Predicting Intra-Game Outcomes with Neural Networks: A Paradigm for Business Strategy

Gautam B. Singh
Professor
Department of Computer
Science & Engineering
Oakland University
Rochester, Michigan, USA.

Kabir Singh
Software Engineering
Ford Motor Company
Wayne State University
Detroit, Michigan, USA

Abstract

The goal of this project is to extend neural networks and mulitlayer perceptrons, commonly used for classification problems, for predicting a continuous value. The specific application we address in this paper is to predict the number of yards a running back will gain in a given play situation or configuration captured with 29 variables. The data utilized comes from past plays of the National Football League. While the application domain considered is game playing, the modeling techniques discussed can be adapted for decision making in developing business strategy with a focus on predicting individual events to enable decision making to reach an overarching goal.

Keywords

Football, Business strategy, Artificial neural network. Machine learning, Feature analysis.

1. Introduction

This project concerns with plays conducted during games of American Football in the National Football League (NFL). The NFL is the premier league for American Football in the world.

American Football is a sport played by two teams with 11 players on a 100-yard field of play. At the end of either side of the field is the “end zone” which is 10 yards long, making the total play area 120 yards long. The team that controls the ball is on the offense. The goal for the offense on a given play is to advance the ball by either running (rushing) or throwing (passing) the ball. The quarter back is responsible for passing the ball while the running back is responsible for running or rushing on any given play.

The offense scores seven points for getting the football into the opposing teams endzone, and scores three points by kicking a field goal. The goal of the defense is to stop the offense from advancing the ball, and to ultimately get their team on offense. The offense must gain at least 10 years with four downs, or loose possession of the ball to the other side.

On any given team, approximately one third of all offensive plays called will be rushing plays. So, it is important for a team to decide when to rush the football. A predictive algorithm that can take the game conditions into consideration and predict the yardage that will be gained by rushing will thus be of immense value to a game in strategi-cally deciding their play and maximize the overall chances of winning the game. Such a predictive model must take into consideration several factors as discussed below which broadly capture the strengths of a given team, the conditions of a game, and a specific parameterization of a given play.

The differentiation of the proposed model is that it is a process centred model applicable for maximizing the overall performance and success of business. As in the game of American Football, each decision such as bids, asset acquisition, pricing and marketing allocation can be similarly modeled as a play. The overall objective is to make the business succeed just like it is to win a football game. Integrating machine learning into each business decision can help us achieve this goal by developing a winning strategy at each step of the way.

2. Problem

The goal of this project will be to analyze a dataset containing information about a number of rushing plays and to build a model that will predict, on a given play, how many yards a rusher will gain.

There are several applications this information will be relevant. It will help augment current domain knowledge to decide when it is more advantageous to rush or pass. It would give the coaching staff on NFL teams better insight into the different attributes that impact a running back’s performance on a given play and how they can be optimized to potentially enhance his performance. This model would also be relevant to NFL broadcasters so that live play-by-play commentators have an additional point of context for action on the field.

3. Dataset Overview

The data from this competition is from the NFL Big Data Bowl hosted on Kaggle from NFL Next Gen Stats. It contains information from 23171 rushing plays that are from 512 games.

This dataset contains 49 different attributes to be analyzed. These attributes can be roughly divided into three categories: player data, game data, and play data. Player Data refers to data that is associated with a given NFL player.

Shown in Fig. 1 are the set of attributes captured for each game. This Game Data is data that stays constant throughout a given NFL game.

Variable Name	Description	Data Type
GameId	Unique game identifier	Nominal
Team	Home or away	Ordinal
Season	Year of the Season	Interval
HomeTeamAbbr	Home Team Abbreviation	Nominal
VisitorTeamAbbr	Visitor Team Abbreviation	Nominal
Week	Week of Season	Interval
Stadium	Stadium where game is played	Nominal
Location	City where game is played	Nominal
StadiumType	Description of environment	Nominal
Turf	Field Surface	Nominal
GameWeather	Game Weather	Nominal
Temperature	Temperature	Interval
Humidity	Humidity	Interval
WindSpeed	Wind Speed	Ratio
WindDirection	Wind Direction	Nominal

Figure 1: Features Captured for a Game

There are four different types of data shown in the table. Two data types for non-numeric data are Ordinal and Nominal. And the two data types for numeric data are the Ratio and Interval data types. Nominal data is a group of non-parametric variables where there is no order defined between the values of the variables. The Ordinal data on the other hand is group of non-parametric values for which some sort of order does exist. Thus, the difference between the nominal and ordinal variables is that ordinal variables can be placed into some kind of order by their position.

Numerical data belongs to two different classes – Interval and Ratio. An interval data is a type of Ordinal data where the difference between two values is meaningful. Examples of interval variables include a student's SAT scores, or week number in a year, person's credit scores. A ratio variable in addition to being an interval data also has a clear definition of zero. When a ratio variable equals 0.0, its significant of non-existence. Some examples of ratio variables include an object's weight, length, medicine dose amount. The connotation of the phrase ratio is that while working with ratio variables, the ratio of two values will have an associated interpretation.

For the Game Data shown in Fig. 1, the temperature, humidity, and the week number when the game is being played are all interval variables while the wind-speed is the only ratio variable used.

Next shown in Fig. 2 is the set of features captured for a given play. The Play Data changes on every play. It is designed to capture features that are dynamic and reflect the specifics that determine the manner in which that specific play will best be handled. Several ratio variable types are used as features for the Play Data. This data captures several details related to how much time is left in the play, which quarter is being played, what stage of the offensive possession the game is being played (i.e. how many downs), the number of defenders near the line of scrimmage, how much distance is needed to be traveled, and the like.

Variable Name	Description	Data Type
Play Id	Unique play identifier	Nominal
Yard Line	Line of scrimmage	Ratio
Quarter	Game Quarter	Ratio
Game Clock	Time on the clock	Ratio
Possession Team	Team with Possession	Nominal
Down	Down of play (1-4)	Interval
Field Position	Side of field of current play	Nominal
Home Score Before Play	Score of home team before play	Interval
NflId Rusher	NFL Id of Rusher	Nominal
Offense Formation	Offense formation	Nominal
Offense Personnel	Offensive team grouping	Nominal
Defenders In The Box	# of Defenders near line of scrimmage	Ratio
Defense Personnel	Defense Formation	Nominal
Play Direction	Direction of play	Nominal
Time Handoff	UTC time of the handoff	Ratio

Variable Name	Description	Data Type
Time Snap	UTC time of snap	Ratio
Yards	Yards gained on play	Ratio
Distance	Yards needed for a first down	Ratio
Visitor Score Before Play	Score of visitor before play	Interval

Figure 2: Features Captured for a Specific Rushing Play

We also have features that capture information about a Player as shown in Fig. 3. These features capture information about a specific player, including their position, speed, acceleration, as well as their height and weight. Some of the information like the position of the player and their orientation is also captured.

Variable Name	Description	Data Type
X	Player position along x axis	Ratio
Y	Player position along y axis	Ratio
S	Speed	Ratio
A	Acceleration	Ratio
Dis	Distance traveled	Ratio
Orientation	Orientation of player	Ratio
Dir	Angle of player motion	Ratio
NflId	NFL ID	Nominal
Display Name	Name of player	Nominal
Jersey Number	Jersey Number	Nominal
Player Height	Height of player	Ratio
Player Weight	Weight of Player	Ratio
Player Birth Date	DOB of Player	Ratio
Player College Name	College of player	Nominal
Position	Player position	Nominal

Figure 3: Features Captured for a Specific Player

Thus, there are three classes of data that are interacting with each other that will be analyzed to get to the final result (Player, Game, and Play Data). Every play is associated with 1 game and every game is associated with multiple plays (approximately 40). Every play is associated with 22 players (11 offensive players and 11 defensive players). This means in the dataset there will be 22 rows of data for every play.

4. Prepossessing

Yard line: This refers to the line of scrimmage where the players are starting the play from. By itself, this is not a very useful metric. If an offensive play begins from the one-yard line, the offensive team could either be one yard away from scoring or be 99 yards away from scoring. This fact can be derived based on the "HomeField" and "PlayDirection" attributes, and while it is possible that a Neural Network can derive the rule based on those attributes independently, it would make the process more robust if this was an attribute that we fed to a model. So, we derived an attribute "YardsToTD" based on the attributes "YardLine", "HomeField", and "PlayDirection." The value of "PlayDirection" was first converted to an integer (0 or 1) assigning right to 1 if the home team is on offense.

Snap and Handoff: To enhance the meaning of some the variables, modifications will applied. This dataset includes the precise UTC time of Snap (time when play starts) and time of Handoff (when the running back is given the ball). Rather than including these two times separately, we included the difference between the two times, i.e. how much time elpased before the quarterback handed the ball to the running back as a latent variable for training the model.

Turf: The turf attribute refers to the surface on which the game is played. There are several details about the turf that were included in the dataset. Intuitively however, the fundamental difference that makes most impact on a play is categorize the turfs into artificial or natural. Since there are relatively few unique values it was possible to create this mapping of the turf into artificial or natural.

Wind direction: The attribute "WindDirection" Attribute has a significant number of missing values. Additionally, among the values it does have, a number of them appear to be a relatively verbose description, requiring NLP techniques to parse. Hence, this attribute was disregarded. The attribute "GameWeather" was also disregarded for similar reasons.

Wind speed: The windspeed data needs to be cleaned before dealing with any missing values. If the value is a number, then that number can be used. Otherwise, “MPH” or “mph” was removed from the value. Additionally, there are attribute values that list a range, so we can take the average of those values. Missing values for wind speed were filled with the mode of this attribute, which was 5.

5. Results

Machine learning using neural network is an example of supervised learning, and specifically a type of inductive supervised learning. Artificial Neural Networks (ANN) architecture is inspired by the architecture of brains where the intelligence is created by collective stimulation and highly interconnected neurons. A neuron is like a gate which collects stimulation from a number of other neurons and “fires” when a weighted collected strengths of its inputs exceeds a threshold. The result of the firing of a neuron is transmitted as an axon potential and becomes the inputs for a number of other neurons. ANNs are trained through examples by a process known as back-propagation where the weights responsible for a wrong decision are reduced by small increments. Similarly, the weights responsible for a correct decision are incrementally increased. These increments are often referred to as the learning rate in this penalty and reward based training process.

5.1 Neural Network Architecture

As previously discussed, the fundamental building block of an ANN is a neuron. An ANN is ultimately a collection of interconnected neurons. One of the problems in Machine Learning is to determine the appropriate architecture of a neural network that is suitable for solving a specific problem. As discussed below, the components of the ANN architecture include the specifications of a neuron, the number of hidden layers, and the number of neurons in each of the hidden layers.

Neuron: A neuron is the fundamental building block of an Artificial Neural Network (ANN). A single input neuron, shown in Fig. 4 is the fundamental building block for neural networks. A neuron perform three functional operations. The figure uses the architecture to describe the i^{th} neuron. First, it multiplies each component of a p-dimensional input vector (x) with a corresponding weight in a weight vector (w_i). Second, the weighted sum is added to a bias (b_i) to produce the net activation value (n_i). Third, the net activation value is transformed using a differentiable transfer function (f_i) to produce the final output of the neuron (a_i). Thus the processing by a neuron is summarized by the following equation Eq. 1.

$$a_i = f_i\left(\sum_{j=1}^{p} x_j * w_i,_j + b_i\right) \tag{1}$$

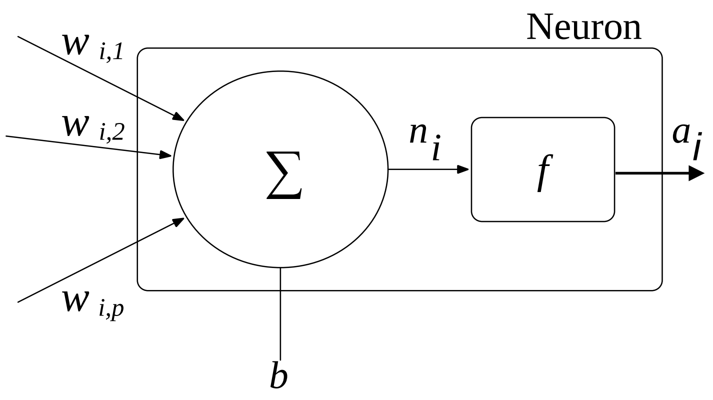

Figure 4: A Neuron – Building Block of Artificial Neural Networks

There are also many types of transfer functions utilized with the linear and sigmoid transfer functions being the ones most commonly utilized. The linear transfer functions are often used as the final stage of a multi-layer network since there is no limit on the values produced by the network. In contrast to the linear transfer function, the sigmoid ($1/(1 + e^{-n})$) transfer function produces an output in the range 0 to 1. This type of transfer function is mostly used in the hidden layers of a network as discussed below.

Note that the subscript *i* is used to designate the i^{th} neuron. A similar functional processing is performed by all neurons in the network. The weights (w_i) and bias (b_i) are both adjustable parameters of a neuron with the training process of the neural networks being such that these parameters are adjusted to achieve a desired behavior.

Hidden Layers: In a multilayer neural network, the processing functions of the neural network is cascades through a series of hidden layers. From a standpoint of terminology, all layers with the exception of output layer, are called the hidden layers. The ANN shown in Fig. 5 thus comprises of two hidden layers.

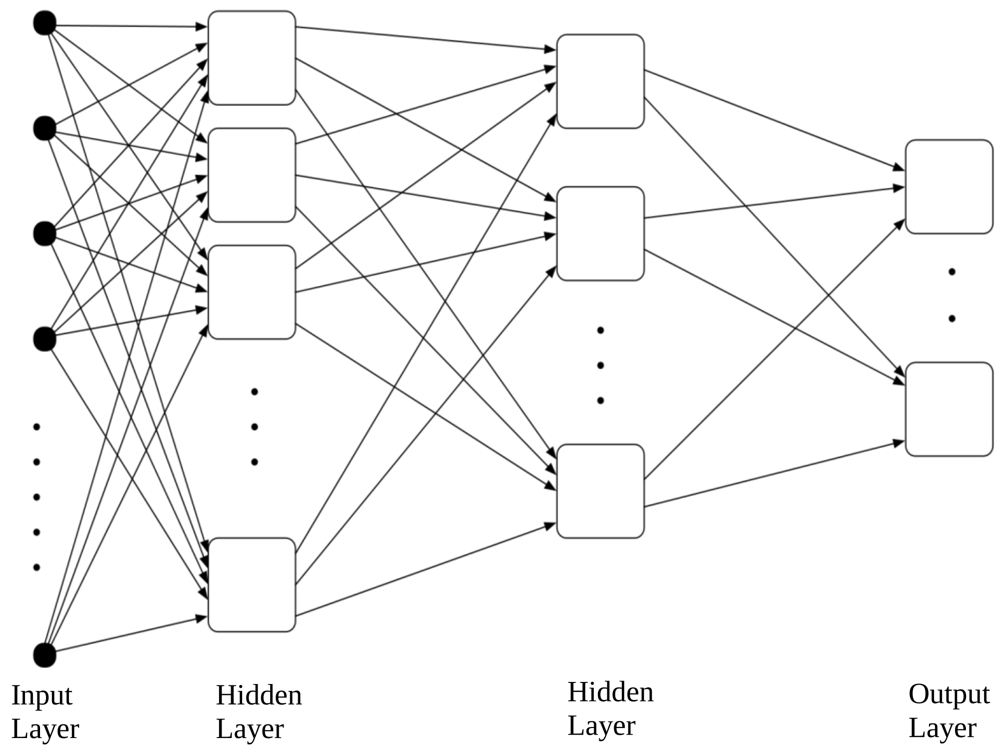

Figure 5: A Feed-forward Neural Network with Two Hidden Layers

The number of neurons in the first hidden layer does not necessary have to equal the number of dimensions *p* to the input vector. Some authors refer to the first hidden layer as an input layer.

Each layer of the network allows for complex modeling of functions to produce a complex clustering of data. Generally, all layers in a specific layer of the network utilize similar transfer functions and biases.

ANN Architecture: In our model, we have utilized an ANN with one hidden layer and one output layer. The hidden layer utilizes a sigmoid transfer function and the output layer utilizes a linear transfer function. The hidden layer has three neurons, and the output layer has a single neuron. The ANN architecture is shown in Fig. 6.

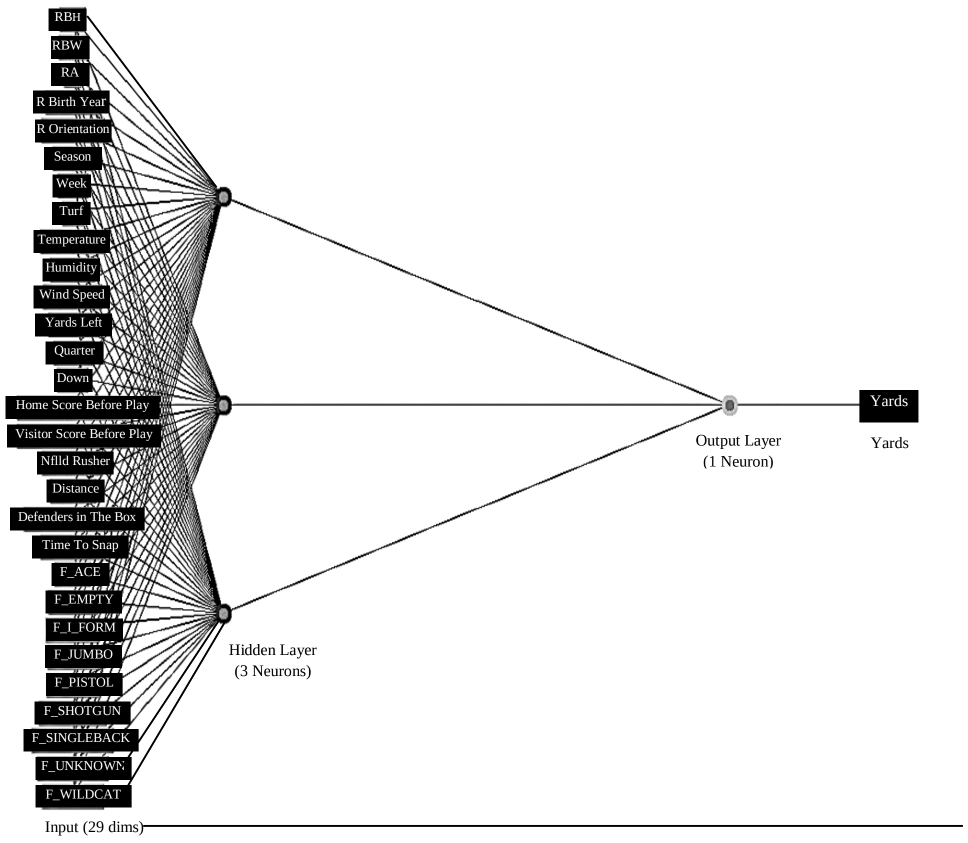

Figure 6: ANN utilized for Predicting Yardage based on a 29-dimensional feature vector to predict yardage for a particular in-game outcome. The network utilized comprises of a single hidden layer with three neurons (with sigmoid transfer function), and a single output layer (with linear transfer function).

5.2 Testing

We utilized the Weka machine learning workbench to build and evaluate the neural network model in this study. One of determining factors for using Weka was its provision for Python API, a popular language being used in Data Science Applications including the Business Analytics and Econometrics.

The process of testing entailed using 90 percent of randomly selected samples for training the model, and using the model to test the accuracy on the remaining samples. For the purposes of evaluating the performance, a margin of error was assigned. Thus, if the margin of error is set to μ, and the predicted value is of yardage gained is ŷ, then the predicted yardage will be deemed to be correct if the $|y - \hat{y}| \leq \mu$, where y is the true yardage from the dataset.

Fig. 7 shows a graph of the accuracy of predicted values as a function of the permitted error in the margin of accuracy.

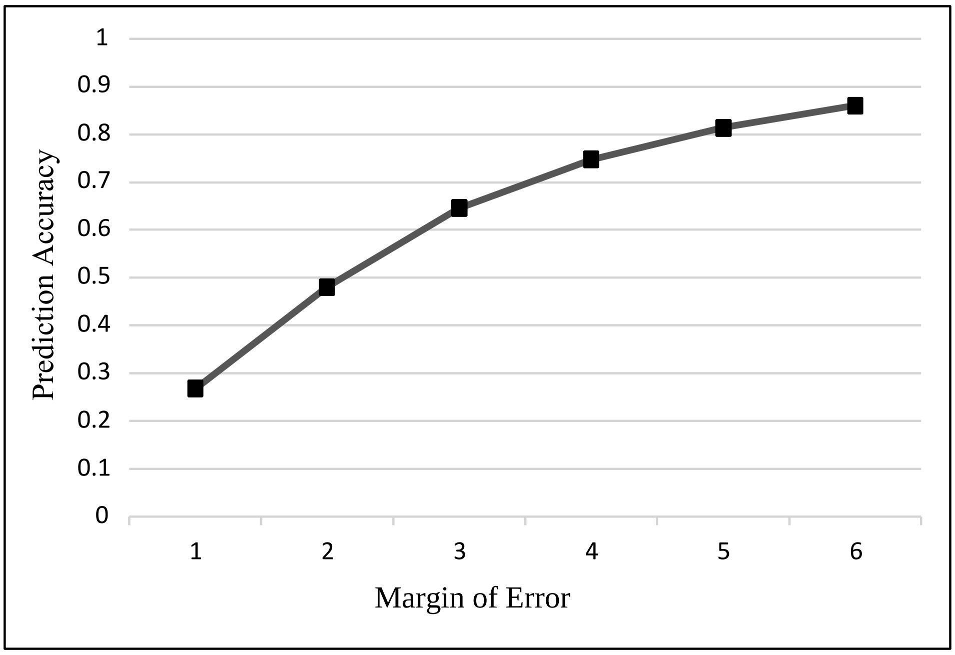

Figure 7: Accuracy of Predicted Yardage as function of Margin of Error.

The dataset was further analyzed to determine the effect of the availability of training samples on the accuracy of prediction. Since the ANN is a type of machine learning model that learns the patterns in data, the training will naturally be inadequate when there is insufficient data to learn from. This is precisely what we observed as shown in Fig. 8.

The histogram in Fig. 8 depicts the frequency of dataset that provides the values of features for a specific yardage. As we see, there is sufficient data available for the cases where the yardage gained is 0 to 5 yards. In contrast, there are not many cases where the runner lost yardage or gained more than 10 yards.

Depicted in Fig. 8 is the error in predicting the yardage for each of the dataset frequency category.

As is quite apparent from this correlation study, the non-availability of data in certain categories prevents the ANN from learning the patterns in the dataset that are specific to that situation leading to negative yard gained or when the yards gained is larger than 6.

This brings into focus the core limitation of inductive learning paradigm and also makes a case for long term data management plan for business seeking to utilize data analytics into strategic decision making process. The model will perform well in those situations which it has encountered in the past. Similarly, the model will continue to be refined with time as more situations are encountered and additional data is added to the database for continual model refinement.

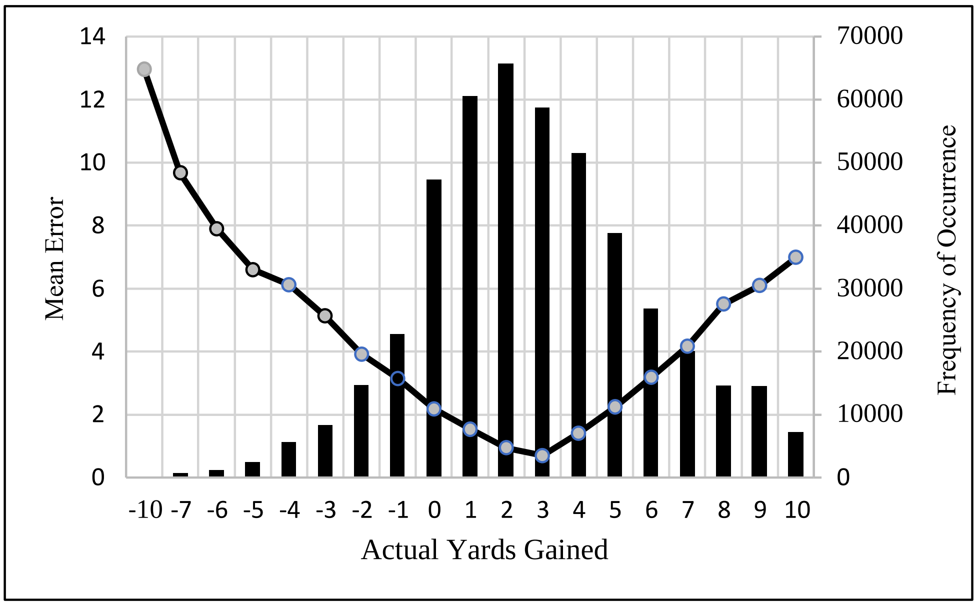

Figure 8: Average Error in Yardage as a Function of Frequency of Training Data.

6. Model Adaptation for Business Strategy

The techniques described in this paper have a wider applications. Specifically, the model discussed in this paper focuses on a continual utilization of a model for strategy determination while the play is going on. Thus, the coach for the game can assess the situation on the field and make a determination using the ANN model to dynamically make decisions to maximize yards gained and thus enhance the overall expectations of winning the game.

A similar strategy is applicable to the development of a business strategy that uses machine learning to make business decisions on a continual basis to improve the overall success of enhancing profits and returns to the stockholders. Machine learning approaches have been used in business settings for specific situations like predicting credit-worthiness, likelihood of bankruptcy, and rating bonds. While these are successful use-cases of machine learning paradigm, the approach discussed in this paper proposes a paradigm for using machine learning for in developing a short term business strategy, like intra-game yard gains, for reaching an overarching goal, like winning the game or running a profitable business.

Similar to using a model for determining the optimal play to make in a given setting, a business can similarly model its strategic decision making using a set of parameters that may affect this managerial decision making. As the specifics for strategic determinations will be unique to each industry, one of the first tasks for a business will be to analyze the features to use for the purposes of training a learning model and from a repository of past decisions and a associating a numerical rating with each of these decisions.

Upon completion of such a repository of past decisions, the business can present a new business situation to a computational model and seek guidance from the computational model to make a decision that maximizes the likelihood of a successful outcome.

7. Conclusions

This paper provided an overview of a process for intelligently modeling a situation, such as a specific conformation in the game of American football, and predicting the outcome of the situation by using knowledge of the outcomes from similar situations encountered in the past. The process of formulating the problem by identifying the factors that impact the situation, developing a learning model, and testing the accuracy of that model were discussed. While the specific situation in this study relates to predicting how many yards are expected to be gained in a given configuration, the paradigm is extensible to situations encountered in a business such as predicting sales, production output, or market share of a new product. While the results are promising, future research is needed to answer questions of predicting with uncertainty, such as accurately predicting outcomes in situations when there is limited or inadequate knowledge to learn from.

8. References

1. *American Football.* (n.d.). https://en.wikipedia.org/wiki/American_football. (Accessed: 2019-12-20)
2. Asuncion, A., & Newman, D. (n.d.). UCI *Machine Learning Repository.* https://archive.ics.uci.edu/ml/index.php.
3. Holmes, G., Donkin, A., & Witten, I. H. (1994). Weka: A Machine Learning Workbench.
4. Kaefer, F., Heilman, C. M., & Ramenofsky, S. D. (2005). A neural network application to consumer classification to improve the timing of direct marketing activities. *Computers & Operations Research, 32*(10), 2595–2615.
5. Kröse, B., Krose, B., van der Smagt, P., & Smagt, P. (1993). *An introduction to neural networks.*
6. Li, J., Cheng, J.-h., Shi, J.-y., & Huang, F. (2012). Brief introduction of back propagation (bp) neural network algorithm and its improvement. *In Advances in computer science and information engineering (pp. 553–558).* Springer.
7. *The National Football League.* (n.d.). https://www.nfl.com/. (Accessed: 2019-12-22)
8. *NFL Big Data Bowl.* (n.d.). Retrieved from https://www.kaggle.com/c/nfl-big-data-bowl-2020/data
9. *The NFL Rule Book.* (n.d.). https://operations.nfl.com/the-rules/2019-nfl-rulebook/. (Accessed: 2019-12-20)
10. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., others (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research,* 12(Oct), 2825–2830.
11. Pendharkar, P. C. (2005). A threshold-varying artificial neural network approach for classification and its application to bankruptcy prediction problem. *Computers & Operations Research,* 32(10), 2561–2582.

12. Surkan, A. J., & Singleton, J. C. (1990). Neural networks for bond rating improved by multiple hidden layers. In 1990 *IJCNN International Joint Conference on Neural Networks* (pp. 157–162).
13. Tan, P.-N., Steinbach, M., & Kumar, V. (2006). *Data mining introduction.* Bei Jing: The people post and Telecommunications Press.