UNIVERSITÀ
DEGLI STUDI
DI PADOVA

INQ0091579 - DEEP LEARNING 2023-2024

Brown Dwarf Detection

By :
Groenlien, Haugland, Svane

Table of Contents

List of Figures

List of Tables

1 Introduction/Theory

Many years ago, astronomers found out that star formation not always produce a regular star[1]. These got the name "Brown Dwarfs", and they don't have enough mass to sustain nuclear fusion at the core, which in turn makes the stars shine[2]. According to the "Hubblesite", this has made them a hard class of stars to detect for researchers, with the first discovery coming in 1995. They also state that the stars do emit some light through radiation, which is caused by gravitational contraction, in the same way Jupiter radiates some energy. However, this radiation diminishes over the years, making it even harder to spot old brown dwarfs. Brown dwarfs are also quite small in size. They could be around the size of Jupiter, while their mass stretches from a lower bound of $13M_j$ and an upper bound for its mass of $80M_j$[3][1]. Both of these factors have made them hard to spot, but also hard to separate from other stellar objects, even though they are actually estimated to account for at least 25% of all stars in our galaxy[4].

Classically, brown dwarfs have been detected by looking at their infrared color spectre[5] . We will instead try to use a dataset of 26 columns and 5569 rows, containing info related to luminosity, spectral type and other astrometrical information of stellar objects(see Appendix A)s. We want to feed this data to a predictive model based on deep learning in order to accurately predict brown dwarfs apart from other stellar objects. From the universal approximation theorem we know that machine learning can learn and approximate any mapping from X to y all by itself[6]. While this is true for machine learning with one layer, called a shallow neural network, it is far from the most efficient way. For efficiency we want to use deep learning and multiple layers to reduce computational cost. We therefore use a deep neural network to find the relevance in the data and do the mapping from data X to ground truth y.

2 Material and Method

2.1 Data preprocessing

To be able to test our model's predictive abilities, we withhold some of data to be our test data for later. About 20% of the data will be used as test data. To increase certainty when training our model, we train it five times in five different folds. In each fold we extract a different 20% of our data to be the test data, thus we will create five models trained and tested against different folds.

In our dataset we had some missing entries which were set to zero. We chose linear regression as our machine learning model to impute our missing values. We assume that the missing data in our dataset is missing at random (MAR). This means that the missing values in our dataset occurs at random. In addition we assume that the values of the missing data can at least in part be explained by linear relationships.

We scale each column vector to unit variance. This is done by subtracting the mean and dividing by the standard deviation. This is done to prevent features of a larger magnitude to dominate the learning process.

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Where μ is the mean and σ is the standard deviation.

2.2 Loss function and Optimizer

In the training process we used binary cross-entropy loss for our loss function:

$$\text{BCELoss}(x, y) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \cdot \log(x_i) + (1 - y_i) \cdot \log(1 - x_i)] \tag{2}$$

[1] $1M_j$ is the equivalent of the mass of Jupiter.

This is the standard practise for binary classification. We have sigmoid as the activation function for the output layer with a range between and including zero and one, which can be interpreted as probabilities. Cross-entropy loss will penalize confidently correct answers heavily and it corresponds to the negative log-likelihood of the prediction.

This characteristic helps our optimizer to determine which direction it needs to go to minimize the loss. Our choice of optimizer for the gradient descent is the Adaptive Moment Estimation, commonly called Adam.

The pseudocode for Adam is:

```
Initialize: $t = 0, \quad m_0 = 0, \quad v_0 = 0$
while not converged do
  $t = t + 1$
  $g_t = \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
  $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$ (Update biased first moment estimate)
  $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ (Update biased second raw moment estimate)
  $\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$ (Compute bias-corrected first moment estimate)
  $\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$ (Compute bias-corrected second raw moment estimate)
  $\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$ (Update parameters)
end while
```

Adam is a stochastic gradient descent technique with momentum and an adaptive learning rate, meaning it is much less prone to get stuck in a local minimum and will converge quicker than pure gradient descent.

2.3 Different metric scores

Our training metric is Matthews correlation coefficient (MCC). This is the metric by which we evaluate the performance of our model, but we also look at the other popular metrics called *Accuracy*, *Precision*, *Recall* and *F1score*.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{3}$$

$$Accuracy = \frac{TP + TN}{TotalSample} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1score = \frac{2}{1/Precision + 1/Recall} \tag{7}$$

2.4 Activation function

As noted earlier, the activation function for our final layer is the sigmoid function. We have chosen it as it is a function that yields a number between 0 and 1 which we again can interpret as probabilities. The activation functions between the other layers will be decided by testing which performs best out of a group of four candidates

Rectified linear unit -

$$ReLU = max(0, x) \tag{8}$$

Tangent hyperbolic -

$$Tanh = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \tag{9}$$

Hard tangent hyperbolic -

$$Hardtanh(x) = \begin{cases} -1 & \text{if } x < -1 \\ x & \text{if } -1 \leq x \leq 1 \\ 1 & \text{if } x > 1 \end{cases} \tag{10}$$

The sigmoid curve -

$$Sigmoid = \frac{1}{1 + \exp^{(-1)}} \tag{11}$$

We ran a deep neural network with architecture $26x10x5x1$x to experiment with the four different activation functions for our hidden layers:

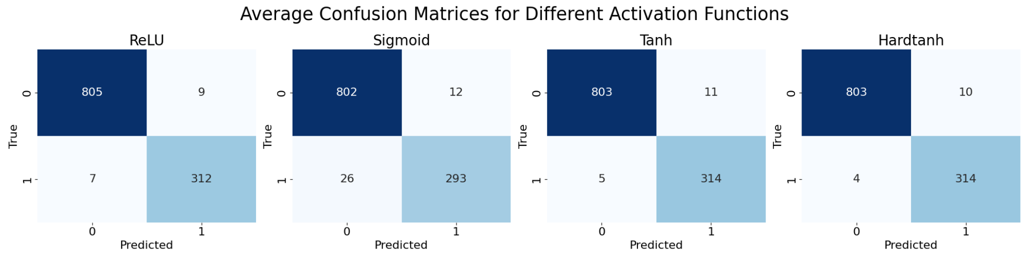


Figure 1: Average confusion matrices, over 5 folds, for 4 different activation functions, for easy comparison.

Looking at the confusion matrices in figure 1, all the activation functions, except for the Sigmoid, seems like good choices for our model. Hardtanh has the highest accuracy of 0.988, and ReLU has the lowest number of FP with 9.

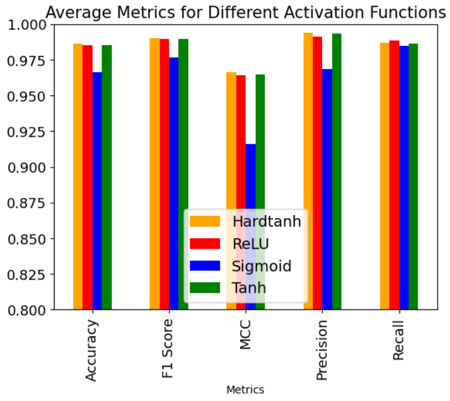


Figure 2: A bar plot over the average metrics, over 5 folds, for 4 different activation functions, for easy comparison.

Looking at figure 2, the ReLU, Tanh and Hardtanh performs quite equally on all metrics, with small fluctuations between them. Hardtanh scores highest on MCC, which is our most important metric.

We ended up by chosing ReLU as our activation function, evened though Hardtanh outperforms it on MCC score. This because it had a small differences, which could be evened out on larger datasets, and because ReLU is a common choice for activation function in most deep neural networks[7].

2.5 Hyperparameters and Architecture

To decide which hyperparameters we are going to use for our model, we used Optuna to decide how many hidden layers and nodes each layer should contain, as well as our optimizer.

The result was a fully-connected neural network with 539 parameters. It has an input layer with 26 nodes, one for each feature. This is connected to the hidden layers where the first layer is 14 nodes where each has a weight and a bias that connects to a second layer of 10 nodes. The final output layer is one node since we are creating a classification model.

3 Results and Discussion

3.1 Convergence

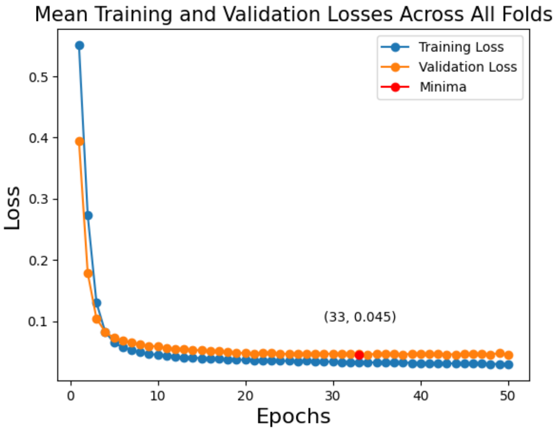


Figure 3: Mean training and validation loss plotted against epochs, with a red dot marking the minimal loss for the validation data. The mean covers all 5 folds.

From figure 3, we see that on average the training and validation loss seem to decrease and converge. The chosen number of epochs seems to be a good choice, since larger values increased the overfitting together with the validation loss. In the same figure, we have marked the minima with a red dot, to find the best number of epochs, when running with a maximum of 50. The result was 33 epochs, but generally a number around this would probably give a good result. The rest of the results were ran with 33 epochs.

3.2 Metric scores

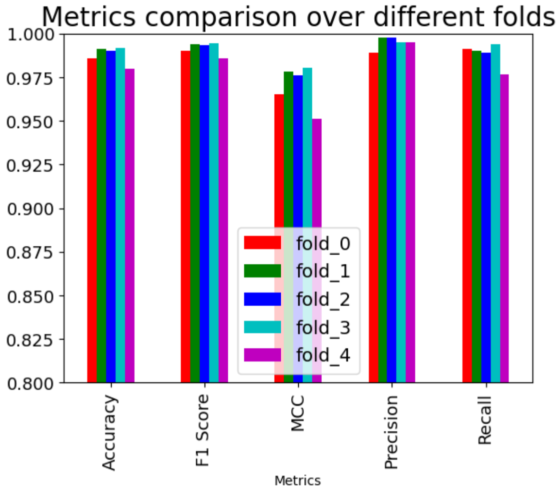


Figure 4: Metric scores over the different folds

Figure 4 shows the chosen metric scores over all the validation data, from the different folds. As one can note, fold 4 usually gets worse scores in general than the other folds, and fold 3 gets some higher scores than any other fold. To evaluate our model, we chose to look at the mean of the interquartile range, assuming that the other quartiles had more deviating results from the global mean. This was done by dropping the 3. and 4. fold.

Metric	Score	Score (only colours)
Accuracy	0.989	0.988
F1 Score	0.992	0.991
MCC	0.973	0.971
Precision	0.995	0.994
Recall	0.990	0.989

Table 1: Mean metric scores of the 2.- and 3.-quartile over the 5 different validation folds.

From table 1, one can see that we achieve a MCC-score of 0.973. The other scores are generally high, reaching around 0.99.

3.3 Confusion Matrix

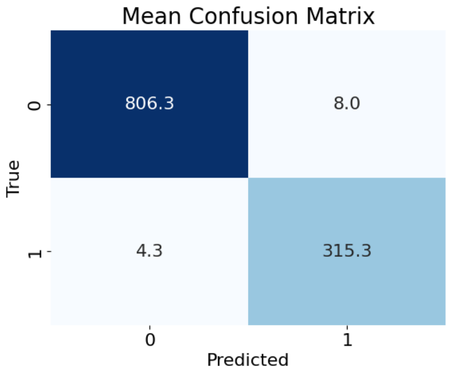
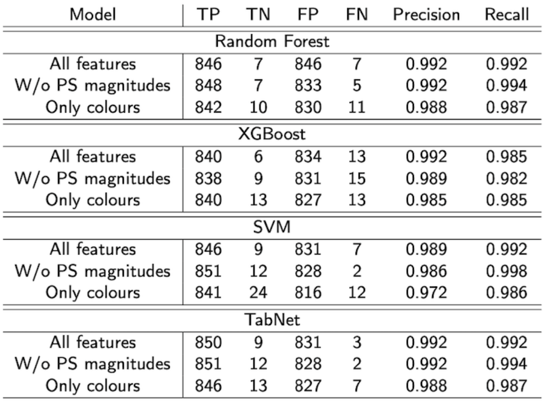


Figure 5: Mean confusion matrix values, over the interquartile range on the test data.

Looking at figure 5, the model seem to miss the same amount of time on classifying brown dwarfs and other stellar objects, with just around 1%. Preferably would we like to be more confident that our brown dwarf predictions are true, rather than having a large amount of other stellar objects as brown dwarfs. As an improvement, we could try to lower the FP while ignoring the FN more. This could lead us to miss out on some brown dwarfs, but will make the final output more valid.

3.4 Comparing with other model



Model	TP	TN	FP	FN	Precision	Recall
Random Forest						
All features	846	7	846	7	0.992	0.992
W/o PS magnitudes	848	7	833	5	0.992	0.994
Only colours	842	10	830	11	0.988	0.987
XGBoost						
All features	840	6	834	13	0.992	0.985
W/o PS magnitudes	838	9	831	15	0.989	0.982
Only colours	840	13	827	13	0.985	0.985
SVM						
All features	846	9	831	7	0.989	0.992
W/o PS magnitudes	851	12	828	2	0.986	0.998
Only colours	841	24	816	12	0.972	0.986
TabNet						
All features	850	9	831	3	0.992	0.992
W/o PS magnitudes	851	12	828	2	0.992	0.994
Only colours	846	13	827	7	0.988	0.987

Figure 6: Performance of other models from the article: ”Machine learning methods for the search for LT brown dwarfs in the data of modern sky surveys” by Aleksandra Avdeeva[8]

To measure exactly how good our model is we need to compare it to other models for predicting brown dwarfs, preferably models which are trained on the same data. We therefore compare it to the results found in the article: ”Machine learning methods for the search for L&T brown dwarfs in the data of modern sky surveys” by Aleksandra Avdeeva[8]. Looking back at table 1, we achieve achieve a slightly higher precision and recall using all feature compared to only features based on colour. This are the same results found by A. Avdeeva. With a precision score of 0.995 and a recall score of 0.990 our neural network based model outperforms the other Random Forest, XGBoost and SVM. The Tabnet model scores higher on recall than our model. It seems that neural networks are a better choice of model for detecting brown dwarfs as both our model and Tabnet outperforms the other models.

In the reference paper they also balanced out the dataset so that the amount of positives and negatives are equal [8]. That is why you see an equal amount positives and negatives in figure 6. We chose not to do this in our preprocessing stage as we did not gain any better test score, and as noted earlier the prediciton rate stayed balanced.

3.5 Comments

In our choice of linear regression for imputing the missing values we assumed that it exists a linear relation that could explain in part some of the relations between the data points. There might exist a better method for imputing the missing data, for example some of the relations might be quadratic.

Because of our choice of a sequential neural network for our model we did not do any feature selection. With this architecture the model should be able to extract the most meaningful relations by itself. This might make our model try to fit noise in our dataset from the less meaningful attributes. This means we could have benefited by having some feature selection done at the preprocessing stage.

4 Conclusion

We managed to create a deep neural network for detecting Brown Dwarfs, with a MCC score of 0.973, by using a dataset with 26 features. This outperforms more traditional machine learning techniques such as SVM, XGBoost and random forest, and scores roughly around the same as TABNet. Compared to classical analysis, only looking at features related to colors, more features results in a better MCC score of 0.2%. We can therefore assume that using more features results in better models, but not with significantly higher.

For better results we could have feature engineered the data more, instead of using all features. We could have looked at how much each feature affected the results, removing unnecessary information, and reducing runtime and dimensions. Balancing the data could have yielded better results. Also, focusing more on lowering FP results would have made the output more valid, since you would increase the percentage of TP, and therefore increased the certainty of the positive result. And in the end as our telescopes and images gets better we can get better data and with that our deep learning models will get even better.

Bibliography

[1] D. C. Palma, *Failed stars: Brown dwarfs*. [Online]. Available: https://www.e-education.psu.edu/astro801/content/l5_p6.html.

[2] H. S. N. Team, *Astronomers announce first clear evidence of a brown dwarf*, 1995. [Online]. Available: https://hubblesite.org/contents/news-releases/1995/news-1995-48.html.

[3] A. J. Burgasser, *Brown dwarfs: Failed stars, super jupiters*, 2008. [Online]. Available: https://doi.org/10.1063/1.2947658.

[4] H. Weitering. '100 billion failed stars may be hiding in the milky way'. (2017), [Online]. Available: https://www.space.com/37401-100-billion-brown-dwarfs-milky-way.html.

[5] F. SAUNDERS, *Search for brown dwarfs*, 2014. [Online]. Available: https://www.americanscientist.org/article/search-for-brown-dwarfs.

[6] K. Hornik, M. Stinchcombe and H. White, *Multilayer feedforward networks are universal approximators*, 1989. [Online]. Available: https://doi.org/10.1016/0893-6080(89)90020-8.

[7] S. SHARMA. 'Activation functions in neural networks'. (), [Online]. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 (visited on 23rd May 2024).

[8] A. Avdeeva, *Machine learning methods for the search for lt brown dwarfs in the data of modern sky surveys*, 2023. arXiv: 2308.03045 [astro-ph.SR].

Appendix

A Dataset parameters

Column Name	Description
RAdeg	Right Ascension (RA) in degrees. RA is one component of the celestial coordinate system used to locate objects in the sky along the east-west direction.
DEdeg	Declination (Dec) in degrees. Declination is another component of the celestial coordinate system, used to locate objects in the sky along the north-south direction.
PS1gmag	Apparent magnitude of the brown dwarf in the Pan-STARRS g-band.
e_PS1gmag	Uncertainty or error associated with the Pan-STARRS g-band magnitude measurement.
PS1rmag	Apparent magnitude of the brown dwarf in the Pan-STARRS r-band.
e_PS1rmag	Uncertainty or error associated with the Pan-STARRS r-band magnitude measurement.
PS1imag	Apparent magnitude of the brown dwarf in the Pan-STARRS i-band.
e_PS1imag	Uncertainty or error associated with the Pan-STARRS i-band magnitude measurement.
PS1zmag	Apparent magnitude of the brown dwarf in the Pan-STARRS z-band.
e_PS1zmag	Uncertainty or error associated with the Pan-STARRS z-band magnitude measurement.
PS1ymag	Apparent magnitude of the brown dwarf in the Pan-STARRS y-band.
e_PS1ymag	Uncertainty or error associated with the Pan-STARRS y-band magnitude measurement.
W1mag, W2mag, W3mag, W4mag	Apparent magnitudes of the brown dwarf measured by the Wide-field Infrared Survey Explorer (WISE) satellite in its four infrared bands (W1, W2, W3, W4).
e_W1mag, e_W2mag, e_W3mag, e_W4mag	Uncertainties or errors associated with the WISE infrared magnitude measurements.
Jmag, Hmag, Kmag	Apparent magnitudes of the brown dwarf in the near-infrared bands measured by the Two Micron All-Sky Survey (2MASS) in its J, H, and K bands.
e_Jmag, e_Hmag, e_Kmag	Uncertainties or errors associated with the 2MASS near-infrared magnitude measurements.

Table 2: Description of dataset parameters