Can A Computer Model Beat the Experts in Predicting NBA Playoff Contenders?

Syed Alle Mustafa - A20519664

Andrew Petravicius A20442138

Problem Statement

With the rise of online sports betting, more and more people are getting into predicting playoff contenders of various sports. As a result, more and more money is on the line. For most people, they decide the playoff contenders not by using some computer model, but by their own human wisdom, intuition, and biases. This is true for even many analysts who make a living by being an expert on the game and stake their reputation on their ability to predict season outcomes. Their method of predicting the playoff teams is the same as the average better, albeit with more wisdom behind it.

The human expert is prone to traps such as confirmation bias, reporting bias, and team favoritism. Which is to say, human experts rarely take the cold, hard numbers as their main consideration. For example, you'll hear many people making bets based on the fact that the team is "due for one". This, obviously, is an example of the gambler's fallacy, but given that the outcomes of sports games are determined by humans and not completely up to chance as common gambling games such as roulette are, there may be more truth to superstition deciding outcomes than we give it credit for.

The players of the sport aren't machines and their performance can and is affected by their mindset. So it may be the case that a team that is "due for one" might perform better than a team that is, numerically speaking, the clear favorite. The only way to know for sure is to compare the computer's performance with human performance.

The goal of this project is to build a model that can predict which teams will make it into the playoffs. The secondary goal is being able to outperform the experts' accuracy rates.

<u>Proposed Solution</u>

The implementation of this project was inspired by Rory Bunker and Fadi Thabtah's paper, Bunker, *A Machine Learning Framework for Sport Result Prediction*, where they tested and reviewed sport prediction models on soccer[1], but they explain that the methodology is general to all sports. In the paper, they emphasize the importance of data preprocessing, as that is what sets apart a machine learning model's prediction from a sport expert's prediction. For predicting the outcome of playoffs we used three models, The first one was a deep learning model with reLU and sigmoid activations and binary cross entropy for calculating the loss. The second one was using the Long short term memory (LSTM) Model, where the future predictions were made based on team performances this season and using performances of previous seasons. The last model to be used was a transformer from the paper Attention is all you need, and is used in time series prediction instead of natural language processing. All three of these models were trained and the accuracy and loss of these models were compared with the experts prediction.

<u>Implementation Details</u>

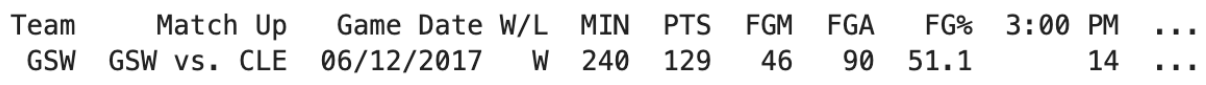
The first step was the collection and preprocessing of data. Luckily, the NBA website has the boxscores for every single game dating all the way back to the 1946-47 season[2]. For this problem, however, not all of that data is necessary. The game has changed so much over the years that game outcomes in the 1960s are likely not relevant to predicting the outcomes of modern day seasons. As a result, we must determine what seasons are relevant to predicting future seasons.

[1] Bunker, Rory P., and Fadi Thabtah. "A Machine Learning Framework for Sport Result Prediction." *Applied Computing and Informatics*, No Longer Published by Elsevier, 19 Sept. 2017, https://www.sciencedirect.com/science/article/pii/S2210832717301485.
[2] "National Basket Association Statistics and Boxscores." *Official NBA Stats | Stats | NBA.com*, https://www.nba.com/stats.

Finding relevant seasons was trickier than was expected since the sport has undergone many changes, especially in recent years. So, using data up until the last major rule change would result in only a few seasons of data, which would obviously not lead to a good model, even with bootstrapping.

As a result, some research was done on the rules of basketball and the culture of the NBA in order to determine what seasons would be relevant to the modern day[3]. In the end, in order to both have a good amount of data and to keep the data relevant, boxscores since the 2000-20001 season were collected.

With the data collected and formatted into comma separated value files, the preprocessing could now begin. The data, in its raw format, lists every game and the boxscore f0r it. For example, one row of data might look something like Figure 1.



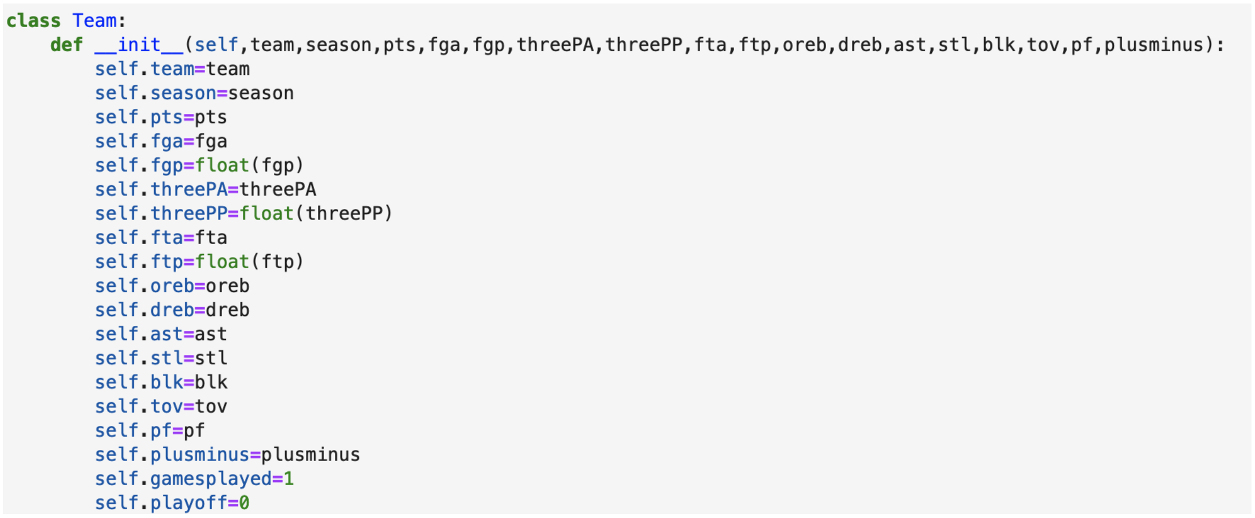
```
Team     Match Up   Game Date W/L  MIN  PTS  FGM  FGA   FG%  3:00 PM  ...
 GSW  GSW vs. CLE  06/12/2017   W  240  129   46   90  51.1       14  ...
```

Figure 1: Data Row Example

The data follows the format of listing the team, the matchup, the date, if the team won or lost, how long the game went on for, the points the team scored, the field goals made, and so on. We are, however, not interested in the individual games, but the season performance of each team. For each team, an object was made to store the information about every game that team

[3] Golliver, Ben. “The NBA at 75: From Mikan to Lebron, How the Game Has Changed.” *The Washington Post*, WP Company, 2 June 2021, https://www.washingtonpost.com/sports/interactive/2021/nba-75-year-anniversary/.

played. For example, the 2007-2008 Bulls would be a different team from the 2008-2009 Bulls, each of which having their own unique seasonal stats.



```
class Team:
    def __init__(self,team,season,pts,fga,fgp,threePA,threePP,fta,ftp,oreb,dreb,ast,stl,blk,tov,pf,plusminus):
        self.team=team
        self.season=season
        self.pts=pts
        self.fga=fga
        self.fgp=float(fgp)
        self.threePA=threePA
        self.threePP=float(threePP)
        self.fta=fta
        self.ftp=float(ftp)
        self.oreb=oreb
        self.dreb=dreb
        self.ast=ast
        self.stl=stl
        self.blk=blk
        self.tov=tov
        self.pf=pf
        self.plusminus=plusminus
        self.gamesplayed=1
        self.playoff=0
```

Figure 2: Team Object

As you can see in Figure 1, not all of the stats were included, such as 3:00 PM or REB. 3:00 PM, which stands for three points made, was not included since it is redundant as three points attempted and three point percentage are included in the data. For this reason, three points made, along with field goals made and free throws made, were not included.

REB, which stands for rebounds, was not included for similar reasons as OREB, offensive rebounds, and DREB, defensive rebounds, were included and provided more information as they take into account the position the team was in at the time of the rebound and can help indicate whether a good offense or defense is more important to a team's playoff chances.

Another column that was dropped was the number of wins per team, as the purpose of this project is to determine if a team will win enough games to reach the playoffs using trackable stats. Given a season that is only a few weeks in, the model should be able to predict which teams will reach the playoffs, which cannot be done accurately just by looking at the win loss column of a team's first few games, but which, we posit, can be done, by looking at their game performance.

In order to build a team object, the files for each regular season were inputted through a function called *interpret*, which would read each row of each file, taking the team name and the season and checking a dictionary to see if that key already existed. If it did, the function would update the stats of that team based on the game the function is currently looking at. If the key didn't exist, it would create the key in the dictionary and initiate the team object with the game's stats.

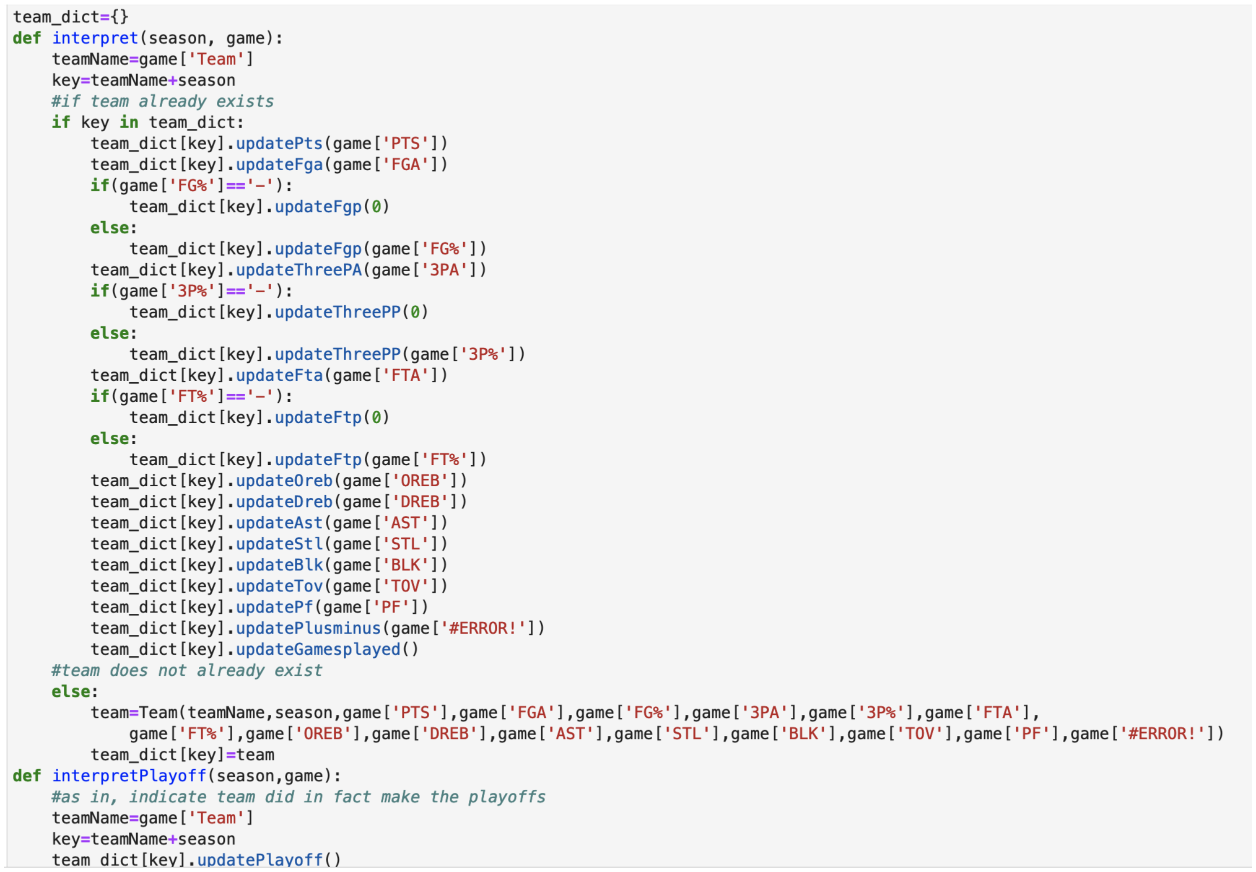
Another function, *interpretPlayoff*, was defined to comb through the playoff stats to determine which teams made it and which teams did not.

In this process, however, a few errors were encountered. One of which being that for some games, one of the stats would be blank, leading to the index having a null value, which had to be dealt with. These null values arose in the percentage features: field goal percentage, three point percentage, and free throw percentage. If, say, a team didn't have any free throw attempts in the game, the value of the free throw percentage would be null, which the function was not able to initially interpret.

Dropping rows with null values was not ideal as teams that tended to have less free throw attempts would have less data attached to them and thus be more biased, since the results for each season were averaged.

An example of this is the 2004-2005 New York Knicks, which went multiple games without having any free throws, and thus had multiple rows with a null value in that column.

Dropping those rows would have deleted multiple data entries for that team just for one missing value. Instead, for each of those null values, they were replaced with a 0 to reflect the absence of penalties that the team was able to incur; a stat which was presently missing from the dataset.



```python
team_dict={}
def interpret(season, game):
    teamName=game['Team']
    key=teamName+season
    #if team already exists
    if key in team_dict:
        team_dict[key].updatePts(game['PTS'])
        team_dict[key].updateFga(game['FGA'])
        if(game['FG%']=='-'):
            team_dict[key].updateFgp(0)
        else:
            team_dict[key].updateFgp(game['FG%'])
        team_dict[key].updateThreePA(game['3PA'])
        if(game['3P%']=='-'):
            team_dict[key].updateThreePP(0)
        else:
            team_dict[key].updateThreePP(game['3P%'])
        team_dict[key].updateFta(game['FTA'])
        if(game['FT%']=='-'):
            team_dict[key].updateFtp(0)
        else:
            team_dict[key].updateFtp(game['FT%'])
        team_dict[key].updateOreb(game['OREB'])
        team_dict[key].updateDreb(game['DREB'])
        team_dict[key].updateAst(game['AST'])
        team_dict[key].updateStl(game['STL'])
        team_dict[key].updateBlk(game['BLK'])
        team_dict[key].updateTov(game['TOV'])
        team_dict[key].updatePf(game['PF'])
        team_dict[key].updatePlusminus(game['#ERROR!'])
        team_dict[key].updateGamesplayed()
    #team does not already exist
    else:
        team=Team(teamName,season,game['PTS'],game['FGA'],game['FG%'],game['3PA'],game['3P%'],game['FTA'],
            game['FT%'],game['OREB'],game['DREB'],game['AST'],game['STL'],game['BLK'],game['TOV'],game['PF'],game['#ERROR!'])
        team_dict[key]=team
def interpretPlayoff(season,game):
    #as in, indicate team did in fact make the playoffs
    teamName=game['Team']
    key=teamName+season
    team_dict[key].updatePlayoff()
```

Figure 3: Interpret and IntrepretPlayoff Functions

The final step in the preprocessing of the data was normalizing it. Since each feature has radically different ranges of values, such as points per game being much higher than personal fouls per game, normalizing the data would prevent the model from weighing the features based on their scale and instead weight the features on their importance. Using sklearn’s scaler function, the data was normalized to be within the range [0,1].

After all of that data was compiled and averaged to find the stats per game for each team, the dataframe, which consisted of each team's seasonal stats, was split into its response and predictor factors. The response, of course, being if the team made the playoffs or not, which is found in the column labeled 'Playoff', while the rest of the columns were used as the predictors, and then later split into test and train datasets. To pass the data to models, firstly, every column was scaled using min max scalar, so that they range between 0 to 1. Then the data is splitted into training and validation sets. Since, the data is very less, only 150 rows were selected for validating the learning of the model. The first model consisted of 3 relu layers, with 64, 32 and 16 neuron respectively, and every layer is followed by a drop of 20% and batch normalization to avoid overfitting, the output layer consists of sigmoid activation and the loss function used is binary cross entropy and adam as an optimizer, the learning rate was 0.001 which is default for adam optimizer. The second model is an RNN model, for training this day was prepared using a time series sequence generator, that iterates over the input and output series. The length of each sample used to train the network is set to 40, where the sampling rate is set to be 10 and batch size of 32 is used. For the neural network layers, one layer of Long short term memory (LSTM) is used with 64 neurons, which is followed by a relu (32 neurons) and sigmoid activation layers respectively.The optimizer is adam and the loss function is mean squared error. This network is trained for 300 epochs. Lastly, a transformer with structure given in report "Attention is all you need" is used. The encoder consists of a Layer normalization, multi head attention, drop out and two 1 dimensional convolution layers. For of these blocks are used for encoding, then a layer of global average pooling is used, after which multiple layers of multilayer perceptron is used which relu activation layer and in the end a softmax activation is used as the output layer. The optimizer used for all three models is Adam with different loss functions.

Results and Discussion

Before looking at how our model performs, we first want to investigate the overall accuracy of the experts. Through looking at previous predictions of the playoff bracket created by CBS Sports, ESPN, and Fox Sports experts halfway through the season of past seasons, we see that, on average, the experts are about 70% accurate at predicting the playoff contenders[4,5,6]. This value was found by scraping past articles of playoff predictions at the halfway point through the season and seeing what percentage of the predictions for the playoff contenders turned out to be accurate.

For example, in the 2015-16 season, ESPN NBA analyst, Fran Fraschilla, successfully predicted 11/16 (~68%) of the playoff contenders correctly. Averaging the results from each of these three sources over the past ten years was how this number was found.

For each of the three models trained, the results are shown below, for both the training and validation sets.

[4] "NBA Playoff Predictions: Cowherd's NBA Finals Pick." *FOX Sports*, FOX Sports, 13 Apr. 2022, https://www.foxsports.com/stories/nba/nba-playoff-predictions-whos-going-to-the-finals.

[5] Windhorst, Brian. "The Hoop Collective: Will Victor Wembanyama's Rise Spell the End of Team USA's Dynasty?" *ESPN*, ESPN Internet Ventures, 15 Nov. 2022, https://www.espn.com/nba/story/_/id/35026261/the-hoop-collective-victor-wembanyama-rise-spell-end-team-usa-dynasty.

[6] Colin Ward-Henninger Apr 23. "NBA Playoffs 2022: Picks, Predictions for Every Round as Experts Take Bucks, Suns, Warriors, Celtics to Go Far." *CBSSports.com*, 23 Apr. 2022, https://www.cbssports.com/nba/news/nba-playoffs-2022-picks-predictions-for-every-round-as-experts-take-bucks-suns-warriors-celtics-to-go-far/.

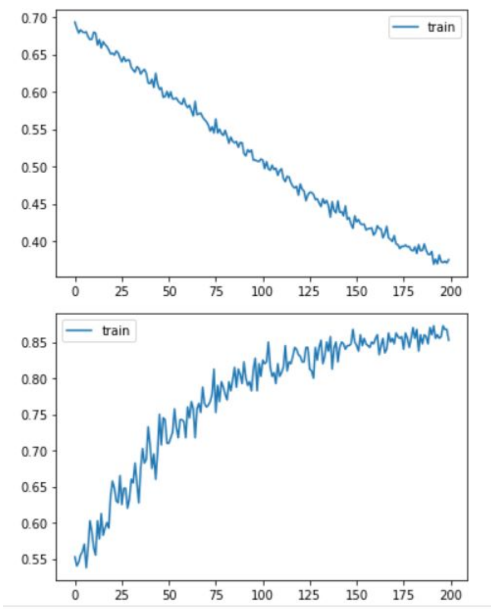


Figure 4: Transformer Training Loss and Accuracy

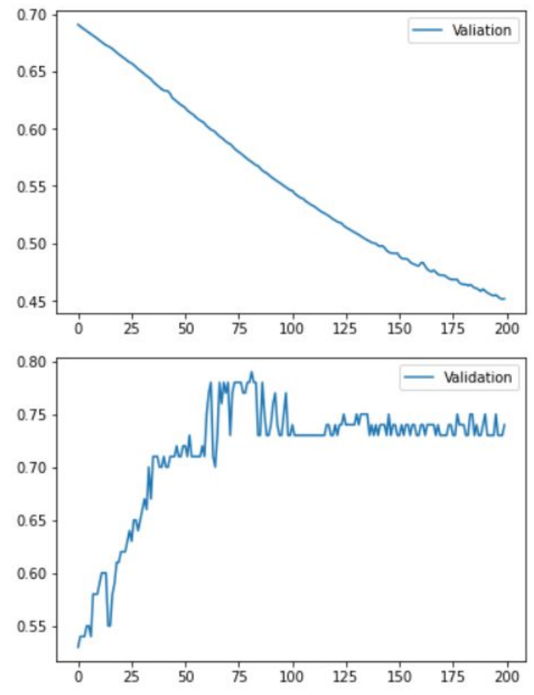


Figure 5: Transformer Validation Loss and Accuracy

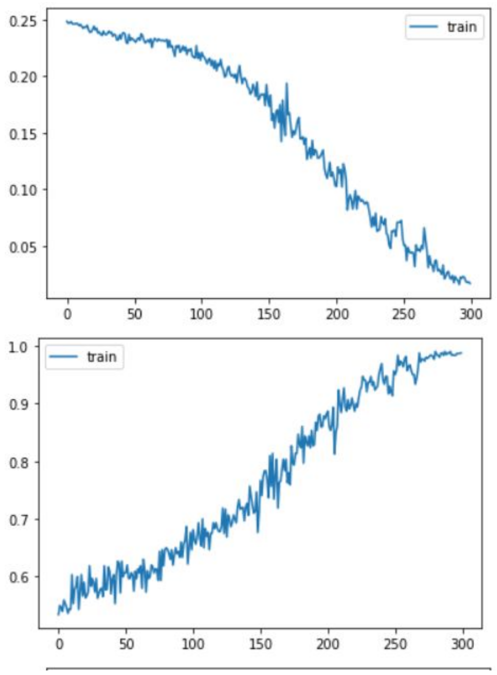


Figure 6: LSTM Training Loss and Accuracy

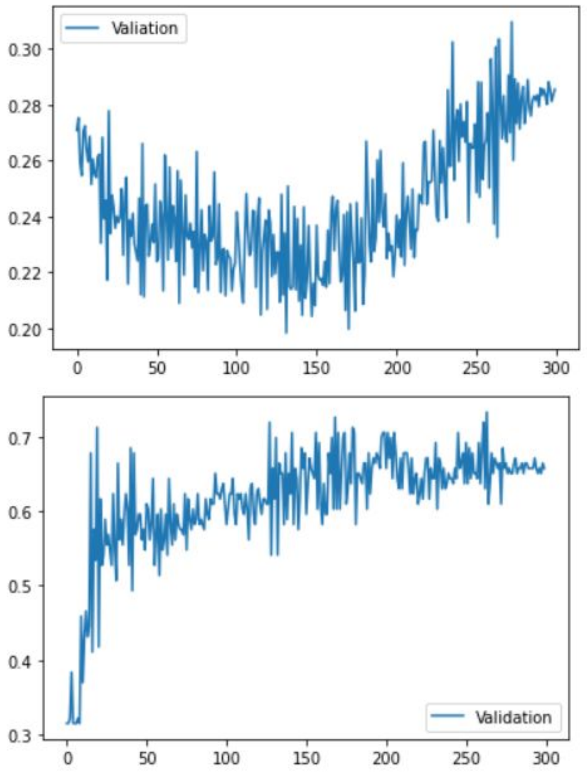


Figure 7: LSTM Validation Loss and Accuracy

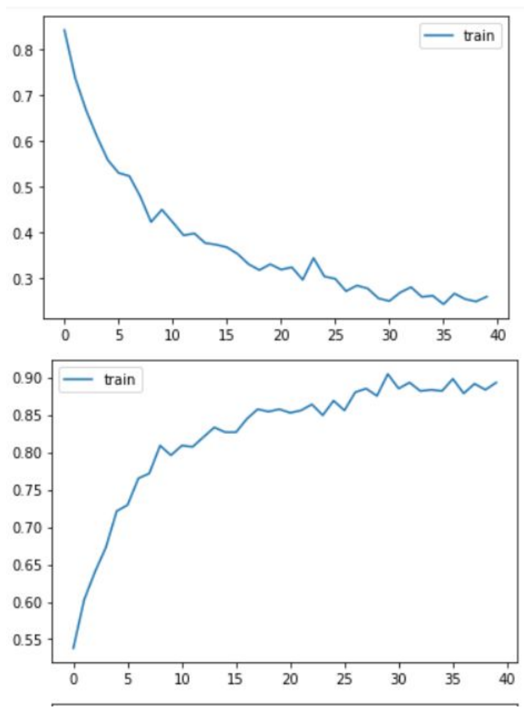


Figure 8: Deep Learning Training Loss and Accuracy

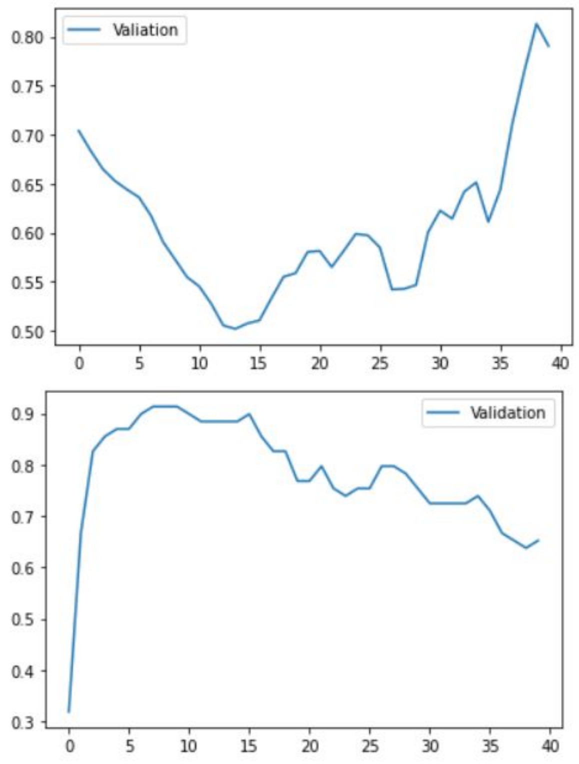


Figure 9: Deep Learning Validation Loss and Accuracy

As the figures show, the transformer and deep learning models accomplished our goal of creating models that outperformed the experts. Our LSTM model, however, failed to reach that criteria, but it did not perform much worse than the experts.

The deep learning model, however, is shown to overfit the data as the epochs progress. Changing the amount of epochs resulted in varied results, which shows a flaw in using models for sports predictions. Since these kinds of models usually operate on limited datasets, it's hard to get consistent results with such frameworks. Bunker and Thabtah also concluded that more developments in modeling are needed to ensure reliably high performance, as the outcome of sports games are more complicated than might be expected.

In conclusion, the results do show the potential of computer models in the field of sport predictions, but more time and development is needed before models will consistently outperform human analysts. The implications of machines outperforming humans in sports bidding are currently unknown, but surely will be revealed in the coming years as models get better and better.

References

1. Bunker, Rory P., and Fadi Thabtah. "A Machine Learning Framework for Sport Result Prediction." *Applied Computing and Informatics*, No Longer Published by Elsevier, 19 Sept. 2017, https://www.sciencedirect.com/science/article/pii/S2210832717301485.
2. "National Basket Association Statistics and Boxscores." *Official NBA Stats | Stats | NBA.com*, https://www.nba.com/stats.
3. Golliver, Ben. "The NBA at 75: From Mikan to Lebron, How the Game Has Changed." *The Washington Post*, WP Company, 2 June 2021, https://www.washingtonpost.com/sports/interactive/2021/nba-75-year-anniversary/.

4. “NBA Playoff Predictions: Cowherd's NBA Finals Pick.” *FOX Sports*, FOX Sports, 13 Apr. 2022, https://www.foxsports.com/stories/nba/nba-playoff-predictions-whos-going-to-the-finals.
5. Windhorst, Brian. “The Hoop Collective: Will Victor Wembanyama's Rise Spell the End of Team USA's Dynasty?” *ESPN*, ESPN Internet Ventures, 15 Nov. 2022, https://www.espn.com/nba/story/_/id/35026261/the-hoop-collective-victor-wembanyama-rise-spell-end-team-usa-dynasty.
6. Colin Ward-Henninger Apr 23. “NBA Playoffs 2022: Picks, Predictions for Every Round as Experts Take Bucks, Suns, Warriors, Celtics to Go Far.” *CBSSports.com*, 23 Apr. 2022, https://www.cbssports.com/nba/news/nba-playoffs-2022-picks-predictions-for-every-round-as-experts-take-bucks-suns-warriors-celtics-to-go-far/.