Machine Learning for Racehorse Ranking Prediction

Adam Gardiner-Hill MSci - Quantum Leap Solutions

May 10, 2022

Abstract

This report investigates the plausibility of utilising machine learning algorithms for the prediction of racehorse rankings based on a variety of the data that QL Solutions collects. Currently, a complex set of statistical analyses are used to predict runner position based upon previous form, ratings and race information. The application of machine learning algorithms to this task has the potential to greatly increase computational efficiency whilst maintaining high predictive accuracy. Moreover, ML programmes would allow for a high level of automation, built on top of the QL architecture already in place for data capture and manipulation. To act as a first proof-of-concept test, rankings data along with seven relevant variables was compiled and then used to train and test a sequential, classification type neural net. The network achieved 47.7% accuracy when predicting an exact rank, and 91.8% accuracy to within one rank of error, both are comparable to results achieved by complex statistical models. The same data was then used alongside the automated machine learning package, Auto-Keras, to investigate whether changes in architecture could produce better results.

Investigation

Data and Pre-processing

For this investigation, data was compiled that considered seven variables hypothesised to have correlation with final ranking; 'distance to first bend', 'mode of previous ranks', 'draw', 'race pace', 'stalls position', 'race distance' and 'strategy'. Aside from their likely relations to rank outcome, these variables were chosen because of their generality across a wide range of races, courses and horses. The variable 'distance to first bend' acts to generalise the importance of initial race positioning across all courses and distances, and the 'mode of previous rankings' and 'strategy' variables generalise previous performances across all horses. These generalisations greatly increase the size of the available data set, at minimal cost to the specific informational detail contained in course, race and horse variables.

Rankings data was modelled on QL's 1-4 ranking system. Rank 1 representing the leader(s), rank 2 being prominent horses, rank 3 the main body of the field and rank 4 the back of the pack.

DtFB	MoPR	Draw	Race P.	Stalls Pos.	Race Dis.	Strat.
...	...	...	...	...	...	...

Table 1: Table illustrating the considered variables used to train the network

In this case, data was only considered from flat courses in Great Britain where races had at least one bend. This was done to eliminate the added complication of "straight" races which would have to have their "DtFB" values modelled as either zeros or infinities for the network. Zeros would dilute the data considerably and be difficult to differentiate from races that start on a bend. Infinities are complex to get a network to recognise, so for the purpose of this "proof-of-concept" test, all straight races were ignored. Other nations and jumps racing data was omitted to save time in the initial investigation phase.

The data was then cleaned to remove any NaNs, and rows with more than one missing value were dropped. For horses who had not previously run, and so had no 'mode of previous rank' variable, a value of 3 was assumed as it is most common. The final input data consisted of 185555 runner instances. These were split 80/20 into 148444 training instances and 37111 test instances.

Neural Network

A standard sequential neural network was constructed using assets from the Keras Python package. This network consisted of an input layer with shape (7,), 4 dense layers with decreasing node numbers (2100 - 210) and an dense output layer with four nodes, one for each classification class. ReLU activation functions were used for the dense layers, and a Softmax activation function was used for the final dense classification layer. Accordingly, the final output was a probability of each ranking being true. The ranking with maximum probability value was then selected as the network's final classification. A categorical cross-entropy loss function was used to train the network over 100 epochs.

As previously mentioned, the training data was also fed into an automated machine learning package, Auto-Keras. Auto-Keras uses evolutionary search principles to trial, interpret and refine model architectures to optimise performance. In this investigation, little accuracy increase was expected as the nature of the task is such that network architecture should yield minimal improvement beyond a plateau. For the automated machine learning

test, a standard Auto-Keras DataClassifier type model was initiated.

Platform and Packages

The analysis was completed using open source Python packages and the Spyder IDE. These included NumPy, Pandas, Keras, Sci-Kit Learn, MatPlotLib and Seaborn, as well as the Auto-Keras package for automated machine learning. A TensorFlow virtual environment was created for this task, and all neural networks were trained on an NVidia GTX 1060 6GB GPU.

Results

Traditional Neural Network

The network performed well compared to the complex statistical models currently used for ranking prediction, achieving an overall accuracy of 47.7% on the test data set. After training over 100 epochs the network accuracy began to plateau, and the expected effects of diminishing returns were seen. Any improvement beyond this point is assumed to be the result of over-fitting to the training data set. The accuracy plot can be seen in figure 1.

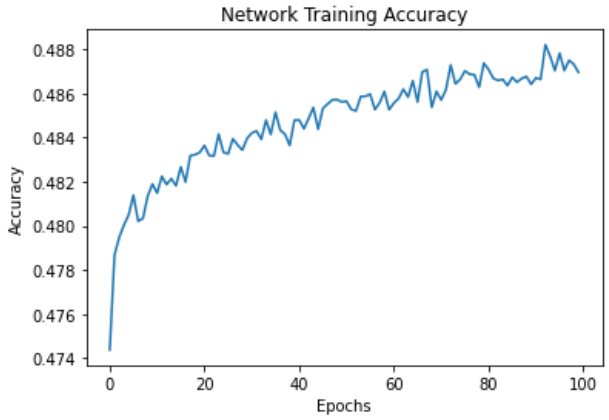

Figure 1: Plot of network prediction accuracy by training epoch. It is clear to see that diminishing returns were obtained after approximately 10-15 epochs, and any improvement beyond this point is likely due to over-fitting to the training data set.

The network was most effective at precisely predicting rank 3, successfully identifying 92% of true values. Performance for rank 1 was also impressive, with correct predictions made in 63% of instances. Despite this, 30% of true rank 1s were classified as rank 3s, a surprising result which would not have necessarily been expected. The network struggled substantially more when trying to identify ranks 2 and 4 exactly, correctly predicting only 6.1% and 8.6% of their instances, respectively. It mischaracterised 87% of rank 2s and 90% of rank 4s as false negative rank 3s, indicating great difficulty in telling them apart. Excluding true rank 1 instances, the network only scored 45.6% accuracy, only 0.15% better than if it had simply predicted rank 3s for all these instances. Figure 2 shows the confusion matrix for these results. Each number represents the percentage of predictions made for that true rank value (i.e. 63% of rank 1s, were predicted as rank 1s, with 6.7% predicted as rank 2s, etc).

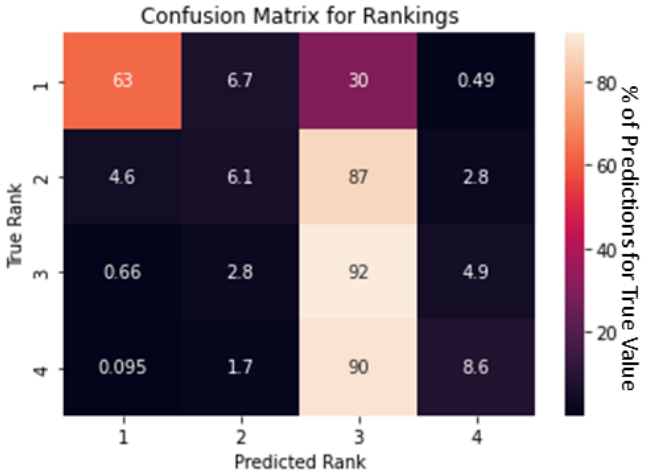

Figure 2: Confusion matrix for the network predictions. Values in squares correspond to the percentage of predictions for that true value (i.e. sum of row = 100%). This was chosen, rather than the percentage of total predictions, to give a better feel for how good the network was with regard to each true value.

To within one rank of error, the network made a correct prediction 91.8% of the time. This was calculated by analysing how many predictions were greater than one rank away from their true value across all instances. Figure 3 shows the distributions of these percentages for each true rank individually, with the bar for correct predictions shown in red.

Automated Machine Learning

When training the automated model, no notable accuracy improvement was seen, with the model achieving 47.6% accuracy across all data points. Whilst Auto-Keras did not prove useful for improving network performance, it did act to reinforce the belief that altering network architecture would yield no further results.

Discussion of Results

Overall, the neural network performed comparably well when predicting horse rankings, achieving 47.7% accuracy on precise prediction, and 91.8% to within one rank of error across all data points.

The introduction of the automated machine learning package Auto-Keras served two purposes. First, it offered the potential for improved performance through better optimised network architecture. Second, even though improved performance was not realised, it acted to confirm that classification accuracy was not being bottle-necked by network structure. It can therefore be inferred that the upper limit of network accuracy is currently limited by the quality of the input data and the nature of the

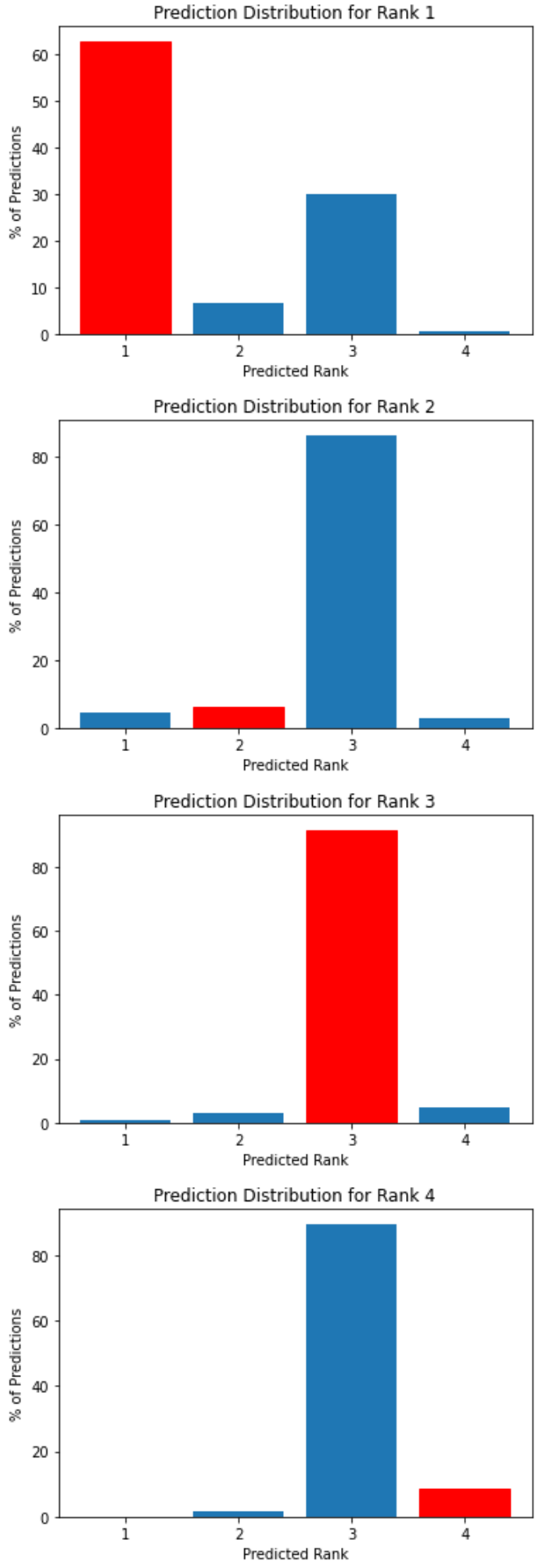

Figure 3: Distributions of prediction values for each true rank, with the bar for correct predictions shown in red. It is clear to see that the network performed best when predicting rank 3s, and also performed well when predicting rank 1s. It struggled greatly with delineating rank 2s and 4s from rank 3s.

relationships between the selected variables and final rankings.

One notable result was the network's relatively good performance when predicting leaders (True Rank = 1). The network exactly identified 63.0% of instances, and predicted to within one rank of error in 69.7% . It is likely that this is due to the nature of our input data, rather than some unique feature associated with rank 1. Most rankings data is manually inputted, and a ranking of 1 has less intrinsic subjectivity associated with it than values 2-4. What may appear to be a 2 to one person, may be a 3 to others, and vice versa. Conversely, entry is certain when inputting a leading horse's position as a 1. For this reason, correlations between our seven variables and a ranking of 1 may be less "diluted" than correlations for other rankings, resulting in more accurate classification by the network. This effect is further evidenced by the difficulties faced when the network is trying to differentiate between ranks 2-4.

It is also possible that rank 1s are easier for the network to predict for a number of reasons. It is important to consider that a subset of horses will be ridden with the goal of always leading because they get the best results when they do so. Other horses are always held up, but may be classed as a rank 2, 3 or 4 when they settle into the fields of different races. This may act to "exceptionalise" rank 1s and make them easier for the network to predict due to the extreme nature of the strategy. Further analysis would be required to determine the degree to which this "dilution" of relationships between the input data and the rankings, caused by human interpretation upon data entry, was responsible for poorer network performance for rankings 2-4.

It also is worth considering the overall distribution of rankings and how this may have affected the training of the network. Figure 4 shows the distribution plots for the true rankings versus the distribution of those predicted by the network. For our true values, 3 is the most common rank by some margin, representing just over 40% of all values. Ranks 4 and 2 are similarly common, representing 25% and 22% of all ranks, respectively. Finally, rank 1s are rarest, comprising only 13% of all values. This distribution is as would be expected, as races will generally have a single leader (rank 1), two or three prominent (rank 2) and rear (rank 4) horses, and a larger main body of the field (rank 3).

In contrast to the true rank distribution, the network predictions show a large over-representation of rank 3s, and substantial under-representations of ranks 2 and 4. Rank 1s were predicted in relative accordance with the true value distribution. This is likely a further manifestation of the network's difficulties when trying to select between ranks 2, 3 and 4. With rank 3 being the most common value, it's possible that the softmax-categorical-crossentropy loss function finds a local minimum by selecting rank 3s in a majority of cases when the class is hard to distinguish - this could be considered a type

of "over-fitting". Cleaner input variables with stronger correlations to their output rank may be needed to reduce this effect, as S-C-CE is by far the most suitable loss function for this type of multi-class classification problem. In future work, it would be worth investigating the effects of ordinal classification techniques, as the S-C-CE loss function does not punish a prediction which is incorrect by two ranks any more heavily than a prediction incorrect by one rank. The introduction of these practices may improve network accuracy, especially to within one rank of error.

The issues with distinguishing ranks 2-4 may have been exacerbated by assuming "mode of previous ranks" equals 3 for first time runners. If a relatively strong correlation exists between "mode of previous ranks" and "rank" then the network may have been trained to slightly over-select for rank 3s compared to others. This issue could be remedied by populating the variable for first time runners with a more representative distribution and this should be tested in future work.

Despite the challenges faced by the network, the result that it is accurate in 91.8% of cases to within one rank of error is highly promising. Currently, our best statistical models are approximately 80% accurate to within one rank, so this would represent a notable improvement.

Further Work

There is a great deal of opportunity for further work in this area. The next logical step would incorporate straight flat and jump races data into training and look to determine whether separate models are required to obtain equivalent results.

It may also be worth investigating techniques for improving the quality of data capture, whether this be through automation or redistribution of rankings post race result. A full correlational analysis of input variables versus their corresponding output rank should also be conducted. This will help us better understand which input variables are having the largest effect when a ranking is predicted, and allow for the removal of extraneous inputs.

QL also uses a more sophisticated rankings system when predicting future events. This incorporates full rankings for each race. Having the network automatically adapt to the required number of runners would be technically challenging, so it's likely that a hierarchy of probable rankings would need to be established for each race based on the 1-4 system. This would be relatively easy to implement by using the output probabilities from the Softmax activation function. The hierarchy could then be used to populate the field for a more complete rankings list.

Finally, it should be acknowledged that this problem could be categorised as an "ordinal" classification problem. That is, it falls somewhere between traditional classification and regression problems. This is because the relation between the ranks 1-4 represents an "order", i.e. they are related in the sense that a rank 3 is closer to a rank 4 than a rank 2 is to a rank 4. In this sense, a prediction to within one rank of error is better than to within two ranks of error. The classification groups are not entirely independent. Ideally, the loss function of the network would punish a prediction that was two ranks away from true more heavily than a prediction that was only one rank away from true. The categorical crossentropy function punishes equally for all degrees of error.

There may be scope for the incorporation of ordinal classification techniques and algorithms to improve network predictive accuracy. The effects of taking an ordinal approach should be investigated and contrasted with simple classification.

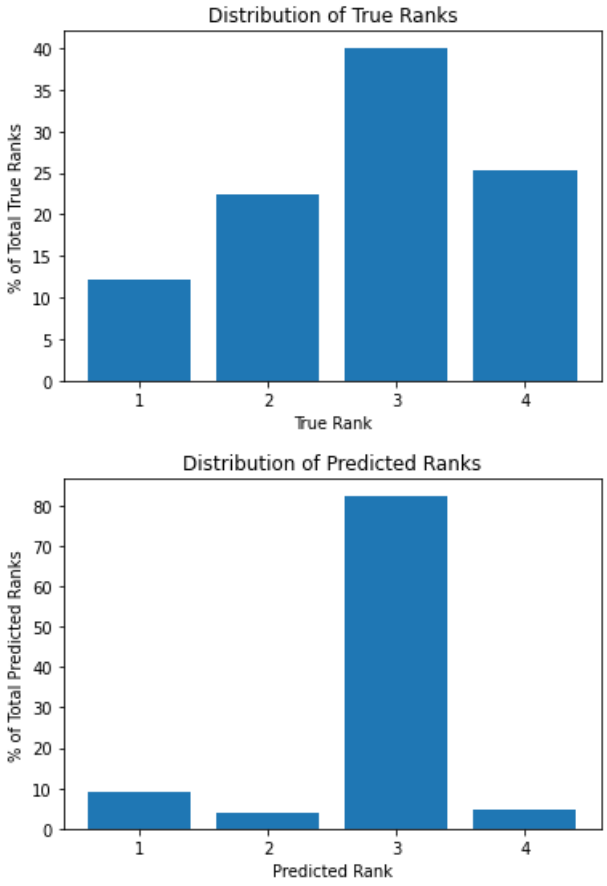

Figure 4: Plots of the distribution of true ranks and the distribution of ranks predicted by the network. It is clear to see that the network has substantially over-estimated the occurrence of rank 3s, and underestimated the overall occurrence of ranks 2 and 4.

Conclusion

This investigation proved successfully that machine learning algorithms can predict racehorse rankings with accuracy comparable to that of our most sophisticated

statistical models. The model achieved approximately 47.7% accuracy for precise ranking prediction, and 91.8% to within one rank of error.

Whilst this test has clearly shown that ML algorithms are capable of tackling the problem of ranking predictions, there are some caveats to our results. Firstly, it must be noted that straight flat races were excluded from the data set used to train the network. Upon the introduction of straight races, we would expect the network performance to decrease due to the increased complexity associated with comparing races with bends and straight races. It may be necessary in the future to train two, separate models, one for each data set. A third model may also be required for training on jumps racing data.

Secondly, the 1-4 rankings system introduces various systematic biases that could, paradoxically, lead to both improved and reduced predictive accuracy. For example, the previously discussed human biases introduced when inputting rankings 2-4 may have made it substantially more difficult for the network to differentiate between those rankings.

Future work should focus on the incorporation of a wider range of race types, and consider the discussed improvements in data collection techniques and pre-processing.