Predicting NCAA Shots

Nick Mazzotta

October 1 2020

1. Introduction

1.1 Background

College sports provide a starting point for young athletes on their way to professional careers. One of the major leagues in this category is the NCAA Men's Basketball League. Player performance in a game is dependent on many factors; some tangible and some not. However, there are some concrete statistics we can utilize to make some informed predictions.

1.2 Problem

We want to determine if a shot will go in based on the circumstances surrounding the shot – not necessarily factoring in the player's skill. We aim to predict the likelihood of a given shot scoring based on only factors that are a circumstance of the game.

1.3 Interest

This analysis could be of interest to sports analysts, coaches, and players for use in improving shot selection and running defensive strategies to limit high percentage shots. It could also be of interest to fans for purposes of fantasy basketball and betting.

2. Data Sourcing and Cleaning

2.1 Data Source

The entirety of the data is available as a public data source in Google's BigQuery platform which can be found here. The specific table used "mbb_pbp_sr" which contains play by play event data for each match. For our usage, we select the most recent year of data available and use a subset of one million rows of event data.

2.2 Cleaning Data

The data was already in a relatively workable state. The data was filtered for only the event types pertaining to shot attempts, and then selected our target year of 2017.

The data in a categorical format, containing such features as “shot type” which can have a value such as “layup” or “jump shot”. Thus, we need to create dummy variables for every feature we want to be used in our algorithm.

Another datapoint that was transformed was the (x,y) coordinates of the shot. The distance from the net was calculated and used in place of these variables.

Since the data was brought in through an SQL query, there is no need to grab features we do not plan to use. For this reason, we select only features we think will impact the prediction:

- tournament
- tournament_type
- team_basket*
- away_name
- home_name
- period
- elapsed_time_sec
- team_name
- event_coord_x
- event_coord_y
- event_type
- shot_made
- shot_type*
- shot_subtype*
- points_scored
- three_point_shot

* later converted to dummy variable

3.3 Data Exploration

First, we can take a look at all of the made shots mapped out:

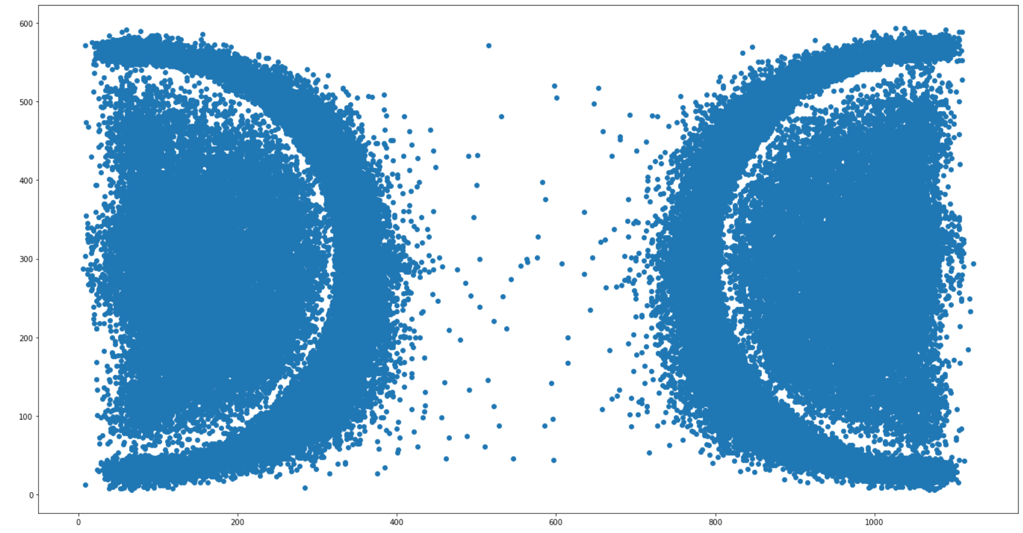
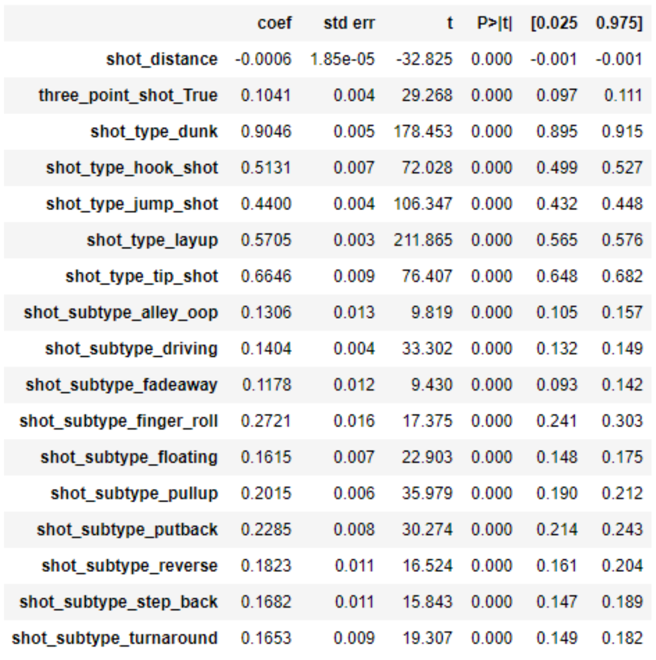


Figure 1. Location of made shots.

There are very few shot attempts from just inside the three-point line as there is high risk and low return.

We check to see if there are any immediate and strong correlations in the features:



	coef	std err	t	P>|t|	[0.025	0.975]
shot_distance	-0.0006	1.85e-05	-32.825	0.000	-0.001	-0.001
three_point_shot_True	0.1041	0.004	29.268	0.000	0.097	0.111
shot_type_dunk	0.9046	0.005	178.453	0.000	0.895	0.915
shot_type_hook_shot	0.5131	0.007	72.028	0.000	0.499	0.527
shot_type_jump_shot	0.4400	0.004	106.347	0.000	0.432	0.448
shot_type_layup	0.5705	0.003	211.865	0.000	0.565	0.576
shot_type_tip_shot	0.6646	0.009	76.407	0.000	0.648	0.682
shot_subtype_alley_oop	0.1306	0.013	9.819	0.000	0.105	0.157
shot_subtype_driving	0.1404	0.004	33.302	0.000	0.132	0.149
shot_subtype_fadeaway	0.1178	0.012	9.430	0.000	0.093	0.142
shot_subtype_finger_roll	0.2721	0.016	17.375	0.000	0.241	0.303
shot_subtype_floating	0.1615	0.007	22.903	0.000	0.148	0.175
shot_subtype_pullup	0.2015	0.006	35.979	0.000	0.190	0.212
shot_subtype_putback	0.2285	0.008	30.274	0.000	0.214	0.243
shot_subtype_reverse	0.1823	0.011	16.524	0.000	0.161	0.204
shot_subtype_step_back	0.1682	0.011	15.843	0.000	0.147	0.189
shot_subtype_turnaround	0.1653	0.009	19.307	0.000	0.149	0.182

Figure 2. Summary of coefficient and P values.

As expected, some shot types such as “dunk” and “tip shot” are strongly correlated with a shot going in. We also observe that the distance from the net has less correlation than expected.

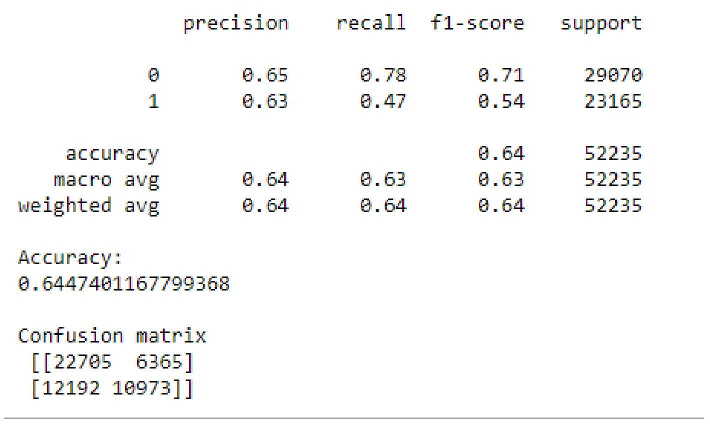
3. Application of Models

3.1 Benchmarking - Logistic Regression

Logistic Regression was used as a benchmark for other models throughout testing. Upon first run, having not factoring the shot distance in, the maximum accuracy achieved was 57.45%.

After calculating and adding shot distance into the model, we were able to increase the accuracy to 63.64% using 500,000 rows.

In the final run, using 1 million rows, we achieved a precision of **64.47%**



```
              precision    recall  f1-score   support

           0       0.65      0.78      0.71     29070
           1       0.63      0.47      0.54     23165

    accuracy                           0.64     52235
   macro avg       0.64      0.63      0.63     52235
weighted avg       0.64      0.64      0.64     52235

Accuracy:
0.6447401167799368

Confusion matrix
 [[22705  6365]
 [12192 10973]]
```

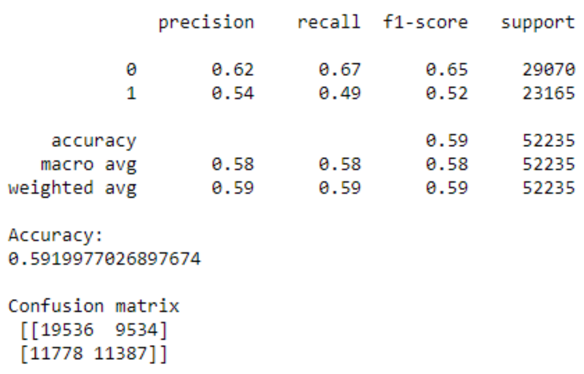
Figure 3. Accuracy Metrics for Logistic Regression Model

3.2 Worst Performing - Random Forest Classifier

This model was implemented as it is known to be accurate for classification problems on large sets of categorical data with high dimensionality.

With 1 million rows, we were able to achieve a precision of **59.20%**

It was observed that this model had a much higher false positive rate than the others, which is evidenced in the confusion matrix.



```
              precision    recall  f1-score   support

           0       0.62      0.67      0.65     29070
           1       0.54      0.49      0.52     23165

    accuracy                           0.59     52235
   macro avg       0.58      0.58      0.58     52235
weighted avg       0.59      0.59      0.59     52235

Accuracy:
0.5919977026897674

Confusion matrix
 [[19536  9534]
 [11778 11387]]
```

Figure 4. Accuracy Metrics for Random Forest Classifier

3.3 Summary of Model Performance

Overall, logistic regression remained the best performing model of the group.

This is a result of it having the lowest false negative rate of all the models, which proved to be the biggest source of inaccuracy in general.

Model	Logistic Regression	Random Forest Classifier	Neural Network	Gradient Boosting	K-Nearest Neighbours	Support Vector Classifier
Accuracy	64.47%	59.20%	64.43%	63.79%	64.33%	

4. Conclusion

In summary, we were able to produce a model that is able to predict the if a basketball shot in the NCAA 2017 season will go in or not with an accuracy of 64.47%. To do this, we used one million rows of event data from NCAA games to build a classification algorithm that can tell us if the shot will go in or not based on such things as shot type, distance from the net, and time left in the game.

This could help coaches and players identify key shooting areas where they are most likely to have success. It also can be applicable to fans and hobbyists interested in sports statistics.

5. Opportunities for Improvement

We were able to increase the accuracy by over 10% from our initial diagnostic run – however there is still a lot of inputs that would help to improve the accuracy. One noticeable gray area in this dataset is any statistical information on the shooter – this is intentional in the current state, as it is more of an analysis of shots based on position and situation within the game as opposed to a measure of the players skill. However, if we wanted to increase the predictive power, incorporating player statistics (namely age, shooting percentage, position) would likely have a large positive impact on the model's accuracy.