of the data entries had input elements that were provided to the model and an output element or a target which the model was expected to predict. Labelled data is necessary for supervised learning problems. Classification of data, which is the objective of the project, is an example of a supervised learning problem. Hence, it was necessary that the data used was labelled. The labelled data was collected from two reliable two-dimensional material databases, the 2DMatpedia Database [17] and the Computational 2D Materials Database [18].

6.1.2 Data Featurizing

The raw data obtained may have very few numerical and computer-interpretable features. It is vital to add other additional engineering features that provide information that differentiates patterns in the data. This step helps to extract more information from existing data. This process is called feature engineering or featurization where the domain knowledge of the data is leveraged to create features which assist the machine learning algorithms to perform better. To featurize the dataframe, a Python library called Matminer was used [23]. Matminer is used for data mining the properties of materials. It contains routines for obtaining data on material properties from various databases, featurizing complex materials attributes (e.g., composition, crystal structure) into physically-relevant numerical quantities (e.g., valence orbitals, oxidation states) and analyzing the results of data mining. An example of data featurizing can be seen here:

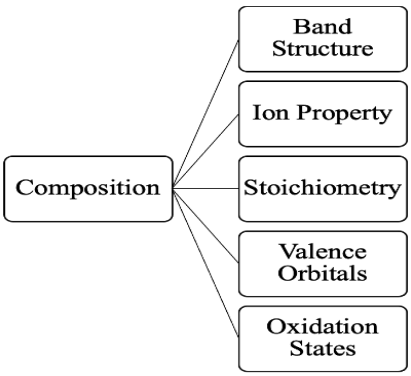

Figure 6.2: An example of the process of data featurizing.

Through the use of the Matminer in the above example, numerical molecular properties or features could be derived from the composition of the material which can be interpreted by a machine learning model. In addition to Matminer, another library called Pymatgen was also used. Pymatgen (Python Materials Genomics) is a robust, open-source Python library for materials analysis [24]. An attempt was made to obtain as many features as available through the Python libraries as new

information could be found in the form of new features and these features could have the ability to predict the target property thereby improving the model accuracy. Some important features that can be obtained from Matminer and Pymatgen have been explained here [23]:

- Yang Omega and Yang Delta: Mix of thermochemistry and size mismatch terms of Yang and Zhang (2012).
- Average D Valence Electrons and Frac S Valence Electrons: Examples of valence orbital attributes such as the mean number of electrons in each shell.
- Band Center: Estimation of absolute position of band center using geometric mean of electronegativity.
- Maximum Ionic Character and Average Ionic Character: Ion property describing whether a composition is charge-balanced.
- Decomposition Energy: The maximum amount of energy which can be released in decomposition [25].
- Band Gap: In solid-state physics, a band gap, also called an energy gap, is an energy range in a solid where no electronic states can exist.
- Energy Per Atom: Kinetic and potential energy of atoms as a result from the motion of electrons.
- Presence of Elements: Fraction of the material containing the element,

6.1.3 Data Pre-processing

Real-world data may be inconsistent, noisy and incomplete. Hence, **data preprocessing** which generates quality data was conducted in a systematic manner as seen in Fig 6.3 [26].

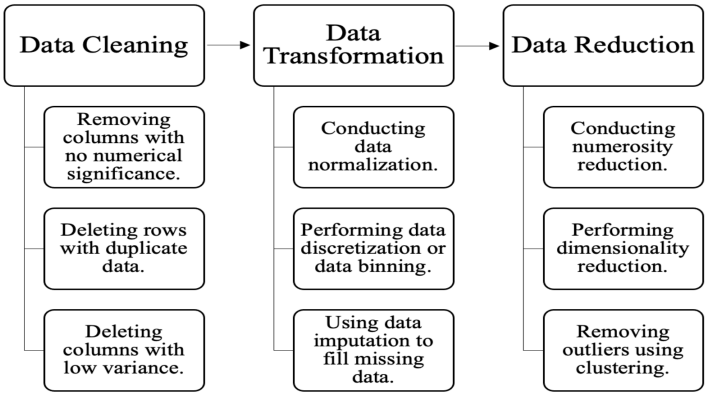

Figure 6.3: The steps for data preprocessing.

Data Cleaning

- In the sub-step **data cleaning**, it is crucial to remove redundant data such as columns with low variance, that is features which do not vary much within itself and have little predictive power [27]. Data with no numerical significance such as date of discovery were removed. Duplicate data entries were removed as well.

Data Transformation

Supplementary Note: For data discretization and data normalisation, it should be noted the target feature should remain untouched.

- The **binning** or discretization or concept hierarchy generation method is used to minimize the effects of small observational errors and smoothen out the data [28]. Data binning groups numbers having continuous values into smaller bins. The number of bins and the width is determined using Sturge's rule:
  number of bins $= 1 + \log_2(\text{length of data})$.

- The goal of **data normalization** is to change the values of numeric columns in the data set to a common scale, without distorting differences in the ranges

of values. This ensures that one attribute is not given greater importance just by virtue of higher magnitudes of the attribute entries [29].

- **Data imputation** is the process of replacing missing data with substituted values (obtained through methods like regression or clustering). Often times, missing data entries are replaced with the average value of the entire column.

Data Reduction

- **Dimensionality Reduction**: In a machine learning model, each feature should be independent otherwise the problem of multi-collinearity that leads to redundant variables in the machine learning model arises. Hence, a pair-plot is used to plot pairwise relationships of each attribute in a dataset. The pair-plots which depict a relation (for instance, linear or exponential) should be investigated and appropriate attribute selection must be performed. An example of the graph can be seen below in Figure 6.4. Here, the x-axis and y-axis are the features and the plots corresponding to each of those are the relations between that feature on the x-axis to the feature on the y-axis. Along the diagonal, we do not see any such plot because that means the feature on the x-axis and the feature on the y-axis are the same. If we see any mathematical relation in any of the subplots of Figure 6.4, it means that there are features in the data that are closely related to each other. For better performance of the machine learning model, all features should be independent of each other. Hence, those related features should be observed carefully and appropriate action should be taken. In the below example in Figure 6.4, the subplot with a red circle shows that there is a linear relationship between the two features. Reading the x-column and the y-column, we realise that the two features are Van der Wall's energy per atom and energy per atom. That means these two features are closely related. Hence, one of the features must

be deleted.

Figure 6.4: The relation between the two attributes, Van der Wall's energy per atom and energy per atom, circled in red is linear which shows the dependent relation between the two attributes. Such relations should be investigated and deleted.

- **Numerosity Reduction**: The code gives the option to reduce the number of data entries. In case the data is extremely huge and it could exponentially increase the time and resources of the machine learning algorithms, this would reduce the size of the data.

- **Clustering**: While ideally clusters should be available for each pair of features, that would result in $\binom{156}{2}$ graphs. Hence, a feature to obtain as many number of graphs as needed has been given in the code. That is, the user can define how many cluster plots he/she would like to see, after that he/she would

enter the rows and columns of each of those cluster plots and the number of clusters he/she would like to see in the plot. This is a method of manual inspection of the data. An example of the clustering can be seen below, a graph of Van derWall's energy against band gap, where the data is divided into four clusters, each red dot being the centroid of one of the clusters.

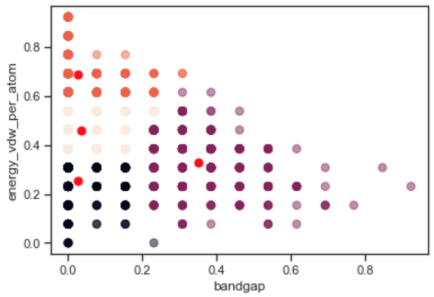

Figure 6.5: Clustering between the features, Van der Wall's energy per atom and band gap.

6.2 Machine Learning

In this section, the machine learning techniques used to create a predictive model are described.

Classical or non-deep machine learning models contain a pair of inputs and desired output and the goal is to learn a mapping between the input and the output spaces [30]. Deep learning is a sub-field of machine learning and relies on layers of artificial neural networks. Convolutional neural networks, which is in turn a sub-field of deep learning, is generally applied to analysing visual imagery [31].

6.2.1 Classical Machine Learning Algorithms Used

In this section, the classical or non-deep machine learning algorithms, which are dependent on human intervention to learn and need labelled datasets to work, are described. Scikit Learn, a machine learning library in Python was used for

conducting machine learning. A plethora of algorithms which were used to conduct machine learning are grouped in representative categories as seen in Fig 6.6.

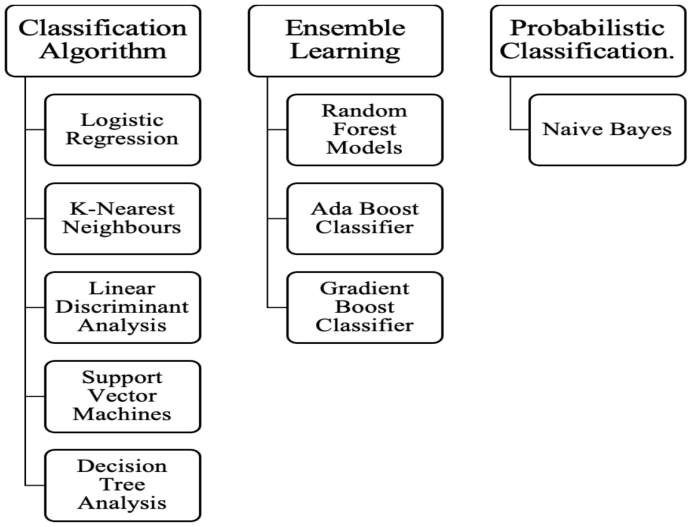

Figure 6.6: The different types of classification algorithms used.

The taxonomy of arranging the machine learning algorithms into different types is important because it helps us determine which algorithm would be the best for the particular dataset at hand.

- **Decision Tree Algorithm**: This constructs a model of decisions based on actual values of attributes in the data.
- **Bayesian Algorithm**: This algorithm applies Bayes' theorem for classification.
- **Clustering Algorithm**: This is typically organized by the modelling approaches such as centroid-based and hierarchal.
- **Dimensionality Reduction Algorithm**: This algorithm seeks and exploits the inherent structure in the data
- **Ensemble Algorithm**: This model contains multiple weaker models that are independently trained and whose predictions are combined in some way

to make the overall prediction [32].

Train/Test/Validation Data Split

Different percentages of training, testing and validation of the data was also explored to ensure that the optimum model could be obtained without overfitting or underfitting the model. Overfitting occurs when a model models the training data so closely that it negatively impacts the performance of the model on the testing data. Underfitting on the contrary refers to a model that can neither work on the training data or the testing data.

6.2.2 Explainable Machine Learning

Previously machine learning models have been black boxes which predict with a certain degree of accuracy. However, in this adoption of machine learning for producing scientific outcome, explainability is important to assess the scientific value of the outcome. It is hence vital to understand the way a specific model operates and the underlying reasons for the produced decisions. However, as seen in the graph below, there is a trade-off of accuracy and interpretability. The inaccurate models are easy to understand while the more accurate model gets progressively more incomprehensible.

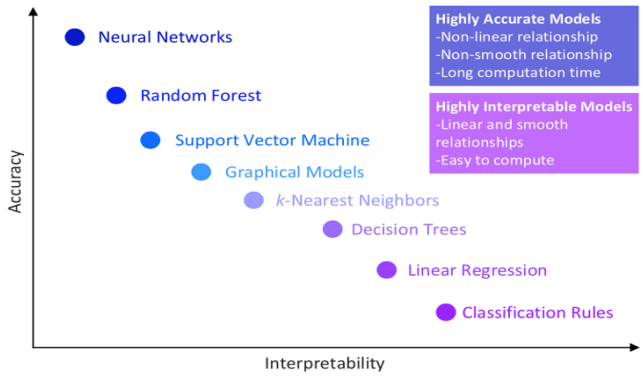

Figure 6.7: The trade-off between accuracy and interpretability for different algorithms [3].

There are different ways of explaining the model at hand. Either the model's interpretability could be achieved by limiting its complexity (intrinsic explainability). If the model is analysed after training, that would constitute a post-hoc explainability. If the interpretation was pertaining to an individual prediction, it would be a local explanation and if it interpreted the entire model it would be a global prediction. [33].

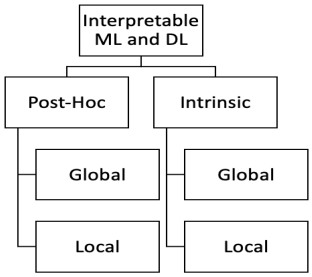

Figure 6.8: The different ways of explaining a machine learning model.

In this project, SHAP or the SHapely Additive Explanantion was used as a global post-hoc approach to explain the models. SHAP is a game theoretic approach which uses the classic Shapley values [34]. Further LIME or Local Interpretable Model-Agnostic Explanations were used as a local post-hoc approach to explain a particular instance in the model [35]. Since intrinsic models compromise the accuracy of the model, they were avoided.

6.2.3 Deep Learning

Deep learning uses multiple layers to progressively extract higher level features from the input. Each layer changes the data into a representation that is more abstract and composite.

To perform deep learning, Keras a deep learning API in Python was used. Two Keras models were used: a simple model called the Sequential model which is a linear stack of layers and a model for more complex architectures called the Functional

API which allows to build arbitrarily graphs of layers. One of the difficulties with deep learning is that the workings of a neural network is not very understandable as seen in Fig 6.9.

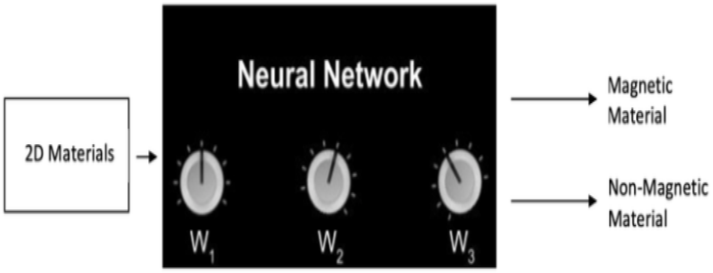

Figure 6.9: The deep learning process depicting the neural networks as a black box with W1 being the weights for the first layer, W2 the weights for the second layer and so forth.

There are numerous parameters which can change in a deep learning model such as the number of layers, the number of neurons in each of those layers, the batch size and the activation functions. Hence, it is important that a systematic experimentation pattern is devised such that the hyper-parameters of the model are optimised.

The deep learning models were explained using the same techniques as mentioned above in the Section 6.2.2. However, as seen in Figure 6.7, the interpretability of the deep learning model reduces with neural networks. Hence, the results obtained after using SHAP and LIME may not be as informative.

6.2.4 Crystal Graph Convolutional Neural Networks (CGCNN)

Apart from developing code, the author used machine learning models published in papers to conduct prediction studies. Xie et al used crystal graph convolutional neural networks to directly learn material properties from the connection of atoms in the crystal [21]. The convolutional neural networks were built on top of the graphs obtained from the crystal structure to automatically extract information

that is suitable for predicting the target property. This model, unlike classical machine learning and deep learning, required the input to be a CIF (Crystallographic Information Framework) and not in a matrix format. A CIF file is an electronic file about specific data items relevant to crystallographic structure determinations and descriptions [36].

The reason that this model was chosen for experimentation was because it used a different form of data, that is a graphical and not tabular form of data. This would help gain insights into the contribution of the crystal structure on the magnetic orientation of a material. Secondly, this model had a proven reliable estimation of density-functional theory (DFT) calculations.

6.2.5 Materials Graph Networks (MEGNET)

Another published model that was used was the MEGNET (Materials Graph Network) Model which was used for property prediction of molecules and crystals [22].

6.3 Reasons for Execution of Three Predictive Techniques

The author chose to execute three different models, classical machine learning, deep learning and convolutional neural networks so that there could be a wider selection range to choose the model that works the best to address the problem. Moreover, approaching the problem from different angles would help gain a deeper understanding of the problem and the underlying issues at hand. The difference in the data needed to execute the different models, CIF for CGCNN and tabular data for machine learning and deep learning, would also help explore the importance of a wider range of material properties in the prediction of the magnetic orientation. We also note that each of the three models have different advantages and disadvantages. For instance, classical machine learning is much more interpretable that

deep learning and deep learning learns high-level features in an incremental manner eliminating the need for domain expertise. Hence, using all three models gives us much more power and efficiency than just using any one of them. Another reason that motivated the author to implement three different models is that future researchers working in this area will have the choice and flexibility to reuse or modify any of the three models that is best suited to the application they have in mind.

6.4 Code

The code used for obtaining the results can be found in this github repository: `https://github.com/RiyaBOT/ME4101A-SethRiyaNimish`

The code has been written in three different languages, Python, Html and Matlab. A brief description of the code can be seen below. Detailed instructions to use the code can be found in the readme.md page of the github repository. After the execution of the Jupyter Notebooks for data preparation, machine learning, deep learning and so on, a document is automatically saved in the folder. These documents contain detailed results of everything that occurred during the execution of the particular code.

Most of the code used was written by the author. The code written in the colour red has been entirely written by the author from scratch. The code in blue has been borrowed from a source and modified by the author to suit the specific project at hand, the code in colour black has been entirely borrowed from a source. Appropriate citations can be found within the code.

- Data Preparation.ipynb: Jupyter Notebook used for data preparation
- Machine Learning.ipynb: Jupyter Notebook used for conducting classical machine learning
- Deep Learning.ipynb: Jupyter Notebook used for conducting deep learning algorithms.
- CGCNN

  - CIF Data
  - Crystal Graph Convolutional Neural Netwroks.ipynb: Jupyter Notebook used for optimising and predicting the accuracy of CGCNN
  - Miscellaneous files such as the model, predict.py written by [21].

- Web Interface
  - model.pkl: Best model obtained during training to classify materials into magnetic and non-magnetic.
  - model2.pkl: Best model obtained during training to classify magnetic materials into ferromagnetic and anti-ferromagnetic materials.
  - averages.csv: Stores the average value of the data stored.
  - data.csv: Stores the data used for training the above models
  - app.py: Helps in reading the input from the file.
  - index.html: Stores the format of the app
  - mlmodel.py: Helps in the facilitation of app.py
  - static: Contains the fonts used in the web interface

- Documents
  - DFDP2DMatpedia.docx: Document containing results after data featurizing on 2DMatpedia.
  - DFDPC2DB.docx: Document containing results after data featurizing on C2DB.
  - ML.docx: Example Document containing results after Machine Learning.
  - DL.docx: Example Document containing results after Deep Learning.
- Pictures: Contains various graphics which are used in the jupyter notebooks
- Requirements Text: Contains the Python libraries which are needed to execute my code.
- Data:
  - Miscellaneous Data: Data(NM/M)for ML, DL
  - CIF Data for CGCNN
  - DatawithFM/AFMClassification.csv
  - Data.json: Ajson file obtained from 2Dmatpdia
  - c2db.db: Adatabase obtained from computational 2d materials databse
- Surface Plots: Matlab Code used for hyperparameter optimisation.
- MEGNET
  - Graph Network.ipynb: Jupyter Notebook to compute the accuracy of the MEGNET model.

7. Results and Discussion

In this chapter, the results obtained through the execution of the techniques described in methodology are discussed in detail. It should be noted that the results obtained here are a typical example of what occurs when the code written by the author is executed. Machine learning techniques are subject to a certain degree of randomness and hence each iteration of the code may result in marginally different results.

7.1 Results for Data Preparation

Here, the results for the first step in methodology, namely data preparation is described. Data preparation was conducted on two sets of data: 2DMatpedia and Computational 2D Materials Database.

7.1.1 2DMatpedia

- Shape of the data: Initially there were 29 columns and 6351 rows.
- Visualization of the Data: Among the 29 data columns, only 8 columns had numerical significance and their relation to each other can be seen in the heat matrix here. The darker the colour, the lesser the correlation between the features. It is crucial to ensure that the features are not correlated with each other since that would lower the performance of the machine learning models and would result in an incorrect feature importance map.

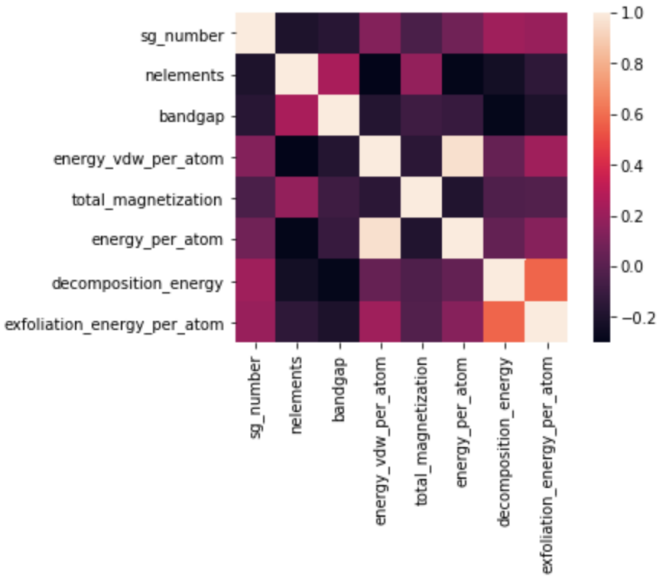

Figure 7.1: The heat matrix of the data obtained from 2DMatpedia.

As can be seen from Figure 7.1, the features, energy per atom and Van der Wall's energy per atom are closely related to each other; hence, the attribute Van der Wall's energy per atom was discarded.

- Data Featurizing: After featurizing the data using matminer and pymatgen, the number of data columns increased to 156 columns.

- Completeness of the Data: There were 6351 materials; however, 2130 data entries (rows) had one or more missing entries in the data columns. However, since the percentage of missing data was 32.16% which is significantly high, data imputation would lead to a biased data. Hence, data imputation was not implemented and rows with NaN were removed.

- Data Discretization: Number of bins created for discretization using Sturge's rule were 13.

- Data Normalisation: Data was normalised to a range of between 0 and 1.

- Initial ratio of data to error was 1.37 which was reduced to 0 after data cleaning.

7.1.2 Computational 2D Database

Since the materials in the Computational 2D Database are stored in the ASE database, which is used for storing and retrieving atoms and associated properties, the process of data featurizing was slightly different from that for 2DMatpedia. That is, the ASE Database uses calculators, which are black boxes that takes in atomic numbers and atomic positions to calculate the energy and forces. However, after using the calculators an excel sheet could be made. Data preparation took place with the same logic mentioned above in Section 7.1.1.

7.1.3 Discussion on the Reason for Use of the Above Databases

As with most scientific fields, meta-analysis of a range of databases and statistics is crucial in testing both the reliability and validity of data to be used for further application [37]. It is vital to not restrict a study to only one database but have multiple databases so that the model can achieve a greater degree of credibility and validation.

The reason why both these datasets are used is because both of them have quite a different set of features. For instance, in Computation 2D Materials Database there are features such as angular momentum, potential energy and so forth which are not present in 2DMatpedia. To get a better understanding of which features play a critical role in determining the magnetic properties of materials, both these datasets were chosen to be tested and their results analysed.

7.2 Results for Machine Learning

As can be seen in Figure 7.2, there were 16 different combinations of experiments which were considered and carried out. Some of these models did not perform with an accuracy of above 75% due to various reasons such as reduced data quality, unsuitable model selection and weak hyper-parameter optimisation. However, there were a considerable number of models which performed well with an accuracy of well over 90% and revealed meaningful insights into the problem at hand.

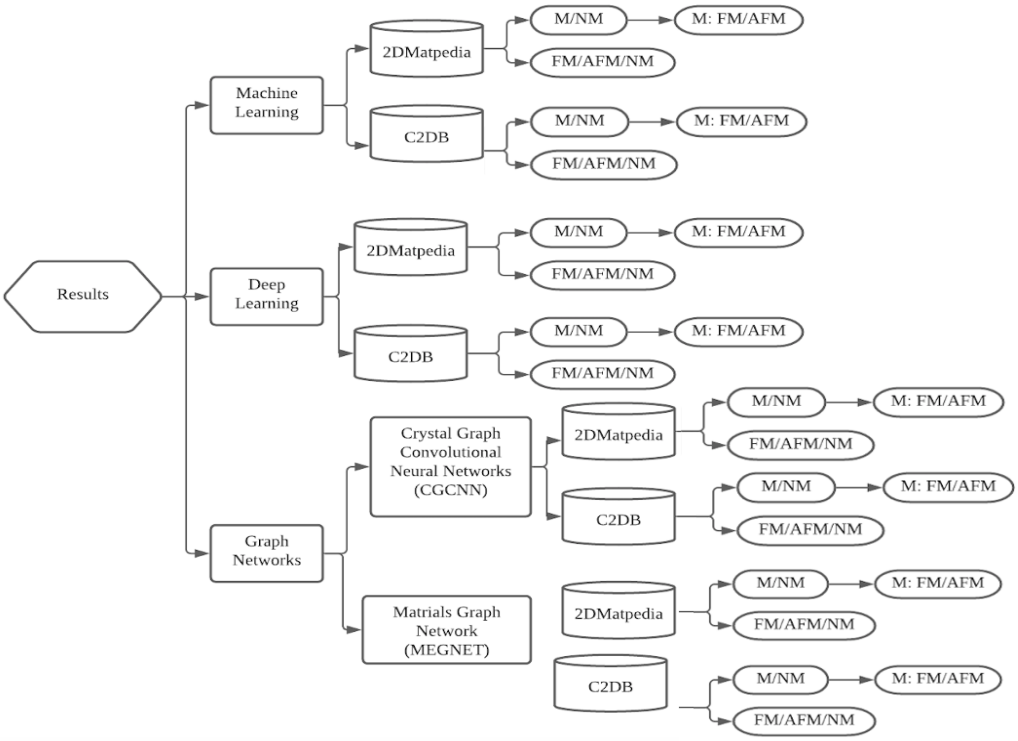

Figure 7.2: The workflow to predict the magnetic properties of a material. Here, M/NM means classification into magnetic and non-magnetic materials, M:FM/AFM, implies that the magnetic material was classified into ferromagnetic and anti-ferromagnetic materials. FM/AFM/NM means classification into ferromagnetic, anti-ferromagnetic and non-magnetic material.

7.3 Results for Classical Machine learning

2DMatpedia

Classification of a two-dimensional material into a magnetic or a non-magnetic material

After conducting experiments by varying different parameters of the machine learning algorithm, the best results were obtained for the ensemble classifier, gradient boost classifier for a training percentage of 80%, with accuracy of 93.37%.

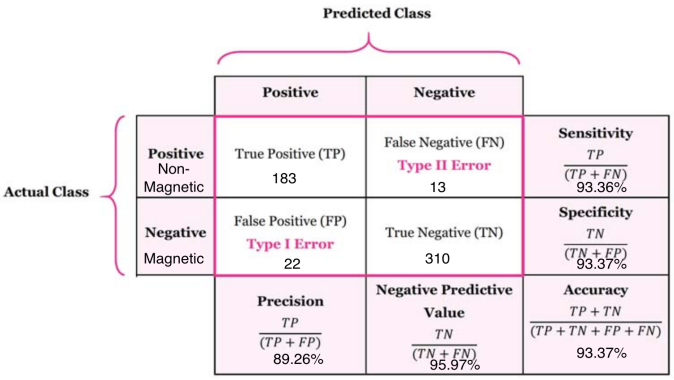

Figure 7.3: The confusion matrix of the results obtained after machine learning using the gradient boost classifier [4].

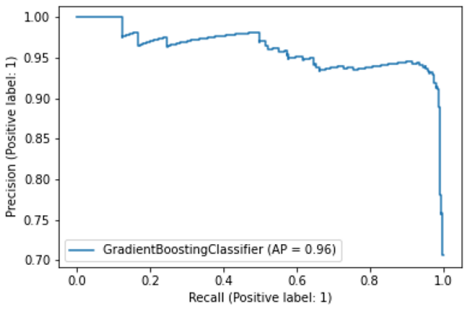

Figure 7.4: The ROC curve.

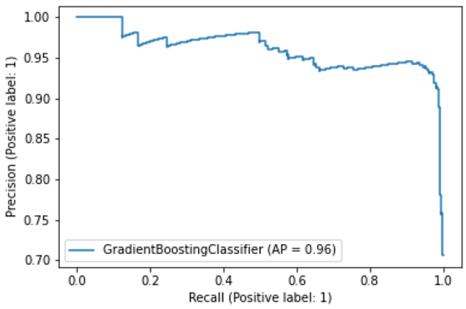

Figure 7.5: The precision recall curve.

The root mean square error (RMSE) is the standard deviation of errors which occurs when a prediction is made on a dataset. The RMSE is 0.257

The most important features are listed here: Band Gap, Number of Magnetic Atoms, Maximum Ionic Character, Presence of Mn and Presence of Cr.

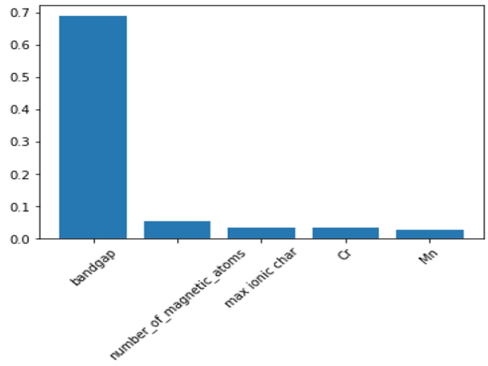

Figure 7.6: The five important features in the prediction of the magnetic properties of a material.

The results for this model are promising since an accuracy of 93.37% is high and its corresponding confusion matrix also shows low false negatives and false positives. While there is no established threshold for a good RMSE, considering that data ranges between 100 units, the RMSE of 0.257 is significantly small. Further, the RMSE for the testing dataset and the training dataset were similar which demonstrates that this model was not over-fit. Some addition performance metrics are as follows:

- **Sensitivity** is the ability of the model to correctly classify a material as non-magnetic.
- **Specificity** is the ability of the model to correctly classify a material as magnetic.
- **Negative Predictive Value** is the percentage of materials classified as non-magnetic among those materials which are actually non-magnetic.
- **Precision** is the percentage of materials classified as magnetic among those materials which are actually magnetic.

In Table 7.1, the best results for the metrics described above are shown.

Metric	Best Value	Algorithm	Training Ratio
Sensitivity	95.23%	Gradient Boost Classifier	90%
Specificity	94.86%	Gradient Boost Classifier	80%
Negative Predictive Value	100%	Gaussian Naive Bayers	80%
Precision	91.5%	Gradient Boost Classifier	80%

Table 7.1: Best metrics obtained in the machine learning model for 2DMatpedia.

Explainability

While the traditional feature importances plot as seen in Figure 7.6 help determine which feature has a greater impact on the model, the SHAP values in Figure 7.7 help determine the positive or negative impact that a particular feature has on the model. The plot below sorts features by the sum of SHAP value magnitudes over all samples, and uses SHAP values to show the distribution of the impacts each feature has on the model output [34]. The colour represents the feature value, red indicates high while blue indicates low. This graph reveals for example that a high number of magnetic atoms has the largest SHAP values, thereby having the largest contribution in terms of determining the magnetic orientation of a two-dimensional material.

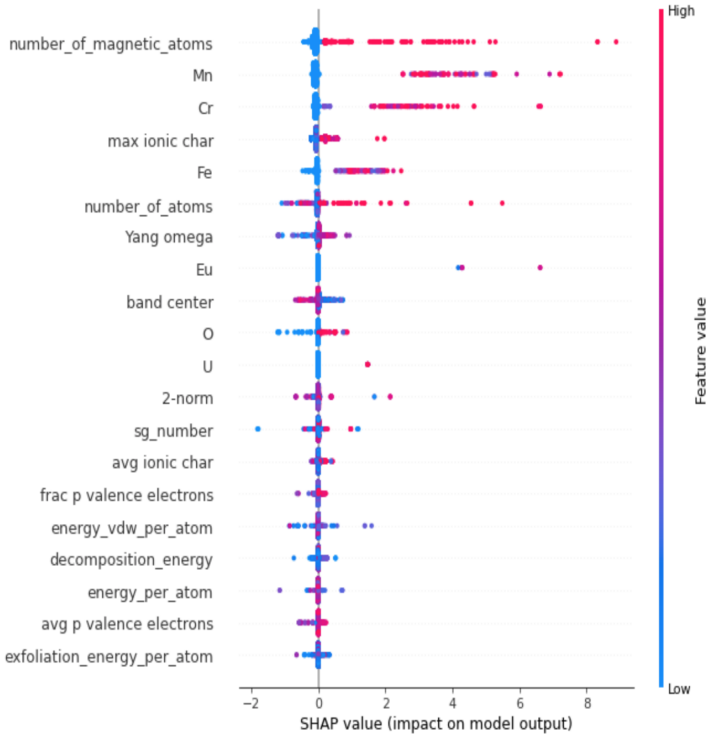

Figure 7.7: The SHAP values of some of the most important features of this model.

The below explanation shows features each contributing to push the model output from the base value (the average model output over the training data-set that was passed) to the model output. Features pushing the prediction higher are shown in red, those pushing the prediction lower are in blue. This is another more advanced metric which can show the numeric SHAP values for the training data. In this particular example of a training data entry, it can be seen that the absence of a magnetic atom pushed the prediction low by about 0.8. The maximum ionic character value of 0.2857 also was responsible for a contribution to make the prediction lower or in the direction of non-magnetism.

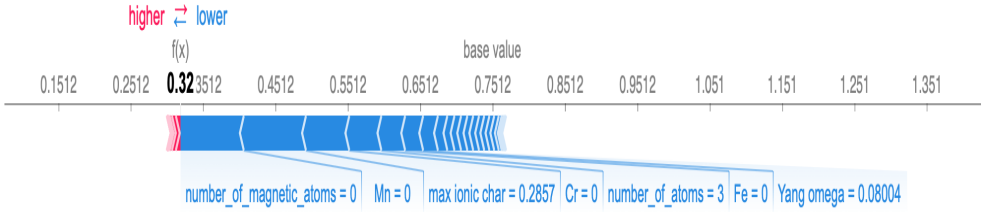

Figure 7.8: The predictive power of each of the features along with the side they push the prediction, i.e the magnetic or the non-magnetic side.

To understand how a single feature affects the output of the model, the SHAP value of that feature versus the value of the feature for all data points is plotted. This is also known as the feature dependence plot.

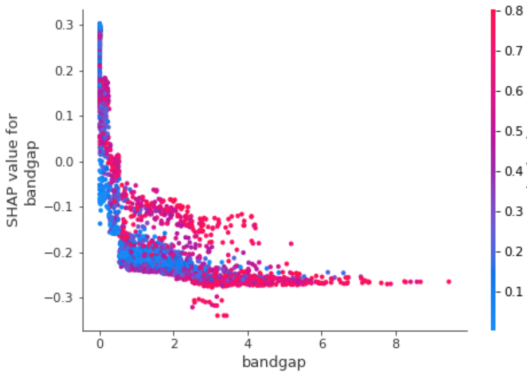

Figure 7.9: Example of how the band gap affects the SHAP values.

Here, we can see that the value of a low band gap gives the highest SHAP values and the higher the SHAP values, the greater the impact of the band gap on the prediction of the magnetic orientation of the material. After analysing how the model works at the larger level, it is imperative to look at how the prediction for each individual data entry was made. The figures below show the reasoning behind the classification of the compound MoS_2 as a non-magnetic material.

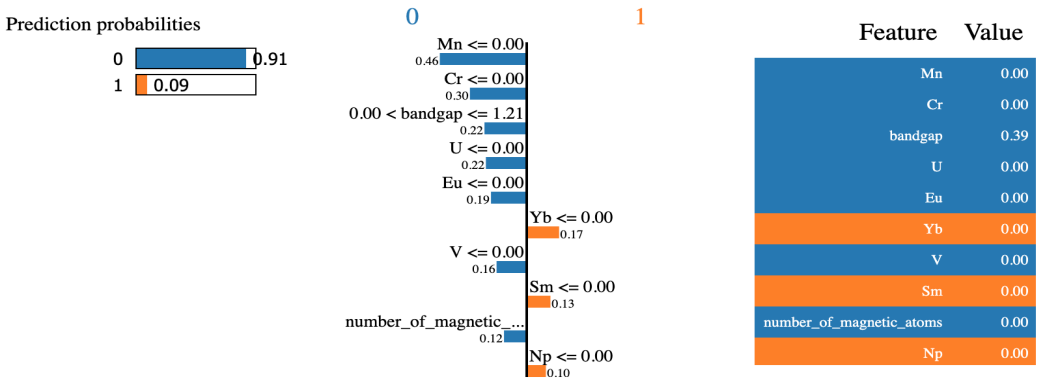

Figure 7.10: The explainability of the choice of prediction for the compound MoS_2 using LIME.

From an individual analysis, we can see the reasoning behind the classification of a particular material into a category. The blue columns drive for a non-magnetic classification while the orange drive for a magnetic classification. Similar results can be obtained for any other material to discern the reasons behind the prediction made by the model.

Threshold of Magnetic Moment

In the previous case, a material was only classified as non-magnetic if its magnetic moment was exactly equal to 0. However, to improve the certainty of the model, the threshold was changed from 0 to $\pm$ 0.5, followed by $\pm$1 and so on. This is so that the number of true-negatives (magnetic materials) could equal the exact number of materials classified as magnetic in the labelled data-set. That is, if the model predicted that a material is magnetic it most certainly is magnetic however, if it is predicted that the material is non-magnetic it could be either non-magnetic or magnetic. However, none of the model metrics gave a result of 100% with the gradient boost classifier and a training ratio of 80%. Hence, the models obtained here were not put to further use.

Threshold	Accuracy	Sensitivity	Specificity	NPV	Precision
0.5	88.1%	89%	82.7%	57.7%	96.6%
1	89.8%	90.7%	83.7%	59.6%	97.2%
2	93.2%	94%	83.3%	54.7%	98.5%
3	94.7%	95.5%	84.6%	60%	98.7%
4	94.9%	96.2%	78.9%	61.2%	98.3%
5	96.5%	97.4%	76.2%	55.2%	99%

Table 7.2: Results with different thresholds of magnetic moment for classification.

Classification of a magnetic material into ferromagnetic material and anti-ferromagnetic material

Using the resources from the following paper [12], a database for anti-ferromagnetic and ferromagnetic materials was created. The best results were obtained for the algorithm Random Forest Classifier with an accuracy score of 94.44%. The four most important features in determining whether the material was ferromagnetic or anti-ferromagnetic were the average ionic character, the maximum ionic character, yang omega and band centre. However, the size of the database was small and the database predominantly comprised of ferromagnetic materials unlike 2DMatpedia, where the number of magnetic materials and the number of non-magnetic material were similar. Hence, this model could not validated enough to achieve the desired credibility.

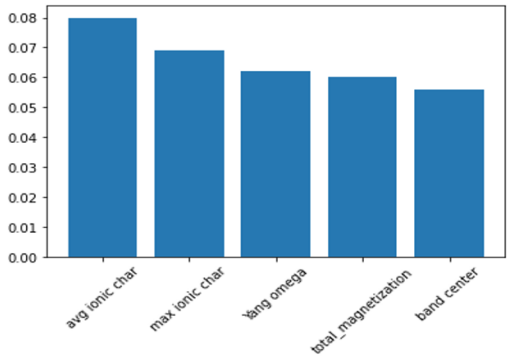

Figure 7.11: The feature importance for classification of a magnetic material into ferromagnetic and anti-ferromagnetic materials.

Domain Knowledge

Machine learning models are incomplete without a sound understanding of the domain knowledge. Here, it can be realised that if the band gap of a material is known from before hand, it is not very difficult to obtain the magnetic properties. Hence, instead of getting a database as it is and only deleting the output feature, it is important to delete the qualities which are computed using the same experiments that can theoretically compute the output feature as well.

To better understand what features are important to the model and better gauge the accuracy of the model, the features which need the same experimentation as is needed to predict the magnetic properties should be discarded. In this case, the band gap data entry was deleted. This reduced the accuracy of the gradient boost classifier model to 82.38%. The most important features were then the average *p*- valence electrons, decomposition energy, number of magnetic atoms, energy per atom and band center while the mean squared error was about 0.42.

Reason why AFM/FM/M did not work well

The reason this did not work well is because there was not enough data which suited it. We had about 2612 non-magnetic elements and 3919 magnetic elements of which only 87 magnetic materials could actually be labelled into AFM and FM. Consequently, the data distribution was thoroughly unequal. Hence, the results obtained for this model, besides being substantially lower than the accuracy of the above model, did not result in a credible model.

Computational Two-Dimensional Materials Database For the C2DB, the highest accuracy obtained was 89.73% for the gradient boost classifier. The confusion matrix is seen in Fig. The sensitivity of the model was 0.90, the precision was 96.67, the specificity was 83.6 and the negative predictive value was 63.75. The mean squared error was 0.32. The feature importances were similar to the results obtained for the model in 2DMatpedia. They shared common features such as the

presence of manganese, presence of chromium, number of magnetic atoms, vanadium and frac s valence electrons. While the negative predictive value and the precision

Metric	Best Value	Algorithm	Training Ratio
Sensitivity	95.23%	Gradient Boost Classifier	90%
Specificity	94.86%	Gradient Boost Classifier	80%
Negative Predictive Value	100%	Random Forest Classifier	90%
Precision	100%	Random Forest Classifier	90%

Table 7.3: Best metrics obtained in the machine learning model for C2DB

were better using the C2DB database, the overall performance of 2DMatpedia was better.

7.4 Results for Deep Learning

There are a number of ways that the optimisation of a model can be done: using experimentation, using intuition, going for depth or borrowing ideas.

Sequential API

Initially, the general notion of 'deeper the layers and greater the number of nodes, the better the performance' was used. However, when this was experimented it was found that this statement was not entirely true. Hence, an experimentation pattern was utilised. An individual optimisation of parameters was conducted as can be seen in Figures 7.12-7.14.

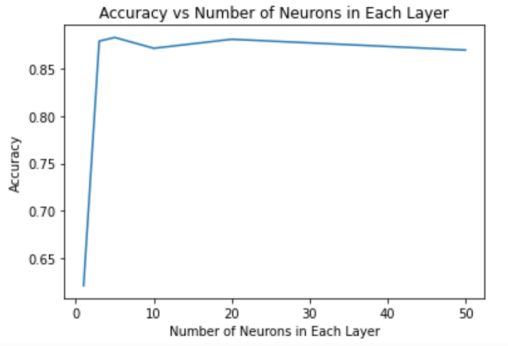

Figure 7.12: Accuracy vs number of neurons in each layer.

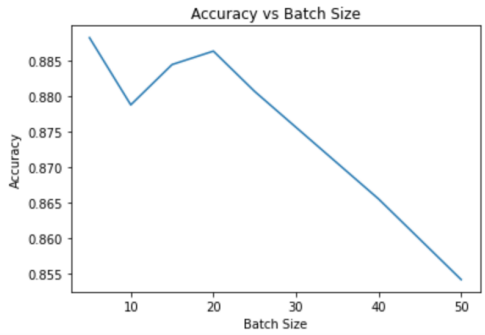

Figure 7.13: Accuracy vs batch size.

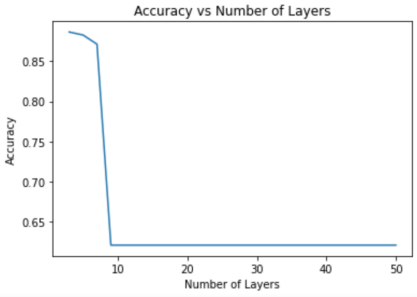

Figure 7.14: Accuracy vs number of layers.

After this individual optimisation it was realised that to obtain the correct optimisation results, it was imperative that the three parameters were changed at the same time. The results can be seen in the 3D graph below with the yellow points being the ones with the higher accuracy. This 3D plot was obtained through Matlab. If the computational power of the CPU had been stronger, then a larger number of points could have been tested.

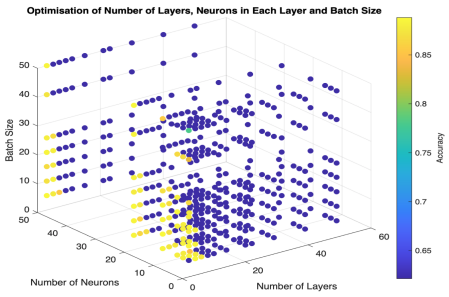

Figure 7.15: Optimisation of batch size, the number of neurons and the number of layers.

The best results were obtained for the 3 layers, 50 neurons and 5 batch-size. The average accuracy after 30 epochs was 88% and it peaked to 89.5%.

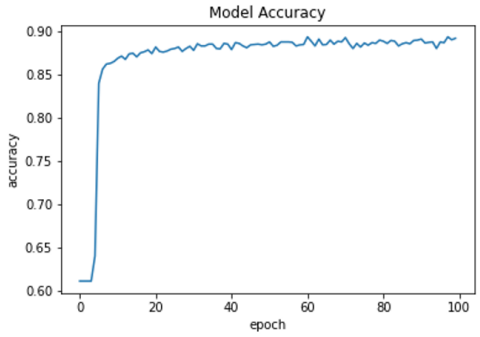

Figure 7.16: The best model accuracy.

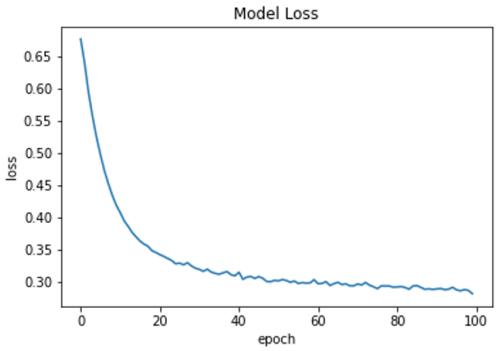

Figure 7.17: The best model loss.

Explainability

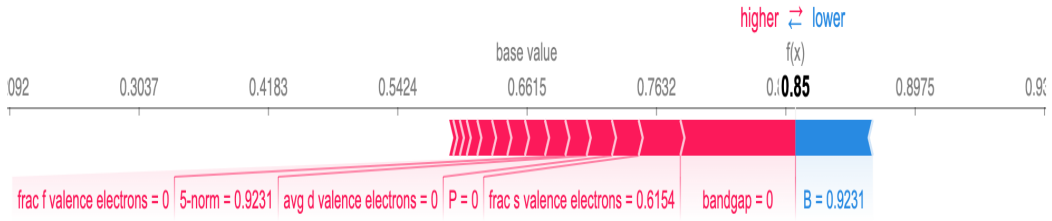

Figure 7.18: The predictive power of each of the feature and to which side they push the prediction, i.e towards the magnetic side or the non-magnetic.

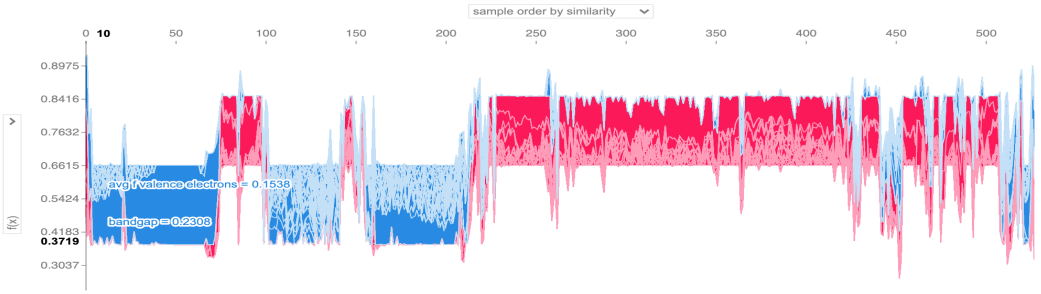

Figure 7.19: Force plot explanations of the entire dataset.

Figure 7.19 shows the stacked SHAP explanations clustered by explanation similarity. Each position on the x-axis is an instance of the data. Red SHAP values increase the prediction, blue values decrease it. This shows better functionality in the Jupyter Notebook has the cursor needs to be moved around the graph to draw results and explanations.

Functional API

The individual and collective optimisation was done in the same way as a sequential model.

This model has 2 dense layers of 64 nodes each. The activation functions were "relu". The accuracy of the model was 86.9% while the loss of the model was 0.46

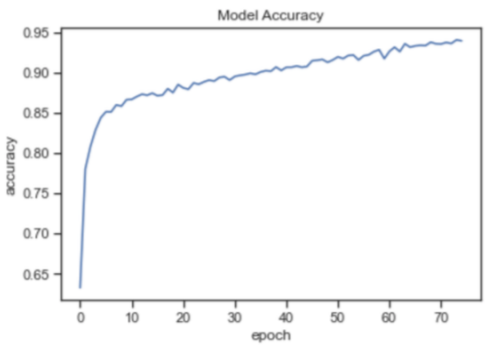

Figure 7.20: The epoch accuracy of the model

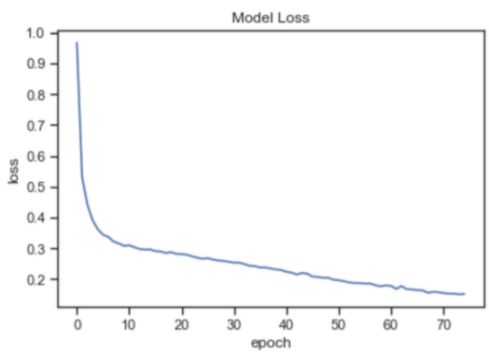

Figure 7.21: The epoch loss of the model

Analysis

The accuracy for the sequential model was significantly better than the functional API model but marginally lower than the machine learning model.

One of the improvements of this model is that a lot of parameters such as the batch size, the number of epochs, the number of dense layers, the nodes in each of these dense layers and the activation functions of the layers could be changed. A for loop which changes each of the parameters to determine the best parameter combination will take considerably long. Hence, a systematic experimentation pattern for parameter optimisation needs to be devised.

7.5 Results Obtained Using Published Models

7.5.1 Crystal Graph Convolutional Neural Networks

For the crystal graph convolutional method to work the hyperparameters need to be optimised. Hyperparameters are important because they directly control the behaviour of the training algorithm and have a significant impact on the performance of the model being trained. With a good choice of parameters the algorithm can perform better.

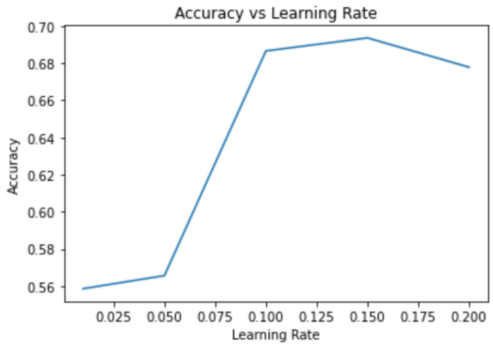

Figure 7.22: The change in the model accuracy with a change in the learning rate.

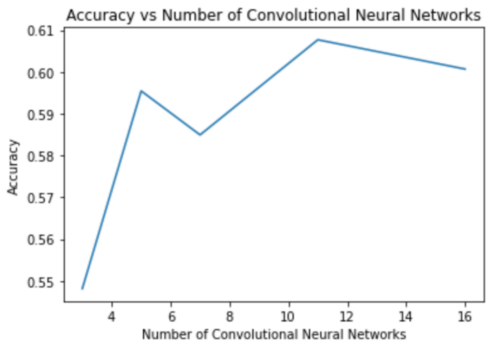

Figure 7.23: The change in the model accuracy with a change in the convolutional neural networks.

The execution of CGCNN resulted in a probabilistic classification where the output was the probability that the material was magnetic. Hence, to ensure the stability of the results which have a degree of randomness, 100 trials of deep learning were performed. The results can be seen below. The results for CGCNN averaged at around 68% and peaked to about 73%. Using the above optimised parameters and performing CGCNN again, we get a slightly improved performance. The results averaged at around 71% and peaked to about 80%.

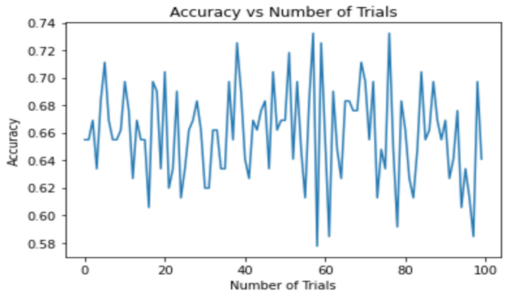

Figure 7.24: The average model accuracy before optimisation.

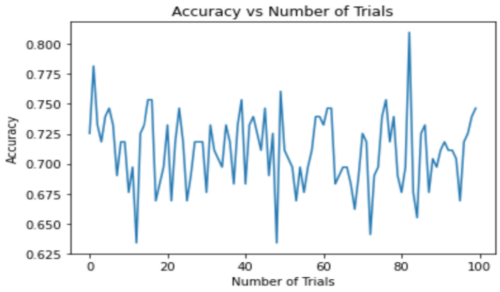

Figure 7.25: The average model accuracy after optimisation.

A reason why the optimisation and results for CGCNN could not be done efficiently is because the time taken for the execution of n number of epochs increased exponentially as seen in the graph below.

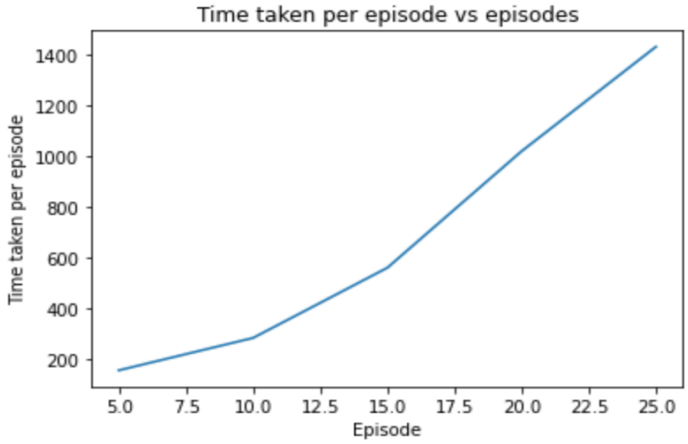

Figure 7.26: Time taken per episode vs number of epochs.

The results obtained from this model were very weak compared to classical machine learning and deep learning.

7.5.2 Materials Graph Networks

This model had very poor results with an accuracy of around 52%. The model performance did not improve even after the fine-tuning the parameters and increasing the number of epochs. A possible reason for this could be under-fitting of the model which results in poor performance of the training data. With a computationally stronger computer, adding more neuron layers and input parameters, adding more training samples and decreasing the regularization parameter, could be tried to check if the performance improves.

7.6 Web Application

After the completion of the machine learning and deep learning models, it is important to deploy the models so that it is available to the end-users. In this case, the models are deployed via web interfaces. With the web interfaces, users would be able to predict the results for their inputs without installing any models or downloading any code on their own computer.

The model is deployed such that other users can easily access them through an API (application programming interface) to make predictions. This was done using Flask and Heroku — Flask is a micro web framework that does not require particular tools or libraries to create web applications and Heroku is a cloud platform that can host web applications [38].

In this web application, 20 of the most important features were used as the input parameters. It is noted that some features may be unavailable to users and hence an option to enter NA has been provided. The reason that the model works with only the 20 features is because the other features have negligible contribution to

the model, as was understood after creating the feature importance plot of all the features. The link to the web application can be found here:
`https://twodferromagnetism-model.herokuapp.com`

7.7 Discussion

7.7.1 Possible Reasons for Better Classical Machine Learning Performance

Executing deep learning requires a large amount of data. In contrast, machine learning often works very effectively with significantly smaller data sets [39]. In our case the data used had a maximum size of 6531 from 2DMatpedia, which in comparison to other standard datasets is comparatively small. Moreover, while deep learning often needs substantial computational power (such as hardware accelerators including multiple GPU's, TPU's and FGPA's), machine learning algorithms work well with less computational power (such as a single CPU) [40]. In this case, the computer that was used was a laptop with a single CPU. Finally, deep learning algorithms are usually applied to problems where the data does not have a row-column structure, such as image classification or language translation. Since this is not the case with our given problem at hand, machine learning worked better than deep learning. The convolutional neural networks algorithm used here took in CIF files as data which was still tabular in format.

7.7.2 Possible Reasons for Better Gradient Boost Algorithm Performance

One of the main advantages of gradient boosting is that its predictive efficiency is substantially higher than that of other algorithms such as random forests [41]. Another advantage is its flexibility: it can optimize a wide variety of loss functions as

long as they are differentiable and it also provides a large range of hyper-parameter tuning options [41]. Because it is also an ensemble classifier which makes use of multiple machine learning models to make a prediction, the gradient boost algorithm worked better for this application. Furthermore, the efficacy of the gradient boost algorithm can be confirmed as machine learning on both the databases, C2DB and 2DMatpedia demonstrated that the gradient boost algorithm was the most efficient.

7.7.3 Discussion on Feature Importances Obtained for the Models

The feature importance in all the models have shown a significant portion of overlapping and thereby reinforcing which features are more important in determining the magnetic orientation of the material. A collated list of the most important features are used as input parameters in the web interface.

7.7.4 Domain Specific Explanation of the Results Obtained

From the above analysis we get a gauge on what chemical and physical features affect the prediction. The presence of magnetic elements greatly increases the chances of a two-dimensional material being magnetic. This is the case with a three-dimensional material as well. It can be seen that the properties in three-dimensional materials greatly translate to the properties in two-dimensional materials. The author is not well versed with chemical implications of a two-dimensional magnetic material, but the results obtained from here with suitable domain knowledge from an expert could be used to create new inferences and new points of view to the existing understanding of two-dimensional materials. Furthermore, inferences could be drawn between the relation of electronic configurations to magnetism and the band gap to magnetism as well.

8. Recommendations for Further Improvements

- Data Wastage

  An additional improvement that can be made is to prevent the wastage of data. We had 6531 data rows or materials; however a significant portion of them (2130 data rows) had a few missing data columns. Hence, these were removed since filling them up using data imputation would lead to a biased data. In the further works, if these 2130 data entries could be used as well such that if the most important features are not empty, the data could be used for the testing or validation of the model. This would make the model more credible with the optimum usage of data.

- Better Hyper-parameter Optimisation

  The current computer that was used for training did not have enough computational power that is needed for the thorough simultaneous hyper-parameter optimisation of the different parameters in deep learning. With GPUs or hardware accelerators that are computationally more powerful, the model could also be trained for a larger number of epochs with optimised hyper-parameters which would significantly improve the performance of the model.

- Improved Quality of Ferromagnetic and Anti-ferromagnetic Datasets

  Currently, the datasets for material classification into magnetic and non-magnetic materials are very comprehensive. However, the data-sets for classification of materials into ferromagnetic and anti-ferromagnetic materials are not of good quality. There is a huge number of ferromagnetic materials in comparison to anti-ferromagnetic materials which biases the model to predict ferromagnetism more often. However, with a suitable addition of more data, anti-ferromagnetic materials in specific could make the model more informed and accurate.

- Automatic Trend Detection

  In the problem at hand, the data is complicated and huge with numerous data entries and data features. For example, the data obtained from 2DMatpedia had 6531 data entries and 156 columns. To analyse the cluster plots between any two features 12090 graphs would need to be checked manually. Hence, there should be an automated way to identify the relation between features to make sure they are not related linearly, quadratically, exponentially etc. Moreover, during the explainability of the graphs as seen in feature dependence plot in Figure 7.9, it would be effective if the relation of each of the features with the model could be interpreted to better understand the model. That would however result in 156 graphs to be analysed. Hence, an incorporation of the automatic detection of trends and patterns in the graph would provide more insights into the model and help the user better understand the results.

- Regression Modelling

  So far a material has been classified as magnetic or non-magnetic. Moving forward, it would be useful if the model could be trained to the extent that it could be used to predict the actual value of the magnetic moment.

9. Conclusions

In summary, among the three different techniques used to screen out two-dimensional ferromagnetic materials from two-dimensional material databases, the classical or non-deep machine learning technique with the gradient boost algorithm gave the most accurate results. Among the two databases used, namely 2DMatpedia and C2DB, the model that learned from the data in 2DMatpedia gave better results with an accuracy of 93.37% to classify a material into a magnetic material and a non-magnetic material and an accuracy of 94.44% to classify a magnetic material into a ferromagnetic and an anti-ferromagnetic material. The main features contributing to the prediction made by the model were band gap, number of magnetic atoms, maximum ionic character, presence of Mn and presence of Cr. Through the thorough investigation of the model reliability and interpretability, it can be established that this model serves as a strong candidate to be used as an initial filtration process during the systematic search of two-dimensional ferromagnetic materials.

References

[1] C. Gong and X. Zhang, "Two-dimensional magnetic crystals and emergent heterostructure devices," *Science*, vol. 363, no. 6428, 2019. [Online]. Available: https://science.sciencemag.org/content/363/6428/eaav4450

[2] Tkatchenko, A. Machine learning for chemical discovery. Nat Commun 11, 4125 (2020). https://doi.org/10.1038/s41467-020-17844-8.

[3] Mori, T., Uchihira, N. Balancing the trade-off between accuracy and interpretability in software defect prediction. Empir Software Eng 24, 779–825 (2019). https://doi.org/10.1007/s10664-018-9638-1.

[4] M. Sisrat, "What is confusion matrix and advanced classification metrics?" *Data Science and Machine Learning*, April 2019. [Online]. Available: https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

[5] C. Buzea, I. I. Pacheco, and K. Robbie, "Nanomaterials and nanoparticles: Sources and toxicity," *Biointerphases*, vol. 2, no. 4, pp. MR17–MR71, 2007. [Online]. Available: https://doi.org/10.1116/1.2815690

[6] "2d materials: An introduction to two-dimensional materials," 2019. [Online]. Available: https://www.ossila.com/pages/introduction-2d-materials

[7] Z. Rui and C. Rebecca, "Mechanical properties and applications of two-dimensional materials," 2016. [Online]. Available: https://www.intechopen.com/books/two-dimensional-materials-synthesis-characterization-and-potential-applications/mechanical-properties-and-applications-of-two-dimensional-materials

[8] Vertikasari P. Ningrum, Bowen Liu, Wei Wang, Yao Yin, Yi Cao, Chenyang Zha, Hongguang Xie, Xiaohong Jiang, Yan Sun, Sichen Qin, Xiaolong Chen, Tianshi Qin, Chao Zhu, Lin Wang, Wei Huang, Recent Advances

in Two-Dimensional Magnets: Physics and Devices towards Spintronic Applications, Research, vol. 2020, Article ID 1768918, 19 pages, 2020. https://doi.org/10.34133/2020/1768918.

[9] M. Piquemal-Banci, R. Galceran, M.-B. Martin, F. Godel, A. Anane, F. Petroff, B. Dlubak, and P. Seneor, "2d-MTJs: introducing 2d materials in magnetic tunnel junctions," *Journal of Physics D: Applied Physics*, vol. 50, no. 20, p. 203002, apr 2017. [Online]. Available: https://doi.org/10.1088/1361-6463/aa650f

[10] P. Huang, P. Zhang, S. Xu, H. Wang, X. Zhang, and H. Zhang, "Recent advances in two-dimensional ferromagnetism: materials synthesis, physical properties and device applications," *Nanoscale*, vol. 12, pp. 309–2327, 2020. [Online]. Available: http://dx.doi.org/10.1039/C9NR08890C

[11] X. Zhang, B. Wang, Y. Guo, Y. Zhang, Y. Chen, and J. Wang, "High curie temperature and intrinsic ferromagnetic half-metallicity in two-dimensional nanosheets."

[12] K. M. . M. S. Kabiraj, A., "High-throughput discovery of high Curie point two-dimensional ferromagnetic materials." *Comput Mater*, vol. 35, 2020. [Online]. Available: https://doi.org/10.1038/s41524-020-0300-2

[13] C. Gong, L. Li, Z. Li, H. Ji, A. Stern, Y. Xia, T. Cao, W. Bao, C. Wang, Y. Wang, Z. Qiu, R. Cava, S. Louie, J. Xia, and X. Zhang, "Discovery of intrinsic ferromagnetism in two-dimensional van der waals crystals," *Nature*, vol. 546, 04 2017.

[14] Y.-C. Lo, S. E. Rensi, W. Torng, and R. B. Altman, "Machine learning in chemoinformatics and drug discovery," *Drug Discovery Today*, vol. 23, no. 8, pp. 1538–1546, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1359644617304695

[15] M. de Jong, W. Chen, R. Notestine, K. Persson, G. Ceder, A. Jain, M. Asta, and A. Gamst, "A statistical learning framework for materials science: Application to elastic moduli of k-nary inorganic polycrystalline compounds," Sci. Rep. 6, 15004 (2016). https://doi.org/10.1038/srep34256.

[16] M. Hellenbrandt, "The inorganic crystal structure database (icsd)—present and future," *Crystallography Reviews*, vol. 10, no. 1, pp. 17–22, 2004. [Online]. Available: https://doi.org/10.1080/08893110410001664882

[17] L. C. M. Zhou, J Shen, "2dmatpedia, an open computational database of two-dimensional materials from top-down and bottom-up approaches." *Sci Data 6*, no. 86, 2019.

[18] S. Haastrup, M. Strange, M. Pandey, T. Deilmann, P. S. Schmidt, N. F. Hinsche, M. N. Gjerding, D. Torelli, P. M. Larsen, A. C. Riis-Jensen, and et al., "The computational 2d materials database: high-throughput modeling and discovery of atomically thin crystals," *2D Materials*, vol. 5, no. 4, p. 042002, Sep 2018. [Online]. Available: http://dx.doi.org/10.1088/2053-1583/aacfc1

[19] W. L. P. A. Jha, D., "Elemnet: Deep learning the chemistry of materials from only elemental composition." *Sci Rep*, vol. 8, no. 17593, Dec 2018. [Online]. Available: https://doi.org/10.1038/s41598-018-35934-y

[20] K. T. Schütt, P. Kessel, M. Gastegger, K. A. Nicoli, A. Tkatchenko, and K.-R. Müller, "Schnetpack: A deep learning toolbox for atomistic systems," *Journal of Chemical Theory and Computation*, vol. 15, no. 1, pp. 448–455, 2019. [Online]. Available: https://doi.org/10.1021/acs.jctc.8b00908

[21] T. Xie and J. C. Grossman, "Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties," *Physical Review Letters*, vol. 120, no. 14, Apr 2018. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.120.145301

[22] C. Chen, W. Ye, Y. Zuo, C. Zheng, and S. P. Ong, "Graph networks as a universal machine learning framework for molecules and crystals," *Chemistry of Materials*, vol. 31, no. 9, p. 3564–3572, Apr 2019. [Online]. Available: http://dx.doi.org/10.1021/acs.chemmater.9b01294

[23] Ward, L., Dunn, A., Faghaninia, A., Zimmermann, N. E. R., Bajaj, S., Wang, Q., Montoya, J. H., Chen, J., Bystrom, K., Dylla, M., Chard, K., Asta, M., Persson, K., Snyder, G. J., Foster, I., Jain, A., Matminer: An open source toolkit for materials data mining. Comput. Mater. Sci. 152, 60-69 (2018).

[24] Shyue Ping Ong, William Davidson Richards, Anubhav Jain, Geoffroy Hautier, Michael Kocher, Shreyas Cholia, Dan Gunter, Vincent Chevrier, Kristin A. Persson, Gerbrand Ceder. Python Materials Genomics (pymatgen) : A Robust, Open-Source Python Library for Materials Analysis. Computational Materials Science, 2013, 68, 314–319. doi:10.1016/j.commatsci.2012.10.028.

[25] "Decomposition energy," *Center for Chemical Process Society*. [Online]. Available: https://www.aiche.org/ccps/resources/glossary/process-safety-glossary/decomposition-energy

[26] S. Zhang, C. Zhang, and Q. Yang, "Data preparation for data mining," *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 375–381, 2003. [Online]. Available: https://doi.org/10.1080/713827180

[27] S. Asaithambi, "Why, how and when to apply feature selection," *Towards Data Science*, Jan 2018. [Online]. Available: https://towardsdatascience.com/why-how-and-when-to-apply-feature-selection-e9c69adfabf2

[28] S. Poddar, "Binning in data mining," *GeeksforGeeks*, Sept 2020. [Online]. Available: https://www.geeksforgeeks.org/binning-in-data-mining/

[29] U. Jaitly, "Why data normalization is necessary for machine learning models," *Medium*, Oct 2018. [Online]. Available: https://medium.com/@urvashilluniya/why-data-normalization-is-necessary-for-machine-learning-models-681b65a05029

[30] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, 2018.

[31] M. Valueva, N. Nagornov, P. Lyakhov, G. Valuev, and N. Chervyakov, "Application of the residue number system to reduce hardware costs of the convolutional neural network implementation," *Mathematics and Computers in Simulation*, vol. 177, pp. 232–243, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378475420301580

[32] J. Bronwlee, "A tour of machine learning algorithms," *Machine Learning Mastery*, 2019. [Online]. Available: https://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

[33] C. Molnar, *Interpretable Machine Learning*, 2019, https://christophm.github.io/interpretable-ml-book/.

[34] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," pp. 4765–4774, 2017. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[35] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," pp. 1135–1144, 2016.

[36] S. R. Hall, J. D. Westbrook, N. Spadaccini, I. D. Brown, H. J. Bernstein, and B. McMahon.

[37] "The benefits of using two datasets," *Ambiental Risk Analytics*. [Online]. Available: https://www.ambientalrisk.com/the-benefits-of-using-two-datasets/

[38] E. T. Sahakyan, "Create an api to deploy machine learning models using flask and heroku," *towards data science*, 2019. [Online]. Available: https://towardsdatascience.com/create-an-api-to-deploy-machine-learning-models-using-flask-and-heroku-67a011800c50

[39] N. Koleva, "When and when not to use deep learning," May 2020. [Online]. Available: https://blog.dataiku.com/when-and-when-not-to-use-deep-learning

[40] M. Heller, "Deep learning vs. machine learning: Understand the differences," January 2020. [Online]. Available: https://www.infoworld.com/article/3512245/deep-learning-vs-machine-learning-understand-the-differences.html?page=2

[41] B. Boehme, "UC Business Analytics R Programming Guide." [Online]. Available: http://uc-r.github.io/gbm_regression#proscons

10. Appendix

Since the data and code collectively take up about 100MB, they have been uploaded in a github repository and the links to those have been given here:

10.1 Code Availability:

The code used for obtaining the results can be found in this github repository:
`https://github.com/RiyaBOT/ME4101A-ShethRiyaNimish`
The link to the web application can be found here:
`https://twodferromagnetism-model.herokuapp.com`

10.2 Data Availability:

The various data used during the execution of the code can be found here:
`https://github.com/RiyaBOT/ME4101A-ShethRiyaNimish/tree/master/Data`