Mark Prediction Performance Analysis with Neural Network, Decision Tree, Maximum Likelihood and Evolutionary Algorithm More Advanced Technique

Haosen Yin

Research School of Computer Science, Australian National University
2601 Canberra, Australia
u6171603@anu.edu.au

Abstract. This paper discusses four different machine learning techniques to evaluate their performances on student marks dataset for a classification task. There are some problems facing such as data preprocessing, decision tree constructions, maximum likelihood implementations, neural network structure with hyperparameter tuning and evolutionary algorithms to select a best model. We will address them in the report step by step. Furthermore, the final results are used to compare with one another and the high-performance technique can be selected as an optimal solution to this classification problem. Future work is also mentioned at the end of the report.

Keywords: Machine Learning, Deep Learning, Data Mining, Neural Network, Decision Tree, Maximum Likelihood, Evolutionary Algorithm, Genetic Algorithm

1 Introduction

Mark prediction is to approximate the final examination mark based on the previous class assessments. It is a useful analysis to investigate some correlations between a student's performance in regular class assessments and the final examination. Instructors may learn the knowledge from this analysis that helps them adjust assessment contents and moreover change teaching methods during the session or for the next session. Students can also gain better learning experiences. For example, if the predicted student marks of the final examination mostly fall into the fail range, the instructors such as teaching team need to consider preparing an easy-level final examination due to the past unreasonably difficult assessments in case many people will complain about the course. Another example is that the instructors are able to build a more reasonable assessment framework from mark prediction analysis which leverages the difficulty of every assessment and gives insights how to design student assignments. The dataset given in the paper (Choi & Gedeon 1995) is a set of assessment marks in an undergraduate Computer Science subject and our goal is to predict the range of the final result such as Distinction (75 or above), Credit (65 to 74), Pass (50 to 64) and Fail (less than 50) that a student will receive. It weighs 60% of the total mark and the remaining 40% is from class assessments evaluated for the prediction. This is essentially a classification task. Therefore, we firstly explore, preprocess and profile the mark data. In the following sections, in order to predict the final examination mark, different machine learning techniques: neural networks, decision trees, maximum likelihood (Milne, Gedeon & Skidmore 1995) and more advanced evolutionary algorithms will be introduced and tested on the dataset in terms of performance in comparison to each other.

2 Method

2.1 Data Preprocessing

The raw data we have is not relatively easy to be converted into an expected Pandas dataframe because there exists many useless headings and characters in the first several lines about the file information, for example, created time. Skipping those rows while reading the file via Pandas inbuilt function is our first step. Moreover, missing values need to be resolved. The "." dots as missing values shown on this dataset are likely to be meaningful because they may interpret that students have applied for redeemable assessments and should be replaced with numeric 0 to reflect it in the other assessments or otherwise it cannot be handled over training and testing. In addition, some attributes provide no values to the final mark categorical prediction task such as student recognition number, course code and tutorial group so we can eliminate all irrelevant attributes affecting the performance potentially. An easy-to-process dataframe is created which contains 10 feature attributes lab2, tutass, lab4, h1, h2, lab7, p1, f1, mid, lab10 and 1 target attribute final. The target values have to be transferred as 4 classes for prediction: Distinction, Credit, Pass or Fail. One way we can encode them is to represent each class as 0, 1, 2 or 3 respectively. Since every assessment marking has its own scale such as 2/3, 3/5 2.5/3 and 19.5/20 (actual/total) and also the final marks are represented by 100 totally, other assessment marks apart from the final one need to be normalized into the same range which reduces value deviation. It is realized through a formula below (Formula 1).

$$\frac{x - \min(y)}{\max(y) - \min(y)}$$

where x is a student's mark for the specific assessment, y is all students' marks for the same specific assessment
(Formula 1: Class Assessment Mark Normalisation)

Consequently, a clear new processed dataframe can also be seen as below (Table 1).

lab2	tutass	lab4	h1	h2	lab7	p1	f1	mid	lab10	final

(Table 1: Processed Dataframe for Prediction Task)

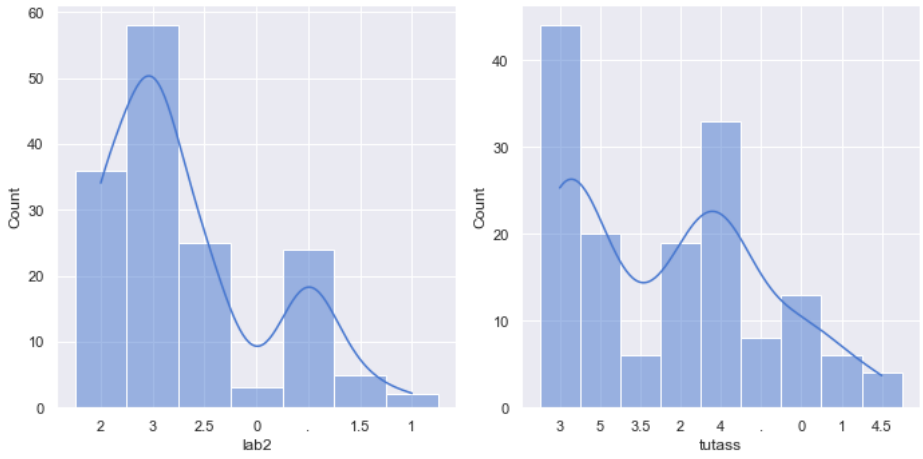
In summary, the raw dataset has been manipulated with appropriate modifications:

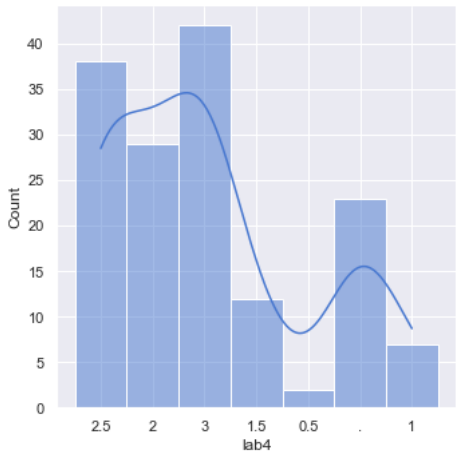
- Skip useless file information
- Resolve missing values that can have meanings
- Eliminate all irrelevant attributes
- Convert the numeric target attribute final into encoded 4 categories: Distinction, Credit, Pass or Fail
- Normalize other different course assessments.

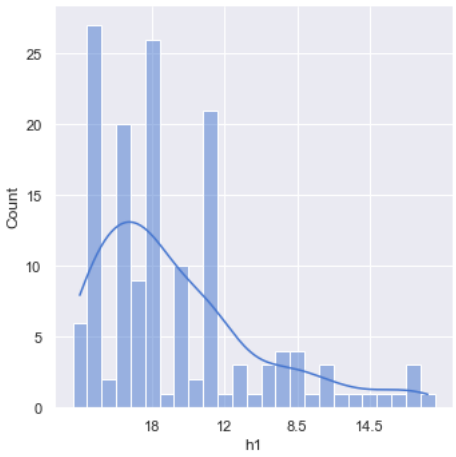
2.2 Data Exploration and Profiling

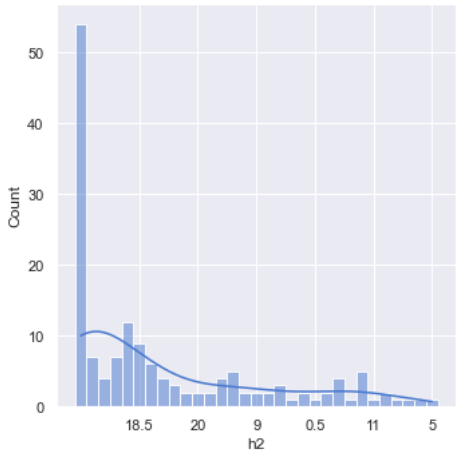
There are 153 data points in the processed dataframe and each data point combines the marks of multiple different assessments over the semester as illustrated in the data preprocessing section. For analysis purpose, we need to explore and profile the raw data in this section. The simple histograms with density curves are plotted below (Figure 1) and a Phik correlation map is also given (Figure 2). In the histograms, it is easy to see the distribution associated with one single attribute such that people can clearly know in which assessments students perform better at the first sight. The Phik correlation map shows how every attribute may have internal relations to each other on the basis of color density. For instance, final is highly correlated with other 8 attributes: mid, Tutgroup, Crse/Prog, tutass, lab4, ES, lab7 and h1. It means if a student achieved decent marks from these class assessments, they will be very likely to have a good performance in the final examination as well. Moreover, the particular tutorial group, the course or the program they belong to also has impact on the final examination marks which can imply some tutor or lecturer is good at teaching but we care about quantifiable continuous numeric features for this final mark prediction task due to the problem that a neural network cannot differentiate categorical and numeric values and it treats both of them in the same continuous space. Thus, the marks of lab2, tutass, lab4, h1, h2, lab7, p1, f1, mid and lab10 should be selected. To conclude briefly, our findings are listed as follows:

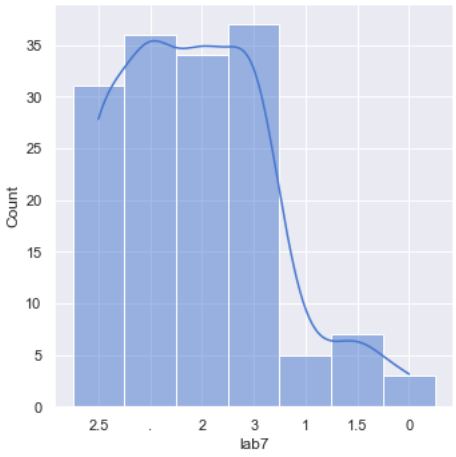
- Rough distributions of all attributes
- Correlations between every two attributes especially for final
- Use numeric features as input feeding the neural networks in the next part

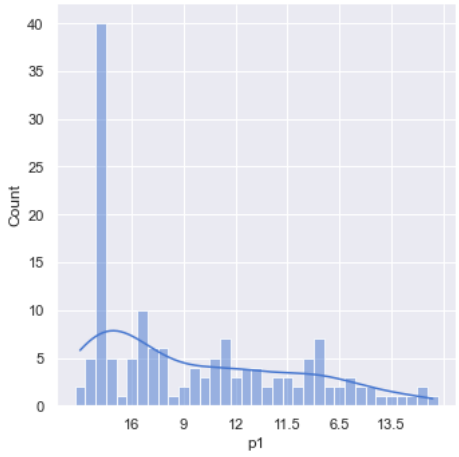


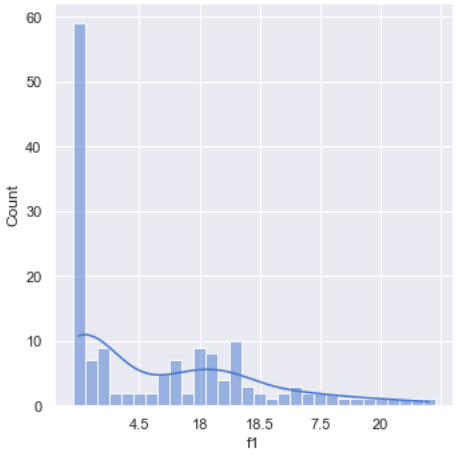

Count
lab4
2.5 2 3 1.5 0.5 . 1

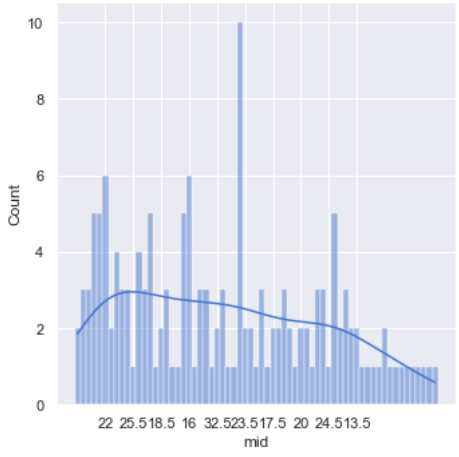

Count
h1
18 12 8.5 14.5

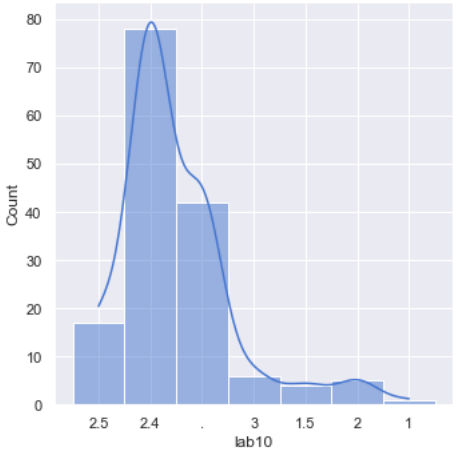

Count
h2
18.5 20 9 0.5 11 5

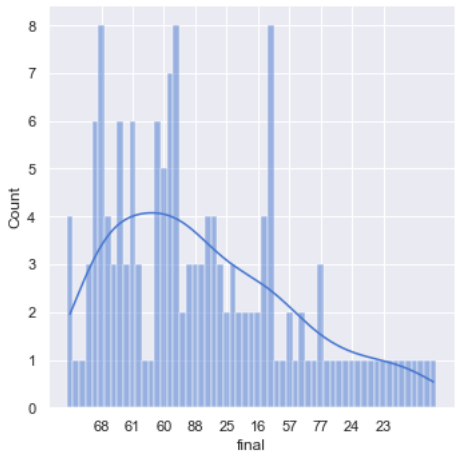

Count
lab7
2.5 . 2 3 1 1.5 0


Count
p1
16 9 12 11.5 6.5 13.5

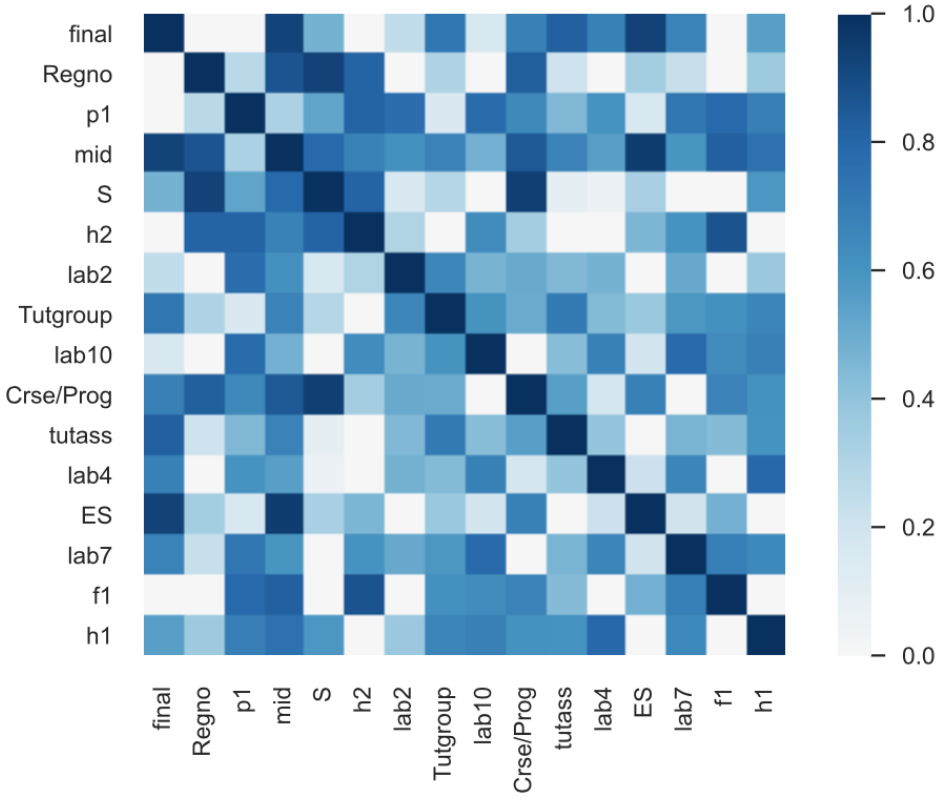

Count
f1
4.5 18 18.5 7.5 20


Count
mid
22 25.5 18.5 16 32.5 23.5 17.5 20 24.5 13.5


Count
lab10
2.5 2.4 . 3 1.5 2 1



(Figure 1: Histograms of Feature Attribute Data)

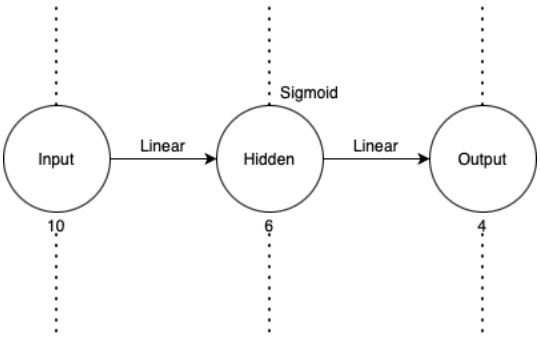


(Figure 2: Phik Correlation Density Map)

2.3 Neural Networks

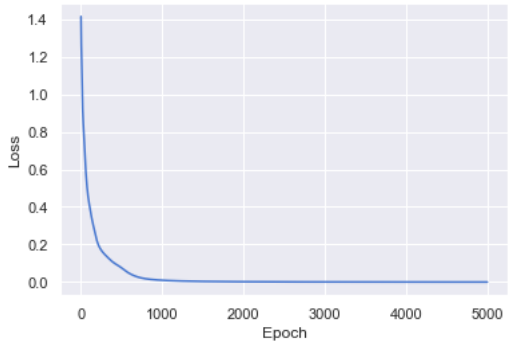
First, a very simple baseline neural network model was implemented (partially adapt from lab2_task_1_glass_binary.py, COMP4660) and that can compare to other techniques applied later for reference. This model has:

- 10 input features
- 2 layers including 1 hidden layer and 1 output layer
- 6 hidden neurons
- 4 output neurons
- 0.05 learning rate
- 5000 epochs
- Fully connected architecture

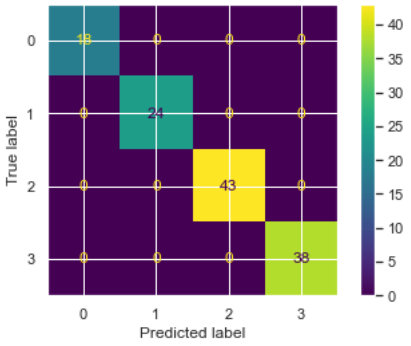


(Figure 3: Baseline Neural Network Model)

It is clearly described in the above illustration (Figure 3). The baseline model took only 10% of the time to develop but would give 90% reasonably good results (Ameisen 2018). We can build a more complex neural network in terms of improving the baseline network accuracy (Merity 2017). How to choose the hyperparameters for the model is mostly empirical. For example, the learning rate of the baseline model is set as 0.05 because it usually needs to start low and fine-tuning as training goes. We trained 80% data as the classic settings and the rest 20% of data was used for testing with cross entropy loss function for classification problem as well as Adam optimizer for faster convergence. Losses and also a confusion matrix were plotted after iterating 5000 epochs (Figure 4, 5).

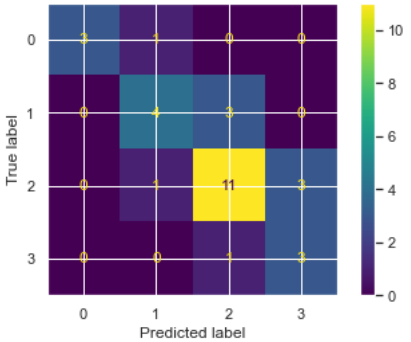


(Figure 4: Losses over Training Epoch Iterations)



(Figure 5: Confusion Matrix for Training)

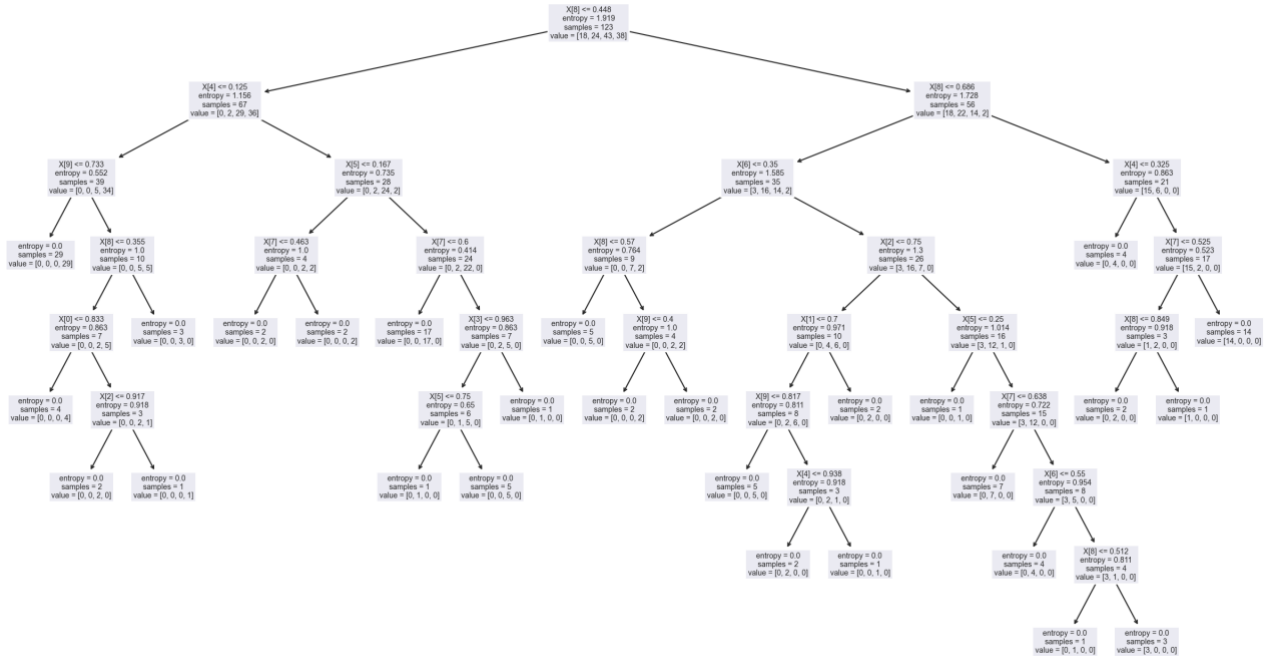
It is clear to see that the loss was reducing over the time while training data and reached at nearly 1000 epochs. A confusion matrix is also provided. They both look good but this may be due to fitting perfectly the data. We are required to apply the baseline model onto our testing data as well. The result shows that the test loss was 4.70 and accuracy was 70.00%. Furthermore, its confusion matrix (Figure 6) tells the performance was not as good as that on the training data so there is certainly space for improvements.



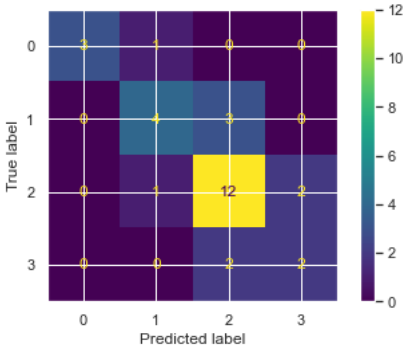
(Figure 6: Confusion Matrix for Testing)

2.4 Decision Trees

ID3 algorithm was used for generating a knowledge base from the dataset (Milne, Gedeon & Skidmore 1995). Therefore, we implemented a ID3 decision tree with Python Scikit-learn module. It helps us select the best fitting attributes with no manual calculations. The tree model is constructed shown below (Figure 7).



(Figure 7: ID3 Decision Tree Structure)



(Figure 8: Confusion Matrix for Decision Tree Testing)

A confusion matrix can also be seen above (Figure 8). On the testing data, the accuracy of this ID3 decision tree model was 73.33% slightly better than 70.00% from the baseline neural network.

3 Results and Discussion

Although some methods such as maximum likelihood classification and evolutionary algorithms are difficult to be implemented, the simple models we already generate still can give good performance in terms of accuracy in comparison to 53.8% for the neural network models and 58.8% for the extracted rules from the dataset paper (Choi & Gedeon 1995). However, evolutionary algorithms, or genetic algorithm which is the most popular type can be very helpful to our classification task because it helps learn the parameters of all the models mentioned above. Alternatively, an approach called Learning Classifier Systems (LCS) or Genetics Based Machine Learning (GBML) can evolve rule sets from the original dataset. These techniques need to take relatively long time and programming skills.

4 Conclusion and Future Work

As mentioned, we finished a baseline neural network, decision tree model and their performance results with full training and testing data. Other methods, maximum likelihood classification and evolutionary algorithms, they are only partly implemented so there is significant amount of research work to be done in the future. We hope it can be completed soon and publish the conclusion by comparing four different machine learning techniques on our student marks dataset.

References

Ameisen, Emmanuel. "Always Start with a Stupid Model, No Exceptions." *Medium*, Insight Fellows Program, (2018), http://blog.insightdatascience.com/always-start-with-a-stupid-model-no-exceptions-3a22314b9aaa.

Merity, Stephen. "Backing off towards simplicity – why baselines need more love." *O'Reilly AI San Francisco and Data Institute's Annual Conference*. (2017).

COMP4660, 2021, lab2_task1_glass_binary.py, https://wattlecourses.anu.edu.au/pluginfile.php/2558474/mod_folder/content/0/lab2_task1.py?forcedownload=1

Edwin Che Yiu CHOI & Tamas Domonkos GEDEON 1995, http://users.cecs.anu.edu.au/~Tom.Gedeon/pdfs/Comparison%20of%20Extracted%20Rules%20from%20Multiple%20Networks.pdf