Using Deep Neural Network to predict the score of upcoming daily NBA Games

Project Report Submitted By

Chandramani
Xiaoyan Feng

Principle of Data Science

2022-23

National Basketball Association (NBA) is the men's professional basketball league in North America. A tremendous amount of research is going on in predicting the scores of upcoming NBA matches as the NBA has managed to record very detailed statistics of each match. This project is a comparative study of various models for prediction of scores a basketball game based on the home and away statistics of each team . We have used 2 types of datasets: All games from 2017 - 2022 and each game's stats extracted by game id which is used for data analysis and all the statistics of the game from the year 1979-2022 played by the player id. The later data is used to train the model . This project run to predict the most likely to predict the scores of NBA season based on the most recent season statistics available. (The balldontlie API updates approximately every 10 minutes). Along with the standard features like win average, points difference, etc., a special emphasis has been given to attributes like playing as a home team or as a visitor team and statistics for the last matches played in last 1 year to predict the scores of home team and visitor team.

1 Introduction

National Basketball Association (NBA) is the men's professional basketball league in North America. Due to the game's global appeal and value, numerous studies are being conducted to forecast game outcomes and compare players, among other things. Nearly all of the factors affecting a basketball game are covered in some of the best data the sports business has to offer according to the NBA. Despite the fact that the data for the games are big and precise, it is still a very difficult task.

1.1 Objective

The objective of this project is to collect raw data from the API provided by `https://www.balldontlie.io/#introduction` using feature engineer and form relevant features which may play an important role in predicting the scores of the basketball match. Then, these features are used to train DNN neural network model with the aim of predicting score of home team and away team. Then we develop a front end website where we collect data of the upcoming NBA matches and using the trained DNN model, we predict the scores of each team. The pre-trained model takes the input of the last 1 year games played by both the teams. For Home team we are getting the stats of the last 1 year where home team played as home and for away team we are getting the stats of 1 year played as away.

2 Related Works

The application of machine learning algorithms to forecast NBA games has been extensively studied. The research technique suggested by Renato Amorim Torres served as a major source of inspiration for our work. Our research aimed to delve deeper into the attributes without limiting ourselves to only the more obvious ones, like a team's victory%. However, Renato's effort served as the inspiration for the characteristics relating to the last 8 matches. Since they tried all the numbers from 1 to 10, and 8 produced the greatest results for their model, we decided to use 20 and get the result.
Another Linear Regression algorithm to predict the winner was used in a paper by M.Beckler,

H.Wang, 33 and M.Papamichael NBA Oracle (Beckler, Wang, Papamichael,2008). Using previous games, they achieved a result of 73% accuracy, which was the best result found.

3 Dataset

All the data is gathered from the balldontlie API which provides access to details about games, teams, and players and is updated every 10 minutes during the season (according to their website). It uses multiple API calls to retrieve the full list of games for each season that the user selects.
The balldontlie API provides data about NBA basketball, including players, teams, games, and statistics. Developers can use the API to retrieve player-specific season averages. The balldontlie API offers JSON formatted responses.

3.1 Fetching Data From API

After a detailed analysis of all the data available on the API provided by 'balldontlie' er decided to use all the games data played in the last half decade (2017-2022) . We extracted all the stats data available from the API. The API contains total of 11931 pages. This API contains all the stats for each team from the year 1979. We requested 100 pages per data which is the maximum allowed limit from the API. Both the game and stats data are then saved in the csv files for the data analysis and machine learning model.
The games data has 50000 rows and 22 columns. The game data contains fetures like home team score, away team score, etc.. The stats raw data contains 1192900 rows and 46 columns

3.2 Data Cleaning

After careful deliberation we decided to use the stats data to use in the machine learning. The stats data contains some unnecessary features which are not basketball scores relevant. So we took following steps to clean the stats data.
We drop the unneeded columns like game period, game time, player name, player height, conference etc.. Then we drop the rows using the time data which are not played completely.The time columns was also cleaned such as replacing '.' with ':' and replacing '0' at end with '00:00'. Then we added the seconds data to the minute column to make it one.

3.3 Exploratory Data Analysis on Games And Players

Since we are using the stats data for machine learning model we used the game data to do some data analysis of how the NBA games has changed over the years. We found following results on doing the data analysis of the games.

1. **Percentage of games won by home team over the years**

   As per the data from the years it is evident that more than 55% of the game is won by the home team over the years, with highest in 1987 where home team won 68% of the game. Over the years almost 60% of the games are won by the home team.

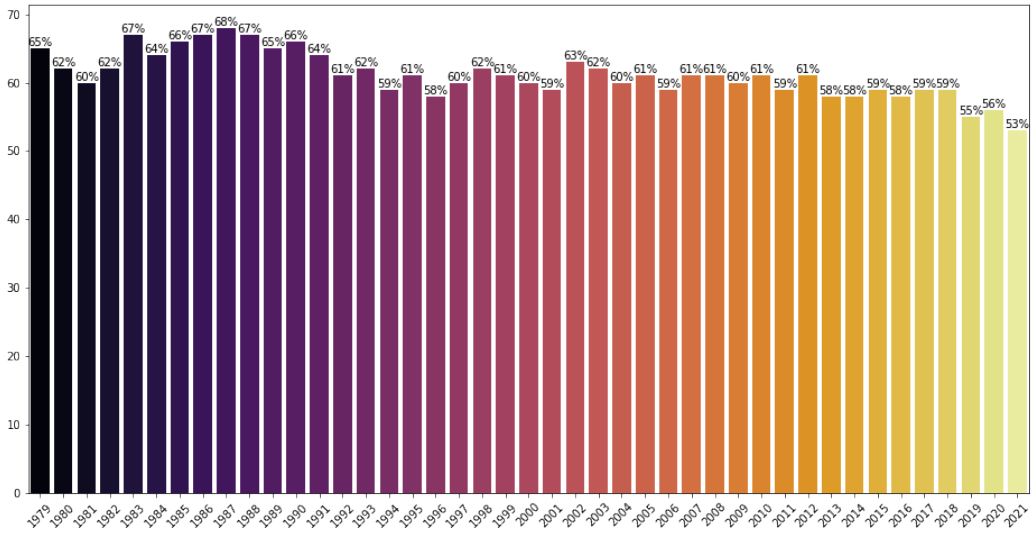


Figure 1: *Home Win % over the Years*

2. **Points scored by Home Teams as compared to Away Teams over the years**

When doing the analysis we found out that the points score by the home team varies in the range from 80-130 whereas the points scored by the away teams varies in the range of 90-120 mostly. Please refer to the histogram plotted.

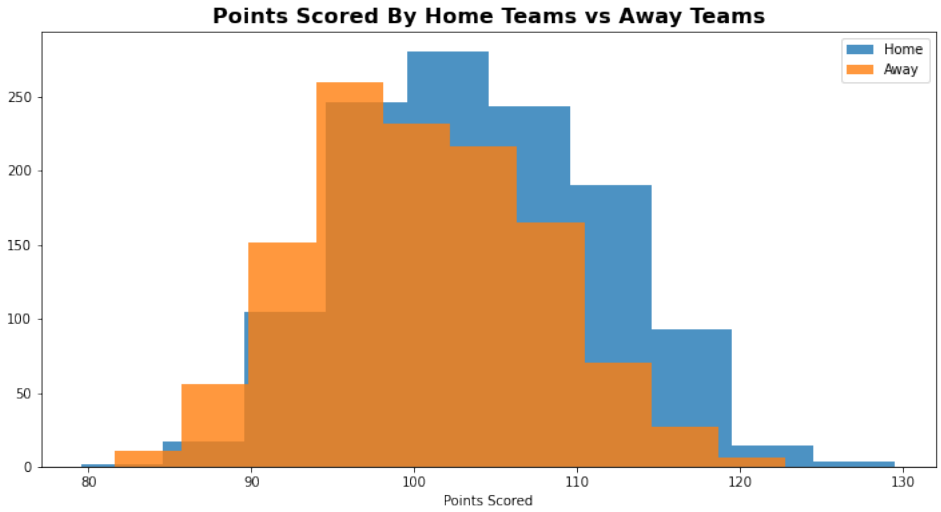


Figure 2: *Points Scored By Home Teams vs Away Teams*

3. **Average distribution of weight and height of NBA Players**

As per the data analysis the average mean of the height of the NBA players is 200cm as compare to the mean of the USA male is 170cm. The highest height of a NBA player is recorded to be 230cm.

The mean weight of the NBA players is 100kg wit most of the weight ranging between 100-120kg and 80-90 kg. The mean weight of the average male in USA is said to be 85kg.

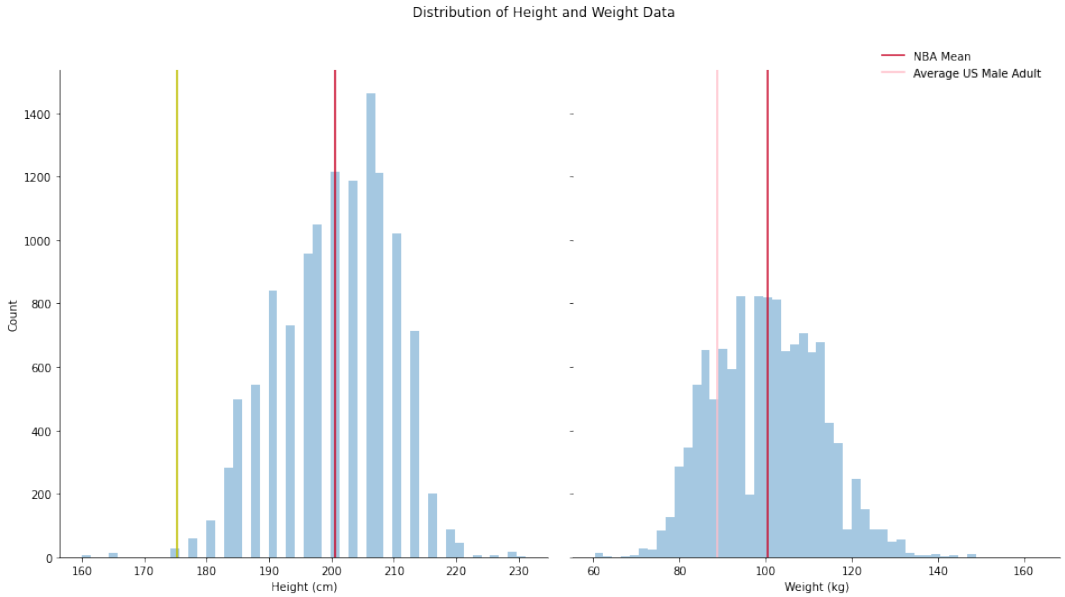


Figure 3: *Average distribution of weight and height of NBA Players*

4 Workflow

In this section we are going to demonstrate the workflow of the project . This includes the pre-processing of the stats data which to be used to train the Deep Neural Network. Then we will train and test the DNN model and calculate the errors. After that we will demonstrate how we create the back-end for the web site to predict the scores of the home team and the away team.

4.1 Data Pre-processing

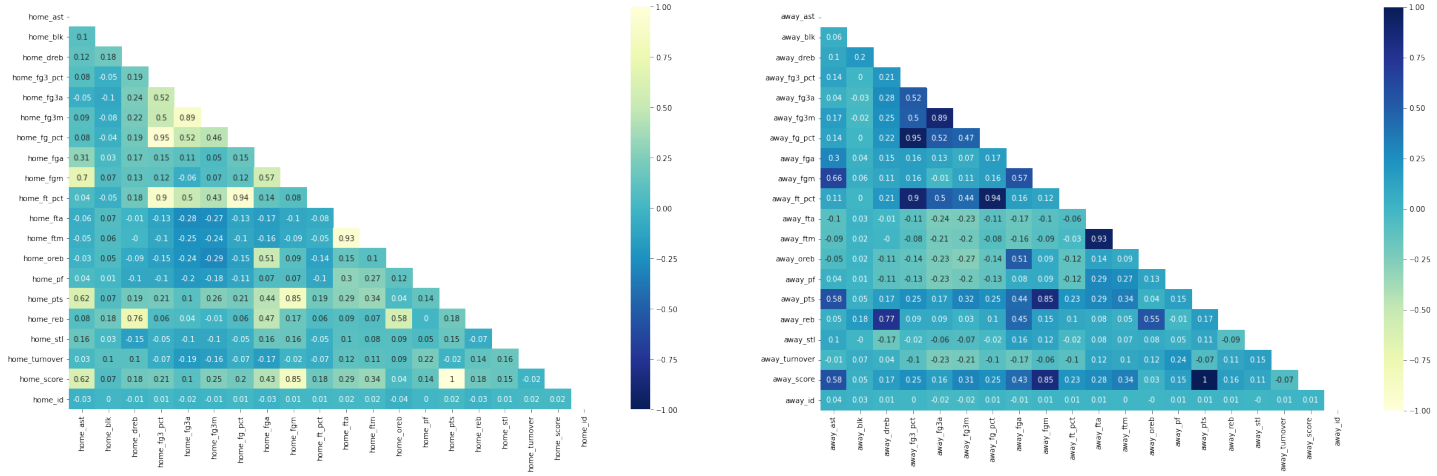
The clean stats has 968619 rows of data which contains the basketball stats according to the players and different team. But we are interested in the team stats only for our machine learning model, more specifically the home team stats and the away team stats. We found out that in 968619 there are 47655 unique game stats,and we are going to use that for our model. We split the data into data frames so that grouping data by game ID doesn't group players by opposing team.

We created an aggregated map where we take sum of the stats of the player by each team like sum of assists, sum of dribbles, mean of 3 pointers percentage, sum of rebounds, sum of 3 pointers etc.. After creating the aggregated map we created two data frames by grouping them via game ID and aggregating them with the created aggregated map.

We then dropped player and team id from both the data sets (Home game and the away game) as it doesn't make sense anymore for our model.

We then take the columns that refer to stats when calculating the rolling average like assists, dribble, points, filed points, turnover etc. in each of the data sets. We also used the home team ID and away team ID as features so that we can learn how match up between the teams affects the scores. The co relation matrix is shown in Figure 4

We added home to the column name of the home_ data and away_to the column names of



(a) *Co relation Matrix of Home* **(b)** *Co relation Matrix of Away*

Figure 4: *Co relation Matrix*

the away data to differentiate when we combine them to a single data. We then aggregated the home statistics and away statistics into the single data.The final data has 47650 rows with 40 features including home team score and away team score.

4.2 Feature Scaling

We used the sklearn library to split the data set into train , test and validation set where 80% of the whole data is split into training set and testing set and then 20% of the training set is used for the validation set.
We popped the home team score and away team score from the training, test and validation data. These two are going to be used in form of two labels for our DNN model.

The data in the remaining datasets after popping out the labels (Home Score and Away Score) has the features that varies very much in terms of the value of the magnitude. So while training the model the Neural Network tends to weigh the features with greater values, higher and consider smaller values as the lower values, regardless of the unit of the values. So we are going to normalize the remaining data.
For normalization of the remaining data we subtracted the mean of the columns from each value of that column and then divide the values with the standard deviation of the columns. The formula used here is like this :

$$x - col[mean]/col[std]$$

where x is the value, mean and std is derived using the describe() function of the python and then transposing it.
Our test data contains 30496 rows, test data contains 9530 rows and validation data contains 7624 rows and all of them have 38 columns/features.

Now all our data is ready for the training and evaluating our model on training , validation and test data respectively.

4.3 Building the Deep Neural Network Model

We used the keras library to build our neural network model. We used the sequential model approach where each layer has exactly one input tensor and one output tensor. Since our model needs to be trained for two different output (Home score and away score) we have two output layers for our model.
The model has dense layers which means each neuron of the layers receives inputs from all of the previous layers. It is most commonly used layer in the model. Our model has one input layer which takes input of equal to the length of the training data and makes 256 sets of outputs. For each of the output there are 3 hidden layers with the same number of units (256). RelU activation function has been used for this model in each layer with a keras initializer to regularize the weights of each layers. The model summary is shown in the below table :

Layer(type)	Output Shape	No of Parameters Trained Each layer	Connected to
Input Layer	[(None, 38)]	0	Connected to both the layers
dense	[(None, 256)]	9984	input_1[0][0]
dense_3	[(None, 256)]	9984	input_1[0][0]
dense_1	[(None, 256)]	65792	dense[0][0]
dense_4	[(None, 256)]	65792	dense_3[0][0]
dense_2	[(None, 256)]	65792	dense_1[0][0]
dense_5	[(None, 256)]	65792	dense_4[0][0]
home_output	[(None, 1)]	257	dense_2[0][0]
away_output	[(None, 1)]	257	dense_5[0][0]

Table 1: *Design of our Neural Network*

As per our modal summary there are total no of 283650 params and all of them are trainable.

4.3.1 Hyperparameters of the DNN model

We choose Mean Squared Error for the loss function because it has the advantage of giving some sense of by how much predictions and true values differ (though this is not perfect, since it is not absolute error), and it has a relationship to the variance of the error term. Further, you do not make the value arbitrarily large by having many observations.
We also use the gradient clipping in our optimizer (Adam) to prevent the explosive gradient. The parameters to train the model are given in the below table:

Optimizer	Adam (from keras)
Loss Function	Mean Absolute Error and Mean Square Error
No of Epoches	500
Batch Size	48
Metrices	MAE,MSE for both home and away score

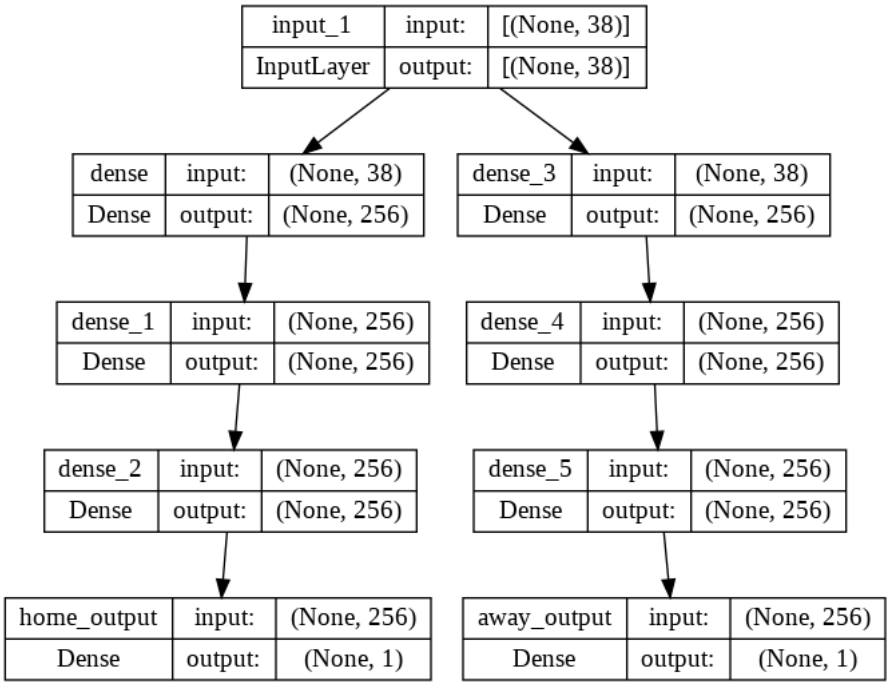


Figure 5: *Deep Neural Network Model*

4.4 Model Training

The model is trained for 500 epochs where weights of each layer is tuned using the training and validation loss. We also kept the history of the training and validation loss in a list. We saved the best model with the least validation loss at the end of each epoch so that we can use it. During our training we find the best model at epoch 474 where validation loss is 0.9145. This model is saved for testing and later used to predict the scores of the teams for our website using some data.

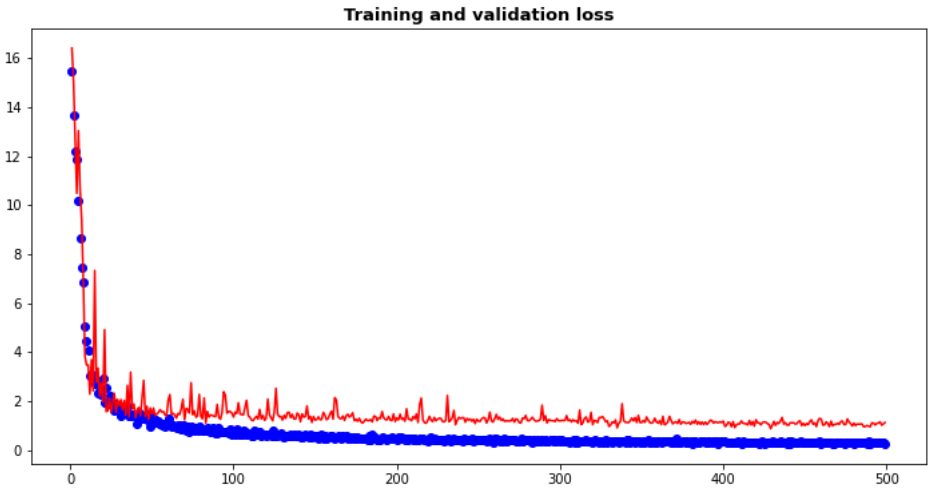


Figure 6: *Training and Validation Loss*

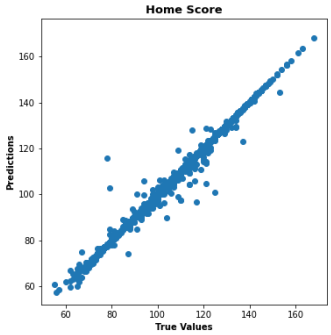
4.5 Result

After training we tested the model on the test set. The test data set contains 9630 rows of data and we evaluated on two metrics : Mean Square Error and Mean Absolute Error. Here are some results of our performance on the two metrics of the evaluation on the test data :2

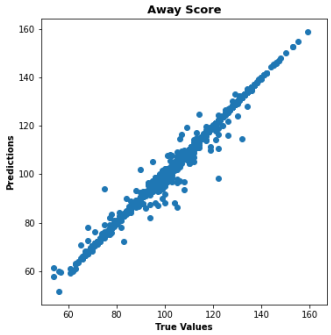
Home Score Output Loss	0.67
Away Score Output Loss	0.63
Home Score Mean Absolute Error	8.25
Home Score Mean Square Error	0.67
Away Score Mean Absolute Error	9.21
Away Score Mean Square Error	0.63

Table 2: *The loss table for the test data*

We also plotted the scatter plot for home and away loss. The accuracy of our model is quite good as per the square error loss and absolute loss error. We plot the scatter plot of our model with respect to the actual value and the predicted value. The plots are shown below.



(a) *Score plot of Home Score*



(b) *Scatter Plot of Away Score*

Figure 7: *Scatter Plot of Scores from Prediction and Actual Scores*

5 Web Page Development

After training the model we decided to use it to predict the scores of the next day upcoming game using the model and also predict the score of previous day games and compare the result. Our web-page has two sections :

1. In one section we are getting the data of all the yesterday game played and predicting the scores of Home team And Away team and then compare the result with the actual scores of both the teams.
2. In the second section we are getting the match which is going to be played today and predicting the cores of home and away team.

5.1 Back End Development

We are going to use our trained Deep Neural Network model to predict the scores of the teams who had played the game yesterday or will play the game tomorrow or later evening.
We will train the pre-trained model again using the last 20 games played by the Home Team as home and Away team as away. The last 20 games data of both the teams are again fetched from the 'balldontlie' API. We are getting the last 1 year data for both the teams and after sorting by date we are getting the 20 most recent games played.
After gaining the most recent 20 games data we again cleaned it, pre-processed it and again normalized the data so that it will be in the same shape as the data we used to train the model before. For these task we created lot of different helper functions that fetched, cleaned and aggregated the required data.

Retraining the Model

The data of the last 20 games played by both the teams is used to train the model again so that model can be adjusted to predict the score according to the most recent data to get more accuracy for the data.
The model was saved in the 'tf' format and was loaded gain using the keras library. We split the data set into train and test set. The train set has 16 games and test set has 4 sets of match. To predict the game score we are going to take average from the 4 predicted score of the model. The model is re-trained for 50 epochs on the 16 games of the train set, so that model weights can be adjusted according to the games played by both the team recently and how they performed in the most recent games.

After retraining the model , it is used to predict. The predictions has two lists containing four scores, one for home team and one for the away team. We took the average of these scores for final predictions. For the yesterday games we then compare the result of the predicted scores and the actual scores of both the team and for upcoming games we trained the model and predicted the scores for both the team.

5.2 Front End Development

Select front-end implementation form

To create a web page, we need to get a domain name, write HTML and create a CSS style sheet. After that, we should write JS code to add logic to the web page. All these steps can be simplified by using Anvil. Therefore, we think this is a good choice.

Design the web page

Firstly, we design our web page via Axure. Secondly, since Anvil has fix pattern for different components, we only need to drag components to get a web's outlines. After that, we write the set-up code for each component in website, for example, the text, image and table, etc..
At last, back to our back end, we add anvil's server driver in colab to provide some interfaces for the front-end calls. And the link of our completed web is `https://X6ICBRGYQ7LZXCEV.anvil.app/4UWUWRKDZDIRPLX4CKFX7HCJ`

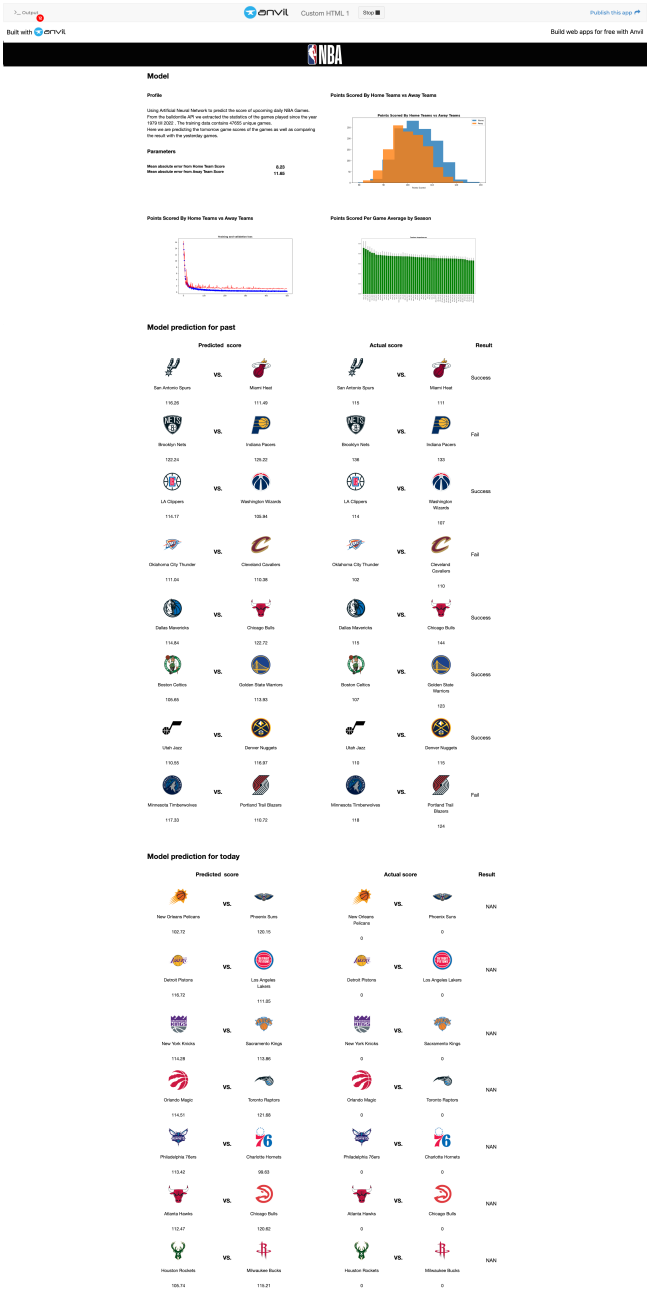


Figure 8: *The Webpage*

Optimization of the display

To get real-time data, we set a timer on the front-end so that it refreshes every 180 seconds. Thus, we can predict the match in progress and display our predicted result on our web page. And before this game starts, it will display 'NAN' in the 'Results' area of our web page. At the same time, '0' is displayed in the real score area.

6 New Findings and What We Learned

6.1 New Findings

1. **Height and weight of NBA Players Each Season**

   The average height and weight of a NBA player show a declining trend through different eras. This shows that players are getting more athletic and trying to add different skills to their resume, this leads to teams trying to utilize one player in multiple positions, making it the team tougher to defend.

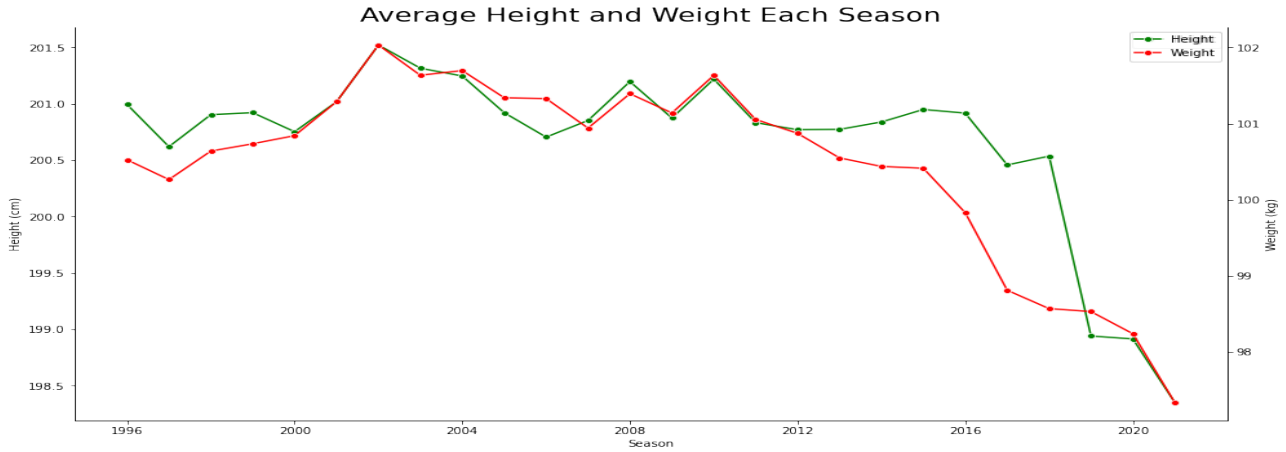


Figure 9: *Average Height and Weight Over the Years*

2. **Points Scored by Guard , Center and Forward Varying over the Season**
   This trend shows that the forwards are improving their scoring through the years, players like **Lebron James** and **Kevin Durant** are examples of high scoring forwards. Also, now teams are relying more on the guards to score, this could be due their ability to shoot 3 pointers or ability to get to the rim for the layup or the dunk.

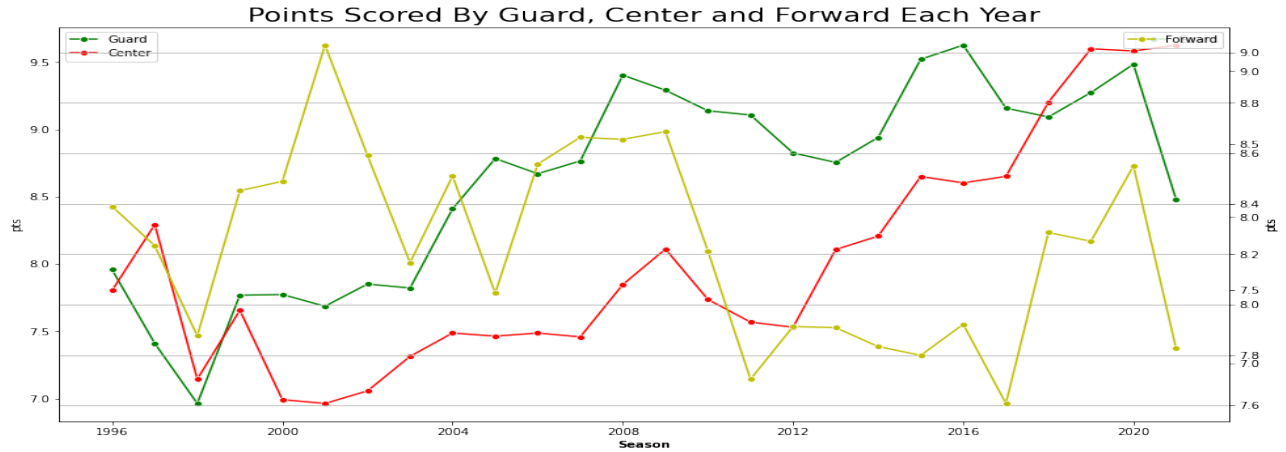


Figure 10: *Points Scored by Guard , Center and Forward Varying over the Season*

6.2 What We Learned

Apart from learning a lot about the NBA games and their we learned a lot about deep learning model and developing a web page. Some of our finding are mentioned here :

1. *Exploding Gradients*
   When we were re-training the our model with the two teams for the web site we ran into the problem of explosive gradients. We learned that the weight were assigned a very large value causing a numerical overflow and or model start predicting 'nan' for one of the home score or away score. To handle this we used 'gradient clipping ' in our Adam optimizer .
   The Gradient Clipping method changes the derivative of the error before propagating it backward through the network and using it to update the weights. This approach include re scaling the gradients given a chosen vector norm and clipping gradient values that exceed a preferred range.

2. *Predicting two outputs from a Neural Network*
   We were taught how to train a regression model for a single output. This is the first time both of us learned how to connect layers of the neural network to predict two outputs from a deeply connected neural network. We learned that the hidden layers of both the outputs run parallel and is connected by single input.

7 Conclusion

We learnt a lot from working on this project, and it will undoubtedly benefit our future academic and professional endeavors. The Mean Absolute Error for our home game score is 8 and for the away score is 9. So our model can predict scores in the range of $\pm$ 8 goals of the aual score and for away we can make prediction in the range of $\pm$ 9-10.
The result is quite good and can we improved more by different hyper parameters or better regression model in the future.
There were some limitations of our project :
Getting the data of last 20 games from the 'balldontlie' API took about 15-20 secs for each team and that's why our website may took a half minute to refresh. Due to this we have to reduce the epochs during the retraining of the model so that it takes less time. This affected our accuracy a little.

8 References

- Balldontlie API : The data is gathered from here
- NBA Page : Getting NBA yesterday and Upcoming games
- Our Web Page Link
- A similar Github project with different datset used for references
- Github Link for our project
- *Prediction of NBA games based on Machine Learning Methods by Renato Amorim Torres*