STATISTICAL SPORTS MODELS IN EXCEL

ANDREW MACK

STATISTICAL SPORTS MODELS IN EXCEL

ANDREW MACK

This book and the content herein are provided for educational purposes. Although the author and publisher have made every effort to ensure that the information in this book was correct at press time, the author and publisher do not assume and hereby disclaim any and all liability to any party for any loss, damage, or disruption caused by errors or omissions, whether such errors or omissions result from negligence, accident, or any other cause.

All quotes, references and screenshots from other sources are intellectual property of and copyright their respective owners and are included for illustrative fair use purposes only.

Created with Vellum

CONTENTS

TABLE OF CONTENTS

1

LET'S HAVE AT IT

> *"Fade the Detroit Pistons on the second quarter moneyline. Blake Griffin and Andre Drummond get short minutes in Q2, and the Pistons usually fall behind significantly by halftime for this reason."*

Guaranteed locks. 42-0-0 trends. Max bomb whale plays. Touts claiming 75% winning records. That's all most people in the sports betting world want. Just tell them who's going to win so they can get some action. Mindless donkey picks.

But not you.

You've purchased this book because, like me, you see the deeply rewarding intellectual challenge that sports betting represents. You know that it's likely fraught with difficulty (spoiler: it is), but you're curious about whether, with the right methods, it might actually be possible to win; to be an +EV bettor. You're here to acquire some tools in that quest.

If that describes you, you've come to the right place. I can't promise you that you'll come out the other side of this book with market beating +EV models, but I am confident that the material contained herein will make you sharper, make your journey to sports betting success faster and will probably help you to become a sports bettor with at least a breakeven expectancy. In doing that, you'll be more than halfway there.

Truthfully, I think that's the first goal every modeller should have – to develop systems capable of breaking even. If you can do that, you're beating the vigorish and consequently you are closer to being a winning bettor than you think.

Let's change gears for a moment to deal with the pink elephant in the room: why on earth would anyone write a book like this? That's a good question, and one that rightfully deserves a healthy degree of scrutiny. After all, if these models are so great, why would I bother to share them when I could wager with them myself?

Let me tell you something you already know: There are some very good books available on sports betting market analysis, market pricing and market psychology. However, there are very, very few resources available for those interested in modelling sports for the explicit purpose of betting.

The reasons for this aren't particularly complicated. First, statistics is a vast field burdened with heavy technical jargon, nuanced formulaic notation and other pitfalls to those not formally trained. It's not as though it's impossible to learn, but it's not exactly user-friendly right from the beginning. It's also not immediately clear for a novice modeller what statistical techniques and concepts to focus on. Certain techniques in statistics are at best unhelpful and at worst patently useless for the sports bettor, and it's not always easy to decipher which is which. Secondly, this knowledge gap creates a barrier to entry that serves as a competitive edge; simply stated, a bettor with a winning model is extremely unlikely to show you how it works. You can't really hold that against them (or me), but it explains the void in publicly available knowledge in this area. That being said, I think this book provides a solution to this lack of knowledge while still allowing me to preserve my current live edges.

The truth is, I have wagered with these models. I have made some money with them, but since then I've upgraded my modelling skills to move on to more advanced software and techniques. In studying for a Master's degree in Data Science (while concurrently finishing a law degree), I've learned a lot about using R, Python, Stan, JAGS, and other software platforms that can be used to create considerably more advanced machine learning sports models.

In all seriousness that's where the future of sports modelling lies. Even Marco Blume, the head of trading at Pinnacle, has come out openly saying Pinny's traders are using machine learning models deploying the Caret package in R and the Scikit Learn package in Python. Someday soon, these platforms are likely to be the only path forward for attacking major sports betting markets. For the meantime however, the Excel-based models in this book, particularly when applied to smaller markets and derivatives, can offer fairly regular flashes of positive expectancy. Discovering precisely which markets these models can still break is something I will largely leave up to you – so that when you do discover one, it won't be completely spelled out here in black and white for the rest of the betting world to cannibalize. The clock is ticking though.

Also, because I've moved on to more sophisticated modelling work on other platforms, I can share these models and ideas with readers without fear of my live betting edges being devoured by the market at large. From this standpoint, I see this book as an opportunity to give back to a community that has helped me tremendously. I have the opportunity to share with you things I wish I'd known when I started out, you get to greatly boost your learning curve and get exposed to some (hopefully) new ideas, and I don't have to part ways with a live edge that constitutes a present source of sports betting income. That sounds like an +EV proposition to me. Hopefully you agree.

Modellers starting out typically encounter three distinct challenges:

1. They aren't aware of the statistical concepts and tools that might be available to them to solve their current sports betting challenges. Also, because they aren't aware of them, they aren't sure where to look for help.
2. They may be aware of the aforementioned tools, but are unsure how to apply them. A good example of this is knowing that the negative binomial distribution is probably useful for modelling basketball scores, but getting stumped trying to apply it to a current problem.
3. Thirdly, they may be aware of how to apply said tool or concept, but unsure how to effectively implement it inside a program like Excel.

This book aims to assist with each of these challenges in turn by making you aware of some of the tools and concepts at your disposal, explaining in approachable language how they can be applied, and finally showing you how to execute them with concrete examples in Excel. This, I believe, will greatly assist most beginning sports forecasters in getting a head start on developing their own process.

Finally, there are a few caveats. You should know that I am not a professional or classically trained statistician. I’ve pieced together all the insights in this book from a combination of unbridled enthusiasm for sports modelling and an embarrassingly long history of trial-and-error. While I’ve made every effort to be statistically rigorous in my work here, shortcomings in my methodology are certainly a possibility. Any errors or omissions are entirely my own, and should you discover one I would welcome your constructive corrections.

Throughout the book I make several recommendations and references to other works and individuals whom I regard highly in the sports betting community. I have no affiliate links anywhere in this book and have not been compensated in any way for these references, so you can rest assured that my suggestions are intended to genuinely assist you and contain no conflicts of interest.

My apologies in advance to my more mathematically sophisticated readers if you find that the beginning chapters progress slowly. Since the entire aim of this work is to make statistical sports modelling techniques more accessible for the beginner, I must ensure that certain fundamental elements are discussed at a reasonable length so as not to leave anyone behind. I promise you we’ll be up to speed shortly.

Now, onward. As my former Welsh electrical foreman used to say: *“Right lads. Let’s have at it.”*

2

Excel is a fairly flexible software platform, but statistically weak on its own. It needs a little help. In order for us to get the maximum utility for our model making purposes, a few additional free add-ons are necessary. The first is the Data Analysis ToolPak, and the second is the Real Statistics in Excel add-on. There are additionally a few paid add-ons that can make your life easier too. Let's cover them quickly.

Data Analysis ToolPak

This is a very basic Excel add-on for statistical analysis which will provide us with the ability to perform linear regressions, plot histograms, perform correlation matrix analysis, and generate random numbers for Monte Carlo simulations, all of which will be explained as we progress through the book. If you already have it installed, you should see in under the "Data" tab in excel at the top of your screen. If you don't have it installed, you'll need to load and activate the package. The exact method may differ depending on your version of Excel, but generally:

1. Click the file tab, click options, then click the Add-Ins category.
2. In the Excel Add-Ins box, make sure "Analysis ToolPak" is checked, then click OK.
3. You should now be able to see a Data Analysis Option under the Data tab.

Real Statistics in Excel

The Real Statistics Resource Pack was created by statistician Charles Zaiontz and can be found at his website[1]. It significantly expands the number of statistical functions we can perform in Excel. Specifically for our sports betting purposes, it will allow us to easily perform logistic regressions, polynomial regressions, ridge regressions and a number of other important tasks. Download it for free and load it into you version of Excel by following the steps that were laid out for the Data Analysis ToolPak.

Solver Add-In

Solver is a free optimization engine for Excel that is crucial for a number of the models that we'll be covering in this book. It will allow us to perform a couple of different model optimization techniques including maximum likelihood estimation and ordinary least squares minimization. It can be loaded and installed using the instructions found above for the Data Analysis ToolPak.

StatAssist: EasyFit

All of the add-ons mentioned before this one are free and absolutely necessary to get the most out of Excel. EasyFit is one of the few paid add-ons that is incredibly useful and won't break your bankroll. While it's not required for following along with this book, it helps us to fit distributions which is very important for sports modelling. I've done most of this exploratory work already so you can simply take my word for it with regards to the distributions we'll be discussing, but this is an additional add-on I use regularly. It can be found at the MathWave website[2].

XLStat

XLStat is a pricey add-on, and not at all necessary for our modelling work. However, it can certainly make your life easier and takes Excel's statistical functionality even further by providing higher level tools like principal component analysis as well as a number of advanced modelling and data exploration tools. I wouldn't necessarily recommend purchasing it due to its high price tag, but it can be useful for sports modelling in Excel. It does offer a free trial and can be found at the XLStat website[3].

Solver Data Mining

This add-on is the Lamborghini of statistical tools for Excel. Quite simply, it maxes out the statistical functionality of Excel by offering many advanced machine learning tools in a simple point-and-click interface. It’s also extremely expensive, with a one year software license costing around $1,000. With functions that include neural networks, naïve bayes algorithms and decision trees, it packs some serious horsepower. While some of these tools are very nice to have, you should probably consider upgrading your skills to either R or Python if you want these additional features. A few textbooks and some YouTube videos can teach you how to use these advanced machine learning functions for free in a program like R, provided you have the time and inclination to teach yourself. That being said, this could be considered the ultimate Excel modelling booster. It has a free trial period and while you won’t need it for following along with this book, it can be found at the Solver website[4].

I think that sufficiently covers preparing our Excel platform for the work that needs to be done. Let’s proceed into what I consider to be the nuts and bolts of basic model building in Excel. Make sure you’ve got the **Solver**, **Data Analysis ToolPak** & **Real Statistics in Excel** add-ons loaded up, and we can move on.

[1] http://www.real-statistics.com/free-download/real-statistics-resource-pack/

[2] http://www.mathwave.com/easyfit-distribution-fitting.html

[3] https://www.xlstat.com/en/

[4] https://www.solver.com/xlminer-platform

EXCEL MODELLING TOOLBOX

> *“It would be very remarkable if any system existing in the real world could be exactly represented by any simple model. However, cunningly chosen models often do provide remarkably useful approximations. For such a model there is no need to ask the question "Is the model true?". If "truth" is to be the "whole truth" the answer must be "No". The only question of interest is "Is the model illuminating and useful?"*
>
> — GEORGE BOX

This section of the book is intended to expose you to some basic ideas you can use to strengthen your modelling arsenal. While it is by no means a complete guide to modelling or statistics, it is my hope that some of the concepts I’ll share will add a few new tools to your betting toolbox. This section will also be a brief introduction to some of the techniques we’ll be using later on so that you’ll already be aware of them by the time you see them used in a model.

What is a Model?

A model is a set of assumptions we have about a particular dataset. While these assumptions are necessary, they are also inherently flawed. This means that all models are flawed to a greater or lesser extent. However, since sports betting is a competition of relative skill amongst the market, as long as our models are more useful (read: less flawed) than the relative competition, we can win with them by taking advantage of identifiable mispricing.

Moving from the abstract to the practical, a model for our purposes is a basic sports betting tool where we will provide inputs and through a series of functions we will derive an output useful in our search for market value. What kind of outputs are we looking for? Usually we want a “number”, by which I mean an expected margin of victory, an expected game total score, or an expected point spread. We then convert this number (our expectation) into a probability by applying an appropriate probability distribution, and subsequently find value by comparing this probability to the implied probability of the current market price for the wager in question. The basic modelling process thus looks like this:

- Inputs
- Functions
- Expectation
- Probability Distribution
- Price Comparison
- Value Discovery

We’ll deal with each element of this process in due course, but it’s helpful to view the entire concept like this. If nothing else, it reminds you that generating an expected “number” on a game is only one link in the chain.

Basic Model Structures

Model structures usually define the relationship assumptions we have about our inputs. As an example, the most basic structure (or relationship assumption) for a sports model imaginable would be:

```
[[Team 1 Factor]-[Team 2 Factor]]
```

This is a very simple structure but demonstrates an assumption that the relationship between the two teams in this case is one of relative strength differential. Once we calculate a differential, we can move on to mapping that difference onto a target outcome variable we can actually use. However, what if the game, as is most often the case, isn’t played on a neutral site? We would be wise to factor in some element for home field advantage [HFA] which is a feature of most modern sports:

```
[[[HFA]+[Home Factor]]-[Away Factor]]
```

As you can see again, this is nothing special. It does however provide us with a slightly more nuanced set of model assumptions that should help us produce a more useful result. What if we want to make the relationship assumption that a team’s strength largely depends on the interplay of strengths and weaknesses between the two teams competing? If we wanted to move forward with this new assumption, perhaps we would like to separate offense [OFF] and defense [DEF] like this:

```
[[[HFA]+[HOFF]+[ADEF]]-[[AOFF]+[HDEF]]]
```

Here we’ve accounted for some degree of interconnectedness between the differential of the two teams in a slightly more complicated way than our previous structure. The take away from this I hope is that the way you structure your model contains assumptions about the relationships between the ratings, data or metrics you are including. Maybe multiplying the factors would be better? Perhaps we should compare team strength to the home and away averages in their respective league? Perhaps these metrics would be better expressed as a percentage of the average? You can find modelling opportunities in the data via new relationships, but also in model assumptions as well. I would encourage you to experiment with different configurations based on your own ideas as we proceed through the book. You never know what you’ll find, and some of my best models came from trying one-off ideas that popped into my head. You won't know that something doesn't work until you've tried it and tested it. Deciding in advance what works and what doesn't is a game for talking heads on sports television. Modellers prefer to find out for themselves.

Prediction Ceilings & Model Benchmarks

Imagine in some parallel universe there exists a genie with two sports models and you are allowed to select one of the models to use for your own wagering indefinitely. One of the models maps the underlying data at approximately 58% and the other maps the underlying data at around 97%. Neither model has been backtested, and you may only select one.

Which model would you pick?

This is a silly example, but I assure you there is a point on the horizon. Many novice modellers might hastily select the model with higher percentage, but there is an underlying assumption in doing so that we should question. The question to ask yourself is: "to what extent is the underlying sport knowable?" As in, once we remove the effects of random variance and chance from the game, what would maximum statistical predictive prowess look like?

The point is that without knowing the prediction ceiling for the sport in question, it's extremely difficult to know which of the two genie's models to choose. Reality check though: no model will ever predict at 97%, genie or no genie! (The 97% model is almost certainly overfitted and tracking noise rather than signal.)

Setting realistic expectations for your models is important because it prevents you from being enticed into overfitting your model in an attempt to improve its in-sample performance. If you knew when you started that sport X has a prediction ceiling of ~62%, then you'd be rightly suspicious of a model (or a tout) that was allegedly performing at 97%, or even 71%. So naturally, it makes sense for us to discuss prediction ceilings in model making, as well as benchmark numbers and reasonable expectations.

Unless you've miraculously discovered Laplace's Demon[1], there is a limit to how well even the best models in the world can predict a given sport. The sharpest markets are very close to that ceiling, and combined with the bookmaker's commission this helps to explain why they are so hard to beat. This is also why softer markets *are* soft; the aggregate market is not yet prohibitively close to the sport's statistical prediction ceiling. Knowing this presents a number of opportunities, because it gives us a benchmark to compare predictive ability and helps us to decide where to focus our modelling efforts. I'm sure you'll agree that it makes the most sense to spend the majority of our time trying to beat markets with a soft "attack surface" (to borrow a phrase from Ed Miller).

Determining the Prediction Ceiling

Let's start at square one: a simple coin flip. Any model worthy of avoiding the recycle bin must surely be able to make predictions better than a random chance coin toss. That's pretty much a no-brainer. Let's try to be a little more ambitious than that. What about a model that just picks the home team every time? How often would such a dunce model be expected to win? It depends on the sport, but generally as was found in this research paper[2]:

Home Team Win % vs. Equal Opponent

- NBA 62%
- NFL 58.9%
- NHL 55.5%
- MLB 54%

That gives us a something to work with. Any model worth its salt should be able to do better than these percentages, at least for raw classification accuracy. How about a model that blindly picks the team with the better record?

Stronger Team Win %

- NBA 67%
- NFL 64%
- NHL 57%
- MLB 56%

As you can see, for many major sports the bare minimum prediction accuracy is rather high. Let's explore another way to conceptualize this using team winning percentages over the course of a season.

Standing Results Ceiling Estimation

Using a method I borrowed/extrapolated from the very talented Tom Tango[3] and Phil Birnbaum[4], we can estimate the amount of variance in a league's standings that are accounted for by skill. The idea behind this is that a model can only reliably predict signals related to skill, so variation that is a result of random chance is not something we can forecast with a model (after all, its random!). Therefore, the lower the percentage of variance that is related to skill, the harder (more random) a sport should be to model and predict. Let's start with an easy to understand formula:

> Variance (observed) = [[Variance (Skill)] + [Variance (Randomness)]]

That's pretty straightforward. We're simply saying that anything not accounted for by skill is the result of randomness. Let's now go and collect a sample of the winning percentages from all teams for a period of several years and then calculate the standard deviation of all winning percentages using the STDEV command in Excel.

Variance due to randomness in a sport where there are no ties would simply be (0.5*0.5)/(# of games in a season). We can then plug all of that into our formula and work out an answer.

Let's try an example with the NBA. In the 2019 season, the standard deviation of winning percentages amongst all teams was 0.146707. The variance due to randomness would be (0.5*0.5)/(82 games), giving us 0.003049. From here we manipulate the formula to tell us about the variance due to skill:

Var (Skill) = Var(Observed) – Var(Randomness)

Variance (Skill) = (((0.146707)^2)-(0.25/82))

Variance (Skill) = 0.021523 – 0.003049

Variance (Skill) = 0.018474

To convert this to the percentage of the standings that are explained by skill we proceed as follows:

Skill = ((0.018474/((0.146707)^2))

Skill = 85.83%

That's a pretty skilled league! This is also a decent estimation of our general prediction ceiling for this sport. It tells us that there is a lot of room for a model to improve beyond simply picking the home team or the team with the better record. The process works the same for most other sports with binomial (win/loss) outcomes. The inverse of the skill percentage also tells us how much randomness we might expect in our game results on the aggregate. We could further use this randomness percentage as a form of confidence interval around our assessment of each team's results.

I worked out the numbers for a few other leagues to save you some time. This method would leave us with the following theoretical prediction ceilings. I don't know if this is entirely accurate, but I think it paints a decent picture of how well we can expect to model a given sport by mapping skill signals:

- NBA: ~86%
- NFL: ~79%
- NHL: ~62%
- MLB: ~62%

Please note that I'm not stating you'll hit these numbers, and since we haven't discussed pricing yet this isn't an indication of betting winning percentage, profit, ROI or anything along those lines. This is simply an assessment of how much about a given sport we can expect to accurately map given a maxed-out (excellent) model. These can be useful for comparing to bookmaker lines based on closing odds to see the level of competition you're up against. If your preferred sportsbook was extremely close to the theoretical prediction ceiling, then we could reasonably expect to be in for a tough grind in trying model our way to an edge. This is a persuasive argument in favour of sticking to smaller markets until your models have the firepower to take on the sharpest minds in the game.

So far we're off to a really good start in understanding what the prediction ceiling for a sport might be. It's also a friendly reminder than major markets are tough to beat with modelling work. However there's always more to the story. In this case, straight up classifying accuracy isn't the only or even the preferred comparison benchmark. Let's dig deeper and discuss some metrics you can use to gauge your models.

Benchmarks: Is Your Model Any Good?

There are a number of ways you can compare your model to the benchmark you're trying to beat, which in our case is your preferred sportsbook. We've lightly touched on accuracy, which is simply the ability to classify winners and losers regardless of price.

I'm sure you can agree though that since we *bet prices rather than teams*, we need something more robust. I can remember asking

Tony Corke (a brilliant modeller who runs the Matter of Stats[5] website) about these different model benchmarks a while back and he kindly helped to shed some light on the different options available. Here are a few metrics you might consider:

R Squared [RSQ]

RSQ is a measure that tells you what proportion of variation in your target outcome variable is explained by your model. In linear regressions, this would essentially answer the question “how much variation in Y is explained by X?” This makes sense when you see the formula, which is [1-(Explained Variation/Total Variation)]. It’s nice to beat the book on an RSQ basis, but it usually isn’t the best way to tell that you’ve got a winning model. Since modelling against a book line is about being “less wrong” more often than not, error focused metrics are normally a better choice. This benchmark can be called in Excel using the RSQ command.

Root Mean Square Error [RMSE]

This is the first general go-to error metric that measures the difference between predicted and observed values. The reason it includes a squared component is to aggregate the magnitude of prediction errors into a single measurement. Without that, positive and negative errors would effectively cancel each other out, leaving us with a very unhelpful metric. Simply stated, RMSE is the square root of the average of the squared errors between our predictions and the observed results. This is a good place to start, but you should know that this benchmark is sensitive to outliers in data. Very large single prediction errors can leave you with a level of overconfidence in your model’s utility. There is no single command for this metric in Excel, but it can be calculated by first creating a column [“Range A”] that calculates the difference between your prediction and the observed results [Observed- Predicted] and then applying the formula:

```
=SQRT(SUMSQ(Range A)/COUNTA(Range A))
```

to Range A in an adjacent column. I'll walk you through this and other benchmark formulas later in the backtesting chapter of this book. There will be pictures to follow along with.

Mean Absolute Error [MAE]

Mean absolute error [MAE] is a metric that should be of interest to you if you are modelling point spreads, totals or any type of binary over/under scoring wager other than fixed odds moneyline winners. If you're trying to beat the point spread, it's handy to know how your model does versus the book line by comparing apples to apples. MAE can help you with that. It can also tell you how accurate your sportsbook's line is on a point spread, giving you a benchmark that you must beat to have a reasonable expectation of winning. Technically MAE is the absolute average difference between predicted and observed values, but all you really need to know is that this is a better indicator of performance for point spread and total bets than RMSE. Like RMSE, there is no single command for MAE in Excel. However it can be calculated easily. First, as with RMSE, create a "Range A" column that calculates the difference between observed and predicted values. Then, apply to Range A in an adjacent column the following formula:

```
=SUMPRODUCT(ABS(Range A))/COUNT(Range A)
```

Log Loss

Log loss is a very helpful tool for measuring prediction error with regards to probabilities, which is a primary concern for sports bettors since we are competing against the bookmaker's probabilities with our own probabilities. A lower log loss number means that your model was making better probabilistic predictions. This is a metric you should strongly consider using. It is incredibly helpful for gauging the utility of your model's probabilistic forecasts. There is no single command for this function in Excel, but it is again fairly straightforward to calculate. First, we need a column that outputs the result of the game as either a 1 or a 0. 1 indicates a win, while 0 indicates a loss from the home team’s perspective. In a column adjacent to our probability predictions for each game we apply the formula:

```
=(Result)*LN(Prediction)+(1+Result)*LN(1-Prediction)
```

Let’s once again call this “Range A”. After populating this column down for each game, in an adjacent column apply the formula:

```
=SUM(Range A)*-1/COUNT(Range A)
```

Brier Score

The Brier Score metric was originally applied to things like weather forecasts and can measure the accuracy of probabilistic predictions for games with mutually exclusive binary outcomes. The lower your model’s Brier Score, the better the predictions are generally. I don’t think it’s as good as log loss, but it is very easy to calculate and can be helpful for comparing models. Technically, it rates the calibration of a set of predictions, but one problem with this metric is that it requires a lot of trials (games) to be reasonably accurate. We once again need a column with the binary outcome of a game indicated as a 1 for a win and a 0 for a loss. In our hypothetical “Range A”, simply calculate the predicted value minus the observed value squared as (predicted-observed)^2. Do this for all games in the dataset. In an adjacent column, calculate the Brier Score by using the formula:

```
=SUM(Range A)/COUNT(Range A)
```

Out of Sample Profits & ROI

Finally, whether your model actually makes money is naturally the gold standard of benchmark metrics. The most straightforward way to do this is to take to "paper trade" your predictions, calculate the profits and losses on an out of sample test you would have expected, and see how your model would have fared. This will be covered in the backtesting chapter.

Sports Betting Distributions

Probability distributions are a fundamental concept for modelling sports outcomes. Wikipedia[6] defines them technically as:

> "…mathematical functions that [provide] the probabilities of occurrence of different possible outcomes in an experiment. In more technical terms, the probability distribution is a description of a random phenomenon in terms of the probabilities of events."
>
> — WIKIPEDIA

Selecting the appropriate distribution for your target outcome variable is important for getting the forecasted probabilities correct. Unfortunately, descriptions of various statistical distributions found in most textbooks and academic literature can be a bit hard to understand. This is really a shame, because statistical probability distributions are fascinating. Provided that you understand the type of data you are working with and it fits the shape and assumptions of a given dataset, you have dozens of different distributions at your disposal to model data and derive probabilities. In sports betting, there are a few really key distributions that you should be aware of for modelling purposes. First though, let's talk about the basic differences between two broad categories of data: continuous and discrete variables.

Continuous Variables

Continuous data is simply data that can take any form from positive to negative infinity on an integer scale. This includes rate statistics like shooting percentages, winning percentages, estimated point spreads and any other metric that can be a positive or negative decimal number. This type of data requires distributions that are generally separate from discrete distributions as we'll discuss momentarily. For example, the normal distribution is a continuous distribution and is best suited for modelling continuous data like point spreads or expected rate statistics. Total game scores, on the other hand, are always whole positive numbers and thus are usually a type of discrete data. This is important to understand, but there's no need for us to go much more in depth about it than that. Just know that if the outcome variable you're trying to model is continuous, you should be using a probability distribution that is appropriate for the data.

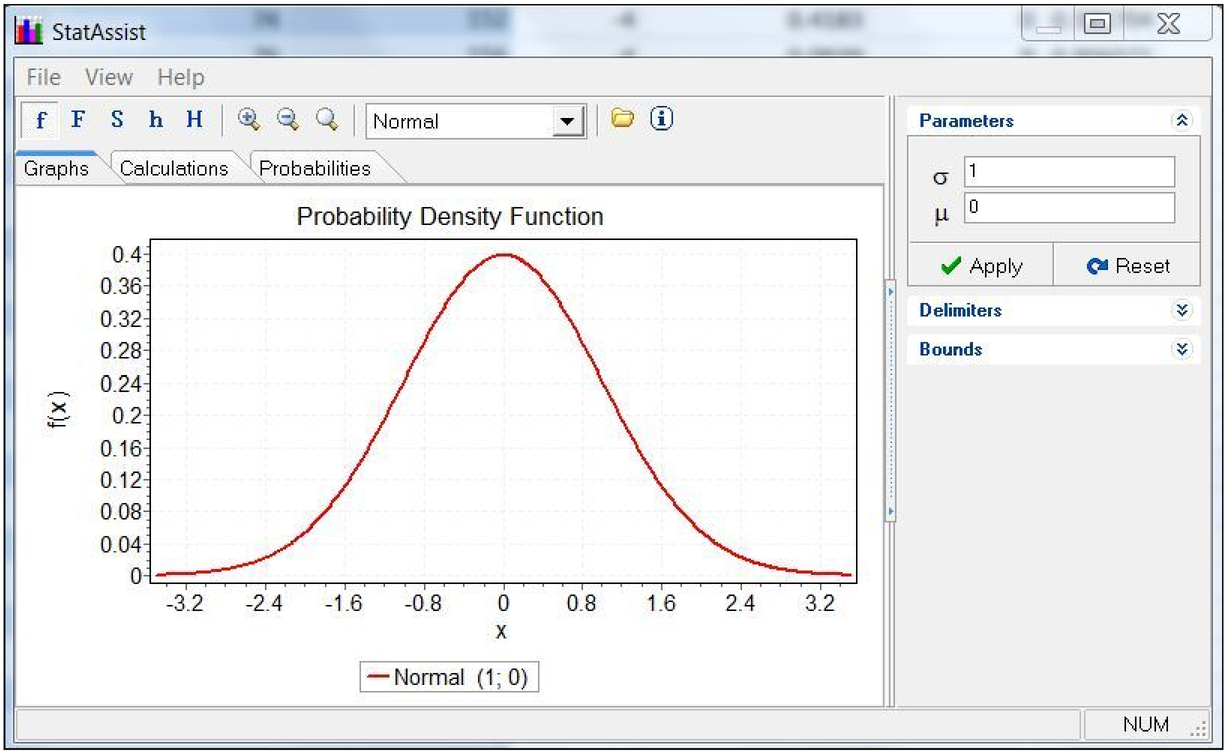
Discrete Variables

Discrete variables or count data are typically whole numbers or units that are counted one at a time, like goals, total game scores or team points. These types of outcome variables require discrete distributions to model their probabilities appropriately. So to continue our game scores example, you might have tried to calculate the probability of an NBA game total going over or under a certain line using the normal distribution before. This isn't quite right – what is needed is a discrete distribution like the negative binomial distribution instead. Again, we don't need to go to in depth on this as long as you understand that the type of distribution you select should be based on your data type, and then subsequently fitted according to which distributions appear to satisfy the data's characteristics.

Selected Sports Betting Distributions

I'd like to briefly introduce you now to a few of the more important distributions in sports betting and some of the suggested uses for each. This is far from comprehensive, but it will give you a few new ideas and tools to add to your modelling toolbox. Let's not worry too much about the finer technical details for now – I'd just like to describe them for you and make you aware of them. We'll use them in Excel later on.

The Normal Distribution



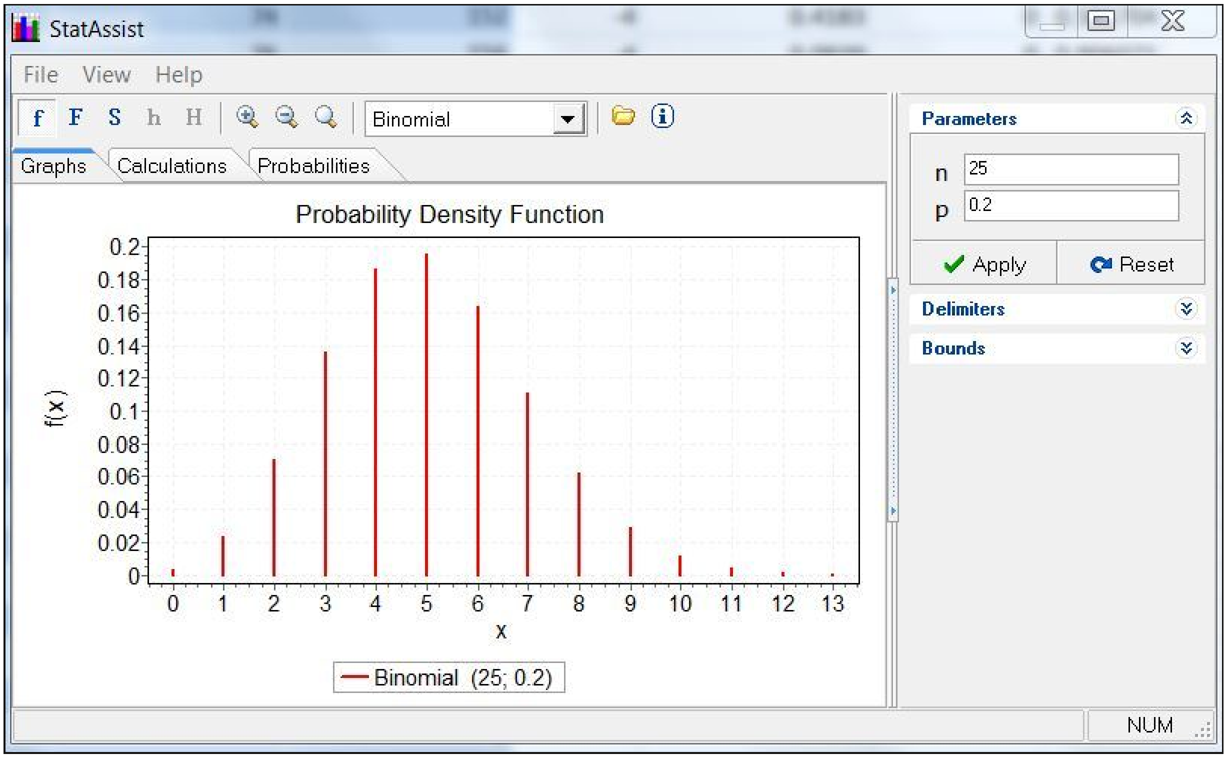
Normal

σ 1

μ 0

Most of you are already familiar with this continuous distribution. It is the bell shaped curve that an incredible number of datasets fall into nicely. It is also the one distribution you are most likely to misuse, because it is not appropriate for every situation. Its use by beginner modellers is much more a function of its ease of use than its appropriateness. I recommend using it as a reasonable approximation of margins of victory and point spreads as well as rate statistics like expected shooting percentages. It's not bad for these purposes, although there are other, slightly more complicated continuous distributions that can often do a better job (like the Weibull[7] or LogNormal[8]). If you're trying to map team points, game totals or other types of discrete data, remember to consider a better-suited distribution. Don't just fire away with this one because it's easy to use!

The Binomial Distribution

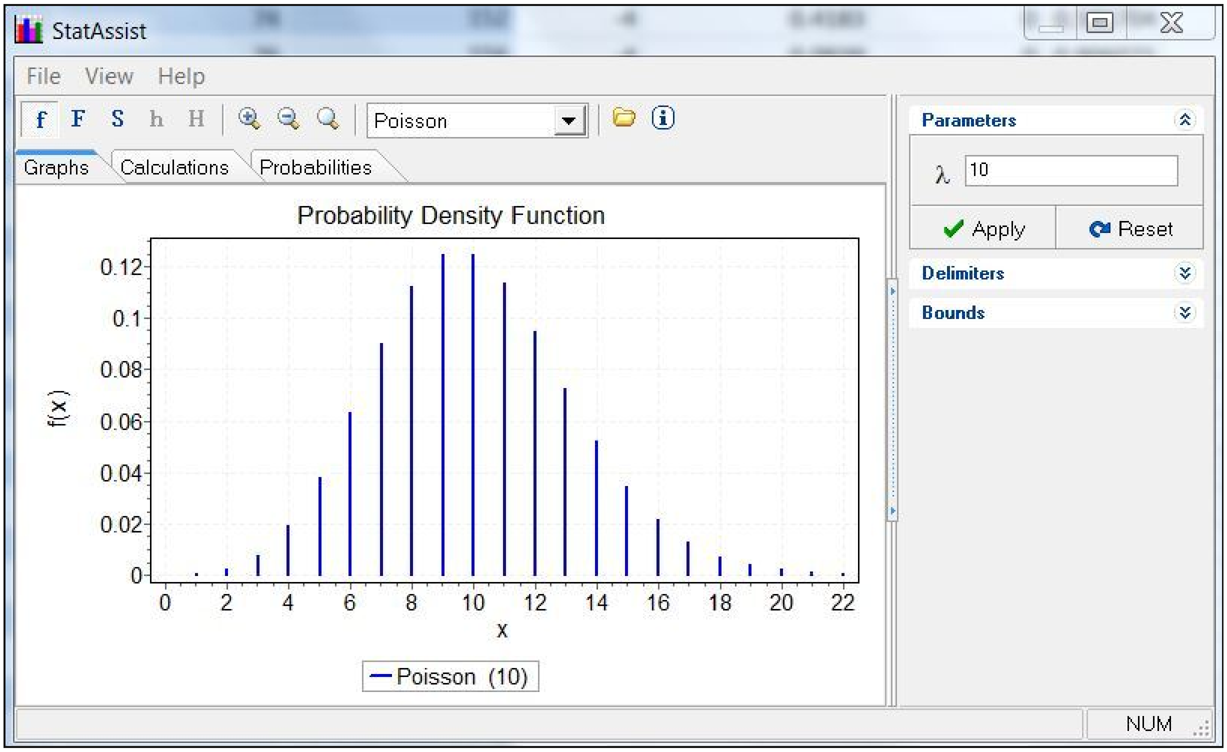


n 25

p 0.2

The binomial distribution is a discrete distribution that maps binary success/failure trials. I don’t use it much, but it can be useful in analyzing betting records and expectations in 2 outcome wagers. It’s also likely there are a number of uses for this distribution that I’m not aware of. We won’t be using it much in this book, but you should know that it exists and is definitely worth looking into.

The Poisson Distribution

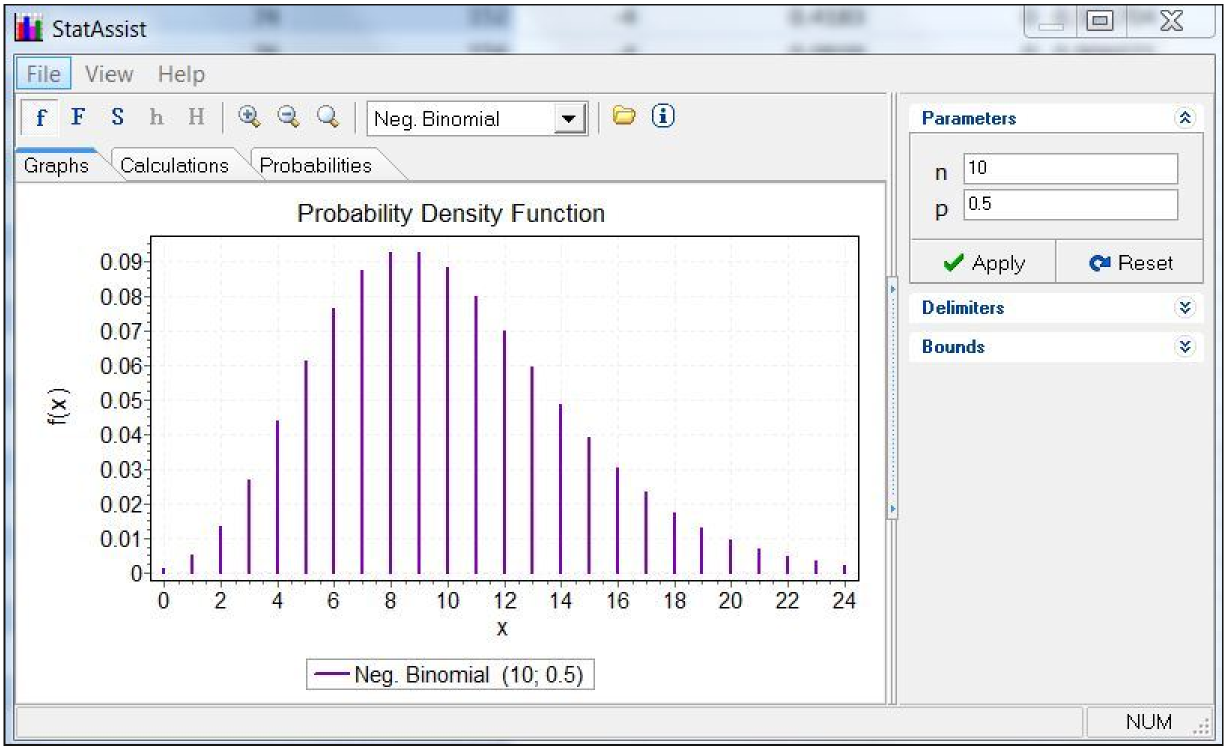


Poisson

λ 10

This is a discrete distribution that is very well known to the sports betting world – primarily because of soccer, hockey and certain prop bets. One of the key assumptions behind this distribution is that the mean (or “lambda”) must equal the variance, or be very close to it. Many people make the mistake of taking this for granted, which can lead to unnecessary modelling errors. Poisson events must also be rare, with a low chance of success despite a high number of opportunities, and counted one at a time. If these assumptions hold reasonably well, this is likely the distribution you’ll want to use. Also of note is the zero-centered or zero-inflated poisson, which is adjusted to factor for lower scoring sports like soccer.

The Negative Binomial Distribution



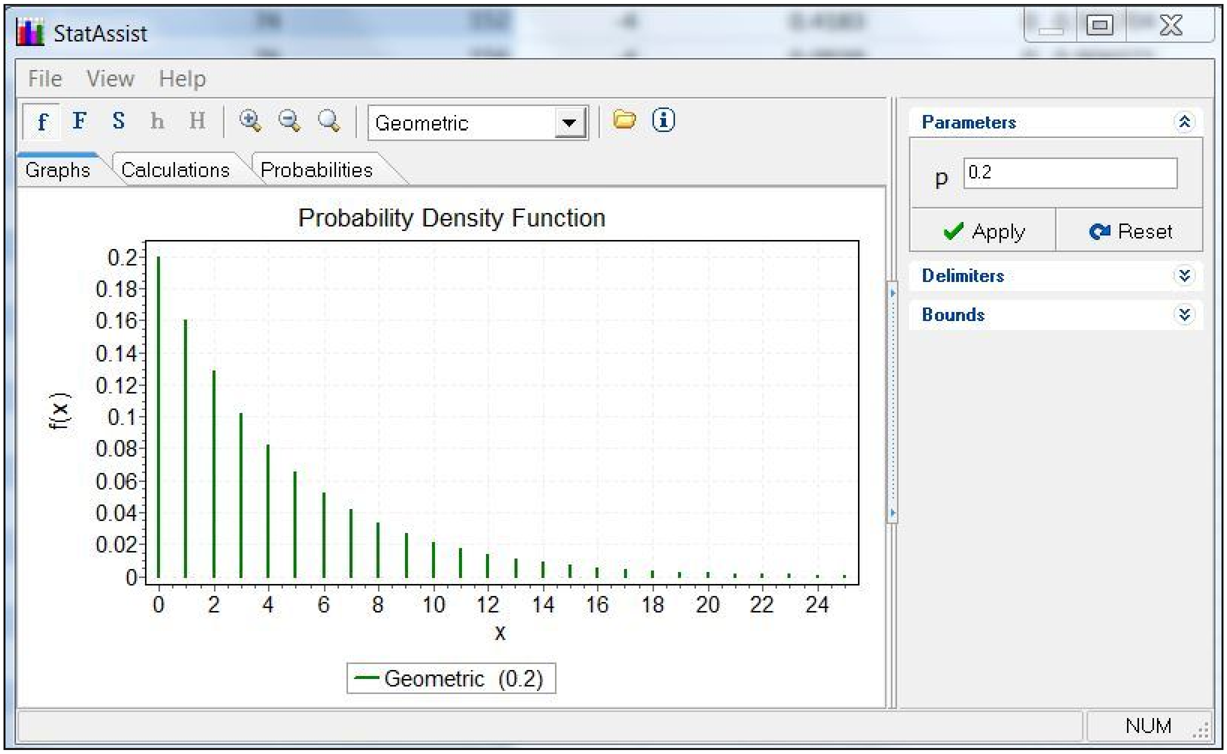
Neg. Binomial

n 10

p 0.5

This discrete distribution is a hidden gem. It's largely avoided because it can be awkward to use at first. It's classically explained as the number of successes in a set of Bernoulli trials before a failure occurs, which can lead to some confusion. That being said, you should be aware that many types of sports betting data can be mapped quite well with this distribution. It's handy because the mean and the variance are allowed to diverge from each other – which means that it features a bit more flexibility than the Poisson distribution in this respect. You might even think of the Negative Binomial like an overdispersed Poisson[9], where the variance exceeds the mean. Good news: I've put together a Negative Binomial distribution function in Excel that you can use without knowing too much more about it. If you're modelling basketball totals, team runs scored in baseball, or certain kinds of prop bets, this is a distribution you'll want to consider.

The Geometric Distribution

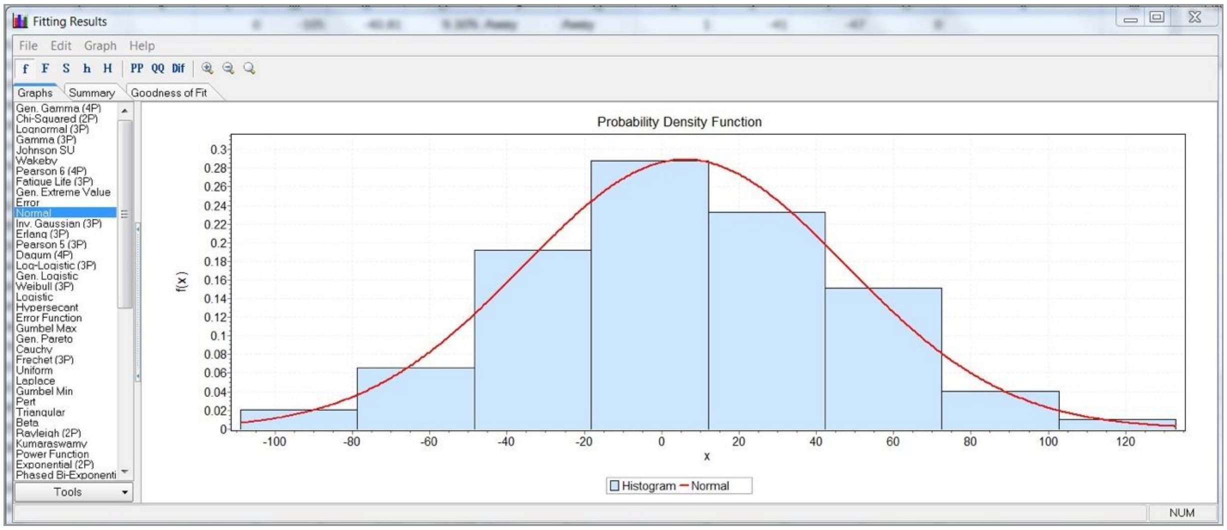


p 0.2

This distribution is slightly less useful than some of the other discrete distributions mentioned, but can come in handy every now and then. It's included more for completeness here than utility. I haven't found a lot of applications for this distribution yet, but you should be aware that it exists as a possibility for modelling discrete data.

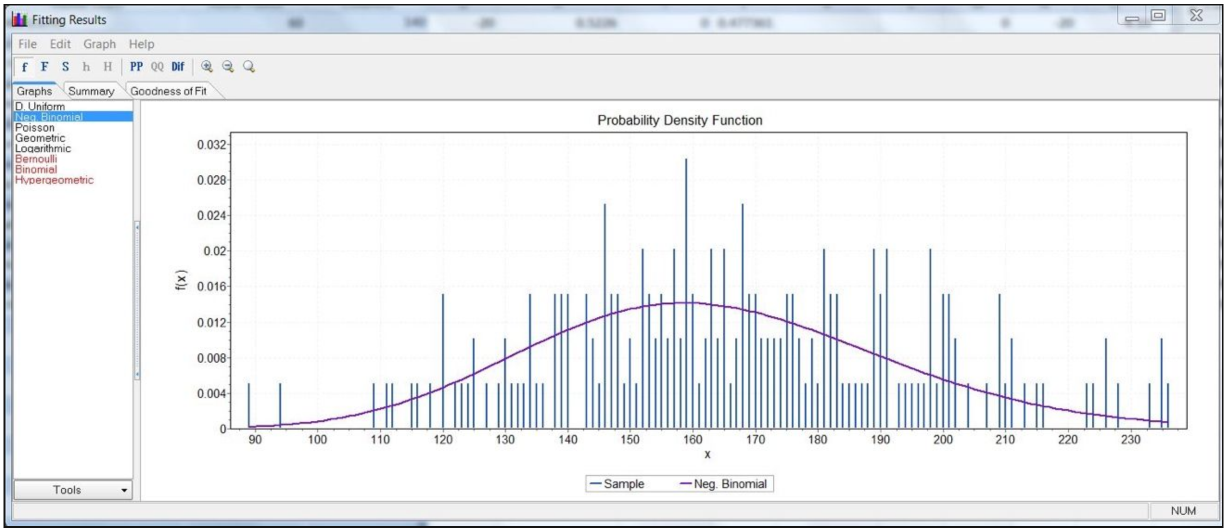
Choosing a Distribution

Now that we've gone over a few of the distributions that can help us to model sports, let's take a look at some examples. I'll run a margin of victory dataset from the AFL through the EasyFit Excel add-on to show you what it looks like. EasyFit is great for tasks like this, as it gives us a quick visual representation of our data which can help us make decisions about which distributions to use on our target outcome variable. Take a look at this AFL home margin of victory data:



As you can see from the histogram, the normal distribution isn't a perfect fit, but it's not bad for this continuous data. We can gauge the goodness of fit using the Chi-Squared test, the Anderson Darling test, or the Kolmogorov Smirnov test, and EasyFit makes this straightforward with its "Goodness of Fit" tab outlining how well each distribution appears to be suited for the data. I usually stick with the Anderson Darling test column because it tends to be the most robust against outliers in the data. Here, the Anderson Darling test indicates that the normal distribution is ranked 7th out of 40 available distributions. Not too shabby, although it appears a Johnson SU or inverse Gaussian might be a better choice.

Let's take a look now at total points scored for each AFL game. This is discrete data, so we'll tell EasyFit to only consider discrete distributions:

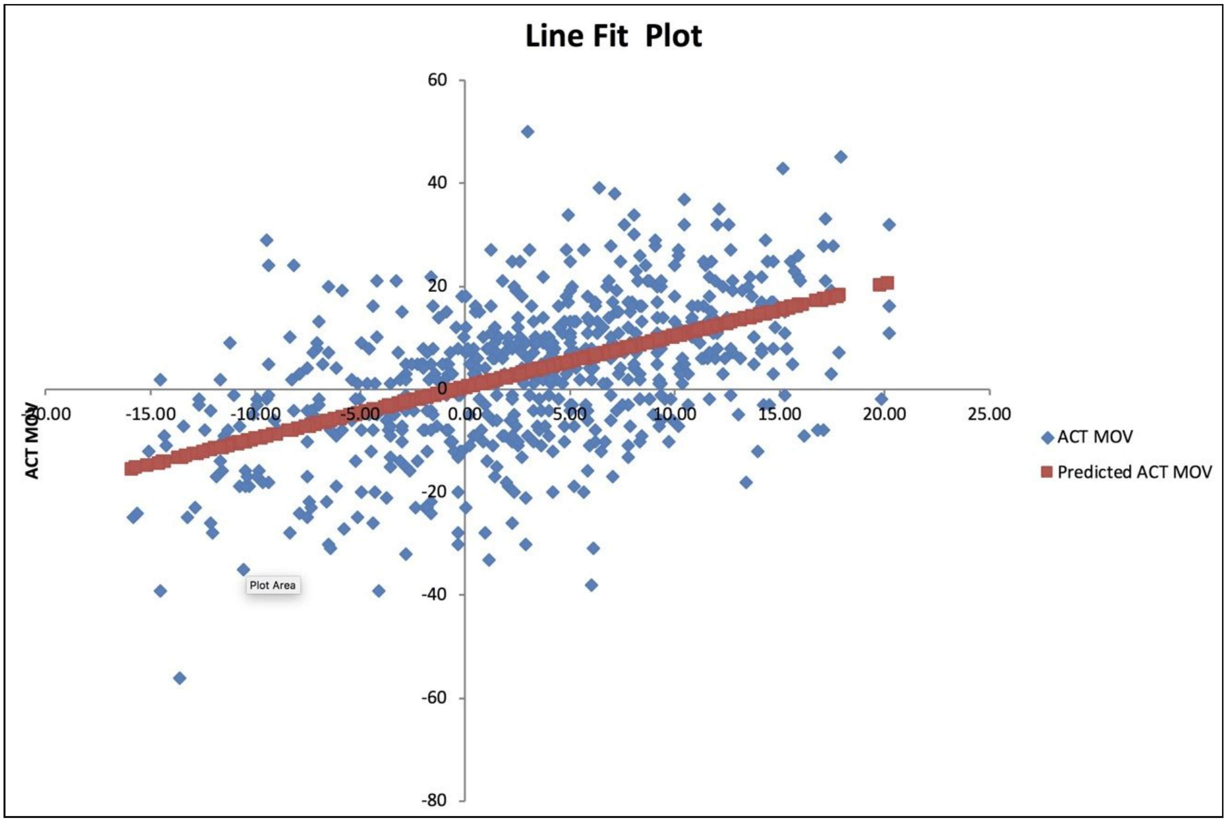


We have a slightly more scattered histogram, but all available tests indicate that the negative binomial distribution is our best fit (as an aside, imagine the kinds of errors that might occur if a modeller erroneously applied a normal distribution to this dataset). This kind of quick and dirty distribution fitting can be very helpful for your sports modelling efforts, as it tends to point you in the right direction early. If you decide not to get EasyFit (which I totally understand), not to worry – I've already completed this work for our purposes in this book. All you have to do is follow along with the models we'll be covering. If you want to explore new datasets however, I'd recommend looking into EasyFit.

Regression Methods

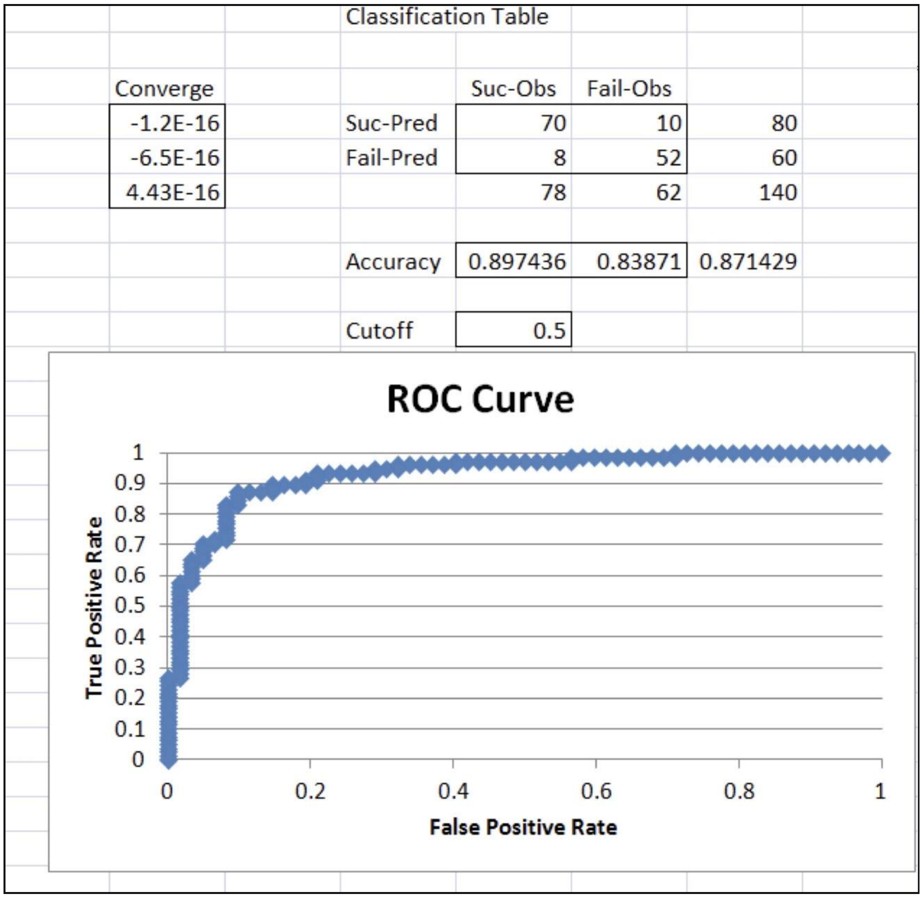
No crash course on Excel modelling would be sufficient without at least mentioning two of the more prominent methods: linear regression and logistic regression. While this is clearly not intended to be a detailed statistical tutorial, I want to briefly introduce the concepts to you so that you will recognize them more clearly when I demonstrate their use in models I'll explain later on. We'll be using both of these methods as mapping functions to connect team ratings to target outcome variables we want to forecast as well as when dealing with ensemble models. We'll also occasionally use linear regression to regress rate statistics for use in larger model configurations.

Multiple Linear Regression



Multiple linear regression[10] is a very simple modelling technique which describes a target output variable (Y) as a function of a number of input variables (X) weighted by various coefficients (weights) along with an intercept. It's so simple in fact that some sports bettors might scoff at the idea of including it in this book. "Surely we can't beat a sportsbook with such an entry-level technique!" While it's true that the simplicity inherent in linear regression brings with it several shortcomings, it is a very useful tool for a number of sports modelling tasks. For example, you might want to regress expected rate statistics like shooting percentages in NHL hockey or EPL soccer in order to incorporate them into a larger model, or add a regression as a mapping function that forms part of a more sophisticated ratings model. It does require a few assumptions about your data to hold true in order for it to work: the relationship must be linear (as would be expected) and the residuals of the model should be normally distributed. We must also make sure our explanatory variables are statistically significant (via p-values[11]) and that any model we put together with linear regression does not overtly suffer from highly correlated input variables, also known as multicollinearity[12]. I'll walk you through all of my uses of linear regression in the upcoming models as they arise, but for now consider this a very light introduction to get you up and running.

Logistic Regression



Classification Table

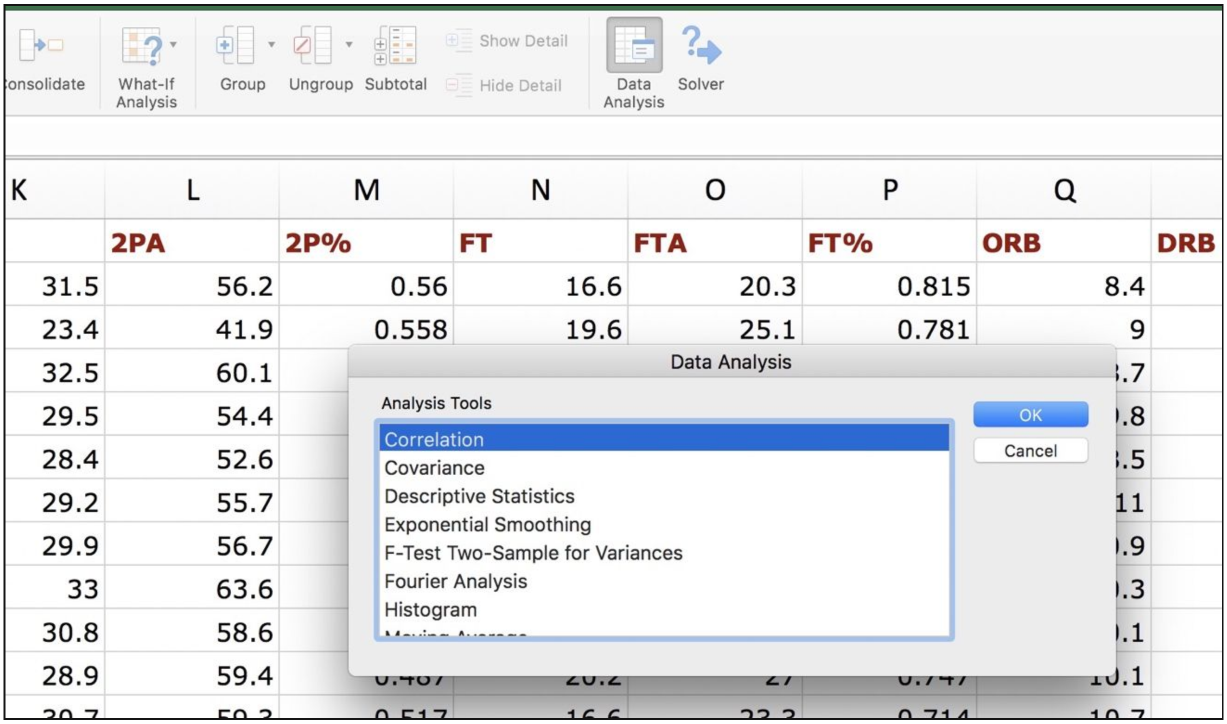
Converge		Suc-Obs	Fail-Obs	
-1.2E-16	Suc-Pred	70	10	80
-6.5E-16	Fail-Pred	8	52	60
4.43E-16		78	62	140
	Accuracy	0.897436	0.83871	0.871429
	Cutoff	0.5		

Logistic regression[13] is a tried-and-true probabilistic classifier that is extremely useful for sports modelling. Technically, it to maps a binary output variable of either 1 or 0 as a function of a number of input variables. What makes it so useful for our purposes is that it can output a classification as a probability between 0 and 1 which makes it applicable to a number of sports modelling tasks, including the mapping of target variables for ensemble models. Despite significant advances in machine learning algorithms, good old logistic regression can be a tough method to beat for binary classification on many datasets. We can also use logistic regression for multiple classification categories (as the multinomial[14] variant), making its utility extend to wagering opportunities that have more than 2 outcomes. Nothing makes an idea easier to understand than a visual example though, so you can be sure you'll see more on this later once we fire up the Excel models.

Correlation Analysis

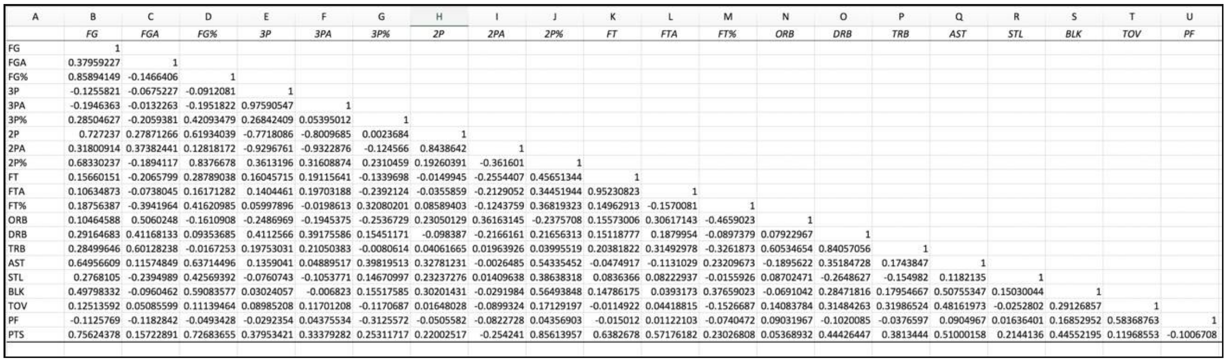
Correlation analysis is an excellent place to start looking at data relationships for the purposes of building a model. It's a foundational way to begin exploring similarity between variables. While it's important to remember that correlated variables don't necessarily imply any causal link, this can be a good way to start piecing together components for your models. Strong correlations indicate that there *may* be a relationship that is causal and warrants further investigation. Excel provides a built in CORREL command, but we can do even better. Using the Data Analysis ToolPak, we can quickly run a Pearson correlation[15] matrix to explore any strong positive or negative relationships that can be found within our dataset.

To begin, let's quickly go to basketball-reference.com and grab some team stats per game from the 2018 season as an example. Copy and paste them into Excel, then open the Data Analysis ToolPak by clicking on "Data Analysis". Then select "Correlation" and select OK.



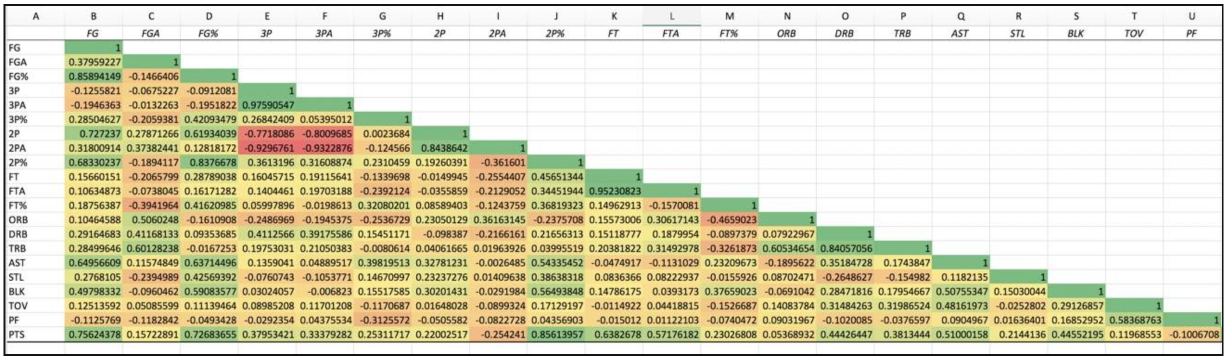
K	L	M	N	O	P	Q	
	2PA	2P%	FT	FTA	FT%	ORB	DRB
31.5	56.2	0.56	16.6	20.3	0.815	8.4	
23.4	41.9	0.558	19.6	25.1	0.781	9	
32.5	60.1					.7	
29.5	54.4					.8	
28.4	52.6					.5	
29.2	55.7					11	
29.9	56.7					.9	
33	63.6					.3	
30.8	58.6					.1	
28.9	59.4	0.487	20.2	27	0.747	10.1	
30.7	59.3	0.517	16.6	23.3	0.714	10.7	

We can then select the entire range of our data, making sure to only include numeric values. I like to include labels in the first row, so I select that checkbox. I then tell excel to produce a matrix for me on a new sheet and select OK. We can then observe the results:



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF
FG	1																			
FGA	0.37959227	1																		
FG%	0.85894149	-0.1466406	1																	
3P	-0.1255821	-0.0675227	-0.0912081	1																
3PA	-0.1946363	-0.0132263	-0.1951822	0.97590547	1															
3P%	0.28504627	-0.2059381	0.42093479	0.26842409	0.05395012	1														
2P	0.727237	0.27871266	0.61934039	-0.7718086	-0.8009685	0.0023684	1													
2PA	0.31800914	0.37382441	0.12818172	-0.9296761	-0.9322876	-0.124566	0.8438642	1												
2P%	0.68330237	-0.1894117	0.8376678	0.3613196	0.31608874	0.2310459	0.19260391	-0.361601	1											
FT	0.15660151	-0.2065799	0.28789038	0.16045715	0.19115641	-0.1339698	-0.0149945	-0.2554407	0.45651344	1										
FTA	0.10634873	-0.0738045	0.16171282	0.1404461	0.19703188	-0.2392124	-0.0355859	-0.2129052	0.34451944	0.95230823	1									
FT%	0.18756387	-0.3941964	0.41620985	0.05997896	-0.0198613	0.32080201	0.08589403	-0.1243759	0.36819323	0.14962913	-0.1570081	1								
ORB	0.10464588	0.5060248	-0.1610908	-0.2486969	-0.1945375	-0.2536729	0.23050129	0.36163145	-0.2375708	0.15573006	0.30617143	-0.4659023	1							
DRB	0.29164683	0.41168133	0.09353685	0.4112566	0.39175586	0.15451171	-0.098387	-0.2166161	0.21656313	0.15118777	0.1879954	-0.0897379	0.07922967	1						
TRB	0.28499646	0.60128238	-0.0167253	0.19753031	0.21050383	-0.0080614	0.04061665	0.01963926	0.03995519	0.20381822	0.31492978	-0.3261873	0.60534654	0.84057056	1					
AST	0.64956609	0.11574849	0.63714496	0.1359041	0.04889517	0.39819513	0.32781231	-0.0026485	0.54335452	-0.0474917	-0.1131029	0.23209673	-0.1895622	0.35184728	0.1743847	1				
STL	0.2768105	-0.2394989	0.42569392	-0.0760743	-0.1053771	0.14670997	0.23237276	0.01409638	0.38638318	0.0836366	0.08222937	-0.0155926	0.08702471	-0.2648627	-0.154982	0.1182135	1			
BLK	0.49798332	-0.0960462	0.59083577	0.03024057	-0.006823	0.15517585	0.30201431	-0.0291984	0.56493848	0.14786175	0.0393173	0.37659023	-0.0691042	0.28471816	0.17954667	0.50755347	0.15030044	1		
TOV	0.12513592	0.05085599	0.11139464	0.08985208	0.11701208	-0.1170687	0.01648028	-0.0899324	0.17129197	-0.0114922	0.04418815	-0.1526687	0.14083784	0.31484263	0.31986524	0.48161973	-0.0252802	0.29126857	1	
PF	-0.1125769	-0.1182842	-0.0493428	-0.0292354	0.04375534	-0.3125572	-0.0505582	-0.0822728	0.04356903	-0.015012	0.01122103	-0.0740472	0.09031967	-0.1020085	-0.0376597	0.0904967	0.01636401	0.16852952	0.58368763	1
PTS	0.75624378	0.15722891	0.72683655	0.37953421	0.33379282	0.25311717	0.22002517	-0.254241	0.85613957	0.6382678	0.57176182	0.23026808	0.05368932	0.44426447	0.3813444	0.51000158	0.2144136	0.44552195	0.11968553	-0.1006708

It's helpful to colour code the matrix values to make them easier to read, so let's do that as well by clicking on "Conditional Formatting" and then selection the first option under "Colour Scales":



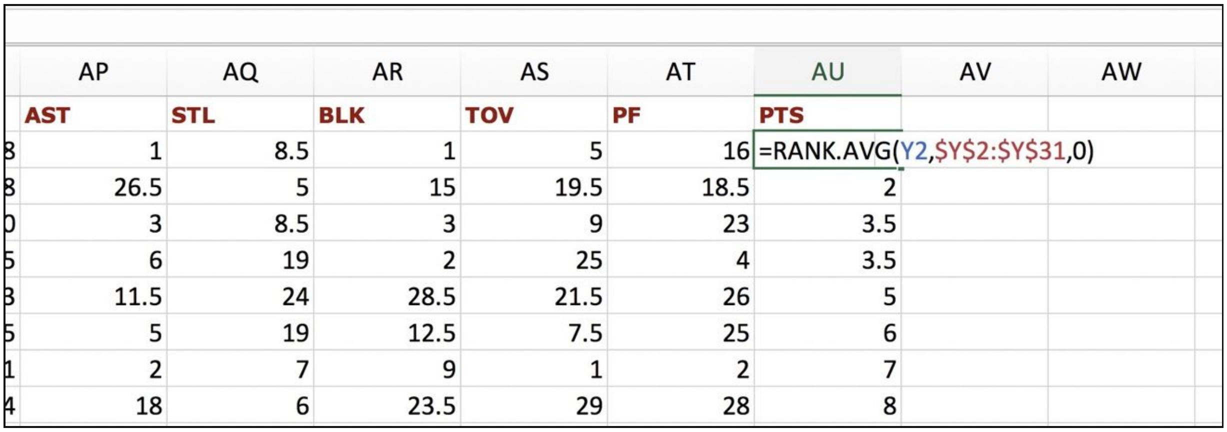
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	FG	*FGA*	*FG%*	*3P*	*3PA*	*3P%*	*2P*	*2PA*	*2P%*	*FT*	*FTA*	*FT%*	*ORB*	*DRB*	*TRB*	*AST*	*STL*	*BLK*	*TOV*	*PF*
FG	1																			
FGA	0.37959227	1																		
FG%	0.85894149	-0.1466406	1																	
3P	-0.1255821	-0.0675227	-0.0912081	1																
3PA	-0.1946363	-0.0132263	-0.1951822	0.97590547	1															
3P%	0.28504627	-0.2059381	0.42093479	0.26842409	0.05395012	1														
2P	0.727237	0.27871266	0.61934039	-0.7718086	-0.8009685	0.0023684	1													
2PA	0.31800914	0.37382441	0.12818172	-0.9296761	-0.9322876	-0.124566	0.8438642	1												
2P%	0.68330237	-0.1894117	0.8376678	0.3613196	0.31608874	0.2310459	0.19260391	-0.361601	1											
FT	0.15660151	-0.2065799	0.28789038	0.16045715	0.19115641	-0.1339698	-0.0149945	-0.2554407	0.45651344	1										
FTA	0.10634873	-0.0738045	0.16171282	0.1404461	0.19703188	-0.2392124	-0.0355859	-0.2129052	0.34451944	0.95230823	1									
FT%	0.18756387	-0.3941964	0.41620985	0.05997896	-0.0198613	0.32080201	0.08589403	-0.1243759	0.36819323	0.14962913	-0.1570081	1								
ORB	0.10464588	0.5060248	-0.1610908	-0.2486969	-0.1945375	-0.2536729	0.23050129	0.36163145	-0.2375708	0.15573006	0.30617143	-0.4659023	1							
DRB	0.29164683	0.41168133	0.09353685	0.4112566	0.39175586	0.15451171	-0.098387	-0.2166161	0.21656313	0.15118777	0.1879954	-0.0897379	0.07922967	1						
TRB	0.28499646	0.60128238	-0.0167253	0.19753031	0.21050383	-0.0080614	0.04061665	0.01963926	0.03995519	0.20381822	0.31492978	-0.3261873	0.60534654	0.84057056	1					
AST	0.64956609	0.11574849	0.63714496	0.1359041	0.04889517	0.39819513	0.32781231	-0.0026485	0.54335452	-0.0474917	-0.1131029	0.23209673	-0.1895622	0.35184728	0.1743847	1				
STL	0.2768105	-0.2394989	0.42569392	-0.0760743	-0.1053771	0.14670997	0.23237276	0.01409638	0.38638318	0.0836366	0.08222937	-0.0155926	0.08702471	-0.2648627	-0.154982	0.1182135	1			
BLK	0.49798332	-0.0960462	0.59083577	0.03024057	-0.006823	0.15517585	0.30201431	-0.0291984	0.56493848	0.14786175	0.0393173	0.37659023	-0.0691042	0.28471816	0.17954667	0.50755347	0.15030044	1		
TOV	0.12513592	0.05085599	0.11139464	0.08985208	0.11701208	-0.1170687	0.01648028	-0.0899324	0.17129197	-0.0114922	0.04418815	-0.1526687	0.14083784	0.31484263	0.31986524	0.48161973	-0.0252802	0.29126857	1	
PF	-0.1125769	-0.1182842	-0.0493428	-0.0292354	0.04375534	-0.3125572	-0.0505582	-0.0822728	0.04356903	-0.015012	0.01122103	-0.0740472	0.09031967	-0.1020085	-0.0376597	0.0904967	0.01636401	0.16852952	0.58368763	1
PTS	0.75624378	0.15722891	0.72683655	0.37953421	0.33379282	0.25311717	0.22002517	-0.254241	0.85613957	0.6382678	0.57176182	0.23026808	0.05368932	0.44426447	0.3813444	0.51000158	0.2144136	0.44552195	0.11968553	-0.1006708

Now we can begin to assess what we see. Suppose we wanted to know if any of these variables were connected to points scored per game. Obviously, many of these variables are connected and we're already aware of that, but the process is the same even when exploring unknown data connections. We immediately see that 2P% or 2 point shot percentage is very strongly positively correlated to points per game. That might be something to explore! Interestingly, we can also see that 2 point shot attempts are negatively correlated to points per game, indicating that the more 2 point attempts a team makes, the lower its average points per game. Could it be that teams with bad shooters are forced to compensate by taking more shot attempts? This is the kind of investigative thought process that can lead to useful model inputs, and while this is a superfluous example, I hope it illustrates the concept well enough. We'll actually do some meaningful correlation work later on, but for now I'd just like you to see the basic process.

Spearman Rank Correlation

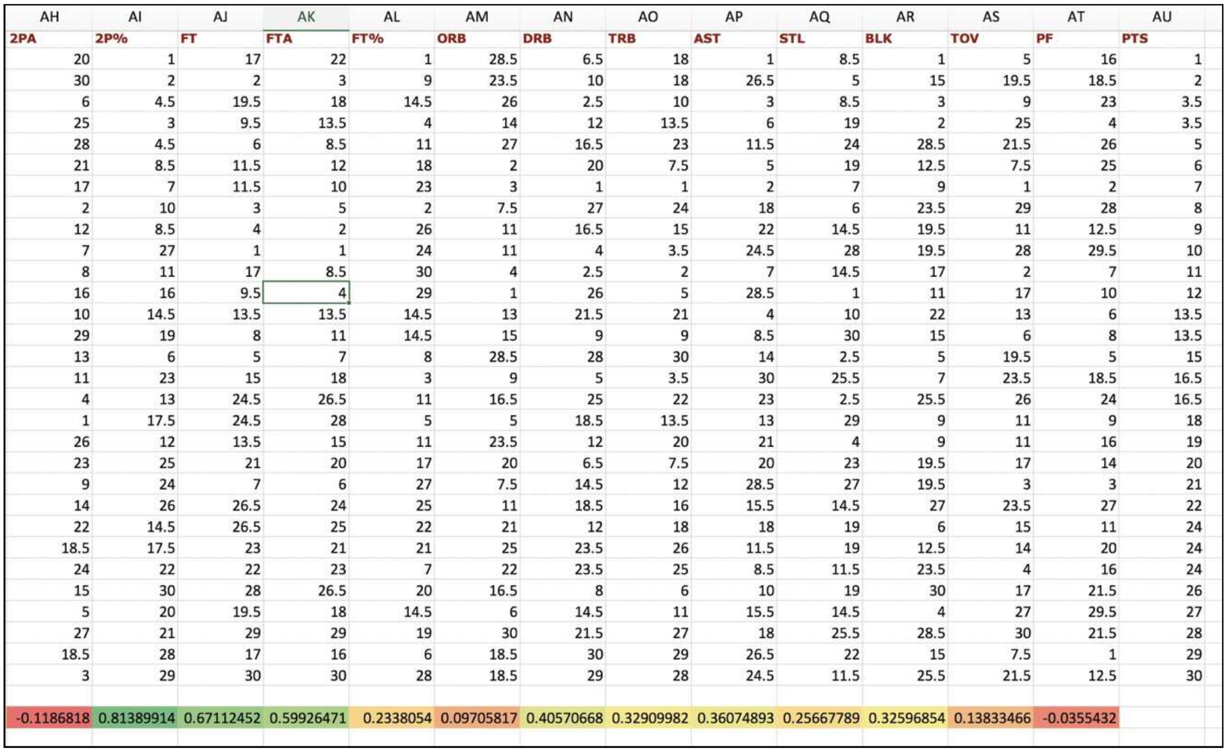
Our first correlation matrix was technically a Pearson correlation matrix, but there are other forms of correlation. The most notable alternative is Spearman rank correlation[16] which is less sensitive to outliers in the dataset. This is simply another way to assess the relationships in our data, but provides some additional advantages. If our dataset possibly contains non-linear relationships (and many sports do), ordinal data[17], or a significant number of outliers, we should probably consider using Spearman over Pearson. While there is no command for Spearman rank correlation in Excel, it is fairly simple to calculate. Unfortunately we must do a bit of legwork to get it up and running.

Spearman correlation determines variable relationships based on rank, so the first thing we need to do is rank our data variables by column using the RANK.AVG command in Excel. We must do this for every variable column we want to analyze, which will double the dataset's number of columns. Notice that our final input in the RANK command is “0” for descending order.



AP	AQ	AR	AS	AT	AU	AV	AW
AST	STL	BLK	TOV	PF	PTS		
1	8.5	1	5	16	=RANK.AVG(Y2,$Y$2:$Y$31,0)		
26.5	5	15	19.5	18.5	2		
3	8.5	3	9	23	3.5		
6	19	2	25	4	3.5		
11.5	24	28.5	21.5	26	5		
5	19	12.5	7.5	25	6		
2	7	9	1	2	7		
18	6	23.5	29	28	8		

Once this is complete, we can use the CORREL command in Excel on our dataset by anchoring the target outcome variable we wish to explore using $ inputs, and populating the command across the remaining variable columns.



AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU
2PA	2P%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
20	1	17	22	1	28.5	6.5	18	1	8.5	1	5	16	1
30	2	2	3	9	23.5	10	18	26.5	5	15	19.5	18.5	2
6	4.5	19.5	18	14.5	26	2.5	10	3	8.5	3	9	23	3.5
25	3	9.5	13.5	4	14	12	13.5	6	19	2	25	4	3.5
28	4.5	6	8.5	11	27	16.5	23	11.5	24	28.5	21.5	26	5
21	8.5	11.5	12	18	2	20	7.5	5	19	12.5	7.5	25	6
17	7	11.5	10	23	3	1	1	2	7	9	1	2	7
2	10	3	5	2	7.5	27	24	18	6	23.5	29	28	8
12	8.5	4	2	26	11	16.5	15	22	14.5	19.5	11	12.5	9
7	27	1	1	24	11	4	3.5	24.5	28	19.5	28	29.5	10
8	11	17	8.5	30	4	2.5	2	7	14.5	17	2	7	11
16	16	9.5	4	29	1	26	5	28.5	1	11	17	10	12
10	14.5	13.5	13.5	14.5	13	21.5	21	4	10	22	13	6	13.5
29	19	8	11	14.5	15	9	9	8.5	30	15	6	8	13.5
13	6	5	7	8	28.5	28	30	14	2.5	5	19.5	5	15
11	23	15	18	3	9	5	3.5	30	25.5	7	23.5	18.5	16.5
4	13	24.5	26.5	11	16.5	25	22	23	2.5	25.5	26	24	16.5
1	17.5	24.5	28	5	5	18.5	13.5	13	29	9	11	9	18
26	12	13.5	15	11	23.5	12	20	21	4	9	11	16	19
23	25	21	20	17	20	6.5	7.5	20	23	19.5	17	14	20
9	24	7	6	27	7.5	14.5	12	28.5	27	19.5	3	3	21
14	26	26.5	24	25	11	18.5	16	15.5	14.5	27	23.5	27	22
22	14.5	26.5	25	22	21	12	18	18	19	6	15	11	24
18.5	17.5	23	21	21	25	23.5	26	11.5	19	12.5	14	20	24
24	22	22	23	7	22	23.5	25	8.5	11.5	23.5	4	16	24
15	30	28	26.5	20	16.5	8	6	10	19	30	17	21.5	26
5	20	19.5	18	14.5	6	14.5	11	15.5	14.5	4	27	29.5	27
27	21	29	29	19	30	21.5	27	18	25.5	28.5	30	21.5	28
18.5	28	17	16	6	18.5	30	29	26.5	22	15	7.5	1	29
3	29	30	30	28	18.5	29	28	24.5	11.5	25.5	21.5	12.5	30
-0.1186818	0.81389914	0.67112452	0.59926471	0.2338054	0.09705817	0.40570668	0.32909982	0.36074893	0.25667789	0.32596854	0.13833466	-0.0355432	

This is not the full traditional formula for Spearman rank correlation, but its quick, easy, and best of all, it works. We can also colour code the correlation values as we did before with the Pearson correlation matrix using Conditional Formatting. We can then continue on with our analysis in the same way as we did with the Pearson correlation matrix.

A Word on Bayesian Statistics

There are two broad level approaches to statistical modelling: Frequentist and Bayesian. Although this book does not focus on Bayesian statistics[18], a short introduction and explanation of its importance will likely be helpful for your future modelling efforts. In short these two branches of statistics hold very different viewpoints on how to conceptualize probability, how to interpret distributions of data, and what questions are important to ask. My personal opinion is that Bayesian approaches appear vastly more useful for sports betting, but I'm not knowledgeable enough to make that argument mathematically just yet. Let's explore both approaches together from a high-level perspective to keep things simple.

Probabilistic Differences

The main point of contention between these two approaches seems to concern the nature of probability, making this disagreement of unique interest to sports bettors who by our very nature are interested in all things probabilistic. Frequentist statistics generally consider (as the name implies) the frequency of a given value in a dataset as an indicator of its probability of occurrence. That is to say the frequency of an event is the hallmark of the probability for that event, which seems fair enough.

Bayesian statistics alternatively consider probability to be more closely related to the degree of certainty (or belief) we might have about an event's occurrence. This is to say that our knowledge of an event has an important connection to the probability of the event's occurrence. So far this may sound a bit dumb. You might be thinking to yourself "So what? What's the big deal? They seem basically the same." You'd have a point, but as I'll hopefully be able to explain in simple terms, this nuanced difference leads to some substantively different implications for how each approach goes about modelling uncertainty. This often can be confusing to sort through when comparing the two approaches, because it's possible to arrive at the same answer using either approach in certain situations. However, I'll do my best to explain what separates them as best I can.

So far though, we have Frequentism: [Probability = Frequency of Events] and Bayesian: [Probability = Degree of Belief of Events]. If you're with me so far, let's keep going.

Probability Distributions

Frequentist and Bayesian methods differ in their conceptualization of probability distributions and the observations within them as well. While the Frequentist approach views distributions as fixed and the observations within them as fluid, the Bayesian approach considers the distributions to be fluid (or adaptable) and the observations within them to be fixed. To put it another way, the Bayesian approach takes the data observations "as they come", and slowly adapts the distribution while increasing its confidence in the probabilities that it implies. Provided I understand this correctly, this seems very intuitive. As we see more results, we should be more confident in our probabilities and it makes sense that the distributions would adapt to fluid (i.e. new) information. This really, is the heart of Bayesian inference: updating our prior beliefs in the face of new information or data observations.

Implications

This leads to two implications that are of interest to sports bettors and modellers: Bayesian statistics allow our probabilistic forecasts to improve in confidence over time as the distribution adapts to the newly observed data, and unlike Frequentist confidence intervals[19], we can be increasingly sure that the mean of our specific data falls within our Bayesian credible interval[20]. If you think of applying this to a sports team or season, where we might want to be more confident of a true skill rating as we observe more and more observations, I think you'll see why this approach could have a number of advantages. That being said, this is a hotly debated topic and both approaches have their undeniable merits. There is an excellent web post series[21] by Jake VanderPlas[22] going over the differences in the two approaches in a more rigorous way. I think you'll find it to be an insightful read. If nothing else, hopefully you've become interested in learning more and finding a way to experiment with Bayesian techniques in your modelling processes. If you are, I highly encourage you to read "Introduction to Empirical Bayes" by David Robinson, which focuses on MLB examples and is very easy to understand and apply in Excel. I think you'll find it to be very approachable.

Bring on the Models

If you've made it this far, you've probably realized there is a lot to learn about statistical modelling. Hopefully though, you've found some of the material covered useful. If you're interested in learning more about basic statistics, I have a number of book recommendations at the end of this book for you to explore. Back to the task at hand though! With no further fanfare, let's take a look at the first models in the book, and move more comfortably into an area I have extensive experience with.

---

[1] https://en.wikipedia.org/wiki/Laplace%27s_demon

[2] https://arxiv.org/pdf/1701.05976.pdf

[3] http://www.insidethebook.com/ee/index.php/site/comments/true_talent_levels_for_sports_leagues/

[4] http://blog.philbirnbaum.com/2011/08/tango-method-of-regression-to-mean-kind.html

[5] http://www.matterofstats.com

[6] https://en.wikipedia.org/wiki/Probability_distribution

[7] https://en.wikipedia.org/wiki/Weibull_distribution

[8] https://en.wikipedia.org/wiki/Log-normal_distribution

[9] https://en.wikipedia.org/wiki/Overdispersion#Poisson

[10] https://en.wikipedia.org/wiki/Linear_regression

[11] https://en.wikipedia.org/wiki/P-value

[12] https://en.wikipedia.org/wiki/Multicollinearity

[13] https://en.wikipedia.org/wiki/Logistic_regression

[14] https://en.wikipedia.org/wiki/Multinomial_logistic_regression

[15] https://en.wikipedia.org/wiki/Pearson_correlation_coefficient

[16] https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[17] https://en.wikipedia.org/wiki/Ordinal_data

[18] https://en.wikipedia.org/wiki/Bayesian_statistics

[19] https://en.wikipedia.org/wiki/Confidence_interval

[20] https://en.wikipedia.org/wiki/Credible_interval

[21] http://jakevdp.github.io/blog/2014/03/11/frequentism-and-bayesianism-a-practical-intro/

[22] https://twitter.com/jakevdp

DOWNLOAD MY SPREADSHEET MODELS

To make things as straightforward as possible, I've included working copies of all of the models and techniques I'll be discussing in this book as a gigantic downloadable collection of Excel spreadsheets. This way, you'll be able to follow along with what we cover. Although I'm biased, I think the utility of this download file is worth many times the cost of this book, and I hope once you go through it you'll feel the same way. Click the following link to download the entire model package file from MediaFire:

SSME Spreadsheet Models Download[1]

I'd very much appreciate it if you kept the location of this download link to yourself, although I recognize it's only a matter of time before it become ubiquitous on the internet. Feel free to use these models for your own purposes, explore new ideas, or try tweaking them for different sports. If you come up with anything interesting I'd love to hear about it. If you make reference to them, I would kindly request a link back to my twitter page.

Finally, I'd recommend making a copy of the original models so that you have working copies and originals. This way, if anything goes wrong while you're experimenting on your own, you have the original models to revert back to. Once you've downloaded the file and made yourself a backup copy we're ready to move on.

[1] For readers of the print edition, here is the download link:

http://www.mediafire.com/file/pjtim9ybll2o8x6/ExcelModels.zip/file

5

The first thing most people do when they attempt to build a model is entirely predictable. They head to the most relevant stats site, scour over various descriptive team statistics, and try to find some correlation or underlying causal relationship that hasn't already been discovered by the mass of bettors attempting to do the same thing.

It's little wonder that some bettors take a dim view of modelling and heavy statistically driven handicapping work. It's hands-down the hardest way to make money betting on sports, and if you approach it like most people, you're likely to get the same result as "most people". Not optimal in a negative sum game like sports betting. Allow me to explain.

There are a few things you must understand. First and foremost, many if not most of the statistics about your preferred sport that are widely available are descriptive rather than predictive. They are telling you about what has happened, but are of questionable value in your efforts to forecast what is likely to happen in the future. Far too few people understand this. This problem is compounded by the fact that there is a large swath of statistical methods entirely dedicated to descriptive efforts. What I mean to say is that there are a lot of ways to accidentally produce a model that is more like a rear-view mirror than a windshield. As you are surely aware, we're looking for a "windshield". Secondly, some of these descriptive statistics are inflated by various leagues and aren't entirely accurate to begin with. There was a shocking article about scorekeepers in the NBA artificially inflating game statistics for certain players not too long ago[1]. To make an even more persuasive case against team statistic based models, a recent research paper applied advanced deep learning algorithms to basic team box score data from the NBA and won a woeful 49% against the point spread while losing 66% of its hypothetical starting bank roll. That's a fairly strong indictment of basic box score model inputs if you ask me. You can read about that 2019 study here[2].

It is for these reasons that I am generally skeptical about team statistic-based models and have explored other ideas considerably. Certainly team stat models can and do occasionally work, but interestingly enough they usually perform somewhere in the middle of the pack compared to other modelling techniques I have tried. “What other modelling techniques?” you might wonder.

I’m glad you asked. Let me introduce you to a branch of “universal” models – models that will work for virtually any sport and any team. These models are known as pairwise comparison[3] or latent rating models. A latent rating model infers a team’s strength or weakness based on observable outcome data. Believe it or not, while these models are not state of the art, a surprising amount of predictive information about a sport can be gleaned from simply applying a latent rating model to the sport’s observable results. Even better, these results can be the outright winner of a game, the total points scored, or a point spread/runline/puckline. This makes this form of model work very useful and it’s where we’ll start – with generalized sport models that do a decent job of mapping any sport we’re trying to forecast using only the observable end-result data.

This is just the beginning, however. There is something else you should know. No single statistical model can be reasonably expected to beat a market, and certainly not a major market. I feel like this bears repeating: If you direct these models individually to the NBA point spread or another comparably sharp market and impetuously fire away your bankroll – you are going to lose. You’ll lose more slowly than if you simply bet on your gut instincts, but lose you will. There are a number of reasons for this which will be explained in due course, but understand that one of the solutions to this is to combine models together into a cluster (or “ensemble”) that can compensate for individual model weaknesses and take advantage of individual model strengths. The other vastly more influential solution is to attack smaller, more obscure markets where these models can win. This is precisely what we should focus on. I’ll show you how to build the individual models, and then we’ll combine them into an ensemble that can beat smaller markets, and at the very least push you in the right direction.

[1] https://deadspin.com/the-confessions-of-an-nba-scorekeeper-5345287

[2] https://www.sciencedirect.com/science/article/pii/S0169207017301152?dgcid=raven_sd_recommender_email

[3] https://en.wikipedia.org/wiki/Pairwise_comparison

THE BRADLEY TERRY MODEL

The Bradley Terry Model [BTM] is an intriguing modelling technique because of the way that it can reverse engineer underlying team strength based on a set of observed game outcomes. Technically, it is a pairwise comparison model, which is a subset of model types in statistics that has been used to map everything from lizard aggression dominance to consumer choices in digital marketplaces. Pairwise comparison models are also quite helpful for sports forecasting, as you might have already guessed. BTM essentially answers the following question for us: “based on the game outcomes we’ve seen - what team ratings would maximize the probability of producing these results?”

In order to achieve this, we’ll be using an excel version of a statistical technique known as maximum likelihood estimation [MLE]. MLE is a statistical method for estimating parameters to maximize a likelihood function that would make the observed data the most probable. While I don’t want to get too bogged down in the details, the bottom line is that it is a technique we can use to reverse engineer team strength ratings based on observed game outcomes, and we can piece together a way to do all of this in Excel. More importantly, especially for some smaller market sports – it turns out a forecast that’s fairly decent.

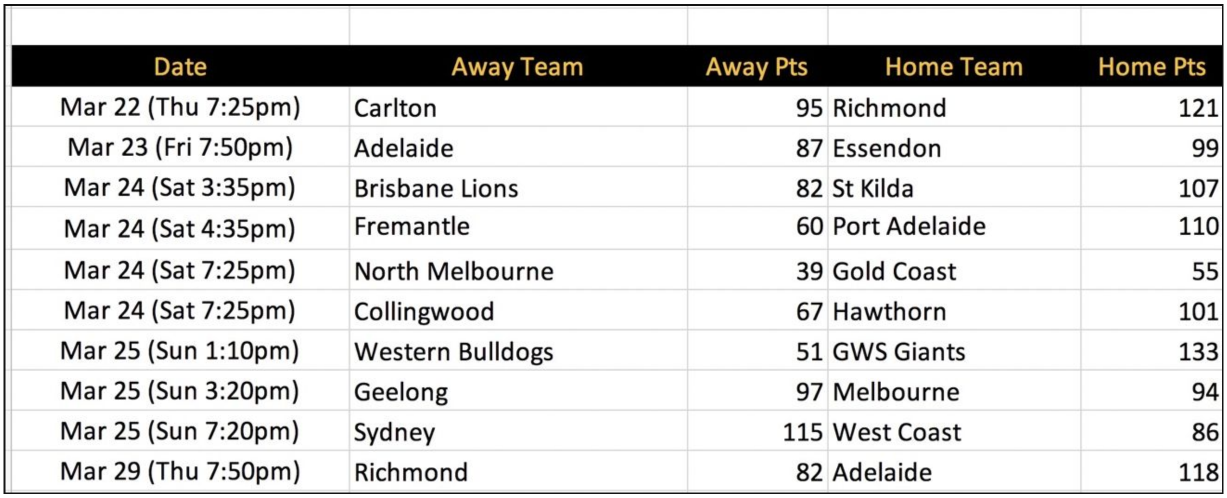
BTM has some advantages and disadvantages. For sports betting, one of its main advantages is that it can create a workable model with the bare minimum of data available in every sport on earth – the final result of each game. This means you can finesse this model into obscure, smaller market sports where the amount of available game data is severely limited. You might not be able to get advanced stats for the Icelandic Women’s Basketball league, but you can definitely get the game scores. If you can make a feasible model in a sport with difficult access to advanced statistics, it can help you to identify opportunities that other bettors might miss. It's usually an even better opportunity if the sport isn’t of widespread commercial interest to fans. Look for sports and leagues with no publicly televised games, sports you’ve rarely heard mentioned in your neighbourhood pub, and teams you’ve never seen a piece of merchandise for. After all, we’re not betting sports to be entertained, though it certainly can provide that - we’re trying to make money!

> *looks up Icelandic Women’s Basketball*
>
> “um….go Skallagrímur!?”
>
> I can’t even properly pronounce that. It doesn’t matter. +EV over everything.

One of the disadvantages of this model is that it is inherently modelling some degree of statistical noise, since our data input (game results) is quite noisy. Some of the games in our sample will naturally have turned out the way they did because of randomness rather than skill, and we need to be careful about over-fitting a model to factors that have questionable predictive value. Another issue with the configuration you’re about to be shown is that it doesn’t factor any sense of time-weighting decay. In other words, it fits all the games with equal relevance as though the first game of a season is just as indicative of a team’s upcoming performance as their last game, or their last 5 games. We know this isn’t a very useful assumption. In fact, there is some research on team strength that suggests it mirrors a random walk process throughout the course of a season, and being able to account for that is something that may lead to an additional advantage for us. Finally, BTM in its original setup doesn’t account for draws, which hampers its utility a bit. This can be overcome however by making some adjustments to the model that I’ll show you later as we dive in to the excel spreadsheet I have provided.

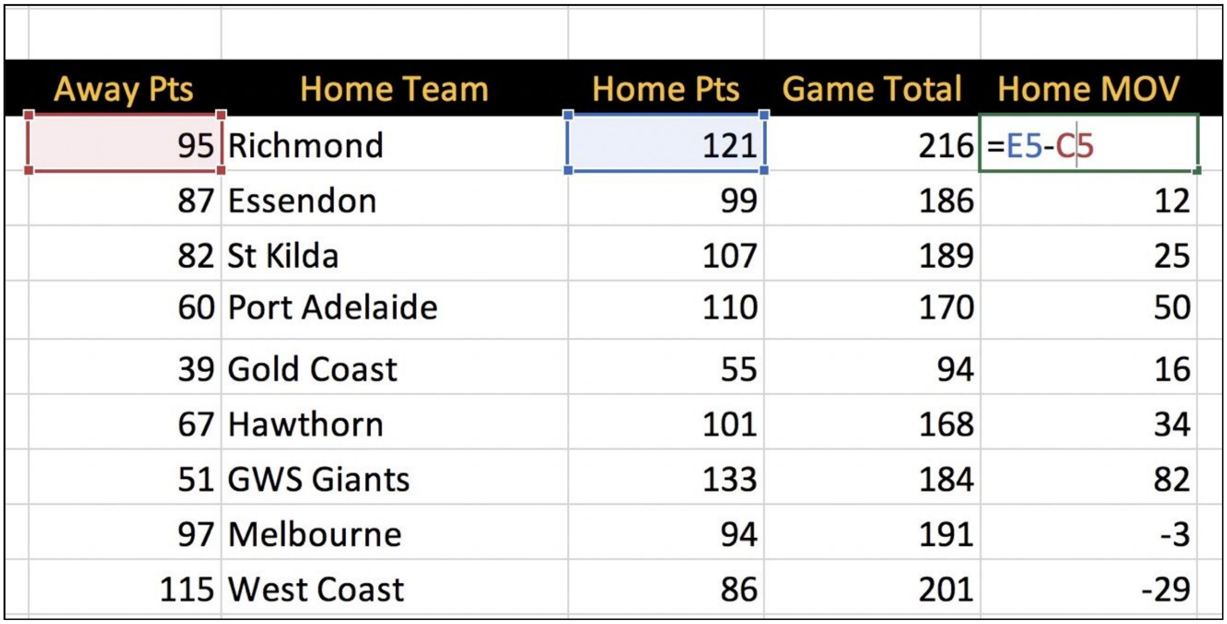
Getting Started

To start, let’s open the “Bradley Terry Model” Excel file. For this example I’ll be using data from the Men’s Aussie Rules Football (AFL) league from 2018. Columns A, B, C, D and E are the input data simply conveying the date of each game, the away team, the home team and their respective scores.



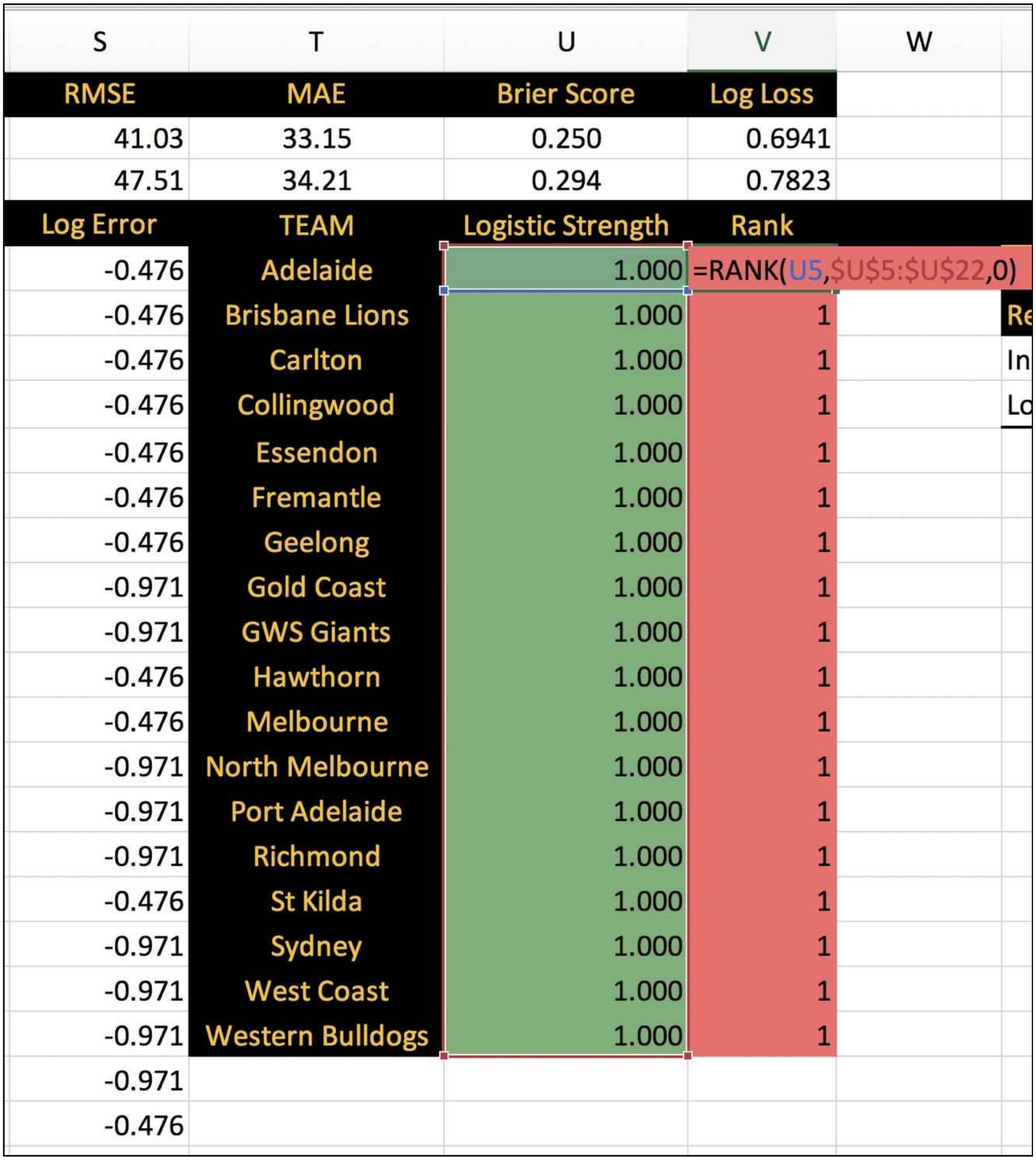
Date	Away Team	Away Pts	Home Team	Home Pts
Mar 22 (Thu 7:25pm)	Carlton	95	Richmond	121
Mar 23 (Fri 7:50pm)	Adelaide	87	Essendon	99
Mar 24 (Sat 3:35pm)	Brisbane Lions	82	St Kilda	107
Mar 24 (Sat 4:35pm)	Fremantle	60	Port Adelaide	110
Mar 24 (Sat 7:25pm)	North Melbourne	39	Gold Coast	55
Mar 24 (Sat 7:25pm)	Collingwood	67	Hawthorn	101
Mar 25 (Sun 1:10pm)	Western Bulldogs	51	GWS Giants	133
Mar 25 (Sun 3:20pm)	Geelong	97	Melbourne	94
Mar 25 (Sun 7:20pm)	Sydney	115	West Coast	86
Mar 29 (Thu 7:50pm)	Richmond	82	Adelaide	118

Now, clearly we are going to want to model either margin of victory [MOV] (for a win probability model or a point spread model) or total points (for a totals model), so let’s add those outcome results in the next two columns for each game. I’ve calculated these from the home team’s perspective, so that a MOV of 17 means the home team won by 17, for example. Totals, naturally, remain the same from either perspective.



Away Pts	Home Team	Home Pts	Game Total	Home MOV
95	Richmond	121	216	=E5-C5
87	Essendon	99	186	12
82	St Kilda	107	189	25
60	Port Adelaide	110	170	50
39	Gold Coast	55	94	16
67	Hawthorn	101	168	34
51	GWS Giants	133	184	82
97	Melbourne	94	191	-3
115	West Coast	86	201	-29

Next, I’ve gone ahead and set up dummy team strength ratings. These are essentially placeholders that the model will eventually tune based on our logistic function in order to arrive at the team ratings that maximize the probability of observing the game results we’ve seen. I include each team, insert a dummy logistic rating of “1.000” and then calculate a rank using the following command in Excel. The $ symbol of the formula anchors the excel range in place to prevent it from moving when I drag the formula to “auto-complete” other cells.



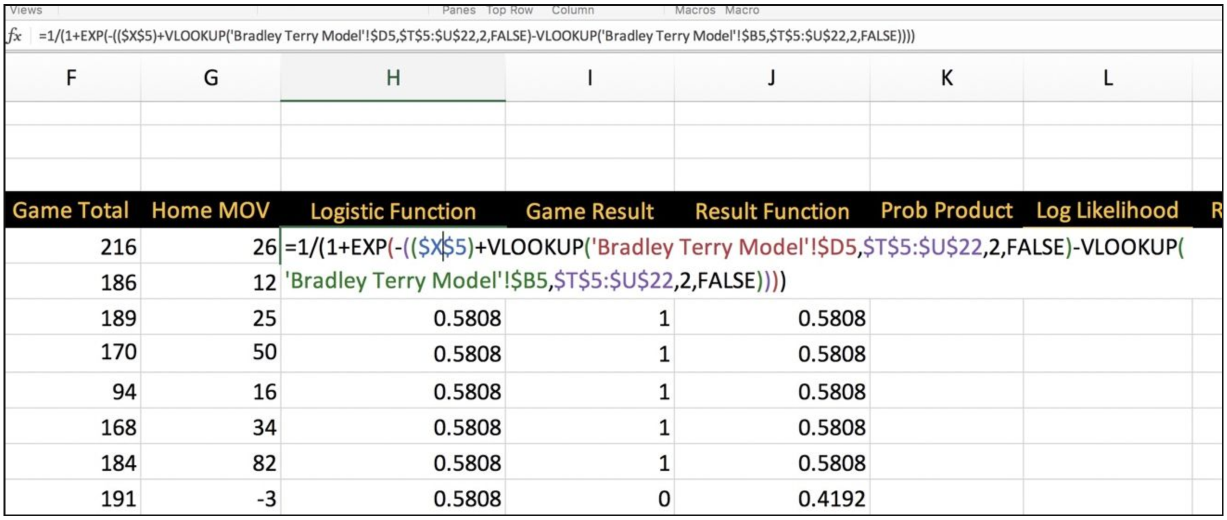
S	T	U	V	W	
RMSE	MAE	Brier Score	Log Loss		
41.03	33.15	0.250	0.6941		
47.51	34.21	0.294	0.7823		
Log Error	TEAM	Logistic Strength	Rank		
-0.476	Adelaide	1.000	=RANK(U5,$U$5:$U$22,0)		
-0.476	Brisbane Lions	1.000	1		Re
-0.476	Carlton	1.000	1		In
-0.476	Collingwood	1.000	1		Lo
-0.476	Essendon	1.000	1		
-0.476	Fremantle	1.000	1		
-0.476	Geelong	1.000	1		
-0.971	Gold Coast	1.000	1		
-0.971	GWS Giants	1.000	1		
-0.476	Hawthorn	1.000	1		
-0.476	Melbourne	1.000	1		
-0.971	North Melbourne	1.000	1		
-0.971	Port Adelaide	1.000	1		
-0.971	Richmond	1.000	1		
-0.476	St Kilda	1.000	1		
-0.971	Sydney	1.000	1		
-0.971	West Coast	1.000	1		
-0.971	Western Bulldogs	1.000	1		
-0.971					
-0.476					

Now we're ready to setup our logistic function. This tells the model how to use the ratings and essentially how we'd like the results to be optimized. The function technically looks like this:

```
=1/(1+EXP(-(HFA+HT-AT)))
```

Where EXP represents the exponent command in Excel, HFA is the home field advantage, HT is the home team rating, and AT is the away team rating. This function would effectively give us the strength differential between the home team and the away team and would thus be suitable for an MOV model that outputs either an outright win probability or a point spread win probability. In order to make this work in Excel, we'll be using the VLOOKUP command. This command allows us to connect different parts of our spreadsheet (or different spreadsheets altogether) for the purposes of integrating our model. You could think of it as a very simplistic call command that retrieves information for us.

What we want to do is tell Excel to reference (or "lookup") the logistic team rating of each team playing in a given game and apply it to our function. This is inputted as the following command:

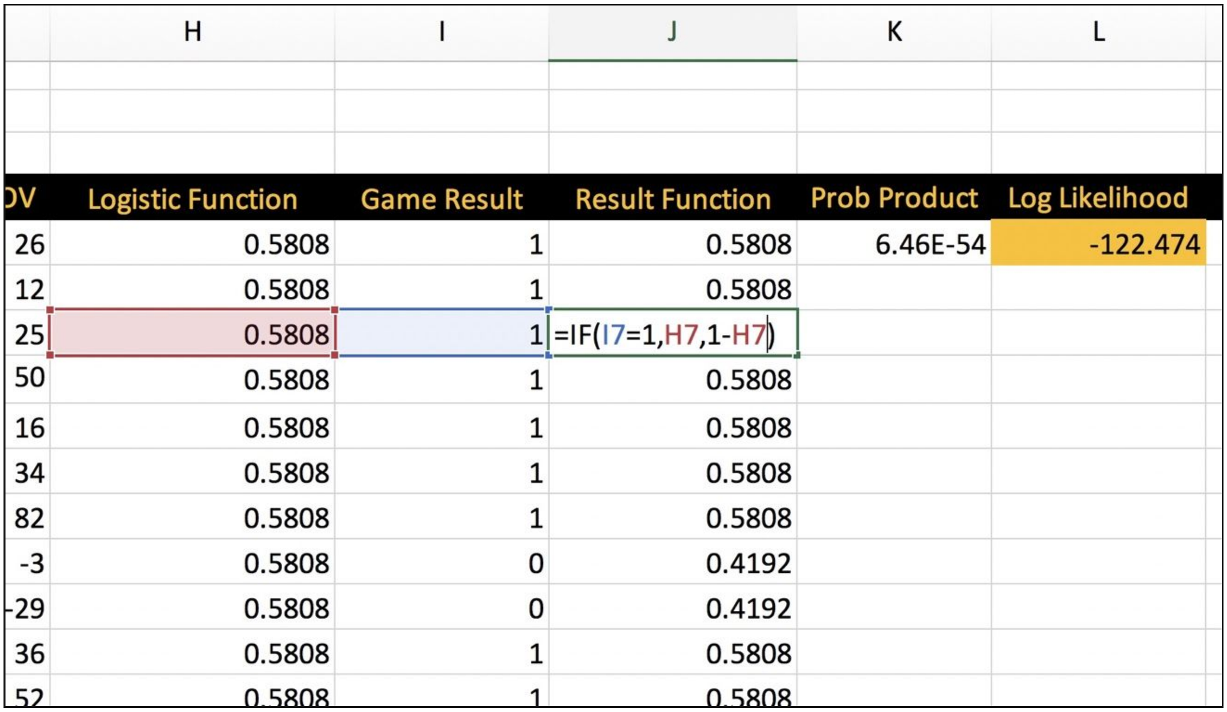


```
=1/(1+EXP(-(($X$5)+VLOOKUP('Bradley Terry Model'!$D5,$T$5:$U$22,2,FALSE)-VLOOKUP('Bradley Terry Model'!$B5,$T$5:$U$22,2,FALSE))))
```

F	G	H	I	J	K	L	R
Game Total	Home MOV	Logistic Function	Game Result	Result Function	Prob Product	Log Likelihood	R
216	26	=1/(1+EXP(-(($X$5)+VLOOKUP('Bradley Terry Model'!$D5,$T$5:$U$22,2,FALSE)-VLOOKUP(					
186	12	'Bradley Terry Model'!$B5,$T$5:$U$22,2,FALSE)))					
189	25	0.5808	1	0.5808			
170	50	0.5808	1	0.5808			
94	16	0.5808	1	0.5808			
168	34	0.5808	1	0.5808			
184	82	0.5808	1	0.5808			
191	-3	0.5808	0	0.4192			

You'll notice now that every value in column H is the same. That's because all of our teams currently have the same dummy rating – so the function sees them all as having the same win probability. Also note that the home field advantage has been added in cell X5 as another dummy variable, which we will later ask Excel to optimize for us. As previously mentioned, the $ symbol in the formulas are simply there to make the range an absolute reference, meaning that as you copy and paste the formula into different cells, or drag/double-click the cell to auto-populate further cells, the range will stay the same.

The next step is to create a result column and a result function column (columns I and J). This is where we will record whether the home team won or lost the game and which probability output to use in the future. 1 represents a home team win, and 0 represents an away team win. For the result function, if the home team wins the game, the result is the value from column H, the logistic function output. If the away team wins the game, we use [1-[Column H value]] instead:



	H	I	J	K	L
DV	Logistic Function	Game Result	Result Function	Prob Product	Log Likelihood
26	0.5808	1	0.5808	6.46E-54	-122.474
12	0.5808	1	0.5808		
25	0.5808	1	=IF(I7=1,H7,1-H7)		
50	0.5808	1	0.5808		
16	0.5808	1	0.5808		
34	0.5808	1	0.5808		
82	0.5808	1	0.5808		
-3	0.5808	0	0.4192		
-29	0.5808	0	0.4192		
36	0.5808	1	0.5808		
52	0.5808	1	0.5808		

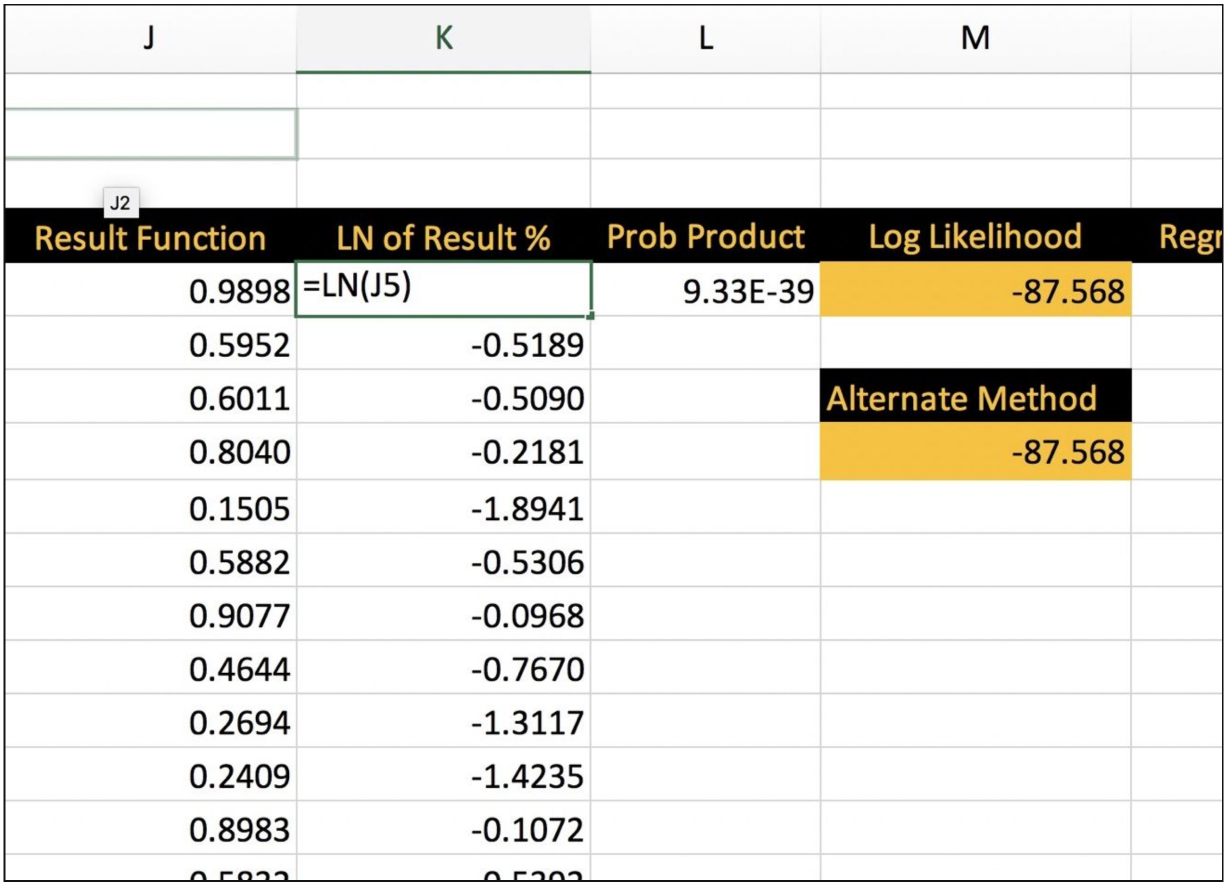
The next step is to calculate the probability product of the entire range of game results in column J, the result function. I have inserted this into cell L5 as follows:

```
=PRODUCT(J5:J182)
```

Our final piece of setup work is to simply convert this probability product into a log likelihood [LL]. I’ve inserted this into cell M5 using the LN command in Excel:

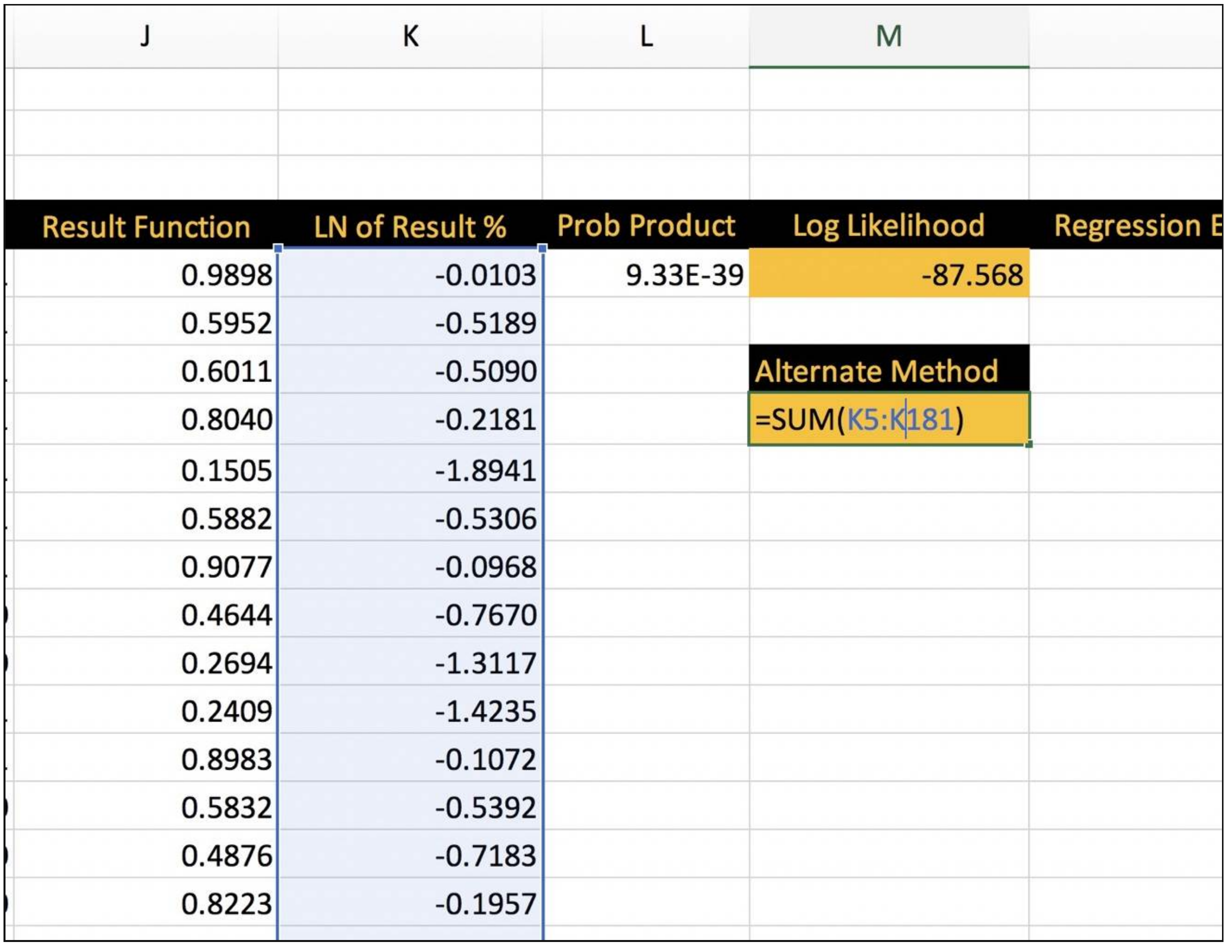
```
=LN(K5)
```

Excel can take issues with the extremely small numbers this method produces. There is another method for calculating Log Likelihood that can come in handy if you have larger datasets which will inevitably produce smaller numbers. Use the LN function on each result probability as shown in column K and then use the SUM function to sum all the resulting numbers.



J	K	L	M	
Result Function	LN of Result %	Prob Product	Log Likelihood	Regr
0.9898	=LN(J5)	9.33E-39	-87.568	
0.5952	-0.5189			
0.6011	-0.5090		Alternate Method	
0.8040	-0.2181		-87.568	
0.1505	-1.8941			
0.5882	-0.5306			
0.9077	-0.0968			
0.4644	-0.7670			
0.2694	-1.3117			
0.2409	-1.4235			
0.8983	-0.1072			

I have provided an example of this method in cell M8. You can use this to replace the LL cell that we want to maximize using Solver.



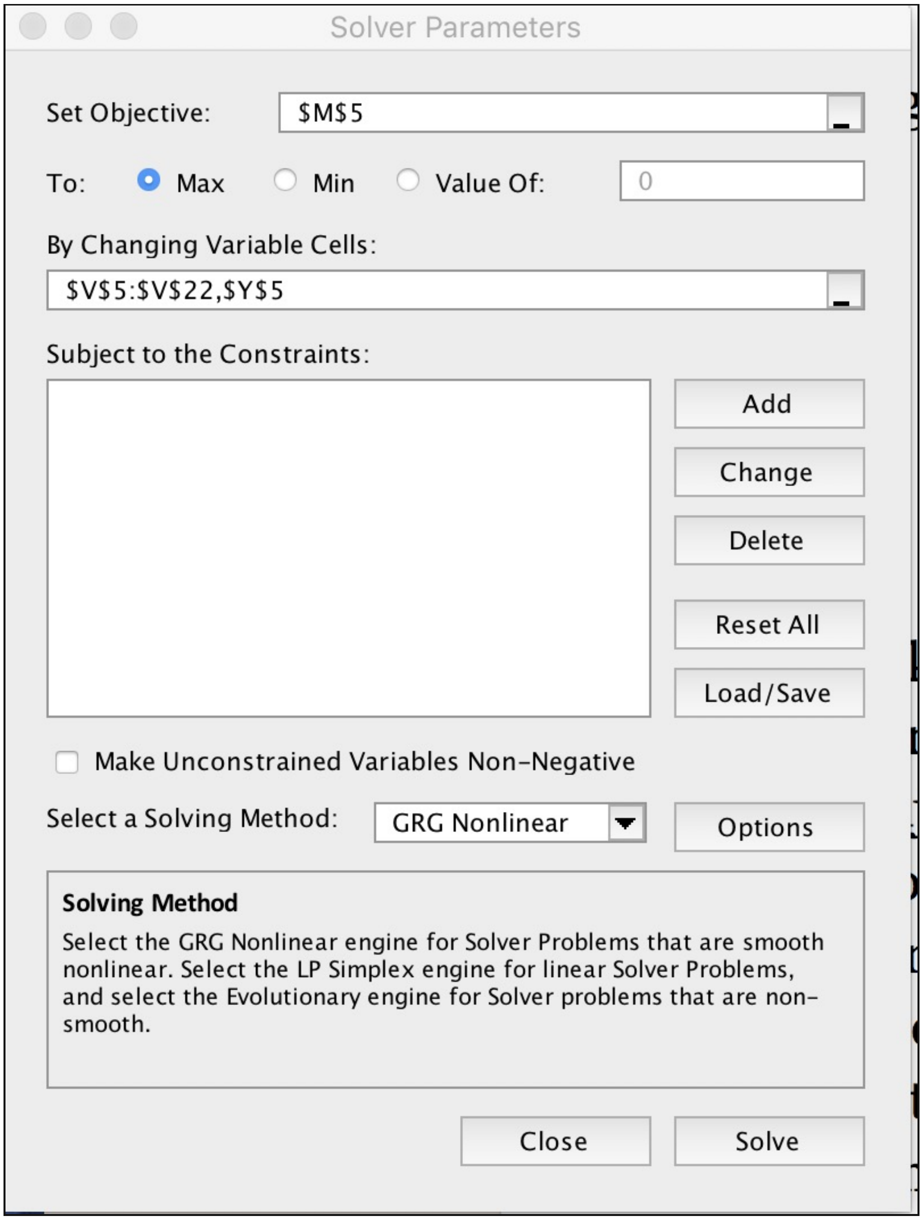
J	K	L	M	
Result Function	LN of Result %	Prob Product	Log Likelihood	Regression E
0.9898	-0.0103	9.33E-39	-87.568	
0.5952	-0.5189			
0.6011	-0.5090		Alternate Method	
0.8040	-0.2181		=SUM(K5:K181)	
0.1505	-1.8941			
0.5882	-0.5306			
0.9077	-0.0968			
0.4644	-0.7670			
0.2694	-1.3117			
0.2409	-1.4235			
0.8983	-0.1072			
0.5832	-0.5392			
0.4876	-0.7183			
0.8223	-0.1957			

This can prevent issues Excel has when dealing with extremely small numbers[1].

Optimizing the Team Ratings

Now we're ready to apply MLE to the model in order to optimize the team ratings. In order to achieve this we'll be using Excel's Solver function. Click on "Data" at the top of Excel, and then click the solver icon (top right) and input the following commands:

> Set Objective: $M$5
>
> To: Max
>
> By Changing Variable Cells: $V$5:$V$22,$Y$5



Set Objective: $M$5

To: ● Max ○ Min ○ Value Of: 0

By Changing Variable Cells:
$V$5:$V$22,$Y$5

Subject to the Constraints:

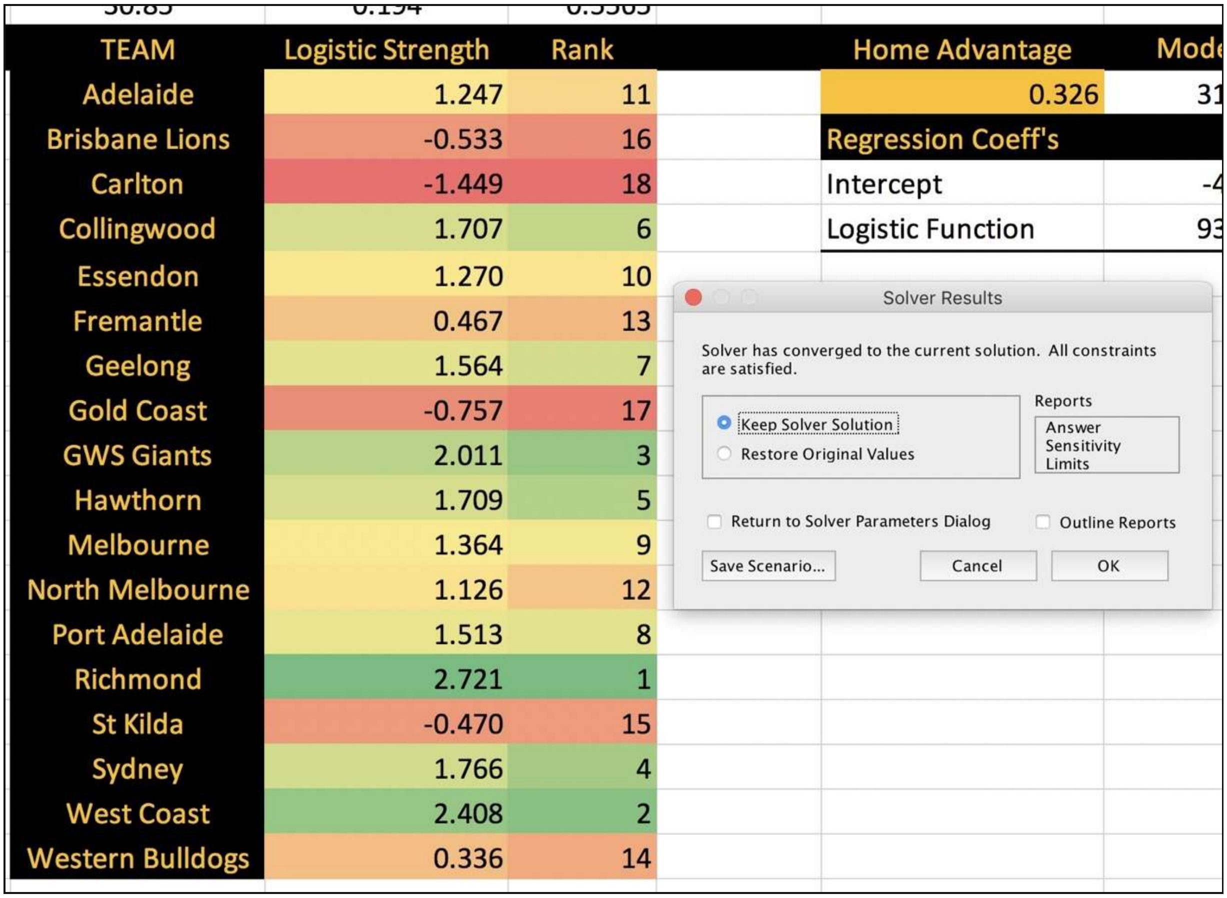
☐ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

What we've asked solver to do here is to maximize the LL found in cell M5 (or M8 using the alternate method) by adjusting the team ratings and the HFA value. This is our Excel version of maximum likelihood estimation. Click Solve and Excel will now start tuning the team rating values and the HFA to produce an optimized result. When Solver is finished you should see a screen like this – including updated team ratings:



TEAM	Logistic Strength	Rank
Adelaide	1.247	11
Brisbane Lions	-0.533	16
Carlton	-1.449	18
Collingwood	1.707	6
Essendon	1.270	10
Fremantle	0.467	13
Geelong	1.564	7
Gold Coast	-0.757	17
GWS Giants	2.011	3
Hawthorn	1.709	5
Melbourne	1.364	9
North Melbourne	1.126	12
Port Adelaide	1.513	8
Richmond	2.721	1
St Kilda	-0.470	15
Sydney	1.766	4
West Coast	2.408	2
Western Bulldogs	0.336	14

Home Advantage	Mode
0.326	31
Regression Coeff's	
Intercept	-4
Logistic Function	93

Solver Results

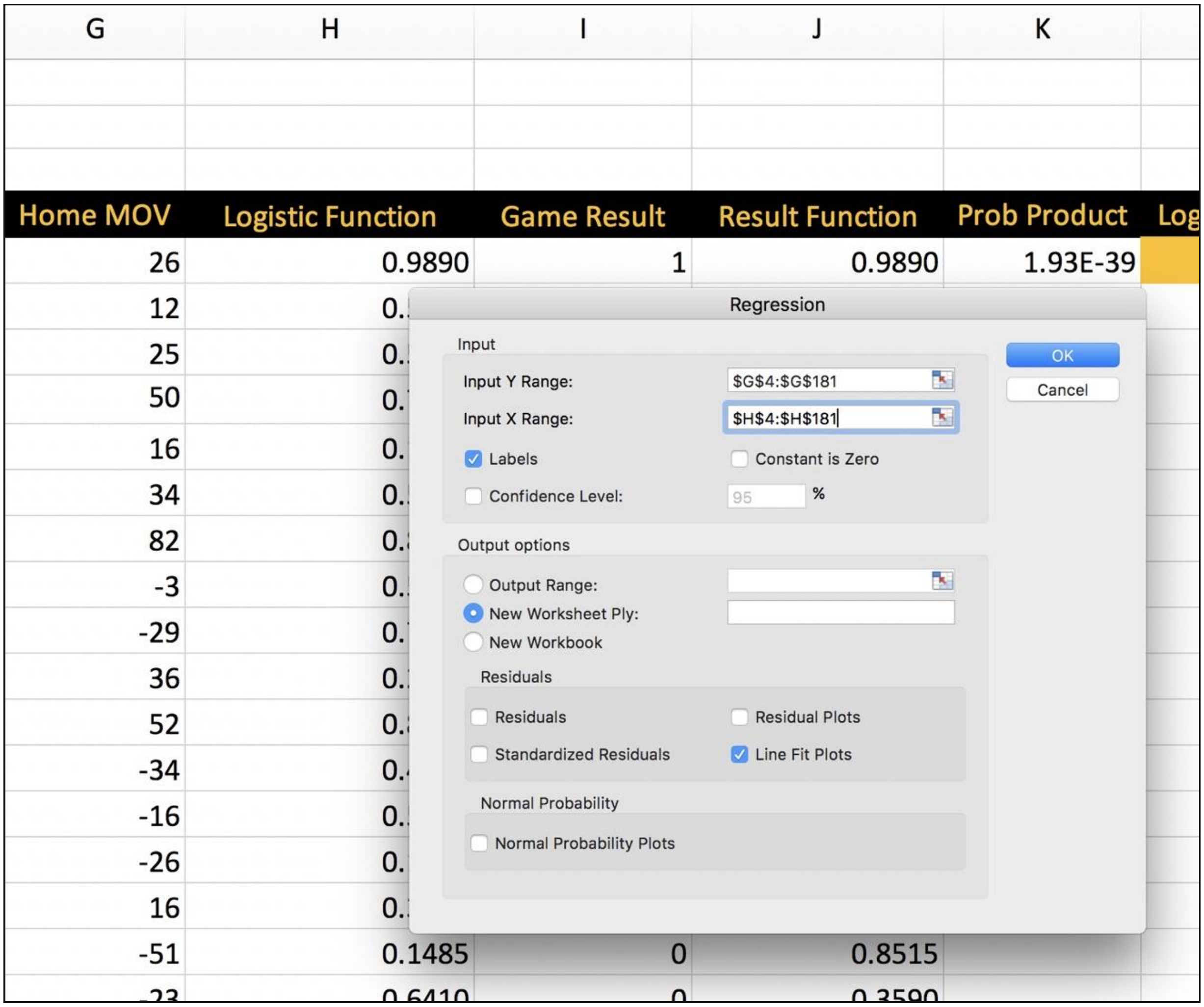
Solver has converged to the current solution. All constraints are satisfied.

● Keep Solver Solution
○ Restore Original Values

Reports
Answer
Sensitivity
Limits

□ Return to Solver Parameters Dialog
□ Outline Reports

It looks like Richmond is quite strong, while Carlton needs to hang their heads in shame. Either way, we now have optimized team ratings that we can use to continue building our model. Our next step is to find a way to map these team ratings to some outcome we actually want to know – like who would win a matchup between two teams of our choosing in the future. In order to do this we have a couple of options, but let's keep it simple and perform a simple linear regression using the resulting home team MOV as the Y variable and the logistic function (which we derived using the team strength ratings) as our X variable. Effectively, what we're going to do is ask Excel, via a linear regression, to explain the outcome margin of victory in column G using the probability output from our logistic function in column H. After clicking on "Data Analysis" and then "Regression", here's how I set this up in Excel:



Home MOV	Logistic Function	Game Result	Result Function	Prob Product	Log
26	0.9890	1	0.9890	1.93E-39	
12	0.				
25	0.				
50	0.				
16	0.				
34	0.				
82	0.				
-3	0.				
-29	0.				
36	0.				
52	0.				
-34	0.				
-16	0.				
-26	0.				
16	0.				
-51	0.1485	0	0.8515		
-23	0.6410	0	0.3590		

Regression

Input

Input Y Range: $G$4:$G$181

Input X Range: $H$4:$H$181

☑ Labels ☐ Constant is Zero

☐ Confidence Level: 95 %

Output options

○ Output Range:

● New Worksheet Ply:

○ New Workbook

Residuals

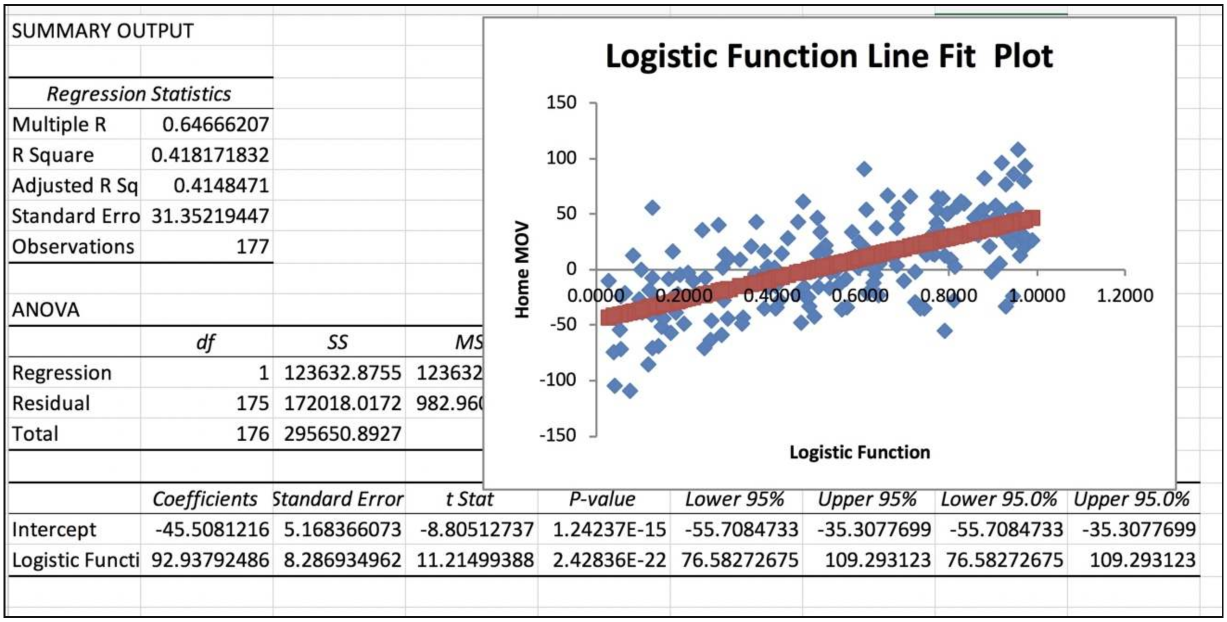
☐ Residuals ☐ Residual Plots

☐ Standardized Residuals ☑ Line Fit Plots

Normal Probability

☐ Normal Probability Plots

As you can see, we're trying to map home MOV from column G as explained by the Logistic Function from column H. I've also selected labels in the first row ("Labels") and asked Excel to include a chart in the output ("Line Fit Plots"). Here is the result:



SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.64666207
R Square	0.418171832
Adjusted R Sq	0.4148471
Standard Erro	31.35219447
Observations	177

ANOVA

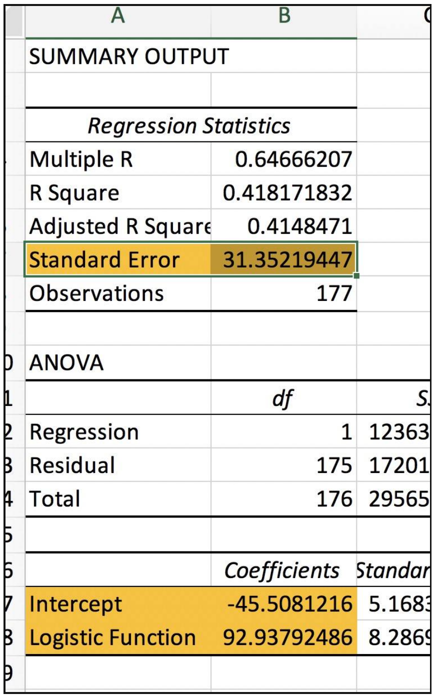
	df	*SS*	*MS*
Regression	1	123632.8755	123632
Residual	175	172018.0172	982.96
Total	176	295650.8927	

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-45.5081216	5.168366073	-8.80512737	1.24237E-15	-55.7084733	-35.3077699	-55.7084733	-35.3077699
Logistic Functi	92.93792486	8.286934962	11.21499388	2.42836E-22	76.58272675	109.293123	76.58272675	109.293123

That's a pretty nice regression line! We have a decent R Squared value, a low p-value indicating statistical significance, and a standard error of 31.35. Also, the residuals appear to be more or less normally distributed which is a key consideration when performing a linear regression. Having mapped our model to a useable output reasonably well, we can now use this information to make a prediction for an upcoming game.

Making Game Predictions

In order to use what we've just built to forecast the outcome of a future game, we'll need to get some information from the linear regression we just ran. Copy the following information highlighted in orange from the linear regression (standard error + coefficients):

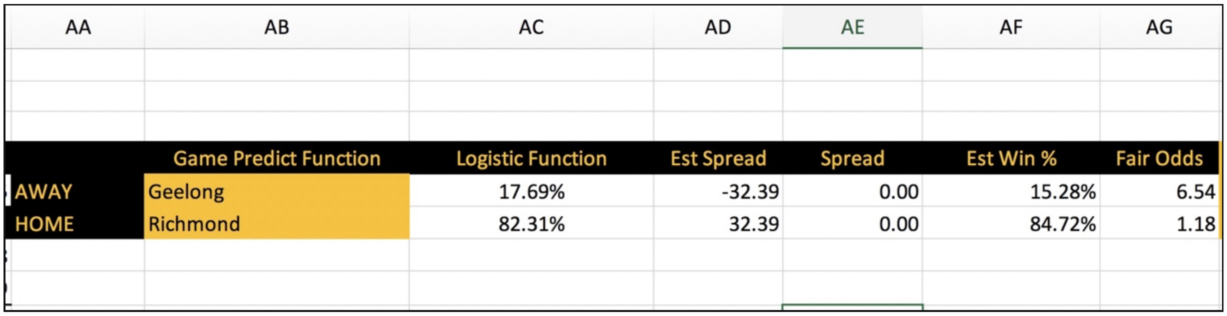


A	B	C
SUMMARY OUTPUT		
Regression Statistics		
Multiple R	0.64666207	
R Square	0.418171832	
Adjusted R Square	0.4148471	
Standard Error	31.35219447	
Observations	177	
ANOVA		
	df	*S*
Regression	1	12363
Residual	175	17201
Total	176	29565
	Coefficients	*Standar*
Intercept	-45.5081216	5.1683
Logistic Function	92.93792486	8.2869

Once we have them copied, let's paste them into our model sheet in the following cells:

> Standard Error: Cell Z5
>
> Regression Intercept: Cell Z7
>
> Logistic Function Coefficient: Cell Z8

Now we need to setup a future match, calculate the logistic function, derive the expected MOV with our linear regression, and then convert that to a win probability. It might sound like a lot to do, but it's not so bad. First I set up the two teams and calculate the logistic function. As you can see, it's exactly the same function as before, but we're now using telling Excel to use VLOOKUP on the teams we have selected for the matchup. You'll find this starting in column AB under "Game Predict Function":



AA	AB	AC	AD	AE	AF	AG
	Game Predict Function	Logistic Function	Est Spread	Spread	Est Win %	Fair Odds
AWAY	Geelong	17.69%	-32.39	0.00	15.28%	6.54
HOME	Richmond	82.31%	32.39	0.00	84.72%	1.18

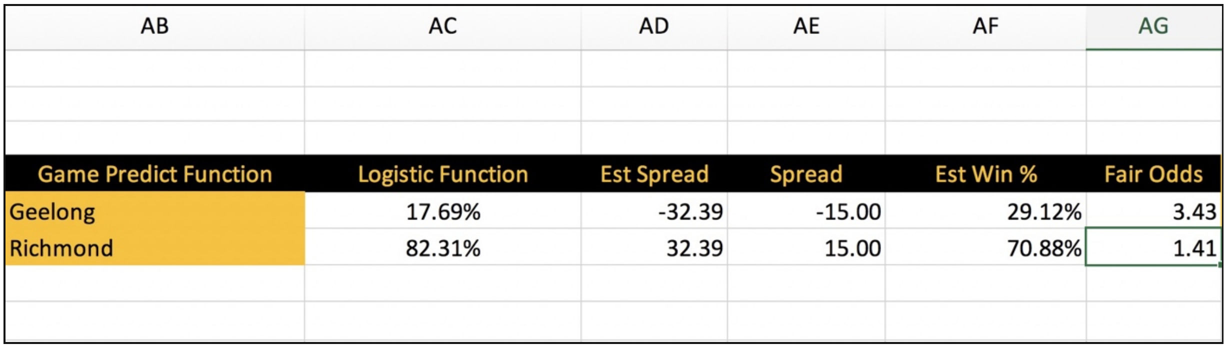
Richmond is at home and Geelong is away. Our logistic function returns 82.31% for the home team and 17.69% for the away team. This is our starting point. However, as you recall, we mapped this to an expected margin of victory earlier using a linear regression, so that's what we'll do now. Under "Est. Spread", we input our regression model for the home team:

```
=$Z$7+($Z$8*AC6)
```

This excel command simply uses the regression coefficients from our earlier linear regression and the home team's logistic function win % to estimate the home margin of victory. In this case, we expect Richmond to win by ~32 points. Looks like Geelong is going to have a tough day on the field. Geelong's expected MOV is simply the inverse of Richmond's, which is inputted into the model as (=–AD6). However, we're not quite done yet. There is one final step, and that is to take this margin of victory and convert it into a probability using an appropriate probability distribution. In this case, since we are modelling margin of victory, a normal distribution is a reasonable fit as we saw in our discussion of various distributions in the previous chapter. The probability of a home team win by any score, therefore, will use the NORMDIST command in Excel as shown (column AF):

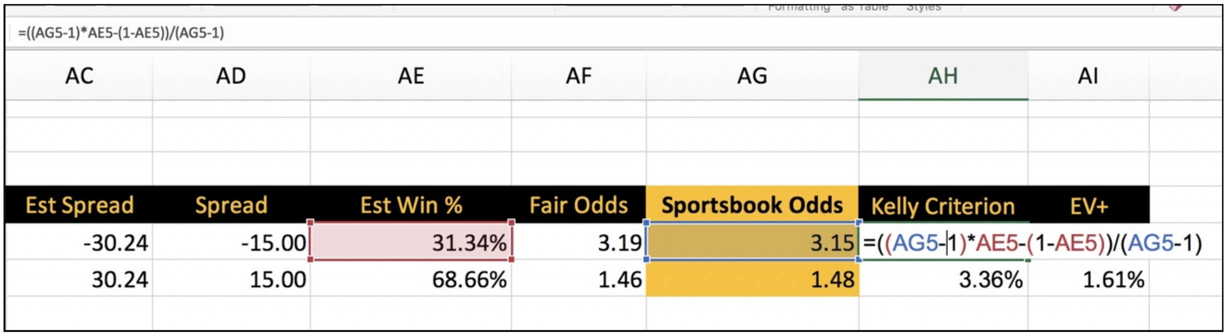
```
=1-NORMDIST(AE6,AD6,$Z$5,TRUE)
```

Here, the probability of Richmond winning by any score is approximately 84.72%.[2] Geelong's win probability is simply the probability of Richmond not winning, or (1-0.8472). It's also important to note that if we would rather get a probability for a specific point spread victory, like say 15 points, all we have to do is change the first value in the NORMDIST command from "0" to "15". We can do this by adjusting the cells in column AE under "Spread". A 15 spread indicates the line on that team to win is -15 points. This means a spread of (-15) is entered as 15. A spread of +15 is entered as (-15). Here is an example:



AB	AC	AD	AE	AF	AG
Game Predict Function	Logistic Function	Est Spread	Spread	Est Win %	Fair Odds
Geelong	17.69%	-32.39	-15.00	29.12%	3.43
Richmond	82.31%	32.39	15.00	70.88%	1.41

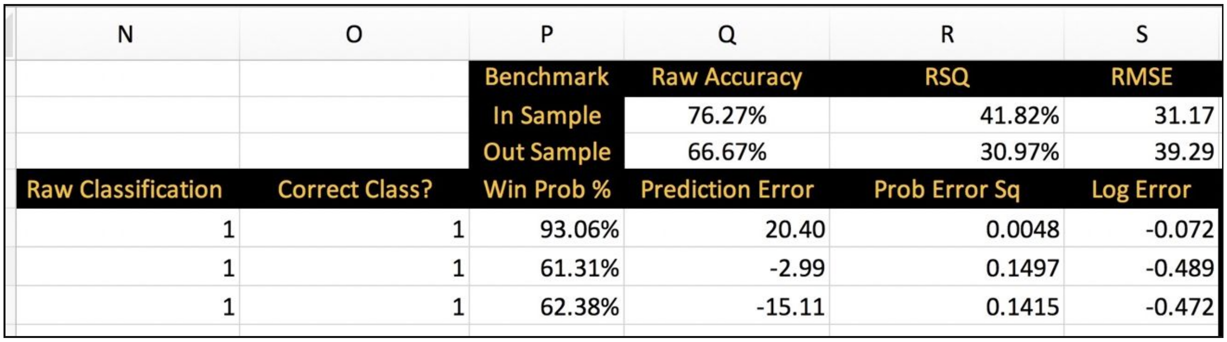
We can see that Richmond is expected to beat Geelong by 15 points approximately 70.88% of the time in this scenario. Finally, we can derive the fair odds for this game in the "Fair Odds" column by simply applying (=1/AF6) in cell AG6, showing the fair odds for Richmond at this point spread to be 1.41 in Decimal odds. The analogous process for Geelong produces fair odds of 3.43. From here, all that is left to do is to find a price offered by a current book line, and assess the value using either a standard formula or some variant of the Kelly Criterion. Let's suppose that the hypothetical line for this game currently posted is Richmond 1.48, Geelong 3.15. We can now calculate the value a couple of different ways:



=((AG5-1)*AE5-(1-AE5))/(AG5-1)

AC	AD	AE	AF	AG	AH	AI
Est Spread	Spread	Est Win %	Fair Odds	Sportsbook Odds	Kelly Criterion	EV+
-30.24	-15.00	31.34%	3.19	3.15	=((AG5-1)*AE5-(1-AE5))/(AG5-1)	
30.24	15.00	68.66%	1.46	1.48	3.36%	1.61%

Calculating the standard expected value (EV) as (Offered Price * Probability-1) returns a value of 4.90% expected edge for Richmond in this game. Calculating full Kelly Criterion, as pictured above, returns a more aggressive value of 10.21%. Just like that, we've created a model, optimized the parameters, forecasted a future game and estimated the expected value on each side. There's only one thing missing – a back test to see if our model is any good when judged by its “out of sample” predictive power. You can see some of the model benchmark metrics at the top of columns Q through W:



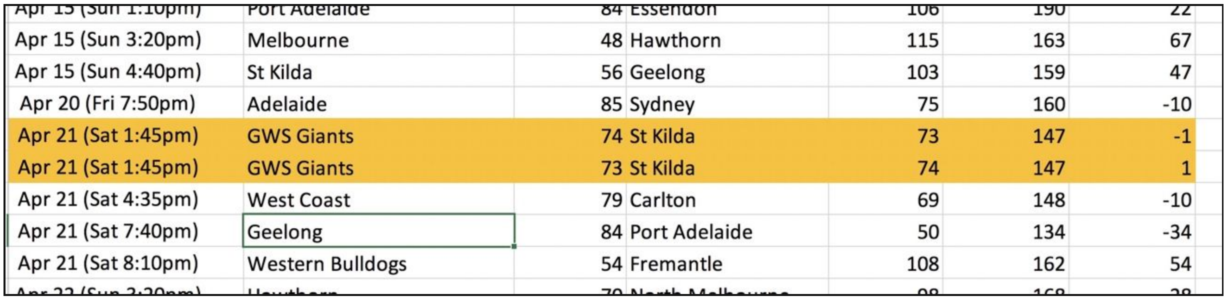
N	O	P	Q	R	S
		Benchmark	Raw Accuracy	RSQ	RMSE
		In Sample	76.27%	41.82%	31.17
		Out Sample	66.67%	30.97%	39.29
Raw Classification	Correct Class?	Win Prob %	Prediction Error	Prob Error Sq	Log Error
1	1	93.06%	20.40	0.0048	-0.072
1	1	61.31%	-2.99	0.1497	-0.489
1	1	62.38%	-15.11	0.1415	-0.472

The easiest way to accomplish a model backtest in Excel is a simple holdout sample, where we purposely exclude a select number of the most recent games from the model optimization process and see how the model performs on these previously unseen games. This is an important step, as a model that has already “seen” games in its optimization stages that we are now asking it to predict is actually describing the games rather than predicting them. In the same way we wouldn't teach a parrot a few phrases and then believe that we've taught it to speak English, we don't want a model that's only good at regurgitating game results it has already seen. We want our model to be decent at forecasting games it hasn't seen. Due to the importance of this element of modelling, I cover this in greater detail in Chapter 14. For now, let's just cover one more additional element of the BTM before moving on to the next model.

Dealing With Draws

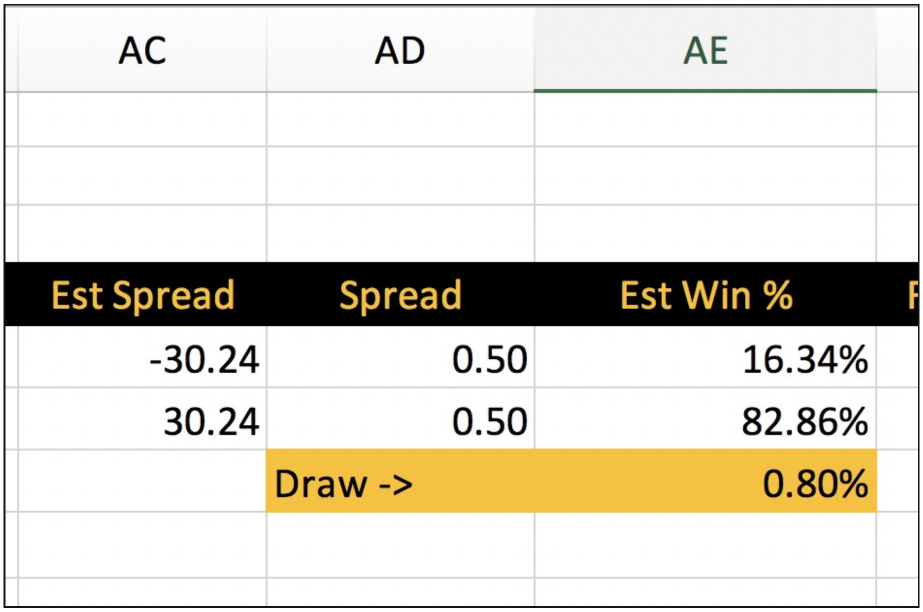
As I previously mentioned, the base configuration of this model has some difficulty dealing with draws. You can see that in the results function of the model, as we pick an either/or option to include in our log likelihood. Basically, as it is currently set up, the model has created a universe where one team has to win and one team has to lose. This presents obvious challenges for the sports bettor looking to apply this model to a sport where draws occur.

There are a few ways to deal with this, and which solution you choose depends a bit on the nature of the sport. In sports where draws occur but are relatively rare, I would suggest treating the draw as an outlier and eliminating it from the dataset. If the sport in question features a regular occurrence of draws like the English Premier League [EPL], I would alternatively suggest removing the draw observation from the dataset and replacing it with a 1 goal MOV victory for each team. The effect of this is that the two will cancel each other out, but still be factored into the optimization process which effectively allows the BTM to account for draws. In other words, include the game twice in the dataset, with a 1 goal win for the home squad, and a 1 goal win for the away squad respectively. Here is an example of inputting a draw from the AFL data set:



Apr 15 (Sun 1:10pm)	Port Adelaide	84	Essendon	106	190	22
Apr 15 (Sun 3:20pm)	Melbourne	48	Hawthorn	115	163	67
Apr 15 (Sun 4:40pm)	St Kilda	56	Geelong	103	159	47
Apr 20 (Fri 7:50pm)	Adelaide	85	Sydney	75	160	-10
Apr 21 (Sat 1:45pm)	GWS Giants	74	St Kilda	73	147	-1
Apr 21 (Sat 1:45pm)	GWS Giants	73	St Kilda	74	147	1
Apr 21 (Sat 4:35pm)	West Coast	79	Carlton	69	148	-10
Apr 21 (Sat 7:40pm)	Geelong	84	Port Adelaide	50	134	-34
Apr 21 (Sat 8:10pm)	Western Bulldogs	54	Fremantle	108	162	54
Apr 22 (Sun 3:20pm)	Hawthorn	70	North Melbourne	98	168	28

There is also the question of how to calculate the probability of a 3 way result using this model. The simplest way is to assign each team a point spread of 0.5, and then to take the remaining probability as the probability of a draw. In this example, I’ve shown the probability of a draw between Richmond and Geelong from the AFL. Each NORMDIST command in excel features a “0.5” instead of a zero. To calculate the draw, which we can consider to be the remaining probability in “the middle” of the distribution, we can use the formula [1-(Richmond Probability + Geelong Probability)] as highlighted in orange below:



AC	AD	AE
Est Spread	Spread	Est Win %
-30.24	0.50	16.34%
30.24	0.50	82.86%
	Draw ->	0.80%

That’s pretty much a wrap for the Bradley Terry model. In addition to walking you through how this particular model works, we’ve also gotten more familiar with the basic Excel spreadsheet model setup used for subsequent models. You’ll find the game prediction, model metrics and other elements of the upcoming models very similar. This consistency comes in handy when we eventually amalgamate the models together into an ensemble later on.

[1] My thanks to Marcus Buckland for bringing this to my attention.

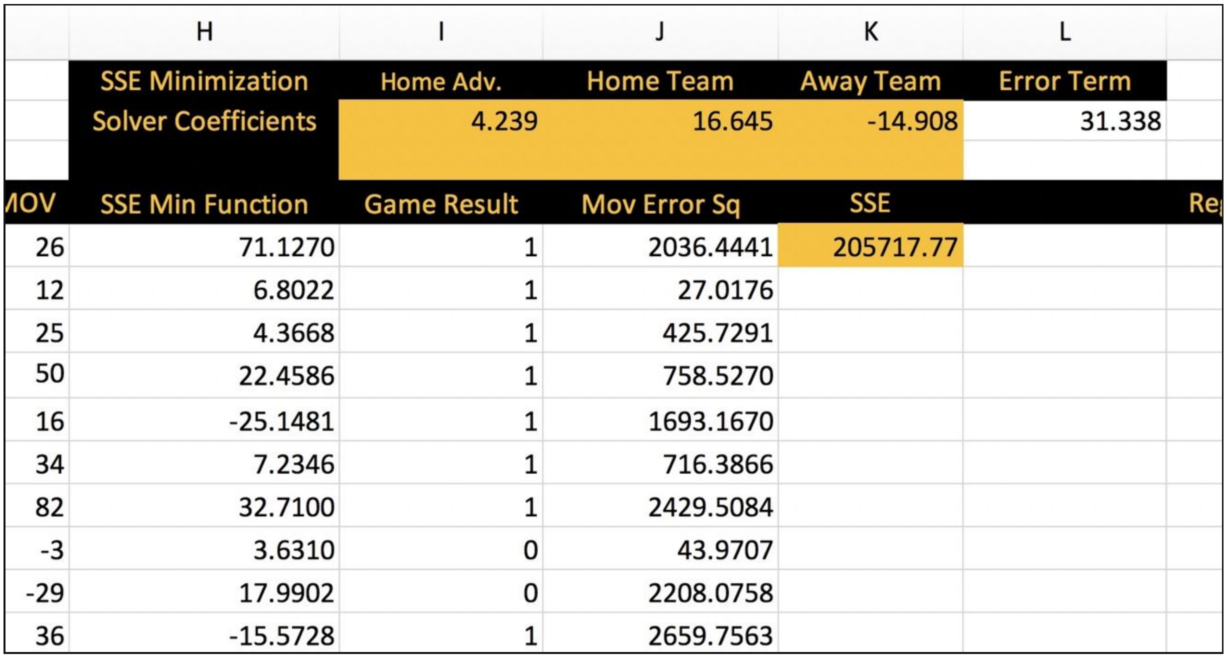
[2] These probabilities may vary slightly depending on how large you make your “in-sample” dataset partition.

THE TOOR MODEL

The Team OLS Optimized Rating [TOOR] model is an extension of the Bradley Terry model in some senses. It effectively borrows the logistic team strength ratings we derived with the BTM and then applies an ordinary least squares [OLS] optimization to them using Excel's Solver function to reduce the sum of the squared error [SSE]. Basically it is a linear regression model using the BTM team strength ratings optimized with OLS minimization. What that means for us is that with only a few slight changes to our BTM configuration we'll have a model that sees the sport we're trying to forecast in a slightly different way – and having multiple models with differing opinions is a good thing. It's also a breeze to setup now that we've already seen the BTM. Let's dive right in.

Getting Started

Start by opening the "TOOR Model" Excel file. We'll once again be using the same AFL data that we used previously in the BTM example. You'll notice that the logistic strength ratings in column U are the ones we derived using the Bradley Terry model. In fact, most of the spreadsheet should look very familiar after working through the last model, except for our model function and our new OLS minimization setup.



	H	I	J	K	L	
	SSE Minimization	Home Adv.	Home Team	Away Team	Error Term	
	Solver Coefficients	4.239	16.645	-14.908	31.338	
MOV	SSE Min Function	Game Result	Mov Error Sq	SSE		Re
26	71.1270	1	2036.4441	205717.77		
12	6.8022	1	27.0176			
25	4.3668	1	425.7291			
50	22.4586	1	758.5270			
16	-25.1481	1	1693.1670			
34	7.2346	1	716.3866			
82	32.7100	1	2429.5084			
-3	3.6310	0	43.9707			
-29	17.9902	0	2208.0758			
36	-15.5728	1	2659.7563			

Our logistic function from the BTM has been replaced by a linear regression formula (“SSE Min Function”) in column H that looks like this:

```
=$I$2+($J$2*VLOOKUP(D5,$T$5:$U$22,2,FALSE))+
($K$2*VLOOKUP(B5,$T$5:$U$22,2,FALSE))
```

This may look complicated, but really we’re just structuring the relationship between the logistic strength ratings for each team in a given game as:

```
[HFA]+[Home Logistic Rating]+[Away Logistic Rating]
```

Effectively, what we’re asking Excel to do is to look up each team playing in a given game, find their logistic strength rating, and then input those ratings into the formula above using the regression coefficient weightings that Solver optimizes for us. Cells I2, J2, and K2 are our linear regression coefficients (weights). Initially, we can just set each of them to 1.000 as a dummy variable. These will be the weights we'll ask Solver to optimize.

Next you'll notice that the result function in column J has changed to "Mov Error Sq" which is the difference between the actual MOV and our predicted MOV squared. This is calculated as follows:

```
=(G5-H5)^2
```

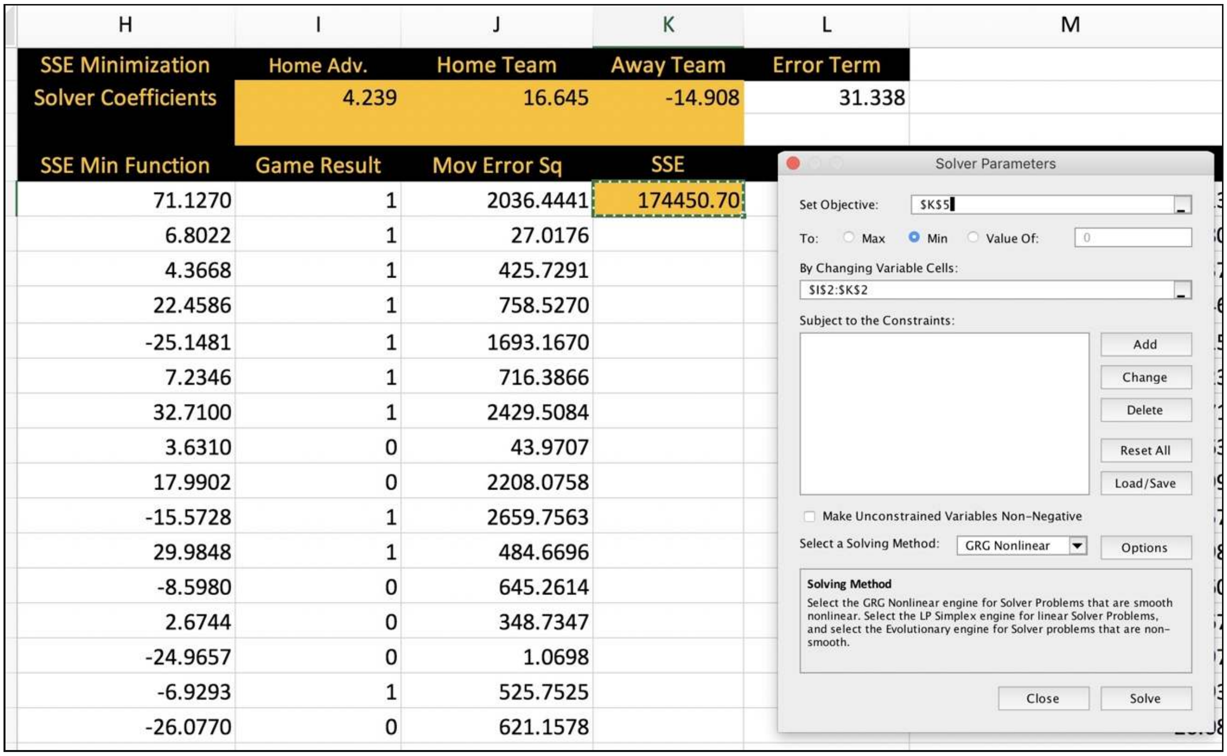
This gives us the squared error for each game prediction. Squaring is helpful because it removes the positive or negative nature of the error and allows us to add all the errors together in a meaningful way. If we didn't square the error, positive and negative values can cancel each other out. Next, in cell K5, the sum of the squared error [SSE] simply sums the error values from column J:

```
=SUM(J5:J182)
```

Once we've completed this, the model is ready to be optimized using Solver.

Optimizing the TOOR Model

Now we're ready to begin the optimization process. Once again, we're going to use Solver for this task, but with a slightly different configuration than before. Instead of maximizing the log likelihood as we did with the BTM, we are going to minimize the SSE in cell K5, using the dummy coefficients we created earlier. It should look like this:



H	I	J	K	L	M
SSE Minimization	Home Adv.	Home Team	Away Team	Error Term	
Solver Coefficients	4.239	16.645	-14.908	31.338	
SSE Min Function	Game Result	Mov Error Sq	SSE		
71.1270	1	2036.4441	174450.70		
6.8022	1	27.0176			
4.3668	1	425.7291			
22.4586	1	758.5270			
-25.1481	1	1693.1670			
7.2346	1	716.3866			
32.7100	1	2429.5084			
3.6310	0	43.9707			
17.9902	0	2208.0758			
-15.5728	1	2659.7563			
29.9848	1	484.6696			
-8.5980	0	645.2614			
2.6744	0	348.7347			
-24.9657	0	1.0698			
-6.9293	1	525.7525			
-26.0770	0	621.1578			

Solver Parameters

Set Objective: $K$5

To: ○ Max ● Min ○ Value Of: 0

By Changing Variable Cells: $I$2:$K$2

Subject to the Constraints:

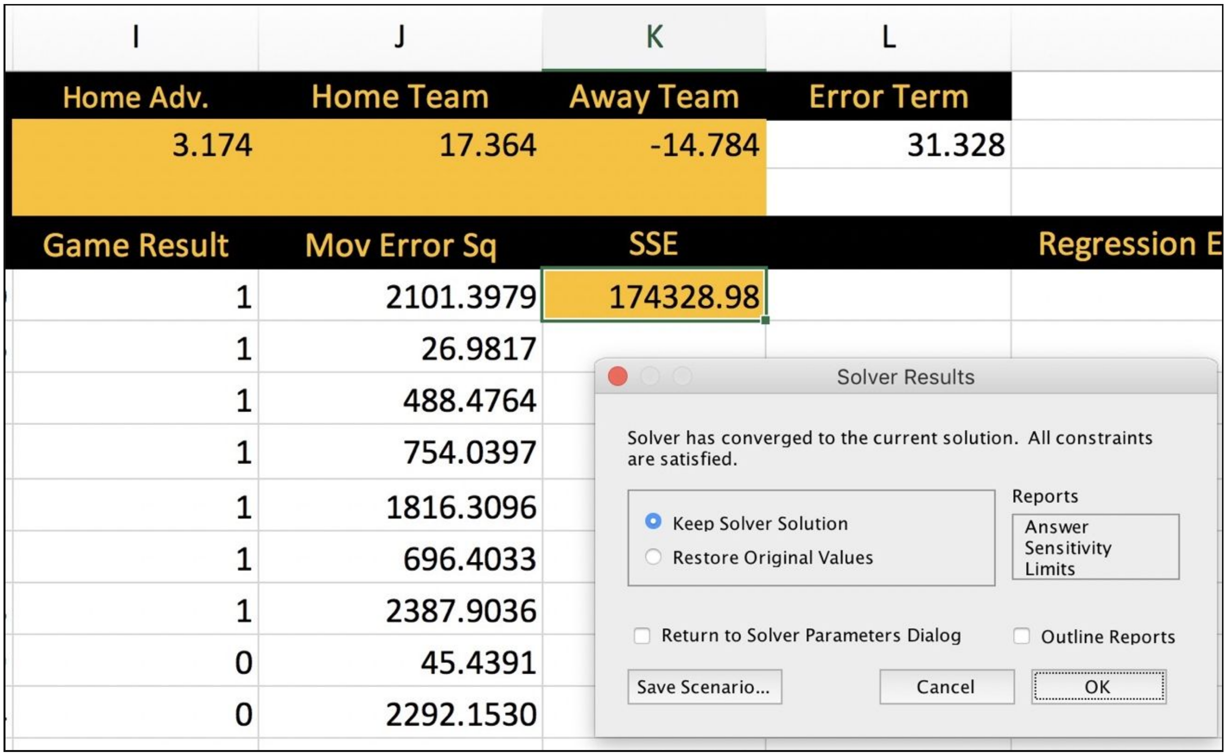
□ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

As you can see, we’re simply asking Solver to minimize the value in cell K5 by tweaking the dummy coefficients we’ve setup in cells I2, J2 and K2. Time to click ‘Solve’ and let Excel do its thing.



I	J	K	L	
Home Adv.	Home Team	Away Team	Error Term	
3.174	17.364	-14.784	31.328	
Game Result	Mov Error Sq	SSE		Regression E
1	2101.3979	174328.98		
1	26.9817			
1	488.4764			
1	754.0397			
1	1816.3096			
1	696.4033			
1	2387.9036			
0	45.4391			
0	2292.1530			

Solver Results

Solver has converged to the current solution. All constraints are satisfied.

● Keep Solver Solution

○ Restore Original Values

Reports

Answer
Sensitivity
Limits

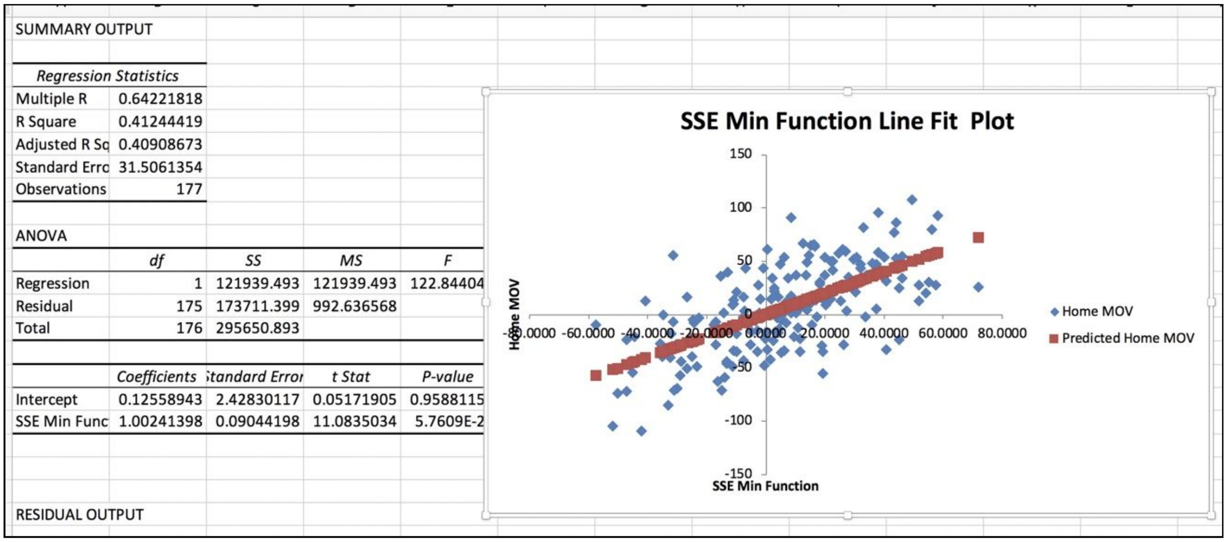
□ Return to Solver Parameters Dialog

□ Outline Reports

It appears Solver has found a solution. You'll notice our dummy coefficients are now optimized to the weights that minimize the model errors. We've optimized the TOOR model parameters and now it's time to continue on with the rest of the process we've used before – mapping these estimates to an outcome variable and then using an appropriate probability distribution to convert that to expected probabilities. This is the same process as was done for the BTM, but we'll walk through it again, just for fun.

First things first, let's setup another regression, using the actual home team MOV (column G) as the Y variable and the TOOR expected home MOV (column H) as the X variable.

Once again, here we're asking the regression "how well can you explain a game's actual margin of victory (column G) using our TOOR model's expected MOV (column H)?" I know that may sound like a simplistic way to describe it, but I've found it to be a helpful way to conceptualize regressions as you explore different ideas. Once our regression is ready to go, simply click "OK" and we'll see how it performs:



SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.64221818
R Square	0.41244419
Adjusted R Sq	0.40908673
Standard Errc	31.5061354
Observations	177

ANOVA

	df	*SS*	*MS*	*F*
Regression	1	121939.493	121939.493	122.84404
Residual	175	173711.399	992.636568	
Total	176	295650.893		

	Coefficients	*Standard Error*	*t Stat*	*P-value*
Intercept	0.12558943	2.42830117	0.05171905	0.9588115
SSE Min Func	1.00241398	0.09044198	11.0835034	5.7609E-2

RESIDUAL OUTPUT

That’s looking alright. It’s comparable, but not significantly better than our BTM. That’s okay though! Some sports are like that. In other sports, one may be significantly better than the other or vice versa. This is one of the many reasons it’s a good idea to develop an assortment of models rather than simply one model. In any event, we can copy and paste the standard error and regression coefficients into our TOOR model sheet and complete the rest of the process, just as we did for the BTM. You’ll find the game prediction function in the same spot as last time, around column AA. Everything works comparatively the same as before.

First we use the “SSE Min Function” from column H, in the same way that we did before with the BTM. Then, we use our linear regression formula to adjust the SSE Min function estimate, bringing our MOV estimate of this game to 27.30 points in favour of Richmond (column AC). You can see that this is slightly more conservative than the BTM model which forecast a 30.24 point MOV for Richmond. Calculating the point spread and the value based on the offered odds is the same as with the BTM.

THE GSSD MODEL

So far we've looked at two models: BTM and TOOR. The BTM forecasted using a series of strength ratings on a logistic scale, and the TOOR model applied an OLS minimization to those ratings to produce an expected MOV. For the Game Scores Standard Deviation [GSSD] model, we'll focus on something a bit different: an OLS minimization using average team scores for home, against home, for away, and against away. Don't be fooled by the relative simplicity of this however – this is frequently among the best performing generalized sport models in this book. I should also mention that wrangling the data in Excel for this model is a bit of a pain, so this setup is going to require some additional elbow grease to fully construct on your own. Not to worry though, I've got the whole thing entirely built, so feel free to simply pick up where I've left off. If you're up for the challenge you can tinker with it yourself. I leave that up to you.

What we'll be doing is using a linear regression structure with Solver for OLS optimization, but this time we'll be solving for coefficients attached to game score averages. The model structure is very similar to the TOOR model, but looks like this:

```
Intercept +PFH+PAH+PFA+PAA
```

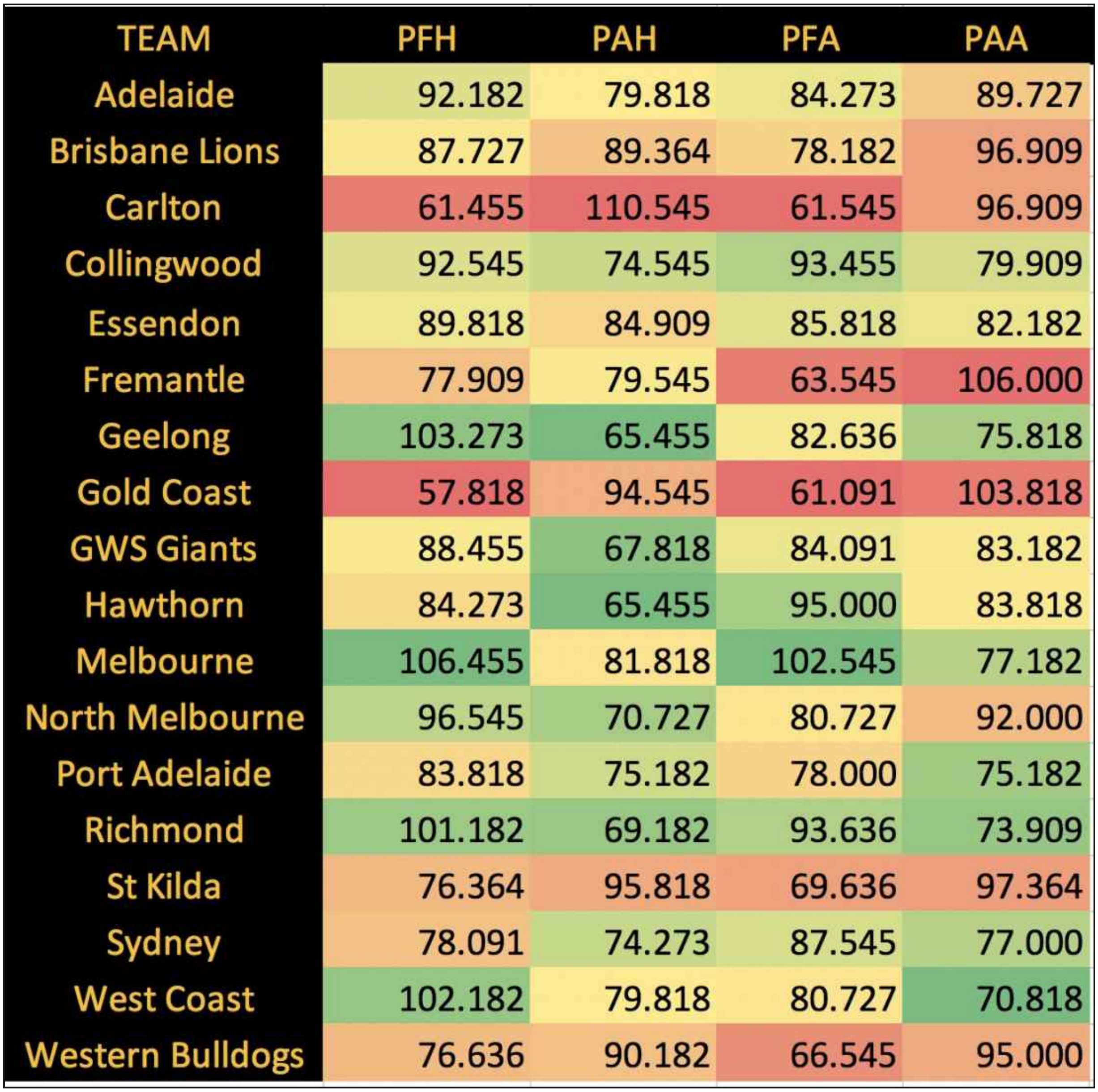
You'll find this formula in column H again under "SSE Min function". Notice that the SSE calculations are done the same way as with the TOOR model. Also note that the coefficient names in I1, J1, K1, L1 and M1 have been changed to reflect the new inputs we'll be using. Hopefully, other than the process of generating the inputs, the rest of this sheet looks rather familiar.

The first thing we want to do is take our AFL game results dataset and create a way to sort each team individually so that we can get these individual scoring averages for each team in a format that

Excel can use. To do this, I first created a sheet named “Game Data” and then created an individual sheet for each team in the league. These sheets reference the “Game Data” sheet, find the relevant team and then sort the results for each team by home and away locations. The result is that we have points scored for and against each team, either at home or away. We then average these stats to produce four statistics for each team: PFH, PAH, PFA, PAA:

- PFH: average points for home
- PAH: average points against home
- PFA: average points for away
- PAA: average points against away

To accomplish this I’ve used a combination of Excel commands mostly involving INDEX, ARRAY and ROW. It was a bit cumbersome to program this in Excel, but the solution I’ve come up with gets the job done. Our desired end result appears in columns T to X of the GSSD sheet:

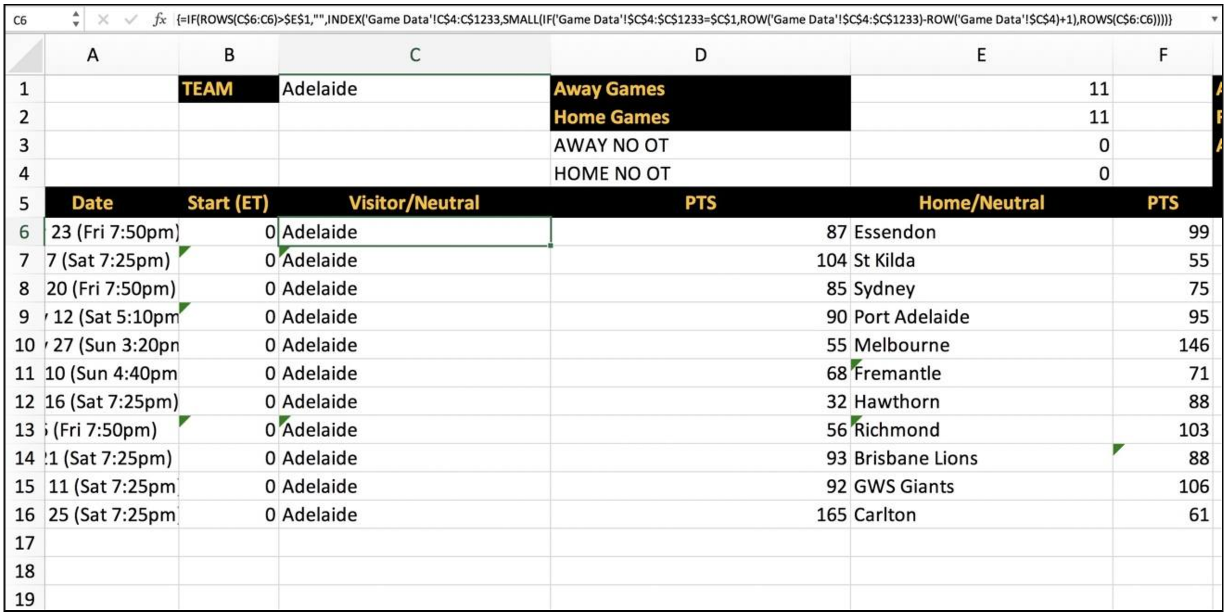


TEAM	PFH	PAH	PFA	PAA
Adelaide	92.182	79.818	84.273	89.727
Brisbane Lions	87.727	89.364	78.182	96.909
Carlton	61.455	110.545	61.545	96.909
Collingwood	92.545	74.545	93.455	79.909
Essendon	89.818	84.909	85.818	82.182
Fremantle	77.909	79.545	63.545	106.000
Geelong	103.273	65.455	82.636	75.818
Gold Coast	57.818	94.545	61.091	103.818
GWS Giants	88.455	67.818	84.091	83.182
Hawthorn	84.273	65.455	95.000	83.818
Melbourne	106.455	81.818	102.545	77.182
North Melbourne	96.545	70.727	80.727	92.000
Port Adelaide	83.818	75.182	78.000	75.182
Richmond	101.182	69.182	93.636	73.909
St Kilda	76.364	95.818	69.636	97.364
Sydney	78.091	74.273	87.545	77.000
West Coast	102.182	79.818	80.727	70.818
Western Bulldogs	76.636	90.182	66.545	95.000

You can see above that we have all of our metrics sorted by team and by home/away situation, which is exactly what we want. I like to colour code the columns so that I can quickly see who is strong and who is weak as shown in the image above. This is once again accomplished using Excel's conditional formatting function and can provide a visual "sanity check" when making sure there are no egregious errors in the model's configuration. As an example, previous models have put Richmond at the top of the league so we can easily scan the colour coded rows and check that Richmond's statistics are dark green (strong) more or less across the board. It appears that they are. Carlton likewise has average scores highlighted in red, reassuring us that they remain likely to take their place at the bottom of the ratings.

Getting Started

The first step is to populate the “Game Data” sheet with our game results data from the AFL. You’ll see that this has already been done. Next, we setup an individual sheet for each team in the league that looks like this:



C6 {=IF(ROWS(C$6:C6)>$E$1,"",INDEX('Game Data'!C$4:C$1233,SMALL(IF('Game Data'!$C$4:$C$1233=$C$1,ROW('Game Data'!$C$4:$C$1233)-ROW('Game Data'!$C$4)+1),ROWS(C$6:C6))))}

	A	B	C	D	E	F
1		TEAM	Adelaide	Away Games	11	
2				Home Games	11	
3				AWAY NO OT	0	
4				HOME NO OT	0	
5	Date	Start (ET)	Visitor/Neutral	PTS	Home/Neutral	PTS
6	23 (Fri 7:50pm)	0	Adelaide	87	Essendon	99
7	7 (Sat 7:25pm)	0	Adelaide	104	St Kilda	55
8	20 (Fri 7:50pm)	0	Adelaide	85	Sydney	75
9	12 (Sat 5:10pm	0	Adelaide	90	Port Adelaide	95
10	27 (Sun 3:20pn	0	Adelaide	55	Melbourne	146
11	10 (Sun 4:40pm	0	Adelaide	68	Fremantle	71
12	16 (Sat 7:25pm)	0	Adelaide	32	Hawthorn	88
13	(Fri 7:50pm)	0	Adelaide	56	Richmond	103
14	1 (Sat 7:25pm)	0	Adelaide	93	Brisbane Lions	88
15	11 (Sat 7:25pm	0	Adelaide	92	GWS Giants	106
16	25 (Sat 7:25pm	0	Adelaide	165	Carlton	61
17						
18						
19						

It seems simple enough, but it takes a fair bit of work to achieve in Excel. Here is the utterly gigantic formula I used for each respective column:

```
=IF(ROWS(C$6:C6)>$E$1,"",INDEX('Game
Data'!C$4:C$1233,SMALL(IF('Game
Data'!$C$4:$C$1233=$C$1,ROW('Game        Data'!$C$4:$C$1233)-
ROW('Game Data'!$C$4)+1),ROWS(C$6:C6))))
```

I’ve populated this across the columns so that it catches each column of data from the “Game Results” sheet, and have done this twice on each individual team sheet: once for their home results and once for their away results. It’s also very important to know that if you want to change or build this yourself, you must enter it as an array function by pressing CRTL+SHIFT+ENTER after inputting the formula into the cells.

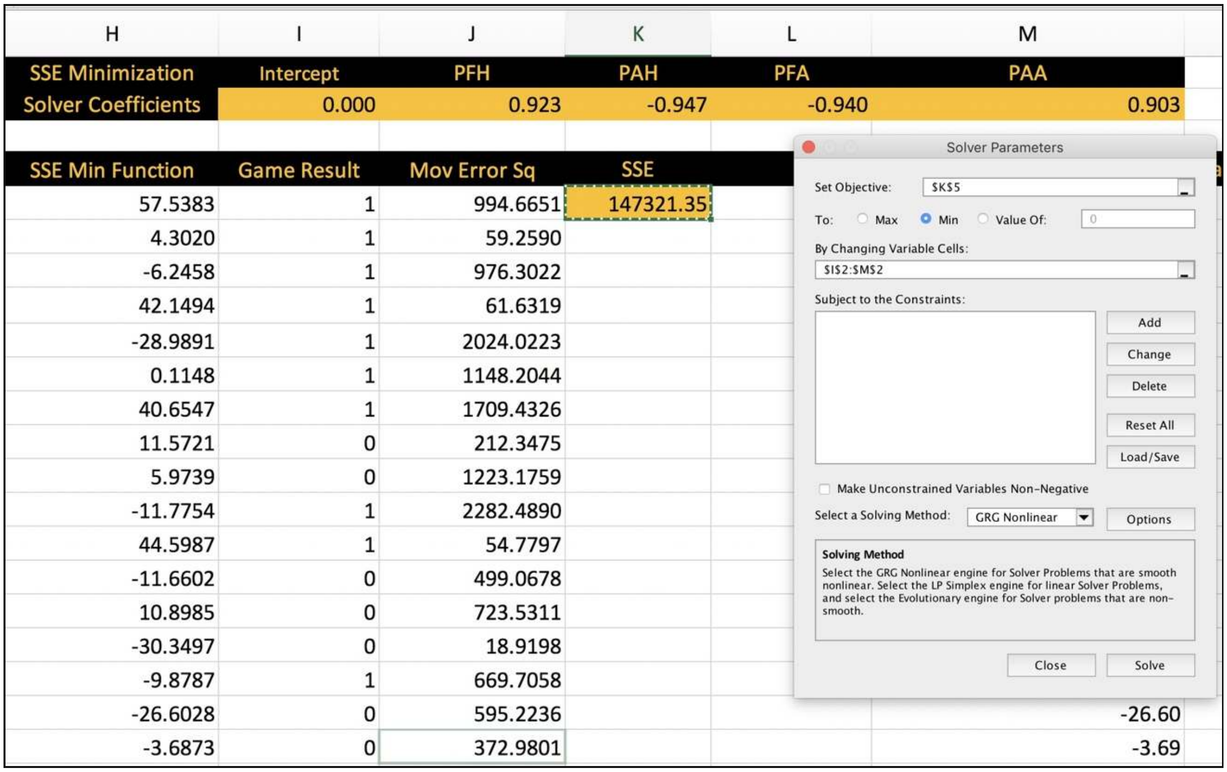
Hard to believe that this entire formula does something so simple right? This should be a big selling point for anyone thinking of learning a more advanced software program like Python or R.

Anyways, as I mentioned before, I’ve already done all of this for you in the example spreadsheets so there is no need to curse Excel as you try to work your way through this on your own. I just thought I’d show you what I’ve done so you can replicate it on your own if you’d like. If you want to modify these sheets for a different sport altogether, the only elements that need to be changed are the name of each individual team sheet itself (name it after the team you are tracking) and cell C1 (the team name) in each sheet. Provided that you format the input data for the “Game Data” sheet so that the columns match, this should work without any issues.

Once all these sheets are complete, we simply transfer the relevant team statistic averages back to our main sheet “GSSD Model” and input them into columns U through X for each team. Once that is complete, we’re ready to get back to the main GSSD sheet and optimize the model using Solver. I’ll make the assumption that you are not planning on trying to build these from scratch and are instead content to move on with what I’ve already created. Time to optimize the model.

Optimizing the GSSD Model

Click on Solver and configure it as shown below. We’re asking Solver to optimize the coefficients in I2, J2, K2, L2 and M2 in such a way that it reduces the sum of the squared error found in cell K5.



H	I	J	K	L	M
SSE Minimization	Intercept	PFH	PAH	PFA	PAA
Solver Coefficients	0.000	0.923	-0.947	-0.940	0.903
SSE Min Function	Game Result	Mov Error Sq	SSE		
57.5383	1	994.6651	147321.35		
4.3020	1	59.2590			
-6.2458	1	976.3022			
42.1494	1	61.6319			
-28.9891	1	2024.0223			
0.1148	1	1148.2044			
40.6547	1	1709.4326			
11.5721	0	212.3475			
5.9739	0	1223.1759			
-11.7754	1	2282.4890			
44.5987	1	54.7797			
-11.6602	0	499.0678			
10.8985	0	723.5311			
-30.3497	0	18.9198			
-9.8787	1	669.7058			
-26.6028	0	595.2236			-26.60
-3.6873	0	372.9801			-3.69

Solver Parameters

Set Objective: $K$5

To: ○ Max ● Min ○ Value Of: 0

By Changing Variable Cells: $I$2:$M$2

Subject to the Constraints:

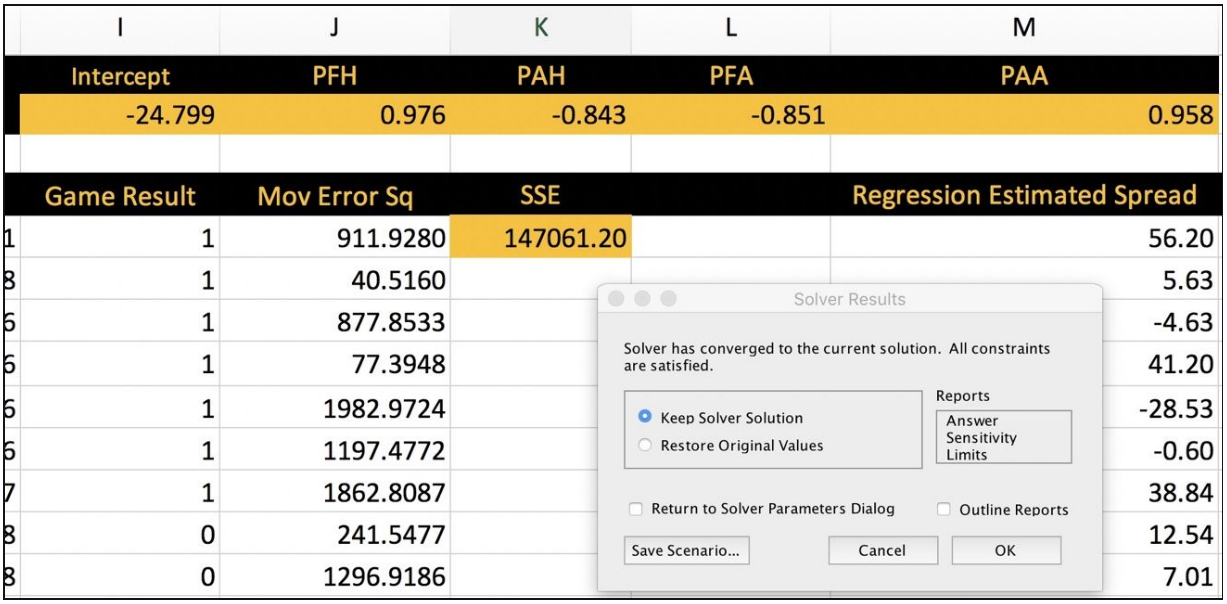
□ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Once you’re ready, click solve and Excel will begin optimizing the coefficients in order to properly weight our model input variables. Since this model estimates MOV from the home team’s perspective, it shouldn’t come as a surprise to see that Excel will find the optimal PFH and PAA coefficients to be positive numbers while PAH and PFA are negative. When Solver arrives at a solution you should see the coefficients have changed as well as a Solver popup box like this:



	I	J	K	L	M
	Intercept	PFH	PAH	PFA	PAA
	-24.799	0.976	-0.843	-0.851	0.958
	Game Result	Mov Error Sq	SSE		Regression Estimated Spread
1	1	911.9280	147061.20		56.20
8	1	40.5160			5.63
6	1	877.8533			-4.63
6	1	77.3948			41.20
6	1	1982.9724			-28.53
6	1	1197.4772			-0.60
7	1	1862.8087			38.84
8	0	241.5477			12.54
8	0	1296.9186			7.01

Solver Results

Solver has converged to the current solution. All constraints are satisfied.

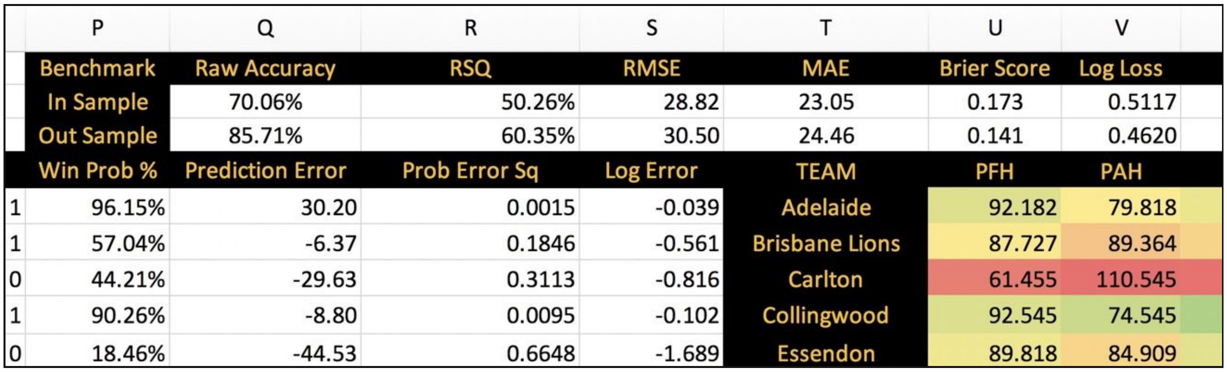
● Keep Solver Solution
○ Restore Original Values

Reports
Answer
Sensitivity
Limits

□ Return to Solver Parameters Dialog
□ Outline Reports

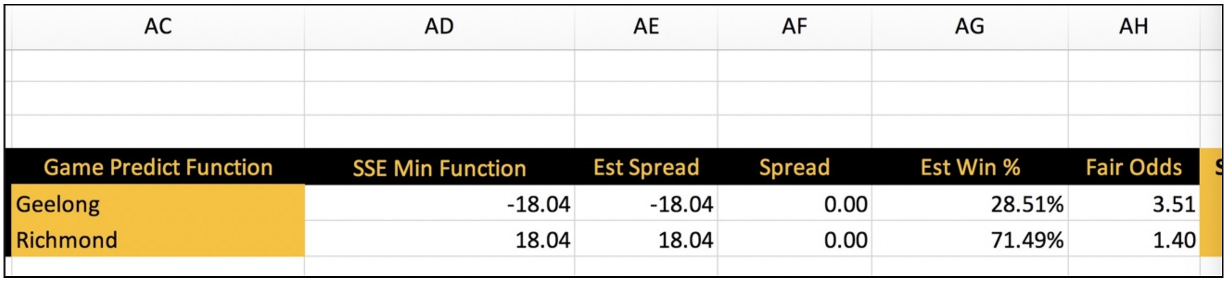
Next, we perform a linear regression just as we did previously to map the model predictions for MOV to actual MOV. For this model, Column G will be the Y variable and column H will be the X variable. As we did with other models, take the resulting regression coefficients as well as standard error and paste them into the main GSSD sheet in column AA.

Once this is complete, you can take a look at the model metrics at the top of columns Q through V, and see how the model performed. In this case it appears the model has done pretty well on this sample of games.



	P	Q	R	S	T	U	V
	Benchmark	Raw Accuracy	RSQ	RMSE	MAE	Brier Score	Log Loss
	In Sample	70.06%	50.26%	28.82	23.05	0.173	0.5117
	Out Sample	85.71%	60.35%	30.50	24.46	0.141	0.4620
	Win Prob %	Prediction Error	Prob Error Sq	Log Error	TEAM	PFH	PAH
1	96.15%	30.20	0.0015	-0.039	Adelaide	92.182	79.818
1	57.04%	-6.37	0.1846	-0.561	Brisbane Lions	87.727	89.364
0	44.21%	-29.63	0.3113	-0.816	Carlton	61.455	110.545
1	90.26%	-8.80	0.0095	-0.102	Collingwood	92.545	74.545
0	18.46%	-44.53	0.6648	-1.689	Essendon	89.818	84.909

Making game predictions with this individual model is the same as with previous models - you'll find the "Game Predict Function" starting in column AC. Simply input the relevant teams, spread (if applicable) and sportsbook price and you'll have a game forecast output in column AG with estimations of bet value following in subsequent columns.



AC	AD	AE	AF	AG	AH
Game Predict Function	SSE Min Function	Est Spread	Spread	Est Win %	Fair Odds
Geelong	-18.04	-18.04	0.00	28.51%	3.51
Richmond	18.04	18.04	0.00	71.49%	1.40

I've breezed through this part of the GSSD sheet a bit quickly because of its similarity to previous models we've already seen. Hopefully by now the general layout of each model sheet and its operation is starting to make sense! If any of this is confusing, I'd recommend going back and looking at the earlier explanation of the BTM and TOOR models, as the prediction function columns are quite similar.

The Z Score Deviation [ZSD] model provides another slightly different way to model sport results by comparing team performance to the mean and standard deviation for league scoring at home and away. Using Z scores, we can estimate how far from the league mean expectation a team's performance should be forecasted to be. Of particular note with this model setup is that the output is an expected team score. This can be used to generate expected margins of victory, total game scores, or individual team scores, making it a versatile tool for a number of different bet types. Also, no more crazy Excel formulas to wrangle data! This model is reasonably neat and tidy.

Getting Started

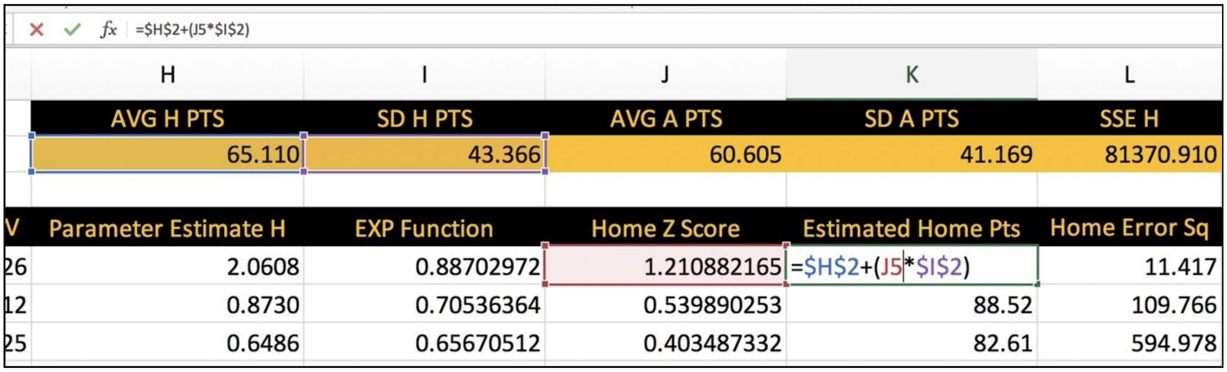
Let's start by opening up the "ZSD" Excel file. You'll notice many familiar elements: columns A through G once again feature game result data, we have some team ratings in columns AB and AC, and the game prediction function parts of the sheet are the same as what we've seen before. The main differences appear in columns H through Q. You'll see that at the top of the columns starting in cell H2, we have statistics showing the average home points (for all teams at home), the standard deviation for average home points, the average away points and the standard deviation for that as well. As I mentioned before, we'll be using these statistics along with the team ratings to forecast each team's expected points/runs/goals. Once we've completed the model setup, we'll fire up Solver and optimize the team ratings to maximize the model's performance.

Estimating Points/Runs/Goals

Column H is our home point parameter estimate function - which is half of our basic model structure. The Excel command for this structure looks like this:

```
=$O$2+VLOOKUP(D5,$AA$5:$AC$22,2,FALSE)-
VLOOKUP(B5,$AA$5:$AC$22,2,FALSE)
```

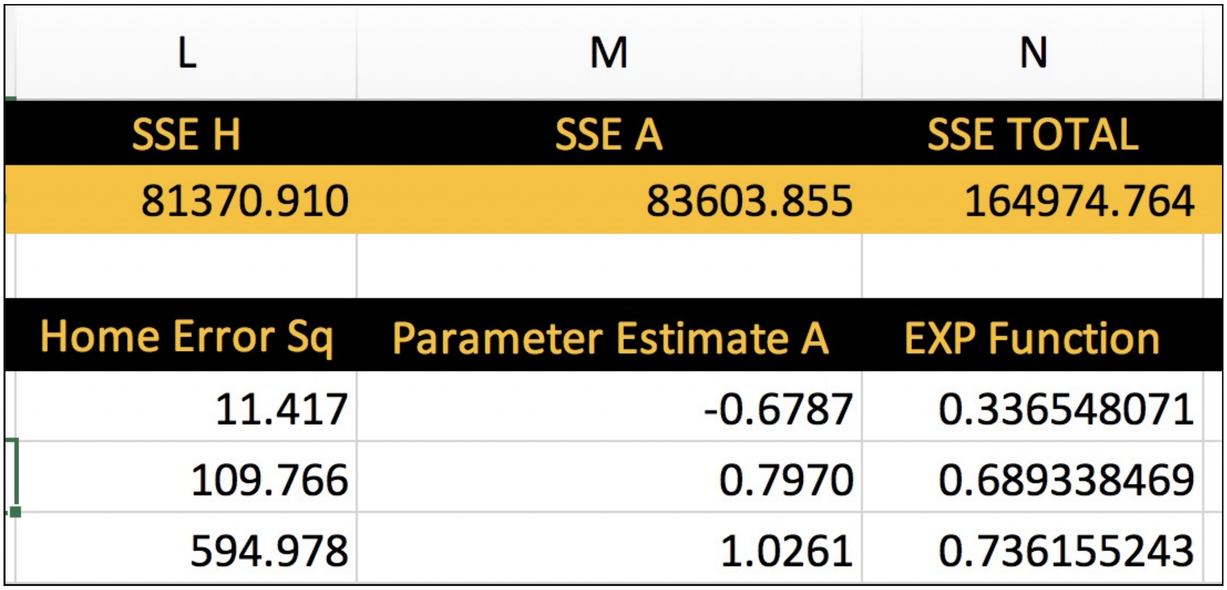
What we’re doing here is effectively asking Excel to lookup the home adjustment factor, the home team rating when at home and the away team rating when away and output a strength rating. Column I then applies a logistic function and column J converts that into a Z score. Column K then takes this Z score combined with the average points and standard deviation to produce a point estimate for the home team. Column L then measures the difference between our forecast and the observed values and squares that difference. Finally, in cell L2 we sum the values in column L.



```
=$H$2+(J5*$I$2)
```

	H	I	J	K	L
	AVG H PTS	SD H PTS	AVG A PTS	SD A PTS	SSE H
	65.110	43.366	60.605	41.169	81370.910
V	Parameter Estimate H	EXP Function	Home Z Score	Estimated Home Pts	Home Error Sq
26	2.0608	0.88702972	1.210882165	=$H$2+(J5*$I$2)	11.417
12	0.8730	0.70536364	0.539890253	88.52	109.766
25	0.6486	0.65670512	0.403487332	82.61	594.978

This process is repeated comparably for the away team in columns M through Q, as you’ll see when you make your way around the sheet. Finally the two SSE values in cells L2 and M2 are added together to produce a total SSE value in cell N2. This is the SSE value we will use solver to minimize by adjusting the team ratings in columns AB and AC.

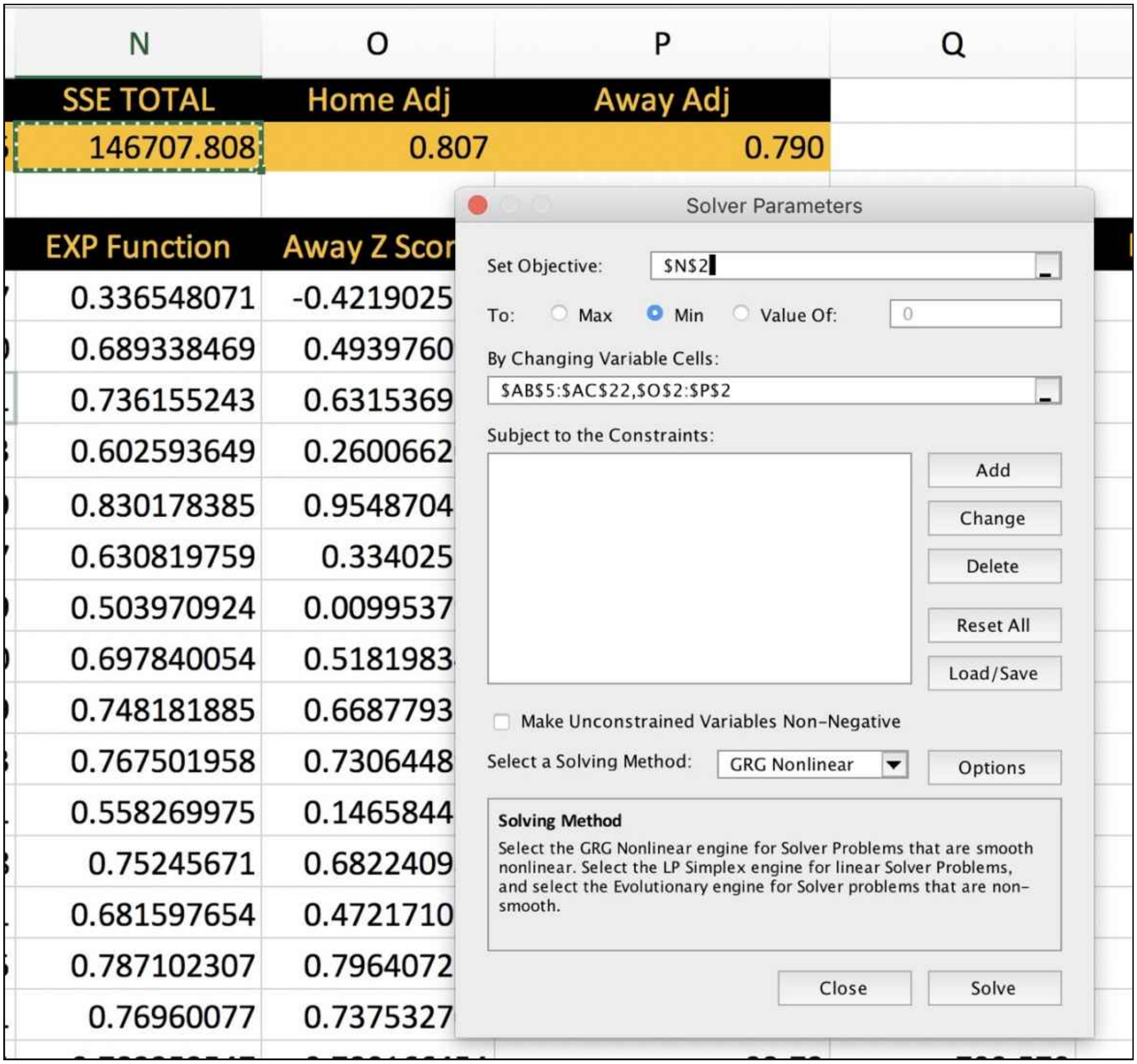


L	M	N
SSE H	SSE A	SSE TOTAL
81370.910	83603.855	164974.764
Home Error Sq	Parameter Estimate A	EXP Function
11.417	-0.6787	0.336548071
109.766	0.7970	0.689338469
594.978	1.0261	0.736155243

Once all of this is setup we're ready to optimize the team ratings using Solver.

Optimizing the ZSD Model

Let's click on Solver to begin optimizing the ratings. Once it loads, configure it as shown below. We want to reduce the total SSE in cell N2 using the team ratings in columns AB and AC, as well as the home and away adjustment factors found in cells O2 and P2:



N	O	P	Q
SSE TOTAL	Home Adj	Away Adj	
146707.808	0.807	0.790	
EXP Function	Away Z Scor		
0.336548071	-0.4219025		
0.689338469	0.4939760		
0.736155243	0.6315369		
0.602593649	0.2600662		
0.830178385	0.9548704		
0.630819759	0.334025		
0.503970924	0.0099537		
0.697840054	0.5181983		
0.748181885	0.6687793		
0.767501958	0.7306448		
0.558269975	0.1465844		
0.75245671	0.6822409		
0.681597654	0.4721710		
0.787102307	0.7964072		
0.76960077	0.7375327		

Solver Parameters

Set Objective: $N$2

To: ○ Max ● Min ○ Value Of: 0

By Changing Variable Cells: $AB$5:$AC$22,$O$2:$P$2

Subject to the Constraints:

□ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

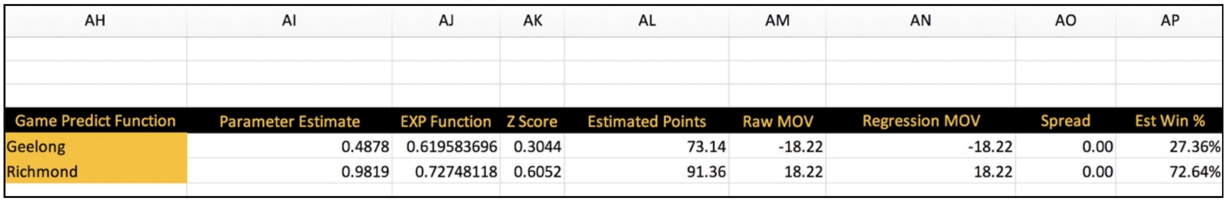
When the settings are ready to go, click solve and Excel will begin optimizing the ratings.

Output Mapping/Linear Regression

Once solver has finished optimizing the ratings, it’s time to use a linear regression to map our expected home MOV in column R to the actual observed MOV in column G. We’ve done this a few times already so you should be able to handle it without too many issues. Once again, open a linear regression from the Data Analysis Toolpak, using the G column as the Y variable and the R column as the X variable. We’ll then copy the regression coefficients and standard error back into the main ZSD sheet in column AF.

I should also point out that you could change the output in column R to a game total by adding the home and away scores together instead of deriving the point differential. Since we’re working through MOV models though we’ll stick with that to make it as easy to understand as possible. I just thought I’d point out that you can tailor this model’s output to things other than MOV. I would encourage to you experiment with this on your own.

Finishing up with the ZSD model, you’ll find the “Game Prediction” function starting in column AH. Other than the particulars of the ZSD model structure, the rest of the game prediction columns operate the same as we’ve seen before.



AH	AI	AJ	AK	AL	AM	AN	AO	AP
Game Predict Function	Parameter Estimate	EXP Function	Z Score	Estimated Points	Raw MOV	Regression MOV	Spread	Est Win %
Geelong	0.4878	0.619583696	0.3044	73.14	-18.22	-18.22	0.00	27.36%
Richmond	0.9819	0.72748118	0.6052	91.36	18.22	18.22	0.00	72.64%

The Power Rank Points [PRP] model is perhaps the most conceptually basic of all the models we've seen so far. It rates teams by their raw point/goal/run strength, making it akin to old-school power rating forecasts. It is similar to a method of estimating team strength explained by *Mathletics*[1] author Wayne Winston, but has a few subtle differences that make it worthy of standalone demonstration. Like the ZSD model, it can be configured to produce an output that is either margin of victory, individual team points, or total game score. This makes it versatile for a number of different wager types we may want to forecast.

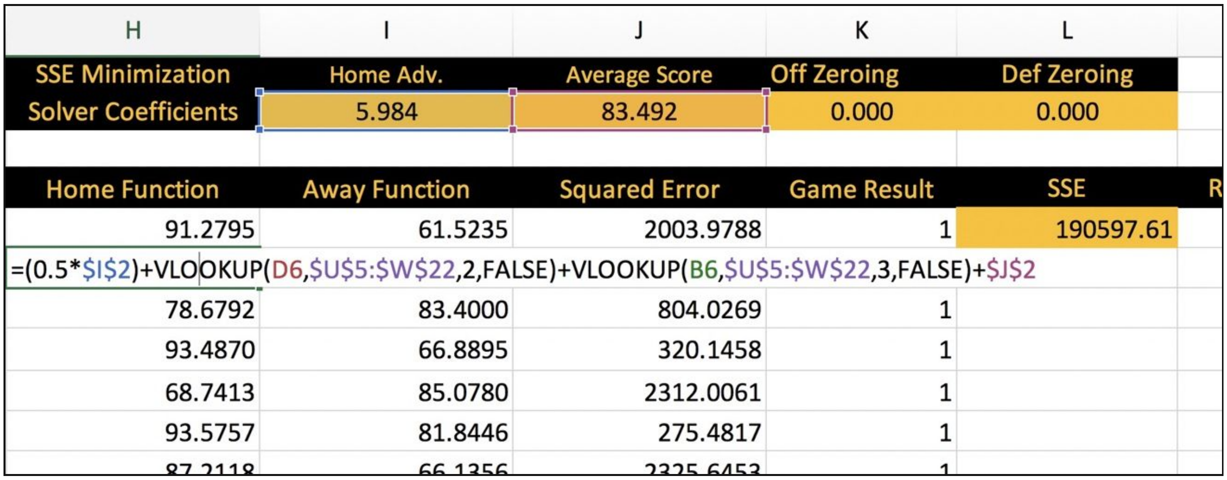
Getting Started

Let's go ahead and open the "PRP" Excel file. You'll immediately see that most of the sheet is the same or very similar to everything we've done up to this point. We have a new home and away function in columns H and I, and new model variables in cells I2, J2, K2 and L2. Let's go over them quickly.

Our basic model function takes half of the home team advantage variable and adds it to the home team rating, then takes the sum of the home team offense and away team defense ratings and finally adds in the average team score for the league. For the away team we do the same thing, but subtract half of the home team advantage variable. The structure of this, found in columns H and I, looks like this:

```
0.5*HFA+(HTOff+ATDef)+League Average Score
-0.5*HFA+(ATOff+HTDef)+League Average Score
```

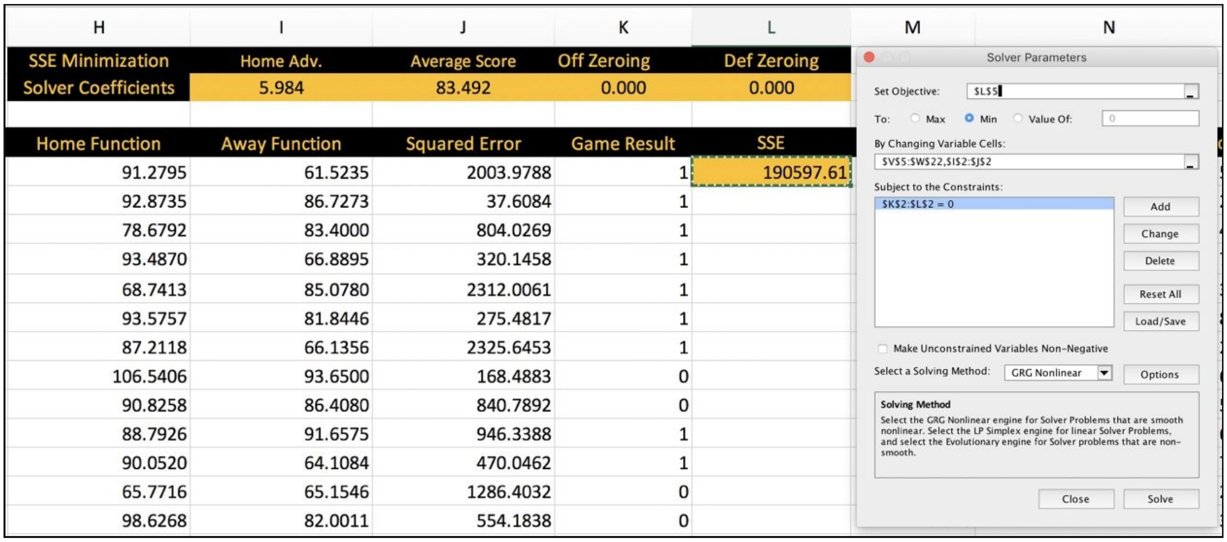
Cells K2 and L2 are the average of the team ratings for offense and defense and are simply there to ensure that the league-wide ratings average to zero when we run our solver optimization. Column J calculates the squared error for both our home and away point forecasts and cell L5 sums these errors in a target cell we will ask solver to minimize for us. As you can see, its very similar to models we've walked through before. It's just a slightly different way of doing things.



H	I	J	K	L	
SSE Minimization	Home Adv.	Average Score	Off Zeroing	Def Zeroing	
Solver Coefficients	5.984	83.492	0.000	0.000	
Home Function	Away Function	Squared Error	Game Result	SSE	R
91.2795	61.5235	2003.9788	1	190597.61	
=(0.5*$I$2)+VLOOKUP(D6,$U$5:$W$22,2,FALSE)+VLOOKUP(B6,$U$5:$W$22,3,FALSE)+$J$2					
78.6792	83.4000	804.0269	1		
93.4870	66.8895	320.1458	1		
68.7413	85.0780	2312.0061	1		
93.5757	81.8446	275.4817	1		
87.2118	66.1356	2325.6453	1		

Optimizing the Model

Click on solver to prepare the model for optimization. This time we're going to ask solver to minimize the SSE value in cell L5 by adjusting the team ratings in columns V and W, the home advantage variable in cell I2 and the average score in cell J2. I've also added an additional constraint to tell solver that we want the average offense and defense rating to be 0, as indicated in cells K2 and L2:



H	I	J	K	L
SSE Minimization	Home Adv.	Average Score	Off Zeroing	Def Zeroing
Solver Coefficients	5.984	83.492	0.000	0.000
Home Function	Away Function	Squared Error	Game Result	SSE
91.2795	61.5235	2003.9788	1	190597.61
92.8735	86.7273	37.6084	1	
78.6792	83.4000	804.0269	1	
93.4870	66.8895	320.1458	1	
68.7413	85.0780	2312.0061	1	
93.5757	81.8446	275.4817	1	
87.2118	66.1356	2325.6453	1	
106.5406	93.6500	168.4883	0	
90.8258	86.4080	840.7892	0	
88.7926	91.6575	946.3388	1	
90.0520	64.1084	470.0462	1	
65.7716	65.1546	1286.4032	0	
98.6268	82.0011	554.1838	0	

Solver Parameters

Set Objective: $L$5

To: ○ Max ● Min ○ Value Of: 0

By Changing Variable Cells: $V$5:$W$22,$I$2:$J$2

Subject to the Constraints: $K$2:$L$2 = 0

□ Make Unconstrained Variables Non-Negative

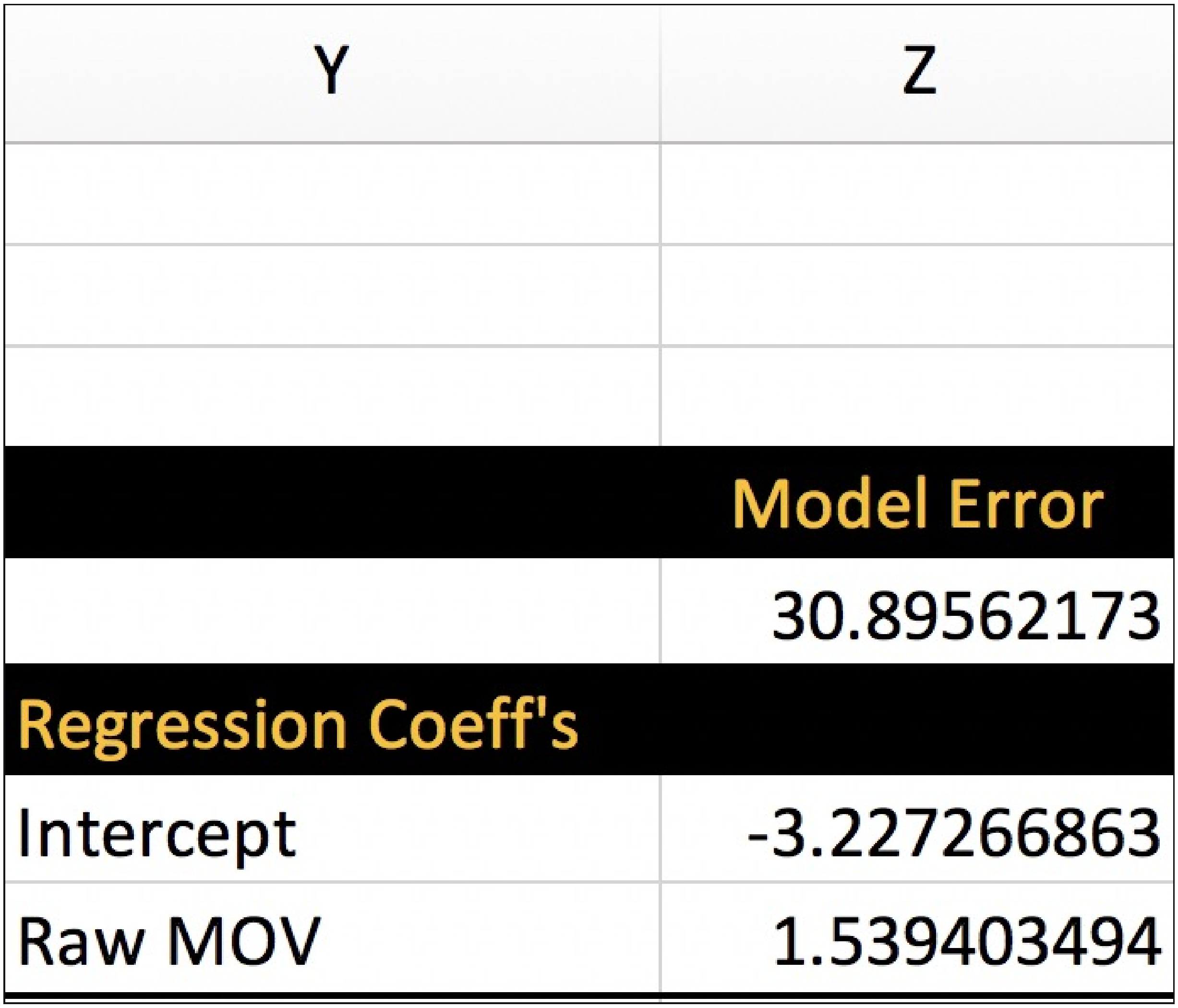
Select a Solving Method: GRG Nonlinear

Solving Method
Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Once the settings are correct, we click solver and let it converge on a solution.

Linear Regression

As with all model before now, we'll once again run a quick linear regression to map our expected MOV to the actual observed MOV. This time our Y variable will be column G and our X variable will be column M. Once we have the coefficients and the standard error, we can copy and paste them into the main PRP sheet in column Z.



Y	Z
	Model Error
	30.89562173
Regression Coeff's	
Intercept	-3.227266863
Raw MOV	1.539403494

From here on out, things function more or less as previously discussed in other models. You'll find the model metrics at the top of columns R through X, and the "Game Predict" function starting in column AB.

With that, we've built 5 models that all have a slightly different way of viewing the same game results dataset. It's time to move on to our next project: combining these models together into an ensemble.

[1] Mathletics, Wayne Winston.

ENSEMBLE MODELS

Ensemble models stem from the concept that several models combined together are often stronger than any single model by itself. However, this is only true if the collection of models is capable of telling you something that no individual model “knows” by itself. To help put that statement in context let’s talk for a minute about ensemble models, diversity of opinion and the wisdom of the crowd.

> “Why is it that one single model can almost never beat the market, but the wisdom of the crowd can prevail, despite the fact that there are almost certainly idiots among the crowd?”

Noted sports betting authors like Joseph Buchdahl[1] have asked this question and made very insightful headway towards answering it. If you’ve spent any time thinking about it, you may have wondered the same thing. How is it that the wisdom of the crowd can be so successful?

I think the short answer is diversity.

Diversity of opinion, to be precise.

The diversity of forecast opinion in the aggregate marketplace explains the phenomenon nicely. We might think of this as a Bayesian updating process whereby a mass of differing opinions shape and tune the market forecast via betting decisions until it becomes altogether very accurate, despite the fact that individual wagers may be…for lack of a better term…idiotic (in kinder data science terms we might call them “weak learners”). After all, the bettors firing on 140 unit max bomb whale plays are in the market somewhere.

Turning now to Excel models, we would be wise to ask ourselves if combining multiple models together couldn’t help our forecasts in a similar fashion. If we can harness a collection of models with differing opinions on a given prospective wager, we might be able to capitalize on the wisdom of the “statistical crowd”.

This general idea is widely used in data science modelling. However, there are a few things to consider before moving on. First, we must be cautious of multicollinearity. In individual models this induces forecast error from highly correlated input variables or what we might consider to be a lack of opinion diversity. Phrased another way, all the explanatory variables are “thinking the same”. It’s also why you’re more likely to miscalculate an instinct bet on your local team if everyone you know in your community has a similar opinion about who is likely to win. Since this atmosphere reinforces your opinion, you’re liable to miss contemporaneous evidence to the contrary. There is a groupthink heuristic effect taking place. This can be as true of collections of models as it is collections of people. For this reason we need to find ways to deal with model forecasts that are highly correlated with one another. Secondly, the more diverse the individual model forecasts can be while still being a decent forecast in their own right, the better. Combining many diverse models seeing the same game from different perspectives is the name of the game.

To that end, what we’ll be doing is taking each of our models and using their individual forecast outputs as inputs in a larger amalgamated model: an ensemble. We’ll explore some different ways to combine the models together, consider ways to reduce multicollinearity and hopefully arrive at a better game forecast. We have logistic team strength, margin of victory, team points and game scores all working together to enhance the combined forecast accuracy beyond any one model left to its own devices.

It’s the wisdom of the statistical crowd.

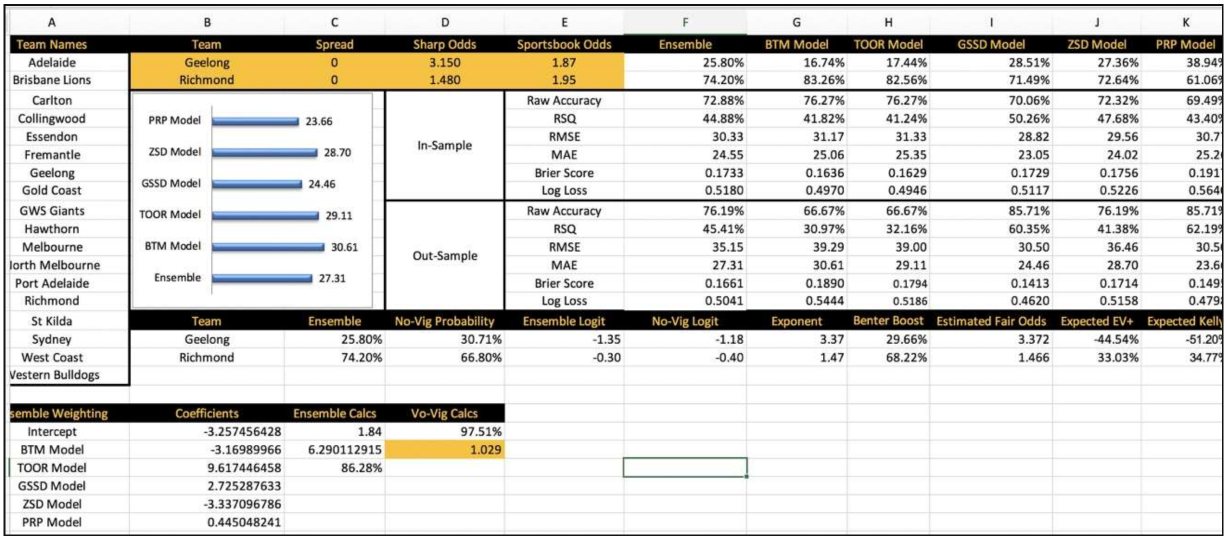
[1] https://www.football-data.co.uk/wisdom_of_crowd_bets

12

So far, we've worked our way through each of the five basic latent rating models in the book. Each of them works decently, and depending on the sport and market they may fare alright on their own. Ensemble models are where we turn things up a notch and attempt to derive a better forecast by combining our first five models in different ways.

Getting Started

Let's open up the "Tutorial" Excel file and view the ensemble model dashboard. This is the main spreadsheet that I use to reference each of the individual model sheets in one location so that we can combine them and assess all of the forecasts at once. You might think of it as a very unsophisticated program. It provides a way to input the two teams playing in a given game and receive forecasts from all of the models simultaneously, which is quite handy. Take a look:

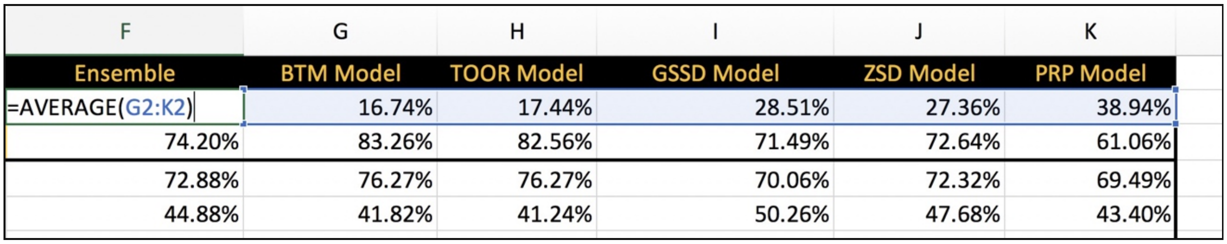


A	B	C	D	E	F	G	H	I	J	K
Team Names	Team	Spread	Sharp Odds	Sportsbook Odds	Ensemble	BTM Model	TOOR Model	GSSD Model	ZSD Model	PRP Model
Adelaide	Geelong	0	3.150	1.87	25.80%	16.74%	17.44%	28.51%	27.36%	38.94
Brisbane Lions	Richmond	0	1.480	1.95	74.20%	83.26%	82.56%	71.49%	72.64%	61.06
Carlton			In-Sample	Raw Accuracy	72.88%	76.27%	76.27%	70.06%	72.32%	69.49
Collingwood				RSQ	44.88%	41.82%	41.24%	50.26%	47.68%	43.40
Essendon				RMSE	30.33	31.17	31.33	28.82	29.56	30.7
Fremantle				MAE	24.55	25.06	25.35	23.05	24.02	25.2
Geelong				Brier Score	0.1733	0.1636	0.1629	0.1729	0.1756	0.191
Gold Coast				Log Loss	0.5180	0.4970	0.4946	0.5117	0.5226	0.564
GWS Giants			Out-Sample	Raw Accuracy	76.19%	66.67%	66.67%	85.71%	76.19%	85.71
Hawthorn				RSQ	45.41%	30.97%	32.16%	60.35%	41.38%	62.19
Melbourne				RMSE	35.15	39.29	39.00	30.50	36.46	30.5
lorth Melbourne				MAE	27.31	30.61	29.11	24.46	28.70	23.6
Port Adelaide				Brier Score	0.1661	0.1890	0.1794	0.1413	0.1714	0.149
Richmond				Log Loss	0.5041	0.5444	0.5186	0.4620	0.5158	0.479
St Kilda	Team	Ensemble	No-Vig Probability	Ensemble Logit	No-Vig Logit	Exponent	Benter Boost	Estimated Fair Odds	Expected EV+	Expected Kell
Sydney	Geelong	25.80%	30.71%	-1.35	-1.18	3.37	29.66%	3.372	-44.54%	-51.20
West Coast	Richmond	74.20%	66.80%	-0.30	-0.40	1.47	68.22%	1.466	33.03%	34.77
Vestern Bulldogs										
semble Weighting	Coefficients	Ensemble Calcs	Vo-Vig Calcs							
Intercept	-3.257456428	1.84	97.51%							
BTM Model	-3.16989966	6.290112915	1.029							
TOOR Model	9.617446458	86.28%								
GSSD Model	2.725287633									
ZSD Model	-3.337096786									
PRP Model	0.445048241									

From this main dashboard, I can type in the teams that are playing in B2 and B3 along with a few other inputs and receive forecasts from every model in one place. I can also view in-sample and out-sample model metrics to see how each model is doing in a comparative context. Note the graph showing model out-sample MAE error metrics. Finally, the dashboard gives us a place to combine the individual models into an ensemble that can hopefully forecast even better than any one model alone. Let's get started with a very simple ensemble technique and we'll progress from there.

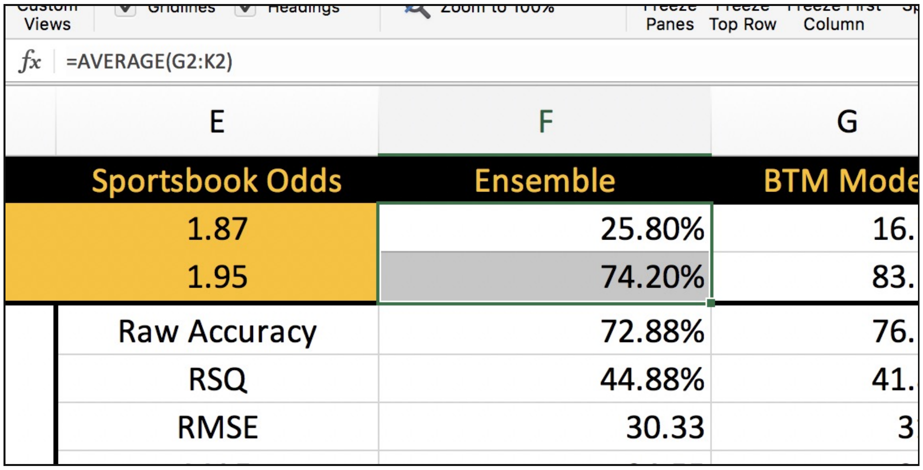
Simple Average Ensemble

The simplest way to combine multiple forecasts together is to average them together. We can achieve this easily in the dashboard by averaging row 2 from column G to K and then doing the same for row 3:



F	G	H	I	J	K
Ensemble	BTM Model	TOOR Model	GSSD Model	ZSD Model	PRP Model
=AVERAGE(G2:K2)	16.74%	17.44%	28.51%	27.36%	38.94%
74.20%	83.26%	82.56%	71.49%	72.64%	61.06%
72.88%	76.27%	76.27%	70.06%	72.32%	69.49%
44.88%	41.82%	41.24%	50.26%	47.68%	43.40%

When we have completed this for both rows, the averaged ensemble probabilities are found in cells F2 and F3.



fx =AVERAGE(G2:K2)

E	F	G
Sportsbook Odds	Ensemble	BTM Mode
1.87	25.80%	16.
1.95	74.20%	83.
Raw Accuracy	72.88%	76.
RSQ	44.88%	41.
RMSE	30.33	3

We can then use these to calculate fair odds in I17 and I18, and use these to estimate the value for a given wager if we wanted to.

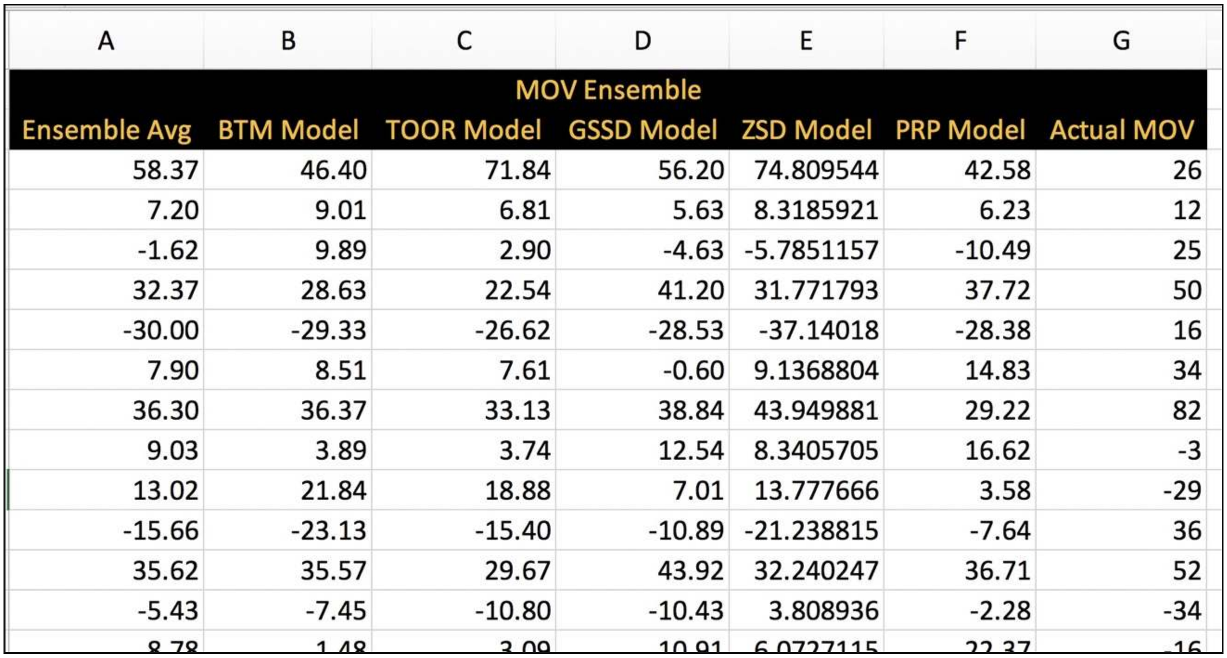
However, we should not be in a rush to do so. There are some important questions we may want to ask ourselves first:

> Should all models be weighted equally?
>
> Are all models necessary and relevant?
>
> Do we have multicollinearity issues?

Surely, there must be a better way to weight these models together. As it turns out, there are a few ways for us to accomplish this in a more calculated manner. Two of the more straightforward methods are to weight the individual model inputs with linear regression or with a logistic regression. Let’s explore each of those in turn. While these are definitely not the only ways to do this, they are both reasonably effective methods and easily implemented inside our Excel spreadsheet.

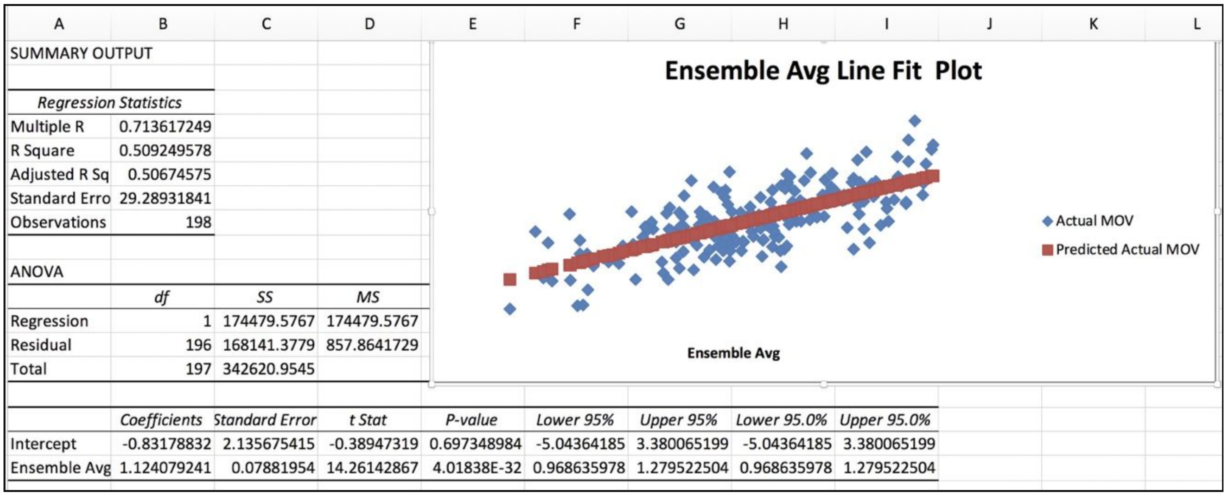
Linear Regression Ensemble Weighting

In order to properly weight our collection of models with a linear regression, I’ve created a sheet in the “TUTORIAL” Excel file called “Ensemble Engine”. This is a sheet that shows all the various model output forecasts for both MOV and home win probability along with both the actual game MOV and a binary [1,0] outcome result for the home team. This is where we will apply different weighting schemes to our models to see if we can arrive at a configuration that is better than the simple averaging method. Take a look at the sheet and notice that our individual model outputs are now being used as inputs - with the outcome variable being the result we are trying to map:



A	B	C	D	E	F	G
MOV Ensemble						
Ensemble Avg	BTM Model	TOOR Model	GSSD Model	ZSD Model	PRP Model	Actual MOV
58.37	46.40	71.84	56.20	74.809544	42.58	26
7.20	9.01	6.81	5.63	8.3185921	6.23	12
-1.62	9.89	2.90	-4.63	-5.7851157	-10.49	25
32.37	28.63	22.54	41.20	31.771793	37.72	50
-30.00	-29.33	-26.62	-28.53	-37.14018	-28.38	16
7.90	8.51	7.61	-0.60	9.1368804	14.83	34
36.30	36.37	33.13	38.84	43.949881	29.22	82
9.03	3.89	3.74	12.54	8.3405705	16.62	-3
13.02	21.84	18.88	7.01	13.777666	3.58	-29
-15.66	-23.13	-15.40	-10.89	-21.238815	-7.64	36
35.62	35.57	29.67	43.92	32.240247	36.71	52
-5.43	-7.45	-10.80	-10.43	3.808936	-2.28	-34
8.78	1.48	3.09	10.91	6.0727115	22.37	-16

We're now set up to test some different ideas using this dataset. Let's first try using a linear regression where the Y variable is column G and the X variables is column A, the simple model average. This will give us a preliminary idea about how well the simple average model might perform. After running the regression in a new sheet, we get the following result:



SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.713617249
R Square	0.509249578
Adjusted R Sq	0.50674575
Standard Erro	29.28931841
Observations	198

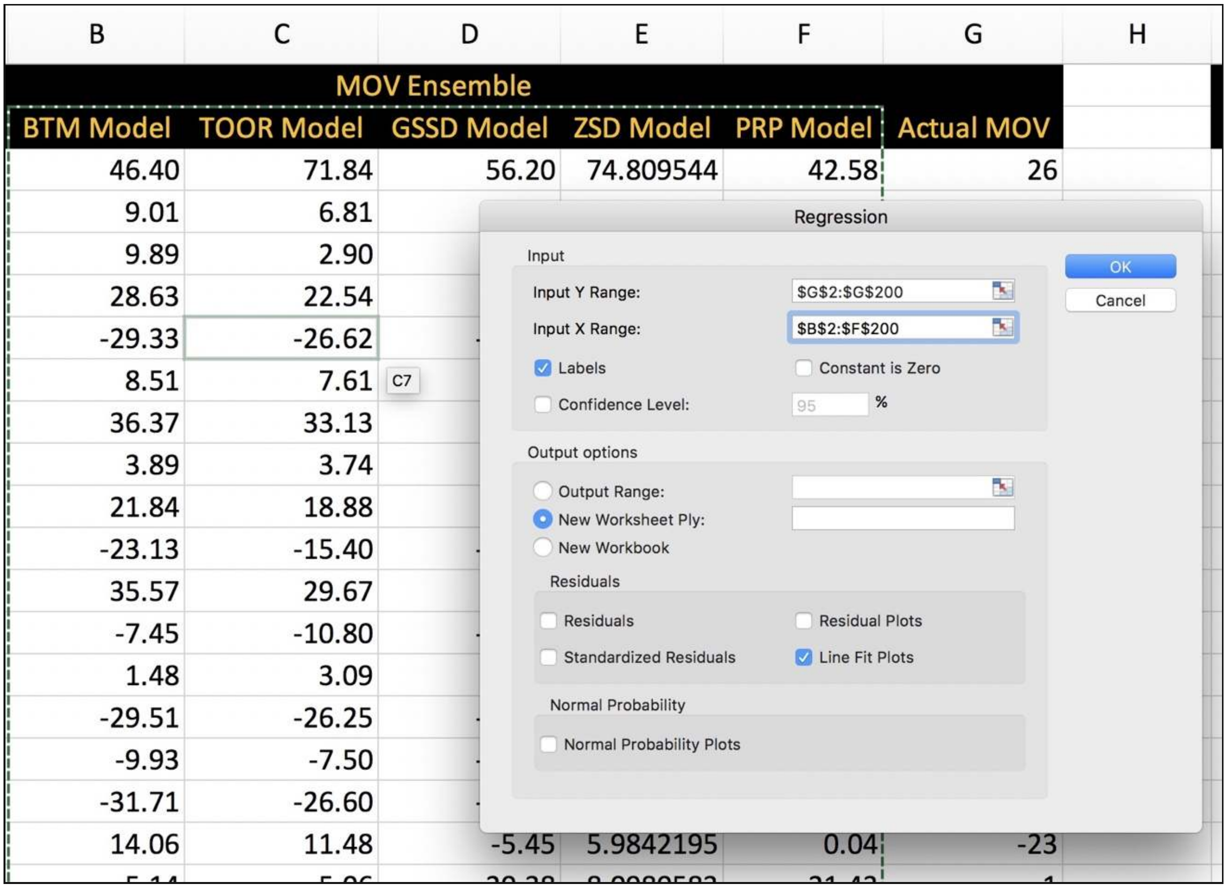
ANOVA

	df	*SS*	*MS*
Regression	1	174479.5767	174479.5767
Residual	196	168141.3779	857.8641729
Total	197	342620.9545	

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-0.83178832	2.135675415	-0.38947319	0.697348984	-5.04364185	3.380065199	-5.04364185	3.380065199
Ensemble Avg	1.124079241	0.07881954	14.26142867	4.01838E-32	0.968635978	1.279522504	0.968635978	1.279522504

With a decent RSQ value and a standard error of 29.28, this simple average model looks alright. We can also see a high t stat value as well as a low p-value, indicating a high likelihood of statistical significance. We could, if we wanted, take these regression coefficients and the standard error and use them similarly to the what we've done in the past - taking the estimated average MOV from the implemented regression, inputting it into our NORMDIST command and deriving an expected win percentage. This would be another way to achieve a simple average ensemble instead of averaging the probability outputs in the dashboard sheet.

While the simple average works alright, it contains a number of issues that may not be useful for our purposes. It might be unreasonable to move forward on the assumption that all of the input models have equal value and should be weighted equally. Perhaps some of our models aren't adding much to the mean expectation of the ensemble. More importantly, we might be inadvertently introducing multicollinearity issues. As a first guess, we know that the BTM and TOOR models are very similar to each other - so similar in fact that we might be better off using only one of them in any given ensemble. It's in our best interest to investigate this further. Let's try a linear regression using all individual model outputs as X variables and the actual game MOV as the Y variable. This should give us an indication of the optimal input model weights.



B	C	D	E	F	G	H
MOV Ensemble						
BTM Model	TOOR Model	GSSD Model	ZSD Model	PRP Model	Actual MOV	
46.40	71.84	56.20	74.809544	42.58	26	
9.01	6.81					
9.89	2.90					
28.63	22.54					
-29.33	-26.62					
8.51	7.61					
36.37	33.13					
3.89	3.74					
21.84	18.88					
-23.13	-15.40					
35.57	29.67					
-7.45	-10.80					
1.48	3.09					
-29.51	-26.25					
-9.93	-7.50					
-31.71	-26.60					
14.06	11.48	-5.45	5.9842195	0.04	-23	

Regression

Input

Input Y Range: $G$2:$G$200

Input X Range: $B$2:$F$200

☑ Labels □ Constant is Zero

□ Confidence Level: 95 %

Output options

○ Output Range:

● New Worksheet Ply:

○ New Workbook

Residuals

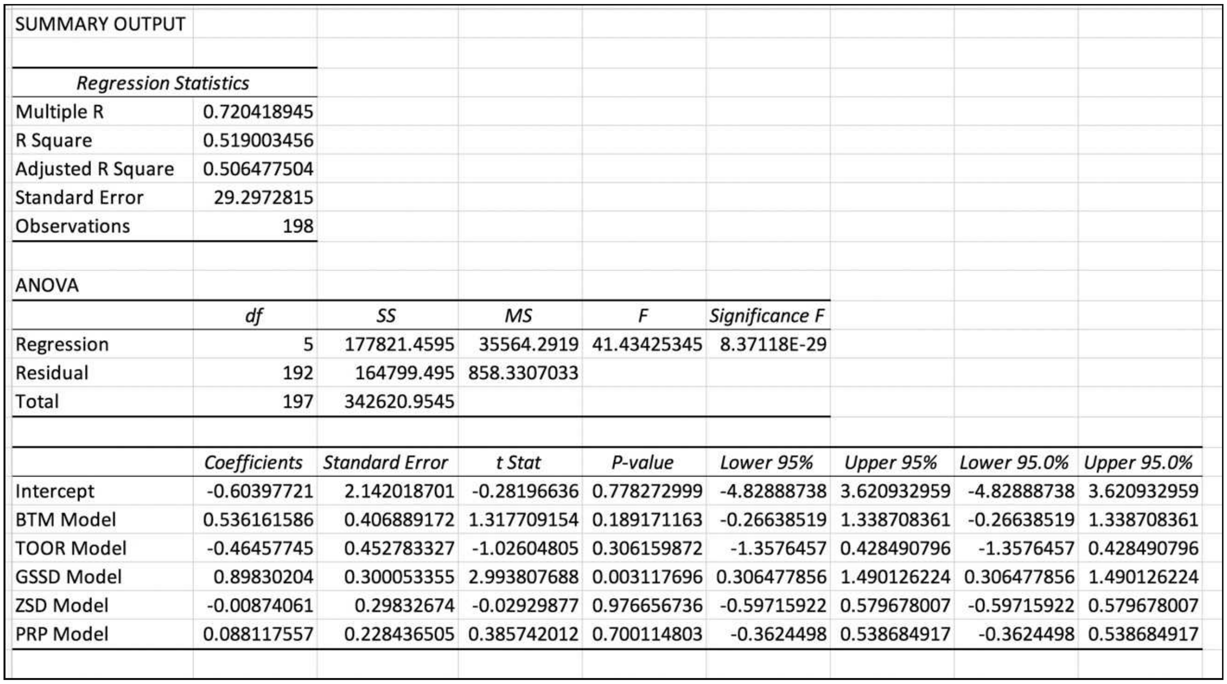
□ Residuals □ Residual Plots

□ Standardized Residuals ☑ Line Fit Plots

Normal Probability

□ Normal Probability Plots

With our regression ready to go, let’s click OK and let Excel work out the numbers. We’ll want to pay attention to a number of metrics on the regression output sheet including adjusted RSQ, standard error, coefficient weights, p-values and t stat values.



SUMMARY OUTPUT

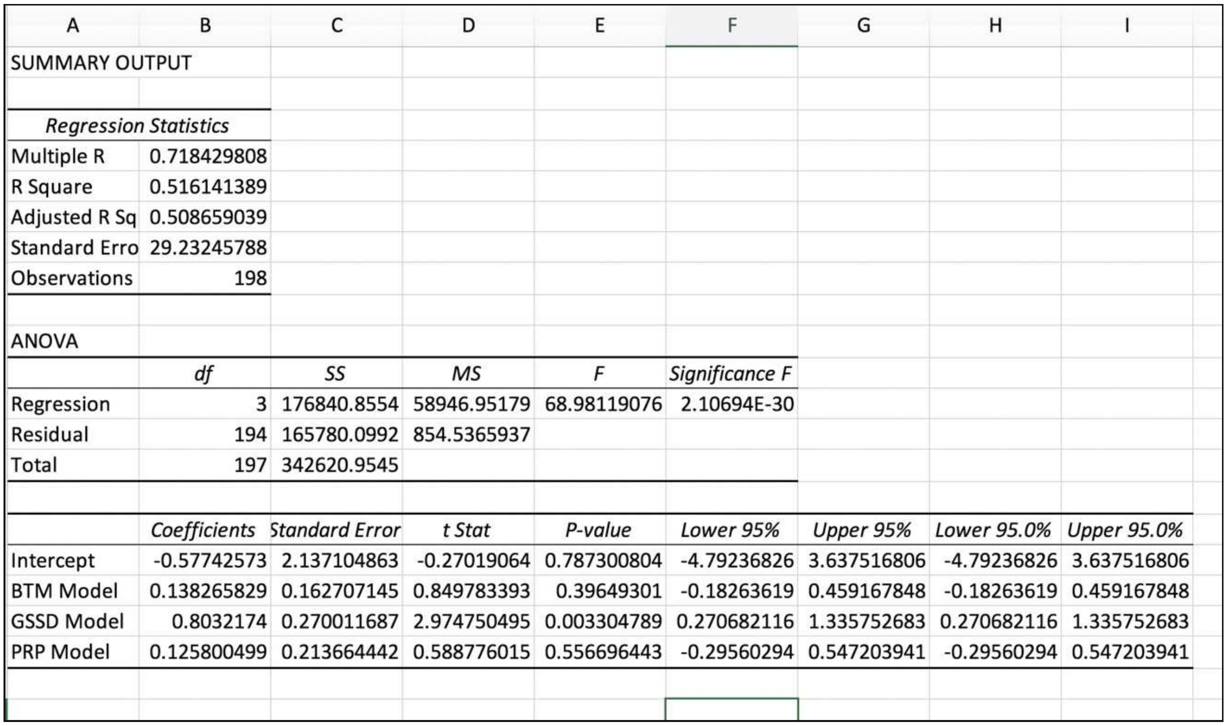
Regression Statistics	
Multiple R	0.720418945
R Square	0.519003456
Adjusted R Square	0.506477504
Standard Error	29.2972815
Observations	198

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	5	177821.4595	35564.2919	41.43425345	8.37118E-29
Residual	192	164799.495	858.3307033		
Total	197	342620.9545			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	-0.60397721	2.142018701	-0.28196636	0.778272999	-4.82888738	3.620932959	-4.82888738	3.620932959
BTM Model	0.536161586	0.406889172	1.317709154	0.189171163	-0.26638519	1.338708361	-0.26638519	1.338708361
TOOR Model	-0.46457745	0.452783327	-1.02604805	0.306159872	-1.3576457	0.428490796	-1.3576457	0.428490796
GSSD Model	0.89830204	0.300053355	2.993807688	0.003117696	0.306477856	1.490126224	0.306477856	1.490126224
ZSD Model	-0.00874061	0.29832674	-0.02929877	0.976656736	-0.59715922	0.579678007	-0.59715922	0.579678007
PRP Model	0.088117557	0.228436505	0.385742012	0.700114803	-0.3624498	0.538684917	-0.3624498	0.538684917

A few things stand out right away. First we must look at adjusted RSQ instead of standard RSQ when considering a model with multiple input predictors. It looks like we have an improvement in adjusted RSQ and standard error in-sample over any of our individual models. This is a promising sign. However, the t stat values suggest that the TOOR model and ZSD model aren't adding a lot to our ensemble. They also have p-values that fail to reject the null hypothesis, meaning among other things that we can't rule out the possibility these model inputs add nothing but noise to our ensemble. Let's try removing the two models in question and running our regression again with only model inputs that have positive t stat values to see what changes we can observe. That means we'll be using BTM, GSSD and PRP model inputs for this regression. Once Excel is finished we can observe the new regression results:



SUMMARY OUTPUT

Regression Statistics	
Multiple R	0.718429808
R Square	0.516141389
Adjusted R Sq	0.508659039
Standard Erro	29.23245788
Observations	198

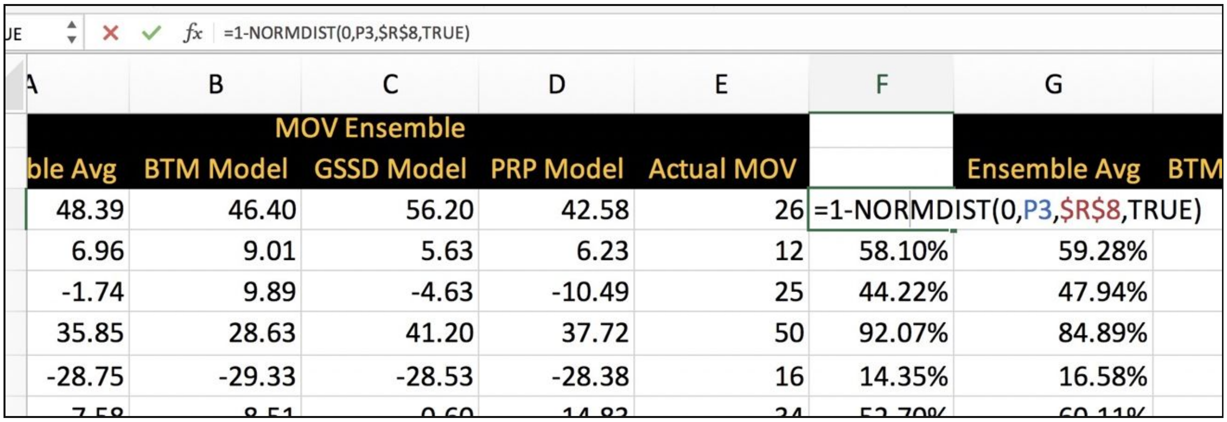
ANOVA

	df	SS	MS	F	Significance F
Regression	3	176840.8554	58946.95179	68.98119076	2.10694E-30
Residual	194	165780.0992	854.5365937		
Total	197	342620.9545			

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-0.57742573	2.137104863	-0.27019064	0.787300804	-4.79236826	3.637516806	-4.79236826	3.637516806
BTM Model	0.138265829	0.162707145	0.849783393	0.39649301	-0.18263619	0.459167848	-0.18263619	0.459167848
GSSD Model	0.8032174	0.270011687	2.974750495	0.003304789	0.270682116	1.335752683	0.270682116	1.335752683
PRP Model	0.125800499	0.213664442	0.588776015	0.556696443	-0.29560294	0.547203941	-0.29560294	0.547203941

Again we see some slight improvements in adjusted RSQ and standard error. Model t stat values are all positive. While the p-values for the GSSD model look alright, there is some concern with the higher p-values for the BTM and PRP models. We would probably want to try some other things to try and increase the likelihood of statistical significance (thereby reducing the p-values to 0.05 or less). Some ideas we might try are a ridge regression or a principal component analysis, both of which can be accomplished using the Real Statistics add-on in Excel. However, while acknowledging this as a potential issue, I think it would be best to move on with this example for the sake of simplicity. It's always a good idea to come back and run more tests in the future.

To finish our example we can simply copy and paste the regression coefficients and standard error into our Ensemble Engine sheet and use this along with the relevant model forecasts to produce a weighted MOV expectation. From there, we can apply our NORMDIST command in excel and derive the relevant win or point spread probabilities.



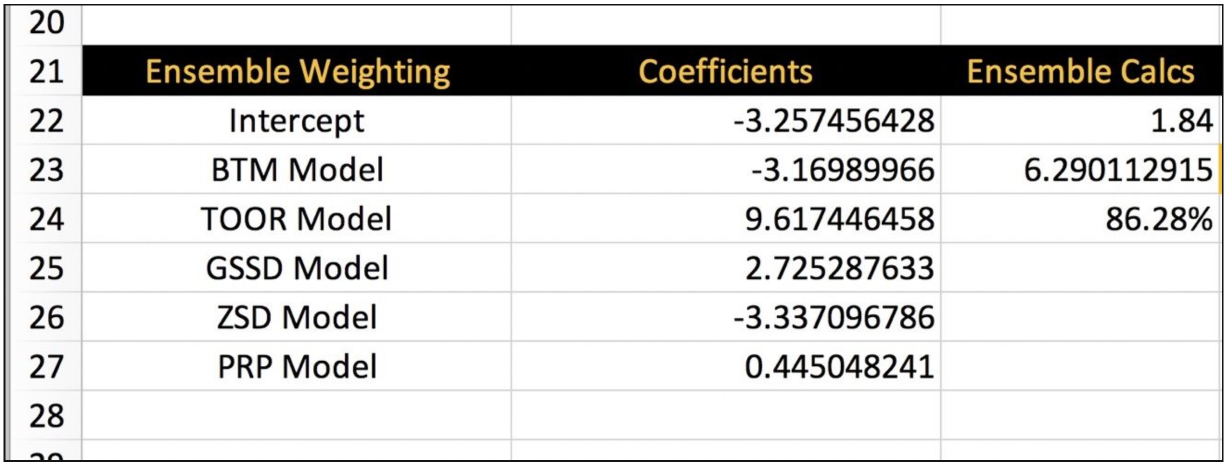
=1-NORMDIST(0,P3,$R$8,TRUE)

A	B	C	D	E	F	G	
	MOV Ensemble						
ble Avg	BTM Model	GSSD Model	PRP Model	Actual MOV		Ensemble Avg	BTM
48.39	46.40	56.20	42.58	26	=1-NORMDIST(0,P3,$R$8,TRUE)		
6.96	9.01	5.63	6.23	12	58.10%	59.28%	
-1.74	9.89	-4.63	-10.49	25	44.22%	47.94%	
35.85	28.63	41.20	37.72	50	92.07%	84.89%	
-28.75	-29.33	-28.53	-28.38	16	14.35%	16.58%	
7.58	8.51	0.60	14.82	24	52.70%	60.11%	

If we wanted to use this in the dashboard, we could simply substitute each model's probability outputs for its MOV outputs, weight them properly with the regression coefficients and then apply the NORMDIST command to produce a win probability for the game we want to look at. However, I think we can do a touch better than this. Let's take a look at what I think is the preferred method for win probability ensembles in Excel - a logistic regression.

Logistic Regression Ensemble Weighting

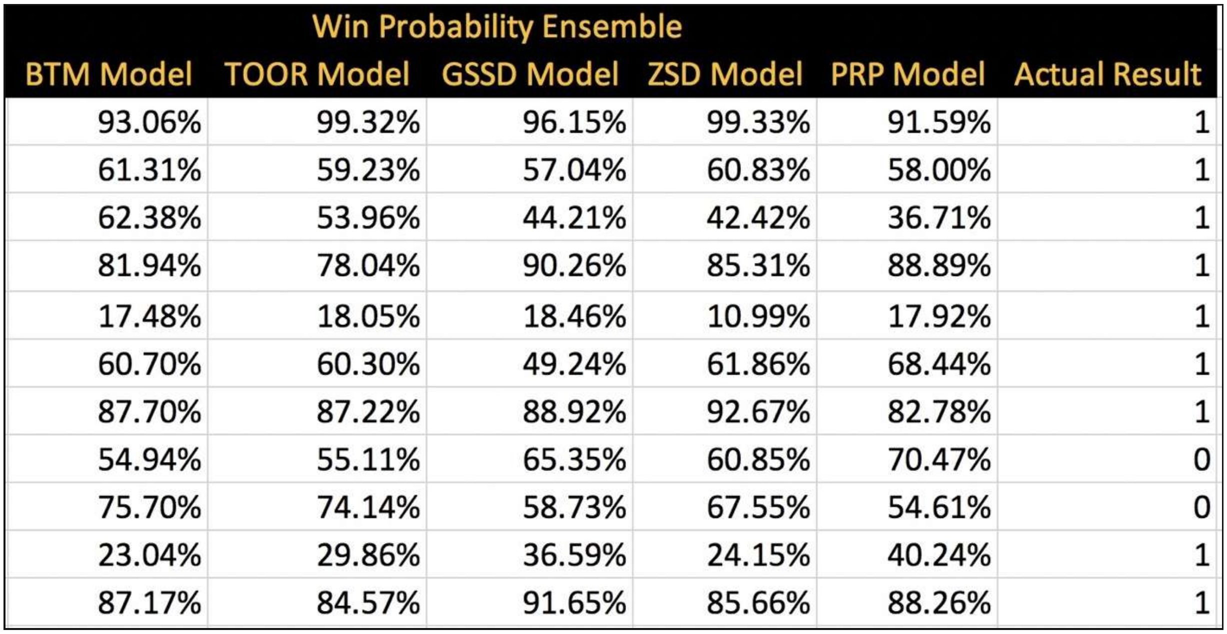
For win probability ensembles in Excel, I think this technique you'll want to look at seriously. Binary logistic regression is capable of classifying teams into winners and losers while simultaneously outputting probabilities - perfect for our purposes. We want to know if team A is going to win, and most importantly what their probability of winning is so that we can derive a price and find value in the market. Logistic regression accomplishes this easily. You can see in the dashboard sheet that I already have coefficients loaded for this technique starting in cell A21.



20			
21	Ensemble Weighting	Coefficients	Ensemble Calcs
22	Intercept	-3.257456428	1.84
23	BTM Model	-3.16989966	6.290112915
24	TOOR Model	9.617446458	86.28%
25	GSSD Model	2.725287633	
26	ZSD Model	-3.337096786	
27	PRP Model	0.445048241	
28			

Getting Started

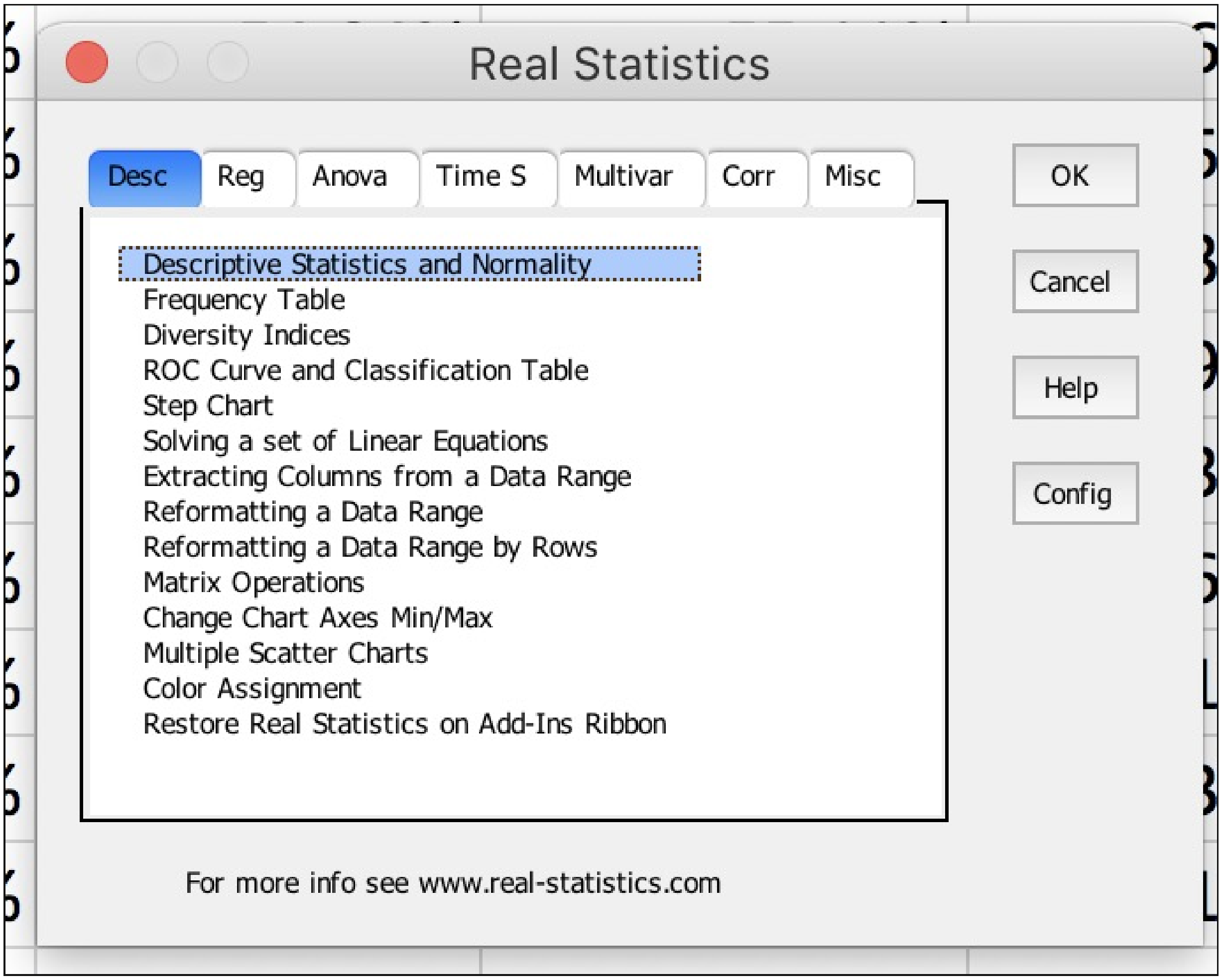
To get started, let's head over to the "Ensemble Engine" sheet. You'll see starting in column I that we have the probability outputs of each individual model along with the game result as a 1 or 0 in column O. 1 represents a win for the home team while 0 represents a loss. These are the two "classes" or categories that we will be asking the logistic regression to sort our home team dataset into.



		Win Probability Ensemble			
BTM Model	TOOR Model	GSSD Model	ZSD Model	PRP Model	Actual Result
93.06%	99.32%	96.15%	99.33%	91.59%	1
61.31%	59.23%	57.04%	60.83%	58.00%	1
62.38%	53.96%	44.21%	42.42%	36.71%	1
81.94%	78.04%	90.26%	85.31%	88.89%	1
17.48%	18.05%	18.46%	10.99%	17.92%	1
60.70%	60.30%	49.24%	61.86%	68.44%	1
87.70%	87.22%	88.92%	92.67%	82.78%	1
54.94%	55.11%	65.35%	60.85%	70.47%	0
75.70%	74.14%	58.73%	67.55%	54.61%	0
23.04%	29.86%	36.59%	24.15%	40.24%	1
87.17%	84.57%	91.65%	85.66%	88.26%	1

What we're asking Excel to do for us here is to take the 5 individual model probability forecasts for each game and weight them optimally so that the logistic regression can accurately classify the home team as either a winner or a loser. We'll also ask that when the regression does that, it outputs a probability between 0-100%. Any team with an estimated win probability of more than 50% will be classified by the logistic regression as a winner and any team with less than 50% win probability will be classified as a loser. We can then use this to gauge the utility of our model in addition to deriving fair odds prices. Let's fire it up.

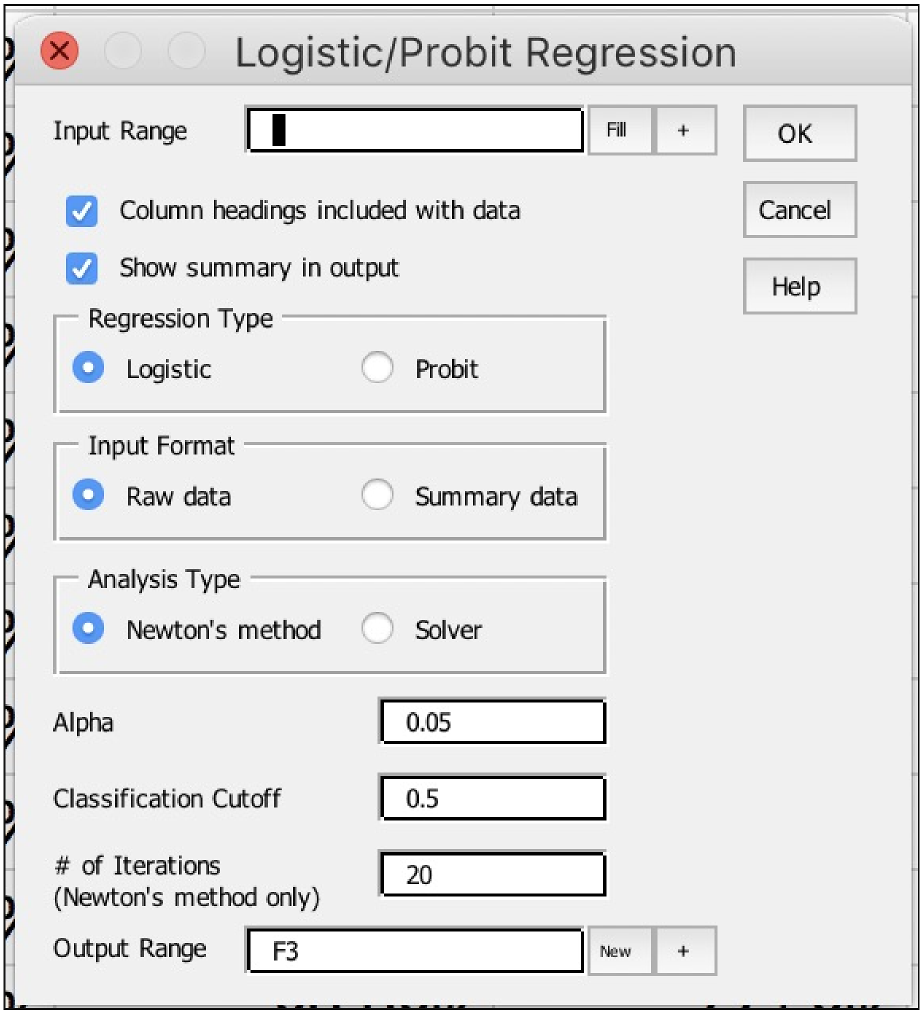
Click on "Add-ins" at the top of Excel and select "Real Statistics". If you are a Mac user, press Control+M[1].


Real Statistics
Desc
Reg
Anova
Time S
Multivar
Corr
Misc
Descriptive Statistics and Normality
Frequency Table
Diversity Indices
ROC Curve and Classification Table
Step Chart
Solving a set of Linear Equations
Extracting Columns from a Data Range
Reformatting a Data Range
Reformatting a Data Range by Rows
Matrix Operations
Change Chart Axes Min/Max
Multiple Scatter Charts
Color Assignment
Restore Real Statistics on Add-Ins Ribbon
OK
Cancel
Help
Config

For more info see www.real-statistics.com

Select the "Reg" tab, and then select "Binary Logistic and Probit Regression" and click OK. You should now see the Logistic

Regression interface on your screen:



Input Range

☑ Column headings included with data

☑ Show summary in output

Regression Type: ● Logistic ○ Probit

Input Format: ● Raw data ○ Summary data

Analysis Type: ● Newton's method ○ Solver

Alpha: 0.05

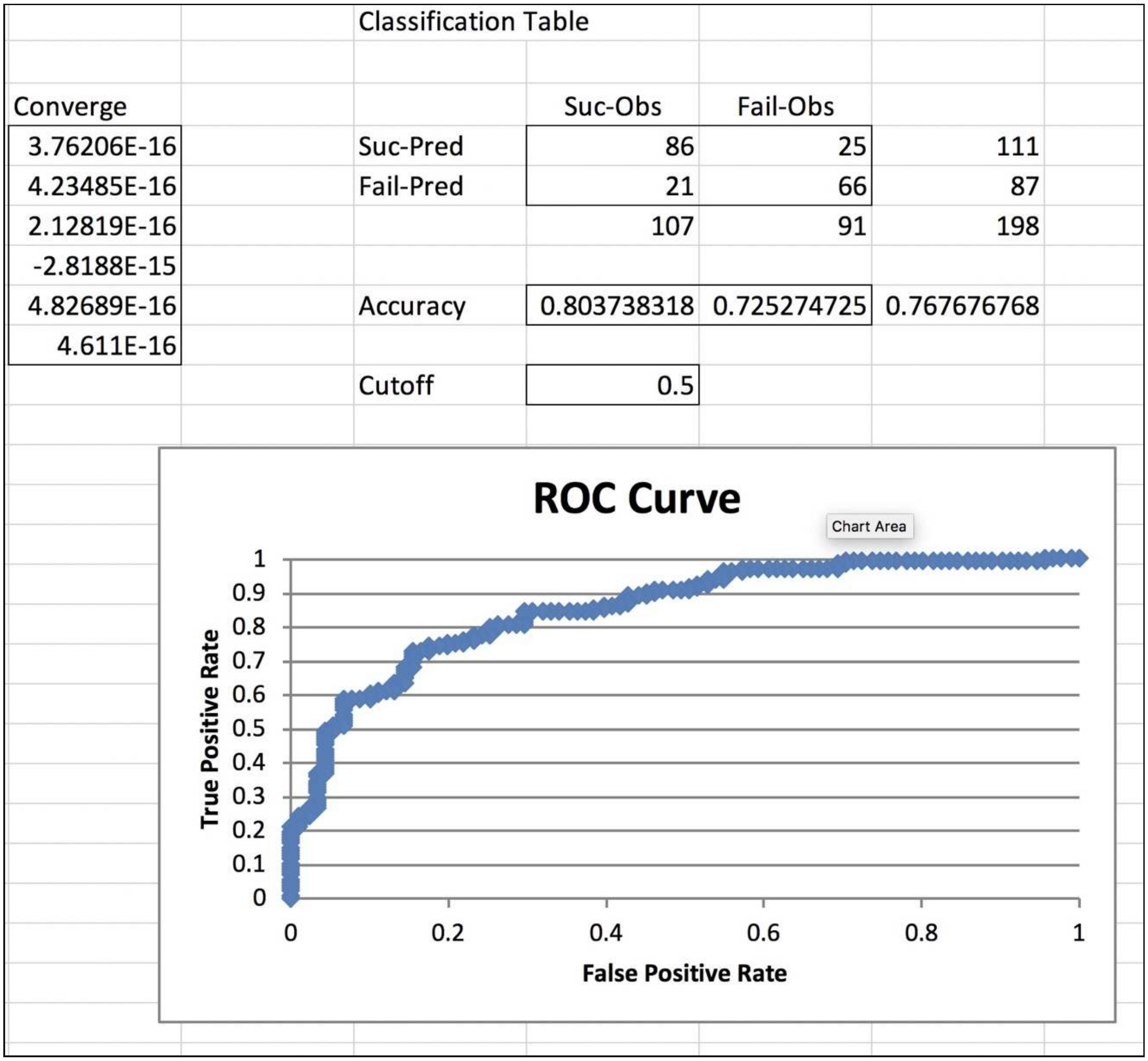
Classification Cutoff: 0.5

# of Iterations (Newton's method only): 20

Output Range: F3

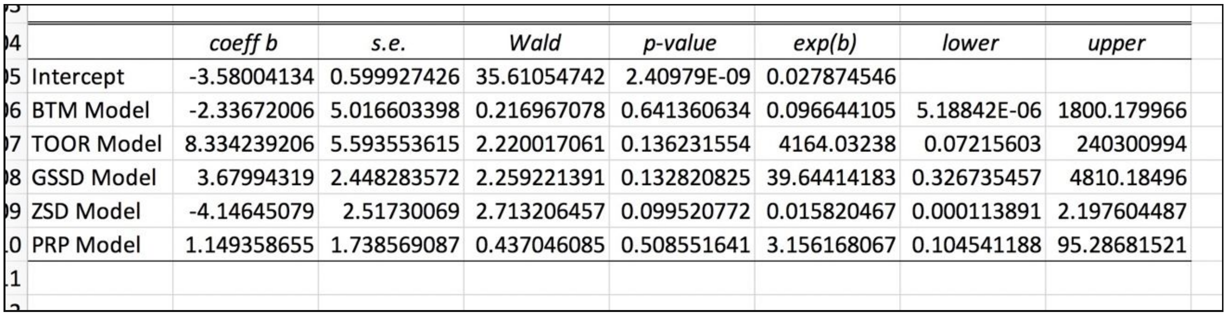
The input range includes all of our model probability outputs as well as the result column. Make sure the regression type is “Logistic”, input format is “Raw data” and the classification cutoff is 0.5 (aka 50%, as previously mentioned). The rest of the settings should be in their default modes as show above. I usually click “new” for the output range so that it creates a separate sheet for our regression results. Once you’re ready, click OK.

You should now be able to see a ROC curve graph as well as a classification table starting at column AE. This classification table shows us how well the logistic regression performed in sorting winners and losers into the right categories. We can see the accuracy for classifying a win as well as classifying a loss:



		Classification Table			
Converge			Suc-Obs	Fail-Obs	
3.76206E-16		Suc-Pred	86	25	111
4.23485E-16		Fail-Pred	21	66	87
2.12819E-16			107	91	198
-2.8188E-15					
4.82689E-16		Accuracy	0.803738318	0.725274725	0.767676768
4.611E-16					
		Cutoff	0.5		

We have ~80% accuracy for winners and ~72% accuracy for losers. That's fairly balanced across both classes, which is something we ideally want to see. We can also see a total accuracy of 76.76% in-sample, which is certainly promising (never get too excited before looking at out of sample performance though!). Let's take a look at the regression coefficients (model input weights) which can be found starting in cell A205:



		coeff b	s.e.	Wald	p-value	exp(b)	lower	upper
4								
5	Intercept	-3.58004134	0.599927426	35.61054742	2.40979E-09	0.027874546		
6	BTM Model	-2.33672006	5.016603398	0.216967078	0.641360634	0.096644105	5.18842E-06	1800.179966
7	TOOR Model	8.334239206	5.593553615	2.220017061	0.136231554	4164.03238	0.07215603	240300994
8	GSSD Model	3.67994319	2.448283572	2.259221391	0.132820825	39.64414183	0.326735457	4810.18496
9	ZSD Model	-4.14645079	2.51730069	2.713206457	0.099520772	0.015820467	0.000113891	2.197604487
0	PRP Model	1.149358655	1.738569087	0.437046085	0.508551641	3.156168067	0.104541188	95.28681521
1								

The coefficient weights and p-values here tell a different story from the one we observed with our linear regression. It looks like we might be better off dropping BTM and the PRP model from this ensemble.

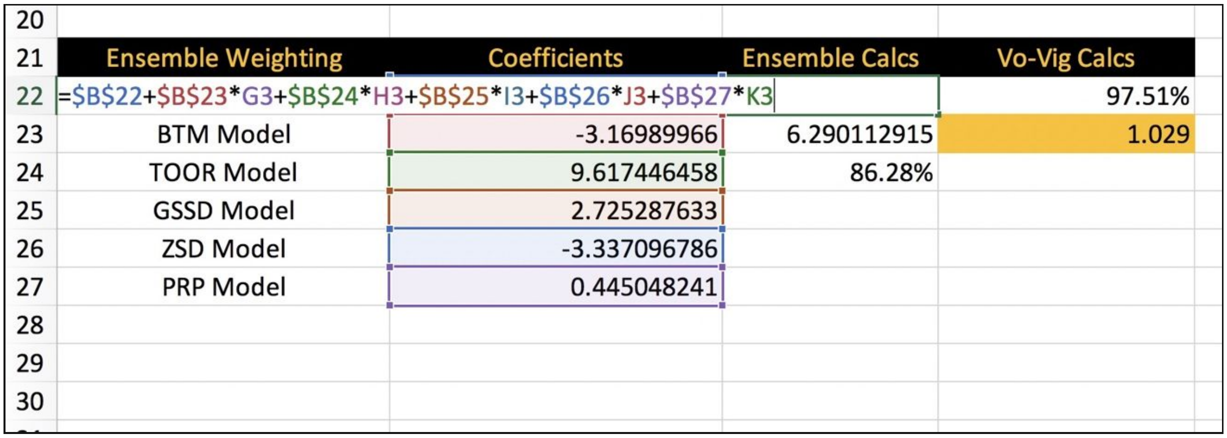
This is the kind of situation where we'll want to test many different possibilities to try and find the optimal solution. Ultimately, we must look for validation in our backtesting. We'll get to that soon, but for now let's carry on with this example to complete it. It's my hope that by progressing through the model steps in sequence without too many detours you'll get a better feel for the process. When you feel comfortable with it, then I would strongly encourage you to investigate ways to increase the statistical rigour of the model inputs. Focusing too much on the nuances can be a bit confusing at times in the beginning though, so I think it's best to proceed in this manner. It's the way I would have wanted to learn it - broad strokes first, supplemented with finer technical details as your comfort level grows.

Let's get back to the subject of the model input coefficients in the meantime. Copy and paste them into our dashboard sheet. You'll see that I've already completed this. You can find these values starting in A22 of the dashboard. We can now use these to derive team win probabilities.

Logistic regression coefficients can't be used quite the same way that linear regression coefficients are, because the output of a logistic regression is what's known as log odds. For that reason we need to run a few additional calculations to convert them to expected win probabilities. The first step is to apply the intercept and coefficients to each model's probability output for the home team. This part is performed the same way as with a linear regression:

(Intercept + Model 1 * Coefficient + Model 2* Coefficient…)

The output of this regression as previously mentioned is known as the log odds. You can find this calculation in cell C22. It looks like this:



20				
21	Ensemble Weighting	Coefficients	Ensemble Calcs	Vo-Vig Calcs
22	=$B$22+$B$23*G3+$B$24*H3+$B$25*I3+$B$26*J3+$B$27*K3			97.51%
23	BTM Model	-3.16989966	6.290112915	1.029
24	TOOR Model	9.617446458	86.28%	
25	GSSD Model	2.725287633		
26	ZSD Model	-3.337096786		
27	PRP Model	0.445048241		
28				
29				
30				

Next, in cell C23 we apply an EXP command to the value in C22. This converts the log odds to raw odds. The formula for cell C23 is:

```
=EXP(C22)
```

Finally, in cell C24 we convert the value from C23 into a probability using the following formula:

```
=(C23/(1+C23))
```

The resulting value in cell C24 is the probability of a home team win. The remaining probability is the probability of an away team win (1-C24). We now have probability outputs from out logistic regression that we can backtest and experiment with.

[1] The Real Statistics add-on can be buggy on the Mac. It still works but some graphs fail to load properly. Nothing to worry about.

13

THE BENTER BOOST

The Benter Boost is a technique that can be used to improve the accuracy of your ensemble model’s probability forecast.

But first…

Let me begin with a little story.

When I was about 8 years old, my father bought an MS-DOS computer chess program called Chess Master. If you don’t remember it, it featured an old mysterious wizard-looking man on the cover of the floppy disk. I remember spending many hours trying to beat that infuriating wizard and on all but the easiest of difficulty settings it was futile. The Chess Master was infinitely sharper than my 8 year old mind.

> Chess Master: 457
>
> 8 yr. old ginger: 0

In my frustration, I changed the game settings to maximum difficulty, and opted to move second as the black team rather than the white. My idea was simple: whatever the Chess Master did during the game, I would mirror exactly. “Let’s see if you can beat yourself, you crotchety old wizard.” I probably didn’t think those exact words, but it was one of those dumb little ideas that was so simple it almost worked. When my father came home from work later that evening, he saw that I had the Chess Master down to his last two pieces on the board and was playing for a stalemate. When he asked me how I had gotten that far, I answered “Whatever he does, I do.” Monkey see, monkey do. Basically.

Sometimes there is an advantage in recognizing you are up against a superior opponent. In this case, I had used that opportunity in my own little 8 year-old way to produce a breakeven expectancy.

Let's connect this to what we're trying to do with sports modelling by transitioning to a discussion about William Benter. For those who may not know, William Benter was an incredibly successful horse racing better and modeller that made an impressive fortune on the Hong Kong horse racing circuit. Perhaps even more incredibly, he wrote extensively about how he did it. If you haven't read any of his work on modelling, you really should[1].

I found the idea of the "Benter Boost" in one of Mr. Benter's papers on modelling horse racing in the early 90's. It reminds me analogously of my childhood Chess Master strategy. Not only has it produced an accuracy boost to my Excel models that I think is reasonably clever, it serves as a good reminder to read as much as you can about other attempts people have made to solve sports modelling problems you are trying to tackle. Approach your learning process like an ensemble and try to absorb what is useful from others.

So what is the "Benter Boost"?

Quite simply, Mr. Benter realized early on, as we all implicitly understand these days, that there is a tremendous amount of information contained inside the betting line of a market. If we think back to our previous discussion about Joseph Buchdahl's wisdom of the crowd model and its Bayesian implications, it's not hard to see why this would be the case.

Here's where it gets interesting. Since what we're ultimately trying to do when creating an ensemble is bring together a collection of models with a diverse group of opinions on a game forecast, *why not make the current betting line being offered on the game one of those models?*

This is precisely what Benter did. He built one of his horse racing models by effectively combining his forecast with the current implied probabilities of the betting market via a logit model[2]. The result was a forecast that included powerful information already incorporated into the line, plus any additional indications from his model that either added or subtracted from the line's projection. It made his projections very sharp. He had the Chess Master trying to beat himself.

While tethering your model to the current market price like this reduces the strength of your ensemble model's signals, it greatly improves its overall reliability. Effectively what we've done is found a way to supplement the current betting wisdom of the crowd with whatever our model might add. It's one of those techniques I wish I'd thought of on my own, but I'll settle for showing you how to apply it to our ensemble models in Excel.

There are a few things to be aware of when using this. You can't just incorporate any old book's line. As you're probably aware, some books (Pinnacle *cough cough*) are much sharper than others. When using this strategy you'll want to incorporate the sharpest market you can find, which only makes sense. Also, I only recommend this technique for situations when your model or ensemble is clearly outclassed by the efficiency of the market. I used this technique originally to post a profit betting on the English Premier League[3] - a tough market to beat with a standalone model. If you're betting softer markets like prop bets or obscure sports, this won't do you nearly as much good because the underlying assumption of this method is that the aggregate market forecast is usually better than your model. In soft markets you want your models to find the very real price inefficiencies that exist and for that reason we shouldn't want to track the crowd with our model forecast too closely. As an example, prop bet lines are frequently way off. Big time. The Benter Boost however is an interesting technique to help you compete with the big boys in major markets. Experiment with it accordingly.

Weighting The Benter Boost

As it was presented, Mr. Benter seems to have used a logit model that optimizes the coefficient weights using maximum likelihood estimation. The process requires a significant amount of data to perform well. However, using his basic model structure I think there's an ad hoc weighting method that can provide a "good enough" solution in many instances.

This quick and dirty weighting can be found by optimizing the ensemble forecast and the market forecast proportionally to the aggregate efficiency of the closing line. For example, if you were targeting a point spread wager and determined that at a given sportsbook the closing line was ~92% efficient, you'd weight the market forecast at 0.92 and the ensemble forecast at 0.08.

You can determine the efficiency of a point spread market like this by running a linear regression using the final margin of victory in your dataset of games as the Y variable and the closing market point spread as the X variable. The resulting coefficient should give you an indication of the aggregate efficiency. In major markets you'll find it is shockingly high[4].

What about the Vig?

There is an argument to be made that incorporating the current implied probabilities of the market won't do us much good if those probabilities are artificially inflated by the bookmaker's commission. It certainly seems logical that we must first attempt to remove the vigorish from the posted line in order to make the underlying probabilities a useful addition to our ensemble. Mr. Benter never articulated this point specifically, so you should be aware that I'm taking a few liberties with my interpretation of his writing. That being said, I think this is something we must seriously consider.

There are a number of ways to remove the vig from the current betting line.[5] While we don't know exactly which method best approximates a bookmaker's baked-in commission, several papers I've read suggest that the logarithmic method does a decent job for binomial outcome wagers.[6]

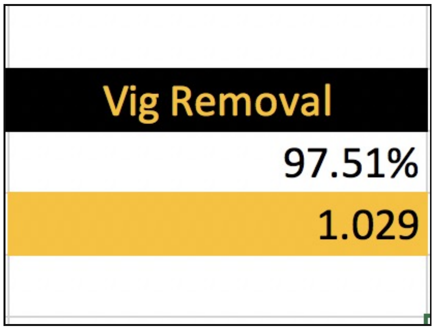
The Logarithmic Vig Removal Method

For a complete explanation of this method, I would recommend you read the excellent work author Joseph Buchdahl has done on the subject[7]. He explains the technical details far better than I would be able to, and has an Excel file available for free download that performs a number of vig removal techniques.[8] What follows is my implementation of the logarithmic method in Excel using solver.

Let's open the dashboard sheet inside the "Tutorial" Excel file. You'll see my adaptation of the logarithmic method starting in cell D21.

Step 1: Estimate No-Vig Odds

First, I enter the decimal odds for each team from our sharp sportsbook source in cells D2 and D3. You'll notice after entering these prices that the cumulative probability found in cell D22 is either more or less than 100%. This is normal, it simply indicates we have to optimize the exponent in cell D23 to strip the estimated commission from the prices.



Vig Removal

97.51%

1.029

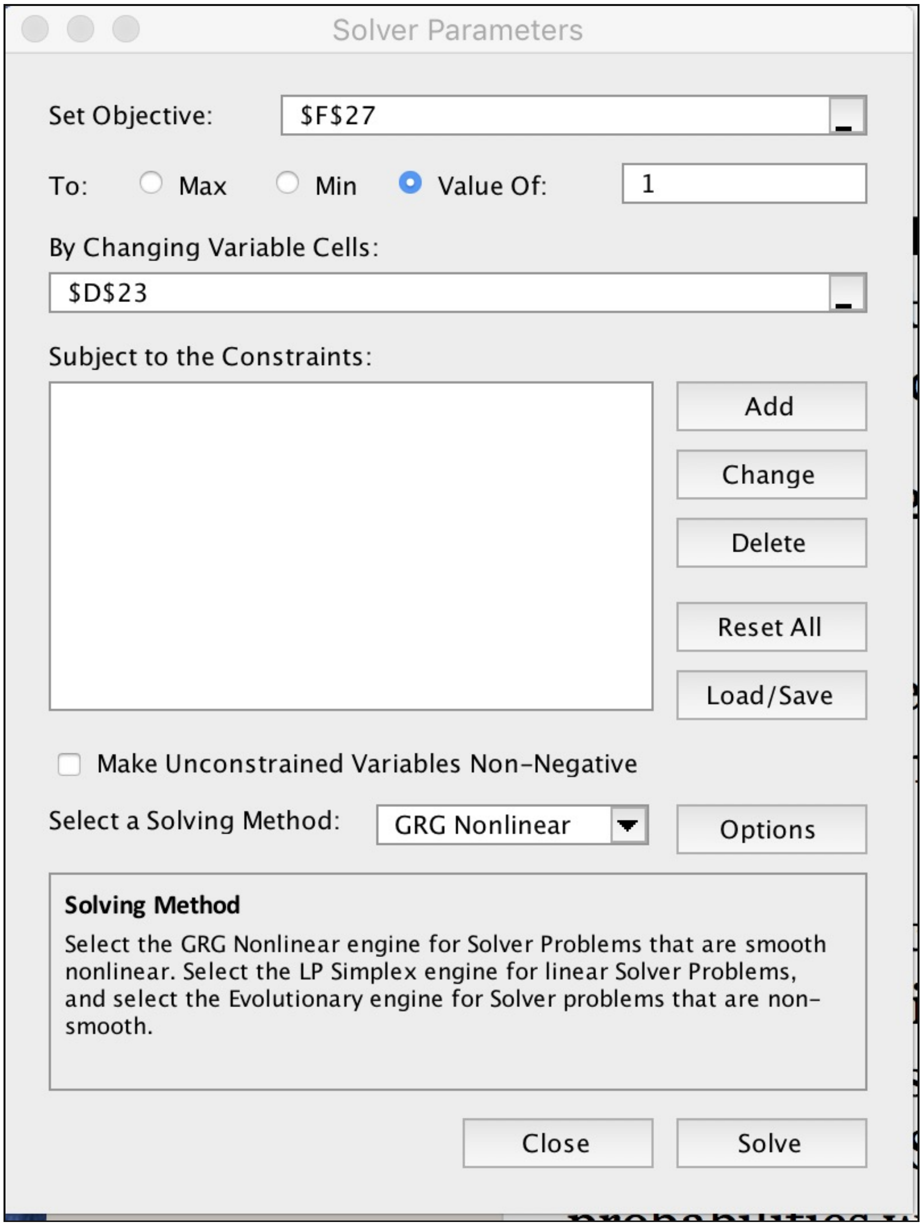
Cells D17 and D18 use the estimated exponent in cell D23 to estimate the no-vig odds using the following formula for each team respectively:

```
=(1/D2)^D23
```

Once we enter the prices for each team, we click on "Data" and then "Solver" to optimize the Log Exponent so that the total probability in cell D22 equals 100%.

Set the target cell to D22 and select "Value of" and then enter "1" in the available entry box. In the "By Changing Cells" entry box, enter

D23. Then, click “Solve”. Once Solver is finished, the no-vig probabilities will now be seen in cells D17 and D18.



Set Objective: $F$27

To: ○ Max ○ Min ● Value Of: 1

By Changing Variable Cells:
$D$23

Subject to the Constraints:

□ Make Unconstrained Variables Non-Negative

Select a Solving Method: GRG Nonlinear

Solving Method
Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

These are the estimated true market forecast probabilities that we will be using in our Benter Boost.

Step 2: Convert Probabilities to Log Odds

Next, we convert the ensemble forecast and no-vig market forecast into log odds by using the LN command in Excel. You’ll see that this has been completed in cells E17, E18, F17 and F18.

Step 3: Apply Weights

In cells E22, E23, F22 and F23, you'll see I have applied weights to each of the log odds outputs from the previous step. Here we have used a hypothetical 80% market efficiency, so the ensemble is weighted at 20% and the market forecast is weighted at 80%. This is simply (log odds * weight) for each of the four log odds outputs from the previous step.

Step 4: Calculate Benter Boost Probabilities

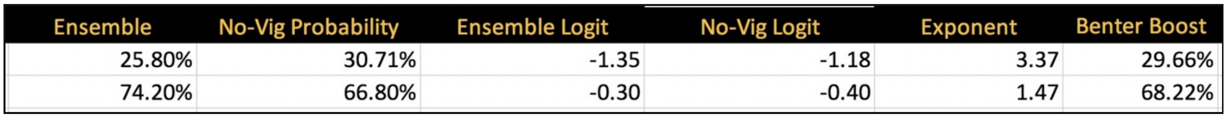
After that, we use the basic formula from Mr. Benter's paper combined with our weights to derive the exponent output found in cells G17 and G18. The formula for each respective team is as follows:

```
=EXP(E22+F22)/SUM(EXP(E22+F22+F23+E23))
```

Finally, to turn this into adjusted probabilities, in cells H17 and H18 we apply the following formula analogously for each team:

```
=G17/(G17+G18)
```

The resulting probabilities are a weighted output of our ensemble probabilities and the no-vig market probabilities. It should go without saying the weights should be adjusted to your estimated market efficiency. We can then take these adjusted probabilities and calculate the fair prices for each game. I have completed this step in cells I17 and I18.



Ensemble	No-Vig Probability	Ensemble Logit	No-Vig Logit	Exponent	Benter Boost
25.80%	30.71%	-1.35	-1.18	3.37	29.66%
74.20%	66.80%	-0.30	-0.40	1.47	68.22%

Finally, you can input the prices from another sportsbook in cells E2 and E3 and find the estimated value for each side of the wager in cells J17,J18, K17 and K18. Using the same sharp sportsbook odds also works, but will naturally show much slimmer values.

---

[1] https://www.gwern.net/docs/statistics/decision/1994-benter.pdf

[2] This appears very similar to a technique described as “logarithmic opinion pooling”[LoOp] in another research paper I’ve read.

[3] It was an admittedly small sample size of ~40 wagers, hardly definitive proof of anything. The model I achieved this with is included in the chapter on specific models for the EPL.

[4] Naturally, the lower the efficiency the more opportunity for modelling to provide an edge. It’s not a bad idea to get a baseline efficiency estimate for all markets you are interested in modelling to get an idea of the degree of difficulty that lies ahead.

[5] https://www.football-data.co.uk/true_odds_calculator.xlsm

[6] Vig removal on wagers with multiple possible outcomes like horse racing appears to be better accomplished by Shin probabilities.

[7] https://www.football-data.co.uk/wisdom_of_crowd_bets

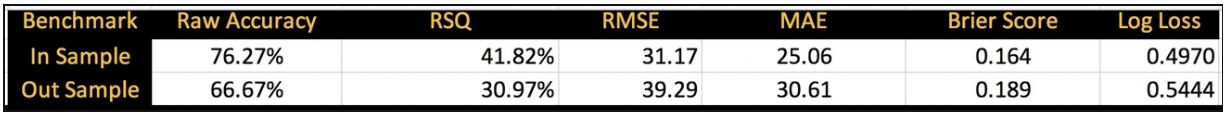
[8] https://www.football-data.co.uk/true_odds_calculator.xlsm

Backtesting is a subject that is both crucially important and vast in scope. You could write an entire book on backtesting methodology and still have things left to say about it.

That being said, no book on sports modelling could be considered complete without at least a cursory look at backtesting. Let’s talk about how to tell if your models show any indications of life and link this with the performance metrics discussed earlier in chapter 4.

In-Sample vs. Out-Sample

If you look at the tops of columns P through V in each of the individual models and also in the center of the dashboard sheet of our “Tutorial” Excel file, you’ll notice that our performance metrics are shown separated as “In-Sample” and “Out-Sample”.



Benchmark	Raw Accuracy	RSQ	RMSE	MAE	Brier Score	Log Loss
In Sample	76.27%	41.82%	31.17	25.06	0.164	0.4970
Out Sample	66.67%	30.97%	39.29	30.61	0.189	0.5444

In-Sample refers to the performance of the model on the data that was used to train the model. It’s data that the model has already “seen” as part of its optimization process. Out-Sample on the other hand is new data that has been purposely withheld from the model during its optimization in order to test model performance on new data. This is the performance we are most interested in, as it gives us an indication of how well the model might be expected to do at predicting games it has never “seen” before. We can then take these metrics and compare models to each other to get a sense of which models are better or worse than others. The simplest way to achieve this in Excel is with a holdout sample - that is, a portion of the dataset withheld from the optimization data on purpose to test model performance.

Holdout Sample

If we go to the Bradley Terry model sheet inside the "Tutorial" Excel file, you'll see by clicking on cells Q2 and Q3 that we optimized the ratings for the model using the game results data only from rows O5:O181 (the "in-sample"). We then tested the model using data the model hadn't been optimized on from rows O182:O202 (the "out-sample"). Generally a good rule of thumb is to use 80-90% of your data for training and 10-20% for testing. In this admittedly small sample demonstration, you can see that we have a total dataset of 197 games of which we used 177 for optimization ("training") and 20 games for testing.[1] This is accomplished in this model sheet by constraining the probability product in cell K5 to only include data from cells J5 to J181. The process works comparably for the SSE optimization models as well. It is important to remember that this will have to be adjusted for your own sports data if the data has more rows (observations) than the example I've used. Try to partition the data as explained above and adjust the model settings accordingly.

Out of Sample Profits

Of course, being able to determine which of the models works "the best" is relative. The only metric that we truly care about is: *do these models make money?* In order to find out, we can incorporate sportsbook odds and run an out-sample profits test to see how a model would've performed using the available odds on games it hasn't seen before. We can then take the number of bets, the profit/loss and the yield and look for indications that we might have something statistically significant.[2]

At this point, I have some bad news: it is exceedingly difficult to arrive at the conclusion that our results are statistically significant. It requires a betting sample of thousands of wagers, and even a normally impressive p-value like 0.001 can and does occur purely from chance rather than skill.[3] To quote Joseph Buchdahl:

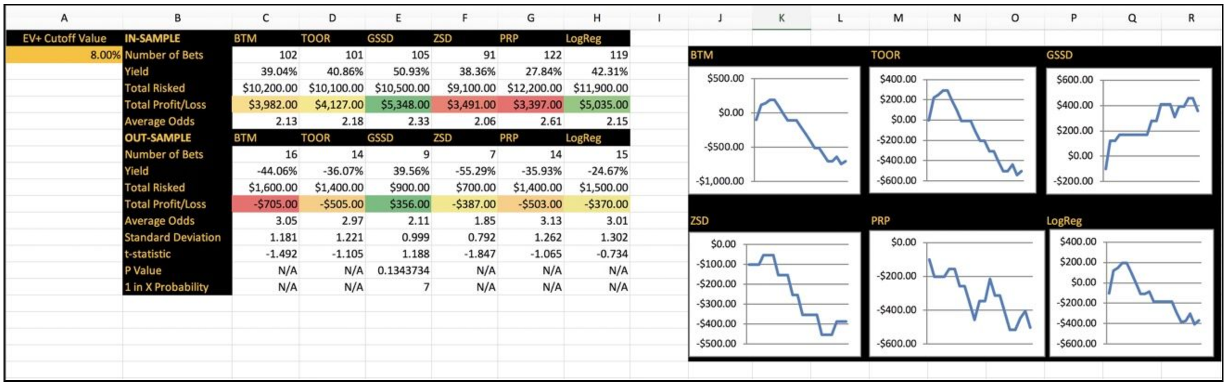
> A p-value of 0.01 means there is a 1% probability a record can arise by chance assuming no skill is present. When p-values are below 0.01 statisticians will typically interpret this to mean that this implies

> there **IS** something else going on like skill, although [you] should not assume this to be the case. For example, if a 1,000 punters submitted betting records, we would expect to see the best one with a p-value of 0.001 arising simply because of chance and no skill.

This means that with our tutorial out-sample of 20 games, any positive results are wildly below the threshold needed to make a determination on whether we have a reasonable expectation of profit in the future. If you placed 20 consecutive bets on a roulette wheel, you might easily show a profit by chance despite having a negative expectancy. It would be foolish to declare that you have found a winning system at that point. Much more data is required. With this in mind as a strong caution against overconfidence, let's walk through basic backtesting and game filtering. Once you get a feel for how it works, you can apply it to your (hopefully much larger) datasets.

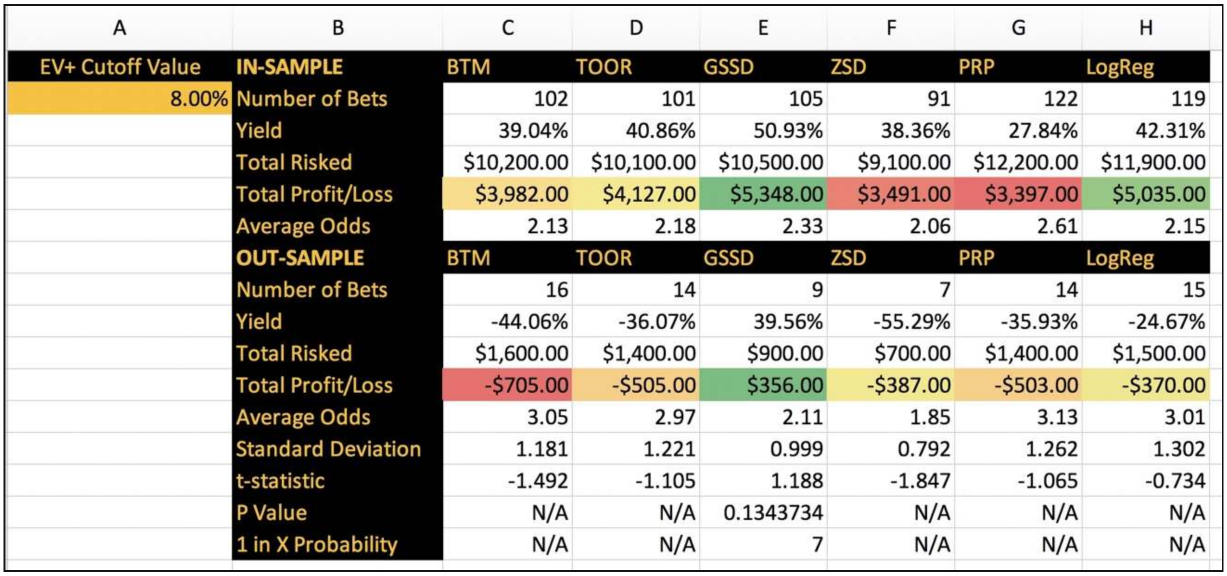
The Backtesting Excel File

Let's open the "Backtesting" Excel file. You should see a dashboard featuring in-sample and out-sample profit tests for all individual models and the logistic regression ensemble model as well. All estimated profits in this file are based on a simple flat staking plan of $100 per wager. From this we can see that the GSSD model on its own is showing a modest out-sample profit of $356 over 9 bets, possibly giving credence to our p-value findings from the earlier chapter on ensemble methods. Since out-sample is where we're looking for signs of life, it's wise to disregard the seemingly impressive but misleading profit tests from the in-sample portion of the sheet up top.



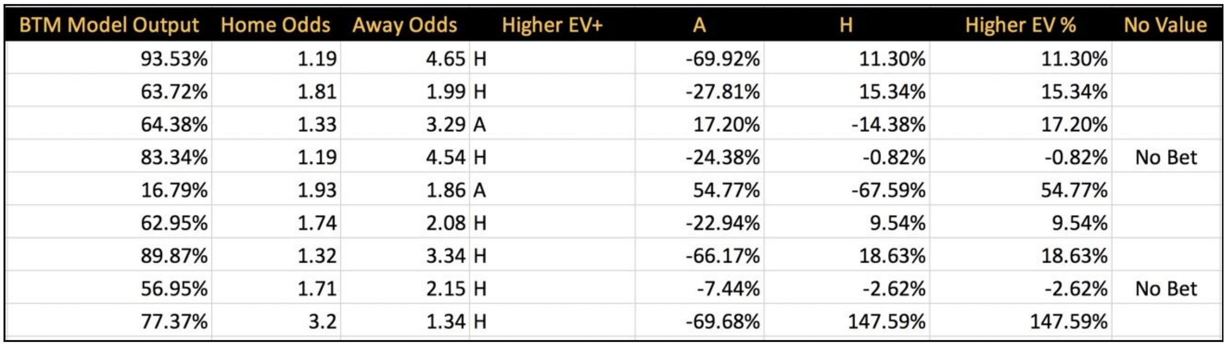
A	B	C	D	E	F	G	H
EV+ Cutoff Value	IN-SAMPLE	BTM	TOOR	GSSD	ZSD	PRP	LogReg
8.00%	Number of Bets	102	101	105	91	122	119
	Yield	39.04%	40.86%	50.93%	38.36%	27.84%	42.31%
	Total Risked	$10,200.00	$10,100.00	$10,500.00	$9,100.00	$12,200.00	$11,900.00
	Total Profit/Loss	$3,982.00	$4,127.00	$5,348.00	$3,491.00	$3,397.00	$5,035.00
	Average Odds	2.13	2.18	2.33	2.06	2.61	2.15
	OUT-SAMPLE	BTM	TOOR	GSSD	ZSD	PRP	LogReg
	Number of Bets	16	14	9	7	14	15
	Yield	-44.06%	-36.07%	39.56%	-55.29%	-35.93%	-24.67%
	Total Risked	$1,600.00	$1,400.00	$900.00	$700.00	$1,400.00	$1,500.00
	Total Profit/Loss	-$705.00	-$505.00	$356.00	-$387.00	-$503.00	-$370.00
	Average Odds	3.05	2.97	2.11	1.85	3.13	3.01
	Standard Deviation	1.181	1.221	0.999	0.792	1.262	1.302
	t-statistic	-1.492	-1.105	1.188	-1.847	-1.065	-0.734
	P Value	N/A	N/A	0.1343734	N/A	N/A	N/A
	1 in X Probability	N/A	N/A	7	N/A	N/A	N/A

In the top left corner in cell A2, you'll see a cutoff value input. This is a form of game filtering. If we set it to 0%, the backtesting sheets hypothetically wager on any game with an expected value over 0%. If we change this to only consider games with an +EV value over a given percentage like 8%, you can observe the backtesting values change.



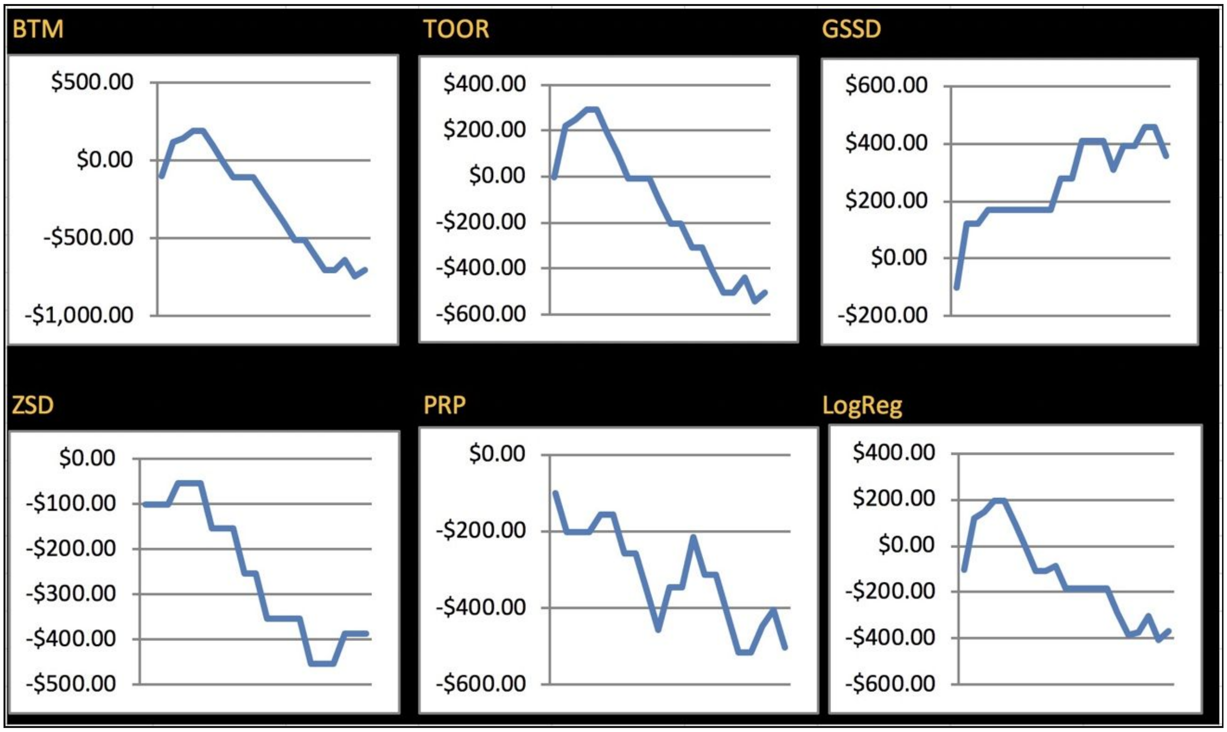
A	B	C	D	E	F	G	H
EV+ Cutoff Value	IN-SAMPLE	BTM	TOOR	GSSD	ZSD	PRP	LogReg
8.00%	Number of Bets	102	101	105	91	122	119
	Yield	39.04%	40.86%	50.93%	38.36%	27.84%	42.31%
	Total Risked	$10,200.00	$10,100.00	$10,500.00	$9,100.00	$12,200.00	$11,900.00
	Total Profit/Loss	$3,982.00	$4,127.00	$5,348.00	$3,491.00	$3,397.00	$5,035.00
	Average Odds	2.13	2.18	2.33	2.06	2.61	2.15
	OUT-SAMPLE	BTM	TOOR	GSSD	ZSD	PRP	LogReg
	Number of Bets	16	14	9	7	14	15
	Yield	-44.06%	-36.07%	39.56%	-55.29%	-35.93%	-24.67%
	Total Risked	$1,600.00	$1,400.00	$900.00	$700.00	$1,400.00	$1,500.00
	Total Profit/Loss	-$705.00	-$505.00	$356.00	-$387.00	-$503.00	-$370.00
	Average Odds	3.05	2.97	2.11	1.85	3.13	3.01
	Standard Deviation	1.181	1.221	0.999	0.792	1.262	1.302
	t-statistic	-1.492	-1.105	1.188	-1.847	-1.065	-0.734
	P Value	N/A	N/A	0.1343734	N/A	N/A	N/A
	1 in X Probability	N/A	N/A	7	N/A	N/A	N/A

The individual sheets themselves use a somewhat tedious series of columns to filter and select the games based on the expected value and the cutoff percentage which you can adjust manually in the dashboard.



BTM Model Output	Home Odds	Away Odds	Higher EV+	A	H	Higher EV %	No Value
93.53%	1.19	4.65	H	-69.92%	11.30%	11.30%	
63.72%	1.81	1.99	H	-27.81%	15.34%	15.34%	
64.38%	1.33	3.29	A	17.20%	-14.38%	17.20%	
83.34%	1.19	4.54	H	-24.38%	-0.82%	-0.82%	No Bet
16.79%	1.93	1.86	A	54.77%	-67.59%	54.77%	
62.95%	1.74	2.08	H	-22.94%	9.54%	9.54%	
89.87%	1.32	3.34	H	-66.17%	18.63%	18.63%	
56.95%	1.71	2.15	H	-7.44%	-2.62%	-2.62%	No Bet
77.37%	3.2	1.34	H	-69.68%	147.59%	147.59%	

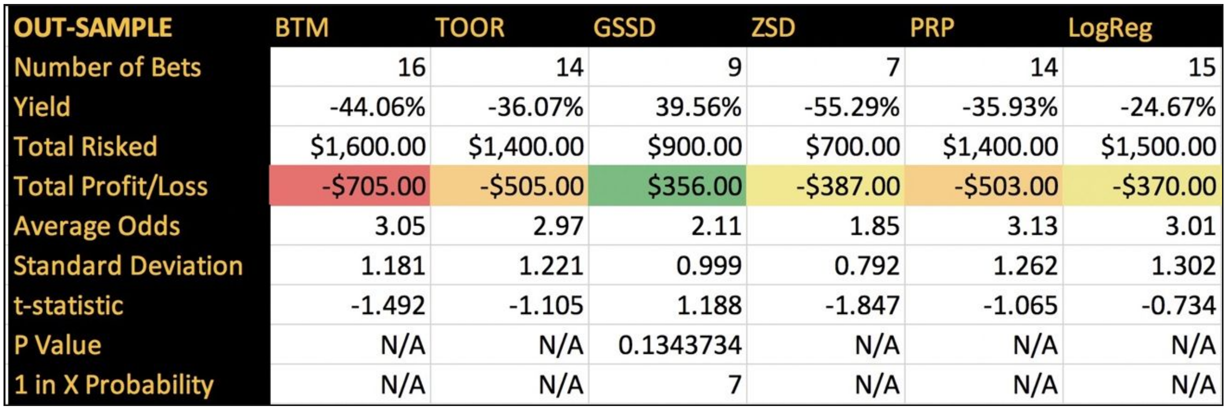
The accompanying dashboard graphs give a simple visual indication of what looks promising and what doesn't. In this case, only the GSSD model is showing a profit. However, as mentioned previously with such a tiny sample of games neither the positive or negative results are particularly meaningful - we'd want to see an out-sample of hundreds if not thousands of games before we can make any reasonable conclusions.



To modify the individual sheets for your own use, all you have to do is change the model probability inputs in column H and then adjust the performance metrics for the number of rows in your dataset.

Searching for Significance

Borrowing Excel formulas from Joseph Buchdahl's yield calculator[4], we can calculate the likelihood of producing our model results by chance and search for hints of statistical significance. You'll find my implementation of the yield calculator formulas in rows 13 to 16 of the dashboard. Negative returns produce an "N/A" value, so only the GSSD model is currently producing any values we can analyze.



OUT-SAMPLE	BTM	TOOR	GSSD	ZSD	PRP	LogReg
Number of Bets	16	14	9	7	14	15
Yield	-44.06%	-36.07%	39.56%	-55.29%	-35.93%	-24.67%
Total Risked	$1,600.00	$1,400.00	$900.00	$700.00	$1,400.00	$1,500.00
Total Profit/Loss	-$705.00	-$505.00	$356.00	-$387.00	-$503.00	-$370.00
Average Odds	3.05	2.97	2.11	1.85	3.13	3.01
Standard Deviation	1.181	1.221	0.999	0.792	1.262	1.302
t-statistic	-1.492	-1.105	1.188	-1.847	-1.065	-0.734
P Value	N/A	N/A	0.1343734	N/A	N/A	N/A
1 in X Probability	N/A	N/A	7	N/A	N/A	N/A

You'll see that the p-value for the GSSD model given the small sample size is 0.1343734, which is well above even a base level of 0.05. We can also see that we would arrive at similar results purely through chance roughly 1 in 7 times. This demonstrates the process fairly well, but our results aren't very determinative due to the sample size issue.

Quite simply, we can't say anything for sure about our AFL models based on this test. Hopefully though, you have a better sense of how the process of backtesting works after walking through this example. In applying this to your sport of choice with a larger sample size you'll be able ground your modelling efforts in a healthy skepticism and be that much more confident when you have enough data to to make a determination with increased statistical rigour.

[1] This is not a large enough sample to confidently say much about how the model might perform in the future. More data is always better. When more data is not available, bootstrapping the samples is one possibility.

[2] http://www.football-data.co.uk/blog/model_testing.php

[3] http://www.football-data.co.uk/blog/P-value_calculator.xlsx

[4] http://www.football-data.co.uk/blog/P-value_calculator.xlsx

15

Earlier this year I bet on the winner of the NBA 3Pt competition during all-star weekend. The Monte Carlo simulation I used to forecast the competition indicated Joe Harris was a great value, as was picking the proposition "anyone other than the Curry brothers". While I'm certainly no expert when it comes to Monte Carlo simulations, I'll walk you through the process I used to forecast this prop bet.

Thought Process: How to Solve This Problem?

How does one go about conceptualizing a problem like this? For me, when I think of prop bets I think of two elements: opportunity and efficiency.

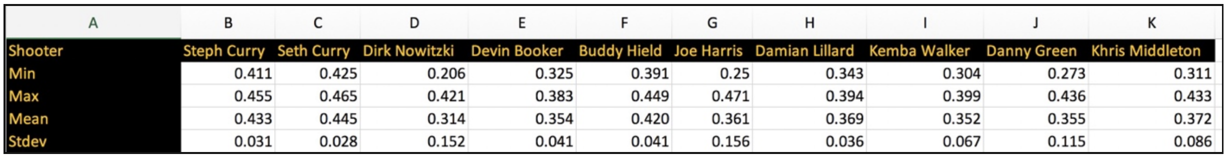
> How many attempts is player X likely to get, and how efficient is player X at converting those attempts into our target outcome?

For the 3Pt competition, we know how many shot attempts each player is going to receive and how many points each one is worth if successfully converted to a made basket. Since opposition defense isn't a concern, the remaining challenge is to decide how efficient each player is likely to be despite our sample size only amounting to a handful of shots. We need a way to factor this uncertainty into the equation. For this, I turned to a technique known as Monte Carlo simulation [MC].

MC is a method that essentially uses random sampling over a set period of trials (simulations) to infer an underlying probability distribution that is useful for calculating fair prices on a given wager[1]. With a little creative thinking we can set up an Excel spreadsheet that simulates 1,000 3Pt competitions and gives us a reasonable estimate for who is likely to win and how often. From there we can derive a price estimate and look for market value.

Getting Started

Open the “3Pt Monte Carlo” Excel file. The first thing I did was look at the 3-point percentages for every player in the competition over the regular season for the last 4 seasons. From these statistics, I took the maximum and minimum 3-point percentage from those years and entered it into a spreadsheet. Then, I calculated the mean 3-point percentage using the average of the maximum and the minimum. Next, I calculated the standard deviation from the maximum and the minimum for each player. Interestingly, while Joe Harris didn’t have the highest mean 3-point efficiency, he did have the largest standard deviation.

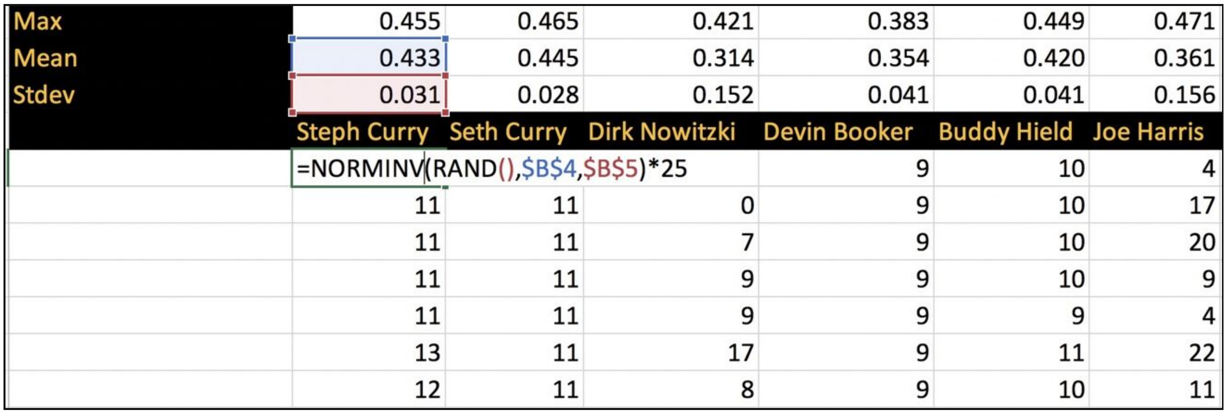


A	B	C	D	E	F	G	H	I	J	K
Shooter	Steph Curry	Seth Curry	Dirk Nowitzki	Devin Booker	Buddy Hield	Joe Harris	Damian Lillard	Kemba Walker	Danny Green	Khris Middleton
Min	0.411	0.425	0.206	0.325	0.391	0.25	0.343	0.304	0.273	0.311
Max	0.455	0.465	0.421	0.383	0.449	0.471	0.394	0.399	0.436	0.433
Mean	0.433	0.445	0.314	0.354	0.420	0.361	0.369	0.352	0.355	0.372
Stdev	0.031	0.028	0.152	0.041	0.041	0.156	0.036	0.067	0.115	0.086

Setting Up The Simulation

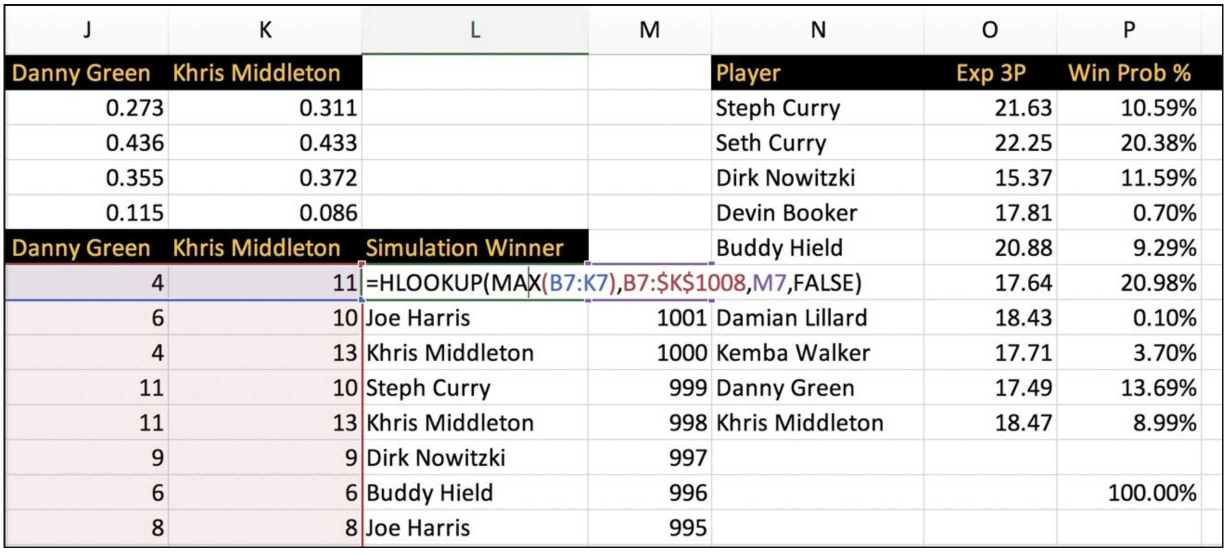
I set up a column for each player in the competition and used the following formula for each respective player:

```
=NORMINV(RAND(),$B$4,$B$5)*25
```



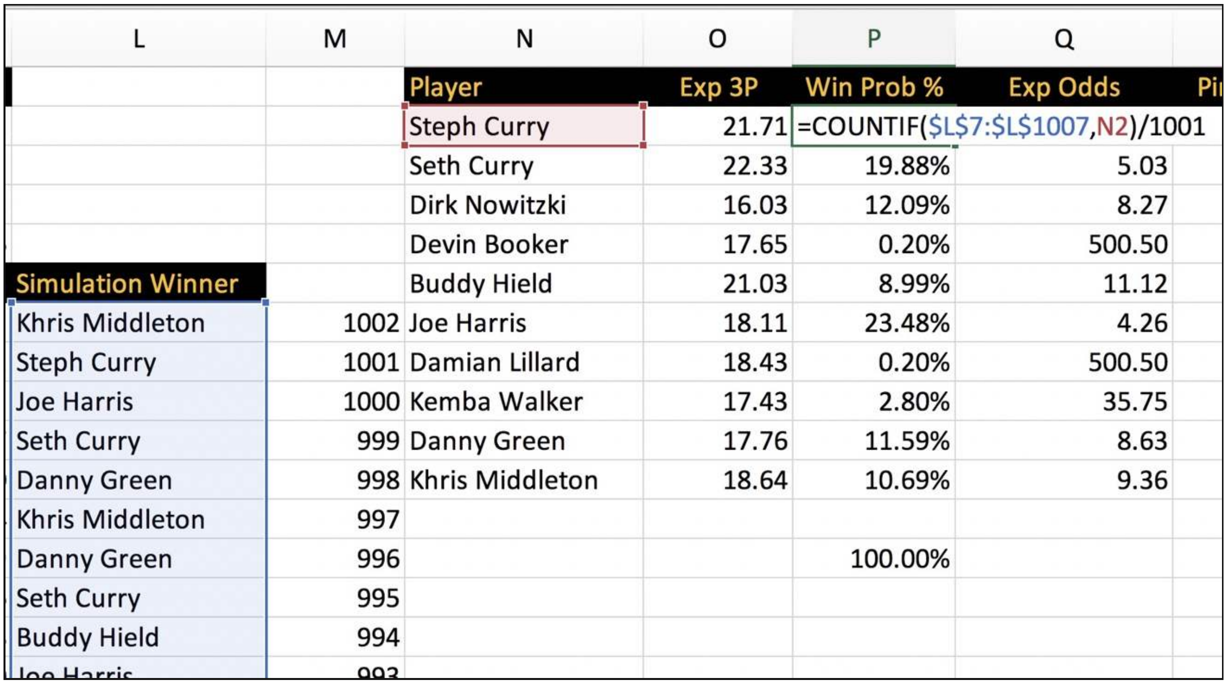
Max	0.455	0.465	0.421	0.383	0.449	0.471
Mean	0.433	0.445	0.314	0.354	0.420	0.361
Stdev	0.031	0.028	0.152	0.041	0.041	0.156
	Steph Curry	Seth Curry	Dirk Nowitzki	Devin Booker	Buddy Hield	Joe Harris
	=NORMINV(RAND(),$B$4,$B$5)*25			9	10	4
	11	11	0	9	10	17
	11	11	7	9	10	20
	11	11	9	9	10	9
	11	11	9	9	9	4
	13	11	17	9	11	22
	12	11	8	9	10	11

The RAND element randomly selects a simulated 3-point efficiency for a given player using the player's mean and standard deviation efficiency and then multiplies it by 25 to account for the shot attempts they will receive in the competition.[2] I did this for each player and then populated the columns down until 1,000 simulated 3pt competitions had been performed. Every time you press a key like [Delete] in a blank cell on the spreadsheet, the simulation will refresh. I used an HLOOKUP command to record the name of the winner in each simulation, then finished it off by calculating the percentage each player won the competition using a COUNTIF command.



J	K	L	M	N	O	P
Danny Green	Khris Middleton			Player	Exp 3P	Win Prob %
0.273	0.311			Steph Curry	21.63	10.59%
0.436	0.433			Seth Curry	22.25	20.38%
0.355	0.372			Dirk Nowitzki	15.37	11.59%
0.115	0.086			Devin Booker	17.81	0.70%
Danny Green	Khris Middleton	Simulation Winner		Buddy Hield	20.88	9.29%
4	11	=HLOOKUP(MAX(B7:K7),B7:$K$1008,M7,FALSE)			17.64	20.98%
6	10	Joe Harris	1001	Damian Lillard	18.43	0.10%
4	13	Khris Middleton	1000	Kemba Walker	17.71	3.70%
11	10	Steph Curry	999	Danny Green	17.49	13.69%
11	13	Khris Middleton	998	Khris Middleton	18.47	8.99%
9	9	Dirk Nowitzki	997			
6	6	Buddy Hield	996			100.00%
8	8	Joe Harris	995			

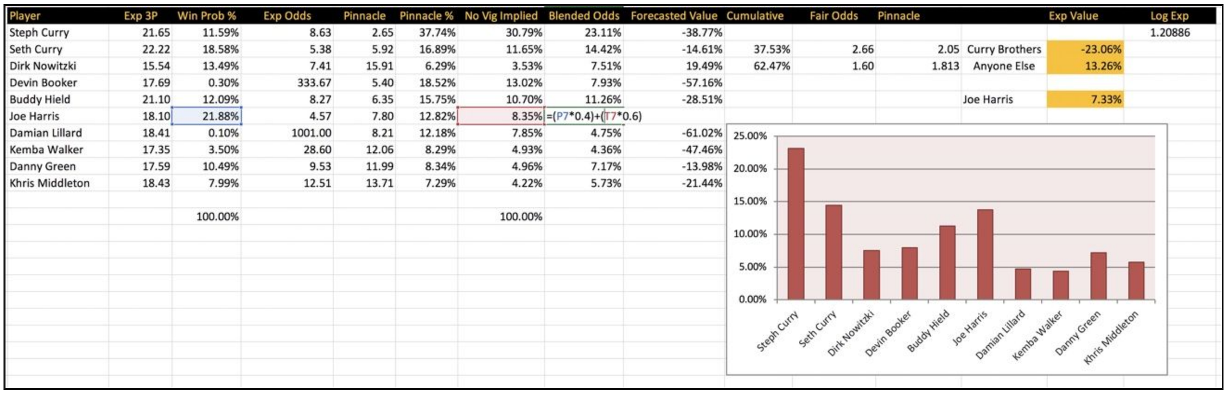
The output was the estimated win probabilities for each player.



L	M	N	O	P	Q	
		Player	Exp 3P	Win Prob %	Exp Odds	Pi
		Steph Curry	21.71	=COUNTIF($L$7:$L$1007,N2)/1001		
		Seth Curry	22.33	19.88%	5.03	
		Dirk Nowitzki	16.03	12.09%	8.27	
		Devin Booker	17.65	0.20%	500.50	
Simulation Winner		Buddy Hield	21.03	8.99%	11.12	
Khris Middleton	1002	Joe Harris	18.11	23.48%	4.26	
Steph Curry	1001	Damian Lillard	18.43	0.20%	500.50	
Joe Harris	1000	Kemba Walker	17.43	2.80%	35.75	
Seth Curry	999	Danny Green	17.76	11.59%	8.63	
Danny Green	998	Khris Middleton	18.64	10.69%	9.36	
Khris Middleton	997					
Danny Green	996			100.00%		
Seth Curry	995					
Buddy Hield	994					
Joe Harris	993					

The Curry brothers definitely looked competitive, but Joe Harris was clearly undervalued based on my calculations. Unsure if I'd made a mistake, I decided I wanted a second opinion.

To double-check my forecast against the market, I weighted my estimates with the no-vig Pinnacle line at a ratio of 40%/60%.[3] Joe Harris still showed decent value. The market appeared to be overvaluing Steph Curry and creating opportunities on other players. With a win probability hovering between 14% and 20% his expected value at 7.80 odds was forecasted to be around 8%. I also had a forecasted value of around 14% for taking anyone other than the Curry brothers to win at a price of 1.81.



Player	Exp 3P	Win Prob %	Exp Odds	Pinnacle	Pinnacle %	No Vig Implied	Blended Odds	Forecasted Value	Cumulative	Fair Odds	Pinnacle		Exp Value	Log Exp
Steph Curry	21.65	11.59%	8.63	2.65	37.74%	30.79%	23.11%	-38.77%						1.20886
Seth Curry	22.22	18.58%	5.38	5.92	16.89%	11.65%	14.42%	-14.61%	37.53%	2.66	2.05	Curry Brothers	-23.06%	
Dirk Nowitzki	15.54	13.49%	7.41	15.91	6.29%	3.53%	7.51%	19.49%	62.47%	1.60	1.813	Anyone Else	13.26%	
Devin Booker	17.69	0.30%	333.67	5.40	18.52%	13.02%	7.93%	-57.16%						
Buddy Hield	21.10	12.09%	8.27	6.35	15.75%	10.70%	11.26%	-28.51%				Joe Harris	7.33%	
Joe Harris	18.10	21.88%	4.57	7.80	12.82%	8.35%	=(P7*0.4)+(T7*0.6)							
Damian Lillard	18.41	0.10%	1001.00	8.21	12.18%	7.85%	4.75%	-61.02%						
Kemba Walker	17.35	3.50%	28.60	12.06	8.29%	4.93%	4.36%	-47.46%						
Danny Green	17.59	10.49%	9.53	11.99	8.34%	4.96%	7.17%	-13.98%						
Khris Middleton	18.43	7.99%	12.51	13.71	7.29%	4.22%	5.73%	-21.44%						
		100.00%				100.00%								

The rest is history. With a little forecasting and a whole lot of luck, Harris rained down 3-pointers and managed to edge out Steph Curry on route to cashing both my wagers. He may have won the competition, but I'm almost positive I was happier about it than he was.

I'm sure if you use your imagination you'll find many other ways to apply this technique to new situations. Happy edge hunting.

---

[1] Incidentally, if you end up reading more about Bayesian statistical methods, you'll see MC come up often as a method to estimate prior distributions.

[2] This doesn't account for the extra points the moneyball is worth, but gives an idea of expected made baskets.

[3] I used the logarithmic vig removal method explained in earlier chapters, optimizing cell T13 to a value of 1 by changing the value in cell AB2.

16

Up until now we've focused mostly on generalized sport models that use the normal distribution for modelling margin of victory. For modelling specific point outcomes other discrete distributions are more appropriate. One NFL specific model I'd like to show you takes the ZSD model combined with a competing negative binomial distribution to determine win probability as well as total score probability estimates.

After a few unsuccessful attempts I was finally able to reconfigure a negative binomial distribution to calculate the probability of a certain point outcome, similar to the manner in which many bettors use the Poisson distribution.[1] The required inputs are the expected mean score for each team and the variance in team scoring. The Excel file I've created gathers this data from the ZSD model and inputs these values automatically. The relevant parameters for the negative binomial distribution are estimated and the lengthy calculation column on each distribution sheet does the rest.

When we've completed this for both teams, the matrix sheet multiplies the possible score probabilities together in a competing distribution matrix. We then sum diagonal sections of the matrix to derive win probabilities, margins of victory and total over/under probabilities. These values are returned when you enter the teams for a given matchup into the game prediction function of the ZSD model sheet starting in column AZ.

Simply input game result data, optimize the model and then enter teams for any game you want to forecast. This model is ready to go right out of the box but I'll walk you through each section of the Excel file so that you have a better understanding of the functions involved.

Main ZSD Model Sheet

When you first open up the "Competing NegBinomial NFL" sheet, you'll notice that it's the ZSD model with a few changes. Columns A through R are new. These columns are where I copy and paste the raw game result data from pro-football-reference.com, and due to the way the data is organized on the website a bit of wrangling is necessary.

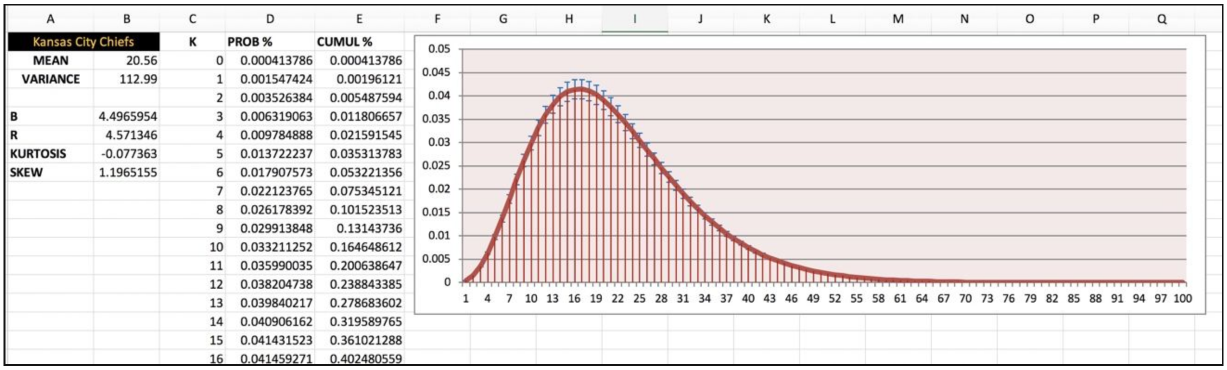
The website data lists teams as “winner” or “loser” which isn’t ideal as we’d rather know who was at home and away and use the scores to figure the rest of the details out. I’ve used a few simple IF commands to extract this data into columns O, P, Q and R so that it can be seamlessly inputted into our standard ZSD model sheet. I’ve also added two cells that calculate the variance of scoring for home and away teams in X2 and Y2. We’ll need these for using the negative binomial distributions later on.

The rest of the model sheet works the same as you’ve seen before. Input game result data, optimize the model using solver, run a linear regression and copy and paste the coefficients and standard error in the appropriate places on the sheet.

The game prediction function starting in column AZ works the same too - the only exception being that we’ll be using our negative binomial distribution matrix to derive the probabilities found in column BE.

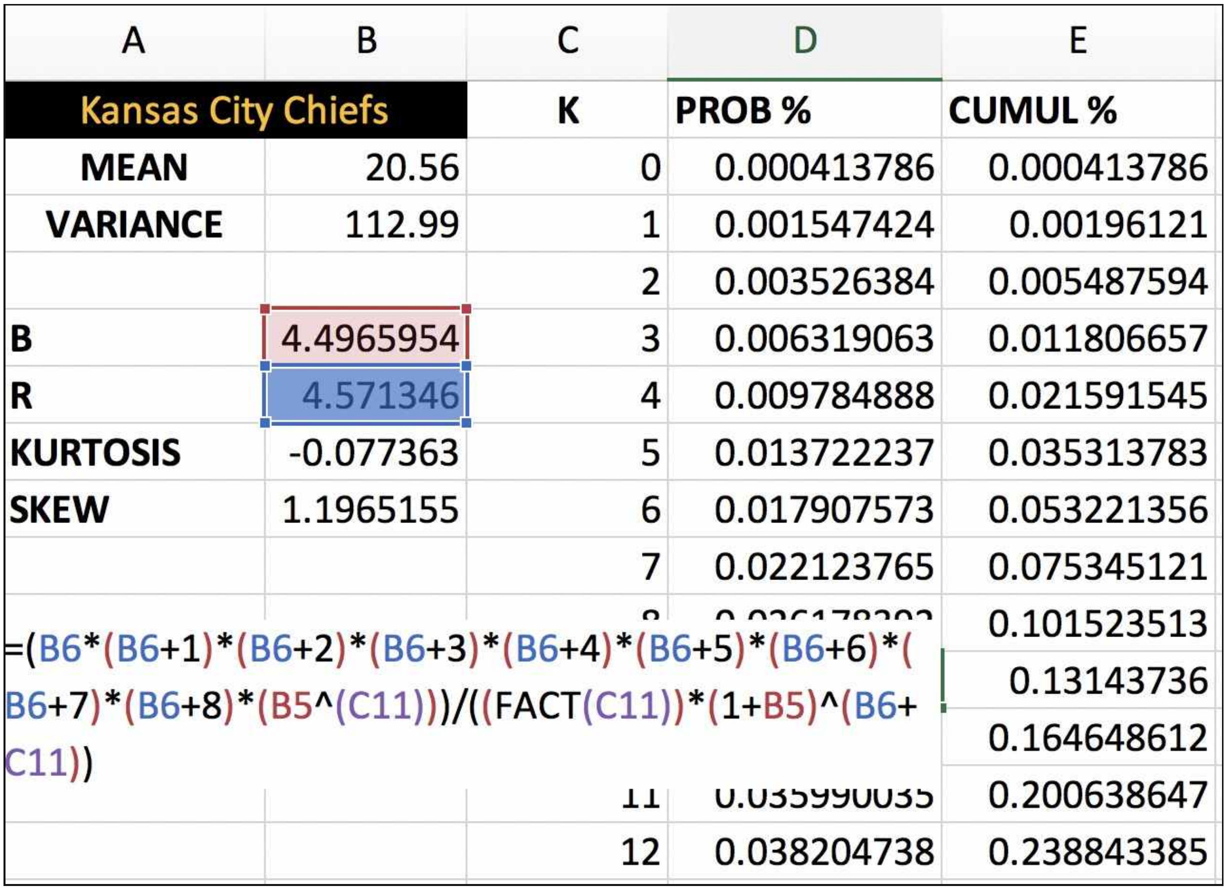
NegBinomial Sheets

You’ll see two negative binomial distribution sheets - one for the home team and one for the away team. These sheets take the expected points value from column BD and the variance values from cells X2 and Y2 from the main sheet and calculate the probabilities of each respective team scoring a certain number of points. This is all done automatically. As we discussed in earlier chapters, the negative binomial is a discrete distribution that allows for variance that exceeds the mean. We can see from comparing the mean and variance in main sheet cells X2 and Z2 that this is indeed the case here. As a result, this distribution models NFL scores decently.



A	B	C	D	E
Kansas City Chiefs		K	PROB %	CUMUL %
MEAN	20.56	0	0.000413786	0.000413786
VARIANCE	112.99	1	0.001547424	0.00196121
		2	0.003526384	0.005487594
B	4.4965954	3	0.006319063	0.011806657
R	4.571346	4	0.009784888	0.021591545
KURTOSIS	-0.077363	5	0.013722237	0.035313783
SKEW	1.1965155	6	0.017907573	0.053221356
		7	0.022123765	0.075345121
		8	0.026178392	0.101523513
		9	0.029913848	0.13143736
		10	0.033211252	0.164648612
		11	0.035990035	0.200638647
		12	0.038204738	0.238843385
		13	0.039840217	0.278683602
		14	0.040906162	0.319589765
		15	0.041431523	0.361021288
		16	0.041459271	0.402480559

Once the mean and variance automatically populate in the home and away distribution sheet, parameters B and R (as well as kurtosis and skew) are calculated for the distribution.[2] From there, the calculations apply parameters B and R to derive probabilities for each possible team score using a fairly lengthy Excel formula. No need to tinker with this, but you can double click on the individual cells to see what I've done. It took longer to figure this out than I'd care to admit.



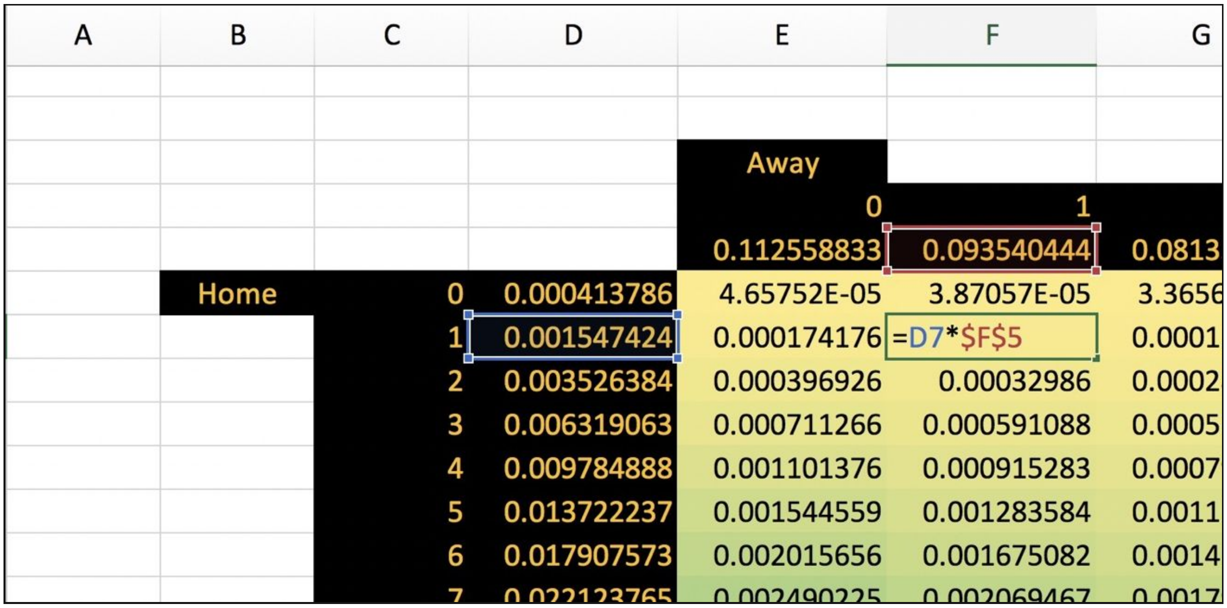
A	B	C	D	E
Kansas City Chiefs		K	PROB %	CUMUL %
MEAN	20.56	0	0.000413786	0.000413786
VARIANCE	112.99	1	0.001547424	0.00196121
		2	0.003526384	0.005487594
B	4.4965954	3	0.006319063	0.011806657
R	4.571346	4	0.009784888	0.021591545
KURTOSIS	-0.077363	5	0.013722237	0.035313783
SKEW	1.1965155	6	0.017907573	0.053221356
		7	0.022123765	0.075345121
		8	0.026178392	0.101523513
				0.13143736
				0.164648612
		11	0.035990035	0.200638647
		12	0.038204738	0.238843385

```
=(B6*(B6+1)*(B6+2)*(B6+3)*(B6+4)*(B6+5)*(B6+6)*(
B6+7)*(B6+8)*(B5^(C11)))/((FACT(C11))*(1+B5)^(B6+
C11))
```

This process is applied to both home and away teams, giving us probabilities for each team scoring every possible score. These can be used on their own for individual team score wagers. If we want to know how each team will fare against the other however, we'll need to pit these distributions head to head in a matrix.

NegBinomial Matrix Sheet

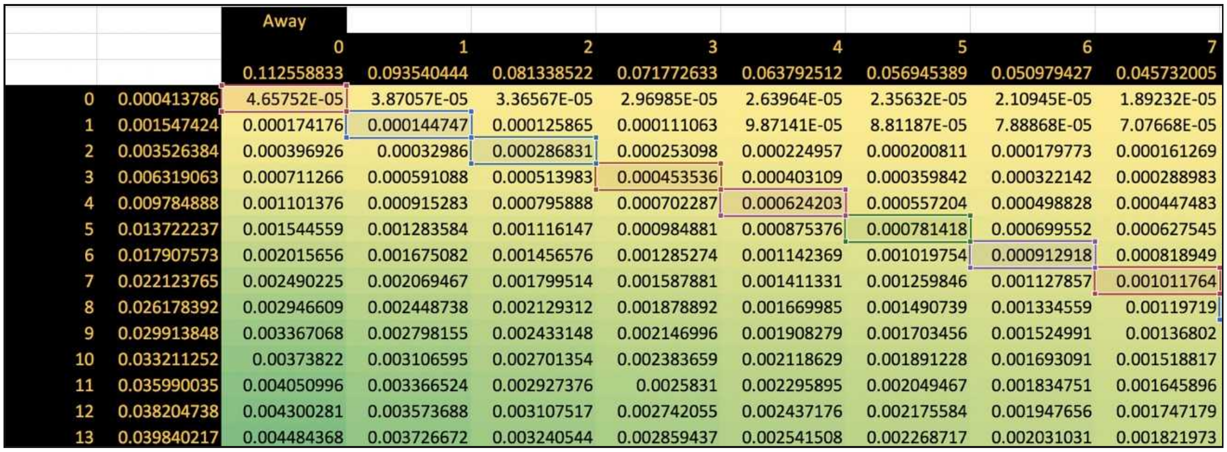
We use the competing matrix sheet to determine win probabilities for each team.[3] The first step is to bring all the probabilities from the individual team sheets into the matrix sheet and create a matrix with them. You'll notice that the home team probabilities are aligned vertically from top to bottom in column D and the away team probabilities are found in row 5 going left to right. We then multiply each of the corresponding probabilities together to produce a point probability between both teams for the exact score outcome in question.



A	B	C	D	E	F	G
				Away		
				0	1	
				0.112558833	0.093540444	0.0813
	Home	0	0.000413786	4.65752E-05	3.87057E-05	3.3656
		1	0.001547424	0.000174176	=D7*$F$5	0.0001
		2	0.003526384	0.000396926	0.00032986	0.0002
		3	0.006319063	0.000711266	0.000591088	0.0005
		4	0.009784888	0.001101376	0.000915283	0.0007
		5	0.013722237	0.001544559	0.001283584	0.0011
		6	0.017907573	0.002015656	0.001675082	0.0014
		7	0.022123765	0.002490225	0.002069467	0.0017

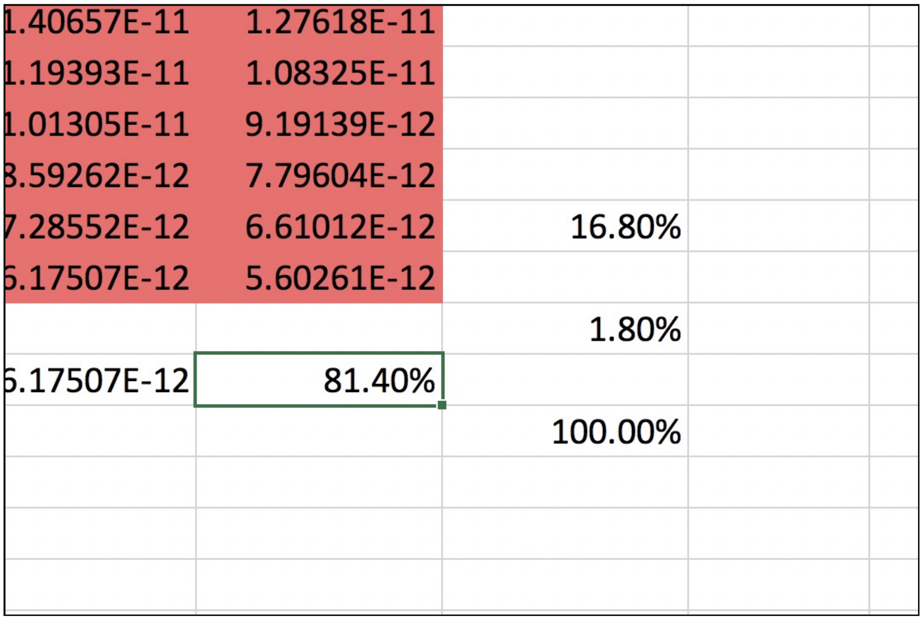
In the above picture, cell F7 represents the probability of the score being exactly 1-1.[4] We calculated this by multiplying each team's probability of scoring 1 point together. This is done for all cells in the matrix, and has already been completed.

Now we can calculate the probabilities of a win or a tie (although they are rare in the NFL). Ties are calculated by using the SUM command in Excel on all the diagonal cells where both teams are expected to score the same number of points. The output value, which represents the probability of a tie, is found in cell DA106.



		Away							
		0	1	2	3	4	5	6	7
		0.112558833	0.093540444	0.081338522	0.071772633	0.063792512	0.056945389	0.050979427	0.045732005
0	0.000413786	4.65752E-05	3.87057E-05	3.36567E-05	2.96985E-05	2.63964E-05	2.35632E-05	2.10945E-05	1.89232E-05
1	0.001547424	0.000174176	0.000144747	0.000125865	0.000111063	9.87141E-05	8.81187E-05	7.88868E-05	7.07668E-05
2	0.003526384	0.000396926	0.00032986	0.000286831	0.000253098	0.000224957	0.000200811	0.000179773	0.000161269
3	0.006319063	0.000711266	0.000591088	0.000513983	0.000453536	0.000403109	0.000359842	0.000322142	0.000288983
4	0.009784888	0.001101376	0.000915283	0.000795888	0.000702287	0.000624203	0.000557204	0.000498828	0.000447483
5	0.013722237	0.001544559	0.001283584	0.001116147	0.000984881	0.000875376	0.000781418	0.000699552	0.000627545
6	0.017907573	0.002015656	0.001675082	0.001456576	0.001285274	0.001142369	0.001019754	0.000912918	0.000818949
7	0.022123765	0.002490225	0.002069467	0.001799514	0.001587881	0.001411331	0.001259846	0.001127857	0.001011764
8	0.026178392	0.002946609	0.002448738	0.002129312	0.001878892	0.001669985	0.001490739	0.001334559	0.00119719
9	0.029913848	0.003367068	0.002798155	0.002433148	0.002146996	0.001908279	0.001703456	0.001524991	0.00136802
10	0.033211252	0.00373822	0.003106595	0.002701354	0.002383659	0.002118629	0.001891228	0.001693091	0.001518817
11	0.035990035	0.004050996	0.003366524	0.002927376	0.0025831	0.002295895	0.002049467	0.001834751	0.001645896
12	0.038204738	0.004300281	0.003573688	0.003107517	0.002742055	0.002437176	0.002175584	0.001947656	0.001747179
13	0.039840217	0.004484368	0.003726672	0.003240544	0.002859437	0.002541508	0.002268717	0.002031031	0.001821973

Values below this diagonal “tie line” represent outcomes where the home team wins. Values about the “tie line” represent outcomes where the away team wins. By summing these values we can calculate the win probabilities for both teams. The probability of an away team win can be found in cell DA104, while the home team win value can be found in cell CZ107.



1.40657E-11	1.27618E-11		
1.19393E-11	1.08325E-11		
1.01305E-11	9.19139E-12		
8.59262E-12	7.79604E-12		
7.28552E-12	6.61012E-12	16.80%	
6.17507E-12	5.60261E-12		
		1.80%	
6.17507E-12	81.40%		
		100.00%	

These probabilities are taken and automatically returned to the main ZSD model sheet in column BE of the game prediction function. From there, we can calculate fair prices and search for market value by comparing our price to the sportsbook line.

This is another way, in addition to a normal distribution based MOV model, that we can attempt to calculate win probabilities for NFL teams. Please remember that you must adjust variance, mean score, standard deviation and other relevant calculations in the main ZSD model sheet to the appropriate number of rows in your dataset.

[1] http://www.walksaber.blogspot.com/2012/07/on-run-distributions-pt-6.html

[2] http://www.walksaber.blogspot.com/2012/07/on-run-distributions-pt-6.html

[3] With some additional work summing the appropriate cells, point spreads and total game score probabilities can also be calculated.

[4] This is an admittedly ridiculous NFL score, but we must factor in every possibility to complete the matrix.

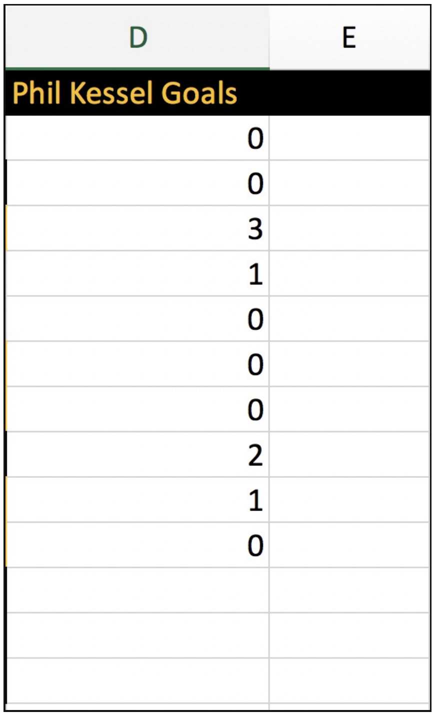
NHL MODEL IDEAS

> How many goals will Phil Kessel score tonight if we only have 10 games of data from the NHL season so far?

One recurring challenge for sports modelling is dealing with small sample sizes. This is particularly true early in the season where an average statistic based on only a handful of games doesn't provide us with a very good indication of the underlying distribution. In many cases with small samples, averages themselves are misleading because the mean and the median don't match - the distributions are being skewed by outlier scores. One way to deal with small sample sizes is to use a resampling method. Here, I'd like to show you how to use the bootstrap resampling method in Excel.

Bootstrapping is a method of random sampling with replacement that helps us to estimate the underlying distribution in our data even when we have a limited sample size. Let's apply it to our Phil Kessel problem and see how it works.

Open the "Bootstrap" Excel file. In column D, we input the data that we want to bootstrap. This this case we'll use the number of goals Phil Kessel has scored in his first 10 games of last season. We could also use points, assists, shots or any other statistic we might be interested in investigating further.

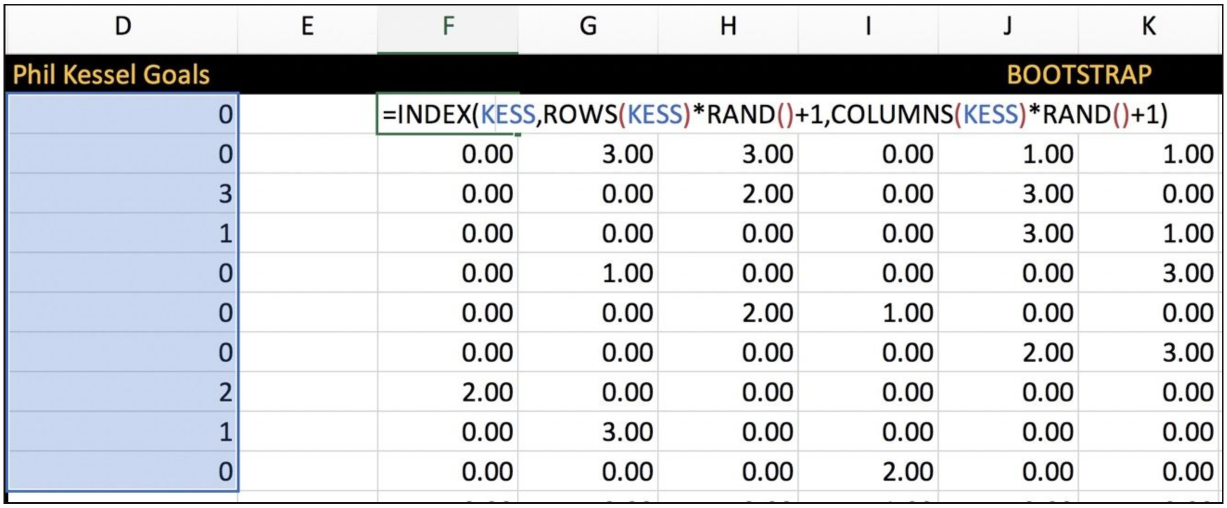


D	E
Phil Kessel Goals	
0	
0	
3	
1	
0	
0	
0	
2	
1	
0	

Next, we'll want to name this data as a range. Select the goal data and right click your mouse, then select "Name a Range". For this example, I've named the range "KESS". You could name it whatever you'd like. Then we click "OK".

Starting in column F is our bootstrap function. It extends from column F to column Y. The following formula is populated across all of the bootstrap columns and extended down to all relevant rows:

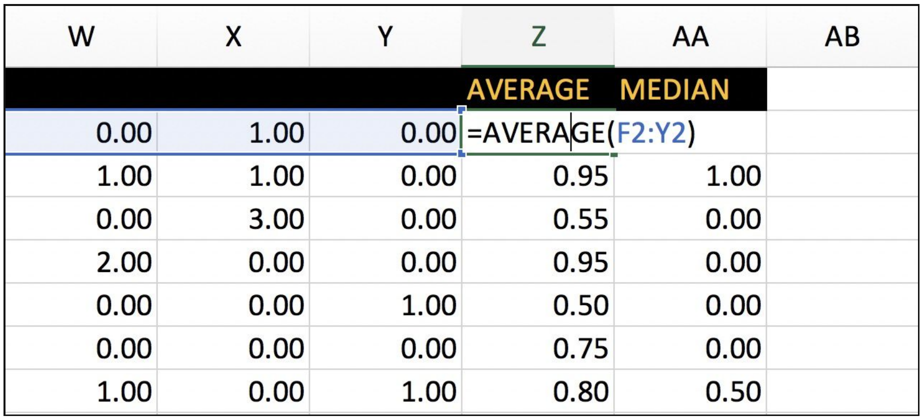
```
=INDEX
(KESS,ROWS(KESS)*RAND()+1,COLUMNS(KESS)*RAND()+1)
```



D	E	F	G	H	I	J	K
Phil Kessel Goals						BOOTSTRAP	
0		=INDEX(KESS,ROWS(KESS)*RAND()+1,COLUMNS(KESS)*RAND()+1)					
0		0.00	3.00	3.00	0.00	1.00	1.00
3		0.00	0.00	2.00	0.00	3.00	0.00
1		0.00	0.00	0.00	0.00	3.00	1.00
0		0.00	1.00	0.00	0.00	0.00	3.00
0		0.00	0.00	2.00	1.00	0.00	0.00
0		0.00	0.00	0.00	0.00	2.00	3.00
2		2.00	0.00	0.00	0.00	0.00	0.00
1		0.00	3.00	0.00	0.00	0.00	0.00
0		0.00	0.00	0.00	2.00	0.00	0.00

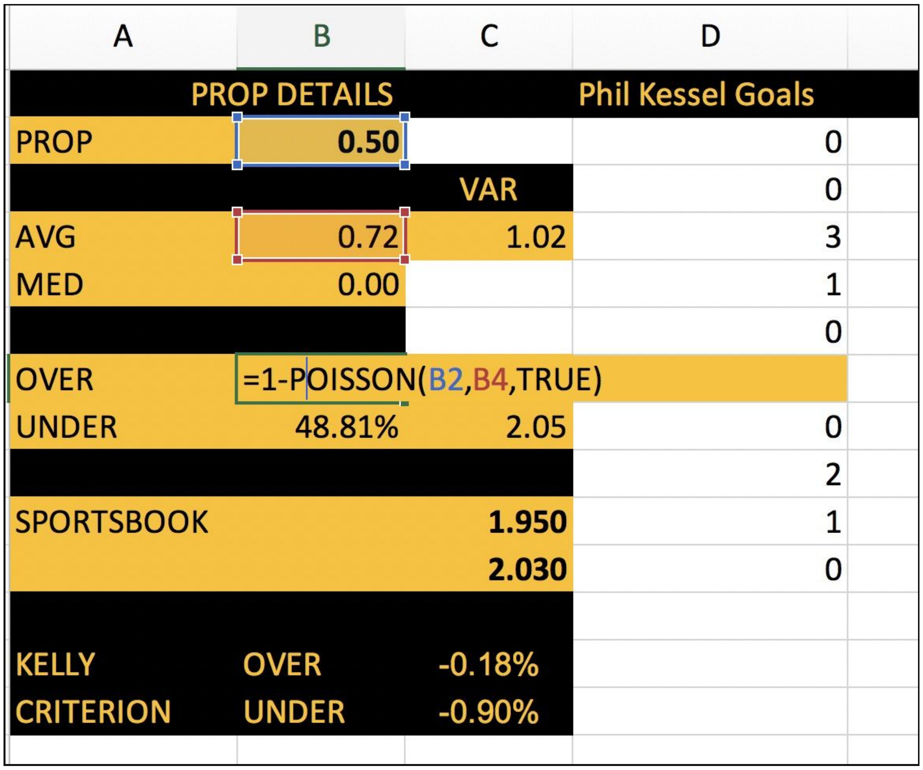
What this does is instruct Excel to randomly sample one of our data observations from the KESS range and then replace it for consideration in the next sample. This process occurs in all cells of our bootstrap, and since we have 20 columns with 499 rows we have now effectively turned our 10 game sample into a 9,980 sample using bootstrapping.

If you press the [delete] key in a blank cell you will see that the bootstrap runs again, with new values. In columns Z and AA we take the average and median of each row in the bootstrap. In cells B4 and B5 we take the average and median of all the calculated averages and medians from columns Z and AA. We also calculate the variance of the bootstrap in cell C4.



W	X	Y	Z	AA	AB
			AVERAGE	MEDIAN	
0.00	1.00	0.00	=AVERAGE(F2:Y2)		
1.00	1.00	0.00	0.95	1.00	
0.00	3.00	0.00	0.55	0.00	
2.00	0.00	0.00	0.95	0.00	
0.00	0.00	1.00	0.50	0.00	
0.00	0.00	0.00	0.75	0.00	
1.00	0.00	1.00	0.80	0.50	

The raw average for Phil Kessel is 0.7 goals per game from our 10 game sample, but our bootstrap returns a range of between 0.68 and 0.72. Ignoring opponent defense in this scenario, we can then take our expectation and use the POISSON command in Excel to calculate the probabilities of Kessel scoring more or less than 0.5 goals. You'll see that this has been completed in columns A and B for a hypothetical prop bet scenario.



A	B	C	D
	PROP DETAILS		Phil Kessel Goals
PROP	0.50		0
		VAR	0
AVG	0.72	1.02	3
MED	0.00		1
			0
OVER	=1-POISSON(B2,B4,TRUE)		
UNDER	48.81%	2.05	0
			2
SPORTSBOOK		1.950	1
		2.030	0
KELLY	OVER	-0.18%	
CRITERION	UNDER	-0.90%	

Pressing the [delete] key in a blank cell refreshes the bootstrap, and from doing this a few times we can see the expected value of a wager on either the over or the under vary a bit.

This technique can be used in many different situations but is particularly useful when faced with small sample sizes or distributions where the mean and the median diverge significantly.

MLB MODEL IDEAS

MLB is a sport with an embarrassment of riches for data analysis and it shows. With new statistics being developed every year and the ability to analyze the game pitch by pitch, major baseball betting markets are no joke. The ensemble approach from earlier chapters is not going to get the job done alone.[1] Smaller baseball markets and leagues are a better place to begin looking for value.

Some of the methods we've looked at already are quite applicable to baseball. For example, the negative binomial distribution combined with the ZSD model that we used for the NFL can model baseball scores. If you divide the expected runs by 9, you could also use the negative binomial distribution matrix to calculate a basic probability for runs scored in the first inning as well.

For this reason I won't be introducing any new models here for baseball, but would rather encourage you to take ideas from other parts of this book and experiment with them using MLB data. There should be lots for you to work with.

[1] Unless of course you combine multiple ratings for team offense, starting pitchers and bullpens. With the amount of work involved in this considered, incorporating some predictive player based statistics is probably the better way to go.

19

Live betting algorithms have weaknesses that can be taken advantage of. About a year ago I was placed on a time delay restriction after taking advantage of one such weakness to make 14 units in a single day on NCAAB live betting totals.

I don't have too many tricks up my sleeve in this department, but this was one of them. It's a dead edge for me, but still available to you until the weakness gets fixed (or you get limited like me). I'm going to share with you now what little I know about live-betting smaller market basketball totals, and the algorithm flaw that I think makes it possible.

Live Betting Weaknesses

There is value on live betting NCAAB unders in certain situations. This, I believe is possible for two reasons:

1. The general market favours the over, and as a result the sportsbook shades the live betting line towards the over; and

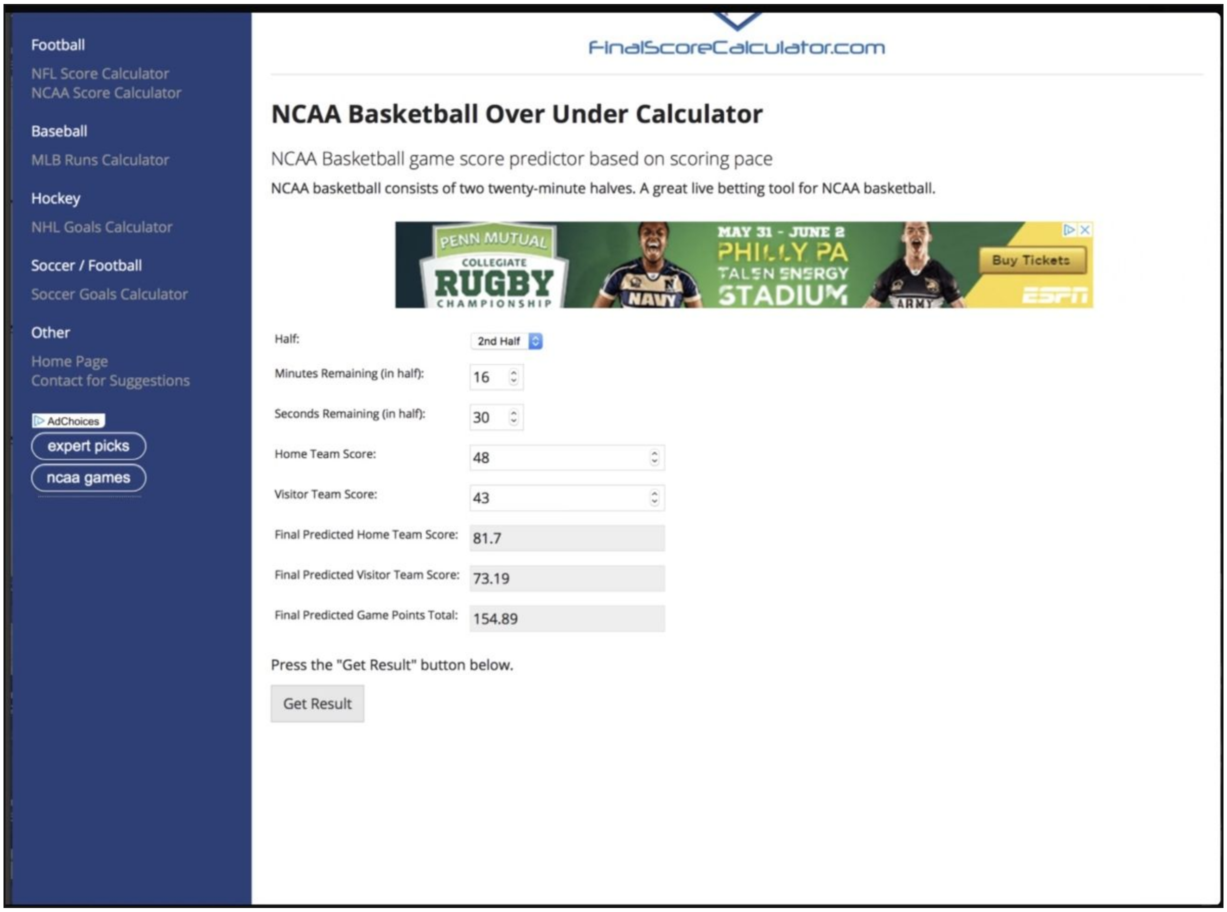
2. Live betting algorithms fail to account for diminishing team motivation in "can't win" situations.

In most games, the line shading isn't enough to provide value beyond the vig on the line, but when one team is leading by a wide margin late in the game it creates an opportunity for bettors.

Teams losing by 10 or more points late in the 2nd half tend to ease up, switch out their starting players and generally stop trying to win. This is not very well accounted for by the live betting algorithm that takes the current scoring rate and the time remaining and makes the assumption that scoring is likely to continue at a similar rate. This combined with the line shading towards the over creates a value opportunity. Bookmakers offering live betting on totals attempt to correct the error by removing the live betting line from the board when there are only a few minutes left in the game - when the discrepancy would become blatantly obvious. However, there is value to be had shortly before this moment.

The Method in a Nutshell

Using a basic calculator from FinalScoreCalculator.com, I would constantly update the projected total in one computer window while watching the live betting total in a separate window on my computer.



NCAA Basketball Over Under Calculator

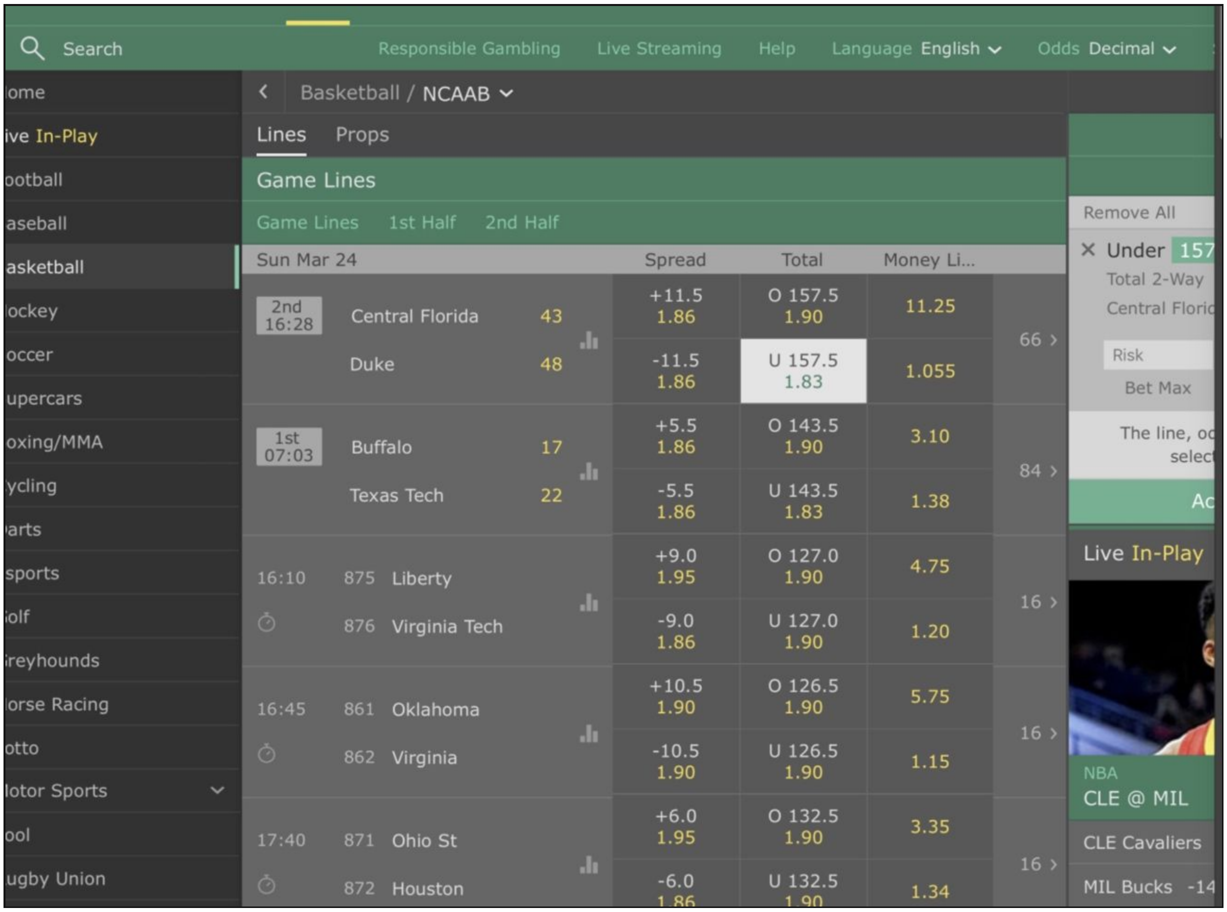
NCAA Basketball game score predictor based on scoring pace

NCAA basketball consists of two twenty-minute halves. A great live betting tool for NCAA basketball.

Half:	2nd Half
Minutes Remaining (in half):	16
Seconds Remaining (in half):	30
Home Team Score:	48
Visitor Team Score:	43
Final Predicted Home Team Score:	81.7
Final Predicted Visitor Team Score:	73.19
Final Predicted Game Points Total:	154.89

Press the "Get Result" button below.

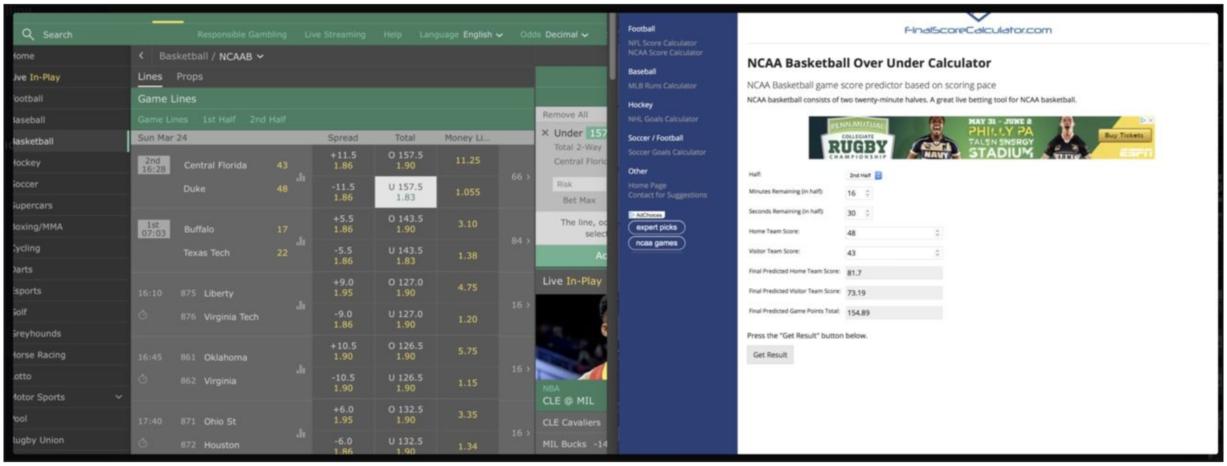
I would focus on games where one team was leading by more than 10 points. I would begin this process with around 8 minutes left in the 2nd half of a college basketball game. If you do this and observe a number of games with the clock winding down you'll notice projected edges in excess of 2.5 points in favour of the under. When you get a feel for just how late you can wait until betting before the line is closed, you can start firing on any value over 2.5 points. 3.5 was my preferred threshold because it required more than 2 free throws or a 3-pointer to break.



Sun Mar 24			Spread	Total	Money Li...	
2nd 16:28	Central Florida	43	+11.5 1.86	O 157.5 1.90	11.25	66 ›
	Duke	48	-11.5 1.86	U 157.5 1.83	1.055	
1st 07:03	Buffalo	17	+5.5 1.86	O 143.5 1.90	3.10	84 ›
	Texas Tech	22	-5.5 1.86	U 143.5 1.83	1.38	
16:10	875 Liberty		+9.0 1.95	O 127.0 1.90	4.75	16 ›
	876 Virginia Tech		-9.0 1.86	U 127.0 1.90	1.20	
16:45	861 Oklahoma		+10.5 1.90	O 126.5 1.90	5.75	16 ›
	862 Virginia		-10.5 1.90	U 126.5 1.90	1.15	
17:40	871 Ohio St		+6.0 1.95	O 132.5 1.90	3.35	16 ›
	872 Houston		-6.0 1.86	U 132.5 1.90	1.34	

Shortly thereabouts in the game, the scoring rate drops significantly and you have a positive expectancy.

By comparing the two numbers and constantly refreshing your projection on a split screen/ double window as show below, you'll be able to find edges on the under in games where there is a point differential of 10 or more late in the 2nd half of games:



Sun Mar 24				Spread	Total	Money Li...	
2nd 16:28		Central Florida	43	+11.5 1.86	O 157.5 1.90	11.25	66 ›
		Duke	48	-11.5 1.86	U 157.5 1.83	1.055	
1st 07:03		Buffalo	17	+5.5 1.86	O 143.5 1.90	3.10	84 ›
		Texas Tech	22	-5.5 1.86	U 143.5 1.83	1.38	
16:10	875	Liberty		+9.0 1.95	O 127.0 1.90	4.75	16 ›
	876	Virginia Tech		-9.0 1.86	U 127.0 1.90	1.20	
16:45	861	Oklahoma		+10.5 1.90	O 126.5 1.90	5.75	16 ›
	862	Virginia		-10.5 1.90	U 126.5 1.90	1.15	
17:40	871	Ohio St		+6.0 1.95	O 132.5 1.90	3.35	16 ›
	872	Houston		-6.0 1.86	U 132.5 1.90	1.34	

NCAA Basketball Over Under Calculator

NCAA Basketball game score predictor based on scoring pace

NCAA basketball consists of two twenty-minute halves. A great live betting tool for NCAA basketball.

Half: 2nd Half

Minutes Remaining (in half): 16

Seconds Remaining (in half): 30

Home Team Score: 48

Visitor Team Score: 43

Final Predicted Home Team Score: 81.7

Final Predicted Visitor Team Score: 73.19

Final Predicted Game Points Total: 154.89

Press the "Get Result" button below.

Get Result

In the example above, the line is 157.5, but our projections are 154.89. This example would've been a no play for me, as the MOV isn't large enough yet and there is still too much time on the clock. That's the general idea though. There's not much else to it but to do it. This method gave me one of the most profitable college ball days of my life before getting put on a time delay.

When I shared this on twitter briefly a few months ago, two people that I'm aware of gave it a try[1]. One did 4 units and the other did 6 units in a single day of college ball on a handful of games. I've since deleted it from my twitter feed, but have made it available to you here.

Enjoy it while you can.

---

[1] A tellingly low number. There isn't exactly a stampede of bettors trying to get sharper.

20

The English Premier League model idea I'd like to show you in this chapter centers around a ZSD model with competing poisson distributions. We'll attempt to zero-inflate our Poisson distribution in order to fit the soccer scoring distribution frequencies better. We'll also make a few additional modifications to account for calculating draw probabilities. We can then take the modified model outputs and use them to estimate exact scores, win probabilities and total scores in an EPL game.

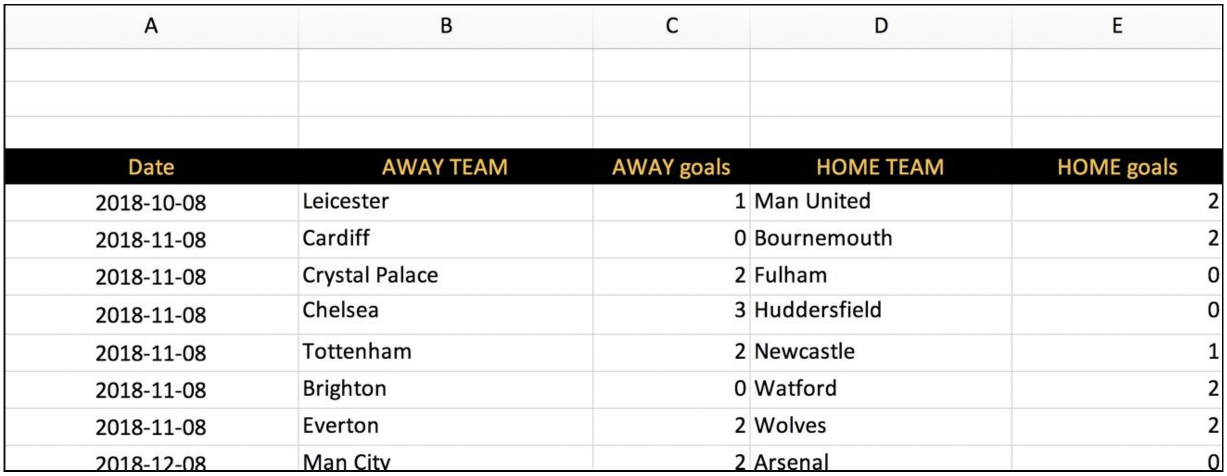
Why Use The Zero-Inflated Poisson?

The Poisson distribution models soccer scoring decently, but has a flaw that is widely understood. As mentioned in earlier chapters, one of the assumptions that must hold for the Poisson distribution to be appropriate is that the mean and variance of the underlying data must be the same. If we take a look at the mean and variance of many soccer leagues for both home and away scoring, we can observe that the mean is different than the variance.

We can often see that there is some under-dispersion occurring. This is what is meant when it is said that the basic Poisson distribution underestimates the probability of lower scoring soccer outcomes.[1] One of the solutions is to adjust the Poisson distribution towards lower scores ("Zero Inflate") so that it produces more accurate probability outputs that effectively account for the increased chance of draws and lower scores in the Premier League. While there are other, more elegant solutions[2], what I've prepared is a simple adjustment that can be easily applied to an Excel model.

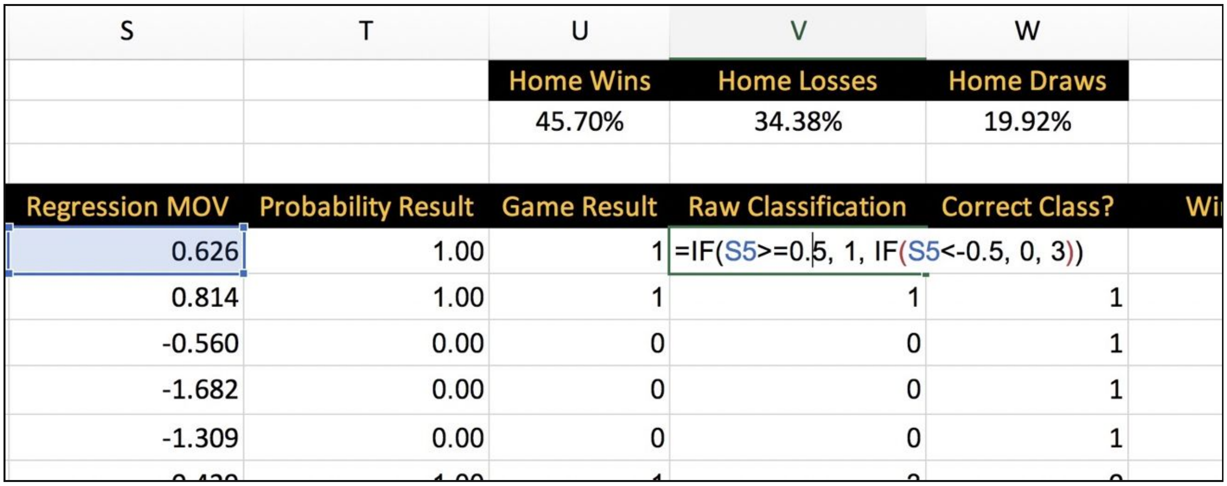
Main ZSD Model Sheet

When you open the "Competing Poisson EPL" sheet, you'll see it's very similar to the ZSD model we covered earlier. The data in columns A through E comes from football-data.co.uk for the English Premier League. I have not conditioned the data to account for draws in this example, but would recommend you do so if you plan on using this by using the method described earlier in this book.[3]



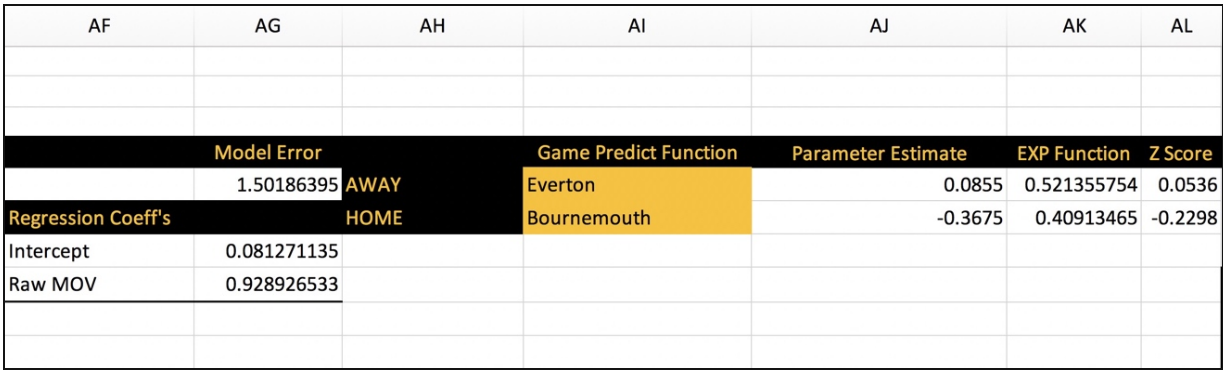
A	B	C	D	E
Date	AWAY TEAM	AWAY goals	HOME TEAM	HOME goals
2018-10-08	Leicester	1	Man United	2
2018-11-08	Cardiff	0	Bournemouth	2
2018-11-08	Crystal Palace	2	Fulham	0
2018-11-08	Chelsea	3	Huddersfield	0
2018-11-08	Tottenham	2	Newcastle	1
2018-11-08	Brighton	0	Watford	2
2018-11-08	Everton	2	Wolves	2
2018-12-08	Man City	2	Arsenal	0

If you look at column V you'll see we are now classifying games into 3 groups: 1 for a home win, 0 for a home loss, and 3 for a draw. We make these raw classifications by using the regression MOV in column S where anything over 0.5 MOV is considered a win, anything less than -0.5 is considered a loss and the middle remaining MOV in between is considered a draw.



S	T	U	V	W	
		Home Wins	Home Losses	Home Draws	
		45.70%	34.38%	19.92%	
Regression MOV	Probability Result	Game Result	Raw Classification	Correct Class?	Wi
0.626	1.00	1	=IF(S5>=0.5, 1, IF(S5<-0.5, 0, 3))		
0.814	1.00	1	1	1	
-0.560	0.00	0	0	1	
-1.682	0.00	0	0	1	
-1.309	0.00	0	0	1	

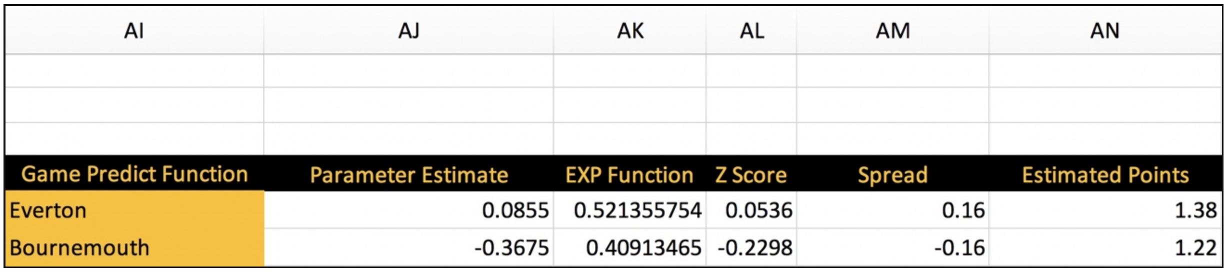
The model sheet functions the same as we've seen before with the ZSD model. Input and condition the data, make spreadsheet adjustments to account for the number of rows(games) in your dataset, optimize the model with solver and then run a linear regression using “Raw MOV” as the X variable and “Home MOV” as the Y variable. Copy and paste the coefficients and standard error into column AG. The game prediction function starts in column AI and can be used to forecast games using the normal distribution margin of victory method. So far, very little should be unfamiliar.



AF	AG	AH	AI	AJ	AK	AL
	Model Error		Game Predict Function	Parameter Estimate	EXP Function	Z Score
	1.50186395	AWAY	Everton	0.0855	0.521355754	0.0536
Regression Coeff's		HOME	Bournemouth	-0.3675	0.40913465	-0.2298
Intercept	0.081271135					
Raw MOV	0.928926533					

Poisson Dist Sheet

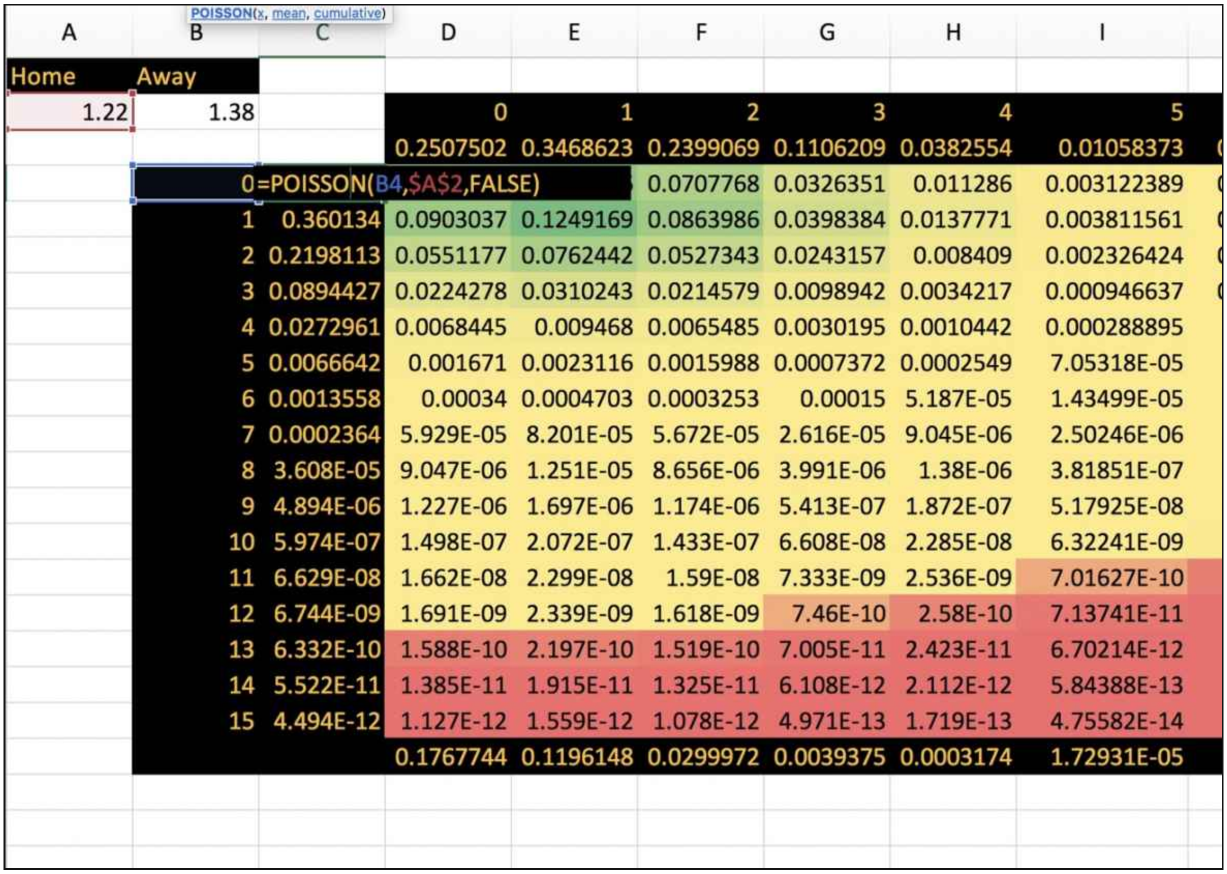
The next thing we’ll do is expand our model similarly to what we did with the NFL model by taking the goal estimates for each team and using a competing Poisson distribution matrix to map the probabilities. In this example we have Bournemouth playing at home and Everton playing away. After entering these teams into cells AI5 and AI6, the ZSD model sheet outputs the expected goals for each team into cells AN5 and AN6. These are the mean expectations that are used for our Poisson distribution matrix.



AI	AJ	AK	AL	AM	AN
Game Predict Function	Parameter Estimate	EXP Function	Z Score	Spread	Estimated Points
Everton	0.0855	0.521355754	0.0536	0.16	1.38
Bournemouth	-0.3675	0.40913465	-0.2298	-0.16	1.22

Let’s open up the “Poisson Dist” sheet. The first thing you’ll notice is that the expected goal values from the main sheet have been automatically imported into cells A2 and B2. Based on these numbers we use Excel’s POISSON command to calculate the probability of each team scoring an exact score of 0 all the way to 15 goals. The home team probabilities are aligned vertically in column C and the away team values span row 3. The following formula is used:

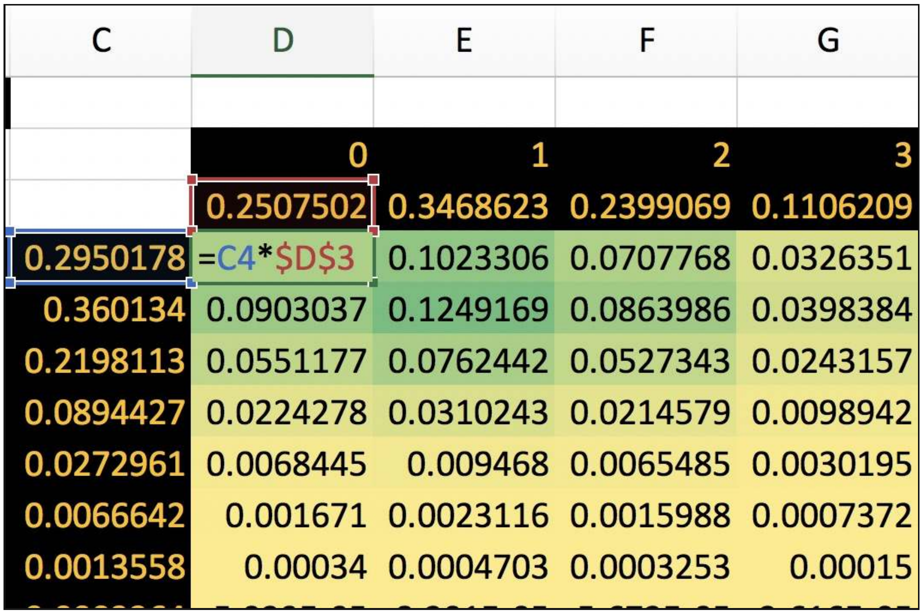
```
=POISSON(# of goals, Mean expectation, FALSE)
```



POISSON(x, mean, cumulative)

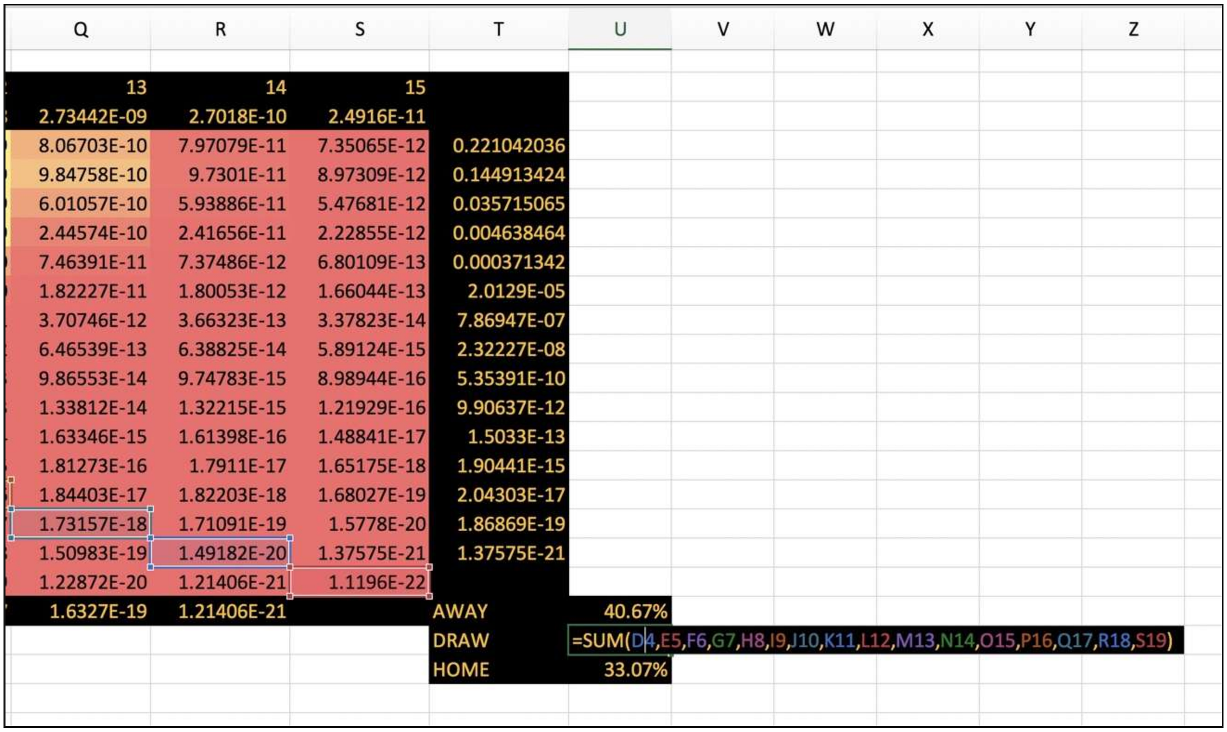
A	B	C	D	E	F	G	H	I
Home	Away							
1.22	1.38		0	1	2	3	4	5
			0.2507502	0.3468623	0.2399069	0.1106209	0.0382554	0.01058373
	0	=POISSON(B4,$A$2,FALSE)			0.0707768	0.0326351	0.011286	0.003122389
	1	0.360134	0.0903037	0.1249169	0.0863986	0.0398384	0.0137771	0.003811561
	2	0.2198113	0.0551177	0.0762442	0.0527343	0.0243157	0.008409	0.002326424
	3	0.0894427	0.0224278	0.0310243	0.0214579	0.0098942	0.0034217	0.000946637
	4	0.0272961	0.0068445	0.009468	0.0065485	0.0030195	0.0010442	0.000288895
	5	0.0066642	0.001671	0.0023116	0.0015988	0.0007372	0.0002549	7.05318E-05
	6	0.0013558	0.00034	0.0004703	0.0003253	0.00015	5.187E-05	1.43499E-05
	7	0.0002364	5.929E-05	8.201E-05	5.672E-05	2.616E-05	9.045E-06	2.50246E-06
	8	3.608E-05	9.047E-06	1.251E-05	8.656E-06	3.991E-06	1.38E-06	3.81851E-07
	9	4.894E-06	1.227E-06	1.697E-06	1.174E-06	5.413E-07	1.872E-07	5.17925E-08
	10	5.974E-07	1.498E-07	2.072E-07	1.433E-07	6.608E-08	2.285E-08	6.32241E-09
	11	6.629E-08	1.662E-08	2.299E-08	1.59E-08	7.333E-09	2.536E-09	7.01627E-10
	12	6.744E-09	1.691E-09	2.339E-09	1.618E-09	7.46E-10	2.58E-10	7.13741E-11
	13	6.332E-10	1.588E-10	2.197E-10	1.519E-10	7.005E-11	2.423E-11	6.70214E-12
	14	5.522E-11	1.385E-11	1.915E-11	1.325E-11	6.108E-12	2.112E-12	5.84388E-13
	15	4.494E-12	1.127E-12	1.559E-12	1.078E-12	4.971E-13	1.719E-13	4.75582E-14
			0.1767744	0.1196148	0.0299972	0.0039375	0.0003174	1.72931E-05

The “false” element tells Excel to calculate the individual probability for each team goal total rather than a cumulative probability. We do this for all possible goal outcomes from 0 to 15 for both teams. Then, like we did with the negative binomial distribution matrix, we multiply the specific outcomes for each team together to get the combined outcome probability for both teams in all scenarios.[4] I like to apply a “heat map” colour coding by using Excel’s conditional formatting to the joint probability cells so that it's easy to visually identify the most likely outcomes.



C	D	E	F	G
	0	1	2	3
	0.2507502	0.3468623	0.2399069	0.1106209
0.2950178	=C4*$D$3	0.1023306	0.0707768	0.0326351
0.360134	0.0903037	0.1249169	0.0863986	0.0398384
0.2198113	0.0551177	0.0762442	0.0527343	0.0243157
0.0894427	0.0224278	0.0310243	0.0214579	0.0098942
0.0272961	0.0068445	0.009468	0.0065485	0.0030195
0.0066642	0.001671	0.0023116	0.0015988	0.0007372
0.0013558	0.00034	0.0004703	0.0003253	0.00015

Like our NFL matrix example walkthrough, the diagonal line of cells from top left to bottom right represents outcomes where both teams score the same number of goals and the match results in a draw. These are summed in cell U21. Above this diagonal "draw line" are probabilities that represent outcomes where the away team wins. These are summed in cell U20. Below the "draw line" are outcomes representing a home team win and these are summed in cell U22. From these probabilities we can then calculate fair odds and carry on with the rest of our value identification process.



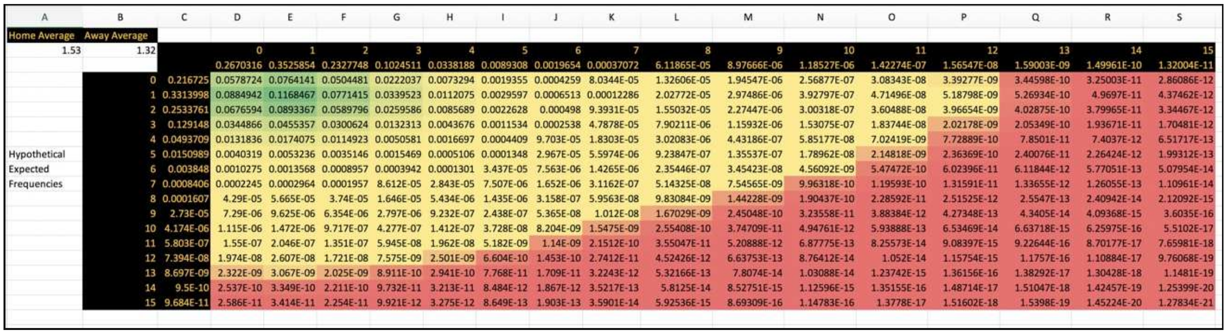
Q	R	S	T	U
13	14	15		
2.73442E-09	2.7018E-10	2.4916E-11		
8.06703E-10	7.97079E-11	7.35065E-12	0.221042036	
9.84758E-10	9.7301E-11	8.97309E-12	0.144913424	
6.01057E-10	5.93886E-11	5.47681E-12	0.035715065	
2.44574E-10	2.41656E-11	2.22855E-12	0.004638464	
7.46391E-11	7.37486E-12	6.80109E-13	0.000371342	
1.82227E-11	1.80053E-12	1.66044E-13	2.0129E-05	
3.70746E-12	3.66323E-13	3.37823E-14	7.86947E-07	
6.46539E-13	6.38825E-14	5.89124E-15	2.32227E-08	
9.86553E-14	9.74783E-15	8.98944E-16	5.35391E-10	
1.33812E-14	1.32215E-15	1.21929E-16	9.90637E-12	
1.63346E-15	1.61398E-16	1.48841E-17	1.5033E-13	
1.81273E-16	1.7911E-17	1.65175E-18	1.90441E-15	
1.84403E-17	1.82203E-18	1.68027E-19	2.04303E-17	
1.73157E-18	1.71091E-19	1.5778E-20	1.86869E-19	
1.50983E-19	1.49182E-20	1.37575E-21	1.37575E-21	
1.22872E-20	1.21406E-21	1.1196E-22		
1.6327E-19	1.21406E-21		AWAY	40.67%
			DRAW	=SUM(D4,E5,F6,G7,H8,I9,J10,K11,L12,M13,N14,O15,P16,Q17,R18,S19)
			HOME	33.07%

Constructing the Zero-Inflated Poisson Adjustment

As was mentioned before, the basic poisson just doesn’t quite get the job done. One way to sharpen the forecast is to adjust the Poisson distribution by comparing hypothetical probabilities to actual frequencies from the league. Here’s one way to do this:

Step 1: Theoretical Matrix Using Home/Away Averages

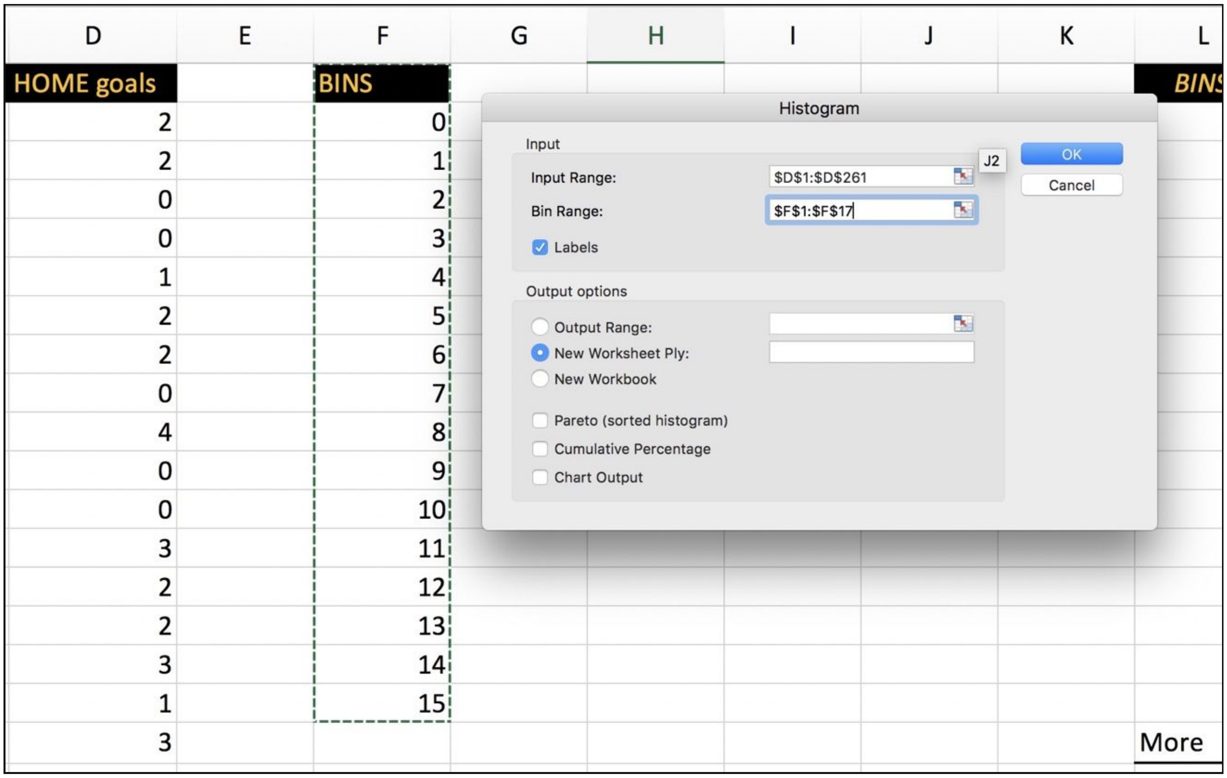
Open the “Zero Inflated Poisson Calcs” sheet. You’ll see 3 matrices. The first one is made by taking the league average expected goals for both home and away from H2 and J2 of the main ZSD sheet and constructing a poisson matrix to calculate the theoretical probabilities for each scoring outcome. Nothing fancy here - this is just our basic Poisson matrix.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
Home Average	Away Average																	
1.53	1.32		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
			0.2670316	0.3525854	0.2327748	0.1024511	0.0338188	0.0089308	0.0019654	0.00037072	6.11865E-05	8.97666E-06	1.18527E-06	1.42274E-07	1.56547E-08	1.59003E-09	1.49961E-10	1.32004E-11
	0	0.216725	0.0578724	0.0764141	0.0504481	0.0222037	0.0073294	0.0019355	0.0004259	8.0344E-05	1.32606E-05	1.94547E-06	2.56877E-07	3.08343E-08	3.39277E-09	3.44598E-10	3.25003E-11	2.86086E-12
	1	0.3313998	0.0884942	0.1168467	0.0771415	0.0339523	0.0112075	0.0029597	0.0006513	0.00012286	2.02772E-05	2.97486E-06	3.92797E-07	4.71496E-08	5.18798E-09	5.26934E-10	4.9697E-11	4.37462E-12
	2	0.2533761	0.0676594	0.0893367	0.0589796	0.0259586	0.0085689	0.0022628	0.000498	9.3931E-05	1.55032E-05	2.27447E-06	3.00318E-07	3.60488E-08	3.96654E-09	4.02875E-10	3.79965E-11	3.34467E-12
	3	0.129148	0.0344866	0.0455357	0.0300624	0.0132313	0.0043676	0.0011534	0.0002538	4.7878E-05	7.90211E-06	1.15932E-06	1.53075E-07	1.83744E-08	2.02178E-09	2.05349E-10	1.93671E-11	1.70481E-12
	4	0.0493709	0.0131836	0.0174075	0.0114923	0.0050581	0.0016697	0.0004409	9.703E-05	1.8303E-05	3.02083E-06	4.43186E-07	5.85177E-08	7.02419E-09	7.72889E-10	7.8501E-11	7.4037E-12	6.51717E-13
Hypothetical	5	0.0150989	0.0040319	0.0053236	0.0035146	0.0015469	0.0005106	0.0001348	2.967E-05	5.5974E-06	9.23847E-07	1.35537E-07	1.78962E-08	2.14818E-09	2.36369E-10	2.40076E-11	2.26424E-12	1.99312E-13
Expected	6	0.003848	0.0010275	0.0013568	0.0008957	0.0003942	0.0001301	3.437E-05	7.563E-06	1.4265E-06	2.35446E-07	3.45423E-08	4.56092E-09	5.47472E-10	6.02396E-11	6.11844E-12	5.77051E-13	5.07954E-14
Frequencies	7	0.0008406	0.0002245	0.0002964	0.0001957	8.612E-05	2.843E-05	7.507E-06	1.652E-06	3.1162E-07	5.14325E-08	7.54565E-09	9.96318E-10	1.19593E-10	1.31591E-11	1.33655E-12	1.26055E-13	1.10961E-14
	8	0.0001607	4.29E-05	5.665E-05	3.74E-05	1.646E-05	5.434E-06	1.435E-06	3.158E-07	5.9563E-08	9.83084E-09	1.44228E-09	1.90437E-10	2.28592E-11	2.51525E-12	2.5547E-13	2.40942E-14	2.12092E-15
	9	2.73E-05	7.29E-06	9.625E-06	6.354E-06	2.797E-06	9.232E-07	2.438E-07	5.365E-08	1.012E-08	1.67029E-09	2.45048E-10	3.23558E-11	3.88384E-12	4.27348E-13	4.3405E-14	4.09368E-15	3.6035E-16
	10	4.174E-06	1.115E-06	1.472E-06	9.717E-07	4.277E-07	1.412E-07	3.728E-08	8.204E-09	1.5475E-09	2.55408E-10	3.74709E-11	4.94761E-12	5.93888E-13	6.53469E-14	6.63718E-15	6.25975E-16	5.5102E-17
	11	5.803E-07	1.55E-07	2.046E-07	1.351E-07	5.945E-08	1.962E-08	5.182E-09	1.14E-09	2.1512E-10	3.55047E-11	5.20888E-12	6.87775E-13	8.25573E-14	9.08397E-15	9.22644E-16	8.70177E-17	7.65981E-18
	12	7.394E-08	1.974E-08	2.607E-08	1.721E-08	7.575E-09	2.501E-09	6.604E-10	1.453E-10	2.7412E-11	4.52426E-12	6.63753E-13	8.76412E-14	1.052E-14	1.15754E-15	1.1757E-16	1.10884E-17	9.76068E-19
	13	8.697E-09	2.322E-09	3.067E-09	2.025E-09	8.911E-10	2.941E-10	7.768E-11	1.709E-11	3.2243E-12	5.32166E-13	7.8074E-14	1.03088E-14	1.23742E-15	1.36156E-16	1.38292E-17	1.30428E-18	1.1481E-19
	14	9.5E-10	2.537E-10	3.349E-10	2.211E-10	9.732E-11	3.213E-11	8.484E-12	1.867E-12	3.5217E-13	5.8125E-14	8.52751E-15	1.12596E-15	1.35155E-16	1.48714E-17	1.51047E-18	1.42457E-19	1.25399E-20
	15	9.684E-11	2.586E-11	3.414E-11	2.254E-11	9.921E-12	3.275E-12	8.649E-13	1.903E-13	3.5901E-14	5.92536E-15	8.69309E-16	1.14783E-16	1.3778E-17	1.51602E-18	1.5398E-19	1.45224E-20	1.27834E-21

Step 2: Actual Observed League Goal Frequencies

Open the “Goal Frequency” sheet. You’ll see the recorded goals for home and away from our dataset, plus a “Bins” column that we’ll use to sort the frequencies of each score outcome. Click on “Data” and then “Data Analysis” and then select “Histogram”. For the first histogram select all the goal data in column D as the input range, then select all the numbers in column F as the bin range. When you hit “OK”, you’ll get an output with the number of times the home team scored exactly 1, 2, 3, etc. goals.



D	E	F	L
HOME goals		BINS	BINS
2		0	
2		1	
0		2	
0		3	
1		4	
2		5	
2		6	
0		7	
4		8	
0		9	
0		10	
3		11	
2		12	
2		13	
3		14	
1		15	
3			More

Histogram

Input

Input Range: $D$1:$D$261

Bin Range: $F$1:$F$17

☑ Labels

Output options

○ Output Range:

● New Worksheet Ply:

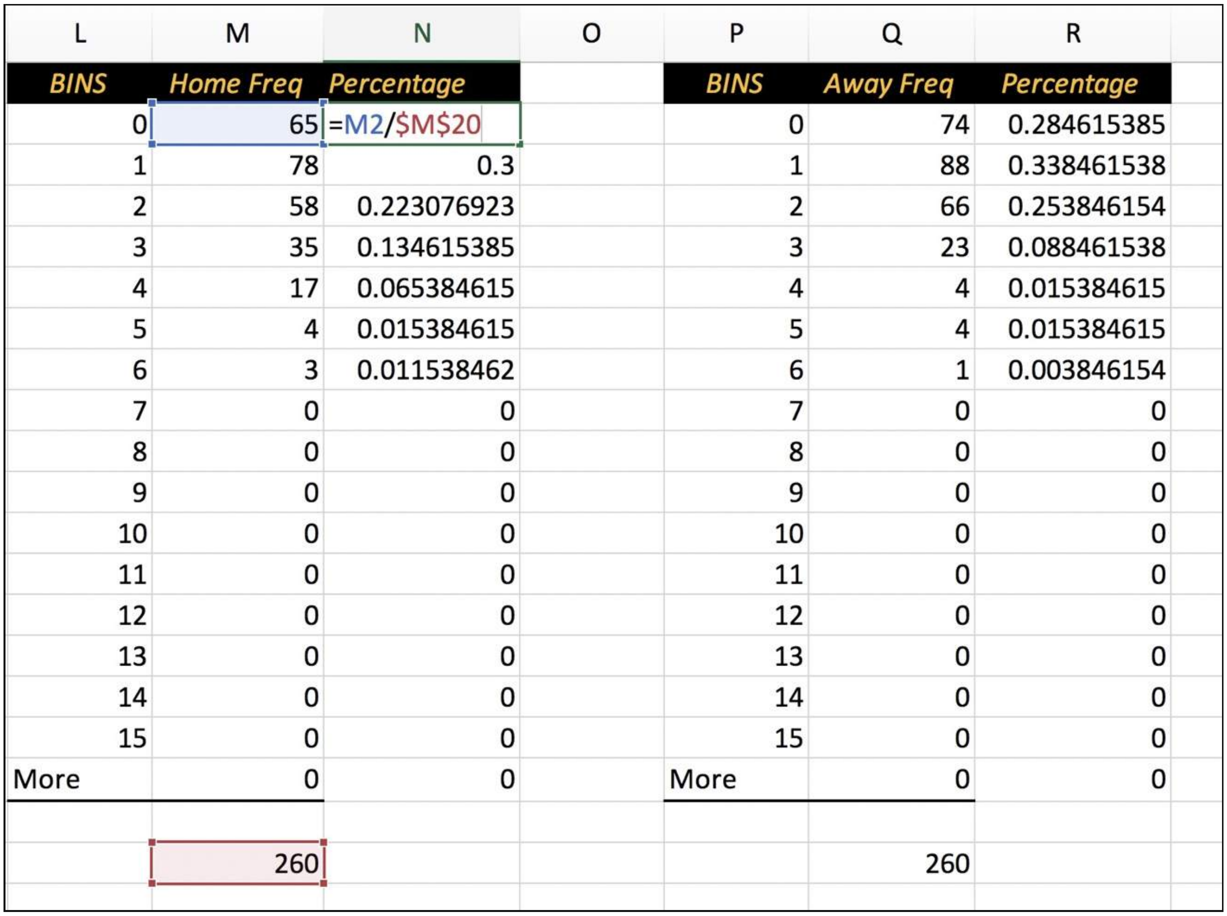
○ New Workbook

☐ Pareto (sorted histogram)

☐ Cumulative Percentage

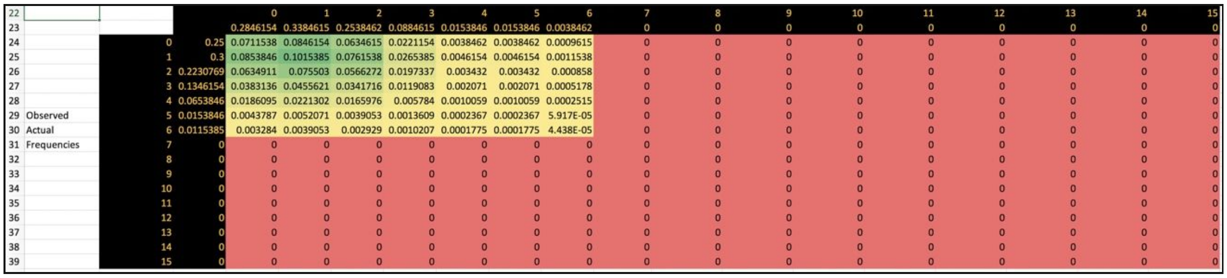
☐ Chart Output

You can find an example of the data output in columns L and M. In cell M20, we sum the goals scored to get a total number. Then, in column N we divide each goal outcome frequency by the total goals scored in cell M20 to produce the frequency as a probability. You can see these probabilities in column N. Repeat this for both the home goals and away goals.



L	M	N	O	P	Q	R
BINS	Home Freq	Percentage		BINS	Away Freq	Percentage
0	65	=M2/$M$20		0	74	0.284615385
1	78	0.3		1	88	0.338461538
2	58	0.223076923		2	66	0.253846154
3	35	0.134615385		3	23	0.088461538
4	17	0.065384615		4	4	0.015384615
5	4	0.015384615		5	4	0.015384615
6	3	0.011538462		6	1	0.003846154
7	0	0		7	0	0
8	0	0		8	0	0
9	0	0		9	0	0
10	0	0		10	0	0
11	0	0		11	0	0
12	0	0		12	0	0
13	0	0		13	0	0
14	0	0		14	0	0
15	0	0		15	0	0
More	0	0		More	0	0
	260				260	

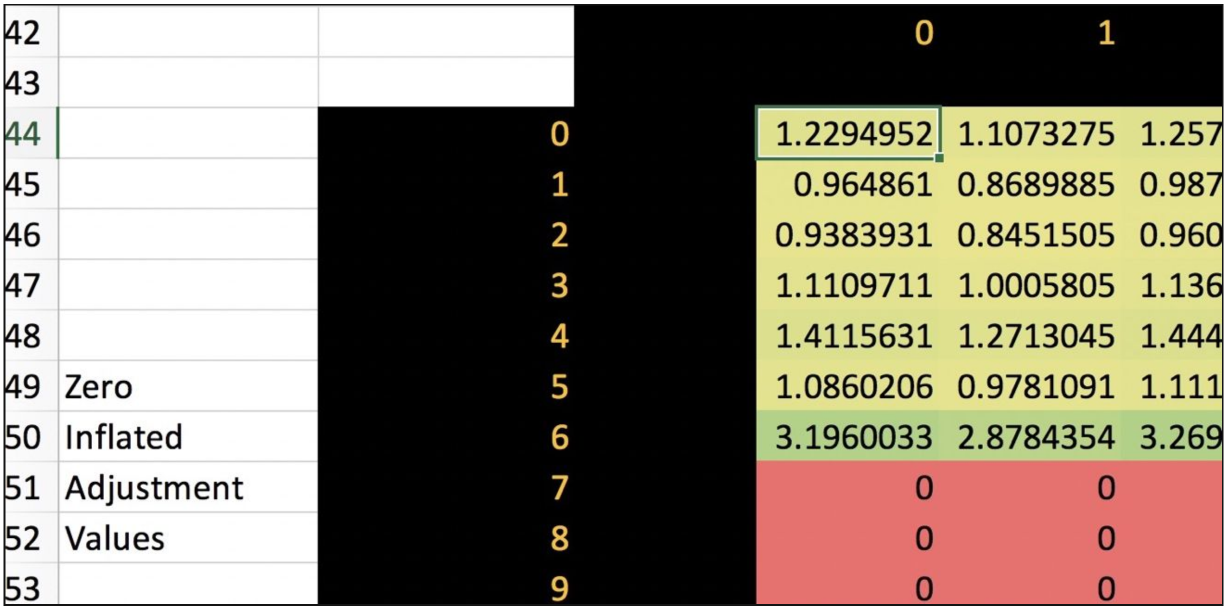
Now let's head back to the "Zero Inflated Poisson Calcs" sheet. Look at the second matrix starting in row 22. This matrix has been created by using the probabilities we just calculated in the "Goal Frequency" sheet, and represents what actually occurred in the EPL dataset that we have. You'll notice that there is a reasonable amount of difference between these observed frequencies and the theoretical frequencies from the first matrix.



22				0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
23				0.2846154	0.3384615	0.2538462	0.0884615	0.0153846	0.0153846	0.0038462	0	0	0	0	0	0	0	0	0
24		0	0.25	0.0711538	0.0846154	0.0634615	0.0221154	0.0038462	0.0038462	0.0009615	0	0	0	0	0	0	0	0	0
25		1	0.3	0.0853846	0.1015385	0.0761538	0.0265385	0.0046154	0.0046154	0.0011538	0	0	0	0	0	0	0	0	0
26		2	0.2230769	0.0634911	0.075503	0.0566272	0.0197337	0.003432	0.003432	0.000858	0	0	0	0	0	0	0	0	0
27		3	0.1346154	0.0383136	0.0455621	0.0341716	0.0119083	0.002071	0.002071	0.0005178	0	0	0	0	0	0	0	0	0
28		4	0.0653846	0.0186095	0.0221302	0.0165976	0.005784	0.0010059	0.0010059	0.0002515	0	0	0	0	0	0	0	0	0
29	Observed	5	0.0153846	0.0043787	0.0052071	0.0039053	0.0013609	0.0002367	0.0002367	5.917E-05	0	0	0	0	0	0	0	0	0
30	Actual	6	0.0115385	0.003284	0.0039053	0.002929	0.0010207	0.0001775	0.0001775	4.438E-05	0	0	0	0	0	0	0	0	0
31	Frequencies	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32		8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33		9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34		10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35		11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36		12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37		13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38		14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39		15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Step 3: Create Adjustments

Since we now have the observed frequencies and the theoretical frequencies we can calculate how much we might want to adjust the basic poisson probability outputs based on our data. You'll see a third matrix in the "Zero Inflated Poisson Calcs" sheet starting in row 42. These are our adjustments. The adjustments are calculated by dividing our observed frequency values from matrix 2 by our theoretical frequency values in matrix 1. For example, our observed frequency for a score of 0-0 is 0.71154 while our theoretical frequency is 0.57872. Dividing these in our third matrix gives us an adjustment value of 1.229495 or 123% of the theoretical frequency value.



42			0	1	
43					
44		0	1.2294952	1.1073275	1.257
45		1	0.964861	0.8689885	0.987
46		2	0.9383931	0.8451505	0.960
47		3	1.1109711	1.0005805	1.136
48		4	1.4115631	1.2713045	1.444
49	Zero	5	1.0860206	0.9781091	1.111
50	Inflated	6	3.1960033	2.8784354	3.269
51	Adjustment	7	0	0	
52	Values	8	0	0	
53		9	0	0	

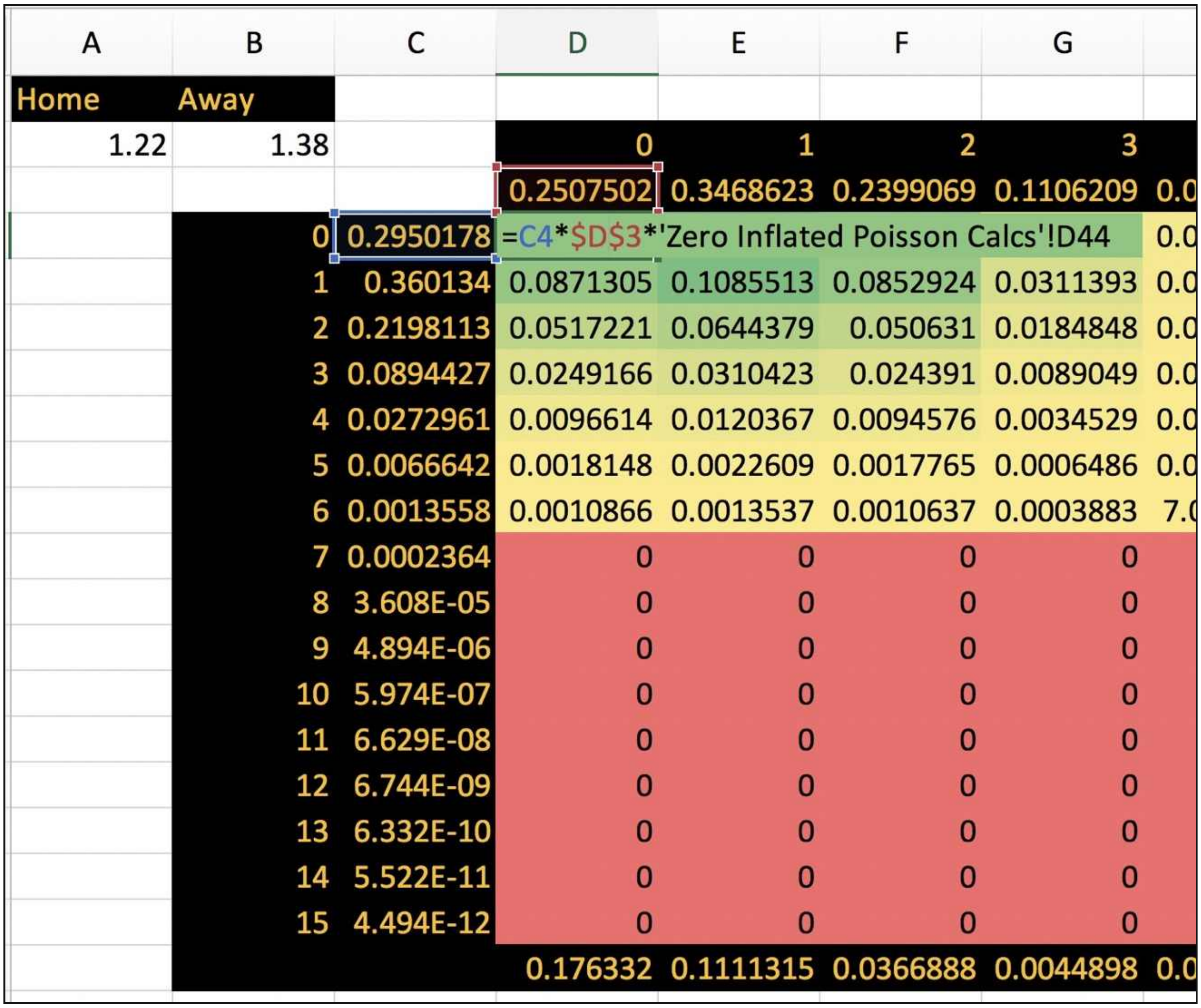
This is done for all relevant cells. Also note that for every league this adjustment could be different - as the goal frequencies can vary a bit. Once this adjustment matrix has been completed we are ready to construct the final Zero Inflated Poisson Matrix.

Step 4: Construct Adjusted Matrix

Open the “Zero Inflated Poisson Dist” sheet. This will be our final adjusted Poisson matrix which takes what we’ve done up to now and uses it to improve the forecast. In cells A2 and B2 you’ll find the imported expected goal values for Bournemouth and Everton (these come from the main ZSD sheet). The matrix works the same as the basic competing Poisson distribution matrix, except that as a final step each cell is multiplied by the relevant adjustment value we calculated earlier. So while before we simply multiplied each team’s probability together, we now have:

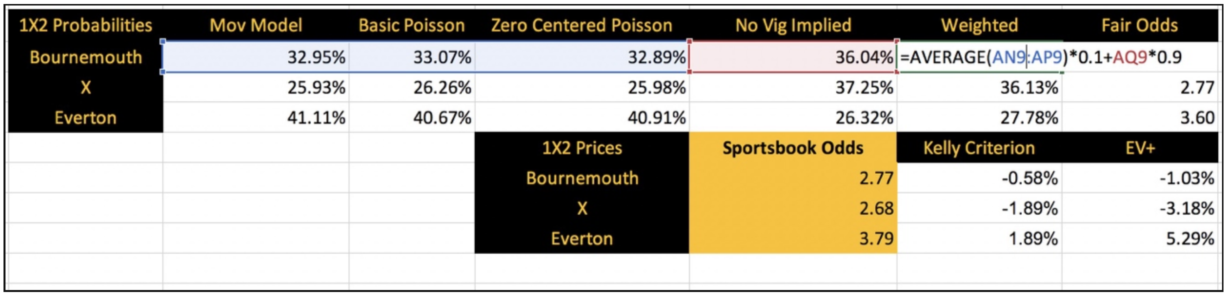
> Team A % * Team B % * Adjustment Value

This is done for all relevant cells, and the probabilities are summed to calculate chances of a draw, home win or away win as before.



A	B	C	D	E	F	G	
Home	Away						
1.22	1.38		0	1	2	3	
			0.2507502	0.3468623	0.2399069	0.1106209	0.0
	0	0.2950178	=C4*$D$3*'Zero Inflated Poisson Calcs'!D44				0.0
	1	0.360134	0.0871305	0.1085513	0.0852924	0.0311393	0.0
	2	0.2198113	0.0517221	0.0644379	0.050631	0.0184848	0.0
	3	0.0894427	0.0249166	0.0310423	0.024391	0.0089049	0.0
	4	0.0272961	0.0096614	0.0120367	0.0094576	0.0034529	0.0
	5	0.0066642	0.0018148	0.0022609	0.0017765	0.0006486	0.0
	6	0.0013558	0.0010866	0.0013537	0.0010637	0.0003883	7.0
	7	0.0002364	0	0	0	0	
	8	3.608E-05	0	0	0	0	
	9	4.894E-06	0	0	0	0	
	10	5.974E-07	0	0	0	0	
	11	6.629E-08	0	0	0	0	
	12	6.744E-09	0	0	0	0	
	13	6.332E-10	0	0	0	0	
	14	5.522E-11	0	0	0	0	
	15	4.494E-12	0	0	0	0	
			0.176332	0.1111315	0.0366888	0.0044898	0.0

You'll notice the range of possible outcomes has collapsed (or "inflated") around lower goal score totals. We've effectively adjusted the basic Poisson distribution to better suit our purposes. These new win probabilities are then imported back to our main ZSD sheet where we can calculate fair odds and look for value. You'll find all three methods (MOV, Basic Poisson and Zero Inflated Poisson) available in the main ZSD sheet starting in cell AM8. You can use them individually, or with various weightings. My preferred weighting for the EPL is to take the average of the 3 calculated probabilities at 10% and the no-vig Pinnacle probabilities at 90%. I encourage you however to experiment with different configurations.



1X2 Probabilities	Mov Model	Basic Poisson	Zero Centered Poisson	No Vig Implied	Weighted	Fair Odds
Bournemouth	32.95%	33.07%	32.89%	36.04%	=AVERAGE(AN9:AP9)*0.1+AQ9*0.9	
X	25.93%	26.26%	25.98%	37.25%	36.13%	2.77
Everton	41.11%	40.67%	40.91%	26.32%	27.78%	3.60
			1X2 Prices	Sportsbook Odds	Kelly Criterion	EV+
			Bournemouth	2.77	-0.58%	-1.03%
			X	2.68	-1.89%	-3.18%
			Everton	3.79	1.89%	5.29%

[1] You've likely heard some version of this sentiment iterated elsewhere. Simply stated, the basic Poisson isn't quite sharp enough to get the job done.

[2] http://www.opisthokonta.net

[3] The method is described in the chapter on the Bradley Terry Model. As a reminder, we replace draws with two new results: a one goal win for the home team and a one goal win for the away team. The net effect cancels out and allows the optimized ratings to model draws more effectively.

[4] Multiplying the probabilities together assumes that each team's scoring is independent of each other, which is a questionable assumption. There are ways to correct for this but let's keep it simple for now.

MMA MODEL IDEAS

I’ve never personally bet on MMA, but I did want to share an idea for converting the generalized sport models we’ve already looked at to suit this purpose.

In order to make hay with our generalized models we first need a way to convert MMA result outcomes into a margin of victory (or a statement on win strength, if you like). Inside the cage, there are many ways to lose. A fighter can get KO’d, TKO’d, submitted, lose by unanimous decision or lose by split decision. Apart from the decisions, these outcomes can occur in any round. What if we assigned a numeric margin of victory to these outcomes?

That just might work.

TKO’s and submissions are fairly comparable as a statement of strength and speak greatly to a fighter’s individual combat style, so it might be a good idea to consider them similarly on an MOV scale. KO’s are rather definitive and should probably be rated at maximum strength. Unanimous decisions are slightly more convincing than split decisions. With that in mind let’s put an initial MOV scale together:

Proposed Strength MOV Rating Scale

- 1. KO
- 2. TKO / Submission
- 3. Unanimous Decision
- 4. Split Decision

Let’s call a split decision a victory of 1 point for a given fighter. Unanimous decisions would be a 2 point MOV. TKO’s and submissions would then be 3, making KO victories a 4 point MOV. Using these point equivalents we could then try applying some of the models we’ve worked with already to any weight division in a major MMA circuit.

We might be able to further distinguish between early round outcomes and late round outcomes. Some questions come to mind:

> Should a round 1 KO be worth more than a round 3 KO? Does it indicate more cage dominance, or is the stamina required by a late round KO victory more indicative of latent ability?

I don’t have the answers to these questions, but you might have some ideas. As I don’t have experience with this market this is really just a brainstorming session. However I’d thought I’d share with you one idea for applying the models we’ve worked with already to a new market. Is there a better way to model MMA? Probably. Give this a try anyways. You never know what you might find.

RECOMMENDED RESOURCES

Recommended Twitter Accounts

@12Xpert

@MatterOfStats

@Quantum_Sport

@EdMillerPoker

@optibrebs

@opisthokonta

@truepokerjoe

@jakevdp

@thepowerrank

@Toirtap

@Tangotiger

@drob

Recommended Books

“Sharper” by True Poker Joe

“Winning Sports Betting” by Masaru Kanemoto

“The Logic of Sports Betting” by Ed Miller

“Fixed Odds Sports Betting” by Joseph Buchdahl

“Squares, Sharps, Suckers & Sharks” by Joseph Buchdahl

“Precision” by CX Wong

“Dr. Z’s NFL Guidebook” by William Ziemba

“Introduction to Empirical Bayes” by David Robinson

“Bayes Theorem: A Visual Introduction” by Dan Morris

“Statistical Rethinking” by Richard McElreath

“Probability for the Enthusiastic Beginner” by David Morin

Recommended Data Resources

hockey-reference.com

pro-football-reference.com

baseball-reference.com

basketball-reference.com

fbref.com

http://www.aussportsbetting.com/data/

football-data.co.uk

https://www.armchairanalysis.com/data.php

ABOUT THE AUTHOR

Andrew Mack is a JD/MSc Data Science student and sports bettor. As a former Journeyman Electrician, he has a passion for using applied math to solve problems.

He has been modelling sports since 2012 and has experience with statistical & machine learning models in Excel, R and Python for most major sports including the NBA, NFL, NHL, MLB, AFL and the English Premier League. Andrew holds an undergraduate degree in social sciences from the University of Victoria and lives in Calgary, Alberta, Canada.

He can be reached on twitter **@Gingfacekillah**.