Artificial Intelligence and Machine Learning 2

Predicting the NBA Most Valuable Player

Daniyal Syed

Contents

1. Introduction

This project uses supervised machine learning algorithms to develop neural network (NN) systems with a data-driven approach. The objective of this project was to train and compare various neural network models in order to determine which one displays the highest accuracy in predicting the NBA's (National Basketball Association) MVP award.

This documentation report is divided into three main sections. The first section outlines the methods used for data collection and data pre-processing. The second section examines the trained neural network models, and the final section analyses the results of each model.

2. Data Collection Method

One of the key challenges in training an NN model is obtaining a sufficient amount of high-quality training data (Long, Shelhamer & Darrell, 2015). There are many different methods for collecting data to train an NN model, like data scraping, API access, manual collection, pre-existing datasets, etc. The method used for this task is manual data scraping. Manually data scraping is a process of collecting data from the web or other sources by manually accessing the data and extracting it (Gruß et al., 2018). This can be a time-consuming and tedious process (Kumar, Dokania & Sharma, 2018), but it may be necessary when data is not readily available or accessible through other means (Ribeiro et al., 2016).

There are several considerations to keep in mind when collecting data for training NNs using manual data scraping. One important factor is the quality of the data (LeCun, Bengio, & Hinton, 2015). It is important to ensure that the data is relevant, accurate and representative of the problem being addressed (Kumar, Dokania & Sharma, 2018). In addition, the quality of the data may vary depending on the source, and it may be necessary to clean and pre-process the data to ensure that it is suitable (LeCun et al., 2015). It is also important to consider the ethics of data collection, including issues such as privacy and consent as well (Ribeiro et al., 2016). Therefore, carefully weighing the potential limitations of this approach is necessary (Kumar, Dokania & Sharma, 2018).

To start this project, a dataset containing player, team and MVP voting statistics was needed in order to enable the models to accurately predict the MVP. These statistics were easily obtained from the Basketball Reference website. However, the data was not available in a single dataset, so it was necessary to scrape various tables from the website in order to gather all the necessary information. The collected data was saved in separate CSV files, which were later merged into a single file to create the NN models.

3. Data Pre-Processing Method

The next step was pre-processing the data. This is an essential step in the training of an NN model, as it can significantly influence the model's performance. It involves preparing the data for use in the training process, including tasks such as cleaning, formatting and augmenting the data (Goodfellow, Bengio & Courville, 2016). Some common data pre-processing techniques include data cleaning, normalisation and augmentation. Data cleaning involves removing or correcting incorrect or missing data points in the training set (Goodfellow, Bengio & Courville, 2016). This can include tasks such as fixing typos, removing duplicates and assigning missing values (Kandel, Fridman, & Rokach, 2012). This is important because even a small amount of noise or missing data can substantially impact the model's performance (Zhang et al., 2016). Another important aspect of data processing is data formatting, which involves ensuring that the data is in a suitable format (LeCun, Bengio, & Hinton, 2015). This can include tasks such as normalising numerical values, encoding categorical variables and splitting the data into training, validation, and test sets (Goodfellow, Bengio & Courville, 2016). Proper data formatting is important to ensure that the NN is able to effectively learn from the data (LeCun, Bengio, & Hinton, 2015).

In the MVP dataset, unnecessary columns were removed. In the player dataset, certain columns and punctuation marks, such as dashes and asterisks, were removed in order to ensure that player names in both datasets were consistent. In addition, some players had duplicate names due to playing for multiple teams in the same year, which required fixing as well. Once the data was cleaned, the MVP and player datasets were combined, and the team dataset was also cleaned. The team dataset was then merged with the previously combined MVP and player dataset.

4. Feature Analysis Method

There are several types of feature analysis methods that can be used to understand the relationships and importance of the features in the dataset. A correlation analysis was performed to identify the relationship between the different features in the dataset and detect which features were the most important for the model and which ones were less relevant.

Another used method was manual feature selection. A subset of the most relevant features from the dataset was selected to use in the model. It also helped in reducing the complexity of the model and improve the performance. This can be a time-consuming and subjective process, but it can be useful for identifying important features that may not be apparent through automated methods (Goodfellow, Bengio & Courville, 2016).

Finally, a data visualisation was performed to help understand and analyse the features in the dataset. The bar charts helped to identify patterns and relationships in the data that were not immediately visible. They also detected potential errors and outliers in the data.

5. Machine Learning Algorithms

Machine learning (ML) is a subfield of artificial intelligence focusing on the development of algorithms and models that can learn from and make predictions on data (LeCun, Bengio & Hinton, 2015). It has a wide range of applications in areas such as image and speech recognition, natural language processing and predictive modelling (Goodfellow, Bengio & Courville, 2016). One type of ML algorithm is the NN, which is inspired by the structure and function of the brain (LeCun, Bengio & Hinton, 2015). NNs are composed of layers of interconnected nodes, or artificial neurons, that process and transmit information (Goodfellow, Bengio & Courville, 2016). They are trained using large datasets and algorithms that adjust the weights and biases of the connections between the nodes, allowing the network to learn and improve over time (LeCun, Bengio & Hinton, 2015).

The following NNs were used to train the models:

4.1 Feedforward Neural Networks

Feedforward neural networks (FNNs) are a type of artificial NNs, used for supervised learning tasks. They consist of layers of interconnected neurons, which process and transfer information through weighted connections. The input layer receives the input data and passes it through the network and produces an output at the output layer.

FNNs have many applications, including image classification (Krizhevsky, Sutskever & Hinton, 2012), speech recognition (Hinton et al., 2012) and natural language processing (Collobert et al., 2011). One of the main advantages of FNNs is their ability to estimate complex non-linear relationships between inputs and outputs (Hornik, 1991). However, they can also be prone to overfitting, especially when the number of hidden units is too large or when the network is trained on a small dataset (Bengio, Simard & Frasconi, 1994). To improve this, various regularisation techniques, such as weight decay and early stopping, can be used (Bishop, 1995).

FNNs are a good choice for predicting NBA MVPs, because they can learn complex patterns in the data and make accurate predictions. They can process large amounts of data and learn from it, which is important when predicting the MVP, because there are many factors that can influence the outcome, such as a player’s statistics and team performance, which can be important when making predictions about the MVP.

4.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are used for processing data with a grid-like topology, such as images. CNNs are composed of multiple interconnected layers, including convolutional layers, pooling layers and fully connected layers. They are applied to various image-related tasks, such as image classification (Krizhevsky, Sutskever & Hinton, 2012), object detection (Girshick et al., 2014) and semantic segmentation (Long, Shelhamer & Darrell, 2015), because they can automatically learn spatial hierarchies of features from raw data, and they generalise well to new images.

One of their main advantages is that they can learn feature hierarchies directly from the data, rather than needing manual feature engineering (LeCun et al., 1989). This allows them to extract high level features from raw data, such as edges and shapes, which can be combined to form more complex features, like objects and scenes. However, just like FNNs, CNNs can also be prone to overfitting.

At first sight, CNNs are not necessarily a good choice for predicting NBA MVP because they are designed for image classification tasks and the used MVP dataset consists of statistics and does not have images. However, it is possible to reshape the data into a 3D tensor, where the first dimension represents the number of samples, the second dimension represents the number of features, and the third dimension represents the number of channels.

4.3 Linear Neural Networks

Linear neural networks (LNNs) are a type of NN that consists of a single layer of linear units, also called perceptrons. They are capable of learning linear relationships between the input data and the output labels. LNNs are used for various problems, including binary classification (Rosenblatt, 1958) and linear regression (Widrow & Hoff, 1960), because they are simple and efficient.

One of their main advantages is their interpretability, as the weights and biases of the linear units can be directly correlated with the importance of the input features (Bishop, 1995). LNNs are also relatively easy to train and do not require complex optimisation algorithms, as the error surface is rounded and can be solved using simple gradient descent (Rumelhart, Hinton & Williams, 1986).

However, they are generally not as powerful as more complex NN architectures, such as FNNs or CNNs, which can learn non-linear patterns in the data. Therefore, LLNs may not be the best choice for predicting NBA MVPs, as there are likely to be complex non-linear relationships between the input data and the output.

4.4 Multilayer Perceptrons

Multilayer perceptrons (MLPs) are a type of FFNs that consists of at least three layers: an input layer, one or more hidden layers and an output layer. One of the main differences between FFNs and MLPs is the number of layers. While FFNs can have any number of layers, MLPs must have at least three. Another difference is that MLPs are designed to learn complex nonlinear relationships between the input and output, while FFNs may be able to learn both linear and nonlinear relationships.

MLPs are a good choice for this task, as they are flexible and can be easily modified by adding or removing hidden layers or units, which can improve their performance.

5. Experiment Design

To conclude, four types of NN methods were used in this project: FNN, LNN, CNN, and MLN. Generally, FFNs, CNNs and MLPs, are well-suited to tasks involving complex patterns and nonlinear relationships (LeCun, Bengio, & Hinton, 2015). They are flexible and can model a wide range of problems, but they may require a larger dataset and more computational resources to train (Goodfellow, Bengio & Courville, 2016). LLNs, on the other hand, are simpler and easier to train, but they are limited to modelling linear relationships and may not be as effective for tasks involving complex patterns (LeCun, Bengio, & Hinton, 2015). The choice of NN architecture depends on the specific problem and the characteristics of the data (Goodfellow, Bengio & Courville, 2016). It is necessary to try multiple approaches and architectures to find the most effective solution.

The initial batch size for the models was 64, with 10 epochs, and this was later modified to a batch size of 32 and 20 epochs.

5.1 Evaluation

Cross-validation is a method of evaluating the performance of an ML model on a dataset (Kohavi, 1995). It involves dividing the dataset into a training set and a test set, fitting the model to the training set, and evaluating the model on the test set. The performance metric is recorded, and the process is repeated a number of times, with different combinations of training and test sets, and the results are averaged to estimate the model's generalisation error (Kohavi, 1995). This method can be used to assess an NN model's performance and tune its hyperparameters and is also useful in preventing overfitting. In this case, cross-validation with 5 folds was used to evaluate the models.

The performance of the models was also evaluated using the following metrics: mean test loss, mean train loss, and standard deviation. The mean test loss, which is the average loss of the model on the test set calculated using a loss function (LeCun, Bengio & Hinton, 2015), is a measure of how well the model is able to make predictions on unseen data. It is an important metric for evaluating the model's performance, as it helps to identify overfitting and underfitting, and to understand the stability and generalisation ability of the model. The standard deviation, on the other hand, is a measure of the spread of the data around the mean. It is calculated as the square root of the variance, which is the average squared deviation of the data from the mean. The standard deviation can be used to understand the variability of the data and to identify outliers. It can also be used to assess the stability and generalisation ability of the model, as it helps to identify overfitting and underfitting.

Accuracy, which is a measure of how well a model is able to correctly predict the target class (Sokolova & Lapalme, 2009), is calculated by dividing the number of correct predictions made by the model by the total number of predictions (Hastie, Tibshirani & Friedman, 2009).

Precision, which assesses the model's ability to correctly predict the positive class (Sokolova & Lapalme, 2009), is calculated by dividing the number of true positive predictions made by the model by the total number of positive predictions made by the model (Hastie, Tibshirani & Friedman, 2009).

Recall, which evaluates the model's capability to identify all instances of the positive class (Sokolova & Lapalme, 2009), is calculated by dividing the number of true positive predictions made by the model by the total number of actual positive instances in the dataset (Hastie, Tibshirani & Friedman, 2009).

The F1 score, which is a metric that combines precision and recall (Sokolova & Lapalme, 2009) and is useful when there is an imbalance in the number of positive and negative instances in the dataset (Hastie, Tibshirani & Friedman, 2009), is calculated by dividing the product of precision and recall by the sum of precision and recall.

6. Results Analysis and Discussion

6.1 Feedforward Neural Network

After creating the FNN model, it became evident that the model was performing poorly. A high test loss and a low accuracy indicated that the model was unable to generalise effectively to the test data. In order to address this issue, the model was fine-tuned by increasing the number of hidden layers and epochs and modifying the optimiser. The resulting model performed well on both the training and test sets, with low loss values and a high accuracy and a small standard deviation in the test set. This suggests that the model is able to generalise well to unseen data and is not overfitting to the training data.

There are several reasons why these changes had a positive impact on the model's performance. Increasing the number of hidden layers can allow the model to learn more complex relationships between the input and output data. Increasing the number of epochs can also give the model more opportunities to learn from the training data. Modifying the optimiser can help the model to converge more quickly and potentially achieve a better local minimum for the loss function.

Performance metrics:
Mean test loss: 0.003
Mean train loss: 0.003
Standard deviation: 0.002
Mean accuracy: 0.997
Precision: 0.011
Recall: 0.895
F1: 0.022

6.2 Linear Neural Network

The initial LNN model did not perform well, as shown by a high loss value and a high standard deviation. This indicated that the model was not accurately predicting the target values and that the predictions were quite varied. To address this issue, the complexity of the model was increased by adding additional layers and increasing the number of nodes in each layer. Fine-tuning the model led to an improvement in accuracy. However, in order to further improve the model, it was necessary to modify the activation function, resulting in the transformation of the model into an FNN. As a result, it can be concluded that the LNN is not an appropriate model for this task.

There are several reasons why the FNN model may have performed better than the LNN model. The FNN model has more flexibility in terms of the types of relationships it can learn, as it can learn non-linear relationships through the use of multiple activation functions. This was particularly important for this task, as the target values did not have a linear relationship with the input data.

Performance metrics:
Mean test loss: 0.408
Mean train loss: 0.409
Standard deviation: 0.782
Accuracy: 0.805
Mean precision: 0.595
Mean recall: 0.790
Mean F1: 0.641

6.3 Convolutional Neural Network

The model demonstrated relatively accurate predictions on the test set, but the predictions were somewhat inconsistent. A low standard deviation is generally preferred, as it indicates that the model is able to consistently make accurate predictions. Overall, the results of the model are satisfactory but not exceptional. In an effort to improve the model, additional hidden layers were added to increase complexity. While this resulted in the model performing better on the training data, it led to a decline in performance on the test data, suggesting the possibility of overfitting and a lack of generalisation to unseen data. To address this issue, the complexity of the model was reduced. The train loss was slightly higher than the mean test loss, which may indicate some level of overfitting, but the difference between the two was small. Overall, these results suggest that the model is capable of accurately predicting the target variable (NBA MVP) based on the input features. The model's performance on the test set is satisfactory, although there is room for improvement in terms of consistency of predictions. Further fine-tuning of the model's architecture and training procedure may be necessary in order to improve the model's performance and generalisation ability.

Performance metrics:
Mean test loss: 0.003
Mean train loss: 0.003
Standard deviation: 0.002
Mean accuracy: 0.995
Precision: 0.12
Recall: 0.950
F1: 0.025

6.4 Multilayer Perceptron

MLPs and FNNs are both FNNs, which means that they have a similar basic structure and can learn relationships between the input and output data through the use of multiple hidden layers. The FNN model uses a variety of activation functions, while the MLP model uses a linear activation function.

The MLP suggest that the model is performing well on the test data, with a low mean test loss and high mean accuracy. The mean train loss is also low, indicating that the model is not overfitting to the training data. The standard deviation of the test scores is relatively low, indicating that the model is performing consistently across different splits of the data.

The mean accuracy of 0.992 indicates that the model is correctly predicting the NBA MVP with a high degree of accuracy. This is a good model for this task, depending on the specific requirements and desired level of accuracy for the application. However, it is worth noting that the results of the MLP model may be influenced by the specific activation function used, and it is possible that other activation functions may lead to different performance results.

Performance metrics:
Mean test loss: 0.014
Mean train loss: 0.014

Standard deviation: 0.012
Accuracy: 0.992

7. Conclusion

The result of the experiment indicates that the feedforward neural network model is the best performing model out of the four. It has the lowest mean test and train loss, the highest mean accuracy, and a relatively low standard deviation. The linear neural network performed the worst, with a high mean test and train loss, low accuracy, and a high standard deviation. The convolutional neural network had a lower mean test loss and a high mean accuracy, but had a higher standard deviation and showed signs of overfitting. The multi-layer perceptron had a similar performance to the CNN, with a lower mean test loss and a high mean accuracy, but also had a higher standard deviation and showed signs of overfitting.

In general, the FNN model seems to have the most balanced and overall best performance out of the four models. It had low loss values, high accuracy and a small standard deviation, indicating that it is able to make consistent and accurate predictions on both the training and test data. While the other models also had relatively low loss values and high accuracy, they had higher standard deviations and either showed signs of overfitting or were not initially able to effectively generalise to the test data. Therefore, the FNN model is the most suitable choice for this task, as expected.

8. References

Basketball-Reference. (n.d.). NBA Awards Voting [Online]. Available at: https://www.basketball-reference.com/awards/awards_2016.html. Accessed on 2 January 2023.

Basketball-Reference. (n.d.). NBA Player Stats: Per Game [Online]. Available at: https://www.basketball-reference.com/leagues/NBA_2016_per_game.html. Accessed on 2 January 2023.

Basketball-Reference. (n.d.). NBA Standings [Online]. https://www.basketball-reference.com/leagues/NBA_2016_standings.html. Accessed on 2 January 2023.

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

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks, 5*(2), 157-166.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12, 2493-2537.

GitHub. (n.d.). Dataquest: Project Walkthroughs [Online]. Available at https://github.com/dataquestio/project-walkthroughs. Accessed on 3 January 2023.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press.

Gruß, D., Ziegler, M., & Jannach, D. (2018). Web scraping for data-driven business models: A systematic literature review. *Journal of Business Economics, 88*(4), 441-471.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction. *Springer Science & Business Media*.

Hinton, G., Deng, L., Yu, D., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine, 29*(6), 82-97.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks, 4*(2), 251-257.

Kandel, S., Fridman, L., & Rokach, L. (2012). Data cleansing: A review. *ACM Computing Surveys (CSUR),* 44(4), 15.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

Kumar, A., Dokania, P. K., & Sharma, S. (2018). A review of web scraping tools. In *2018 2nd International Conference on Computer Science and Engineering (ICCSE)* (pp. 764-767). IEEE.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation, 1*(4), 541-551.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3431-3440).

Ribeiro, B., Gama, J., Zaslavski, A., & Jannach, D. (2016). A survey of web data extraction systems. *ACM Computing Surveys (CSUR),* 49(2), 31.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386-408.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors*. Nature, 323*(6088), 533-536.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427

Widrow, B., & Hoff, M. E. (1960). Adaptive switching circuits. Institute of Radio Engineers, Western Electronic Show and Convention, *Convention Record, 4,* 96-104.

Zhang, C., Chen, S., Zhang, C., & Zhao, J. (2016). Data preprocessing in classification: A review. *IEEE Access, 4,* 2412-2426.