Who is an All-Star?

Machine Learning for the NBA

Ezekiel Sung  
 Computer Science  
 California State University of Sacramento  
Sacramento CA US ezekielsung@csus.edu

Hung Truong  
 Computer Science  
 California State University of Sacramento  
Sacramento CA US hungtruong@csus.edu

ABSTRACT

The yearly announcement of the NBA All-Stars is a highly anticipated event that fans across the globe look forward to. Fans look forward to seeing their favorite player be named an All-Star and some may even have money on the line. Our goal for this project is to develop a model that can predict whether a player is an All-Star given their player statistics. We used both a Fully Connected Neural Network model and a Convolutional Neural Network without transfer learning. Since there are no pre-existing designs, we developed and experimented our models from scratch. Our results were that we were able to predict an NBA All-Star with an accuracy of around 96%, given their statistics.

1 Introduction

Every year, the NBA hosts a highly anticipated event halfway through the season called the NBA All-Stars. This All-Stars event often showcases the league’s top and star players in a three-day event. A total of 20 players will be selected for the main All-Stars event and will play against each other to compete to be the best of the best. Voting for these players to be a part of the All-Star games is often done through public voting, current players voting, and media voting. Being named an All-Star is a high honor amongst the community and on top of that, being named an All-Star multiple time is an even higher honor.

We approached this problem by first obtaining a dataset of all NBA players, their statistics and how many times they were selected as an All-Star. We took that dataset and condensed it to only the data that was essential and important in determining an All-Star. Some of the essential attributes we kept were points, assists, rebounds, and their position. We then use this data and train two different models, a Fully Connected Neural Network, and a Convolutional Neural Network. Using the model, we predict it with the test dataset and compare the results of each model to determine the best model.

We contributed a fully connected neural network and a convolutional neural network.

We will show the algorithm design from the beginning and explain a Fully Connected Neural Network model and a Convolutional Neural Network model pre regularization. Then we will show how we used regularization and subsequently show a post regularization of both the models. After showing the algorithm design, we will explain the experimental evaluation and the methodologies we used. Finally, we will show the results of our experiment.

2 Algorithm Design

Overview of our algorithm design is to import the data and prep the data for our models. Then we try and train different models pre and post regularization and compare the results of each model.

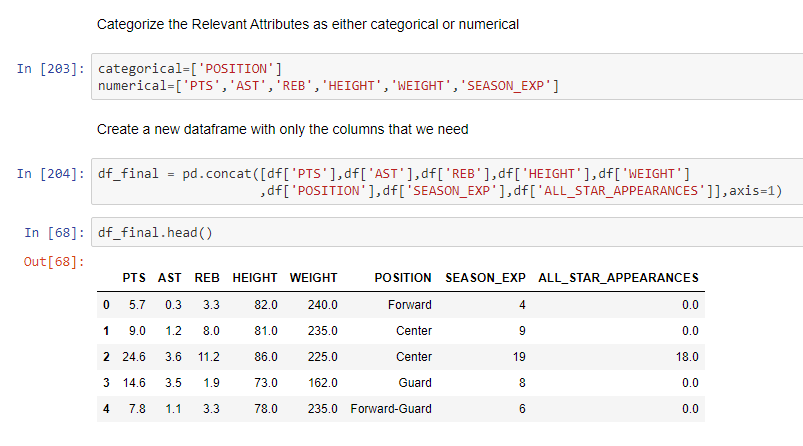
2.1 Prepping the Data

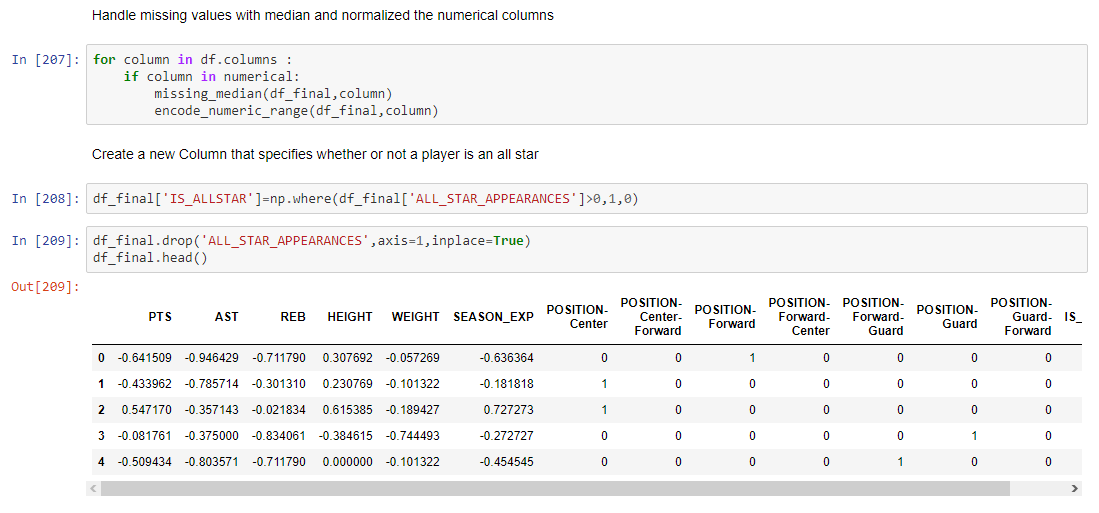
Our dataset was provided to us in a sqlite format, so we had to import sqlite3 and pandas to tensorflow. Then we connected to the sqlite file and executed a query statement to retrieve the data we need and store it inside a pandas dataframe. The results are shown with df.head() which shows the first 5 rows of the dataframe.



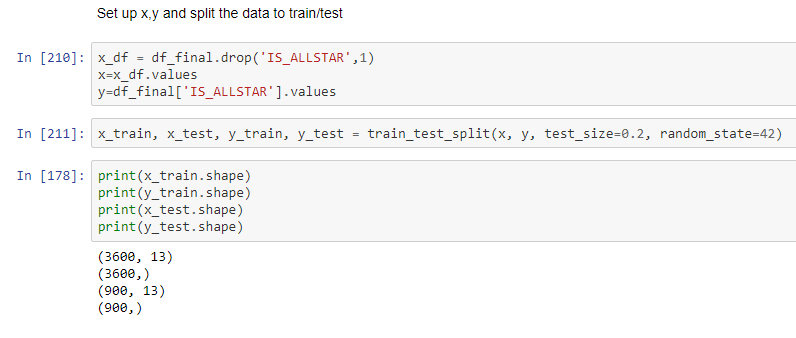
Figure 1: Import dataset and save as pandas dataframe

We categorized the relevant attributes as either categorical or numerical for future use. Then we created and concatenated a new dataframe that holds all the columns that are relevant to our problem. We one hot encoded the categorical column, Position and dropped a column that was created and unneeded because of the one hot encoding.

Figure 2: Categorize Attributes and Create New dataframe

We looped through each numerical column as defined before and handled any missing values the data might have by filling in the median value. In addition, we normalized all the numerical columns as well because our models require normalized data. Then we created a new column “IS\_ALLSTAR” and we looped through “ALL\_STAR\_APPEARANCES” and if the player has been an All-Star, we input a 1 (true) into the new column and 0 (false) if they have not. 

**Figure 3: Handle Null Values and Added “IS\_ALLSTARS”**

We set up the x dataset by dropping “IS\_ALLSTAR” and saving it into a new dataframe and having x be the values of the new dataframe. For y we set that to be the values of the “IS\_ALLSTAR” column. After setting up x and y, we split the dataset into 80% train and 20% test. 

**Figure 4: Set Up Train/Test Dataset**

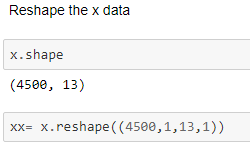
2.2 FNN Model

We first created a checkpointer to save the best weights throughout the training of our model. Then we set up a for loop to run the fitting of the model a couple times to ensure the best results. We first defined the model and added an input layer. The input dimension of our input layer is the shape of our x dataset. We defined the activation and added another dense layer. Finally, we added an output layer with the dimension of 2 because our model has two outputs (0,1) and with activation of SoftMax. We set up a monitor for earlystopping to prevent overfitting the model. Then we go ahead and fit the model and after we train it, we load back up the best weights saved with the checkpointer. Finally, we used the newly trained model to predict the x\_test and displayed the final accuracy score. 

**Figure 5: FNN Model**

2.3 CNN Model

The Convolutional Neural Network model takes an input that is an image, so we had to reshape our x data to mimic an image. We reshaped the x data to (4500,1,13,1).



**Figure 6: Reshape x Data**

For the Convolutional Neural Network model, we first defined it as CNN. Then we added the input convolutional layer with an input shape of (1,13,1). Added a maxpooling layer and another set of cov2d layers. We then added a flatten layer with an addition of a dense layer, a dropout layer and then finally an output layer with activation SoftMax. We then compiled and split the newly shaped data to test and train. Similar to the FNN model, we added a checkpointer and monitor to prevent overfitting, fit the model in a for loop and made predictions with the x\_test. Finally, after the prediction, we displayed the final accuracy score of the model.



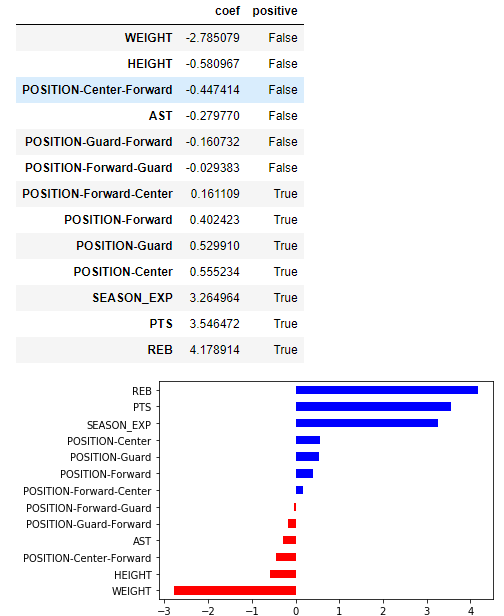
**Figure 7: CNN Model**

2.4 Regularization

We first set up a list of all the attributes in the dataframe. Then we created a linear regression regressor and used it to predict with the x\_test dataset. 

**Figure 8: Set Up Regressor**

With the regressor, we were able to display the importance of each attribute. Here we see that the two positions forward-center and forward guard were of least impact. Although they were positions, we decided to drop them to see what results we would get.



**Figure 9: Results of Regularization**

3 Experimental Evaluation

3.1 Methodology

The data we used were statistics of all NBA players from every single game since 1946. The dataset includes data on 4500 total players on 30 different teams. The specific data we used for our models include points, assists, rebounds, height, weight, season experience, All-Star appearances, and the position they play. After prepping the data, we were left with a dataset of 4500 players and their statistics. For both different models, we split the data into 80% train and 20% test.

The experiment setting is to determine whether a player is all-star material given their basketball statistics. The model is given a dataset of all NBA players and their relevant statistics and whether they were an All-Star or not. With that being said, the model will learn and train with the data to be able to predict new players or see what it will take statistically for a player to be deemed an All-Star.

We used accuracy scores to initially compare all the different variations of the models. For the best models of each network, we compared the confusion matrix and the classification report which includes precision, recall and f1 score.

We implemented a fully connected neural network model and a convolutional neural network model. There were no other existing implementations of this problem, so we did not have other methods to compare to.

3.2 Results

3.2.1 Pre-Regularization FNN Model

We ran multiple trials of different parameters and regardless of what parameters we have, we were getting roughly the same accuracy score for all of them which was around 95%. Our worst FNN model pre regularization was given to us with sigmoid activation, sgd optimizer with learning rate of .001, and just 1 extra dense layer with 5 neurons. These settings gave us an accuracy score of 94%

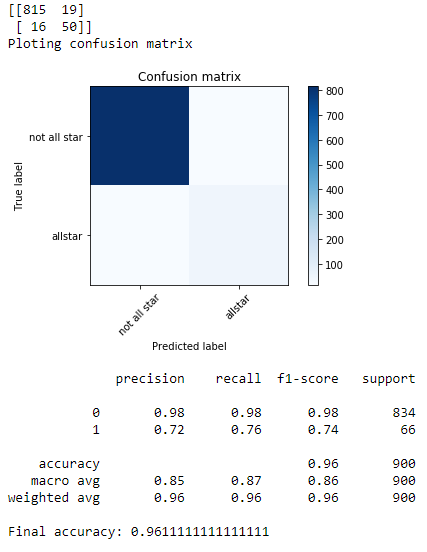


**Figure 10: Worst Pre-Regularization FNN Model**

We were able to conclude that after many different trials that despite us adding more hidden layers and neurons, the final accuracy score does not vary a significant amount. For that reason, we kept most of our models with 1 hidden layer and 5 neurons. We also found that regardless of which optimizer we used, having a learning rate of around .001 gave us slightly better results while maintaining a reasonable training time. In addition, the Adam optimizer also gave us very slightly better results as well which leads us to our best pre regularization FNN model. This model consists of tanh activation, Adam optimizer, learning rate of .01 and 1 dense layer with 5 neurons. 

**Figure 11: Best Pre-Regularization FNN Model**

This model gave us a final accuracy score of 96% and as shown are the precision, recall and f1- scores.

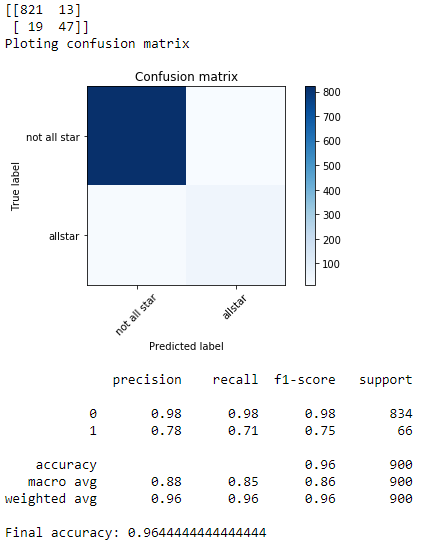


**Figure 12: Best FNN Model Results**

3.2.2 Pre-Regularization CNN Model

Similar to previous FNN models, we found very similar results. Much like the FNN models, all the final accuracy scores of the different trials are very close to each other even with drastic variations in settings. We found better success with the Adam optimizer compared to sgd. Our best CNN model before regularization consists of Relu activation and the Adam optimizer.  Two conv2d layers, one being the input layer and 2 maxpooling layers, a flatten layer, 1 dense layer with 1024 neurons and a dropout layer.

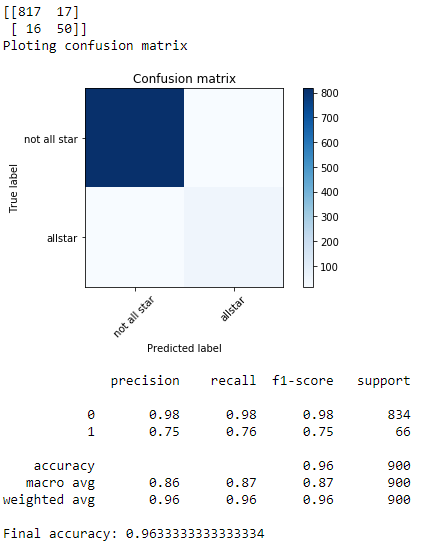
The final accuracy score is 96.4% which is higher than the FNN model. But looking at the confusion matrix, the FNN model was able to predict 3 more All-Stars correctly.



**Figure 13: Best Pre-Regularization CNN Model Results**

3.1.3 Post-Regularization FNN Model

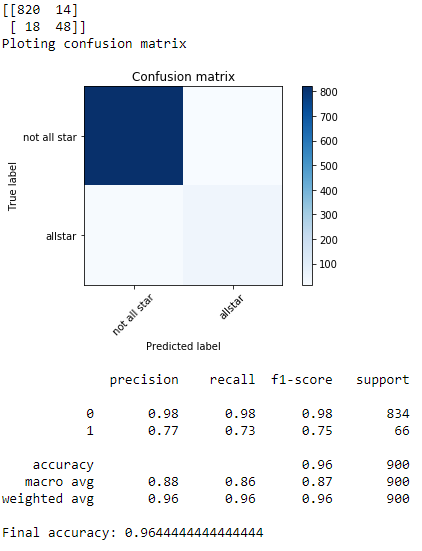
After regularization we tested out the same models to see if it improved our models or not. After many trials we have concluded that although it may seem like it increased just a tiny bit, the change was not significant enough to conclude that it was clearly any better after removing the insignificant features.

The best post regularization FNN model was given to us with Relu and Adam optimizer. We doubled the number of neurons and added 1 extra layer to see if there were any improvements and it was very little. After regularization, there was an increase of .02% for the FNN model. 

**Figure 14: Best Post-Regularization FNN Model Results**

3.2.3 Post-Regularization CNN Model

Similar to the FNN model, after regularization the results were not any different than the pre regularization. For our best post regularization, the final accuracy score was .01% lower than pre regularization. Although according to the confusion matrix, it was able to predict 1 more All-Star than the previous CNN model.



**Figure 15: Best Post-Regularization CNN Model Results**

4 Conclusion

Overall, after many different trials and different models, the results are fairly similar. Although the final accuracy score may be very slightly higher for the CNN models, the FNN models were actually more accurate in actually predicting All-Stars. Even after removing features that were of least importance, the results did not vary much, which we can conclude that these models have reached their potential with the given dataset. Our final best model was given to us by a Fully Connected Neural Network model that had the irrelevant features removed. We deemed it the best because it was able to predict the most All-Stars and was able to predict the most non-All-Stars.

5 Work Division

Ezekiel was in charge of the project proposal and putting together a proposal presentation. He also did the data cleaning and prepping for the project. Then he was in charge of the regularization and FNN model. Hung was in charge of finding the right dataset. Once the data was prepped, he was tasked with the CNN model. Once the project was done, he made the report. After the report was done, Hung and Ezekiel worked together to plan out how the presentation will go and created the presentation slides.

6 Learning Experience

Unlike other projects we had, this one was the most interesting one. Mainly it was because it is a project that we had to come up with and decide on how it will go. We are both big basketball fans, so working on this project made us realize that if the subject is something you are interested in, you tend to have a lot more fun. Although we ran into trouble when working with the models like getting the same prediction scores even though we changed the parameters, we learned that it is a common issue and using regularization to find the most important features can solve that problem. We think that the model may have reached its potential because the dataset that we have did not have enough players that were All-Stars, so our models did not have enough data to learn from. In addition, we would like to try instead of using a player’s career statistics to train the model, to use each season’s statistics as a data point. That will result in more accurate data and more data for model to learn from.

Overall, it was a great experience to find a dataset to work with and creating a problem that we could tackle from that dataset.

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00