MLND Capstone Project

Using Machine Learning to predict NCAA basketball results

Márcio Luiz Bezerra Lopes Júnior

April 2019

1 Project Overview

Data has been stored and used in sports for quite some time, even in the early years of many sports leagues, numbers like Points Scored (in basketball) or Yards Passed (in football) have been used by critics and teams to track athletes' performances. In Baseball Reference (Sports Reference LLC, 2000), for instance, it is possible to find statistical data from baseball games dating back to the 19th century.

More recently, as more and more sophisticated statistics and algorithms were and are still being developed, analytics plays a bigger role than ever in professional sports, and every professional team in the Big Four American leagues has its own analytics department or, at least, analytics experts. (Steinberg, 2015)

A well-known case about the use of Analytics in Sports is the late 1990's/early 2000's Oakland Athletics. Not as well financially as some of their MLB rivals, the A's used data analytics to keep them competitive even with a low budget, by finding undervalued players – who would look considerably better on a statistical level than on the baseball critics/scouts analysis. This case was the subject for the book *Moneyball* (Lewis, 2003), and the 2011 film based on it.

There are several articles in the use of ML in sports: Kannan *et al.* (2018) projects the success of NBA Draft prospects based byusing ML and Data Analytics; Warner (2010) attempts to predict the margin of victory in NFL games; and Arndt and Brefeld (2016) predicts the future performance of soccer players. Besides those, Pischedda (2014), Cheng *et al.* (2016), Tim and Sandjai (2018) have all attempted to predict match results for different kinds of sports.

2 Problem Statement

Given datasets about NCAA Basketball, the project will target the making of a predictive model for basketball (or college basketball) that will be fed with data about the opposing teams in a single match, and then it will decide which team will win that match. There will be three different outputs:

1. **Binary:** simply guesses which team wins (0/1).
2. **Margin of victory:** guesses the score difference of the match.
3. **Chance of victory:** chances of each team winning that match (in percent).

While the *binary* model is a classification problem, the other two are regression problems, but since there isn't an actual true way of calculating *chance of victory* – it's a subjective value (unlike *points scored* and *who wins*) – the two regression problems will actually be turned into the same problem, with the *chance of victory* being calculated from the predicted *margin of victory.*

In order to make the classification and regression, the data from NCAA will be used to find season statistics from teams playing in the college competition. This data will be expanded through the calculation of *advanced statistics* (4.2) and addition of *national rankings.* Data will be formatted in a one-versus-one style, appending the retrieved data to all regular season and post season games since 2010 – which is the first season where *play-by-play* data is recorded. Machine learning models will be trained using this data in order to predict both outcomes. Finally, the result from the regression problem will be used to calculate the *chance of victory.*

3 Evaluation metrics

Each of the three problems stated will have its own evaluation method, as they have different characteristics. The chosen evaluation metrics are those that matched existing benchmark models that were used to evaluate this project's results.

Binary

The Binary model is a classification problem, and to compare it to other binary classification articles, it will be evaluated by its accuracy. Accuracy is a statistical measure that shows the percentage of correct classifications.

$$Accuracy = \frac{Correct\,predictions}{All\,predictions}$$

Margin of victory

The margin of victory problem will be evaluated by two measures, the mean absolute error (MAE) and the root mean squared error (RMSE).

The MAE is given by:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t|$$

Where:

- n is the amount of predictions
- e_t is the difference between the predicted value and the true value

The RMSE is given by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Where:

- n is the number of games played
- $\hat{y}_i$ is the predicted margin of victory
- y_i is the actual margin of victory

Chance of victory

In order to use Kaggle's best leaderboard scores as benchmark models, the evaluation metric for the chance of victory prediction will be the same as in the competition. That is, the Log Loss.

$$LogLoss = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

Where:

- n is the number of games played
- $\hat{y}_i$ is the predicted probability of team 1 beating team 2
- y_i is 1 if team 1 wins, 0 if team2 wins
- $log()$ is the natural (base e) logarithm

4 Data Analysis

4.1 Datasets

The analysis of the data in this project is restricted to the data provided by NCAA and Google Cloud in the 2019 March Madness ML Competition. The datasets can be split into 6 groups:

1. **NCAA Results:** outcome of NCAA matches, some detailed with the boxscore, some only with the points and winners, and some with only the winners.
2. **NCAA General Information:** NCAA season organization, including the teams, their conferences, players and coaches.
3. **Geographical:** cities (general) and also cities where matches were played.
4. **Team rankings:** several national rankings that try to order the best teams in college basketball, from season's start to end. Uses Massey Rankings' file.
5. **Play-by-play:** several events that happened during the games, at what time they happened, and which player was responsible.
6. **NCAA Tournament Bracket:** how the NCAA national tournament seeding and games are decided.

These datasets will be used in the search and calculation of numbers that are relevant to decide a basketball game probable winner. These numbers will mostly be based on common statistics, and also *advanced statistics.*

Although the main goal is to predict post-season *March Madness* results/outcomes, it is important to see how the post-season data is considerably smaller than the regular season, as it can be seen in Figure 1.

This is important because even though the amount of usable data in this project is quite large, it consists mostly of regular season data. *March Madness* gets its nickname from the popular view that unexpected results are more likely to happen in the NCAA Tournament, and a lot of data might be necessary in order to get a deeper view of *why* is it that favorite teams become "more beatable" at the Tournament. Unfortunately, as it can be seen in Figure 1, a single regular season has more than double the amount of games as all post-season results since 2010.

4.2 Advanced Statistics

In a basketball boxscore, traditionally a few numbers are registered: points, rebounds, assists, steals, blocks, shots and fouls. Usually, players and teams are judged by the average numbers they have of those cited. Although this may be

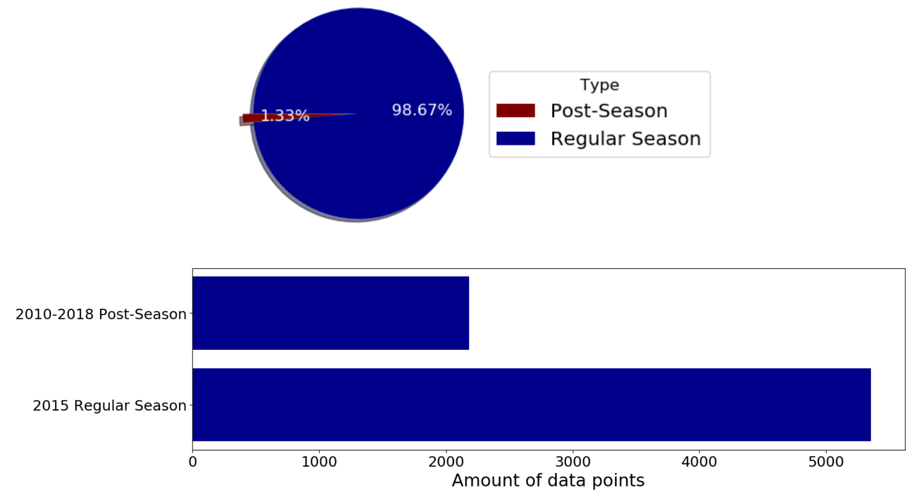

Figure 1: Difference of regular season and post-season data.

sometimes enough to compare teams or individuals, it is something generic that doesn't consider **how** their games were played.

In order to find a metrics that are able to show teams' tendencies, advanced statistics were creating be establishing important relationships between existing stats. Some key factors in advanced stats is the consideration for the pace in which games were played, and the measurement of stats per offensive possessions instead of games. An example of the importance of the difference of calculating through possessions instead of the traditional "per game" are Paul Westhead's teams:

Paul Westhead is a coach who always mada his teams play at an extremely fast pace, shooting as fast and as much as possible, even if it means taking very contested shots, or shots far away from the basket. He applied this style of play in college basketball, as well as in the NBA and in the WNBA. If you check the team statistics from the 1990-91 NBA season, the Denver Nuggets, then coached by Westhead, led the entire league in points per game, which normally leads people to believe that their offense was good that year, but when it is calculated how many points they scored *per possession*, they were only the 21st best out of 27 teams that played in the NBA. In other words, they were not efficient, they just played through more possessions, or on *faster pace*, than any other team, leading them to more points made (and allowed).

This is an exaggerated example, since Westhead's coaching style is very unique in basketball, but it goes to show how a good offense cannot be evaluated simply by how many points they score. Some advanced stats then use mathematical approximations to calculate team's efficiency through *possession*.

Another trend that is growing with the increasing amount of data in sports is the style-of-play type: data that tells where a team shoots from the most, how often a player *drives to the basket*, among other things. A commonly used feature that makes use of that kind of data is the *shooting map*.

As it can be seen in Figure 2, the style-of-play is currently changing in NCAA basketball, with 3-point jump shots being taken more often, and 2-point jump shots taken less often. Having that kind of data, specially in one-versus-one predictions might be useful to find how different styles play each other and which statistics are more important to each style-of-play.

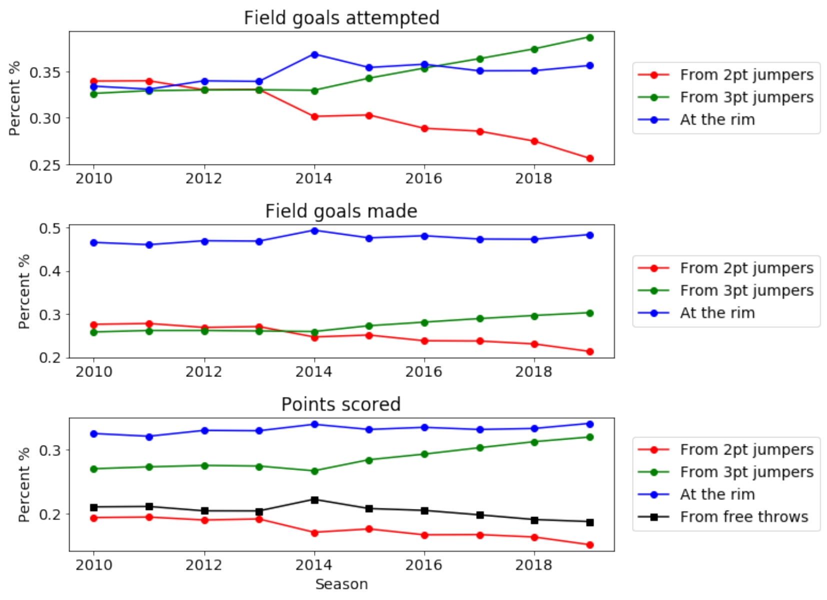

Figure 2: Shooting trends in NCAA.

4.2.1 Possession and Pace

A team is in **possession** when it has control of the basketball. The time of possession goes from when the team got the ball until it shoots it or loses it somehow. Basketball analytics calculate a possession by the equation:

$$POSS = 0.96 * (FGA + 0.44 * FTA - ORB + TOV)$$

And the pace of the game, which averages the amount of possessions per 40 minutes is given by:

$$PACE = \frac{40}{MP} * \frac{POSS + OppPOSS}{2}$$

4.2.2 Shooting Map

One of the ways to understand which style a team plays, and also how efficiently a team is in different spaces on the court is by building something called "shooting map", in which you mark all the places where a player on that team has taken a shot, forming a type of heatmap.

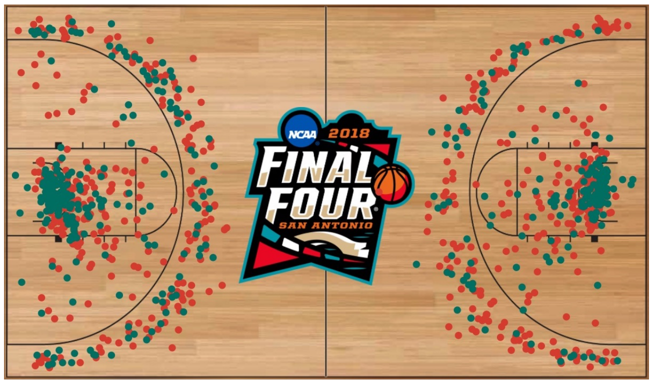

Figure 3: 2018 NCAA Final Four shooting map.

Although there is no data on the exact shot location, by checking the play-by-play event files, there can be found six event types that are related to shots/scoring: 3-jump, 2-jump, 2-layup, 2-tip, 2-dunk, 1-free. Each having a 'made' and 'miss' kind. These events are able to tell if teams are more likely to shoot 3-point shots, 2-point jumpers or shots at the rim (dunks, layups, tips).

Considering that the prediction will be team-versus-team, having knowledge of teams' playing tendencies might play a key role on detecting the winner.

4.3 Metrics

By using the concepts explained and some commonly used *advanced stats*, the following stats were calculated and used as the main features for this project.

4.3.1 Offensive/Shooting

Abbrev	Name	Equation	Description
PPG	Points per game	PTS	Points scored per game.
OffRtg	Offensive Rating	$\frac{PTS}{POSS}$	Points scored per possession.
PP	Play percent	$\frac{FGM}{FGA+TOV-OR}$	How often the offense ends in a field goal.
FG%	Field goal percentage	$\frac{FGM}{FGA}$	Percentage of shots that end up in points.
3FG%	3-point field goal percentage	$\frac{3FGM}{3FGA}$	Percentage of 3-point shots that end up in points.
eFG%	Effective field goal percentage	$\frac{FGM+0.5*3FGM}{FGA}$	Weighted percentage of shots that end up in points.
TS%	True shooting percentage	$\frac{PTS}{2*(FGA+0.44*FTA)}$	Shooting efficiency measure.
FTR	Free throw rate	$\frac{FTA}{FGA}$	How many free throws are taken for every shot attempt.
3PR	Three point rate	$\frac{3FGA}{FGA}$	How many team shots are 3-pointers.
RimRt	Rim rate	$\frac{RimFGA}{FGA}$	How many team shots are taken at the rim.
2PJ-FG%	2-point jumper FG%	$\frac{2PJ-FGM}{2PJ-FGA}$	Field goal percentage of 2-point jumpers.
Rim-FG%	Rim shots FG%	$\frac{RimFGM}{RimFGA}$	Field goal percentage of shots at the rim.

4.3.2 Defensive

Abbrev	Name	Equation	Description
OppPPG	Opponent points per game	OppPTS	Average amounts of points allowed per game
DefRtg	Defensive rating	$\frac{OppPTS}{OppPOSS}$	Average points allowed per opponent's possession
DPR	Defensive plays rate	$\frac{BLK+OppTOV-STL}{OppPOSS}$	Frequency of defensive plays per opponent's possession
BLK%	Block percentage	$\frac{BLK}{OppFGA}$	Percetange of opponent's shot attempt blocked
BLKR	Block rate	$\frac{BLK}{OppPOSS}$	Frequency of blocks per possession

4.3.3 Rebounding

Abbrev	Name	Equation	Description
ORPG	Offensive rebounds per game	OR	Average offensive rebounds per game
DRPG	Defensive rebounds per game	DR	Average defensive rebounds per game
OR%	Offensive rebound percentage	$\frac{OR}{OR+OppDR}$	How often the team wins the rebound battle on offense
DR%	Defensive rebound percentage	$\frac{DR}{DR+OppOR}$	How often the team wins the rebound battle on defense

4.3.4 Passing/Ball control

Abbrev	Name	Equation	Description
AST%	Assisted shots percentage	$\frac{AST}{FGM}$	How many made shots are assisted
AST/TOV	Assist per turnover	$\frac{AST}{TOV}$	Ratio of assists per turnovers
TOVR	Turnover ratio	$\frac{TOV}{FGA+0.44*FTA+AST+TOV}$	Percentage of a team's possessions that end in a turnover.
ASTR	Assist ratio	$\frac{AST}{FGA+0.44*FTA+AST+TOV}$	Percentage of a team's possessions that end in an assist.

4.3.5 Winning Record and Rankings

The winning record of a team during the regular season is also considered and will be used to help select the winner. For obvious reasons, winning more is a factor to make a team more likely to win.

Unlike other sports leagues, though, the NCAA has over a hundred college teams competing in many divisions, and the teams' schedules are entirely different. So not only it's important to know the amount of wins, but also in which conference the team plays in. For that, a "conference level", based on rankings, will be added as feature.

5 Benchmark

For the *chance of victory* prediction, which is the one required by the NCAA–Google Cloud competition, Kaggle provides its users with a leaderboard, which shows the performance of other users' algorithms. By comparing the results with the top scores on the leaderboard, it will be possible to evaluate this prediction's results.

The *margin of victory* will use as benchmark model the article written by Heit *et al.* (1994), which uses people's opinion to guess NBA games' points difference. It has a mean absolute error of 6.4 to 7.1. But the main benchmark model will be Beaudoin and Duchesne (2018), which also uses machine learning applied to NCAA, predicting margins with a root mean squared error of 11.22.

Finally, *binary* predictions will use three articles as benchmark models: Forsyth and Wilde (2014), Zimmermann (2016), and Lin *et al.* (2014). The best prediction accuracy for each of these was, respectively: 73.22%, 65% and 65.15%.

6 Methodology

6.1 Data pre-processing

First, the data from the teams' statistics from each match and for every season was extracted, saving the average stats for every season. Some other numbers were calculated by those stats and added as new columns, and these seasons stats were saved in a new data file, where every combination of team and season was linked to their respective stats. Then, the same results file was used to extract the amount of wins and losses for every team during each season, also saving it by team and season.

Massey's Rankings files were used to get every team's rankings on the last days of each regular season. These rankings are a subjective analysis made by sports specialists. There are several different rankings, so the mean ranking value was taken for every team. That ranking information was also used to rank conferences by their teams' "stregth". These team ranking and conference ranking information were saved in another file, indexed as the other files mentioned.

The play-by-play data was also processed in order to extract information for each team's style of play (volume of 3-pointers, jump-shots or dunks). All files were processed together and organized.

Those files were then added to each match-up file, containing the season where the NCAA Tournament and Regular Season match was played, and the IDs for the two teams playing. In order to feed the machine learning models, the matches' data was normalized in a 0 to 1 scale, and the label data columns were transformed into several binary columns for each label. The then full normalized dataset was split into a train set and a test set. Specific post-season test sets for each post-season were also created.

6.2 Selected models

This project has three goals, two of them are regression problems, while the other is a classification problem. The same algorithms were applied to each case: Support-vector Machines (SVM), AdaBoost and Neural Networks. Linear Regression was also tested for the regression models, and Logistic Regression for the classification model.

Linear regression

Linear regression is an algorithm that uses known inputs and outputs to define a linear function. It gives each input a weight (constant value) and uses them to calculate the future inputs. Linear regression is a pretty simple algorithm and it tends to achieve good results with linear data, but it doesn't have a way of dealing with nonlinear data.

$$output = w_0 + w_1 * i_1 + w_2 * i_2 + w_3 * i_3 + ... + w_n * i_n$$

Logistic regression

Logistic regression is a **classification** algorithm that classifies an input into binary categories. It does so by training and refining a function whose outputs can only go from 0 to 1. This function is built by using division and the exponential function.

$$p = \frac{1}{1 - e^{-(w_0 + w_1 * i_1 + w_2 * i_2 + w_3 * i_3 + ... + w_n * i_n)}}$$

The relationship between Logistic regression and Linear regression can be seen in the values that power the e constant, as they form a linear function.

Support vector machines

Support vector machines, or **SVM**, is an algorithm that uses nonlinear mapping to transform the data into a new – higher dimensional – space. This space can represent nonlinear functions in a linear form, which allows SVM to find linear boundaries that can "split" the data. This new space creating technique is called *the kernel trick*.

On the classification problem, on this new space, the SVM then searches for a hyperplane that separates the data with a *maximum* margin. Which is the hyperplane that has the largest distance to closest points on each "side" of the data. For the regression problem, it also reduces the distance to the data points, but it introduces a variable ϵ which defines an area around the boundaries where the distance (or error) is ignored, it will only adjust to points outside of it.

The ability of handling nonlinear boundaries makes SVM a very accurate algorithm, but it is extremely slower to train than most ML techniques.

AdaBoost

AdaBoost is a popular *boosting* algorithm that uses the results of other ML algorithms, weights those results, and uses both the results and weights as inputs to make the final prediction.

In AdaBoost, an algorithm (i.e. *Random Forest*) will be used to make predictions, then, a second instance of that same algorithm will be trained to detect outputs which the first one was wrong at predicting, and that continues with a third instance, fourth, fifth, and so on, until a predetermined number of instances is reached.

It then uses a weighted sum of the outputs from each of these instances to calculate the final result. As AdaBoost depends on other algorithms, it can be considered a *meta*-algorithm.

Neural network

Neural networks, or *artificial neural networks*, are processing structures that use several smaller structures, called *artificial neurons*, to build complex systems.

Each neuron is a simple structure that makes an operation on the data it receives and then passes it forward. The neurons connect with each other, and by summing their simpler operations, they can create more sophisticated structures. Typically, a neuron is made of: inputs, inputs' weights, a constant (*bias*) and a function. It also usually has an activation function, which normalizes the output of its function.

Neurons are grouped into *layers*. Every neural network has at least an *input layer*, a *hidden layer*, and an *output layer*. The *input layer* handles the inputs that the network receives, the *output layer* makes the final operations for the prediction, and the *hidden layer* handles all operations between those two. Neural networks' sizes are commonly measured by how many *hidden layers* they have.

Artificial neural networks have this name because they were originally based on biological neural networks, the "combination" of neurons that is responsible for the human nervous system communication.

7 Implementation

As the focus was split on different algorithms, there was very limited focus on refining the models, basically staying on tests with AdaBoost – for estimators and maximum depth – and logistic regression – for the optimizer. There wasn't much improvement compared to the Scikit Learn default values. Processing limits also were limitations, as SVM took too long to train, and needed to be limited to 10,000 iterations.

The used SVM were linear and were limited to 10,000 iterations, which was enough to obtain some valid results. The AdaBoost model used had 100 estimators and a maximum depth of 3. The Logistic regression model used Limited-memory BFGS as optimizer. All other defining aspects were the same as the default values used by the Scikit Learn module.

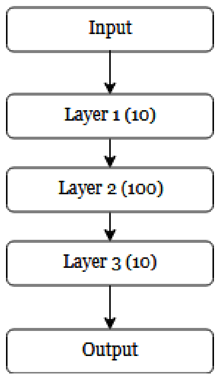

Figure 4: Neural network.

Neural network

The neural network was built on *keras* and *python*. Its architecture had a single input, a single output, and a three layers deep hidden layer, each of these with a ReLU activation function. The output layer uses a *sigmoid* function to make the prediction from the hidden layers' outputs.

Input format

The same input was used for each of the algorithms used. It consists of a *pandas dataframe* that keeps information about the game itself (*phase*, *season* and *hometeam*), and about both team's regular season statistics.

7.1 Winning chance calculation

In order to calculate a team's chance of victory from the margin of victory prediction, I've used pythegorean mathematics as in the following equation:

$$chance = \frac{(n+m)^x}{(n+m)^x + (n-m)^x}$$

Where:

- m is the predicted margin of victory

- n is a fixed normalization value
- x is the powering factor

The x value will change according to the margin of victory predicted, in order the minimize the log loss. Thus, x can be treated as a function of m:

$$x = f(m)$$

8 Results

The results of this project are shown in the following sections for each of the three problems, with their respective evaluation metrics. The results will be split by algorithm and type of data. Test data is made mostly by regular season results, as can be seen in Figure 1, and can be treated as the result for regular season matches. Train data is shown for information purposes (e.g., checking for overfitting) but won't be used for analysis. The remaining results are for each post-season tested, and the overall post-season average.

8.1 Binary Win/Loss

Table 8.1 shows the accuracy on predicting the correct result of a game. The best post-season average accuracy was obtained with AdaBoost (67.49%), which also had the best test accuracy (78.62%).

8.2 Margin of victory

Table 8.2 shows the RMSE for the model's predicted points margin of victory and the actual difference. The lowest error on post-season data was achieved when using SVMs (11.40) and the lowest on test data – "regular season" – was achieved with Linear Regression (10.01).

8.3 Chance of victory

Table 8.3 shows the log loss evaluation for the tested algorithms. The lowest loss for post-seasons was achieved by SVMs (0.579) and the lowest on the test data was achieved by Neural Networks (0.487).

It is important to state again that this result is derived from the *margin of victory* predictions, in which SVMs also had the best results.

[2]SVM models were limited to a maximum of 10,000 iterations.
[3]Test and train includes all seasons from 2010 to 2018, regular seasons and post-seasons.
[4]Average of all post-seasons.

Season	**NN**	**Log. Reg.**	**AdaBoost**	**SVM**[2]
Test[3]	77.19	78.18	78.62	63.55
Train[3]	77.13	78.12	78.62	63.43
2010	68.36	71.88	65.63	57.81
2011	65.67	64.18	67.16	58.21
2012	69.03	73.13	73.13	58.21
2013	59.70	62.69	65.67	55.22
2014	71.27	68.66	70.15	58.21
2015	67.54	67.16	71.64	59.70
2016	65.30	61.19	64.18	62.69
2017	70.15	64.18	68.66	61.19
2018	65.67	67.16	61.19	59.70
Avg[4]	66.97	66.69	67.49	58.99

Table 1: Binary prediction's accuracy.

Season	**NN**	**Lin. Reg.**	**AdaBoost**	**SVM**[2]
Test[3]	10.13	10.01	10.37	11.53
Train[3]	10.13	10.04	10.45	11.54
2010	11.28	11.30	12.94	11.11
2011	11.98	11.92	13.18	12.07
2012	10.56	10.60	12.32	9.73
2013	13.98	14.06	14.98	13.31
2014	11.27	11.38	12.76	11.15
2015	10.14	10.07	11.63	10.34
2016	13.80	13.32	15.50	12.15
2017	11.64	11.86	12.84	10.88
2018	13.03	13.03	13.82	11.86
Avg[4]	11.97	11.95	13.33	11.40

Table 2: Margin of victory prediction's root mean squared error.

Season	**NN**	**Lin. Reg.**	**AdaBoost**	**SVM**[2]
Test[3]	0.487	0.497	0.508	0.551
Train[3]	0.489	0.499	0.509	0.549
2010	0.604	0.611	0.660	0.587
2011	0.626	0.628	0.612	0.613
2012	0.607	0.596	0.604	0.607
2013	0.580	0.588	0.600	0.589
2014	0.577	0.582	0.594	0.574
2015	0.571	0.549	0.593	0.544
2016	0.606	0.603	0.651	0.584
2017	0.563	0.562	0.614	0.510
2018	0.611	0.618	0.653	0.607
Avg[4]	0.594	0.593	0.620	0.579

Table 3: Chance of victory prediction's log loss.

9 Evaluation

Binary

The binary models, as already stated, had an accuracy around 67%. Forsyth and Wilde (2014), which uses data from ESPN, including some already calculated – and more complex – advanced statistics, had in their best-case scenario an accuracy of 73.22%. Zimmermann (2016) predicted NCAA post-season results with an accuracy close to 65%. Lin *et al.* (2014) predicted NBA games at an accuracy of 65.15%. The results of the project had a higher accuracy than the two latter models, but the 6 percent difference to the first is a signal that a few things can certainly be done to improve the model in the future.

Margin of victory

Heit *et al.* (1994) was a psychology experiment that attempted to predict the margin for NBA regular season games by using people's *guesses* after given some information, and obtained MAEs from 6.4 to 9.9. This project had MAE results around 8 and 9 on quick non-reported tests – made only for this comparison – which is inside the benchmark model reported results. Since the article was based on a different league – which has less teams, less conferences and more games – getting results inside their range of MAE is a positive sign.

Beaudoin and Duchesne (2018) used machine learning to predict NCAA regular season games using ranking systems, with a RMSE around 11.2. The **test** data in this project had a RMSE of 10.01 at bests, which shows that the box-score and advanced statistics were successful in improving the prediction, and the post-season prediction results were able to reach similar numbers as their regular season results.

Chance of victory

The chance of victory results were considerably worse than those obtained by other data scientists on the Kaggle challenge. In my project, the log loss post-season average for the best model was 0.579, while on the Kaggle leaderboard, teams/competitors were able to predict results for 2019 with log losses close to 0.4.

10 Conclusion

Results for this project were not as good as I thought they could be, especially on the *chance of victory* prediction. I believed that by using style-of-play statistics, I could have given the learning model a way of discovering bad matchups between teams, and that it would ultimately improve the results. Instead, it remained pretty close to results of other projects that did not use as much information. I still believe the styles-of-play have an important role in match prediction, but perhaps it will require a little more specific data and probably player-level analysis (which was purposely not explored in this project) in order to make actual good use of it. Another aspect to consider as to improve the model would be the analysis at the moment of each match, for instance: statistics from the last n games before the specific game.

The regression problem was satisfactory for me on the *margin of victory*, as I believe the MAE and RMSE to have good values when compared to benchmark models. The *chance of victory* did not achieve a good result, and I think that happened mostly because of the way I chose to calculate the probability. As mentioned before, a *chance of victory* is a very subjective measurement, and I did not take as much time to find a good equation to predict the win probability. With little adjustment, though, as the margins are already calculated, I believe I can improve this on the future.

As for the features, tree classifiers were used to measure which features were more relevant or related to the outcome prediction. For the initial analysis, **home court advantage**, wins/losses and general statistics – that deal with overall efficiency (offense and defense) – were considered more relevant, as can be seen in Figure 5. The outstanding lead by home court advantage might help explain the difference of accuracy in predicting regular season and post-season results, as all post-season games in the NCAA Tournament are played in predetermined – neutral – host cities.

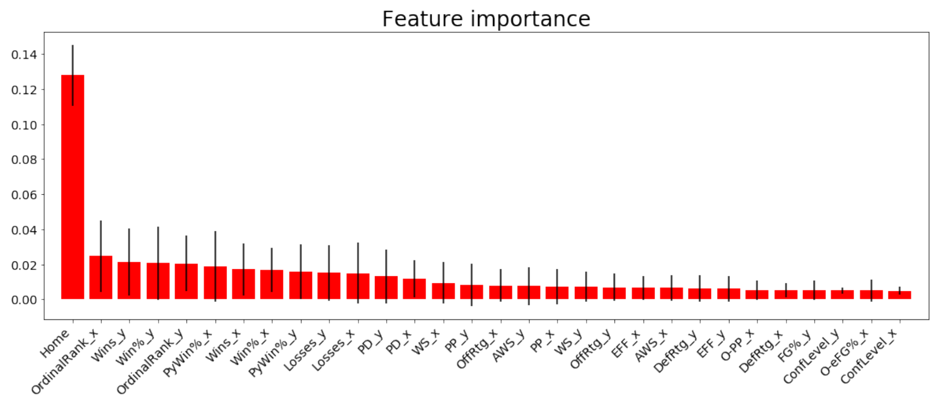

Figure 5: Feature importance

By filtering othrt types, it is possible to see, in Figure 6, the relevance of specific boxscore statistics only. In this case, it's interesting to see that advanced measurement stats as *offensive rating*, *defensive rating* and *play percent* were indeed considered more important than the traditional *points per game* statistic.

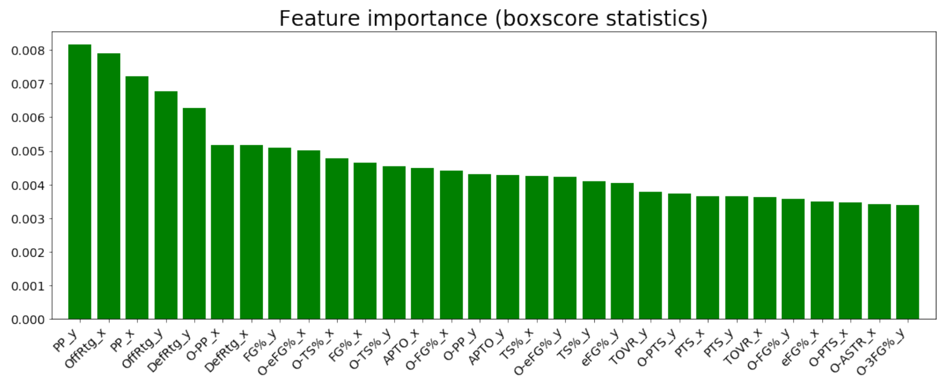

Figure 6: Feature importance for boxscore statistics

Finally, I hope to continue improving these results without the data limitation I've set to this project. By collecting more (external) data on the subject, by using more NCAA's and other basketball leagues' data, and by finding other meaningful aspects to extract from the databases analyzed here. These are key points to future improvements in both the classification and the regression problems.

References

[1] Arndt, C. and Brefeld, U. (2016). *Statistical Analysis and Data Mining: The ASA Data Science Journal* **9**.

[2] Beaudoin, D. and Duchesne, T. (2018). *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology* **232**, 315–322.

[3] Cheng, G., Zhang, Z., Kyebambe, M., and Nasser, K. (2016). *Entropy* **18**, 450.

[4] Forsyth, J. and Wilde, A. (2014). A Machine Learning Approach to March Madness.

[5] Heit, E., Price, P., and Bower, G. H. (1994). A Model for Predicting the Outcomes of Basketball Games. volume 8, pages 621–639. John Wiley & Sons, Ltd.

[6] Kannan, A., Kolovich, B., Lawrence, B., and Rafiqi, S. (2018). *SMU Data Science Review: Vol. 1 : No. 3 , Article 7* .

[7] Lewis, M. (2003). *Moneyball: The Art of Winning an Unfair Game*. Norton paperback. W.W. Norton.

[8] Lin, J., Short, L., and Sundaresan, V. (2014). Predicting National Basketball Association Winners.

[9] Pischedda, G. (2014). Predicting NHL Match Outcomes with ML Models.

[10] Sports Reference LLC (2000). Baseball Reference. https://www.baseball-reference.com/.

[11] Steinberg, L. (2015). CHANGING THE GAME: The Rise of Sports Analytics. https://www.forbes.com/sites/leighsteinberg/2015/08/18/changing-the-game-the-rise-of-sports-analytics/.

[12] Tim, E. and Sandjai, B. (2018). Predicting the outcomes of MLB games with a machine learning approach.

[13] Warner, J. (2010).

[14] Zimmermann, A. (2016). *Statistical Analysis and Data Mining: The ASA Data Science Journal* .