Predicting MLB Runs Scored: Model Evaluations

Miguel Corona

Oct 2020

1 Introduction

This document contains the efforts involved in constructing a neural network capable of predicting the number of runs a team can score provided the Batting Average, On-base Percentage, and Slugging Percentage. The model of interest shall be capable of predicting the number of runs scored throughout the duration of the 162 game MLB season. The significance in predicting the number of runs scored is to project the number of games the team can win as described by the Pythagorean Win Percentage. An accurate prediction of Runs Scored permits teams to determine whether their current roster is capable of contending or whether adjustments to the roster are necessary to contend.

2 Building a Model

2.1 Data Split

The development set was partitioned such that the training set is composed of 75% of samples and the remaining 25% are reserved for the validation set. No shuffling of the data was performed prior to the partitioning of the data into their respective sets. A randomization of samples was applied to the years in order to acquire a validation set that contains a similar proportion of winning and losing teams to that of the training set. A season contains low scoring and high scoring teams as the Runs Scored corresponds to the games won.

2.2 Linear Regression Model

A linear regression model was developed to establish a baseline for the candidate model evaluations. The requirement is to develop a neural network whose performance exceeds that of a linear regression model. A depiction of the linear regression model is found in Figure 1. The model is simple as it only contains four trainable parameters.

The linear regression model was acquired by training on the training set and its performance was observed through the validation set. The learning curves of

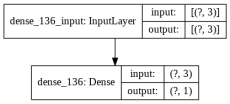


Figure 1: Linear Regression Model

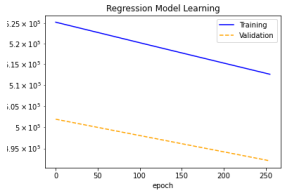


Figure 2: Linear Regression Model Learning Curve

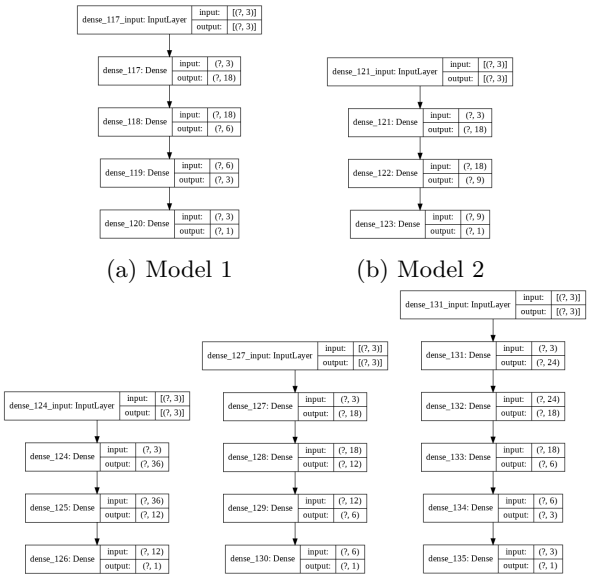
the linear regression model are depicted in Figure 2. The results of the learning curves indicate that the linear regression model does not have the capacity to learn from the data as no convergence of loss is present in either the training set nor the validation set. The training loss and validation loss trend in a parallel fashion with no indication that a suitable learning curve can be generated. The loss metrics contained within Table 1 indicate that the model performs better on the validation set than it does on the training set. The mean square error and mean absolute error are marginally better in the validation set than on the training set. The model yields negative R Square values further suggesting that the model lacks the capacity to learn from the data.

2.3 Candidate Models

A number of candidate models were developed and evaluated to identify a suitable network for predicting the Runs Scored. The architectures of these candidate are contained within Figure 3. Each candidate was trained with the same training set and evaluated evaluated the same validation set. The best network will be the one that yields the minimal loss on the validation set as the validation loss serves as an estimate for the generalization loss.

Metric	Train	Valid
MSE	512 520.326	492 147.050
MAE	711.234	696.290
R2	−61.856	−54.387

Table 1: Linear Regression Loss Metrics



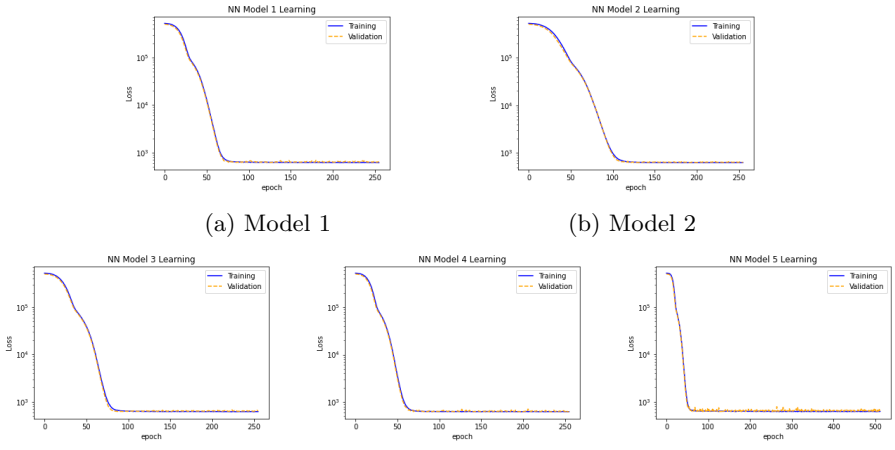
(a) Model 1 (b) Model 2

(c) Model 3 (d) Model 4 (e) Model 5

Figure 3: Candidate Neural Networks

The metrics captured for evaluation are the Mean Average Error, Mean Squared Error, and the R Square for the training and validation sets. The R Square is captured to observe the manner in which the variability of the input features affect the model. [1] The range of the R Square is measured between the values of 0 and 1 if a suitable value can be acquired and the R Square value shall assist in identifying the candidate model. The desired candidate was identified by a combination of the MAE and the R Square as the a minimal error is desired along with minimal susceptibility to the variability in the data. Simpler hypotheses are desired due to their ability to generalize.

The candidate models were trained and the resulting learning curves are depicted in Figure 4. The learning curves indicate that each model is capable of learning from the data albeit with varying convergence. Model 2 has the simplest architecture within the candidates yet it manages to yield a comparable loss on the training and validation sets much more slowly than the other candidates. Network depth plays a factor as depicted between the learning curves of Model 1 and Model 2. Model 1 and Model 2 contain the same number of neurons with the distinction that Model 1 has these neurons split across two hidden layers whereas Model 2 has them within a single hidden layer. Model 5 was developed as an augmented version of Model 1 by adding an additional layer with 24 neurons and its loss converges the quickest of all the candidates though it suffers more pronounced fluctuations in the validation loss. The number of neurons also affects the convergence as demonstrated by the learning



(a) Model 1 (b) Model 2

(c) Model 3 (d) Model 4 (e) Model 5

Figure 4: Candidate Models' Learning Curves

curves of Model 3 and Model 4. The learning curve of Model 3 is similar to that of Model 1 even though Model 3 is shallower than Model 1. Model 3 is a variant of Model 2 with each layer's neurons being a multiple of the original model's layers. Model 4 is a variant of Model 1 and no objective improvement was observed in the learning curves indicating that the number of parameters affected by both the number of layers and neurons per layer affect the model to a certain extent.

The metrics of interest are contained within Table 2. The loss metrics of the candidate models indicate that the added complexity of neural networks yield better results for regression problems. The linear regression is a simple model consisting within one neuron and therefore only has four trainable parameters as opposed to the hundreds present in the candidate models. The MSE and MAE is significantly improved with the neural networks as they were capable of learning from the data. The R squared value of the candidate models is also bound to within the region of of 0 and 1 as expected due a bounded loss that was absent in the linear regression model. The performance in the loss metrics show display the effects of the hyperparameters on the learning process. Model 1 outperformed the shallower Model 2, but its performance was comparable to Model 3 which is an augmented version of Model 2. The metrics of Model 4 yield an improved MSE and MAE on the training data compared to Model 1 yet does not improve on the validation set suggesting that the added neurons don't yield a benefit. Although the loss metrics indicate that a complex model outperforms a linear regression model, it also demonstrates that performance on the validation set degrades as more complexity such as that between Model 1 and Model 5 as Model 5 contains a supplementary initial layer.

Metric	Train	Valid
MSE	619.538	640.036
MAE	19.769	19.838
R2	0.924	0.928

(a) Model 1

Metric	Train	Valid
MSE	623.619	640.303
MAE	19.995	19.992
R2	0.924	0.928

(b) Model 2

Loss

Metric	Train	Valid
MSE	614.267	636.026
MAE	19.790	20.059
R2	0.925	0.928

(c) Model 3

Metric	Train	Valid
MSE	615.694	620.694
MAE	19.779	19.851
R2	0.924	0.93

(d) Model 4

Metric	Train	Valid
MSE	661.576	655.194
MAE	20.636	20.911
R2	0.919	0.926

(e) Model 5

Table 2: Candidate Models Loss Metrics

2.4 Model Represented as a Function

A function was developed to simulate the tensor operations that occur within the neural network to yield predictions. The trainable parameters from the best performing candidate were read into the function along with the data to acquire the simulated model's predictions. The outputs of the function were compared to the model's outputs and to the true outputs. The results of the comparisons are contained within Table 3. The loss metrics between the function's outputs and the data are marginally worse than those of the model. The loss metrics between the function's output and the model's output are minimal. These discrepancies are believed to be attributed to floating point error accrued per individual tensor operation such as acquiring a dot product via combination of broadcast multiplication and a summation.

Metric	Train	Valid
MSE	30.010	24.181
MAE	4.520	4.180
R2	0.996	0.997

(a) Function Loss vs Model

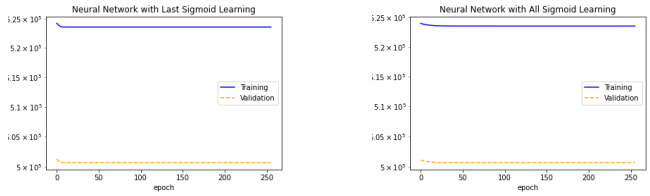
Metric	Train	Valid
MSE	633.667	663.405
MAE	20.140	20.209
R2	0.922	0.925

(b) Function Loss vs Data

Table 3: Function Loss Metrics

2.5 Effects of Incorrect Activation on Model

The effect of utilizing the incorrect activation function were observed to acquire confidence in the proposed model. The problem of interest is that of regression such that the output is a prediction for Runs Scored a team can yield given a team's BA, OBP, and SLG. Two additional training cycles were performed on the selected model from the candidates. One training cycle involves utilizing a sigmoid as the activation function for the final layer in the network and the second involves setting all activation functions to sigmoid.



(a) Last Layer Sigmoid (b) All Layers Sigmoid

Figure 5: Effects of Incorrect Activation Functions

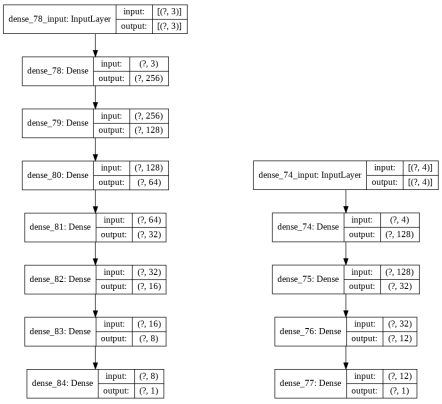
The learning curves of both experiments are contained within Figure 5. The learning curves indicate that the model is unable to yield a meaningful output if there is at least one incorrect activation function regardless of its location within the network. The models yield poorer results than those of a linear regression models. The linear regression models at a minimum demonstrate a decreasing loss that is absent from complex models with incorrect activation functions. The models with the incorrect activation functions behave similarly to a model with insufficient capacity as the training loss and validation loss remain large and constant throughout the duration of training.

3 Overfitting Models

Overfitting is the phenomena in which the model fails to generalize data and only performs well on the training set. The training loss and validation loss diverge such that the model can only provide reliable predictions for the training data. Factors such as an overly complex architecture or prolonged training can attribute to overfitting. The model's architecture should not be overly complex in order to prevent the training data from being within the model's capacity or memorization ability. This portion of the experiment deals with the efforts involved in developing a neural network that overfits the training data with the use of significantly more complex architectures than those present in the candidate models for one model and adding the output as an input to a second model.

3.1 Identifying an Overfitting Model

A model whose architecture is more complex than the underlying patterns within the data is capable of overfitting. Overly complex networks are capable of memorizing the output of each training input instead of learning and may not be able to provide a reasonable output for unseen data. One model identified to be capable of overfitting is depicted within Figure 6. The overfitting model is drastically more complicated than the candidate models in terms of the depth and the number of neurons per layer. The number of trainable parameters between the overfitting model and the candidate models are of different magnitudes and thus the training epochs were adjusted to account for the additional training that is required.



(a) Overfit

(b) Overfit with Output as Input

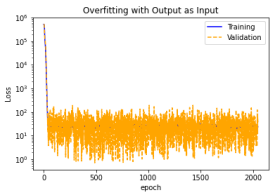
Figure 6: Overfitting Models

The learning curves of the overfitted model are depicted in Figure 7. The overfitted model yields a learning curve whose validation loss slightly diverges from the training loss and fluctuates throughout its training. The magnitude of the fluctuations of the validation loss indicate that the model is unable to yield reliable predictions as the model may be memorizing the output of the training samples and minimizing the effects of updating the loss function to preserve the known outputs. The loss metrics contained within Table 4a yield drastic differences in the MAE and MSE between the training set and validation set solidifying the belief that the loss will continue to diverge. The magnitude in the differences in the training loss and validation is greater than those observed in the candidate models. The candidate models with their simpler architecture were capable of yielding lower loss than the complex model.

3.2 Overfitting with Output as Input

A simpler model architecture capable of overfitting the data was identified by providing the desired output as an input for each training sample. The exercise of adding and the output onto the input solidifies whether the developed model is capable of learning. The training set was modified to contain an additional feature to correspond with the output and the value was normalized via the Z-Score similarly to other features . The outputs remained unchanged as the desired predictions for the model are the Runs Scored. The model capable of overfitting provided that the the output is given as an input is depicted in Figure 6. The model is far more complex the candidate models developed, but much simpler than the previous overfitting model.

(a) Overfit



(b) Overfit with Output as Input

Figure 7: Overfitting Models Learning Curves

Metric	Train	Valid
MSE	726.806	895.188
MAE	21.809	24.489
R2	0.911	0.899

(a) Overfitting Model

Metric	Train	Valid
MSE	14.747	16.251
MAE	3.706	3.901
R2	0.998	0.998

(b) Overfitting Model Output as Input

Table 4: Overfitting Models Loss Metrics

The learning curves of the overfitted model with the output as an input are depicted in Figure 7. The validation loss behaves does not have a noticeable divergence, but instead fluctuates wildly across the training. The fluctuations are absent in the training loss as the model memorizes the outputs of each training input. The oscillations in the validation loss reach values that exceed those of the training loss, but also reach levels where the model performs better on the validation set. The corresponding metrics acquired from the training are contained within Table 4b. The MAE and the MSE on the training set and validation set are the minimum values observed in the experiment. The addition of the output minimizes the respective losses, but also displays the

disparity between the validation and training loss as the relative differences are greater than those in the candidate models.

4 Conclusion

It was observed that a neural network can outperform a linear regression model in predicting the Runs Scored based on the offensive statistics. The added complexity in the networks permits them to learn more from the data than the linear regression model due to the additional trainable parameters. The network benefits from the added complexity, but only to a certain extent as the performance may peak or even degrade for more complex architectures. Complicated networks take more time to train or even worse may overfit the data to yield a model incapable of generalizing the data. The complexity of the network, in terms of the trainable parameters, is influenced by the hyperparameters such as the depth or the number of neurons per layer. The usage of the activation functions also plays a role in the model's performance as it was also observed that a model whose hyperparameters perform well on the data can yield unusable results if it utilizes the incorrect activation function.

References

[1] Wu, "3 Best metrics to evaluate Regression Model", towards data science, Medium, https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b