Electrical Grid Stability Classification: A Comparative Study of Multilayer Perceptron and Support Vector Machine

Daniel Addai-Marnu

Daniel.Addai-Marnu@city.ac.uk

Abstract

This study aims to compare and evaluate the performance of Feedforward Multilayer Perceptron (MLP) and Support Vector Machines (SVM) algorithms in the classification of a sample of pulsar candidates collected during a High Time Resolution Universe survey. Different models are assessed using various hyper-parameter tuning techniques to arrive at the best models for each algorithm. The best models are critically evaluated to compare their performance on the classification task. Both algorithms performed very well with MLP performing slightly better than SVM in classifying electric grid stability.

1.0 Introduction

Pulsars are an uncommon type of neutron star that produces radio emissions that can be detected here on Earth and are of significant scientific interest [1]. The radiation of pulsar beam sweeps through the sky as it rotates, and this creates a visible pattern of broadband radio emission when it reaches our line of sight [2]. Such patterns are repeated periodically as the pulsar rotates rapidly; hence pulsar search requires looking for periodic radio signals.

Every pulsar creates a slightly different pattern that varies slightly with every rotation. A possible signal detection known as a' candidate' is averaged over several rotations of the pulsar and, with inadequate information, each candidate can potentially describe an actual pulsar. In practice, however, most detections are generated by radio frequency interference (RFI) and noise, making it difficult to identify valid signals. Machine learning, in particular classification systems, are now being widely adopted to label pulsar candidates to aid rapid study.

This paper aims to critically evaluate two models (Feedforward Multilayer Perceptron (MLP) and a Support Vector Machine (SVM)) designed to determine pulsar candidate authenticity based on pattern recognition of specific signal attributes. Different model configurations and data distributions are investigated to tackle the problem of pulsar detection.

The following sections will briefly discuss the two algorithms of choice, namely MLP and SVM, with their pros and cons. We also present and describe the initial raw dataset, the data processing and results of exploratory data analysis. Afterwards, we formulate our hypotheses about how the two algorithms compare to each other. Then, we turn to a description of the methodologies and the experiments used to run them. Finally, we critically evaluate our findings and confirm whether or not they support our hypotheses.

2.0 Summary of Algorithms

2.1 Support Vector Machines (SVM)

SVM is a supervised binary classification algorithm with a high generalisation capability that relies on the notion of constructing an optimal hyperplane to optimise the margin of distance between two data classes [3]. It can be generalised to handle the multi-class classification cases using either a one- vs-one approach (a binary classifier for each pair of classes) or a one-vs-all approach (a binary classifier for each class).

PROS	CONS
• It is very effective in the higher dimension; independent of feature space dimension. • Outliers have less impact as the hyperplane is only affected by the support vectors. • Effective in cases where the number of features (dimensions) is greater than the training samples. • SVM is ideal for extreme case binary classification and is relatively memory efficient.	• Does not perform very well in case of overlapping classes, when there is more noise in the dataset. • There is no probabilistic explanation for the predictions as the classifier works by placing data points below and above the hyperplane. • Selecting the appropriate kernel function can be tricky as kernels tend to increase algorithmic complexity. • It is massively outclassed in perceptual tasks: vision, text, speed etc.

SVM can be adapted to perform regression tasks using different loss function, which involves a margin of tolerance for its decision boundary. The basic form of SVM focuses on linear separability, but the problem of non-linearly separable data can be solved using the kernel trick [3]. The data is mapped into a higher-dimensional space that can find a hyperplane to separate the data. Everything is then projected back onto the initial lower-dimensional space, which gives rise to non-linear decision boundaries.

2.2 Multilayer Perceptron (MLP)

The Multi-Layer Perceptron is a type of feedforward artificial neural network (ANN) consisting of at least three layers of nodes: an input layer, a hidden layer and an output layer [4]. The MLP is supervised classifier that can be adapted to perform both regression and classification tasks by controlling the number of output neurons in the output layer and their activation functions.

PROS	CONS
• They are flexible, can be used for both regression and classification and can use any data that can be made numeric. • Universal approximator: a mathematical model with approximation functions and can model with one hidden layer. • Different forms of regularisation to deal with overfitting and improve convergence, example: L1/L2 penalty terms, early stopping, drop-out etc. • Does not make assumptions regarding the underlying probability distributions. • The hidden neurones act as feature extractors: extracts useful information during learning.	• Requires and relies on lots of training data which leads to the problem of over-fitting and generalisation. • It is time consuming and computationally expensive to train on traditional CPUs. • Difficult to adjust the learning rate, and it has many hyper-parameters to consider. • It is difficult to know how each independent variable is influencing the dependable variables, thus, falls under the category of black boxes.

MLP employs backpropagation, a supervised learning technique, for training. The hidden layer(s) extract useful features (information) during learning and assign adjustable weighting coefficients to the input signals of the input layers. Gradient descent based optimisation procedures are typically employed in MLP training. The algorithm training involves an iterative process of forward passing through the network an input signal, computing an output error relating to the target value prediction and propagating the error back to update the associated weights.

3.0 Hypothesis Statement

Generally, the notion of performance is one of significance and can refer to various aspects of the learning algorithm. For this purpose, we aim to compare SVM and MLP on different fronts to address the following hypotheses formulated to capture how these algorithms compare and behave.

- Both algorithms are expected to perform considerably better in this prediction task than random guessing. SVM is thought to be the ideal candidate in binary classification tasks. That said, SVM should not be far off.
- Space and time complexity are usually difficult to estimate because they are dependent on algorithm implementation. However, MLP should be slower at first, but faster as the number of samples increase.
- Decision Boundaries help understand the expressive ability of a learning algorithm. Given the right configuration, both algorithms are expected to output arbitrarily non-linear decisions lines. That said, SVM should have the upper-hand in identifying areas of uncertainty.

4.0 Dataset

The dataset used in the study is HTRU2 dataset describing a sample of pulsar candidates during the High Time Resolution Universe Survey (South) retrieved from UCI Machine Learning Repository [5]. The dataset contains 17898 observations, 16259 false examples caused by noise and 1639 real pulsar examples checked by human annotators.

The dataset consists of 9 attributes; 8 continuous attributes and a single class attribute. The first four attributes consist of the mean, the standard deviation, excess kurtosis and skewness of the integrated profile of the candidate, whereas, the next four attributes after also consist of the same details of the DM-SNR curve of the candidate respectively. The final attribute is the class label of 0 (negative) and 1 (positive), which indicate whether a candidate is a pulsar or not. Table 1 shows a sample of the dataset.

mean_ip	std_ip	kurt_ip	skew_ip	mean_ds	std_ds	kurt_ds	skew_ds	class
140.5625	55.68378214	-0.23457141200000000	-0.699648398	3.199832776	19.11042633	7.975531794	74.24222492	1
102.5078125	58.88243001	0.465318154	-0.515087909	1.677257525	14.86014572	10.57648674	127.39357960000000	1
103.015625	39.34164944	0.323328365	1.051164429	3.121237458	21.74466875	7.735822015	63.17190911	1
136.75	57.17844874	-0.068414638	-0.636238369	3.642976589	20.959280300000000	6.89649891	53.59366067	1
88.7265625	40.67222541	0.600866079	1.123491692	1.178929766	11.4687196	14.26957284	252.5673058	1

Table 1. Sample of dataset

4.1 Data Analysis

Given the objective this study, the class distribution is of prime importance given the models under consideration. We observed a class imbalance with 16259 false observations against 1639 positive observations. Since the difference in the number of class observations is great, we applied a balancing technique called SMOTE to equilibrate the observations [6]. The negative class was kept the same (16259) and additional observations were generated to increase the positive class observations to 16259 (equal to that of the majority class).

We also normalised the independent variables (continuous variables) to increase the performance of our models. Finally we performed feature importance analysis using decision trees to which variables influence the class prediction most. Table 2 shows the statics of the positive class before and after smote. The negative class was unaffected as it is the majority.

Attributes	Positive (before smote)				Positive (after smote)			
	Mean	Stdev	Max	Min	Mean	Stdev	Max	Min
Mean of the integrated profile	56.69	30.01	139.26	5.81	56.48	29.67	139.26	5.81
Standard deviation of the integrated profile	38.71	8.03	83.80	24.77	38.52	7.57	83.80	24.77
Excess kurtosis of the integrated profile	3.13	1.87	8.07	-0.09	3.14	1.86	8.07	-0.09
Skewness of the integrated profile	15.55	14.00	68.10	-1.14	15.60	13.81	68.10	-1.14
Mean of the DM-SNR curve	49.83	45.29	119.58	0.49	49.52	44.93	119.58	0.49
Standard deviation of the DM-SNR curve	56.47	19.73	109.66	7.66	56.46	19.45	109.65	7.66
Excess kurtosis of the DM-SNR curve	2.76	3.11	30.88	-1.86	2.73	3.00	30.88	-1.86
Skewness of the DM-SNR curve	17.93	50.90	1017.38	-1.87	17.10	44.72	1017.38	-1.87

Table 2. Statistics of positive class before and after smote

5.0 Methodology

This section presents details of the training, validation and testing conducted. It also describes the architecture, experimentation and evaluation implemented in developing the models. The general evaluation approach is to holdout 30% of the dataset for testing and comparison purposes between the best performing models of each algorithm. The reason for the allocation of 30% of the dataset for testing is to ensure that the generalisation performance estimates have improved numeric stability and precision. The remaining 70% is used for training and model selection.

Accuracy is the chosen metric chosen to compare the models. The reason being the non-existence of class imbalance because of smote and also the cost implication of misclassifying a data point is the same regardless of the class label. We also make use of confusion matrix to get a more fine-grained look into the models' performance. Learning curves based on increasing the number of data points are computed and plotted to observe the rate of convergence in training and validation accuracies with the best configurations for both models. This essentially helps to diagnose and determine whether models under-fit, over-fit, need more data to improve performance, etc. Decision boundaries produced by SVM and MLP are analysed to further empirically demonstrate the shape of each boundary. Time and space complexities are also investigated with varying number of examples and relative size of models to achieve a complete study on model comparison.

5.1 SVM Architecture And Parameters

A typical grid search was performed over the key hyper-parameters: box constraint, kernel type and selected kernel parameters. The box constraint controls the maximum penalty imposed on margin-violating observations (misclassification cost) and helps in preventing over-fitting (regularisation). Increasing the box constraint assigns fewer support vectors; however, it can be computationally expensive [7].

The kernel function (type) is dependent on the complexity of the distribution of the data. Kernel functions tried out are linear, radial basis function (RBF) and polynomial. The final hyper-parameter is the selected Kernel parameter: for RBF it is the RBF scale, and for the polynomial function it is the polynomial order.

5.2 MLP Architecture And Parameters

In MLP, grid search was not feasible as the models take excessively long to train, and the number of hyper-parameters to tune is high. A successful alternative is random search which has been proven to yield better results [8]. Nevertheless, in this case, Bayesian optimisation was chosen as an even more sophisticated probabilistic search.

Bayesian optimisation usually is used in the context of minimising expensive cost functions and has been shown to work particularly quite well for model selection [9]. The hyper-parameters considered in this approach include the number of hidden layers, the number of hidden neurones (repeated in each hidden

layer), the learning rate and momentum, the training or optimiser procedure and the activation function of the hidden neurones.

Naturally, being a classification task, soft-max activation function is employed in the output layer along with cross-entropy as loss function. The number of input and output neurones is determined by the number of independent variables and the number of classes respectively.

6.0 Experimental Results And Final Models

For SVM, to handle the large number of models involved in the grid search, we start with a wide range of parameters, narrow it down to best kernel type, and kernel parameter, then further tune the misclassification cost around the best kernel. Since there is no standard definition of the over-fitting range, a fair threshold of 5% of the difference between training and validation scores is adopted for this project. The best output SVM model has an RBF kernel function with a kernel scale of 1, a box constraint of 80 and a validation accuracy of 96.87.

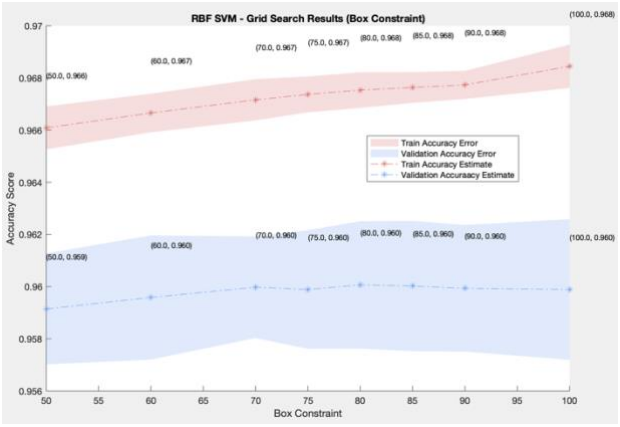

Figure 1. Stage 2 RBF best box constraint search

For MLP, we run 200 iterations of Bayesian optimisation over the hyper-parameters. The maximum number of Epochs was set to 500, and an early stopping when the validation score does not increase for eight consecutive steps was implemented. From figure 2, we see that after about 10 iterations, we reach the minimum possible level and remain constant afterwards. The best performing MLP contains one hidden layer with 34 neurones, a learning algorithm of scaled conjugate gradient [10], a log-sigmoid activation function for the hidden neurones and a validation accuracy of 95.45.

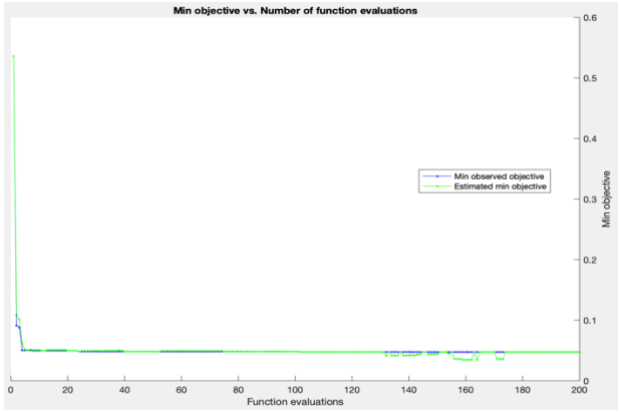

Figure 2. MLP Bayesian optimization results

Figure 3 shows the learning curves of both models. Both algorithms start converging at 10,000 training data points with MLP having a greater error interval. Both algorithms seem to converge to their ultimate best performing validation score. In measuring of the training time, a clear polynomial curve relationship is observed for SVM while the MLP exhibit a linear relationship. Apart from that, linear relationships are observed for both models when the number of attributes is varied instead. MLP is slower than SVM when the number of training examples is limited, but picks up after initial learning is achieved.

The decision boundaries in figure 4 show that both algorithms manage to capture the underlying form of the data even when the classes are not perfectly separable. However, SVM tends to be better at detecting regions of uncertainty (due to lack of data) as seen in the probability decision graphs. All the same, both algorithms are capable of outputting arbitrarily non-linear decision boundaries.

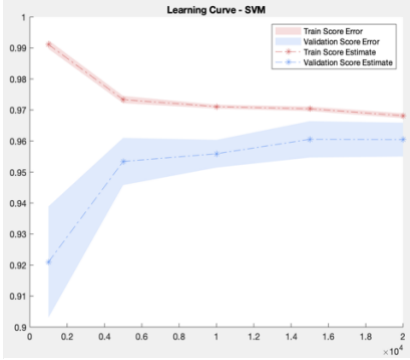

Figure 3. Learning curves of models

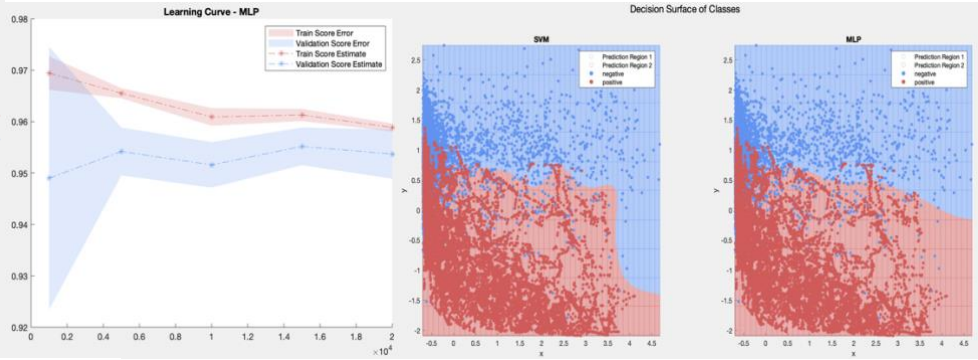

Figure 4. Decision boundaries of models

7.0 Analysis And Critical Evaluation of Results

The confusion matrices of the final models are shown in figure 5. Both models performed similarly well, with SVM producing a higher accuracy by 0.05%. This is because of the random weight initialisation of the MLP and SVM being ideal for extreme cases of binary classification. Figure 6 shows a ROC (Receiver Operation Curve) to check the quality of the classifiers. The AUC (area under a ROC), a measure of the classification accuracy, is not affected by class imbalance in this instance which is advantageous. Although both algorithms have similarly good AUC, SVM has better AUC overall.

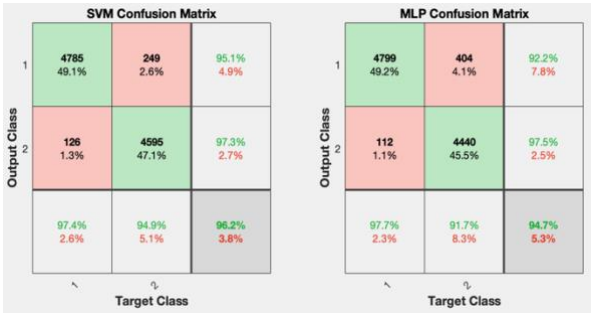

Figure 5. Confusion matrices

In the SVM hyper-parameter tuning, over-fitting is heavily dependent on the hyper-parameters as they have a strong influence on the model's complexity, which determines the flexibility of the decision boundary. Weak generalisation on unseen data is typical with large box constraints except for RBF SVM models with high kernel scale values which negates this effect. Unlike the SVM error rate, which continuously decreases as box constraint increases, kernel scale values contain a minimum point at which the SVM model achieves its optimum accuracy; thus, do not follow a similar pattern [11]. In this study, One-vs-One (OvO) is used instead of One-vs-All (OvA) owing to these reasons; OvA is computationally costly given the size of our dataset, and also OvA and OvO are known to produce similar results in practice. Thus, OvO is practically and theoretically suitable in this case.

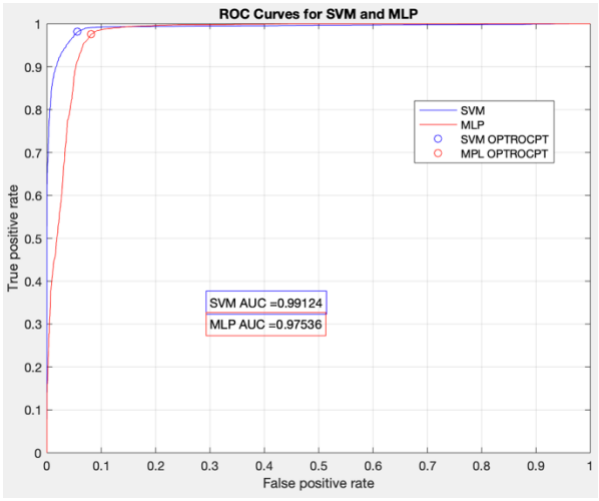

Figure 6. ROC curves (with AUC)

Bayesian optimisation is used in MLP to reduce the number of models built in order to reach optimal performance. Iterations of 200 are carried out, but it converged after about 62 and could have ended searching when the curve remained constant over a few iterations (like an early stopping). A grid search could have taken a few hundred models, given the initial search space, before achieving the same performance. Early stopping was utilised to stop training the models when they exhibited signs of overfitting to ensure optimal generalisation performance of the current network configuration.

The difference in the train and validation scores of the learning curve suggests no over-fitting both models. The validation performance of both models converge to around 95-96% with increasing data size, and the addition of more data would not show any drastic improvement. This supports our initial hypothesis of similar performance of both models with SVM edging out slightly better.

The training time complexity findings also support the initial hypothesis. The SVM is quadratic whiles the MLP is linear. This suggests two regimes for the SVM; the first where it is faster and the second where it is slower. The MLP is almost linear because it is still operating in one regime (slower), given the maximum number of training examples (2500).

Finally, both algorithms output correct prediction regions to majority of the data points in figure 5. The difference is that the decision boundary of SVM exhibits more structure in that it captures uncertainties, that is to say, marking areas without enough data points. The reason probably being the geometric form of RBF kernel function. This goes on to support the initially stated hypothesis.

8.0 Conclusion And Future Work

This study compares and evaluates the performance of SVM and MLP in the classification of a sample of pulsar candidates. This was conducted in terms of the predictive performance, space and time complexity, nature of decision boundaries and learning curves.

Hyper-parameter tuning the models proves essential as both algorithms possess many hyper-parameters. Significant improvement is achieved with the appropriate parameter values. Simple grid search sufficed for SVM because the number of hyper-parameters is manageable whiles Bayesian optimisation was very effective for the MLP search converging to an acceptable level quickly. Both SVM and MLP had similar results in the training, validation and testing stages as expected. However, the SVM showed to be a better algorithm for this classification task.

Training time is a significant issue due to the large dataset; an effort to improve SVM can include integrating AdaBoost to produce more compact classifiers with fewer support vectors while significantly reducing training time. Another investigation worth doing is using different training techniques, such as bagging, to understand feature extraction taking place in the data. The idea in case of MLP is to reduce the dependency on data allocation and random initial neurone weights to minimise their effect.

References

[1] Keith, M. J., Jameson, A., van Straten, W., Bailes, M., Johnston, S., Kramer, M., Possenti, A., Bates, S. D., Bhat, N. D. R., Burgay, M., Burke-Spolaor, S., D'Amico, N., Levin, L., McMahon, P. L., Milia, S., Stappers, B. W. (2010). The High Time Resolution Universe Pulsar Survey – I. System Configuration And Initial Discoveries, *Monthly Notices of the Royal Astronomical Society*, Volume 409, Issue 2, December, Pages 619–627.

[2] Lorimer, D. R., and Kramer, M. (2005). *Handbook of Pulsar Astronomy*, Cambridge University Press.

[3] Cortes, C. and Vapnik, V., (1995). *Support-vector networks. Machine learning*, 20(3), pp.273-297.

[4] Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

[5] HTRU2 Data Set, accessed 08 March 2020, UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/HTRU2

[6] Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002). SMOTE: Synthetic Minority Over-Sampling Technique. *Journal of Artificial Intelligence Research*, 16, pp.321- 357.

[7] Andrew, A.M. (2000). *An Introduction to Support Vector Machines and Other Kernel Based Learning Methods by Nello Christianini and John Shawe-Taylor*, Cambridge University Press, xiii+ 189 pp., ISBN 0-521-78019-5.

[8] Bergstra, J., and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb), 281-305.

[9] Brochu, E., Cora, V. M., & De Freitas, N. (2010). A Tutorial On Bayesian Optimization Of Expensive Cost Functions, With Application To Active User Modelling And Hierarchical Reinforcement Learning.

[10] Møller, M. F. (1993). A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4), 525-533.

[11] Han, S., Qubo, C. and Meng, H. (June 2012). Parameter Selection In SVM With RBF Kernel Function. *World Automation Congress*, (pp. 1-4). IEEE.

Appendix 1: Glossary

Appendix 2: Implementation Details