Capstone Project Ben Taborsky
Machine Learning Engineer Nanodegree 8/29/2016

MLB Handicapping With Machine Learning

Definition

Project Overview

Long considered America's National Pastime, baseball has also always been known for the statistical literacy of its followers, from children memorizing their favorite players' batting averages to Bill James launching the Sabermetric revolution. Much of the publicized statistical work done in sports seems to focus on strategies that teams can use to improve their odds, with Billy Beane of the Oakland Athletics a famous early adapter (as portrayed in the 2003 Michael Lewis book Moneyball, and the 2011 movie by the same name). Another prominent use is post hoc analysis to understand what happened in a particular game or season.

Rather than attempting to uncover strategic truths about baseball, this project is an attempt to use simple, widely available statistics to produce accurate estimates of the probability that one team will beat another team.

Problem Statement

The goal of this project is to assemble a number of features that together define a baseball game and to use them to train a model that outputs the probability that the home team wins the game. The steps are the following:

1. Obtain data from retrosheet.org
2. Process data, resolving issues such as double headers and duplicate player names
3. Create relevant features and reduce dimensionality using PCA
4. Train a classifier to output home team win probability
5. Evaluate classifier based on appropriate benchmarks

Metrics

Since the output of the model is a probability rather than a prediction, the error metric used will be cross-entropy, or log-loss, rather than percent accuracy or F1 score.

$$CE = -y \log \hat{y} - (1 - y) \log (1 - \hat{y})$$

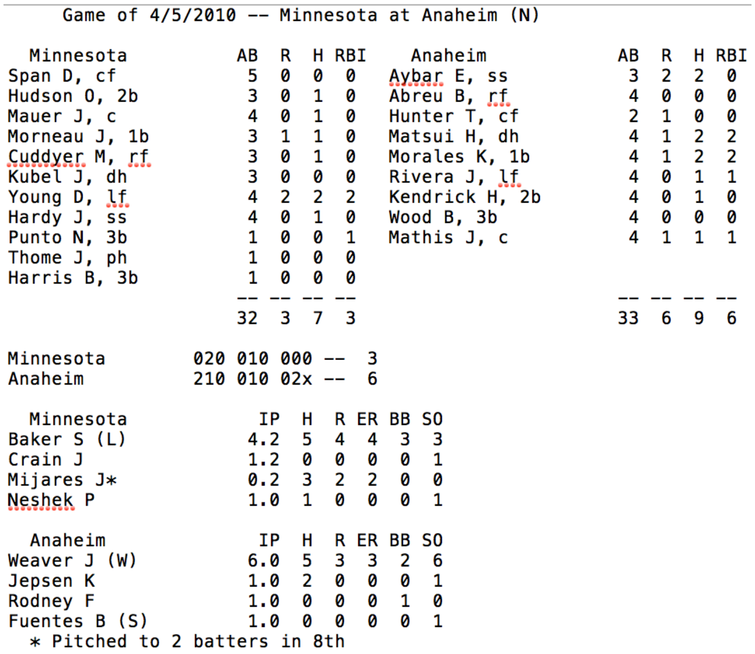
It's important to use this metric instead of precision, recall, or F1 score because in sports handicapping (another term for the estimation of advantages), simply determining the favorite or underdog is not very useful. Instead, for every given example it's crucial to know how likely the outcome is.

Analysis

Data Exploration

Retrosheet has box score data for the vast majority of major league baseball games going back to the late 1800s, but in order to keep the scope of the project reasonable, the analysis is limited to the seasons 2002-2016.

In its rawest readable form, the box score data from retrosheet is in this format:



```
     Game of 4/5/2010 -- Minnesota at Anaheim (N)

  Minnesota          AB  R  H RBI    Anaheim           AB  R  H RBI
Span D, cf            5  0  0  0   Aybar E, ss          3  2  2  0
Hudson O, 2b          3  0  1  0   Abreu B, rf          4  0  0  0
Mauer J, c            4  0  1  0   Hunter T, cf         2  1  0  0
Morneau J, 1b         3  1  1  0   Matsui H, dh         4  1  2  2
Cuddyer M, rf         3  0  1  0   Morales K, 1b        4  1  2  2
Kubel J, dh           3  0  0  0   Rivera J, lf         4  0  1  1
Young D, lf           4  2  2  2   Kendrick H, 2b       4  0  1  0
Hardy J, ss           4  0  1  0   Wood B, 3b           4  0  0  0
Punto N, 3b           1  0  0  1   Mathis J, c          4  1  1  1
Thome J, ph           1  0  0  0
Harris B, 3b          1  0  0  0
                     -- -- -- --                       -- -- -- --
                     32  3  7  3                       33  6  9  6

Minnesota        020 010 000 --  3
Anaheim          210 010 02x --  6

  Minnesota            IP  H  R ER BB SO
Baker S (L)           4.2  5  4  4  3  3
Crain J               1.2  0  0  0  0  1
Mijares J*            0.2  3  2  2  0  0
Neshek P              1.0  1  0  0  0  1

  Anaheim              IP  H  R ER BB SO
Weaver J (W)          6.0  5  3  3  2  6
Jepsen K              1.0  2  0  0  0  1
Rodney F              1.0  0  0  0  1  0
Fuentes B (S)         1.0  0  0  0  0  1
  * Pitched to 2 batters in 8th
```

The data from the box scores supplies some basic batting and pitching stats, and you can easily compute batting average and earned run average. However, it does not include several important statistics such as plate appearances or walks, which are required to compute on base percentage, or the number of times the batter strikes out. It also does not include any defensive statistics for batters, like putouts and assists. Finally, potentially important numbers like total pitches and strikes thrown are omitted for pitchers.

Luckily, these statistics can be calculated by constructing the play-by-play record from retrosheet. Unluckily, this is not quick or easy and requires a fair amount of processing and space. The raw retrosheet files need to be converted into useful CSV files and then parsed for the relevant stats. For this reason, files for both batters and pitchers with all of the important statistics already extracted are supplied.

From the data available for download, three separate types of files were created. One is for batters, and contains one row per batter who appeared in a game - up to 162 per

season per player. Another is for pitchers, with an identical structure. The final file type is a list of players who started each game. Baseball is somewhat unique in that managers are required to reveal their starting lineups prior to the game. T
hese players tend to play a greater role than starters in other sports where reserves and substitution patterns are more important. The bullpen (available relief pitchers) is a notable exception.

Finally, the dataset contains a row for every player in every game they participate in, and include the following statistics, which are explained in detail in the Baseball Stat Glossary at the end of the report:

Player Type	Statistics	Feature Count
Batters	AB, R, H, RBI, BB, SO, PA, BA, OBP, PO, A	11
Pitchers	IP, H, R, ER, BB, SO, ERA, Pitches, Strikes, Starter	9

For the seasons under consideration, the dataset includes a total of 35,522 games. The batters dataset contains 1,011,690 rows, for an average of 28 per game, or 14 per team per game. Meanwhile, there are 275,422 pitching rows, or about 3.5 per team per game. This implies that there are an average of 14 batters and 3.5 pitchers per game for each team. This number seems about right since non-batting pitchers are present as well as non-starting hitters.

Some additional processing was necessary for the sake of unique player names and unique team and date combinations. Because grouping by team and date will be done to select which players will make up the feature set for a game, doubleheaders present a problem because there isn't a unique game identifier. For this reason, and with the suspicion that the second game of a doubleheader might feature tired players not at their best, the second games of doubleheaders were removed from the dataset. For players with identical names, a "2" was added to one of them arbitrarily. There were no sets of three players with identical names.

Exploratory Visualizations

Since modeling a stationary system is much easier than trying to model a changing one, these figures examine a few important aspects of the dataset by season. Figure 1 shows home field advantage by year. As shown in the figure, home field advantage fluctuates yearly in a fairly small range, from just under 52% to around 55.5%, with an average of 53.5% and a standard deviation of about 1.1%. Most importantly, while there is variation, it's small and there doesn't seem to be a trend.

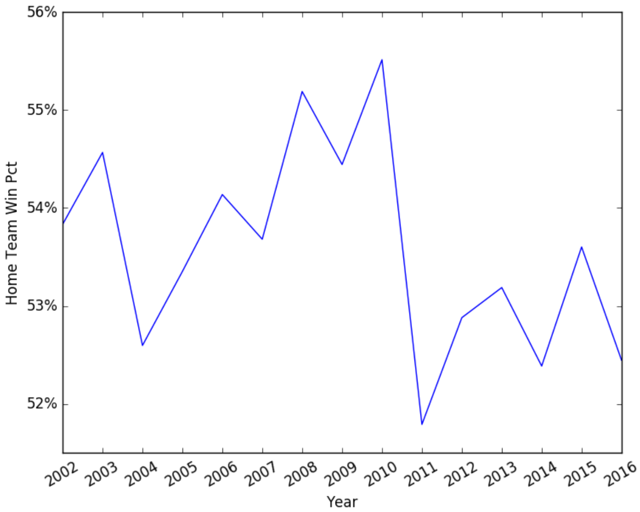


Figure 1: Home field advantage by year

Figure 2 also shows some important features by year, tracking the importance of prominent offensive and defensive statistics over time. It shows the correlation between average on base percentage (OBP) and win percentage among the 30 teams for each season. The red line is the same figure, but for earned run average (ERA). These two statistics are commonly accepted as among the most important offensive and defensive indicators: teams who have better numbers should win more games. Since ERA is better when lower, the sign of the correlation was flipped here to make the numbers comparable. Figure 3 is included to make it a little clearer exactly what is shown in Figure 2. It plots the 30 teams' average OBPs for the season against their win percentages for 2013; the obvious high correlation between these two variables corresponds to the blue peak in 2013 in Figure 2.

As seen in Figure 2, both statistics have significant correlations to win percentage, though the relationships' magnitudes are certainly not constant. Despite the fluctuations, there does not appear to be convincing evidence that the game has changed in lasting ways over the years. Also, judging based on the higher magnitude of the correlations between ERA and winning percentage, pitching quality might be a more consistent and important predictor than a team's batting ability.

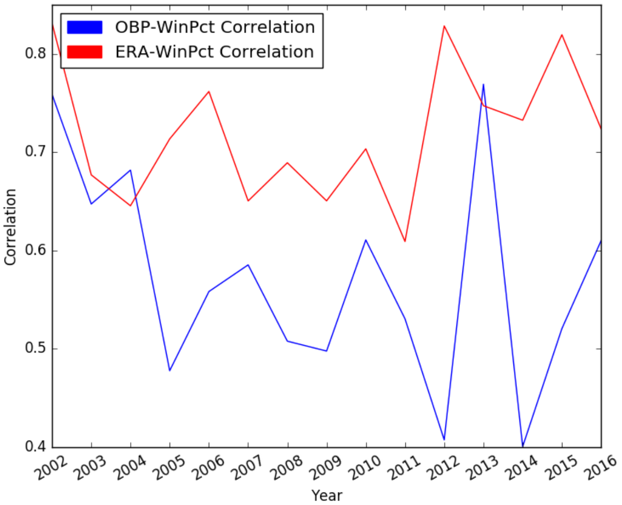


Figure 2: Correlations of Important Statistics to Win Percentage

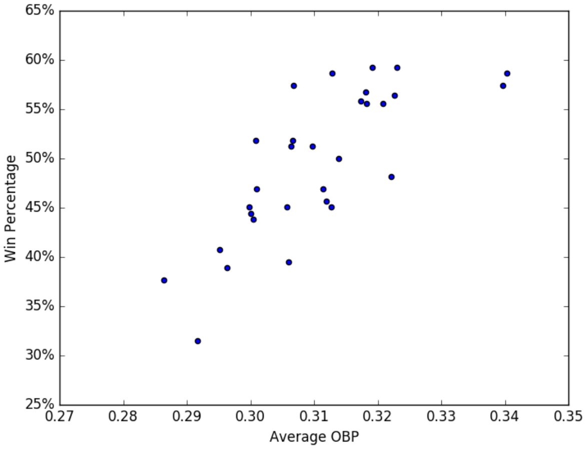


Figure 3: Win Percentage vs Average OBP for All 2013 Teams

Algorithms and Techniques

The first major machine learning technique used in this project is principal components analysis. This is necessary because without it, the number of features would be quite high. Each observation is one game, defined by 11 batting statistics for each of 20 players and 9 pitching statistics each for 4 players (the two starters and an aggregation of each team’s relief pitchers). This adds up to a total of 256 features. Reducing this number significantly should lead to both quicker processing time and a lower risk of over-fitting.

At this point the problem becomes a typical classification problem, so the right plan is to choose a model type just complex enough to capture the dynamics of the system. The four models employed here are K nearest neighbors, logistic regression, random forests, and support vector machines.

K nearest neighbors is included without much hope of success. It's a model that works best when each feature is similarly important, which seems unlikely to the case here. However, when it does produce a low-bias model, that model tends to be robust due to its simplicity. Also, it would be interesting to be able to look at a meaningful group of "similar" games that the model relied on for each prediction.

Like K-NN, logistic regression is a simple and therefore generally robust classifier. Its linear decision boundaries, on the other hand, might be more likely to train a low-bias classifier. Additionally, it is a very fast model both in training and prediction. A logistic regression is also very easy to interpret given that it generates coefficients for each feature.

Random forests and SVMs are included in case the system is much more complex than a logistic regression can deal with. Both are slow to train, and relatively opaque in terms of interpretation, especially the SVM. They are also more likely to overfit the data, though the risk of this is lower with random forests, as an ensemble model. On the other hand, the pattern matching ability of the SVM may prove essential to avoid a high-bias model.

For each of these, grid search over commonly optimized parameters is used to see how well each algorithm can do, and the one that achieves the lowest validation error is selected. For similar validation errors, the simplest accurate model is chosen, in the order of K-NN, logistic regression, random forest, and SVMs.

Benchmarks

The final model's performance will be judged based on three important benchmarks. The first is a probability of 50% for every home team, the probability one would have to guess in the absence of any information. The second, a prediction of 53.5% for every home team, reflects the average home field advantage discussed above. These naïve benchmarks should be easy to beat. The final and most important benchmark is the money-line, set by professional bookmakers and influenced by bettors in much the same way investors move the prices of financial assets. This is not widely available data, but due to my interest in sports handicapping I have been manually collecting it since the beginning of the 2015 season from a variety of online sources. On the portion of the test set for which there is money-line data, the benchmark errors are:

- 50/50: 0.6931
- Home field advantage: 0.6892
- Money-line: 0.6737

Without complex modeling of injury issues, batting and pitching rotations, manager quality, and team chemistry, it is hard to imagine a model learning enough to beat the money-line on average. Nevertheless it will be interesting to see how much of the gap between the benchmarks can be bridged with only the simple features available here.

Methodology

Data Preprocessing

While data cleaning steps (such as making sure every player has a unique name, and eliminating double headers) are described above, further preprocessing was necessary before training models. First, team totals must be removed from the dataset because they would skew the averages of additive statistics like hits and runs. The team totals are used to generate exploratory figures, and to determine the winner of each game.

Having removed team totals, the next step is to calculate bullpen statistics. The bullpen is the set of relief pitchers available to a team. They tend to pitch the last 3-4 innings of a game, or more if the starter's performance is poor. This responsibility for around half of the game's pitching is critical given what was learned in Figure 2 about the correlation between ERA and winning percentage. Unfortunately there is no way to know with certainty who a team's relievers will be for a given game, since they change from day to day. As an imperfect way to include them in the model without attempting to guess who will participate, all non-starting pitchers' performances are averaged or summed as appropriate on a given day and they are treated as "the bullpen" rather than as individual players.

Demeaning and normalization occur next. While this is not necessary for all of the models that will be trained, it improves the performance of some, and doesn't hurt the performance of any.

Then, exponentially weighted moving averages are calculated for every player and for each statistic, starting over at the beginning of each season. The spans for these averages were decided somewhat arbitrarily, and are 50 games for batting stats and 20 games for pitching stats, with a minimum of 5 observations. The window is longer for batters than pitchers because batters play every game, while pitchers play only about every fifth game. The windows are intended to be long enough to filter out noise, but not too long to miss real trends. If no modeling technique is successful with these settings, they may be altered later. In that case, however, they must be considered tuned hyper-parameters, so it would be ideal to leave them as initialized.

If a player has not appeared enough times to have any moving averages at all on a particular day, that record is dropped. If he has a partially filled set of moving averages, any missing ones are replaced by "low" values, two standard deviations above or below the mean, depending on what is "bad" for a given statistic. Presumably the statistic is missing because the player is relatively unskilled at whatever it measures.

Because PCA must be fit using only the training set, the data is split into training, validation, and test sets at this point. The 3.5 most recent seasons, 2013-2016, are assigned to the test set (the 2016 season is ongoing). The most recent seasons are used to make sure that a model trained using past data remains relevant going forward. It is also important that the seasons for which the money-line benchmark is available are included in the test set. Data from 2002-2012 are split into training and validation alternately - odd seasons are used in training while even seasons are held aside for validation. Exceptions were made for 2002 and 2012, which were added arbitrarily to the training set to increase the number of training examples.

5 PCA components are calculated for batters, pitchers, and the bullpen separately. 5 components is a radical improvement over the 9 or 11 statistics per player under consideration, and for each player type they explain over 95% of the variance. This step reduces the number of features from 256 to 110, still high but much more reasonable.

At this point the data is sorted into its final form according to the records of who started in a given game, with the principal components of the 10 away players and bullpen first, followed by the home team. If one of the teams starts three or more unknown batters or has an unknown starting pitcher, the game is thrown out for having insufficient data. If there are one or two unknown batters, their PCs are calculated using the “low” values described above. The target is calculated using batting team totals, and is set to 1 for a home victory and 0 for a home loss.

Implementation

Having performed somewhat extensive preprocessing, the modeling portion of the project was fairly straightforward. Grid search CV was used for each model to find the best set of parameters. The scorer used for the grid search was log-loss, so in the sklearn.metrics.make_scorer function, greater_is_better had to be set to False and needs_proba was set to True.

Refinement

The searches were conducted on a 2013 MacBook Pro, using all 8 cores. Even with this parallelism, there were far too many random forest parameter sets to test in a practical amount of time. It was therefore tested in stages, unfortunately ignoring some of the parameter space. Every other search terminated fairly quickly.

The following table is a summary of the results:

Model	Parameters	Values Tested	Best Values	Best Training; Validation Scores	Time Taken
K-NN	n_neighbors p	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 2, 3	1024 2	0.6859; 0.6862	13.4 min
Logistic Regression	C penalty	0.0001, 0.0005, 0.001, 0.01, 0.1, 0.5, 1, 5, 10, 100 l1, l2	0.1 l1	0.6731; 0.6824	9.3 sec
Random Forest	n_estimators max_depth max_features min_samples_split min_samples_leaf	120, 300, 500, 800, 1200 5, 18, 15, 25, 30, None log2, sqrt, None 1, 2, 5, 10, 15, 100, 200, 250 1, 2, 5, 10, 20, 40, 100	800 15 None 200 10	0.6038; 0.6809	Tested in stages. Total around 120 min
SVM	C kernel	0.001, 0.01, 0.1, 1, 10, 100, 1000 rbf, sigmoid	0.001 rbf	0.6629; 0.6818	11.2 min

As discussed above, the best model is the simplest one that achieves a good, generalizable solution. With this in mind, logistic regression is the winner. Though the validation error calculated for its best parameters is slightly higher than for either the best random forest or SVM, there are several other factors in its favor. First, logistic regression is very fast at both fitting models and making predictions, while the others train very slowly, especially random forests when there is no upper limit on the max number of features and the number of estimators is high. Second, the large gulfs between training error and validation error for the more complex models, especially the random forest, raise troubling questions about reproducibility, while the distance is much smaller for logistic regression. Next, many more parameter sets were mined to find the best random forest validation error, increasing the risk of over-fitting, though this was not the case for the SVM. Finally, a logistic regression will lead to the most interpretable model by far. Due to all of these factors, the improvement in validation error required to select a random forest or an SVM over a logistic regression would be much greater than the small performance boosts found here.

A final important consideration is if enough data has been used so that additional data would not have a meaningful impact on prediction ability. To test this, logistic regressions were fit to subsets of the training data and then applied to the validation set to track improvement. As can be seen in Figure 4, it appears that a terminal level of error has been reached.

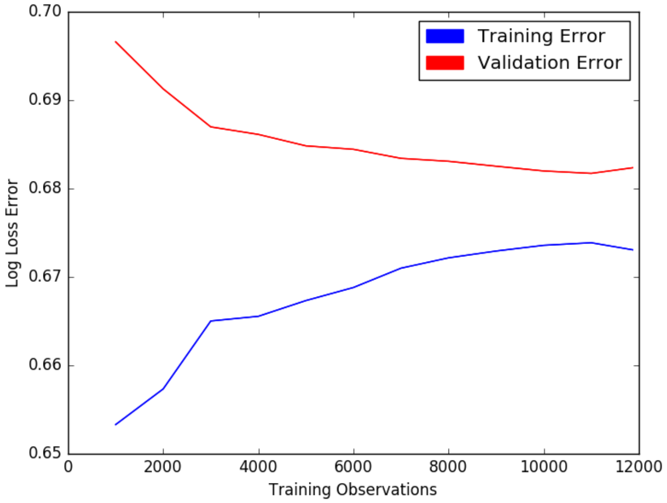


Figure 4: Logistic Regression Learning Curves

Results

Model Evaluation and Validation

Having chosen the logistic regression with C = 0.1 and penalty type of l1, based on its performance on the validation set, the moment of truth is running it on the test set to make sure it achieves comparable performance. The model's log-loss error for the test set turns out to be 0.6843, 0.0019 higher than its performance on the validation set. This number is hopefully not different enough to indicate that the grid search-induced hyper-parameter fitting has compromised the model's usefulness. Though this question cannot be finally resolved without reference to the benchmarks discussed above, it is also useful to try to make sense of the meaning of the model's learned parameters. An intuitive interpretation of the model's results could be a powerful argument for its explanatory power.

The first step here must be an analysis of the PCA performed during preprocessing – without this there is no hope that we can decipher the regression coefficients. Figure 5 shows the contributions of each feature to the 5 batting principal components. Below is a set of speculative, educated guesses for the meaning of each PC:

- PC 1: Overall player quality. Ten of eleven battings statistics indicate good performance, while only SO does not. Since SO is the only one with a positive sign, a highly negative number tends to identify a high quality player.
- PC 2: Batting style. This component identifies the difference between power hitters who may have lower batting averages but a higher number of hits, RBIs, or home runs (in more plate appearances), and strategic batters who more reliably get on base to set up scoring opportunities.
- PC 3: Defense vs. offense. Lots of plate appearances and defensive statistics, but poor batting statistics. Likely highest for catchers, who are bad batters but record putouts every time an opposing batter strikes out.
- PC 4: Defensive type. With putouts and assists opposed while batting statistics barely appear, this component contrasts players like shortstops and outfielders who are more likely to start a defensive play with an assist than finish it with a putout.
- PC 5: Pitchers. The final PC looks like an identifier for players who shouldn't be in the game at all, but still get lots of plate appearances. Terrible batting and very low defensive numbers. Pitchers are judged on different statistics, don't record many putouts or assists, and they're generally miserable batters.

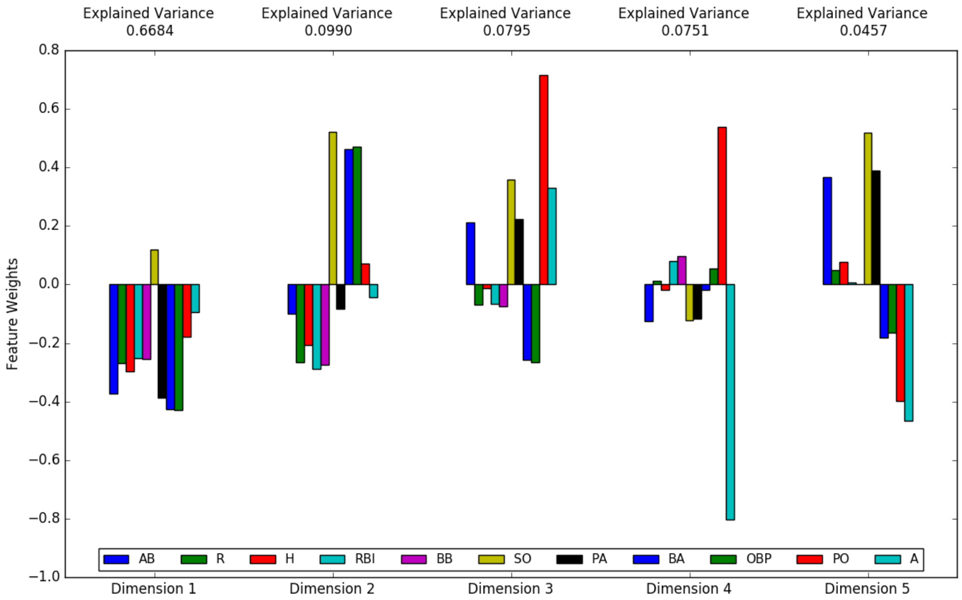


Figure 5 – Batting PC contributions

Meanwhile, Figure 6 shows the pitching PCA contributions. The explained variance is more concentrated in the first couple of principal components, which are easily interpretable.

- PC 1: How much the pitcher plays. Doesn't say a great deal about quality, as ERA is almost completely unrepresented and the weights of the other statistics are not radically different – on the other hand, bad pitchers don't generally pitch much.
- PC 2: Almost untransformed ERA.
- PC 3: Pitching quality apart from ERA. The difference between negative pitching statistics such as hits and runs, and strikeouts. A little mystifying that ERA and strikeouts are on the same side, but this should probably be understood as a consequence of pitching style rather than quality, especially because ERA figured so dominantly in the second PC.
- PC 4: Control. A precise pitcher doesn't walk many batters, and he gives up more hits as a result.
- PC 5: Dominance. High value means the pitcher throws fewer innings, fewer pitches, and fewer strikes but opponent plate appearances result in more runs and strikeouts than average. The alternative might indicate a pitcher who relies on other defenders to put batters out via caught fly balls or groundouts.

The bullpen PCA is nearly identical to the pitching PCA. The biggest difference is that its 5th PC has its signs reversed. For this reason they are not shown here.

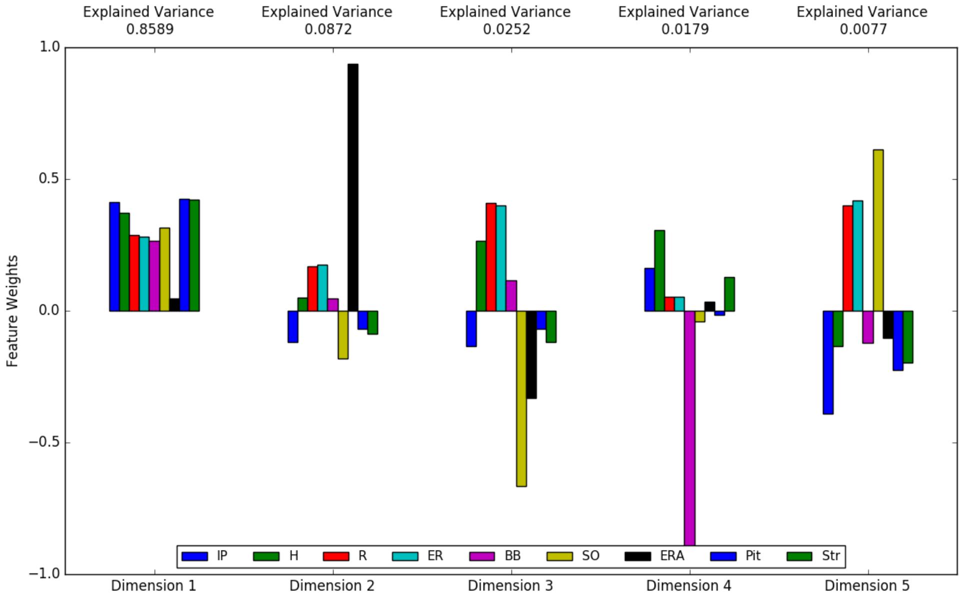


Figure 6 – Pitching PC contributions

Using our analysis of the PCA results as a framework, it's now possible to attempt an interpretation of the logistic regression coefficients. Figure 7 presents them all together in a heat map, charted by player and principal component. Blue sections indicate that a positive value in that principal component for that player increases the chances for a home victory, and red means the opposite. The signs can be bit tricky to keep straight because sometimes a "good" value for a PC is negative. For example, above we defined the first batting PC as a rough indicator of overall player quality, but the better the player, the more negative the number, so the coefficients for the home batters all show up red.

The first striking feature of the heat map is the central importance of pitching. All of the darkest, boldest colors on the map belong either to starting pitchers or the bullpen. This is not a surprise, as it fits both with the exploration discussed above and the conventional wisdom that pitching is the most important part of baseball. It's also true that the importance of pitching is divided between just two players, rather than nine for batting. Regardless, the coefficients here tell a reasonable story. A starting pitcher with a high first PC tends to stay in the game longer before yielding to lower quality relievers. If his third PC is low he is getting many more strikeouts than he gives up in hits or runs, and a high fourth PC indicates high control. The very high importance of the fifth PC is interesting, indicating that the extent to which the pitcher dominates his team's defense is a primary factor in win probability.

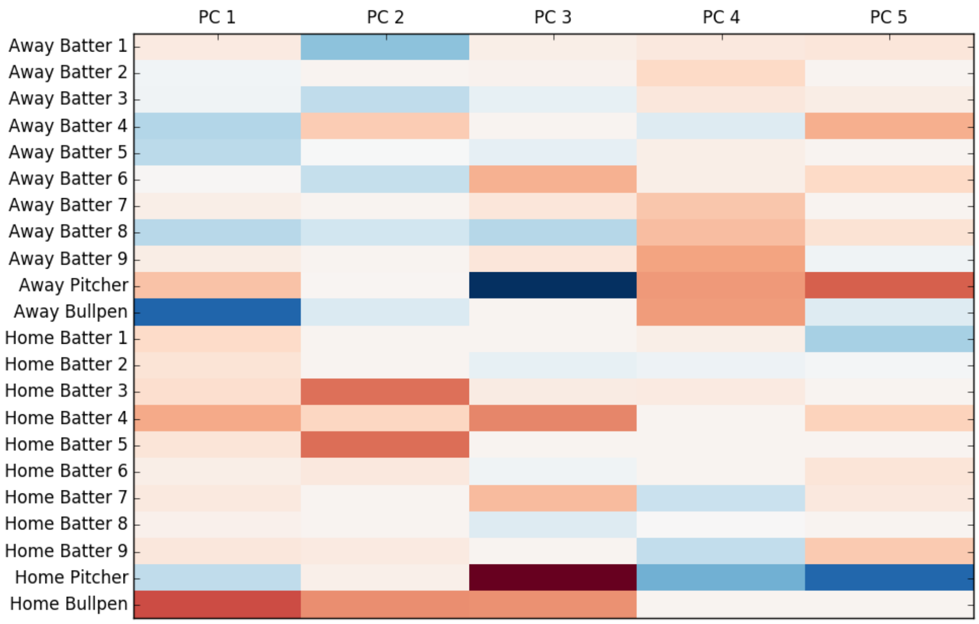


Figure 7: Logistic Regression Coefficient Heat Map

The batting coefficients are also generally reasonable. It makes sense that increasing the first principal component should have a negative impact, since it's essentially the opposite of overall skill. It's also reasonable that there is a pattern of relative importance among the batters, peaking in the middle of the lineup where power hitters are tasked with batting in runners who've already reached base. We can explain the pattern in the second PC, which we called an indicator of hitting style above, in the same way.

Meanwhile there are also patterns in the hitting coefficients that defy easy explanation. For both home and away teams, it is apparently good to stack the end of the hitting lineup with players who have large positive differences between putouts and assists. A baseball expert might be able to explain this, but a casual understanding of the game yields little.

Also, there are some examples of fairly large coefficients where both the home and away players have impacts in the same direction. For example, increasing the fifth PC for the 4th batter in the lineup, traditionally the team's best power hitter, will decrease the home team's chance of winning whether it's the home team's 4th batter or the away team's: what's good for one team is bad for the other. This seems unreasonable, but with this many parameters it's probably not likely that all of the coefficients make sense out of context. Furthermore, without a detailed analysis of standard errors, coefficient size is only a rough indicator of statistical significance, so it could be the case that the ones that make little sense are driven by noise.

Justification

The full test set errors for model and the two applicable benchmarks are:

- 50/50: 0.6931
- Home field advantage: 0.6903
- Logistic regression: 0.6843

The differences between these numbers seem small, but their significance is easily understood by considering that home field advantage is one of the most enduring ways to explain performance in a given game, and having it in the playoffs is the primary reward for winning in the regular season. The difference between including this factor and assuming that the games are just coin flips, in log-loss terms, is 0.0028. The improvement made by switching to the logistic regression is a further 0.0060, more than double the gains made by recognizing home field advantage.

On the smaller portion of the test set for which money-line data is available, the error numbers change slightly. Including the money-line error, they are as follows:

- 50/50: 0.6931
- Home field advantage: 0.6892
- Logistic regression: 0.6833
- Money-line: 0.6737

For this rather small section of the test set (1.5 seasons), the gap between the sophisticated money-line benchmark and the naïve home field advantage is 0.0155. The logistic regression finds itself in between, having bridged about a third of the gap.

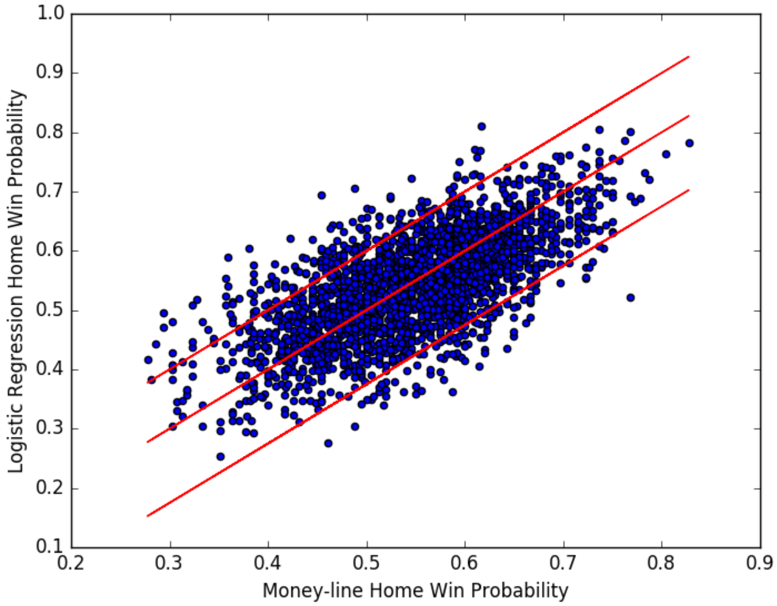
In terms of an ideal solution to the problem, the model developed here makes significant progress, but there is clearly room for improvement. Its predictions are not accurate enough to replace a skilled handicapper. On the other hand, we know that the money-line functions like any other market, and is vulnerable to inefficiencies unrelated to real value. For example, one team's fans may be more enthusiastic or numerous and place more or larger wagers, or an injury to a beloved star might send a team's money-line plummeting further than it should. The logistic regression, on the other hand, cannot be affected by factors like these. Wide divergences between the model and the money-line, therefore, may indicate possible pricing errors to look into more closely.

Conclusion

Free-Form Visualization

Figure 8 is a scatter plot comparing the money-line probabilities to the model's predicted probabilities. The middle red line is the money-line plotted against itself, and the other two red lines represent possible boundaries for the "wide divergences" discussed above. The top one is 10 percentage points higher than the money-line while the bottom is 12.5 percentage points below (the extra 2.5 points reflects the spread between the money-line values for the home and away teams, which is the amount a bookmaker earns over time). As can be seen in the plot, the model tracks the money line fairly well, with the vast majority of its predictions within 10 percentage points of sophisticated handicappers. The model does not appear to be as ready to make confident predictions as the money-line. In other words, it tends to produce higher estimates when the money-line considers a home team a significant underdog and lower estimates when the home team is a heavy favorite.

Figure 8: Money-line Predictions vs Model Predictions



Reflection

This project proceeded along the following path:

1. Manual daily collection of money-line data
2. Research on important baseball statistics
3. Finding, downloading, and cleaning data from retrosheet
4. Pre-processing data, including demeaning, normalization, and calculating moving averages
5. PCA
6. Input creation
7. Model selection through exhaustive grid search
8. Model interpretation
9. Comparison to benchmarks

Step 3, acquiring and cleaning the data, was difficult and exhausting. The dataset included more than 35,000 baseball games since 2002, so every cleaning step was fairly labor intensive. Further data issues were sometimes discovered further along the analysis pipeline, which required starting over. For example, I had no idea baseball teams sometimes played two games in a single day until I discovered duplicate records and errors while putting together game inputs in step 6. If I were to repeat the project from the beginning I would be more careful to establish unique player IDs and team IDs.

I was surprised also by the amount of time exhaustive grid search could take for certain models. I had to develop a suboptimal step-by-step approach for random forests, searching two-dimensional spaces and trying to fix parameters as I went. Even with these time savings, it was still necessary to configure the grid search function to use all of my computer's cores.

The most interesting portion of the project was the interpretation of both the PCA components and the regression coefficients. This required some reading into baseball strategy and positional tendencies, and it was rewarding to discover patterns in the data that supported certain conventional wisdoms. For example, when searching for an explanation for batting PC 3, I found out that catchers are often terrible batters. Because playing catcher can quickly ruin your knees, promising batters are given different defensive positions to play to prolong their careers. Learning this made the principal component's meaning much more understandable.

It was also rewarding to finally plot the logistic regression's predictions against the money-line and find that, by and large, they tend to agree. People sometimes think of the bookmakers' lines like sports gospel, so it's nice to be able to approximate it with a fairly simple model, using fairly simple statistics.

As discussed above, the model would not be useful as a way to automate handicapping. However, it may very well be useful as a starting point for finding games with lines that are "off," either for the bookmakers to reconsider their estimates or for bettors looking for opportunities.

Improvement

It is my belief that with the data and techniques applied here, the logistic regression solution is close to optimal. However, there is a wealth of additional statistics that, if added, could yield better results. For example, OBP is no longer considered the best measure of batting quality; now it is OBP plus slugging percentage, which is a measure of the number of bases gained per at bat. There are ways to quantify the win probability contributed or subtracted by a player in a given game and ways to quantify their performance in high stress situations. Unfortunately using this data would require either a great deal of additional processing, or finding a free source, which might prove difficult.

I would also be interested in using neural networks; unfortunately they are not supported in the current version of scikit-learn. It is possible that with the right architecture they could learn interesting nonlinear ways to combine the features.

Finally, in this project I have made no effort to include non-starting offensive players. While it would be difficult to do this accurately, replacement hitters are often better than the player they replace, so it might improve accuracy to include them. Similarly, the importance of the bullpen, according to the regression coefficients, suggests that modeling it more precisely might be worthwhile.

Baseball Stat Glossary

A: Assist – Any defensive player who touches the ball before a putout is recorded by another player gets an assist.

AB: At-bats – A player gets an at-bat every time he appears at the plate and does not walk or sacrifice himself.

BA: Batting average – A classic statistic for evaluating batters, calculated as H/AB, BA has fallen somewhat out of favor as people have realized that walking to first counts nearly as much as getting there due to a hit (nearly because other runners may not get the chance to advance).

BB: Walk (or, base on balls) – A player "walks," and is awarded first base, if the pitcher throws him 4 pitches that fall outside the strike zone, and he doesn't swing at them.

ER: Earned runs – Almost the same as Runs, but does not include errors by members of the defense other than the pitcher.

ERA: Earned run average – The most commonly accepted statistic for evaluating pitchers, ERA measures earned runs per game and is calculated as 9*ER/IP

H: Hits – A player records a hit every time he reaches at least first base without walking.

IP: Innings pitched – A self-explanatory statistic. In case of incomplete innings pitched, the number advances by outs. For example, if a player records 5.2 IP, this means he was removed from the game after having pitched 5 complete innings and getting 2 outs in his 6th.

OBP: On base percentage – OBP measures how frequently a batter reaches a base per plate appearance. Unlike BA, it includes occasions such as walks or the batter being hit by a pitch, but does not include circumstances outside the batter's control such as reaching base due to a defensive error.

PA: Plate appearances – The simpler but less common version of AB, Plate Appearances include every time the player appears to bat.

PO: Putout – A defensive player that physically records an out, whether by tagging a batter, catching a batted ball, stepping on a plate, or catching a third strike.

R: Runs – An offensive player gets a run if he scores for his team by crossing home plate. A pitcher is charged for having allowed a run if an offensive player he allowed to get on base scores.

RBI: Runs batted in – If a player's plate appearance results in a run being scored, by him or by a runner already on base, he records an RBI. There are very few exceptions. If advancing through a walk or being hit by a pitch scores a run, it counts despite not actually having been “batted.”

SO: Strike out – A batter strikes out, and a pitcher earns a strike out, when three strikes are pitched. A strike is a pitch where the batter swings and misses, or the batter doesn’t swing when the pitch was in the strike zone. A player can get his first or second strike, but not his third, on a foul, which is when the batter hits the ball but it lands outside the fair boundaries.