CS230

Neural Network Models for Predicting NFL Play Outcomes

Xuyi Guo
Department of Computer Science
Stanford University
xuyguo@stanford.edu

Abstract

Sports analytics has revolutionized the way that teams operate, from recruiting players to making calls on the field. American football stands out as a major sport where each play begins as a set play, rather than the game progressing continuously. We try to predict the outcome of American football plays based on information on game state and play selection. Taking a multi-class classification approach, we experimented with different neural network models. None of the models were able to meaningfully predict the outcome of the plays based on the features provided.

1 Introduction

In American football, each play begins with the two teams of 11 men lining up across from each other on their respective sides of the line of scrimmage. Once a play begins, they play until either the ball is downed or someone scores. Then for the next play, the teams set up again, with the line of scrimmage moving accordingly based on yards gained. This gives football a discrete set of "pre-snap" (before a play begins) game states based on the position on the field, score, time remaining, and yards to go. We experimented with a multi-class classification approach to see if we could predict the outcome of a play based on game state using a neural network. The data we used was play-level data of every NFL play from 2009 to 2018, from a data set created by Horowitz, Yurko, and Ventura [1].

We were interested primarily in the number of yards a play would gain, and considered a regression approach. However, we ended up going with a classification approach instead since it would be more clear how to interpret whether a result was good or not. Our baseline model was a 1 layer neural network with softmax regression. We further experimented by adding more layers to the model from 2 through 6, changing the number of nodes in the hidden layers, and adjusting other hyper-parameters. The models overwhelming predicted plays to be of 0 gain, with only the deeper models adding in some predictions of short gains. This resulted in poor accuracy of about 22 percent.

2 Related work

Sports analytics has taken off alongside the growth of machine learning. A number of papers have incorporated deep learning together with player tracking data. Yurko et al. present a model to predict the result of a play in real time using player tracking data [3]. Burke uses tracking data on quarterbacks specifically, combining deep neural networks with player tracking data to try to evaluate quarterback decision making [5]. There is also substantial literature in trying to predict coaching decisions. Fernandes et al. compare several different models for predicting whether a play is going to be a run or a pass, using a combination of play level data and player data. [6]

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

We took the 2009-2018 data and split it into training and validation/test data by year. Training data was years 2009-2017 and validation data was year 2018. We decided to split the data by year since the goal of the project was to try to predict the outcome of football plays. In a real-world application, a football coach would have all the data from previous seasons to aid in predicting the outcomes of plays in the coming season, so we split the data to reflect that. After pre-processing the data, we had 286524 training examples and 29219 validation examples.

The data set we used was already pre-processed, but we contributed some of our own pre-processing as well. First, we limited plays to only "pass" and "run" type plays. This excludes special teams plays, such as kick-offs, punts, and field goals. Special teams plays are subject to different rules from passing and running plays and do not have a comparable yards gained statistic to passing and running plays. We also removed the plays that resulted in penalties. The interpretation of the yards gained from a penalty is also not comparable to a penalty-free play and there was no information about the location of the play (e.g. right or left side of the field) when a penalty was called. Four additional features were added to the data set. Those were a count of the number of run plays the offensive team had run in the game prior to the play being called, a count of the number of pass plays the offensive team had run in the game prior to the play being called, the average yards gained on run plays the offensive team had run in the game prior to the play being called, and the average yards gained on pass plays the offensive team had run in the game prior to the play being called. These features were created to try to offer a measure of how well the team was doing in their run and passing attacks.

The categorical features we used included: home team, away team, team on offense, team on defense, play type, and specifics about the location of the play if it was a run play. The running play features included run location, which indicates whether the run was to the left, right, or up the middle, and run gap, which indicated where the run was directed relative to the offensive line. We did not include passing play location information for two reasons. First, it had too much information about the yards gained because the data was taken after the play occurred. This was one of the weaknesses of the data set we had for this problem. The data set specified whether the play was a pass or run and where the play was located, but only after the play was actually run. This means that we do not know what play was actually called by the coach and information about the location of the play results from the decisions made by the players while the play was occurring. Also, the pass plays had no location information in the cases that the quarterback was sacked. That means the absence of location information on a pass play would perfectly predict that the quarterback was sacked. For all the categorical features, we converted them to one-hot matrices.

The numerical features we used included information about the time left in the game, yards for a first down, position on the field, the four features we created about the average outcomes of a team's previous run and pass plays, indicators about the position of the quarterback in the formation, and information about the state of the game, such as score and timeouts remaining.

We created eight output categories. Six of the output categories were based on yards gained, one was for plays that resulted in a turnover, and one was for plays that resulted in a touchdown. The categories were as follows:

1. Negative yards gained.
2. Zero yards gained.
3. Between 1 and 4 yards gained.
4. Between 5 and 9 yards gained.
5. Between 10 and 19 yards gained.
6. Over 20 yards gained.
7. Turnover.
8. Touchdown.

Turnover and touchdown had priority over yards gained. That is, if a play resulted in a turnover or touchdown it would be categorized as such regardless of the yards gained. Zero yards gained was the most common result.

4 Methods

We approached the problem using neural networks with the ReLU (rectified linear unit) activation function for the hidden layers defined as:

$$f(x) = max(0, x).$$

The ReLU function is a non-linear function, which is a piece-wise composition of linear functions. The ReLU activation function was chosen because it's relatively faster to train and overcomes vanishing gradient issues. For the output layer we used a softmax activation function defined as:

$$p(y = j|z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{j=0}^{k} e^{z^{(i)}_k}}.$$

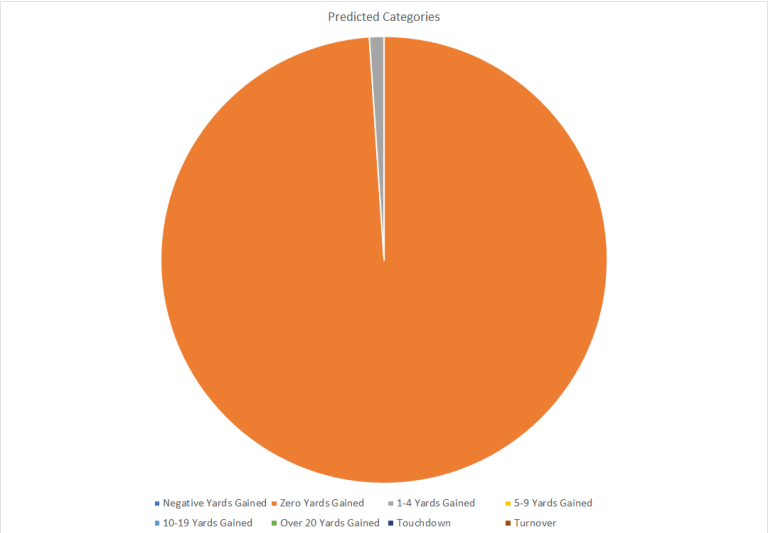
The softmax function serves as a generalization of the sigmoid function from logistic regression for multi-class classification. It takes an input vector and normalizes it into a vector of probabilities proportional to the exponentials of the input vector. From these probabilities we select the highest one as the prediction for that example. Then the corresponding loss function was the cross-entropy loss of the softmax function:

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{N} y^{(i)} \log \hat{y}^{(i)}.$$

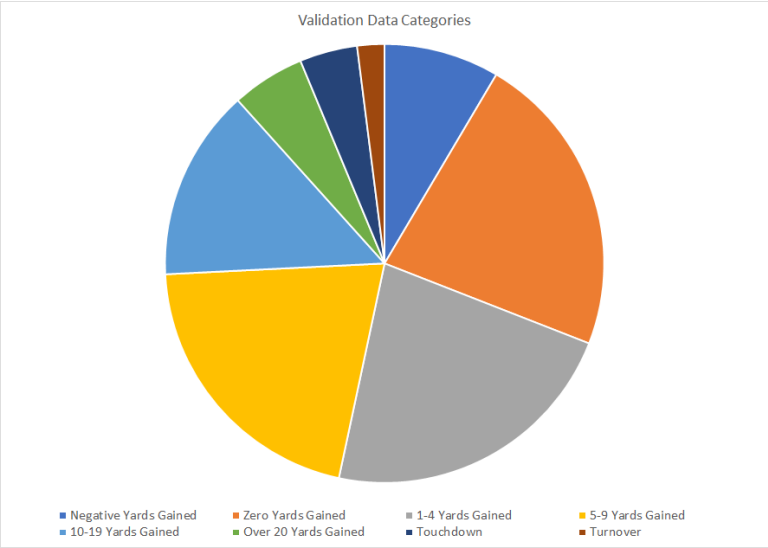
We used the Adam optimizer that was introduced by Kingma and Ba in 2015 to minimize cost [4]. We used mini-batch gradient descent, randomly selecting a subset of the training data at each step rather than trying to take the gradient over the entire data set. Mini-batch gradient descent was used to help speed up learning. For the implementation of the neural network, we used the Tensorflow library. [7]

5 Experiments/Results/Discussion

We experimented with a number of different hyperparameters for our neural network, but none of our models were able to achieve good accuracy on the validation/test set or the training set. For the depth of the neural network, we tried neural networks ranging from 1 to 6 layers. For the number of nodes in the hidden layers, we experimented with values going by powers of 2 from 32 to 2048. For mini-batch size we chose 4096. We settled on 4096 since it performed well in terms of learning speed. Initially all the models we tried were predicting zero gain for all plays, so we introduced dropout to see if it would help rectify what may be an issue of over-generalization. We tried dropout in increments of 0.05 from 0.1 to 0.5. The deeper models with four or more layers did make predictions other than zero gain with dropout rates of 0.2 or higher. However, the predictions were still overwhelmingly zero gain and accuracy was around 22 percent.



The true distribution from the validation data was far different, although zero gain was the most common category.



Higher dropout rates such as 0.4 or above saw more diverse predictions, but actually had lower accuracy around 19 percent.

6 Conclusion/Future Work

We tried numerous configurations of a neural network model that uses the softmax function for multi-class classification to try to predict the outcome of plays in the NFL based on game state. None of those models saw much success, with most of them consistently predicting the most common category, zero gain. The inclusion of player data could be a useful of addition in generating predictions in the future. Fernandes et al. presented a novel method of quantifying player ability, pulling their rating from the video game *Madden* [6]. Other architectures could also have merit. Since football games consist of a sequence of plays and the effectiveness of plays earlier in the game may be an indicator of how effective they will be later, exploring the use of sequence models could be valuable.

References

[1] Horowith, M., Yurko, R., Ventura, S., (December, 2018). Detailed NFL Play-by-Play Data 2009-2018. Retrieved April 22, 2020 from https://www.kaggle.com/maxhorowitz/nflplaybyplay2009to2016?select=NFL+Play+by+Play+2009-2018+

[2] Pelechrinis, K., Winston, W., Sagarin, J., & Cabot, V. (2019) Evaluating NFL Plays: Expected Points Adjusted for Schedule In *Machine Learning and Data Mining for Sports Analytics*, pp. 106–117. Springer International Publishing.

[3] Yurko, R., Matano, F., Richardson, L., Granered, N., Pospisil, T., Pelechrinis, K., & Ventura, S. (2020). Going deep: models for continuous-time within-play valuation of game outcomes in American football with tracking data. Journal of Quantitative Analysis in Sports. 10.1515/jqas-2019-0056.

[4] Kingma, D., Ba, J. (2015). Adam: A Method for Stochastic Optimization. *ICLR 2015* The Hilton San Diego Resort and Spa. May 7-9 2015. Retrieved from https://arxiv.org/pdf/1412.6980.pdf

[5] Burke, B. (2019). DeepQB: Deep Learning with Player Tracking to Quantify Quarterback Decision-Making Performance.

[6] Fernandes, C., Yakubov, R., Li, Y., Prasad, A., Chan, T. (2020) Predicting plays in the National Football League. Journal of Sports Analytics.

[7] Abadi et al. (2015) TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.