
DAWSON REED
HANDBOOK
The Greyhound Betting Formula
A Quantitative Approach to Racing Data

HANDBOOK

The Greyhound Betting Formula

A Quantitative Approach to Racing Data

DAWSON REED

© 2026 Dawson Reed. All rights reserved.

Contents

1 Why Quantitative Betting Works

2 The Greyhound Data Landscape

3 Setting Up Your Data Stack

4 Pulling Data Through APIs

5 Cleaning and Validating Data

6 Race Conditions and Track Variables

7 Dog Form and Performance Variables

8 Working With Market Odds

9 Probability for Greyhound Bettors

10 Building Your First Rating System

11 Statistical Models You Can Use

12 Backtesting Your Betting Ideas

13 Measuring Real-World Performance

14 Bankroll and Staking Plans

15 Iterating Without Overfitting

16 Your Quantitative Betting Plan

CHAPTER 1

Why Quantitative Betting Works



Every greyhound race is a small auction. Eight runners, a starting trap for each, a starting price, and a few seconds of careful work deciding whether to play or pass. Most bettors spend their energy trying to find the right dog. The more profitable question, by a country mile, is whether the dog in front of you is being offered at the right price. That single shift — from picking dogs to pricing outcomes — is what this book is about, and it is worth more than any list of tips you will ever be sent.

From Picking Dogs to Pricing Outcomes

If you have spent any time around greyhound racing, you have probably heard the same kind of advice over and over. Back the dog in form. Follow the trap draw. Trust the tipster who says "this one can't be beaten". Sometimes that advice even wins. The trouble is, winning one race tells you almost nothing about whether you are a good bettor. It tells you whether one race went your way, and races — like dog races — are full of small accidents of timing, luck and break.

A tipping service hands you selections. It might be a free column on a Saturday paper, a paid Telegram channel, a Discord group, or a mate who has been watching dogs for thirty years. The promise is simple: tell me who is going to win. The problem with that promise is that it skips the most important question in betting — is the selection worth the price?

Imagine a greyhound races off at 1.50 (1/2). Even if your tipster picks the winner 70% of the time, you are losing money in the long run, because every winner pays back less than a fair price. The market has already priced in the fact that this is the most likely winner.

To make a profit, you have to either find dogs whose true chance is higher than the market thinks, or — just as importantly — pass on dogs whose true chance is lower than the market thinks.

This is the difference between tipping and pricing. A tipster hands you a name. A pricing mindset hands you a number — your own estimate of the chance this dog wins — and then asks: is this number higher than what the bookmaker is implying? If yes, you have a bet. If no, you walk away, even if the dog is short enough that you "have to be in it".

> **GOOD TO KNOW**
> The odds on a betting exchange or in a bookmaker's market are not random. They are the crowd's best guess at each dog's chance, adjusted for the bookmaker's margin. Your job is not to second-guess the crowd emotionally — it is to estimate the chance more accurately than the crowd does.

The same logic applies even when you don't have a tip. Many greyhound bettors place a wager because a dog "feels" right. They back it at 4.50 (7/2) without ever asking themselves whether the true chance of winning is closer to 1-in-5 (which would price it at 5.00 or 4/1) or 1-in-3 (which would price it at 3.33 or 5/2). If your gut says 4.50 is value but it isn't, you have just handed the bookmaker a small donation. Do that a hundred times and you have handed them quite a large one.

Quantitative betting is the discipline of making that judgement with evidence rather than instinct. You collect data, you build a small estimate of each runner's chance, you compare it to the market, and you bet only when the two disagree in your favour. It is a quieter way to bet. It produces fewer "winners" in the gossipy sense. It produces more winners in the bank-statement sense.

The Maths of a Small Edge

One of the most off-putting facts for new bettors is the size of the structural disadvantage they are up against. Take a typical greyhound race priced up by a bookmaker. Six runners, each given a "fair" chance of about 1-in-6, or roughly 16.67%. If a bookmaker offered those prices exactly, the implied probabilities across all six would sum to 100%. They don't. They sum to something larger — usually around 125% in greyhound racing — and that extra ~25% is the bookmaker's built-in margin, sometimes called the overround.

~25%

The bookmaker's built-in margin in a typical greyhound race (overround of ~125%), representing the structural disadvantage every bettor must overcome

Source: How to Win Greyhound Betting (Data-Driven Strategy Guide)

That single number explains a lot of the misery in greyhound betting. The favourite is usually priced to win around 35% of the time, but the implied probability at the head of the market is closer to 40–45% once the short pricing on the favourite is stripped back to fair value. So even though favourites win often, they do not win often *enough*, at the prices they are offered, to turn a profit.

In graded UK greyhound races, favourites strike at somewhere between 31.6% and 42% depending on the grade and the sample you look at. That is a wide range because favourites in A1 open races behave differently from favourites in graded company. But the pattern holds: the favourite is the most likely winner, just not by enough to clear the bookmaker's margin.

~33%

The strike rate of favourites in graded UK greyhound races — a baseline that shows why backing favourites on instinct does not generate profit once the overround is accounted for

Source: How to Win Greyhound Betting (Data-Driven Strategy Guide)

AVOID THIS

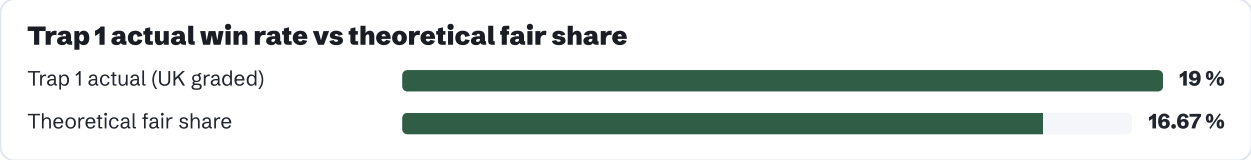
Judging your betting on strike rate alone. A 40% strike rate at 1.80 (4/5) prices is a losing strategy. A 28% strike rate at 4.00 (3/1) can be a winning one. The two numbers only matter together.

At first glance, a 35% strike rate sounds healthy. But if the favourite is being priced at around 2.50 (6/4) on average, the implied chance is 40%, and your edge is negative. You are losing on average even though you are winning more often than the dog loses. This is one of the most counter-intuitive truths in betting — frequency of winning is not the same as profitability. Profitability lives in the gap between your estimate of a dog's chance and the chance implied by the price.

So how do you actually make money at a sport with a 25% built-in tax? By being slightly better than the market on a small number of bets, and then repeating that small advantage over many, many bets. A 3% edge doesn't sound dramatic. But over a season of betting, the maths of compounding takes over.

Think about it this way. Suppose you place 1,000 bets, each at level stakes of £10, and your edge is 3% on each bet on average. Your expected profit is roughly £3 per bet × 1,000 bets = £3,000, before taking variance into account. Some weeks you will be down, some weeks you will be up. Over the long run, the edge accumulates. The reason this works is that your 3% edge applies to every bet, not just to the wins. The bookmaker's margin applies to every bet too, in the other direction. Whichever side has the bigger, more reliable edge eventually wins.

This is the second big mindset shift this book asks of you: stop chasing the home run, and start paying attention to the long-run math. A single bet is mostly noise. The signal lives in hundreds and thousands of bets, well-tracked and well-priced.



One small, repeatable edge in greyhound racing comes from trap bias. In UK graded greyhound races, Trap 1 wins around 18–19% of races, against a theoretical fair share of 16.67% (1-in-6). That gap of 1.5–2.3 percentage points is small, but it is real, it is measurable, and — crucially — it is bigger at some tracks than others.

> **18–19%**
>
> Trap 1's actual win rate in UK greyhound races, illustrating a measurable, repeatable deviation from the expected 1-in-6 baseline that quantitative tracking can convert into an edge
>
> Source: How to Win Greyhound Betting (Data-Driven Strategy Guide)

A quantitative bettor doesn't dismiss this kind of edge as "too small to matter". They log it, track it, and price it into every race they look at. If Trap 1 is genuinely a 19% chance on average, then any time Trap 1 is being offered at odds that imply more than 19% (i.e. longer than 5.26, or 4/1 and bigger), there is potentially a small bet to be made. If Trap 1 is being offered at odds that imply less than 19% (i.e. shorter than 5.26), there is probably a bet to be had against it, either by laying on the exchange or by backing another runner. This is the entire game in miniature: estimate, compare, bet only when the comparison is in your favour.

What a Quantitative Mindset Actually Looks Like at the Track

"Quantitative" is one of those words that sounds intimidating. It brings to mind PhDs, supercomputers, and people who can talk about standard deviations without flinching. The good news is that you don't need any of that to start. What you need is a different set of habits, applied to the same greyhound races you already watch.

The first habit is **writing things down**. Every race you look at, every bet you place, every price you took. If you can't measure it later, you can't improve it. The greyhound racing database that quietly powers everything in this book starts here — with a few columns in a spreadsheet: date, race time, dog, trap, grade, distance, going, BSP, your estimated probability, the bet, the outcome. It looks dull. It is the most valuable thing you will ever build.

The second habit is **estimating the probability before you look at the price**. This sounds odd, but it is the heart of the matter. If you write "I think this dog has a 25% chance of winning" before seeing the odds, you have a benchmark. If the odds then imply 20%, you have found a small edge. If the odds imply 40%, you would be wise to pass — the market is more optimistic than you can justify. Without that prior estimate, the odds just look like "short" or "long", and that is a useless frame.

> **TRY THIS**
> Pick one meeting this week. Before you look at any odds, write down your estimated chance for the favourite in each race. Then look at the odds and convert them to implied probabilities. Note down the races where your estimate is meaningfully higher than the implied probability. You will quickly see which races were "value" by your own reckoning and which were not.

The third habit is **treating results as feedback, not validation**. A win doesn't mean your method was right. A loss doesn't mean it was wrong. A good method can lose on a Tuesday and a bad method can win on a Wednesday. What matters is whether your method produces positive expected value over hundreds of bets, not whether last weekend felt lucky. That is why this book spends so much time on tracking, logging and measurement — because without those, you are flying blind and your brain will quietly convince you that you are winning when you are not.

The fourth habit is **passing on most races**. This is the hardest one for most bettors. If your edge is small and only shows up in certain situations — a particular trap, a particular grade, a particular track, a particular going — then the right answer to most races is "no

bet". A serious quantitative bettor might look at thirty races on a Saturday and place two or three bets. The discipline to say no is what protects the edge when it is there. It is also what stops your bankroll leaking when it isn't.

Tipping services versus a pricing approach

Aspect	Tipping service	Pricing approach
Main output	A list of names to back	An estimated chance for every runner
Decision rule	"Back this one"	"Bet only when my estimate exceeds the market's implied chance"
Relationship to odds	Often ignored or treated as decoration	Central to the decision
Use of past data	Selective — only the wins get quoted	Comprehensive — every race is logged
What you learn from a loss	Often nothing — "unlucky"	Whether your estimate was wrong, and by how much
Time horizon for results	Short, often weekly	Long — hundreds or thousands of bets
Best suited to	Casual recreational betting	Sustainable, measurable edge

What This Book Will and Won't Promise You

Let's get the promises out of the way, so we can work together honestly. Greyhound racing is full of books, tipsters and systems that promise easy money. This is not one of them.

This book will not:

- Hand you a list of dogs to back on Saturday.
- Reveal a "secret" system that prints money with no work.
- Guarantee any particular profit, strike rate, or return on investment.
- Replace the need for discipline, record-keeping and patience.
- Turn you into a professional bettor overnight.

This book will:

- Show you how to collect greyhound racing data in a structured, repeatable way.
- Walk you through cleaning that data so you can actually trust it.
- Introduce you to the racing variables that move results — track, trap, grade, going, race shape, dog form and market signals.

- Give you simple, transparent methods for turning those variables into your own estimated probabilities.
- Show you how to test those estimates against history before you risk any real money.
- Explain how to measure whether your system is actually working once you start betting for real.
- Help you build a staking plan so that a working edge can survive the inevitable quiet weeks.

> **AVOID THIS**
> Treating this book as a tipster in disguise. The value is not in the specific selections. The value is in the system — the way you collect, analyse, price and stake — that you build around them. The selections come and go. The system keeps working.

Who This Book Is Really Written For

You will get the most out of this book if you have been betting on greyhound racing for a while, you have realised that picking dogs based on instinct or weekly form guides is not enough, and you are curious about whether a more structured approach could work for you. Maybe you are a hobbyist who has noticed your tipping service has a losing six-month record. Maybe you are a sharp punter who is wondering if you are leaving money on the table. Maybe you are a data-curious person who would rather build a small model than chase another "best dog" tip.

You don't need a maths degree. You don't need to code for a living. You don't need expensive software. A spreadsheet, a willingness to learn, and a few evenings a week are enough to start. The early chapters assume nothing more than basic comfort with a computer and a normal level of curiosity. As the book progresses, the methods get a bit more sophisticated, but never so much that they require a PhD to understand.

If you have ever been frustrated that a tip landed but you didn't, or annoyed that a dog you fancied at 5.00 (4/1) drifted to 8.00 (7/1) without you acting on it — and you suspect the answer isn't "more tips" but "better decisions" — then you are exactly the person this book is for.

GOOD TO KNOW

Most of the value in this book compounds quietly. The methods get sharper the more races you log. A small improvement in your ability to estimate probabilities, applied across hundreds of bets, is worth far more than any one "must-back" selection. The compounding is in the system, not in any single bet.

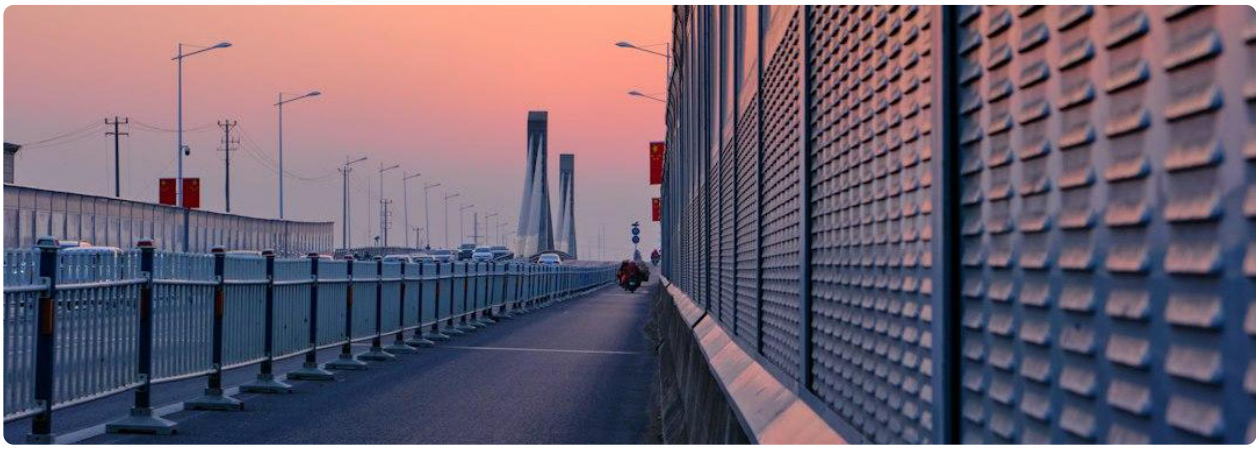
Think of this book as a quiet apprenticeship. Not a series of sure things, but a structured way to make better decisions at the track. The first step is to look at every race not as "which dog do I like?" but as "what is this dog actually worth, and is the price worth paying?". Once that question becomes automatic, the rest of the book is just a set of tools to help you answer it well.

Next, we'll look at where the data itself comes from — the many sources of greyhound racing data available in the UK and Ireland, what each one gives you, and how to think about which ones are worth your time and money.

CHAPTER 2

The Greyhound Data Landscape

Before you write a single line of code or place a single bet, you have to decide what you're going to feed into your thinking. The quality of any quantitative system is bounded by the quality of the data underneath it, and greyhound racing has more data on offer than most newcomers realise, scattered across more sources than are easy to navigate.



This chapter is a map of that landscape. It won't tell you which source to choose outright, because the right answer depends on what you want to build, how much time you have and how much you're willing to spend. What it will do is walk you through the main categories of data available to UK and Irish greyhound bettors, explain what each one actually contains, and flag the practical and legal things you need to know before you start pulling anything into your system.

What "data" actually means in greyhound racing

When people talk about greyhound data, they tend to mean very different things. A newspaper tipster might call a dog's last three finishing positions "data". A trackside observer might call a section time "data". A quantitative bettor should think in broader categories, because the variables you haven't considered are often the ones with the most value.

It helps to split the available information into four buckets, and that's how the rest of this chapter is organised:

- **Race-day information** — the race card, runners, draw, grades, distances, track and going.
- **Historical results** — finished positions, beaten margins, race times, sectional splits, ratings.
- **Market data** — the odds offered by bookmakers and exchanges before and during a race.
- **Reference and contextual data** — trainer records, kennel affiliations, official ratings, going reports, track schemas.

A useful quantitative setup will eventually pull from all four. For now, your task is to know they exist and to start forming a view about which combinations give you the cleanest picture of what is likely to happen at, say, 8.42pm at Romford on a Tuesday.

Race cards, results archives and form sources

The race card is the front door. Every meeting — whether it's a graded fixture at an independent UK track, a Premier Greyhound Racing (PGR) event or a meeting in Ireland — publishes a card in advance, typically 24 to 48 hours before the first race. The card tells you which dogs are running, over what distance, from what trap, in what grade, and for what kind of prize. Without the card you can't really begin.

Cards are widely available in three ways. Print and web tipsters publish condensed versions, but these usually strip out the detail that matters for modelling, such as the full set of grades a trainer has runners in or the dog's full seasonal record. Track websites publish their own cards, often with running comments and sectional times attached, and these are a good baseline. Dedicated greyhound results services, both free and paid, aggregate cards across many tracks and let you look up historical meetings going back several years.

For results, the picture is similar. Track sites archive their own meetings, but they tend to disappear from the front page quickly and the deeper history can be patchy. National greyhound bodies in the UK and Ireland publish official results, though typically with a delay and without the granular timing data you may eventually want. The most consistent source of historical results, including margins, sectional times and beaten lengths, comes from the paid feeds discussed below, but free aggregation sites have improved considerably and are worth trialling before you spend anything.

> **GBP 1.5 billion**
> UK greyhound betting turnover in 2022-23, down 23% in real terms over three years
> Source: How to Win Greyhound Betting

One thing to bear in mind as you sample these sources: not all of them define a "win" the same way. Some count only graded races, some exclude walkovers, some treat dead-heats as half-wins. The definitions may seem trivial until you try to build a strike-rate column and find your numbers shifting by a couple of percentage points depending on where you pulled them from. We'll deal with this in detail when we get to cleaning and validating data, but it's worth knowing now that "the same race" can have subtly different outcomes recorded in different places.

Comparing common results sources

Source type	Coverage	Typical update speed	Granularity	Common limitations
Track websites	One track at a time	Minutes after the race	Finishing order, basic times	Short history, inconsistent fields
Free aggregation sites	Multiple tracks, mostly UK and Ireland	Same day	Results, basic form, often limited cards	Variable depth, ads, sometimes incomplete
Paid results feeds	Multiple tracks, deep history	Near real-time	Results, sectionals, comments, ratings	Subscription cost, contract terms
National governing bodies	All licensed tracks in their jurisdiction	Hours to days later	Official results, ratings, sometimes grades	Slower, fewer timing splits

Sectional times, grades and official ratings

This is where greyhound data starts to look genuinely interesting from a modelling perspective, because these are the variables that move beyond "who won" and into "how they won and how strong the opposition was".

Sectional times split a race into segments, typically the run to the first bend, the middle section and the run to the line. Not every track publishes them in a form that is easy to consume, and definitions vary. Some feeds report three splits, some report two, some report cumulative times only. When you start using sectionals, your first job is to make sure that a "first-bend sectional" recorded at one track really means the same thing as one recorded at another. If they don't, treat them as separate variables rather than one combined feature.

Grades are the levellers of greyhound racing. A grade tells you the class of competition a dog is currently racing in — A1, A2, OR, S1, D1, and so on. The letters and numbers differ by track, which is one of the trickiest things about greyhound data and a recurring theme in this chapter: standardisation is patchy. A dog dropping from A1 to A2 at Romford isn't directly comparable to a dog dropping from an Open race to an A1 at Crayford, even if the words sound similar. Encoded properly, however, grades are extraordinarily useful because they tell you, in effect, how the sport itself has judged each dog's recent level.

Official ratings are issued by the relevant regulatory body for each licensed track. They are recalculated periodically using the dog's recent results and form. Ratings are appealing because they are produced by people with access to the full picture, including information you don't see, but they are also lagging indicators and they update infrequently. For modelling they work best as a baseline or a sanity check, rather than as your primary feature.

> **TRY THIS**
> Pick one dog that has run three times in the last fortnight and write down, in your own words, what each of those runs tells you about its current form. Then pull the same three races from three different sources and compare. You'll quickly see where the small inconsistencies live.

Where odds and market data actually come from

Odds are a special category of data because they are not facts about the race. They are the collective view of the betting public, adjusted for bookmaker margin and sometimes sharpened by sharp money. As a quantitative bettor you have two reasons to collect them.

The first is that the market price is, on average, your best single estimate of a horse or dog's probability of winning. It's not always right, and it is systematically biased in certain places, but it is hard to beat with a single naive model. The second reason is that the gap between *your* estimated probability and the market's probability is exactly what tells you whether a bet has value.

In greyhound racing, odds come from a small number of places. Bookmaker websites and apps publish prices on each race, usually from a few hours before the off until a couple of minutes before the start. Betting exchanges, where greyhound liquidity exists at the bigger meetings, let you see both back and lay prices, which gives you a deeper view of where the market is moving. Specialist odds comparison sites aggregate prices across many bookmakers and sometimes show historical odds movements.

There is a hierarchy of quality here. An odds feed that captures prices at a single point in time is useful but limited. A feed that captures the full price curve, ideally with timestamps down to the second, is what you want if you plan to model how the market reacts to news such as a late non-runner or a market mover on the exchanges. We will return to this in the chapter on market odds, but when you are choosing a source now, decide early whether you want just the final price or the journey to it.

33% Win rate of favourites in graded UK greyhound races, establishing the baseline performance benchmark that data-driven bettors must beat

Source: How to Win Greyhound Betting

Legal, licensing and responsible gambling basics

This is the part of the chapter most people skip and later regret. Greyhound betting in the UK and Ireland is regulated, and the rules affect how you can collect and use certain kinds of data, particularly anything that touches price feeds or betting-related APIs.

In the UK, betting on greyhound racing is legal but only through operators licensed by the Gambling Commission. Using an unlicensed operator is against the law and removes most of the consumer protections you would otherwise have. In Ireland, betting is regulated by the Revenue Commissioners and licensed operators are listed publicly. If you plan to place bets as part of testing your system, make sure you are using a properly licensed bookmaker or exchange.

On the data side, the rules are less obvious but no less important. Paid data feeds come with licences, and those licences usually prohibit you from redistributing the raw data or using it to power a tipping service you sell to others. The terms vary, but reading them is part of choosing a feed. A source that looks free may quietly prohibit commercial use; a paid source may grant you personal use but forbid syndication.

AVOID THIS

Don't assume that because data is visible on a website it is free to scrape or republish. Many feeds, including some odds comparison sites, prohibit automated collection in their terms of use. If you plan to pull data automatically, check the licence first.

Finally, a word on responsible gambling. Quantitative betting can feel detached because you're working with numbers, but it is still gambling and the risks are real. The book you're reading is about building a system for deciding what the numbers are actually worth, not about promising outcomes. Set deposit limits with your bookmaker, use the

responsible gambling tools your operator provides, and never bet more than you can comfortably afford to lose. If betting stops feeling like a structured analytical exercise and starts feeling like something else, step back. If you are in the UK, GambleAware and the National Gambling Helpline are good first ports of call; in Ireland, Problem Gambling Ireland offers confidential support. This is general information, not a substitute for advice tailored to your own situation, and a qualified professional can help if you have specific concerns.

Practical considerations when picking your sources

You can now see that the data landscape is wider than a single feed, and the right approach is usually to combine two or three sources rather than rely on one. Before you commit, run through this short list.

Choosing a greyhound data source

- [ ] Does it cover the tracks and countries you want to bet on?
- [ ] Does it include the historical depth you need (ideally several years)?
- [ ] Does it provide the variables you actually plan to model (sectionals, grades, ratings)?
- [ ] Are the definitions consistent across tracks, or will you need to standardise?
- [ ] What are the licence terms, and do they match how you intend to use the data?
- [ ] How often is it updated, and does that match the timing of your betting?
- [ ] What does it cost, and is there a free trial you can use to evaluate it properly?

That last point matters. Most paid feeds offer a trial window. Use it. Pull a sample of races, build a small table in a spreadsheet, and see whether the data is as complete, consistent and timely as advertised. The cost of finding this out during a trial is an afternoon. The cost of finding it out after you've built a model around it is a rewrite.

Finally, remember that the data landscape is not static. New feeds appear, existing ones change their terms, and regulatory requirements evolve. Build your system so that swapping a source is a configuration change rather than a rebuild. The chapters that follow — particularly the ones on setting up your data stack and pulling data through APIs — assume you have made at least a tentative choice. If you haven't, treat that as your first concrete task before moving on.

KEY TAKEAWAYS

- Greyhound data falls into four buckets: race-day information, historical results, market data, and reference or contextual data.
- Free sources are a sensible starting point but vary widely in depth, history and consistency.
- Paid feeds typically offer more complete records, including sectionals, grades and ratings, but come with licence terms you should read.
- Odds are not facts about the race; they are the market's collective view and the most important benchmark for your own model.
- Always check licensing and responsible gambling rules before you begin collecting data or placing bets.

CHAPTER 3

Setting Up Your Data Stack

Every quantitative betting system begins the same way: with a tidy folder, a clear file name and a backup you actually trust. Before any clever model can do its job, the data underneath has to live somewhere sensible, in a shape you can query, share and rebuild if anything goes wrong. This chapter is the unglamorous plumbing. Done well, you will hardly notice it. Done badly, you will spend more time fighting files than finding an edge.



The Real Question: What Scale Are You Working At?

Before choosing tools, it helps to know roughly how much data you are dealing with. Greyhound racing is not as enormous as horse racing, but it is not tiny either. A useful rule of thumb: if you are starting with a single meeting format and a handful of tracks, you can stay in a spreadsheet for a while. The moment you want every UK and Irish meeting for several years, you have outgrown it.

> **~25,000** Approximate number of BAGS (Bookmakers Afternoon Greyhound Service) races produced per year in the UK, illustrating the scale of raw race data available to build a betting data stack
>
> Source: howtowingreyhoundbet.com

That number is the calm side of the picture. Each race carries around six runners, and each runner carries a handful of attributes: finishing time, sectional splits, trap number, weight, grade, going, race distance, trainer, recent form and so on.

150,000

Estimated individual data points generated annually from BAGS racing alone (25,000 races × 6 runners), highlighting the volume of finishing times, sectional splits, trap positions, weights and grades a data stack must handle

Source: howtowingreyhoundbet.com

A spreadsheet copes with a few thousand rows without complaint. It starts to groan somewhere between 50,000 and 200,000 rows, depending on how many lookups and formulas you stack on top. Two hundred thousand is not a theoretical figure either. After a couple of years of UK and Irish data, with a reasonable set of derived columns, you can reach that range on a single laptop. If you want to keep a rolling five-year history, you are firmly in database territory. So the first decision is honest: are you building a weekend hobby model or a multi-year archive? Neither answer is wrong, but the tool should fit.

Spreadsheets Versus Simple Databases

Spreadsheets and databases are not enemies. They each do a job. The mistake is using one where the other belongs. A spreadsheet is excellent for:

- Quick exploration of a small slice of data
- Manually reviewing a handful of races to sanity check a model
- Sharing a single file with a friend or collaborator who does not code
- Recording notes, observations and overrides in a human-readable way

A simple database is better for:

- Storing tens of thousands of rows that you want to filter, join and summarise
- Running the same queries repeatedly without copying formulas down
- Keeping your data in one place rather than scattered across ten monthly files
- Building a pipeline where new data lands cleanly on top of old

Here is a practical comparison:

Spreadsheets versus simple databases for racing data

Need	Spreadsheet (Excel, Sheets, Calc)	Simple database (SQLite, DuckDB)
Capacity before slowing down	Tens of thousands of rows comfortably	Hundreds of thousands to millions of rows
Sharing with non-technical collaborators	Excellent, everyone can open it	Requires a viewer or a small script
Risk of accidental edits breaking the whole file	High, especially in long formulas	Low, queries are read-only unless written otherwise
Reproducibility of calculations	Patchy, formulas drift over time	High, queries are saved and re-run
Version control with Git or similar	Difficult for binary files	Easier if you keep raw data as CSV or Parquet alongside
Learning curve	Very low	Low to medium, basic SQL is enough to start

A common path, and a sensible one, is to use both. Keep a spreadsheet as a viewing and review tool. Store the canonical data in a database file or a folder of CSV files, then export a filtered slice to a spreadsheet whenever you want to look at it with fresh eyes. This is not a compromise. It is the standard approach for many small quantitative projects. If you are choosing a database file format, two options stand out for a one-person setup. SQLite is a single file you can drop into any folder, open with a free tool and query with simple SQL. DuckDB is similar but optimised for analytical queries across larger tables, which is exactly what you do when summarising races by grade, distance or track. Both are free, both run on Windows, Mac and Linux, and neither needs a server. For most greyhound bettors, either one is plenty.

Choosing a Tool You Will Actually Use

The best tool is the one you will open three nights a week for a year. A theoretically perfect stack that you abandon after a fortnight is worse than a rough stack you actually use. For most people reading this book, the realistic choices narrow to three:

1. **Spreadsheet-only.** Excel, Google Sheets or LibreOffice Calc, with a tidy folder of CSV files feeding them. This is the lowest friction path and works well for a single track or a small set of meetings.
2. **Database plus a little SQL.** A SQLite or DuckDB file, plus a basic query tool. SQL itself is a small language. You can learn the handful of statements you need in an afternoon and reuse them forever.

3. **Python or R plus a database.** If you are comfortable with code, or willing to learn, this gives you the most flexibility. Python with pandas, or R with dplyr, lets you reshape and summarise racing data quickly, then drop back to a spreadsheet for inspection.

There is no shame in option one. There is also no future in it if you want to keep three years of history and run it through a model. Move up when the data asks you to, not before. A few practical notes on picking:

- **Pick the tool that matches your weakest habit, not your strongest.** If you know you will not write code at midnight after a long day, do not pretend you will. Build a spreadsheet-first system and be honest about it.
- **Avoid exotic stacks.** You do not need Kubernetes, a cloud warehouse or a streaming pipeline. You need a folder, a database file and a way to read them.
- **Keep your data file portable.** A single SQLite file you can email yourself is more resilient than a fancy cloud setup you cannot reconstruct without a login.
- **Test your stack with one meeting first.** Before pulling in months of history, get a single meeting end-to-end. If that flows cleanly, scale up.

> **TRY THIS**
> Before installing anything new, write down the three things you want to do with your data this month, such as "list all winners at Monmore over 480m in 2023", "compare trap-1 strike rate by grade" and "export today's runners to a spreadsheet". Choose the tool that lets you do those three things with the least friction. That is your stack.

Folders, Names, Backups and the Raw Versus Cleaned Split

Once you have a tool, the boring decisions matter more than the clever ones. A folder structure you understand in six months is worth more than a clever schema you forget in six weeks. A layout that has served many small data projects well looks like this:

- **data/raw/** – files exactly as they arrived from your source. Never edited, never overwritten. If you ever doubt a cleaned number, you can come back here and check.
- **data/processed/** – cleaned and merged files that your queries and models read. Anything in here can be rebuilt from the raw folder, so losing it is annoying but not fatal.
- **notebooks/** or **queries/** – your scripts and saved queries, named so you remember what they do.

- **exports/** – anything you have pulled out for review, sharing or a betting decision. Treat them as disposable.
- **backups/** – dated snapshots, kept somewhere physically separate, as discussed below.

The single most important rule is the raw versus cleaned split. Raw data is sacred. You never modify it in place. Every change you make to clean it, normalise a column name, fix a date format or drop a stray row, happens in a script that writes to the processed folder. That way, when you improve your cleaning logic six months from now, you can rerun it from scratch and get a consistent result. If you edit the raw file by hand, you have lost that ability. Naming conventions matter more than they sound. A few rules of thumb:

- **Use dates in ISO format.** YYYY-MM-DD sorts correctly in any list and is unambiguous in UK English.
- **Include the source in the filename.** Something like 2024-05-12_monmore_results_api.csv tells you at a glance where it came from and what it is.
- **Avoid spaces and special characters.** Stick to letters, numbers, underscores and hyphens. Tools and command lines are happier.
- **Version, do not overwrite.** If you rerun a query and the result changes meaningfully, save the new file alongside the old one with a version tag rather than overwriting.

Now the part nobody enjoys but everybody needs: backups. Three copies, two different storage types, one kept offsite. That is the classic rule, often shortened to 3-2-1, and it applies as much to a racing spreadsheet as to a wedding photo album. A practical version for a one-person betting project looks like this:

1. **Your working folder.** The files you use day to day.
2. **A local backup.** An external drive, a USB stick, or a separate folder on a different drive. Updated weekly.
3. **An offsite copy.** A cloud folder you trust, or a drive kept at a different physical location. Updated weekly or monthly.

> **AVOID THIS**
> Storing your only copy of three years of cleaned data in a single folder on your laptop's desktop. Drives fail, laptops get stolen, and a ransomware attack or an accidental drop can wipe the lot in a moment. Losing the raw data is recoverable; losing both raw and cleaned data together is the end of your model.

Automation helps, but only if it actually runs. A backup script that you forget to trigger is not a backup. If you use a cloud service, check occasionally that files are syncing and that you can open them from a fresh device. A backup you have never restored from is a backup you have never tested.

Putting It Together

Here is a checklist you can work through the first time you set up your stack. You do not need to finish all of it in one sitting. The goal is a system that survives a few months of use without falling over.

Your data stack setup checklist

- [ ] Decide your scope honestly: one track, several tracks, or full UK and Ireland coverage
- [ ] Pick a primary tool: spreadsheet only, database plus SQL, or Python/R plus a database
- [ ] Create a top-level project folder with subfolders for raw, processed, queries, exports and backups
- [ ] Adopt a naming convention for files, including dates in ISO format and source identifiers
- [ ] Set up a backup routine with at least one local and one offsite copy
- [ ] Pull a single meeting through your pipeline end-to-end before scaling up
- [ ] Write a short README explaining the folder layout and how to rebuild processed data from raw

To make this concrete for your own situation, spend a few minutes filling in the worksheet below. It is deliberately short. The point is to commit to choices before the data starts arriving.

Planning your data stack

1. Which tracks and meeting types do you actually want to cover, and roughly how many races per week is that?
2. What is your primary tool for storing and querying data, and what is your fallback if it breaks?
3. Where will your raw files live, and what naming convention will you use from day one?
4. How will you back up your data, how often, and where does the offsite copy live?
5. What is the one query or summary you want to be able to run easily within a week of setting this up?

That last question matters more than it looks. If your stack can answer a useful question on day one, you will keep using it. If it just sits there waiting for "real analysis" to begin, you will quietly drift back to scribbled notes and tipster emails. The point of a data stack is not to be impressive. It is to give you a reliable place to put numbers and a reliable way to ask questions of them. Once your stack is in place, the next step is feeding it. The following chapter looks at how to use racing APIs to pull results, runners and odds into your system on a schedule, and how to design that pull so your stack does exactly what it was built to do.

KEY TAKEAWAYS

- Match your tools to the scale of data you actually plan to keep, not the scale you imagine keeping
- Spreadsheets and simple databases complement each other; one is for viewing, the other for storing
- Keep raw data untouched and build cleaned versions through repeatable scripts
- Use clear, dated, source-tagged names so future you can recognise the files
- A simple, tested backup routine is more valuable than a clever stack you forget to use

CHAPTER 4

Pulling Data Through APIs

The firehose is already on. By mid-afternoon most weekdays, race meetings across the UK and Ireland are filling row by row with finishing positions, trap numbers, weights and traded odds, and a handful of API endpoints will hand you most of that data within seconds of asking. This chapter is about turning those endpoints into a routine, so that you wake up each morning with a clean, scheduled dataset sitting in your own database rather than a pile of screenshots.

Think of the work ahead as plumbing. You are not yet analysing, modelling or betting. You are simply deciding which taps to turn on, how often to run them, and what to do when one of them leaks. The plumbing is unglamorous, but it is the layer everything else in this book rests on. Get it right now and the rest of the project becomes a sequence of small, well-defined jobs; get it wrong and you will spend months wondering whether your edge is real or whether the data underneath it has a leak of its own.

The Anatomy of a Racing API

An API is just a door. You send a request, the server checks whether you are allowed in, and it hands back a parcel - usually JSON, sometimes CSV - containing the data you asked for. Racing data providers structure that door around a small number of predictable endpoints, and the trick is to learn which ones to ask for and what shape the response will take. Field names will vary from provider to provider, but the underlying entities are remarkably consistent across the industry.

The endpoints you will lean on most are roughly these:

- **Meetings or fixtures.** Lists the tracks running on a given date, with race times and the number of races. This is your calendar.
- **Races.** The individual races within a meeting, usually keyed by track, date and race number. May include distance, grade, going and race type (flat, hurdles, marathon).
- **Runners.** The dogs entered in a particular race, with trap number, weight, official rating, breeder, owner and a form summary.
- **Results.** The finishing order, finishing times, sectional splits where available, and any comments from the judge (for example, "crowded at first bend" or "slow away").

- **Odds.** Market prices at one or more snapshots through the day - morning line, off-time, and sometimes the full price ladder with traded volumes.
- **Dog profile.** A single dog's career record, often paginated by season or by meeting.
- **Tracks.** Static information about each track - direction, length, surface, bend configuration and home country.

Not every provider will give you all of these, and some will tuck extra fields inside a single nested object. That is fine. The point is to recognise the shape of the data and to insist on the elements your future self will actually need. The BAGS (Bookmakers Afternoon Greyhound Service) circuit alone produces something like

races a year, and that is before you count the independent and Irish tracks, so the volume you can pull through these endpoints is substantial.

What to capture from a racing API endpoint

- [ ] A stable race identifier (track + date + race number, or a provider's own ID)
- [ ] Date and off-time, stored in UTC so your daylight-saving maths stays sane
- [ ] Distance, grade and going as separate fields, not buried inside a string
- [ ] Trap number, weight and any official rating per runner
- [ ] Finishing position, finishing time and sectional splits per runner
- [ ] One or more odds snapshots per runner, each with its own timestamp
- [ ] The provider name and the version of the response schema, so future-you can spot when things have changed

You will not use every field immediately, but capture them now. It is far easier to ignore an extra column than to realise two years from now that you forgot to record trap number, and that the historical fix-up is a small nightmare. If a field is optional in the response, store it as *missing* rather than pretending it does not exist.

Authentication and the Art of Asking Politely

Most racing APIs authenticate with a simple key sent as a header or query parameter, though a few use OAuth tokens or signed requests for the more sensitive endpoints. Either way, the rules of the road are the same: keep the key secret, send it only to the provider's domain, and never paste it into a public notebook or a screenshot of your code.

> **TRY THIS**
> Store API keys in an environment variable or a small secrets file that is excluded from version control, rather than hard-coding them into your pull script. The first time you accidentally push a key to a public repository is the last time you forget.

Every provider also imposes rate limits - usually a number of calls per minute, sometimes per hour or per day, and sometimes per endpoint. Read the small print before you write a single line of code. Common patterns look like 60 calls per minute on a free tier, 5,000 calls per hour on a mid-tier plan, or 100,000 calls per month on a heavy data plan. Your job is to stay comfortably under those ceilings rather than brush against them. Rate-limit responses are usually polite but firm - a 429 status with a "Retry-After" header - and repeated infractions will get your key suspended.

Two habits make this easy. First, **cache aggressively**. If you have already pulled today's meetings once, do not pull them again five minutes later just because a script restarted. A tiny local cache - even a dictionary written to a JSON file with a short expiry - will save you from a rude email and from the same five-second bills. Second, **batch where you can**. Most providers will let you ask for a whole date, or a whole meeting, in a single call, which is much friendlier than a hundred individual requests fired in a tight loop.

> **AVOID THIS**
> Looping over every race and firing a fresh odds call per race during the afternoon rush. That is how you get throttled at exactly the moment you most want the data, and on a busy ten-race card you can burn through a daily allowance in a few minutes.

A reasonable rule of thumb is to keep your peak call rate at a third of the published ceiling. That gives you headroom for retries, for the occasional batch job you forgot was scheduled, and for the provider themselves having a wobble.

Scheduling Your Pulls: Daily, Hourly or On Demand

Not all racing data ages at the same speed. Results are fixed once the judge has called them and will never change, so you can pull them overnight and forget about them. Pre-off odds are moving targets - they refresh every few seconds in the last minutes before a race - and chasing them is a job for a separate process that streams rather than polls. Form and runner profiles drift slowly; once a day is plenty for those.

A clean weekly rhythm for a small quantitative operation looks something like this:

A weekly API pull schedule

Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7
Pull tomorrow's fixtures and runner declarations for every track on the calendar	Pull pre-off odds snapshots at three times: morning, midday and 30 minutes before the first race	Pull results and finishing times for yesterday's meetings as soon as they are final	Pull sectional splits, weight changes and any race-replay metadata	Pull settled BSP or industry SP values once they have been reconciled	Refresh dog profiles for any new entrants not seen in the past fortnight	Backfill any missing races, retry any failed calls, and archive raw response payloads

You do not need to drive all of these by hand. A simple cron job on a small server, a scheduled task in your cloud account, or a managed workflow tool like GitHub Actions, Prefect or Apache Airflow will run the schedule for you. The important thing is that the schedule is written down somewhere, even if it is just a comment at the top of the script, because in six months you will not remember what runs when, and the day you change provider you will be glad of the map.

For UK greyhound data alone, the scale adds up quickly. BAGS racing throws off roughly

runner-level data points a year - one row per dog per race, capturing finish times, sectional splits, trap positions and weights. That is the size of the table you should expect to be sitting in your database by the end of your first full season of pulling, and it is the raw material every chapter from here on will be working with.

You will also need a sensible policy on **backfills**. When you first connect to a new provider you will almost certainly want to pull years of history in one go, and that is fine - but throttle it. A single Python script happily firing one call per second for six hours is still well under most ceilings, but a careless *while True* loop is how people end up locked out of an API the day before a major meeting.

When Things Go Wrong: Logging, Gaps and Schema Changes

APIs do not stay the same forever. A provider will add a field, rename a column, retire an endpoint, or simply have a bad afternoon. If you do not log your own activity, you will not know what is missing, when it went missing, or which version of the data your model is built on. Three practices will save you from silent disasters.

First, log every call. At minimum, record the timestamp, the endpoint hit, the parameters sent, the response status code, the response size and a hash of the payload. A simple append-only log file is fine; a small SQLite table is better. The point is that six months from now you can answer the question, "what did this endpoint actually return on 14 March?" without guessing. Logs are cheap. Regret is not.

Second, log every failure with enough context to retry it. A bare "API call failed" is not enough. Capture the URL, the parameters, the status code and the body of the error. You will be grateful when, on a wet Tuesday in October, a single meeting's odds never arrived and you have to rebuild it from a saved request rather than reverse-engineer it from a blank stare. Most modern HTTP libraries make this straightforward with a small wrapper that wraps every call in a try/except and writes to a structured log.

Third, expect gaps. Even the best providers will occasionally miss a race, misreport a trap number, or return a result an hour late. Decide in advance how you want to handle missingness, because your future self will be in too much of a hurry to think it through on the day. Common-sense options include:

- **Flag and keep.** Store the row with a missing-value marker such as *null* or *NaN* and let downstream analysis decide whether to drop or impute it.
- **Drop and log.** Exclude the race entirely and record the reason in a rejections table. Useful when the gap is structural - the judge never called the result, or the meeting was abandoned.
- **Retry, then escalate.** Re-pull the same endpoint two or three times at increasing intervals before giving up and flagging the race for human attention.

A useful habit is to keep a small "data issues" notebook - a spreadsheet, a Markdown file, anything - where you jot down anything odd you notice in the raw responses. A race with seven runners instead of six. A trap number of zero. An odds snapshot stamped at 02:00 the next morning. These notes are gold when you come to clean the data in the next chapter, and they are also how you gradually learn the personality of each provider.

It is also worth being honest, even at this early stage, about the opponent. You are not just collecting data; you are collecting data in order to compete with a market that sees the same information and more, often faster. A simple logistic regression built on greyhound form data pulled from an API has been shown to pick winners at around

> **33%**
>
> Winner-prediction accuracy of a simple logistic regression model built on FastTrack API greyhound form data versus the market's starting-price benchmark, showing how API-fed models can be quantitatively compared to the market
>
> Source: Betfair Data Scientists (GitHub)

accuracy, while the market's starting-price favourite wins at roughly 41.7%. The market is the bar, not an afterthought. Your quantitative system is being built to find prices that are *wrong*, and that only matters if you measure yourself against the same benchmark you

eventually plan to beat.

Pulling It All Together

By the end of this stage you should have three things in place. A clear inventory of the endpoints you rely on and the fields each one returns. A pull schedule, automated to the point where it runs without you. And a logging setup that tells you, calmly and exactly, what worked, what failed and what changed. If any of those three is missing, the next chapter will hurt more than it needs to.

You now have data in your own system, on your own terms, in roughly the shape you will use it in for months to come. The next step is the unglamorous one: making sure it is actually trustworthy before you trust it. Cleaning and validation is where a lot of quiet betting systems quietly go wrong, and it is where we are headed next.

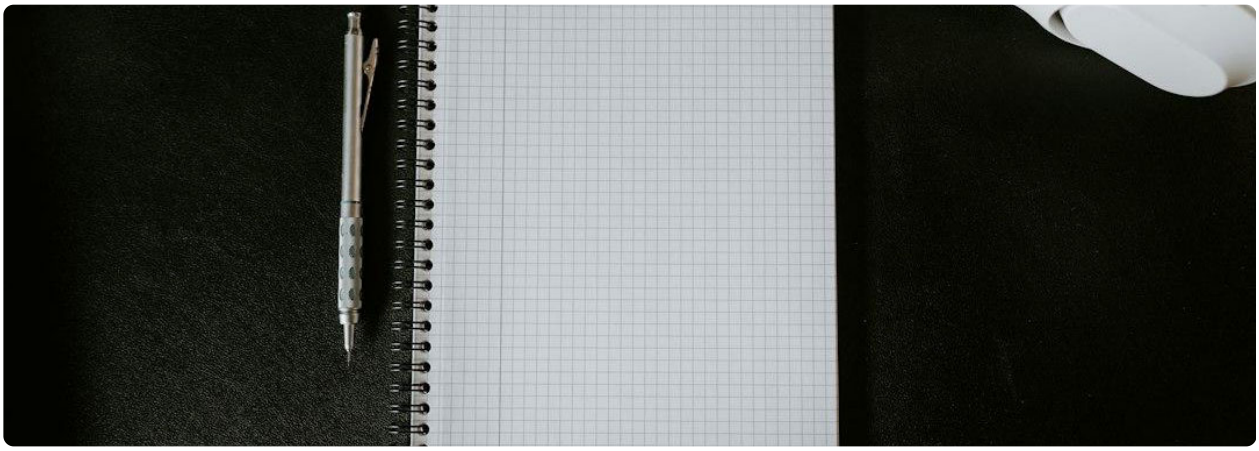
KEY TAKEAWAYS

- A racing API is a handful of predictable endpoints - meetings, races, runners, results, odds, dog profiles, tracks - and you should capture more fields than you need today.
- Keep API keys out of code, stay comfortably under rate limits, and cache aggressively so you do not get throttled.
- Schedule pulls by data type: fixtures and results daily, odds on a tight pre-off loop, profiles weekly.
- Log every call and every failure with enough context to retry, and decide in advance how to handle missing or odd rows.
- The market's starting-price benchmark is your real opponent; measure your API-fed model against it from day one.

CHAPTER 5

Cleaning and Validating Data

Every winning greyhound betting model is built on top of a slightly boring pile of cleaned rows. The flashier work comes later, but if the foundation is shaky, the conclusions drawn on top of it will be too. This is the chapter where you slow down, get your hands dirty with the data itself, and turn a messy feed into something your future self can actually trust.



Cleaning and validation is unglamorous on purpose. It is also where most home-built greyhound models quietly fall apart. A race card pulled from a feed can look perfectly tidy until you start asking it real questions, at which point you find a trainer whose name changes spelling every six months, a track that has three different labels, distances measured in metres on some rows and yards on others, and a few "runners" who never actually stepped onto the sand.

What cleaning is actually for

Cleaning is not about polishing data so it looks nice. It is about removing every plausible way your future model can be lied to. The two distinct goals are worth separating:

- **Consistency.** The same thing should always be written the same way. "Shelbourne" and "Shelbourn" are two strings but one track. Your code cannot know that unless you teach it.
- **Trustworthiness.** Every record should describe what you think it describes. A finishing position of zero probably means a non-runner, not a dead-heat win. Without

validation rules, your model will treat that row as a true outcome and quietly learn the wrong lesson.

Most greyhound bettors skip straight from "I have a CSV" to "I have a model". The chapter in between is where the edge is either protected or destroyed. Treat cleaning as part of the model, not a separate chore.

The first pass: catching the obvious stuff

Before you write clever logic, run a series of blunt checks on every new dataset. These checks take minutes and catch the largest class of bugs.

Duplicates from API retries

If you are pulling data on a schedule (see Chapter 4), some races will arrive more than once. A 24-hour retry during an outage can land the same race card on your desk at two different timestamps. Build a deduplication step that decides what counts as a unique race: a combination of track, race identifier, race date and off-time is usually enough.

Typos and case differences

"Sheffield" and "sheffield " (with a trailing space) are not the same string to a database. The same applies to trainer and dog names. A useful first move is to trim whitespace and convert everything to title case before doing any matching. You will not catch every typo that way, but you will catch the cheap ones.

Non-runners dressed as runners

This is one of the most common and most damaging errors. Race cards usually list every dog that was entered, including those who were later withdrawn. Results feeds sometimes carry a finishing position of zero, an empty time, or a special status like "scr" or "withdrawn" tucked into the comments column.

Without a clear rule, these rows slip into your dataset as legitimate losers. The model learns that a dog with no form finished last, and your edge bleeds away. Decide once, at the cleaning stage, what counts as a runner. A common rule: a runner must have a real finishing position between 1 and the field size, and a non-empty finishing time.

> **GOOD TO KNOW**
> Some feeds list reserve dogs separately. Reserves that never ran should be excluded from your analysis, but reserves that *did* run are simply runners and should be treated as such.

Empty traps

If a trap number is missing or zero, the dog did not run. Same rule applies: drop the row.

Standardising track names, grades and distances

Once the obvious stuff is removed, the next task is to standardise the columns your model will actually read. Three fields in particular cause repeated trouble.

Track names

Many greyhound venues are listed under more than one spelling: "Shelbourne Park", "Shelbourne", "SHELB". Create a small lookup table that maps every variant you see to a single canonical name. The same applies to whether you include "Park" or "Stadium" in the label. Pick one version and stick to it.

Two things to watch for:

- **Two different tracks with similar names.** "Perry Barr" and "Perry Beeches" are different venues. A naive fuzzy match will merge them. Always check that any merging rule maps to a real, single track.
- **Track rebrands.** Venues occasionally change name. Decide whether you treat the rebranded track as a new entity or as a continuation of the old one. For most quantitative work, treating it as a continuation is fine, but record the change in your cleaning log.

Grades

Grade labels are some of the messiest data you will encounter. A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, OR, S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, H1, H2, H3, H4, G1, G2, G3, G4, G5, N1, N2, N3, N4, N5, P1, P2, P3, P4, P5, T1, T2, T3, T4, T5, T6 — the list goes on. Some feeds use a single letter plus a number, others use a letter plus a number and an extra suffix, others mix upper and lower case.

Normalise everything to upper case, strip whitespace, and decide whether you want to keep the full grade or collapse it to a coarser band (for example, "Open", "A", "B", "D", "S"). For most quantitative work, keeping the full grade as one feature and a coarsened band as another gives you flexibility later.

Distances

Distances are usually expressed in metres in UK and Irish greyhound data, but the format varies: "480m", "480 m", "480", "480.0". Strip the unit, cast to a number, and you're done in most cases. Watch for distances expressed in yards — that almost always means a different code path (an older feed or a vendor you do not usually use).

Two more date-and-time gotchas:

- **Off times stored in different time zones.** Normalise to a single time zone (UTC is usually a sensible choice).
- **Date as a string in mixed formats.** "01/03/2024" is ambiguous between 1 March and 3 January depending on the feed. Convert to a proper date type as soon as possible, ideally at the import stage, so nothing downstream has to guess.

Handling missing values in a principled way

Missing values are not all the same, and treating them as one thing is a recipe for silent errors. There are three useful strategies, and the right choice depends on what the missing value actually means.

Strategy 1: Drop the row

Use this when the missing value is one of the things you are trying to predict. If a row has no finishing position, no finishing time, and no clear withdrawn status, it is not a record you can learn from. Drop it and move on.

Strategy 2: Impute a sensible default

Use this for variables you will use as inputs to your model. For example, if the sectionals for a particular race are missing, you might impute the median sectional at that track and distance. The important word here is *sensible*. Imputing zero would quietly tell the model that the dog had a sectional time of zero seconds, which is nonsense and will distort every calculation it touches.

Strategy 3: Flag the missing as its own value

Sometimes "missing" carries information. A dog that has no recent form might be returning from injury or might be a newcomer; either way, the absence is meaningful. Rather than imputing a number, create a flag column — sectional_missing, last_run_missing — and let the model decide how to use it.

A useful rule of thumb: if a missing value is rare (under 5% of rows) and clearly random, imputation is usually fine. If it is common or correlated with something interesting, flagging is safer. Whatever you choose, do it once, in one place, and document it.

Validation: sanity checks that catch real bugs

Cleaning without validation is just rearranging the same mistakes. Validation is the step where you ask the cleaned dataset to prove it is internally consistent and externally plausible.

Quick validation checks for any greyhound dataset

- [ ] Total races per year match the rough scale you expect from your source
- [ ] Field sizes fall within sensible bounds (typically 4 to 6 for most UK tracks, up to 8 at some venues)
- [ ] Winning strike rates per trainer, kennel or trap fall within a plausible range
- [ ] Distribution of finishing times at a given track and distance is bell-shaped, not bimodal with a weird spike
- [ ] Proportion of non-runners is stable over time and roughly matches the vendor's reported rate

The first check is the easiest sanity test. Compare the cleaned dataset against the rough scale of what your source actually produces.

> **14%** A real-world trainer strike rate at a specific venue cited as below the random baseline for six-runner fields — a worked example of a validation benchmark for checking whether cleaned data and a model are producing sensible probabilities.
> Source: HowToWinGreyhoundBet.com

A trainer winning 14% of races at a six-dog track is below what you would get by picking a dog at random (roughly one in six, or about 16.7%). That is exactly the sort of figure that should set alarm bells ringing — either the trainer is genuinely below average, or the dataset has a bug. If you see strike rates that low across many trainers at many tracks, the data is the problem, not the trainer.

Other useful validations:

- **Field size distribution.** Plot the count of races by field size. You should see a small number of clear peaks (4, 5, 6, sometimes 8). If you see a long tail of 1s, 2s or 99s, you have a non-runner or aggregation bug.

- **Trap bias.** Across a large dataset, trap 1 should not win, say, 60% of the time. A lopsided trap distribution usually means finishing positions are misaligned with trap numbers.
- **Track totals.** Sum the races per track per year and compare against any public totals. If you have 80% of what the source claims, you are missing data, and any per-track analysis will be biased.

Validation is not a one-time task. Run it every time you pull a new batch of data, and treat a sudden change in any of these distributions as a red flag worth investigating before trusting the new data.

Documenting every cleaning rule

This is the chapter's most important paragraph, even though it is the shortest task. Every rule you have applied so far — how you treat non-runners, which track names map to which, how grades are normalised, how missing values are handled — needs to be written down somewhere. Not in your head, not in scattered comments, but in a single document that lives next to the data.

A simple cleaning log looks like this:

Your cleaning log entries

1. Which field did you standardise, and what is the canonical form?

2. What rule decides whether a row is kept or dropped?

3. How did you handle missing values in each important column?

4. Which checks did you run to validate the cleaned output?

5. What changed since the last version, and why?

There are two practical reasons to do this. First, when something goes wrong six months from now — and something always does — the log tells you what assumptions are baked into the data. Without it, you will spend hours rediscovering rules you forgot you wrote.

Second, it makes your cleaning reproducible. If you want to rebuild the dataset from scratch, or share the workflow with someone else, the log is the spec.

> **AVOID THIS**
> Tweaking cleaning rules on the fly during backtesting. If you change how non-runners are handled in the middle of a backtest because the results look bad, you are not validating a model — you are curating the answer. Decide the rules before you start asking questions of the data, and only revisit them when the validation checks themselves flag a real problem.

A practical order of operations

If you are starting from scratch, the following sequence is a sensible default. It is not the only order, but it is the one with the fewest nasty surprises.

1. **Import raw data** into a staging table with no transformations.
2. **Run the blunt checks**: duplicates, empty traps, missing finishing positions, obviously invalid rows. Drop them.
3. **Standardise the textual fields** you know are messy: track names, grades, trainer and dog names.
4. **Cast types** properly: dates as dates, distances as numbers, times as numbers.
5. **Handle missing values** for input variables, using one of the three strategies above.
6. **Run validation checks** against known benchmarks and distributional expectations.
7. **Write to a clean table** that all downstream analysis reads from.
8. **Update the cleaning log** with anything new.

Each of those steps is small on its own. Together they turn a stream of race cards into a trustworthy dataset you can build a model on without flinching every time you look at the numbers.

The next chapter moves from the data itself to the conditions the race is run under. Once you trust your rows, the next question is which situational features deserve their own column.

CHAPTER 6

Race Conditions and Track Variables

A greyhound can post near-identical sectional times at two different venues and still finish a length apart in the final chart, simply because the geometry underneath the race was never the same. That is the whole point of this chapter. Before your model ever looks at the dog in the box, it has to look at the box the dog is in, the shape of the circuit, the surface under the claws and the conditions attached to that specific race. Treat every race as if it happened on a generic greyhound track and you are throwing away a slice of the puzzle that disciplined bettors have already monetised. Race conditions and track variables are the situational layer underneath form, and they explain a surprising amount of finishing position before a dog has even broken from the traps.

Track shape, distance and going

The first thing to capture is the physical layout of the circuit. UK and Irish greyhound tracks are not interchangeable ovals cut from the same template. Some are tight, sharp-bend circuits where leaders pocket in around the first turn and never get caught. Others are wide and galloping, with long home straights where back-runners can swallow up ground through the later stages. The same dog, dropped from a wide galloping circuit onto a tight turning track, can lose two or three places simply because the geometry of the bend no longer suits its running style. Track circumference is a tidy, single-number proxy for this whole shape question. Two contrasting circuits at the ends of the spectrum make the point well. Romford sits at the tight end at around 350 metres round, which is why it tends to favour rails-hugging, rail-leading types who can hold their line through the first bend. A wider circuit at the other end of the scale, around 437 metres round, gives dogs more time to find their stride and more space to come off the bridle in the back straight. Anywhere in between produces a different running pattern again.

350 m

Track circumference of two contrasting UK circuits, used by the source to demonstrate the spectrum from tight, sharp-bend layouts to wide, galloping ones that shape racing style.

Source: greyhoundracingbettinguk.com

You will not need the precise circumference for every track to start. A simpler three-bucket encoding is enough for most models:

- Tight circuits under roughly 380 metres round
- Medium circuits around 380 to 410 metres
- Wide, galloping circuits over 410 metres

Inside each bucket you will still have variation, but this coarse cut already helps a model separate a sharp-track specialist from a freerunning galloping type. As your stack matures you can replace the buckets with the actual measured circumference. Distance is the obvious second layer. Race distances in UK greyhound racing commonly range from about 230 metres for the shortest sprints up to roughly 900 metres for the longest staying events, with a heavy cluster between 400 and 700 metres. Treat distance as a categorical variable rather than a single continuous one. A dog with a strong 480-metre strike rate is not necessarily the same proposition at 640 metres, even though both events might be run on the same track. Encode the distance band and use it as a filter, not a smooth interpolation. Going covers the state of the surface. Most UK tracks are now sand-based rather than grass, which reduces variability compared with the old turf era, but you still see Slow, Good and Fast designations on the racecard. Going matters most when it changes abruptly between meetings, or when a dog has a body of form heavily weighted toward one type of surface. Treat going as a categorical variable too. Resist the temptation to convert it into a numeric score until you have evidence that the ordering is genuinely ordinal rather than just a set of different states.

Box draws and known trap biases

The draw is one of the most studied variables in greyhound racing for a simple reason: it has a measurable, repeatable effect on finishing position. Across the GBGB-licensed tracks, the inside trap wins roughly 18 to 19 per cent of the time, against a baseline of one-in-six, or about 16.6 per cent, that you would expect if trap number made no difference at all.

> **~18–19%**
>
> Win rate of trap 1 across GBGB-licensed tracks, sitting two to three percentage points above the random-expectation baseline of 16.6% because of inside-rail geometry at the first bend.
>
> Source: howtowingreyhoundbet.com

That two-to-three-point edge looks small in headline terms, but it is consistent and it is the kind of structural bias a quantitative system can price into its probability estimates without any clever modelling. The reason is geometry, not magic. On most UK circuits the first bend favours the inside because the dog in trap 1 has the shortest path to the rail and

the shortest distance to run around the turn. By the time the outside trap reaches the same point, the leader is already into the bend. On tighter tracks the effect is larger; on wider tracks it shrinks but does not disappear. You will want to encode the draw in a way that captures three things:

1. The raw trap number, 1 to 6, as a categorical variable.
2. A track-specific trap bias, which is the historical win rate of each trap at that venue. The 18 to 19 per cent figure for trap 1 is a national average. At any given track it could be 22 per cent or 14 per cent, and that local number is the one that belongs in your model.
3. An interaction between trap number and distance. The advantage of trap 1 is larger at sprint distances and shrinks as races get longer, because the inside dog has less time to be caught.

A common mistake is to assume trap 1 is always the most likely leader. It usually is, but not always. On some tracks trap 2 or even trap 3 leads more often. On a few tight circuits the wider traps can actually swing outward into the first bend and avoid the worst of the early crowding. Always check the venue-level data before encoding a universal trap bias into your model.

Grade, class and race type

Every greyhound race sits inside a grading system that is doing more work than it gets credit for. A1, A2, A3 and so on describe the competitive level of the race, with lower numbers meaning stronger graded fields. D-grade races are beginners and maidens; S-grade races are stakes events at the top of the tree. Different tracks use slightly different grading ladders, and some have open-age or age-restricted categories layered on top. The important point for a quantitative bettor is that grade is a proxy for the average quality of the field. A dog with a strong strike rate in A1 has been beating stronger fields than a dog with the same strike rate in A6, even if the raw numbers look identical. You have two reasonable choices:

- Encode grade as a categorical variable and let your model learn grade-specific patterns.
- Restrict your analysis to a single grade band to begin with, then widen out as you grow comfortable.

Either works. The thing to avoid is mixing grades together and treating a 25 per cent strike rate in D4 as equivalent to a 25 per cent strike rate in A2. It is not the same signal. Race type adds another layer. Standard graded races, graded finals, open races, puppy races, veteran races and category competitions all have slightly different dynamics. Stakes finals tend to produce slower sectional times because the field is more even and there is more bumping in the early stages. Puppy and maiden races are messier by nature, with dogs learning their trade, so form reads less cleanly. Encode race type as its own categorical variable and look at how your model performs within each subtype before you trust an overall number. You will also find useful variation between meeting types. Midweek cards tend to attract smaller, less competitive fields. Saturday evening cards at the bigger stadia are often deeper. If you can encode day-of-week and meeting size alongside grade, do so. These features are blunt, but they are cheap to add and they pull their weight.

Recency, lay-offs and scheduling patterns

A greyhound's recent run cycle matters, and it matters in two opposing directions. The first is fatigue. A dog that ran three times in the last fortnight is more likely to be at the wrong end of its preparation cycle than a dog that has had a fortnight off. The second is sharpness. A dog that has been off for six weeks may have lost the feel of the track and will need a run or two to find its rhythm again. Both effects are real, and they pull in opposite directions, which is why recency cannot be reduced to a single number. Treat recency as a small set of features:

- Days since last run, as a continuous variable.
- Runs in the last 14 days, as a count.
- Runs in the last 30 days, as a count.
- A flag for first run back after a lay-off of 30 days or more.

Then let your model learn the shape of the relationship, rather than imposing a rule of thumb on top of it. Some models pick up a U-shaped response where moderate freshness is best; others find a smoother decay. Either pattern is plausible. Scheduling patterns are an underrated source of edge. Greyhounds do not run every day; they rotate through cards across the week, and each track has its own rhythm of which races are competitive and which are weaker. You will see, for example, that certain Tuesday cards at smaller stadia are consistently lower-quality fields than the Friday cards at the same venue. Once you have a few months of data, you can identify these rhythms and either treat them as features or simply restrict your betting universe to the higher-quality cards.

> **31.6% to 42%**
> Range of favourite win rates across the 18 GBGB-licensed stadia in 2021 (lowest at Kinsley, highest at The Valley), illustrating how much track-to-track variation a bettor must account for.
> Source: howtowingreyhoundbet.com

That track-level variation in favourite win rates, sitting anywhere from about 31.6 per cent at the lowest venue to 42 per cent at the highest, is a useful reminder. If the favourite wins 42 per cent of the time at one track and 32 per cent at another, then a one-size-fits-all pricing model is going to misprice a meaningful chunk of races at one of those venues. Track identity, on its own, is already a meaningful explanatory variable.

Encoding these as variables systematically

Once you have your list of condition and track features, the discipline is in how you encode them. A few ground rules will save you pain later. Always use categoricals for non-ordered states. Distance, grade, trap, going and race type are not smooth scales. They are buckets. Use one-hot or target encodings rather than raw integers. Separate track identity from track features. Track identity is a single column with one label per venue. Track features, such as circumference and surface type, are separate columns that may take the same value across multiple venues. Confusing the two is a classic source of double-counting. Build a venue lookup table. For each track, store its circumference, the home bend configuration, the standard race distances it stages and the historical win rate per trap. Every race record joins to this table by venue, so that you only ever have one source of truth for the shape of each circuit. Keep the date. If a track resurfaced, or moved from grass to sand, or altered its first bend in 2019, the historical data is from two different eras. A simple track-active-from date lets you decide how to handle that.

Encoding race conditions cleanly

- [ ] Treat distance, grade, going, trap and race type as categorical variables, not continuous ones
- [ ] Store a venue lookup table with one row per track and join it to every race record
- [ ] Replace national-average trap win rates with track-specific trap win rates once you have enough data
- [ ] Add recency as a small set of features rather than a single days-since-last-run number
- [ ] Capture day-of-week and meeting size alongside grade to surface quiet cards
- [ ] Drop races with missing venue, distance, trap or going rather than imputing them

Finally, build a small set of validation rules for these features specifically. Every race record should have a non-null venue, a non-null distance within the valid range for that venue, a trap number between 1 and 6 and a going value drawn from the allowed list. Missing or out-of-range values in this layer are almost always worth dropping rather than imputing, because the race itself is the unit you are pricing, and a broken unit is worse than a slightly smaller dataset. The temptation at this stage is to over-engineer. You do not need a hundred track-level features. A dozen well-chosen ones, kept clean, will explain more variation than fifty messy ones. Build the small set first, watch how it performs against your ratings, and only add new features when the data genuinely asks for them. A useful exercise before you move on is to look at your own shortlist of tracks and answer a few questions about each. You will spot patterns that the national averages hide.

Your local tracks

1. Which of your regular tracks feels tightest, and which feels the most galloping? Does the racing style on those cards match what the circumference suggests?

2. For each track, which trap wins most often at sprint distances, and which trap wins most often at staying distances?

3. Are there specific cards during the week, such as Tuesday or Thursday, where favourite win rates or field sizes look noticeably different from the rest of the week?

4. How often does the going change between meetings at each venue, and which dogs in your database seem most affected by surface changes?

5. Are there any tracks where grade labels behave differently, for instance where A3 at one venue feels like A5 at another?

Why the conditions layer matters at the pricing level

The whole point of this chapter is that a dog's chance of winning is not a fixed property of the dog. It is the dog's ability multiplied by the conditions it is asked to express that ability in. If your model treats the conditions as constant, then it has already taken a stance that the inside trap does not matter, the grade is irrelevant and the track shape is

interchangeable. None of those are true. When you sit down to price a race, the first pass should always be to ask what kind of race this is, at what venue, over what distance, on what surface, with what trap draw, before you look at the dogs themselves. If you cannot answer those questions cleanly, you are not yet ready to ask the dog-level questions that come next. The good news is that this layer is the most stable, the most repeatable and the least likely to lie to you. Track shape and distance do not change between meetings. Trap bias is structural. Grade is set in advance. Once you have encoded these features properly, they will quietly improve every model you build on top of them, without ever needing to be clever. This is general guidance for anyone building a quantitative greyhound betting system, not a promise of results. Past performance and venue averages do not guarantee future outcomes, and any system that uses them should be tested and validated on your own data before you risk money on it. Treat this layer as the foundation, not a finished answer.

KEY TAKEAWAYS

- Race conditions and track variables are a separate feature layer underneath dog form, and they explain real finishing-position variation on their own
- Encode distance, grade, going, trap and race type as categorical variables joined to a venue lookup table, not as raw numbers
- Use track-specific trap win rates rather than the national average, because local geometry can swing trap 1's strike rate by several points
- Capture recency as a small set of features so that fatigue and sharpness can be modelled separately
- A clean, small set of condition features will do more work than a large, messy one

With that foundation in place, the next step is to turn from the race itself to the dogs in it, and start asking which variables about a greyhound's past performance are worth trusting and which ones quietly leak future information into your model.

CHAPTER 7

Dog Form and Performance Variables

A dog crosses the line first and your instinct says "good dog". A greyhound finishes fourth by three lengths and your instinct says "off day". Both reactions might be true, but neither tells you how to price the next race that dog runs in. That's the work of this chapter: turning what a greyhound has actually done into features a model can use, without borrowing any information from the future.

The leakage trap — and why it matters here more than anywhere

Before a single feature gets built, you have to fix the biggest hazard in any quantitative dog-form work: leakage. Leakage is what happens when a feature uses information that wouldn't have been available at the moment you'd have placed your bet. The model looks brilliant in backtests and falls apart on race day, and you cannot understand why. Dog-level features are the most common place for leakage to sneak in, for three reasons. First, greyhound form is dense and full of compound fields — "won last time at the track in similar grade" feels safe but can quietly include future knowledge. Second, some publishers of form data include post-race adjustments or late scratches that change the result after the fact. Third, it's tempting to include a feature like "winning distance last race" without realising the published figure was calculated with hindsight once the rest of the field had finished.

> **AVOID THIS**
> Including any field derived from the full finishing order of a race when you're computing features as if you'd bet before that race. If the field couldn't have known the result, neither can your feature.

The rule, repeated here because it's worth repeating: every dog feature you build must be computable from information that was available at the off, in the same race you're trying to model.

Recent finishes, positions and beaten margins

The most natural starting point is recent finishing positions, but "natural" is not the same as "useful". A dog finishing second last time tells you less than you think, unless you know what was in front of it, how far behind it was, and at what track and grade. Strip those out

and you have a number that's barely better than a coin flip. What you actually want from recent form is a small set of features that capture *shape*, not just outcome. Good candidates include:

- **Position in last N races** — but with N small (3 or 4), because greyhounds race often enough for older results to fade quickly.
- **Beaten margin in lengths** for the last run or two, ideally from sectional timing rather than visual description, because length margins are noisy in tight finishes.
- **Whether the dog finished in the first three** as a binary flag over a recent window — coarser than position but more stable.
- **Finish behind the winner**, expressed as a margin, separately for races where the dog led at the first bend versus those it didn't. The early-lead pattern is a real signal in many tracks.

A useful discipline here is to track the number of *runs* you've used. If a dog has had only two starts back from a long break, your "last three" features will be quietly leaning on information that's years old. Either exclude the dog from those features until sample size is back, or weight recent runs more heavily than distant ones. Be careful, too, with finishing position data from aggregated form sites. Some combine maiden, graded and open-race results without telling you, so a dog's "recent form" might be a mixture of fields at a small provincial track and feature races at a major stadium. Keep the source consistent across all rows, or you'll be comparing chalk and cheese.

Time-based measures and sectional data

Positions tell you who crossed the line first. Times tell you how fast the dog actually went. Where you can get them, times are usually worth more than positions, because a dog can finish fourth in a fast race and still have run a personal best, while a slow win is more often a function of the field than the dog. The standard time-based features worth building:

- **Best time over course and distance** — track and distance specific, because greyhound times are heavily influenced by both.
- **Last race time**, normalised to the race's standard if you can compute it. Many form providers publish an "expected time" or "standard time" per race; the difference between actual and standard is your normalised feature.
- **Sectional times** if your source provides them: the time to the first bend, and to the third bend. These are particularly valuable because they disentangle early speed from

finishing speed.

- **Timeform-style speed figures** if you have access to a feed that publishes them. Treat them as an external rating, not as ground truth, and remember they were computed at the time with whatever rules that publisher used.

If you don't have sectional times, don't despair. The difference between best time and standard time at the same course and distance is usually enough to start with, and it costs nothing extra to compute. One quiet issue with times: the clock is human. Some tracks started and stopped it manually in older data, and there are variations between sources for the same race. If you're combining feeds, normalise or pick a single source for any feature involving time, otherwise you'll be modelling the source rather than the dog.

Career stage, age and experience

Greyhounds are not mature athletes on a slow curve. They peak, hold for a stretch, and decline — and the shape of that arc varies by individual. Two pieces of dog-level context are worth capturing explicitly:

- **Age in months**, or age in days since the dog's first recorded race. Both work; days-since-debut tends to be slightly more informative for very young dogs because they enter the racing population at different ages.
- **Number of career races**, sometimes called experience. A dog on its fifth career start is in a very different place from one on its fiftieth, even if they're the same age.
- **Days since last race**, which captures both layoff (often a positive sign after a break, sometimes a negative one if the dog is older) and the freshness angle for short-notice runners.
- **Races at this specific track**, separately from races at this distance. Some greyhounds are clearly course-lovers or course-failures; the feature is simple but useful.

There's a real temptation, when you have age, to encode it as a polynomial or binning — "young / prime / veteran". Resist the urge to start there. Let your model learn the shape from the raw age feature first, and only bin if a consistent non-linearity shows up across many dogs. Hard bins bake in your assumption before the data has spoken. Also: experience interacts with grade. A dog with five career starts running in open grade is a different proposition from the same dog in A1, even at the same track. Don't grade-strip these features yourself; let the race-level grade variable (which you'll handle in a different chapter) do that work alongside the dog-level ones.

Trainer and kennel signals — handle with care

It would be strange to leave the trainer out of a dog-form feature set. UK and Irish greyhound racing is dominated by a relatively small number of trainers and kennels, and the differences between them are real. A first-time-out dog from a top kennel can carry useful information about preparation and schooling that you can't see in the form. But trainer features are also where the most subtle leakage and overfitting happens. A few cautions:

- **A trainer's overall strike rate** is too blunt. A big kennel will have thousands of runs and the strike rate barely moves quarter to quarter; a small trainer's strike rate can swing wildly from month to month on a few winners.
- **Trainer × track** and **trainer × grade** are more useful than a single overall figure, but each split reduces your sample size. Don't split further than your data can support.
- **Recent trainer form** (last 30 or 60 days, same track) often beats career trainer form, because it picks up short-term signals like a kennel hitting form or going cold.
- **Watch for newly-licensed trainers and aliases**. Some big operations appear under multiple licence names across regions, which can split their record and make them look worse than they are.

The honest framing of trainer features is that they're a *correction* to the dog's own form, not a substitute for it. Use them to nudge, not to override.

Building your dog form feature set

- [ ] Decide which N (last 3 or 4 races) you'll use, and document it so it's fixed before any backtest.
- [ ] Pick one time-source for any timing feature, and don't mix sources within a single feature.
- [ ] Capture beaten margin separately from finishing position, so the model can use them independently.
- [ ] Add age (days since debut) and experience (career races) as two distinct features, not one combined figure.
- [ ] Record days-since-last-race as its own column — don't fold it into age.
- [ ] Build trainer features at track×grade splits, not as a single global rate.
- [ ] For every feature, write down the exact computation as you'd do it on race day, with no future data allowed.

Your dog form variables — what to include?

1. Which recent-run window length (N) will you use, and why?
2. Do you have access to sectional times? If not, what's your fallback for early-vs-finishing speed?
3. How will you handle a dog returning from a long break in your "last N" features?
4. Will you use age in months, days since debut, or both — and at what granularity?
5. What's your plan for trainer×track×grade splits, given your sample size?
6. How will you spot and reject features that quietly include future information?

Putting the pieces together

A reasonable starter set of dog-level features for greyhound racing looks something like this — pick a handful, not all, and let the modelling stage tell you which earn their keep.

Starter dog form features

Feature	What it captures
Average finishing position, last 3 runs	Recent competitive level
Beaten margin last run (lengths)	How the run actually went
Best time minus standard time, course and distance	Raw ability adjusted for conditions
Sectional to first bend, last run	Early pace and break
Age in months	Career stage
Career races to date	Experience
Days since last race	Freshness and layoff signal
Trainer win rate, this track, last 60 days	Recent kennel form at venue

Notice the absence of any feature that says "winning distance" or "won by" — those are computed from the finishing order of the field, and a single race's winner is unknowable before it runs. Stick to features you could have computed at the off, and you'll save your future self a long debugging session. One last thought before the chapter closes. Trap draw matters in greyhound racing, and trap advantages at compact tracks are real — the inside traps win more often than chance would predict, and the outside traps win less.

20–22%

Approximate win percentage for trap 1 at compact UK greyhound tracks over sprint distances, illustrating the inside-trap advantage (vs. a random expectation of 16.7%).

Source: Crayford Greyhound (form guide)

Trap is a race-level variable though, not a dog one, and it sits alongside the form features rather than inside them. Treat dog form as what the dog brings; treat race conditions as what the race sets up; treat the market, which you'll meet in the next chapter, as what other people think it's all worth.

KEY TAKEAWAYS

- Every dog feature must be computable from information that existed at the off — leakage is the single biggest hazard in dog-level work.
- Positions are useful, but beaten margins and time-based features usually carry more signal per row.
- Age and experience are two distinct features, and you should let the data show you their shape rather than pre-binning them.
- Trainer features are a correction, not a substitute for the dog's own form — keep them recent and split by track and grade.
- Pick a single time source per timing feature and stick with it, or you'll be modelling the data feed rather than the dog.

The dog-level features you've built here are one of three inputs your eventual model will weigh. The second is the race itself: track, distance, grade, trap draw, going. The third is what the betting market is saying the dog is worth. Chapter 8 is where you start looking at the market properly, as both an input and a benchmark for the numbers you've just built.

CHAPTER 8

Working With Market Odds

Every greyhound race is a small auction and the odds are the bids. Read them well and you have a free forecast handed to you in real time. Read them poorly and you will either chase prices that have already moved, or back dogs the smart money has quietly stepped away from. This is not a chapter about finding tips hidden in bookmaker prices. It is about treating the market as a noisy probability machine you can both learn from and argue with.



What the Market Is Actually Telling You

Decimal odds are not a price tag. They are a probability statement dressed up as a number. Take a dog quoted at 4.00: that means an implied chance of one in four, or 25%. A dog at 2.50 implies 40%. A dog at 10.00 implies 10%. The maths is simple: **implied probability equals one divided by decimal odds**. Once you can do that conversion in your head, you stop seeing odds as a price and start seeing them as a forecast.

Convert every runner in a race and add the probabilities together. You almost never reach 100%. You usually land somewhere between 105% and 130%. The gap above 100% is the *overround*, and it is the bookmaker's built-in margin, the cut they take before any winners are paid out. That margin is the cost you pay for the convenience of a fixed-price product, and it is the reason the favourite pays less than its true chance of winning.

125%

Typical overround (bookmaker's built-in margin) in a standard six-runner greyhound race, equivalent to a ~25% bookmaker edge.

Source: How to Win Greyhound Betting

A 125% overround on a six-runner race is roughly the norm in the UK and Ireland. That 25% gap is not profit for the bookmaker on every race, because favourites do win sometimes and outsiders do get backed. It is the long-run cut that funds the whole operation. When you understand that, two things change about how you read a card. First, you stop treating odds as a measure of how strong a dog is. They are a measure of how the market thinks the dog will do, plus margin. Second, you start looking for races where your own probability estimate is meaningfully different from the implied one, because that gap is where your edge lives.

A useful exercise before you do anything else: pick a race card, convert every runner to implied probability, and add them up. The total tells you the overround for that specific race. Anything north of 115% on a six-runner graded race is normal. Anything approaching 100% is suspicious in the other direction, and usually means you have looked at an exchange price rather than a bookmaker price. The overround is not just a number to know. It is a number to track, because it tells you how generous the day is being to your betting budget.

Bookmakers Versus Exchanges: Two Markets, Two Voices

The price on a bookmaker's site and the price on a betting exchange are not the same animal, even when they show the same number. A bookmaker price is set by a trader who manages a book and tries to balance liability on every outcome. An exchange price is the live clearing point between punters willing to back a dog and other punters willing to lay the same dog. They are formed differently, they move differently, and they are worth different things to your model.

How big is the UK greyhound market you are trying to read?

> **£1.5bn** UK greyhound betting turnover in 2022-23, down 23% in real terms over three years.
> Source: How to Win Greyhound Betting

That £1.5bn figure is the total amount staked in 2022-23, and it has been shrinking in real terms for a few years. Less money in the pool generally means slightly less efficient prices, which is good news for anyone willing to do their own thinking. It also means you cannot assume the market is always sharp, especially in lower-grade meetings or midweek cards where liquidity is thin. A 20% favourite on a busy Saturday graded final is a much more reliable number than a 20% favourite in a B2 over 480 metres on a wet Tuesday at a small track.

Bookmaker prices versus exchange prices

Aspect	Bookmaker price	Exchange price
Who sets it	A trader balancing the book	Backers and layers meeting on a platform
Margin	Baked into every price, often 5 to 25% in total	Commission only, typically 2 to 5% on winning bets
Movement	Triggers when imbalance gets large	Live, with every matched bet
Strength	Stable, easy to take, available everywhere	Often closer to true probability on busy races
Weakness	Margin inflates favourites and deflates outsiders	Thin on small fields and quiet meetings
Best use	Taking a price before the off	Benchmarking and reading live signals

For most of the work in this book, your benchmark should be the exchange price taken close to the off, when liquidity is at its highest. Bookmaker prices are useful as the actual input you stake at, but use the exchange as your probability reference, and remember to subtract a sensible commission factor when you convert exchange odds back into an implied probability. The two numbers will look similar on big races and quite different on small ones, and that difference is itself a piece of information.

Tracking Movement and What It Means

Odds move for reasons, and the reasons are themselves data. A dog that opens at 5.00 and drifts to 7.00 in the hour before the off is being quietly abandoned. A dog that opens at 5.00 and shortens to 3.50 is being backed. Both of those pieces of information deserve a place in your model, but neither deserves to be obeyed without question.

The honest way to think about market moves is as signals of informed money, with three caveats attached.

- **Time horizon matters.** A move from 5.00 to 4.50 over six hours before the off is far less informative than a move from 3.00 to 2.20 in the final ten minutes. Long, slow drifts often reflect balancing, not insight.
- **Volume matters.** A 10% shortening on a race with £2,000 matched is a different thing from a 10% shortening on a race with £80,000 matched. Always check matched volume before reading anything into a move.
- **Liability matters.** A trader at a bookmaker shortening 6.0 to 5.0 because they have taken too much at the bigger price is not giving you a tip. They are protecting margin. Treat bookmaker moves with more scepticism than exchange moves.

A reasonable rule of thumb: log the opening price, the price one hour before the off, the price ten minutes before the off, and the starting price. The shape of that trail tells you more than any single number. A dog that shortens steadily through all three checkpoints is a different proposition from one that crashes late because a syndicate's money has just hit the pool. A dog that drifts on every checkpoint is in genuine trouble, and a dog that opens long and contracts late is sometimes the most informative move of all.

One more thing worth noting. Steam moves, where many dogs in the same card shorten together, often signal that a well-funded operation is racing multiple entries across the night. They are signals, but they are signals about someone else's plan, not necessarily about which dogs are fastest. Use them as a guardrail on your own overconfidence rather than as direct input to a bet.

The Favourite Fallacy

If you want a single statistic that should change how you read a betting market, it is this one.

A 33% win rate for favourites in graded UK greyhound races means the market favourite loses roughly two races out of three. People who back favourites because they feel safe are paying for a feeling, not for accuracy. The market knows favourites do not win often. That is why the favourite's odds are bigger than one would expect from a true probability standpoint. The odds already reflect the fact that the most likely winner still loses more often than it wins.

What that 33% hides is the spread. The favourite at a strong, predictable track like Crayford or Nottingham will win closer to 40% of the time. The favourite at a more chaotic track or in a lower grade will win closer to 31%. If you build your own ratings, you should be able to predict, for each race, whether the favourite's true win rate is closer to the top or the bottom of that range. That distinction is where your edge lives, not in the fact that the favourite wins about a third of the time in general.

The same logic applies to second favourites, third favourites and the long shots at the back of the card. The market consistently under-prices the favourite relative to the true risk of being overturned by the field, and it consistently over-prices the rank outsider because punters love a long shot. If your model can identify the races where the favourite is actually weaker than the odds suggest, or the races where an outsider has a hidden shot, the gap between your estimate and the market's estimate starts to look like a job description.

When It Makes Sense to Disagree With the Market

This is the heart of the chapter. A quantitative model is not worth building if it just agrees with the market and lags it by a few seconds. The value comes from the moments when your probability estimate and the market's implied probability disagree by enough to overcome the margin and the variance.

For a six-runner race with a 125% overround, every dog has to be discounted by roughly 4 to 5% just to break even with the bookmaker. If your model says dog A should be 25% and the bookmaker has him at 20% implied, you might think you have an edge, but after the margin you are essentially at break-even, and you have not even accounted for the commission you will pay or the variance you will face. The honest threshold for betting is bigger than most beginners assume, and the discipline of writing it down before you look at the prices is one of the most useful habits a quantitative bettor can build.

> **TRY THIS**
> Before you place your next bet, write down your own probability estimate, your minimum edge threshold, and the overround of the race. Then check the market. If the price still clears all three, take it. If it does not, walk away and log the race anyway.

> **AVOID THIS**
> Being contrarian for the sake of it. The market is wrong often enough to matter, but it is right more often than you would like. Disagree with specific numbers for specific reasons, not as a habit.

There are three situations where disagreement is reasonable, and one where it usually is not.

1. **The market is pricing in crowd behaviour you can see past.** A trainer whose kennel has been on a cold run often gets an extra length or two added to the price, even when form numbers say the dog should be competitive. If your data disagrees with the narrative, you have a candidate.
2. **The market is slow to react to new information.** Track conditions changed in the last hour, a key runner has been scratched in a connected race, the betting forecast says something your form numbers do not. Any of these can move your estimate faster than the market moves its price, especially in smaller meetings.
3. **You have a sample the market has not weighted properly.** A dog returning from a long break, a trap change, a first run at a new venue. If you have enough history to

make a calibrated estimate and the market is using an out-of-date one, the price is likely to drift towards you once it catches up.

The fourth situation, the one to avoid, is disagreeing with the market because you have a strong feeling. Feelings are not data. They might be right for a reason your model has not captured, but unless you can write that reason down as a variable, it does not belong in a quantitative system. If you cannot turn your hunch into a number, log it as a hunch and move on. Sometimes hunches are right. They are just not measurable, and this book is about measurement.

Plumbing Market Odds Into Your System

By the time you have collected enough races, you will have several ways to use the market in your model. The simplest is as a baseline: assume the market is right, and only bet when your model is sufficiently far from it. The more ambitious approach uses the market itself as a feature, feeding the implied probability into a model alongside form, track and conditions, and letting the model learn how to blend them. Either way, the market is no longer a mysterious force. It is a column in your database, with a timestamp and a source.

Whichever approach you take, three habits will save you from a lot of self-deception.

Reading the market before you bet

- [ ] Convert every runner's odds to implied probability and add them to confirm the overround
- [ ] Note the opening price, the price one hour out, and the price ten minutes out
- [ ] Check matched volume on the exchange for the race
- [ ] Compare the bookmaker starting price to the exchange starting price for material drift
- [ ] Log your own probability estimate before you look at the market again
- [ ] Only bet where the gap between your estimate and the market price clears your edge threshold

First, always log your estimate before you look at the market. Once you have seen the price, you cannot un-see it. Your brain will quietly bend your number towards the price you just read, and you will call it analysis. Second, compare your model's output to the market's implied probability rather than to other bookmakers' prices. The exchange is your cleanest benchmark, especially in the final minutes before the off. Third, treat the market as a feature that decays. A price taken thirty seconds before the off is worth far more than a price taken eight hours earlier, and your model should weight it accordingly.

The market is rarely wrong by a lot, and it is wrong in roughly the same way across thousands of races. That is what makes it useful as a benchmark. Your job is not to outsmart it on every race. Your job is to find the small, repeatable slice of races where your structured data says something the market has not fully priced in yet. That is the entire quantitative game in one paragraph.

KEY TAKEAWAYS

- Decimal odds are probability statements with a margin baked in: implied probability is one divided by the odds
- Bookmaker prices carry a built-in overround, often around 25% on a six-runner race, so the favourite still only wins roughly a third of the time
- The exchange is usually a cleaner benchmark than any single bookmaker, especially close to the off when liquidity is high
- Market moves are signals, not gospel: check the volume, the timing and whether you are looking at a trader balancing a book or real money flowing
- Disagree with the market only when you can name the reason in data, and only when the gap clears the overround plus your minimum edge

Once you can read a market reliably, the next step is turning your own estimates of probability into something a model can learn from. That means sitting down with the basic probability ideas every greyhound bettor needs before going near a regression or a staking plan, which is exactly what the next chapter is about.

CHAPTER 9

Probability for Greyhound Bettors

Probability is the language that betting actually speaks, even when nobody on the terraces is using it.



Over the last few chapters we have talked about data, APIs, cleaning, encoding variables and reading market odds. Now we need the math that ties all of it together. This chapter is the hinge between collecting information and using it to make decisions. If you have ever wondered why a "good" system can still lose for three weeks, or why backing every favourite does not print money, the answer lives in probability. Not in tips, not in form, not in the eye of a pundit - in probability.

Base Rates: Your Anchor to Reality

A **base rate** is the starting probability of something happening before you look at any special information. In greyhound racing, the simplest base rate is the theoretical 16.6% chance of any one trap winning a six-runner race. That is not opinion; it is arithmetic. It is the floor of expectation.

Real races drift from that floor. Trap 1 in UK greyhound races wins roughly 18% to 19% of the time, not 16.6%.

That looks small. It is not. It is exactly the kind of measurable gap that quantitative systems are built to find. It also tells you something useful: track geometry matters, draw matters, and the naive one-in-six guess is wrong.

The bigger anchor is the favourite. Across graded UK greyhound races, the favourite wins about a third of the time.

Read that again. The favourite - the dog the market has collectively decided is most likely to win - crosses the line first roughly *one race in three*. If you spent a year backing every favourite at the prices the market offered, you would lose money. The reason is the next big idea we need to introduce: the overround, which we will come to shortly.

Base rates matter because they tell you what "normal" looks like. Without them, every losing bet feels like a surprise and every winner feels like proof your system works. With them, you have a yardstick. You can say: "My model rated this dog at 22%. The base rate for this trap-draw combination is 17%. So I am seeing a real edge of 5 percentage points, before I even look at price." That single sentence is most of the game.

Sample Sizes and Independent Events

Greyhound races are short, frequent, and **independent**. Each new race forgets the last one. The trap 4 dog that finished last on Tuesday is not "due" a win on Friday. The dog that bolted past the field at Monmore last week is not "hot" in any repeatable statistical sense - unless the underlying factors (track, grade, fitness, draw) still apply, in which case it is those factors, not the recent result, doing the work.

This matters when you are building variables. A dog's last three races give you roughly three data points. Three data points is not enough to reliably distinguish a 15% chance dog from a 25% chance dog. The noise is bigger than the signal at that scale. To get a stable read on a dog's true ability you typically want dozens of past runs - and even then you need to adjust for the grade it was running in, the track it was running at, and the quality of the opposition.

Stated side by side, the three probability anchors we care about most look like this:

If greyhound outcomes were perfectly random, every bar would sit at 16.6%. The fact that trap 1 sits slightly above that line and favourites sit well above it is the headroom your model is searching for. It is also the headroom the bookmaker's margin eats into, which we will get to.

The independence point has another practical consequence: there is no such thing as a "trend" in the strict sense. There are *streaks*, which we will come to, but they are not momentum. They are variance. Treating them as momentum is the most expensive

mistake a beginner can make, because it leads to chasing losses, doubling stakes on "due" winners, and abandoning systems that were actually working.

Expected Value: The Only Number That Matters Long-Term

Expected value (often shortened to EV) is the average outcome of a bet if you could place it an infinite number of times at the same price. Suppose a dog has a 25% chance of winning and you are taking 4/1 (decimal odds 5.0, so you receive 4 units profit per 1 unit staked). The expected value works out as:

EV = (0.25 × 4) − (0.75 × 1) = 1.00 − 0.75 = +0.25 units per 1 unit staked.

That bet is profitable in the long run, even though it will lose three times out of four. Most people hear "loses three times out of four" and walk away. A quantitative bettor hears "+0.25 EV" and pulls out the staking plan.

This is the first big mental shift of the chapter. **Winners are not the goal. Positive expected value is the goal.** A betting system that wins 30% of the time at prices with average EV of +0.10 is more profitable than one that wins 45% of the time with negative EV. Winners feel good; EV pays the bills.

Working out expected value on a single bet

- [ ] Estimate the dog's true win probability (p) from your model or base rates.
- [ ] Note the decimal odds offered (d) at the price you are actually taking.
- [ ] Compute EV = (p × (d − 1)) − ((1 − p) × 1).
- [ ] Place the bet only when EV is meaningfully positive and the edge exceeds the overround.

Two more things worth knowing. First, EV is an *average*, not a promise. A +0.25 EV bet will lose more often than it wins if the price is right - and that is normal. Second, EV is only meaningful relative to the price you actually take. If your model says 25% but the best price available is 3/1 (decimal 4.0), the EV drops to zero. Price is half the bet. Your probability is the other half.

Variance, Streaks and the Gambler's Fallacy

Even a perfectly positive EV bet will lose sometimes. It will lose in clusters. This is not a bug; it is mathematics.

Suppose your model identifies bets that win 30% of the time at prices that imply 25%. The EV per bet is positive. But over 100 bets, the number of winners will not be exactly 30. It will be somewhere around 30, give or take the square root of (100 × 0.30 × 0.70), which is about 4.6. So you might win 25 times, or 35 times, or anywhere in a fairly wide band - and that is before any real-world wobble (non-runners, track bias, late odds moves, data errors) adds further noise.

The **gambler's fallacy** is the belief that past losses make future wins more likely. They do not. A fair coin that has come up heads five times in a row is still 50/50 on the next toss. So is your greyhound system after five losers in a row. The mistake is not recognising this; the mistake is letting it change your staking or your selections. Once you start "going with the flow" or "backing what's overdue", you have left the world of probability and re-entered the world of superstition.

Streaks will happen. Long losing runs of 10, 15, even 20 bets are entirely possible with a sound system. If your staking is sized to survive them, you keep going. If it is not, you go bust or, worse, you abandon a working system right before it recovers. That is why a staking plan is not optional - it is part of the model.

To see the price side of the same problem, consider the typical greyhound market. The bookmaker's prices for a six-runner graded race sum to about 125%, not 100%.

That extra 25% is the **overround** - the bookmaker's margin. For every £1 wagered across all six dogs, only about 80p is returned to bettors in aggregate. To break even, your collective selections need an average true probability meaningfully higher than what the market implies. That is a tall order, and it is exactly why a quantitative edge has to be both real and properly sized before you can expect to come out ahead.

From Probability to Price: Setting Your Threshold

Now we connect the dots. The practical output of any probability model is a number between 0 and 1: the chance this dog wins. To turn it into a betting decision, you need a price threshold.

The basic rule looks like this:

1. Estimate the dog's true win probability, *p*.
2. Convert the available price to implied probability: 1 ÷ decimal odds.
3. Compare: if *p* is meaningfully greater than 1 ÷ odds, you have an edge.
4. Decide how big that edge needs to be to justify a bet.

How big is "meaningfully greater"? That is the judgement call, and it is where the overround comes in. Because the market is already inflated by roughly 25% in a typical greyhound race, a small theoretical edge can disappear into the margin. If your model says a dog has a 22% chance and the market is implying 20%, you do not actually have a 2% edge in real terms - you have a 2% edge against an already-inflated benchmark, and once commission, errors and bad luck are factored in, you are probably still losing.

A common discipline is to require a minimum edge of, say, 5 percentage points over the implied probability before placing a bet. The exact number depends on your model's accuracy, your staking plan and how much variance you can stomach - but the principle is fixed: **do not bet unless your edge is bigger than the noise you know is in your own numbers**.

A few practical pointers to keep in mind when turning probabilities into prices:



- **Implied probability is your enemy as well as your friend.** It is the cleanest number you can compare your model to, but it already contains the bookmaker's margin, so any edge must be measured against it, not added on top of it.
- **The shorter the price, the smaller the percentage-point edge you need to clear** - but the more accurate your probability has to be. A 1.5/1 shot you think is 50% (implied 40%) gives you a 10-point edge; a 10/1 shot you think is 12% (implied about 9%) gives you a 3-point edge. Both can be value, but the 10/1 bet tolerates far less error in your own estimate.
- **If you cannot explain the edge, you probably do not have one.** If your model says 28% and the market says 20% and you cannot point to a feature (a recent sectional, a grade drop, a draw advantage) that drives the gap, the gap is most likely noise.
- **Round numbers are useful, but round them honestly.** A 22% chance is not 25%. Do not inflate your own probabilities to chase a threshold.

A useful exercise is to write, for each bet you actually place, three numbers on the same line: your model's probability, the market's implied probability, and the edge in percentage points. If the edge is below your threshold, skip the race. If it is above, stake according to your plan. Over a month, this single discipline does more for your betting than any fancy model upgrade.

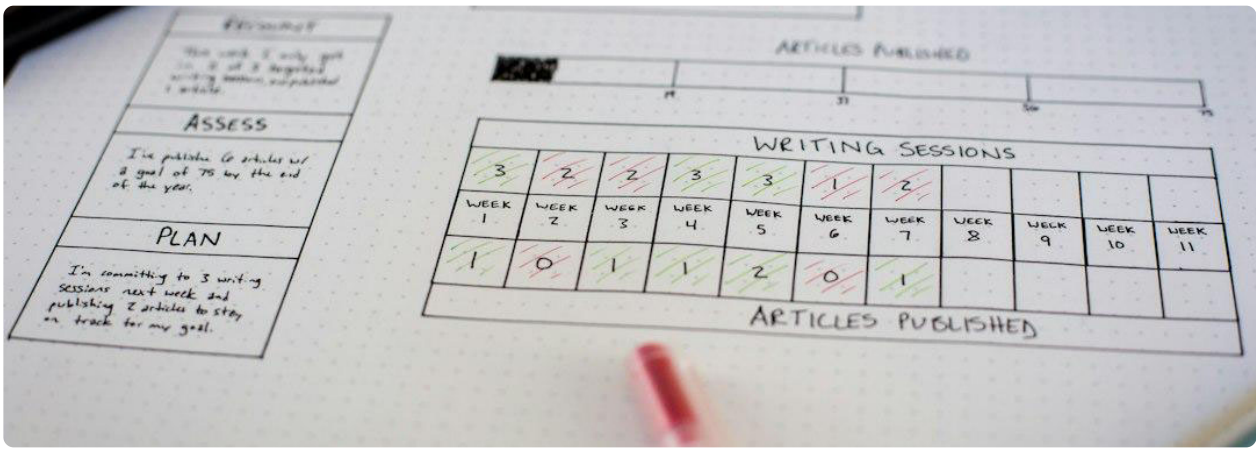
That, really, is what probability is for. It is not an abstract subject you studied once; it is the working language of every bet you place. Once it sits in your head as a tool rather than a theory, the next step is to give it somewhere to live: a rating system you can actually explain to yourself six months from now.

KEY TAKEAWAYS

- Base rates are your anchor: around 16.6% theoretical per trap, around 18% for trap 1, around 33% for favourites.
- Independence matters: every race starts fresh, and short winning or losing streaks are not momentum.
- Expected value, not strike rate, is what makes a betting system work over time.
- Variance is real; size your stakes to survive losing runs of 15 to 20 bets even with a sound model.
- Only bet when your probability beats the market's implied probability by more than the overround and your own estimation error.

CHAPTER 10

Building Your First Rating System



A rating system you cannot explain to yourself six months from now is a rating system you will quietly abandon. The temptation, when you finally sit down to build one, is to throw every variable you have read about into a spreadsheet, weight them by feel, and hope the rankings make sense. Sometimes they will. More often you will stare at the output, decide it "looks wrong", and start tweaking without any clear idea of why. That is how a quantitative approach turns into a vibes-based one without you noticing.

The goal of this chapter is deliberately modest. You are going to build a small, transparent rating that takes a handful of sensible variables, combines them in a way you can write down on one page, and produces a ranked list of runners for any given race. Nothing fancy. Nothing you would not feel comfortable defending if a fellow bettor asked, "why did that dog come out top?"

Start With a Variable Shortlist, Not a Spreadsheet of Dreams

The first decision is the most important one. Before you write a formula, decide which inputs your model is actually allowed to look at. The right starting size for a first rating is somewhere between four and eight variables. Fewer than that and you are probably ignoring information that matters. More than that and you are building a system you will struggle to interpret, which is exactly the trap you are trying to avoid.

A useful way to think about variables is to sort them into three loose buckets: **trap variables** (early race position), **form variables** (recent finishing positions, consistency), and **context variables** (track, distance, grade, going). The exact split will depend on what

data you have cleaned and structured in earlier chapters, but the principle is the same: mix a couple of inputs from each bucket so that no single dimension dominates.

One variable that consistently earns its place in a first rating is early pace, because in short-field sprint races the dog leading at the first bend has an enormous advantage. Research on tight UK sprint tracks suggests that around

> **60%** The share of sprint races at tight UK tracks (Crayford, Romford, Sunderland) won by the dog leading at the first bend — identified as the single strongest predictive factor to weight in a rating model.
> Source: Greyhound Racing Betting UK

of races at venues like Crayford, Romford and Sunderland are won by the bend leader. That is not a marginal effect; it is the single strongest practical signal you can lean on. If your data lets you record first-bend position from historical races, it deserves a meaningful weight.

A sensible shortlist for a beginner system might look like:

- **Recent finishing positions** over the last three to five races, with a recency bias.
- **Best recent time** at the distance, or a personal best at the track.
- **First-bend position** in past races at the same track or similar trap draw.
- **Grade** the dog is currently racing in, with adjustments for step-ups or step-downs.
- **Trainer or kennel strike-rate** at the venue, only if your data captures it cleanly.
- **Days since last race**, to flag fatigue or freshening.

That is six variables. It is enough to feel substantial and small enough that you can hold the whole model in your head. Resist the urge to add anything else until you have watched this version run for a while and learned what it does well and where it falls over.

Turn Raw Numbers Into Comparable Scores

Raw inputs do not combine well. A "best time of 28.40 seconds" and a "trainer strike-rate of 18%" live on completely different scales, and if you add them together you are really just adding noise. So the second step is to normalise every variable into a score on a common scale, typically 0 to 100, where higher is better.

There are a few simple ways to do this, and you do not need anything more complicated than a spreadsheet column.

The simplest is a **linear rescale**. Take all the values for that variable across the field for today's race, find the minimum and maximum, and map each runner's value onto the 0 to 100 range using the formula *(value − min) / (max − min) × 100*. This is easy to explain and works well when the values are reasonably well-behaved.

A second option is **rank-based scoring**. Instead of using the raw value, you rank the runners from worst to best on that variable and assign scores like 0, 20, 40, 60, 80, 100. This is robust against outliers and very intuitive: "this dog was third best for early pace, so it gets 40 points". The downside is that the gap between, say, second and third is treated the same as the gap between fifth and sixth, which is sometimes unrealistic.

A third option is **z-score normalisation**, where you express each value as how many standard deviations it sits from the mean. This is statistically tidy but harder to read on a scorecard, and is best left for later iterations.

Three normalisation approaches at a glance

Approach	How it works	Strength	Watch out for
Linear rescale	Maps raw value between min and max onto 0–100	Keeps real gaps between runners	Sensitive to one extreme value
Rank-based	Assigns 0–100 by position in the field	Robust, easy to explain	Loses magnitude of differences
Z-score	Expresses distance from the mean in standard deviations	Statistically clean	Hard to read on a scorecard

For a first model, rank-based scoring is usually the friendliest. You can read every line of the output and understand exactly where each runner sits.

> **GOOD TO KNOW**
> Whatever scale you choose, apply it *within the race* rather than across your whole historical database. You are not asking whether a dog is good by global standards; you are asking whether it is good relative to the other dogs in this race today.

Weighting: How Much Does Each Variable Matter?

Once each variable is on a 0 to 100 scale, you can combine them into a single rating by applying a weight to each. The weights are how you tell the model, "early pace matters more than days since last ran", or whatever your judgement tells you based on what you have read and observed.

Weights do not need to be exotic. A reasonable starting point is to give the strongest predictive variables the largest share, and let the rest contribute smaller amounts. If you believe early pace is the dominant factor at sprint venues, it might carry a weight of, say, 1.5 or 2.0, while a context variable like days since last race might carry 0.5. The exact numbers matter less than the principle that the weights sum to a sensible total and reflect a defensible view.

A clean way to set initial weights is to start equal and then deliberately adjust based on what you already know about the sport. You are not optimising on past results at this stage; you are encoding prior knowledge. The point of a transparent model is that the weights can be discussed, argued about and revised openly, not that they emerged from a black-box procedure.

Once weights are set, the rating for each runner is simply the sum of *weight × normalised score* across all variables. Most spreadsheets will handle this with a single formula per dog. The runner with the highest total is your model top pick for that race.

> **TRY THIS**
> Write the weights on a piece of paper next to your screen, not just inside the spreadsheet. If you cannot justify each one in a sentence, it is probably wrong for now.

Turning Ratings Into Useful Outputs

A raw rating on its own is not very helpful. What you really want is something that tells you, for today's card, which dogs are most worth backing. There are a few sensible ways to turn a rating into a betting decision.

The most direct is a **ranked list**. Sort runners from highest rating to lowest. Your top one or two become your model picks. This is the simplest output and the easiest to sanity-check against the form in the Racing Post.

A more useful version is a **rating edge** compared to the market. Take your rating, convert it into an implied probability using a simple method (for example, divide each rating by the sum of all ratings for the race), then compare that probability to the implied probability from the bookmaker's odds. If your model says a dog is 25% likely to win and the market says 18%, that is an edge worth investigating. We will go deeper into market odds in a later chapter, but even at this stage, comparing your rank to the market odds is a useful habit.

A third output, useful for sanity-checking, is a **confidence band**. Rather than just producing a single rating, also flag whether the top runner is clearly ahead of the field or whether the race looks tightly bunched. Tightly bunched ratings are a signal that the race is hard to call and probably not worth a large bet.

Building your first rating, step by step

- [ ] Pick four to eight variables from trap, form and context buckets
- [ ] For each variable, choose a normalisation method and apply it per race
- [ ] Assign weights that reflect what you genuinely believe matters most
- [ ] Calculate each runner's total weighted score
- [ ] Sort runners into a ranked list and compare with the market
- [ ] Keep a one-page written summary of the model beside your screen

Sanity-Checking Against Actual Results

The most important habit you can build is to check your model against reality, not against how it *feels*. There are three quick checks you can run even before you think about formal backtesting.

First, look at how often your **top-ranked runner actually wins**. As a rough benchmark, favourites win around

of graded UK greyhound races, varying between roughly 31% and 42% depending on the venue. If your top pick is winning meaningfully more often than that across a reasonable sample of races, you are onto something. If it is winning at the same rate as the favourite, you have built an expensive way of saying "back the favourite".

Second, check whether your rating produces **different rankings from the market**. If your top three picks for every race match the three favourites in the market, your model is essentially mirroring the odds rather than pricing anything independently. Some agreement is fine, but a model that never disagrees with the market cannot generate an edge, because the market already includes the bookmaker's structural margin, which is typically around

across a six-runner race.

Third, look at **where your model is wrong**. Pick the last twenty races where your top-rated dog lost. Read the form by eye. Was there a clear trap draw issue, a step up in grade, or a long lay-off that you did not capture in the variables? Those are the races that tell you what to add next. Equally, look at the races where your model correctly picked an outsider at a bigger price. Those are the races that tell you what the model is actually good at.

> **AVOID THIS**
> Do not "fix" the model the moment one race goes wrong. A single result is noise. Look for patterns across at least thirty races before changing weights or variables.

Designing your own first rating

1. Which four to eight variables will you include, and which bucket does each sit in?
2. Which normalisation method will you use for each variable, and why?
3. What weight will you give to early pace, and what is your justification?
4. What does the output look like for a typical six-runner race, and how will you read it?
5. How will you record results so you can compare your top pick to the favourite each week?

Keep a One-Page Description of the Model

This is the bit most beginners skip, and it is the bit that makes everything else easier. Write a single page that describes your rating system in plain language: which variables you use, how each is normalised, what weights you have assigned and why, and what the intended output is. Keep it next to your screen. Update it whenever you change anything.

Six months from now, when you look back and wonder why your model rated a particular dog so highly that day, that page is what saves you. It turns "the spreadsheet said so" into "I weighted early pace twice as heavily as days since last ran, and on that day the dog had a strong first-bend record". That is the difference between a system you can audit and a system you have to take on faith.

The same page also makes it easier to explain your approach to anyone else, which is a surprisingly good test. If you cannot describe your model to a friend in a couple of minutes, the model is probably doing things you do not understand yet. That is not a disaster, but it is a signal to simplify before you add anything new.

What a Good First Model Looks Like in Practice

A first rating worth trusting tends to have three characteristics. It uses a small number of clearly defined variables. It produces rankings that are stable across races but not identical to the favourite every time. And when you look at its winners and losers, the pattern makes sense to you in terms of greyhound racing, not just in terms of numbers.

It will not be profitable yet, and that is fine. Profit comes from how you stake, how you compare to the market, and how you iterate over time. The rating is the foundation: a transparent, explainable way of turning racing data into a single number per runner. Everything else in the rest of the book builds on top of that.

In the next chapter we will look at how to move beyond a hand-tuned rating into more rigorous statistical models that still stay interpretable, so that you can decide whether the extra complexity is worth it for the way you want to bet.

KEY TAKEAWAYS

- Start with four to eight variables drawn from trap, form and context, not dozens
- Normalise every variable to a common 0–100 scale before combining them
- Assign weights that reflect what you genuinely believe matters most about the sport
- Produce a ranked list, then compare it against the market rather than trusting it alone
- Sanity-check your top pick's strike-rate against the favourite baseline of around one in three
- Keep a one-page written description of the model so you can explain it to yourself later

CHAPTER 11

Statistical Models You Can Use

If your hand-tuned rating from Chapter 10 picks the winner more often than a coin flip, you're already ahead of most greyhound punters. But there is a ceiling, and you will hit it sooner than you think. Once you have a handful of weighted variables, the next step is to let the maths do the weighting for you, fairly and consistently, across thousands of races. That is what statistical models are for. And the good news is that the most useful ones are not black boxes. They are tools you can open up, argue with, and explain to yourself six months from now, which is the same standard we set for ratings.



Why ratings reach a limit

A rating is a small set of rules you have written down. Maybe trap 1 is worth plus 4 points, last-run finishing position counts double, and a fresh dog after 21 days off loses a point. It works because you have distilled experience into a formula. The trouble is that the weights are guesses, the variables interact in ways you cannot easily capture, and there is no objective way to know whether trap 1 is really worth plus 4 or plus 6.

Statistical models do not magically know the answer either. What they do is search through the data in a structured way, balancing the contribution of each variable against the actual win rates you have recorded. The result is rarely a flashy insight. Most of the time it is a slightly better, slightly more honest version of what you would have done by hand. That is exactly what we want, since this is a system for deciding what numbers are actually worth, not a hunt for the best dog.

20-22%

Win rate for trap 1 at a typical compact UK track over standard sprint distance (vs 16.7% random expectation), illustrating the measurable edge a statistical model can capture from trap draw data

Source: Crayford Greyhound

That number is worth pausing on. Trap 1 wins roughly 20 to 22 per cent of the time at a typical compact UK track over the standard sprint distance, against a random expectation of 16.7 per cent across a six-dog race. That gap, a few percentage points, is the kind of measurable edge a model can capture from a single variable. The market often knows this already, but the size of the effect, and how it changes by track, is exactly the kind of thing a well-built model should be able to articulate for you.

Logistic regression: the workhorse

If you only ever learn one statistical model, learn logistic regression. It is the bread-and-butter tool for turning a list of variables into a probability of winning. The idea is simple: you feed in your features, and the model spits out a number between zero and one, which you can read directly as a win probability.

The mathematics behind logistic regression is more gentle than its reputation suggests. For each variable, the model learns a coefficient. A positive coefficient pushes the probability up, a negative one pushes it down. Add them all together, run the result through a squashing function, and you have a probability. You can sit at a kitchen table with a calculator and roughly replicate the logic. The computer just does it faster and more honestly.



In practice, you do not need to write the algorithm yourself. You can use a library like scikit-learn in Python, or a stats package in R, or even a logistic regression routine in a spreadsheet for very small problems. The point is that you understand what is happening under the bonnet.

Building a logistic regression in practice

- [ ] Pick a clear outcome: did the dog win, yes or no, in each historical race.
- [ ] Choose a small set of variables you trust, such as trap number, recent finishing position, days since last run, and track identifier.
- [ ] Make sure every variable is numeric or has been turned into a numeric flag, for example one column per trap.
- [ ] Split your data into a training set and a holdout set, and never look at the holdout while fitting.
- [ ] Fit the model on the training set and read off the coefficients.
- [ ] Test the resulting probabilities on the holdout set and check whether they beat the market.

One important habit: always include a "random" baseline alongside your model, even if it feels silly. If a six-dog race with no information has a 16.7 per cent chance per runner, that is your floor. Anything your model produces should be meaningfully better than that floor on out-of-sample data, or it is not doing useful work.

Historical analogues: learning from similar races

Logistic regression is not the only tool, and it is not always the best one. Sometimes the most honest answer comes from finding races that looked like the one you are trying to price, and asking how the runners in those races actually performed. This is the idea behind historical analogues, sometimes called similar-race lookups.

The logic is intuitive. If you have a six-dog race tonight at a track with a 400-metre circumference, sprint distance, an A1 grade, and a favourite in trap 1, you can search your database for races that share most of those features and see how often trap 1 won. If the answer is 24 per cent across two thousand similar races, you have a defensible estimate of the favourite's chance. No coefficients required.



This approach has real strengths. It captures interactions between variables without you having to specify them. It is also very easy to explain, because you can point to the actual analogue races that shaped the estimate. Its weakness is that it depends on having enough similar races in the first place. For exotic grades, distances, or smaller meetings, your sample can shrink to nothing, and the estimate becomes unreliable.

> **TRY THIS**
> When you build an analogue, list three to five variables that matter, such as track, distance, grade, trap, and a rough form band, then look at the win percentage of the trap you are interested in across the resulting slice. Treat anything based on fewer than a few hundred races as provisional and rerun it when you have more data.

Penalised models: letting the maths ignore the noise

Once you start adding variables, you will hit a problem. Some of them are genuinely useful. Some of them are pure noise. A standard logistic regression will happily give noise a strong coefficient if it happens to line up with a few lucky winners in your training data. That is how overfitting begins.

Penalised models, often called regularised regression, are designed to deal with exactly this. The two common flavours are ridge regression, which shrinks coefficients towards zero, and lasso regression, which can shrink some coefficients all the way to zero, effectively dropping variables. The "lasso" effect is useful because it gives you a kind of automatic feature selection: noisy variables get quietly pushed out, and the survivors are the ones the model considers worth keeping.

For greyhound data, where you have many plausible variables and not always a huge amount of clean data per race, penalised models are a sensible default. They are also still interpretable. You can still read the coefficients and ask why a variable got a positive or

negative weight. They are not the same as a black-box neural network, and that matters for a system you want to trust over months and years.

Comparing common modelling approaches

Approach	Strength	Weakness	Best used when
Hand-tuned rating	Transparent, easy to edit	Weights are subjective, hard to scale	You are starting out and want a baseline
Logistic regression	Probabilistic, interpretable	Struggles with complex interactions	You have a handful of solid variables
Historical analogues	Captures interactions naturally	Needs large, similar samples	Common race types with deep history
Penalised regression	Handles noisy variables gracefully	Slightly more tuning required	You have many candidate features

Whichever approach you choose, the principle is the same. The model is a tool for pricing, not for predicting the future with certainty. A 22 per cent probability is a price, and the price is what tells you whether a bet is worth taking.

Track geometry and why one model is not enough

Trap draw is a good example of a variable that looks simple but behaves differently from track to track. The compact tracks, where the run to the first bend is short and the inside traps have a tight advantage, will give you a strong trap effect. The larger tracks, with wider bends and longer runs, soften it considerably.

> **18%** Win rates for inside vs outside traps at larger UK tracks with wider circumferences and longer runs to the first bend, showing how track geometry attenuates the trap-draw variable a model should weight
>
> Source: Crayford Greyhound

The two numbers above are not there to be memorised. They are there to make a point. If you fit a single model across every track and every distance, it will average these effects and miss both tails. A better approach is to either include track as a categorical variable, so the model can learn a separate trap effect per venue, or to fit separate models for groups of similar tracks.

This is one of the quiet advantages of statistical modelling over a single hand-tuned rating. A rating assigns one value to trap 1. A model can quietly learn that trap 1 at Crayford is worth more than trap 1 at a bigger, straighter track. You do not have to argue

with it. The data does the arguing for you.

Comparing your prices to the market

A model that produces a win probability is only useful if you compare it to something. That something is the market. The bookmaker's odds, and the exchange prices where they exist, are the consensus opinion of every punter in the world, including sharp ones. They are not always right, but they are rarely stupid.

> **15-20 pence per**
> Typical bookmaker overround — the steady, predictable loss rate a non-value bettor can expect, used as a benchmark for value-betting models
> Source: Greyhound Racing Odds UK

That figure, fifteen to twenty pence lost per pound staked, is the steady drip of the overround. It is what an uninformed bettor can expect to lose in the long run. Your model exists to find the spots where the market is wrong enough, often enough, to overcome that drip and put you on the other side of the ledger.

The comparison itself is straightforward. For each runner, you have your model's probability, say 25 per cent, which corresponds to fair odds of 4.0, or 3/1. If the bookmaker is offering 5.0, or 4/1, you have a value bet. If the bookmaker is offering 3.0, or 2/1, you do not, no matter how much you like the dog. This is the heart of quantitative betting: you are not asking who will win, you are asking whether the price is fair.

> **AVOID THIS**
> Treating your model's probability as truth without checking it against the market. Even a well-fitted model can be biased. If your model says 30 per cent and the market says 12 per cent, one of you is wrong, and the burden of proof is on the model, not on the price.

A useful exercise is to plot your model's probabilities against actual win rates. If your 25 per cent shots are winning 25 per cent of the time out of sample, the model is well calibrated. If they are winning 30 per cent, it is conservative, which is a nice kind of error to have. If they are winning 20 per cent, the model is overconfident and needs attention.

Keeping models interpretable on purpose

It is tempting, once you have seen what logistic regression can do, to reach for more powerful tools: gradient boosting, neural networks, anything with a longer name. Some of those tools are genuinely powerful, but they also tend to be opaque. You cannot easily

ask a complex model why it liked a particular dog, and that is a problem when the data shifts, as it does when a track changes its surface or a trainer's runners suddenly drop in form.

For most greyhound bettors, interpretable models are the right choice. Logistic regression, penalised regression, and analogue methods all have the virtue that you can stare at their output and form a judgement about whether it makes sense. That judgement is a feature, not a limitation. It is how you catch silent errors before they cost you a month's bankroll.

There will be a chapter on backtesting next, and most of what we have built here is exactly the kind of thing that needs careful testing before you trust it with real money. Treat this chapter as the moment you stopped drawing your weights by hand and started asking the data to draw them for you, with you still in the room asking questions.

KEY TAKEAWAYS

- Statistical models improve on hand-tuned ratings by learning variable weights from the data rather than guessing them.
- Logistic regression is the most useful starting point because it produces probabilities directly and stays interpretable.
- Historical analogues are a powerful complement, especially where race types repeat often enough to build a sample.
- Penalised models help you ignore noisy variables automatically and reduce the risk of overfitting.
- A model's output is only useful when compared to market prices, since value betting is about price, not prediction.

With a working model in hand, the natural next question is the one every quantitative bettor eventually asks: how do I know it would actually have made money? That is the job of backtesting, and it is more slippery than it looks.

CHAPTER 12

Backtesting Your Betting Ideas

The backtest of your dreams is dangerous. It is the version where every "if" lines up, where the dog you would have backed at 4.0 wins at 4.0, where there are no non-runners, no late rule changes, no commission, and where your model was somehow built without ever seeing the data you are testing it on. Most betting systems, including some genuinely thoughtful ones, die the moment their author steps from the backtest into the live market. The good news is that a backtest that survives reality is not hard to build. You just have to refuse the comfortable version and write the kind that argues back.



What a backtest is really doing

A backtest is a thought experiment with discipline. You are asking: *if I had used this rule, on this data, with this knowledge, how would I have done?* The phrase "with this knowledge" is doing the heavy lifting. Anything that crosses from the future back into your rule will make the test look better than it is. That is the central problem of backtesting, and almost every trap on the page is a flavour of it.

For greyhound betting, you are usually backtesting one of three things: a rating model that ranks runners, a set of filters that excludes races, or a staking plan attached to either. The shape of the test changes a little for each, but the discipline does not. You need historical odds, historical results, the full list of runners that went to the traps, and a clean record of what your rule would have told you to do, race by race, in the order the races actually happened.

> **GOOD TO KNOW**
> A backtest is not a proof. It is a filter. Its job is to remove ideas that are unlikely to work, not to confirm that one will. Treat a passing backtest the way a doctor treats a clear blood test: useful, not conclusive.

Building a realistic historical odds dataset

The single biggest reason backtests mislead is the odds they use. If your historical record only contains the *winner* and a snapshot of the starting price, you have lost half the story. You do not know what price the third favourite actually opened at, you do not know whether the favourite drifted in the final minute before the off, and you do not know what was available in the thirty seconds before the traps flew, when real bets get placed.

Try to capture three layers of price for each runner where you can:

- **Morning price or 24-hour price**, taken well before off, useful for testing "early value" strategies.
- **Final show price**, as close to the off as your feed allows, useful for testing what a bettor placing a bet a few minutes before the race would have seen.
- **Starting price (SP)**, the industry reference price, useful as a benchmark and as a fallback when finer data is missing.

Where your feed only gives you SP, say so. Pretending SP is the same as the price you would have taken is one of the quieter ways people inflate their backtest return. SP has a known bias: it tends to overpay short-priced favourites and underpay outsiders, because bookmakers trim favourites late when money arrives. If you backtest a "value outsider" strategy on SP only, you will flatter yourself.

> **5/1**
> Average price at which a 25% strike rate is described as highly profitable, illustrating the price threshold needed for a low win rate to deliver value over a backtest.
> Source: Greyhound Racing Odds UK (Value Odds Tracker)

That matters because a strategy that picks 5/1 shots at a 25% strike rate is a quietly powerful one. A backtest built on the wrong price column can easily convince you that this rule is a small edge, when really it is just SP being generous to outsiders. The moment you switch to a feed that reflects real market depth, the edge can shrink dramatically or vanish.

Walk-forward testing versus a single split

The classic mistake is to build a model on two years of data, test it on the third year, and call that a pass. It is, at best, a pass on one window. Greyhound racing is not stationary. Tracks get resurfaced, trainers change kennel arrangements, grading policies shift, betting volumes move between meetings, and the population of dogs turns over completely every few seasons. A rule that worked in 2022 may already be a fossil by the time you read this.

Walk-forward testing accepts that fact. Instead of one train-and-test split, you do many, stitched together. You train on, say, the first six months, test on month seven. Then you train on months two to seven, test on month eight. Then months three to eight, test on month nine, and so on. Each test window is out of sample because your rule did not see it during its construction. Each result is then stitched into a longer out-of-sample run.

Single split versus walk-forward

Aspect	Single train-test split	Walk-forward testing
Out-of-sample windows	One	Many, stitched together
Robustness to regime shifts	Low, one bad window kills it	Higher, several windows averaged
Risk of overfitting to a single year	High	Lower
Computational effort	Low	Moderate to high
What it tells you	Whether the rule worked then	Whether the rule tends to work across time

The honest output of a walk-forward is a chart of cumulative profit that moves through many regimes, not a single headline number. You want to see your equity curve spend more time going up than down, and you want to see it hold up across a variety of conditions, not just one hot streak.

> **TRY THIS**
> Run your rule on at least four walk-forward windows that include a slow month, a fast month, a meeting dominated by short-priced favourites, and one dominated by long shots. If it cannot survive that variety in the backtest, it will not survive a calendar year in the real world.

Account for commission, non-runners and withdrawals

The cleanest model in the world loses to careless accounting. Three line items drain every backtest that ignores them.

1. **Commission or exchange commission.** If you are simulating bets on a betting exchange, take 2 to 5% off winning returns for every bet. If you are simulating bets with a traditional bookmaker at SP, your commission is the overround baked into the price and you should leave a margin for it. Either way, do not assume you keep 100% of returns.
2. **Non-runners.** A dog withdrawn after the overnight declaration but before the traps changes the entire race. If your rule said "back the second favourite if the favourite is absent", you need to know whether the favourite was actually absent at the time of betting. Most APIs tell you the final field, not the field at every prior moment, so this takes careful handling.
3. **Rule-4 deductions.** When a non-runner leaves the race, bookmakers reduce the payout on remaining runners. If your backtest simply assumes everyone collects at full odds, your returns are overstated.

Then there is the more subtle leak: a rule that tells you to back a dog, but the dog was already backed by the market and has shortened to a price where your edge has gone. Walking through your historical odds by *time of bet*, not by final price, is what closes that gap.

The chart shows what strike rate a flat-stake backer needs just to break even at each price before commission. A selection that wins 33% of the time at 2/1 looks decent on a strike-rate chart but is barely paying for itself. Add 5% commission and the picture worsens. A backtest that celebrates a 30% strike rate at evens is celebrating something that loses money.

> **6/4** Average price at which even a 40% strike rate is unprofitable, the counter-example used to show why backtests must price-check selections, not just count winners.
> Source: Greyhound Racing Odds UK (Value Odds Tracker)

That is exactly why price-checking matters in a backtest, not just winner-counting. A 40% strike rate sounds excellent in isolation, but at an average price of 6/4 it is a losing strategy once overrounds are factored in. If your backtest reports strike rate without showing the average price your bets were actually taken at, treat it as incomplete.

Your backtest lab notebook

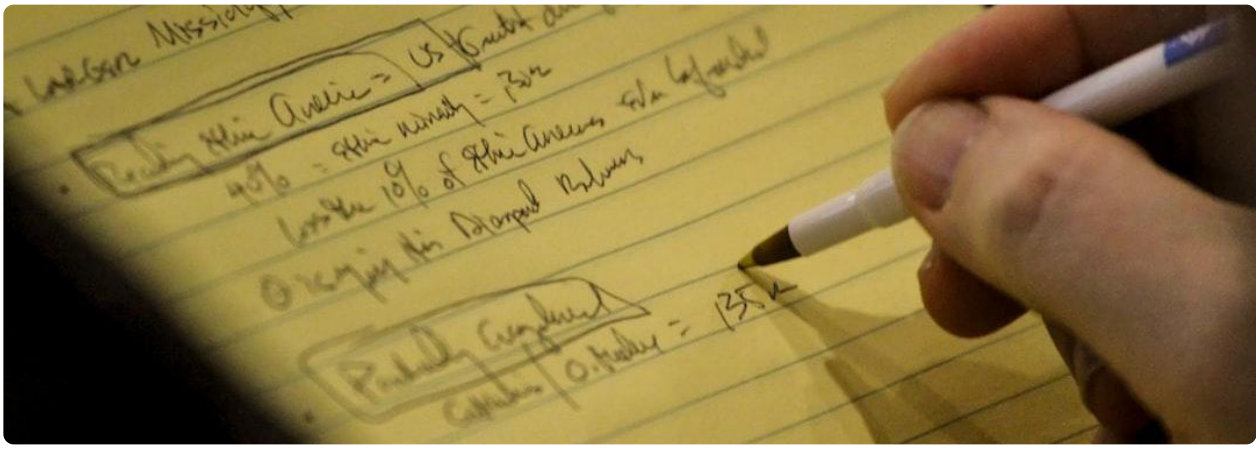
You will run dozens of backtests as you iterate, and you will forget what you did in each one within a fortnight. The only defence is a disciplined record. For every test, write down the version of your rule, the dataset it ran on, the date range, the walk-forward

windows, the commission assumption, and the headline numbers: number of bets, strike rate, average price, return on investment, longest losing run, biggest drawdown.

Backtest recording checklist

- [ ] Rule version and a one-sentence description of what it does
- [ ] Dataset version and date range covered
- [ ] Walk-forward window length and step size
- [ ] Odds source: morning, show, or SP
- [ ] Commission rate assumed and where it is applied
- [ ] Treatment of non-runners and rule-4 deductions
- [ ] Number of bets, strike rate, average price, ROI
- [ ] Longest losing streak and peak-to-trough drawdown
- [ ] Notes on anything unusual about the period covered

That last item matters. A backtest covering December through February is not the same animal as one covering June through August. Tracks, trainers, and meeting volumes shift with the calendar. If your best backtest happens to coincide with a particularly favourable slice of racing, you need to know that before you bet on it.



AVOID THIS

Re-running a backtest with a small tweak until the numbers improve. That is overfitting by another name, and you will convince yourself that the version of the rule that just happens to look best on this slice of history is the one to trust. Pick your rule honestly, run it once, record it, and only change it for a reason you can write down.

Reading the result honestly

Once you have a clean backtest with all the leaks patched, the question becomes how to read it without lying to yourself. Three checks help.

First, compare to a baseline. How would a flat-stake strategy of backing every favourite at SP have done over the same window? How about backing every dog in every race? Your rule should beat a trivial baseline, not just look good in the abstract.

Second, look at the distribution, not just the average. A rule with a 6% ROI that hit it consistently across many small months is more trustworthy than a rule with a 12% ROI that made all its money in three races. Pay attention to variance as much as to mean.

Third, stress test it. Re-run the same test on a completely different slice of data, ideally from a different track or meeting type, and see whether the shape of the result holds. A rule that works only on one grade of racing or one track shape is fragile.

What comes after the backtest passes

A backtest that survives all of this is a starting point, not a finish line. It tells you the rule has plausibly worked in the past, on data your construction did not see, under assumptions that mirror how you would actually have bet. That is the bar. Anything below it is wishful; anything above it, for now, is too much to ask.

The next question is the harder one: how do you know whether it is still working once you start betting for real? That is where the metrics in the next chapter come in.

KEY TAKEAWAYS

- A backtest is a filter, not a proof; its job is to rule out weak ideas, not to crown winners.
- Use the price you would actually have taken, not SP, and keep separate columns for morning, show and SP where you can.
- Prefer walk-forward testing over a single split so your rule is tested in many regimes, not one lucky window.
- Always deduct commission, handle non-runners carefully, and apply rule-4 deductions where relevant.
- Record every test in a notebook with rule version, data range, windows, assumptions and headline numbers.
- Compare your rule to a trivial baseline, watch the distribution of returns, and stress test on a different slice of racing.

CHAPTER 13

Measuring Real-World Performance

The first losing run feels personal. Even when you know it is coming, even when you have prepared yourself with backtests and spreadsheets, watching your bankroll shrink in real time tests every assumption you ever made about edge and discipline. The honest truth is that every quantitative system will have weeks, sometimes months, where the numbers look wrong. The job of this chapter is to help you read those moments properly: to separate the noise from a real signal, and to know whether your system is quietly working or quietly failing.

Real-world performance measurement is where the book stops being about building a model and starts being about *trusting* one. It is also where most bettors quietly give up on good systems, and quietly keep bad ones going. A few well-chosen metrics, kept consistently over time, are the difference.

Why live results always look worse than your backtest

If you took the time to follow the backtesting chapter properly, you will have produced a set of historical results that looked encouraging. Then you started betting live and the early weeks felt a bit underwhelming. This is normal and worth understanding in advance so you do not panic and abandon a working system.

There are three honest reasons live returns underperform backtested returns. First, **slippage**: between the moment your model flagged a bet and the moment you actually placed it, the odds moved. Second, **selection friction**: in a backtest you take every signal; in real life you sometimes miss a race, stake the wrong amount, or skip a bet because the price disappeared. Third, **market adaptation**: once a pattern is consistently being taken, the market often moves slightly against it. None of these are reasons to dismiss your backtest. They are reasons to budget for them and to measure live performance on its own terms.

> **GOOD TO KNOW**
> A reasonable rule of thumb is to subtract one to two percentage points from your backtested yield before you treat it as your live expectation. It is a rough adjustment, but it keeps your mental model honest while you gather real data.

The three numbers that actually matter

You will be tempted to track dozens of metrics. Resist that. Three numbers, recorded weekly, will tell you almost everything you need to know about whether your system is working.

The three core performance metrics

Metric	What it measures	How to read it
Yield	Profit as a percentage of total stakes (e.g. +4% means £4 profit per £100 staked)	The headline number. Positive and stable over time = your edge is real. Drifting toward zero or below = trouble.
Strike rate	Percentage of bets that win	Useful for sanity-checking yield. A very high strike rate at short prices can be a warning sign, not a triumph.
Return on investment (ROI)	Total returns divided by total stakes	Often used interchangeably with yield. Strictly, ROI can include the return of your stake plus profit; yield is profit-only. Pick one and stick with it.

Of these, **yield** is your north star. Strike rate is only meaningful in relation to the average odds you are taking, and ROI becomes misleading if you ever stake varying amounts without normalising. A flat-stake yield of two to four per cent over hundreds of bets is a meaningful edge in greyhound betting. Anything above that is genuinely impressive and worth defending.

One thing to remember: the bookmaker has a margin baked into every race. The structural advantage they hold is the overround, the gap between the true probability of all runners and the implied probability of the prices on offer.

> **5%** Bookmaker overround, the theoretical profit margin built into greyhound race odds, which a bettor must overcome to be profitable in the long run.
> Source: Towcester Racecourse

That five per cent is the minimum you must overcome before you start making money. If your long-term yield is sitting below that figure, you are not beating the market, you are losing to it more slowly.

The painful numbers nobody likes to track

Yield tells you whether you are winning. It does not tell you how much pain you will have to endure on the way. For that, you need two more metrics, which most recreational bettors ignore and most professionals obsess over.

Drawdown

Drawdown is the percentage fall from your highest bankroll point to your lowest subsequent point. If your bankroll peaks at £1,200 and later hits £960, you have experienced a 20 per cent drawdown. Drawdown tells you how much of your bankroll you should be prepared to give back during losing runs, because they *will* happen even with a working edge.

Longest losing run

This is the longest streak of consecutive losing bets your system produces. It matters for two reasons. First, psychological: knowing in advance that your system has historically produced a twelve-bet losing streak makes it easier to ride out an eleven-bet losing streak without breaking your staking rules. Second, structural: your staking plan needs to be sized so that a typical losing run does not wipe out a meaningful slice of your bankroll.

> **TRY THIS**
> From your betting log, calculate your worst drawdown to date and your longest losing run. Write them down somewhere visible. Knowing these numbers in advance removes most of the panic from future bad patches.

Separating luck from skill over small samples

This is the hardest part of performance measurement and the part that ruins most well-meaning bettors. After fifty bets, your yield could swing by ten percentage points purely because of variance. After two hundred, swings narrow. After a thousand, you can start to trust the number.

The right way to think about your live results is as a *distribution*, not a single answer. If your model suggests a true edge of three per cent, your actual yield after two hundred bets could plausibly land anywhere from roughly negative two per cent to positive eight per cent, just from luck. That is not failure. That is mathematics.

A simple way to keep yourself honest is to track not just your running yield but also how many bets you have placed. Write down a rough minimum sample size you want to reach before drawing strong conclusions, then keep betting the same way until you get there. Common sense thresholds:

- **Under 100 bets:** treat results as anecdotal, useful only for spotting catastrophic errors.
- **100 to 500 bets:** starting to be informative, but still well within noise range.

- **500 to 1,000 bets:** meaningful signal emerging if the figure is stable.
- **Over 1,000 bets:** you can begin to trust the headline number.

The structural feature of greyhound racing is worth keeping in mind here. Most UK races contain a fixed number of runners, which means your probability field and your overround exposure are consistent from race to race.

> **6 or 8**
>
> Number of greyhounds competing in a typical UK greyhound race, a structural feature that determines the size of the probability field and the size of the overround the bettor faces.
>
> Source: Towcester Racecourse

Because the field size is roughly constant, your variance per bet is fairly predictable. That helps a little with sample size calculations, but it does not shortcut them. You still need the volume.

The betting log: your single most important tool

If you only adopt one habit from this chapter, make it this one. Keep a clean, consistent, complete betting log from day one of live betting. The log is not optional. It is the device that turns a sequence of bets into a dataset, and a dataset is what makes everything in this book possible.

What to record, every single time, no exceptions:

What goes in every line of your log

- [ ] Date and time placed
- [ ] Track and race number
- [ ] Dog name and trap number
- [ ] Odds taken (decimal)
- [ ] Stake (in units, not pounds, if you want to scale later)
- [ ] Model rating or probability
- [ ] Reason for the bet (one short phrase, kept consistent)
- [ ] Result and profit or loss for that line

The single most valuable discipline is *consistency*. Whatever fields you choose, record them the same way for every bet. If you sometimes record your reasoning and sometimes skip it, the log loses most of its analytical value. If you sometimes skip a bet because you forgot to write it down, the log becomes biased toward winners, and biased data will quietly mislead your future decisions.

A practical question is whether to log your bets by hand, in a spreadsheet, or in a small database. Any of these is fine. The best system is the one you will actually use every day. A spreadsheet with one row per bet and the columns above is the minimum viable log. A simple SQLite table with a few extra columns for your model output is better still.

Common traps when reading your own numbers

Once you have a log and a few weeks of data, the temptation is to start interpreting every movement as a story. Some of those stories will be true. Most will not. A few patterns deserve to be flagged because they catch nearly everyone.

Recency bias

The last twenty bets feel more meaningful than the two hundred before them. They are not. A losing weekend at the end of a profitable month is mathematically less informative than the month itself. Stick to your pre-set sample size milestones before drawing conclusions.

Cherry-picking which bets to log

If you log winners carefully and losers loosely, your log is no longer a record of your system. It is a record of the bets you remember liking. This is one of the fastest ways to convince yourself a broken system is working.

Changing rules mid-stream

Adding a new variable to the model halfway through a sample is one of the cleanest ways to corrupt performance data. If you change something, start a fresh sample from that date and label the old one as closed.

Letting stake size drift

If your staking plan says one unit and you quietly stake two when you feel confident, your yield is no longer comparable to your backtest. Use a fixed unit size relative to your starting bankroll and do not adjust it on feel.

> **AVOID THIS**
> Do not start a new betting log every time you tweak a model. Either keep the old log and label sections, or close it formally. Half-merged logs are worse than no log at all because they look authoritative.

Putting it together: a monthly review habit

Once a week is too often to draw conclusions from your log. Once a year is too rarely to catch a broken system early. A monthly review is the right cadence for most bettors, and it does not need to be elaborate. Three numbers from the month, three numbers from the running total, and one honest paragraph of notes.

Monthly performance review

1. What was my yield this month, on a flat one-unit stake?
2. What was my strike rate at the average odds taken this month?
3. What was my largest drawdown so far in the live sample?
4. Did I stick to my staking plan for every bet, without exception?
5. Did I log every single bet, including the ones I felt uncertain about?
6. If my yield this month was negative, can I explain it with luck, or is something structurally wrong?

The last question is the one that matters most. A negative month early in a live sample is almost always noise. A negative month at bet eight hundred is a different conversation. The log gives you the context to know which.

What performance measurement is really for

The honest purpose of measuring performance is not to feel good about winning weeks or bad about losing ones. It is to give yourself permission to keep doing what is working and to stop doing what is not. Without measurement, every decision is driven by mood. With

measurement, you can be patient with variance and ruthless with systems that are genuinely broken.

The other quiet benefit is that a clean log lets you do something almost no recreational bettor ever manages: you can compare your actual live results to your backtested expectations, and update your beliefs accordingly. Over time, that feedback loop is the edge. Not any single bet, not any single model, but the slow accumulation of honest data about what your system actually does in the real world.

Your bankroll is the place where all of this lands. Yield tells you whether you have an edge. Strike rate tells you what that edge looks like in practice. Drawdown tells you what it costs emotionally and financially to harvest it. The log is the device that connects all three.

TRY THIS

Before placing your next live bet, set up your log with the eight fields listed earlier. Add ten rows of test data from your last ten bets, even if you have to estimate the odds and reasoning. Getting the structure in place is ninety per cent of the work.

KEY TAKEAWAYS

- Yield is the headline metric; positive and stable over hundreds of bets is the working definition of an edge.
- Drawdown and longest losing run are the metrics that protect your bankroll and your sanity during inevitable bad patches.
- Small samples lie, so set minimum bet counts before you trust the numbers.
- The bookmaker's structural margin means you must clear a meaningful hurdle before live results become genuinely profitable.
- A consistent betting log is the single most valuable tool you will build, and the discipline of keeping one is half the work.

With a working log and a set of metrics you trust, the natural next question is how to size your bets so that the edge you have measured can actually survive. That is the territory of the next chapter, where we turn from measuring returns to designing a staking plan that lets them compound.

CHAPTER 14

Bankroll and Staking Plans



A clever model with sloppy staking is a slow way to go broke. You can build a beautifully tuned greyhound rating system, price up every race with care, and still watch your bankroll leak away because you sized your bets like a punter rather than a quant. Money management is not the glamourous end of the business, but it is the part that decides whether your edge ever pays you back for all the evenings you spent cleaning data.

Why your bankroll deserves as much attention as your model

Variance is the quiet enemy of every quantitative bettor. Even a model with a genuine long-term edge will produce long, uncomfortable losing runs, weeks where nothing lands, and races where the dog you backed at 4.0 gets shuffled into third by a head. A good staking plan does not pretend those runs will not happen. It assumes they will and makes sure the bankroll is large enough, and the bets small enough, to come out the other side.

Before you stake a single pound, you also need to accept a hard truth about greyhound markets. Most six-dog greyhound races are built around a substantial bookmaker margin:

> **~125%** The bookmaker's overround on a typical six-dog greyhound race, representing the structural margin bettors must overcome (≈25% margin).
> Source: howtowingreyhoundbet.com

That ~125% book means the market is taking roughly 25% off the top before a ball is released. To come out ahead you do not just need a model that picks dogs well. You need a model that prices races better than the market does, on average, over hundreds of bets.

Staking is what carries you through the hundreds of bets while you wait for the maths to settle.

Flat staking: the honest baseline

Flat staking is the simplest possible system. You pick a unit size, and every qualifying bet is exactly that unit. If your unit is £10 and your model flags three dogs on tonight's card, you back each one for £10. If it flags one dog, you back it for £10. No drama, no adjustments, no panic.

The appeal of flat staking is that it makes it very hard to lie to yourself. You can look at your results at the end of the month and know exactly what you have risked, what you have returned, and whether your strike rate and average price are in line with your model's expectations. That clarity is genuinely useful when you are still refining a system.

The standard unit size recommended for greyhound value betting sits at a very modest percentage of your total bankroll:

> **2%** Recommended unit size as a percentage of bankroll for a flat-staking plan, e.g. £10 units on a £500 bankroll.
> Source: howtowingreyhoundbet.com

So if your betting bankroll is £500, your unit is £10. If your bankroll is £1,000, your unit is £20. This is small enough to absorb normal losing runs and large enough that winning bets actually move the needle. The mistake most beginners make is setting their unit at 5% or 10% because they want to grow quickly. That is a fast way to turn a working system into a frustrating one.

> **AVOID THIS**
> Adjusting your unit size up because you are "on a roll" or down because you are "chasing". Both behaviours destroy the statistical properties that make flat staking safe. The unit is fixed. Your feelings are not part of the formula.

Proportional staking: growth with growing risk

Proportional staking scales your unit with your current bankroll. If your bankroll grows, your unit grows with it. If your bankroll shrinks, your unit shrinks with it. The classic example is the 2% rule again: 2% of whatever your bankroll happens to be this morning.

This approach has a real psychological and mathematical advantage. During winning periods, you naturally bet more, so winning streaks are rewarded more heavily. During losing periods, your bets shrink, so you bleed less. In theory, your bankroll curve becomes smoother and your long-term growth more consistent.

The downside is that proportional staking also amplifies the bad days. A 30% drawdown in a flat-staking system is painful. A 30% drawdown in a proportional system means your next 100 bets are all smaller than the previous 100 bets, and recovery is slower. Proportional systems also tempt you to fiddle. After a bad week, the temptation to switch back to flat, or to switch to a higher percentage, is enormous.

The table below sets the two approaches side by side so you can see the trade-offs clearly.

Flat staking versus proportional staking

Aspect	Flat staking	Proportional staking
Unit size	Fixed at start	Percentage of current bankroll
Behaviour in winning streaks	Steady growth	Accelerating growth
Behaviour in losing streaks	Steady decline	Slower decline, slower recovery
Ease of record keeping	Simple	Requires frequent recalculation
Risk of ruin	Predictable	Lower in good runs, higher if not capped
Best for	New bettors, model testing	Established bankrolls, long horizons

The Kelly criterion in plain language

The Kelly criterion is a famous staking formula from information theory. The short version is that it tells you the fraction of your bankroll you should risk on a bet where you believe your probability of winning is higher than the market's implied probability. The exact Kelly fraction is your edge divided by your price minus one, written as a decimal.

Suppose your model prices a dog at 3.50 but the bookmaker offers 5.00. Your edge is roughly 0.50 implied versus 0.20 implied, and your full Kelly fraction comes out somewhere in the 10 to 15% range depending on how you calculate it. That is enormous. In practice, almost no serious bettor uses full Kelly, because full Kelly assumes you know your edge precisely, and nobody knows their edge precisely.

Full Kelly produces wild bankroll swings. It assumes the worst losing run will be roughly twice as bad as the worst one you have seen. If you have not seen very many bets yet, that assumption is dangerous. Most professional and semi-professional bettors use a fraction

of Kelly, typically a quarter, an eighth, or a sixth. This reduces volatility dramatically while keeping most of the long-term growth benefit.

For a greyhound bettor running a quantitative model on smaller markets, a sensible working range is between a quarter and an eighth of Kelly. If your model output implies a full Kelly stake of 8% of bankroll, you actually stake 1% to 2%. That range lines up neatly with the 2% unit size from the flat-staking section, which is reassuring, because it suggests the conventional wisdom and the formal maths agree.

Maximum bet sizes and total exposure limits

Kelly tells you how much to stake on a single bet given your edge. It does not, on its own, protect you from betting on three dogs in the same race, or five dogs across a busy evening card. That is where exposure limits come in.

A sensible maximum bet size is a hard ceiling, set in advance, that no rating or price ever overrides. Many greyhound bettors cap single bets at around 5% of bankroll regardless of what Kelly suggests. If the maths ever tells you to stake 8% or 12%, you take that as a signal that something has gone wrong with your model rather than an instruction to risk a large slice of your bankroll.

A daily or weekly exposure cap is equally important. Without one, a busy Saturday meeting with fourteen qualifying bets can suddenly put a third of your bankroll at risk in a single afternoon. A common starting point is to limit total weekly exposure to between 10% and 20% of bankroll, so that even a brutal week cannot wipe you out. As your bankroll grows, you can review and adjust these caps based on actual results rather than gut feel.

Before you go live, run through this short list. It will save you from a lot of avoidable pain.

Set up your staking plan

- [ ] Decide your total betting bankroll and treat it as untouchable spending money.
- [ ] Choose flat or proportional staking and write down the rule in plain English.
- [ ] Set your unit size as a small percentage of bankroll, in the 1% to 2% range.
- [ ] Set a maximum single bet, ideally no more than 5% of bankroll.
- [ ] Set a weekly or daily exposure cap, around 10% to 20% of bankroll.
- [ ] Decide your Kelly fraction, if used, and stick to it for at least 200 bets before changing it.
- [ ] Keep a separate ledger for staking rules and review it monthly.

Bankroll size and the role of the minimum bet count

There is one more number worth knowing. A value-betting approach at a typical 20% to 25% strike rate will go through very ordinary losing streaks. To survive those streaks without panicking, or worse, breaking your staking rules, your starting bankroll should be able to cover a meaningful number of staking units:

50 bets

Minimum number of staking units the starting bankroll should cover so a value-betting approach can survive normal losing streaks at a 20–25% strike rate (minimum).

Source: howtowingreyhoundbet.com

Fifty units of headroom means that if your edge is real but slow to manifest, you can keep placing bets at your chosen size without dipping into your rent money or your reserve fund. With a £500 bankroll and £10 units, you are sitting on 50 units of runway. That feels sensible rather than generous, which is exactly the point. Greyhound betting is not meant to feel generous in the short term.

Responsible gambling habits and when to seek help

Quantitative betting is still betting. A model can be elegant, the maths can be sound, and the staking plan can be disciplined, and it is still possible to gamble in a way that hurts you and the people around you. The rules below are not optional extras. They are the foundations under everything else in this chapter.

Use money you can genuinely afford to lose. Treat your betting bankroll as entertainment spending, like a gym membership or a season ticket. If losing the whole bankroll would affect your rent, your bills, or your relationships, the bankroll is too large.

Set hard time and money limits before you start a session, and stop when you hit them. Most betting platforms let you set deposit limits, session reminders, and time-outs. Use them. They are not signs of weakness, they are signs of someone who wants to keep doing this for years rather than weeks.

Watch for the warning signs. Chasing losses, betting when stressed or tired, hiding betting activity from family, increasing stakes to chase a feeling rather than an edge, and obsessing over results to the point where it affects your mood or sleep are all red flags. If any of these show up, step back and talk to someone you trust.

In the UK and Ireland, free and confidential support is available through organisations such as GamCare and GambleAware. They are set up precisely for people who want to think clearly about their gambling. There is no shame in contacting them, and they can help with practical tools such as self-exclusion from betting platforms if that is what you need.

> **TRY THIS**
> Write down three lines tonight and keep them next to your laptop: your total bankroll, your unit size, and your maximum weekly exposure. If you ever feel tempted to break any of the three, the answer is always to stop for the evening rather than to break the rule.

To make the staking plan feel real rather than abstract, spend ten minutes with this worksheet. The numbers should fit on a single page. If they feel too tight, your bankroll is too small for your chosen unit. If they feel too loose, you are probably under-staking and leaving growth on the table.

Your staking numbers

1. What is your total betting bankroll, in pounds, that you can genuinely afford to lose?

2. What unit size does the 2% rule give you for that bankroll?

3. What is the smallest whole-pound bet you can actually place at your usual bookmaker?

4. How many staking units does your bankroll cover at that unit size?

5. What single-bet ceiling, in pounds and as a percentage, will you never exceed?

6. What weekly exposure cap, in pounds and as a percentage, will you never exceed?

7. If your bankroll falls by 30%, what will your new unit size be under proportional staking?

The point of all of this is not to make betting exciting. The point is to make it survivable, and to give your quantitative edge the room it needs to express itself. A working model with a boring staking plan will outperform a flashier model with heroic staking every single time, over the kind of horizons that greyhound betting actually rewards.

KEY TAKEAWAYS

- Treat your bankroll as a tool for surviving variance, not as a lump sum to be won or lost quickly.
- A unit size around 2% of bankroll is a sensible default for most greyhound value bettors.
- Kelly fractions of a quarter to an eighth are usually safer than full Kelly in real markets.
- Cap single bets and total weekly exposure to prevent one bad day from defining your season.
- Responsible gambling habits, including deposit limits and external support, are part of every serious staking plan.

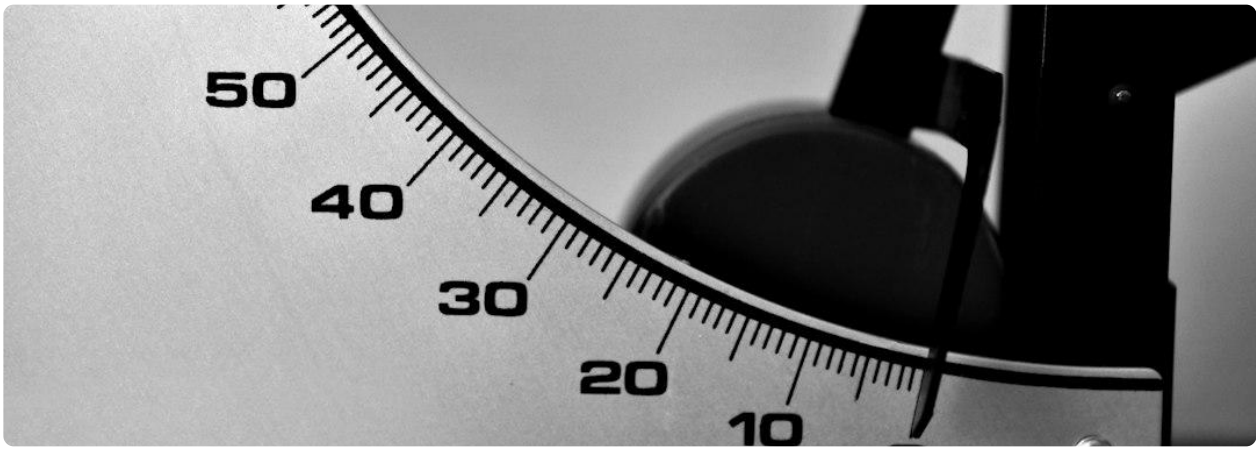
The next chapter turns to the trickier side of model-building: how to keep improving your ratings and probabilities without quietly fitting your system to yesterday's winners. That is where disciplined staking and disciplined iteration come together.

CHAPTER 15

Iterating Without Overfitting

There is a quiet moment, somewhere around your fiftieth tweak, when you add a new variable to your model, your backtest lights up, and you feel a small, dangerous thrill. The hit rate ticks up. The return on investment nudges higher. You tell yourself, *this one really is the missing piece*. Then you take it live and the edge quietly disappears. That feeling is the fingerprint of overfitting, and almost every quantitative bettor has to learn, the hard way, how to stop reaching for it.

This chapter is about the discipline that sits between *using* a model and *improving* one. It is the least glamorous work in this book, and it is the work that decides whether your system quietly compounds over months, or whether it crumbles the first time the racing calendar turns over.



The trap that catches almost everyone

Your model is a summary of what you believe matters in a greyhound race. The moment you start changing it to fit what already happened, you stop summarising belief and start memorising history. Greyhound racing is noisy. Dogs get interfered with, fall, miss the break, get crowded on the first bend. A hundred small random things happen in every race, and they cancel out over hundreds of races only if your sample is honest.

The problem is that you have a thousand knobs to turn. Which track? Which grade? Which weight range? Trap draw? Recent form over two runs or four? Days since last race? Sectional times if you have them? Each one is a tiny opportunity to nudge a backtest

figure upwards. Add enough nudges, and you can make almost any historical dataset tell you whatever story you want to hear.

Overfitting is what happens when you let the dataset write the story. Underfitting is what happens when you are too lazy to listen to it. The art is sitting in the uncomfortable middle.

> **AVOID THIS**
> Adding a new variable every time your backtest result dips. The next change is rarely the answer. The honest answer is usually that noise is noise.

Out-of-sample testing and holdout periods

The single most useful idea in this whole chapter is older than greyhound betting: never test a model on the same data you used to design it. You want to set some of your history aside, build the model without looking at it, then evaluate it on the set-aside data as if it were the future.

In practice, this means three slices of time, used for different things:

- **Training set.** Where you design features, choose weights, and form the model.
- **Holdout set.** A block of races immediately after the training set, used to test the model once and only once.
- **Live set.** The races you bet on with real money after the model has been frozen.

The holdout is sacred. You only get to look at it once. If you peek, tweak, peek again, you have used it up and you might as well not have bothered. This is a small piece of self-imposed discipline, but it is the difference between an honest backtest and a tidy piece of fiction.

A common pattern is to use something like the most recent six or twelve months as a holdout, build on everything before that, and then rotate the window forward as the live set grows. The exact window does not matter; what matters is that there is always a gap of untouched data between what you used to decide and what you used to judge.

Three stages of evaluation at a glance

Stage	Purpose	Allowed to peek	Risk if abused
Training	Build the model	Yes, freely	Overfitting
Holdout	Honest test	Once, then freeze	Self-deception
Live	Real money	Continuously logged	Confirmation bias

Watching for too many variables and tweaks

Every variable in your model is a small bet that it will keep working tomorrow. Some of those bets are well-founded. Most of them, in the early days of a model, are guesses dressed up as features. A reasonable rule of thumb, and one that keeps you honest, is to ask how many races you have per variable in your training set. If you have fifty races for every coefficient you are estimating, you are probably fine. If you have eight races per coefficient, you are mostly fitting the noise.

This is also where you need to be honest about *leaks*. A leak is any feature that uses information you would not actually have known at the time you would have placed the bet. Things like final finishing positions, post-race sectional times, or weights taken from the race result itself are obvious leaks. Less obvious ones include using the dog's *final* grade at the end of a season to predict races earlier in that season, or accidentally including a race that took place *after* the race you are trying to predict.

Two practical habits help here. First, write down, in plain English, exactly what each variable represents and where it came from. If you cannot describe it in a sentence without using the word "after", it is probably leaking. Second, when you remove a feature, see whether the model still works. Robust features survive being removed. Fragile features collapse, and that is information.

> **GOOD TO KNOW**
> A model with fewer variables that you understand deeply will usually beat a model with many variables that you half-understand. Interpretability is not a luxury; it is your first defence against overfitting.

Treating model changes like small experiments

This is the part of the chapter where you take on the role of your own sceptical colleague. Every change you make to a working model should be treated as a small experiment with a hypothesis, a procedure, and a verdict.

Before you add a feature, remove one, or change a weight, write down three things. What do you expect to happen? How will you measure whether it did? How many races do you need before you trust the answer? If you cannot answer those three questions in writing, you are not running an experiment; you are fiddling.

A useful structure looks like this:

Pre-change experiment checklist

- [ ] Write the change in one sentence, including the data field it uses
- [ ] State the hypothesis: what you expect to improve, and by roughly how much
- [ ] Pick the sample size: minimum races on the holdout before any verdict
- [ ] Decide the rule in advance for what counts as success or failure
- [ ] Agree with yourself that you will not repeat the test if it fails the first time

The last point matters more than it sounds. If you only count a change as real when it works and quietly revert it when it does not, you have not run an experiment; you have run a search. Searches look like progress on paper and behave like a slot machine in practice.

Consider a worked example. Imagine you wonder whether the number of days since a dog's last race helps predict winners. Your hypothesis is that dogs racing too frequently, or after long breaks, underperform. You have a working model that does not use this feature, and you propose adding it. You commit to testing it on a holdout of one thousand races, you set a rule that you will only keep the feature if it improves hit rate at the top of the betting list, and you decide in advance what minimum improvement you are willing to trust.

Run it. Look at the answer once. Make the call. Move on. That is the entire discipline.

The numbers above are a made-up illustration, not a study, but they show the shape of what tends to happen when variables pile up. Hit rate climbs as you add genuinely useful information, plateaus, and then drops as the model starts fitting noise. The danger zone is the right-hand end of the chart, where each new tweak looks like it is helping until you test it on untouched data and it does not.

The unseen work: tracking your changes

One of the cheapest habits you can adopt is a model log. Every change goes in, with the date, the reason, the sample size, and the result. It does not need to be fancy; a simple spreadsheet is enough. The point is that six months from now, when something starts to drift, you will be able to look back and see what you did, when, and what you expected.

This log is also your defence against a very common form of self-deception, which is *remembering the wins and forgetting the losses*. The brain is a poor record keeper for tiny disappointments. A spreadsheet is excellent.

Questions to answer before any change

1. What exactly am I changing, and what does the new feature measure?
2. Is every input something I would have known before the race started?
3. How many races will I need on untouched data before I trust the result?
4. What is the smallest meaningful improvement I would consider worth keeping?
5. If this change fails, will I revert it without arguing with myself?
6. What is the simplest version of this change I could test first?

Knowing when the right move is to change nothing

This is the hardest lesson, and it is the one that separates long-term quantitative bettors from people who keep chasing the same edge across the calendar. Some weeks, the right answer is to leave the model alone. Some months, the right answer is the same.

The signs that you are about to change the model for the wrong reasons are easy to spot once you have seen them a few times. You have just had a quiet week. You have read a forum thread about a factor someone swears by. You are bored. You want a feeling of progress. None of those are reasons to touch a working system.

The signs that you should change something are also recognisable. Your holdout result has drifted below your live result for long enough that the gap is no longer plausible. You have noticed a real, documented change in the racing product, like a new track surface, a new grading policy, or a shift in the calendar. You have a specific, testable hypothesis with a sample size in mind. Those are reasons.

A small but useful practice is to set a minimum interval between changes. A model that only gets revisited every two or four weeks is, almost by construction, harder to overfit than one you tweak daily. It also forces you to let changes breathe. A new variable that looks brilliant on week one and disappointing on week three might just be unlucky on week three. You owe it the dignity of a longer sample before you decide.

> **TRY THIS**
> Schedule a fixed weekly or fortnightly window for model review. Outside that window, the only changes you make are emergency fixes for clear, documented data problems. Everything else can wait.

A gentle iteration cycle

If you want a structure to lean on, the cycle below is simple and it works. You do not have to follow it slavishly, but it gives you somewhere to put your energy on the days when you are tempted to tweak for the sake of it.

The iteration cycle

> **Freeze**
> Decide the current model is the version you will bet with for the next period
>
> **Observe**
> Log live results without changing anything, and note surprises
>
> **Hypothesise**
> Write down at most one or two changes you are considering, and why
>
> **Test**
> Run each hypothesis on the holdout using your pre-agreed sample size and rule
>
> **Decide**
> Adopt, reject, or park the change based on the result, not on a feeling
>
> **Record**
> Log the decision, the reason, and the sample size in your model log

The word "small" is doing a lot of work in this chapter. Big changes to a working model are almost always a mistake, because they trade a system you understand for one you do not, and they often coincide with exactly the sort of confidence that overfitting feeds on. Tiny changes, tested honestly, accumulated over months, are how most quantitative models quietly improve.

Putting it together

You are not trying to build the perfect model. You are trying to build a process that lets you improve a good model slowly, without fooling yourself along the way. The four ideas in this chapter are how you do that: keep a clean holdout you only look at once; be suspicious of every new variable and every tweak; treat each change as a small experiment with a written rule; and accept that, most weeks, the most disciplined thing you can do is leave well enough alone.

Greyhound racing will keep producing noise. The dogs will keep interfering with each other, the breaks will keep being uneven, the grades will keep being reshuffled. Your job is to make sure your model is listening to the signal underneath all that, not to the noise on top of it.

> **KEY TAKEAWAYS**
>
> - An honest backtest needs an untouched holdout you only evaluate once; peeking uses it up.
> - Every new variable is a bet on the future, so keep their number small relative to your races.
> - Treat each change as a written experiment with a hypothesis, a sample size, and a verdict rule.
> - Keep a simple model log so future you can see what you changed, when, and why.
> - The most underrated skill in quantitative betting is the discipline to change nothing for a while.

That leaves one final piece: turning everything from the first fifteen chapters into a practical, week-by-week plan you can actually follow. That is what the closing chapter is for.

CHAPTER 16

Your Quantitative Betting Plan

The whole point of this book has been to replace hunches with a process you can explain to yourself on a quiet Tuesday evening, six months from now. A quantitative greyhound betting system isn't a magic rating that prints money. It's a habit: a way of capturing data, testing ideas and adjusting when reality pushes back. This chapter is about turning everything you've read into a working weekly rhythm, with clear checkpoints for when to keep going and when to step back.

From a Bookshelf to a Betting Workflow

Most bettors who read a book like this feel energised for a few days, then quietly drift back to the old habits: a tipster's text, a flashy card, a quick bet on the favourite. The drift isn't because the ideas were wrong. It's because there was no plan for the boring middle bit - the week after the week after you set everything up.

So before any checklist, a single reminder. The numbers you'll be working against are not on your side. A standard six-runner UK greyhound race carries roughly

> **125%** Approximate overround (bookmaker's built-in margin) on a standard six-runner UK greyhound race, equivalent to a ~25% structural edge for the bookmaker that any quantitative plan must overcome.
> Source: How to Win Greyhound Betting

of overround, meaning the bookmaker's take is baked into the prices before your dog even leaves the trap. Your system has to be good enough, consistently enough, to grind through that margin and still leave something for you. There is no clever weekend that fixes that. Only weeks of careful work do.

Treat the rest of this chapter as the operating manual for your system. It isn't a list of "best practices" borrowed from somewhere else. It's the routine that lets everything from Chapters 1 to 15 actually pay rent.



The First Thirty Days: Setting Up Properly

The temptation, after reading a book on quantitative betting, is to bet on day one. Resist that. The first thirty days are about getting the plumbing right so that every bet you place from day thirty-one onwards is based on a dataset you trust.

Break it into four rough weeks. Week one is about data. Week two is about your stack. Week three is about the model itself. Week four is about going live - carefully. The checklist below is deliberately specific. Print it, pin it up, tick it off.

Thirty-day setup checklist

- [ ] **Week 1 - Data sources:** Pick one primary results feed (paid) and one free verification feed (the official racing regulator site works well). Confirm coverage of the tracks and grades you actually want to bet on.
- [ ] **Week 1 - Historical depth:** Decide on a minimum history window (three years is a sensible starting point). Note any gaps or months where the feed looks thinner than usual.
- [ ] **Week 1 - Storage:** Choose a simple, durable store - a local SQLite database, a CSV folder with a clear naming scheme, or a small cloud database. Avoid anything that requires a developer to maintain.
- [ ] **Week 2 - Tooling:** Settle on one analysis environment (a spreadsheet, a notebook environment, or a lightweight script) and stick with it for at least the first quarter.
- [ ] **Week 2 - Schedule:** Set up an automated pull, or a fixed daily slot, so results land in your system within an hour of the final race.
- [ ] **Week 2 - Validation rules:** Write down three sanity checks you'll run on every new file (for example: row counts match the card, no missing trap numbers, no duplicate races).
- [ ] **Week 3 - Model skeleton:** Build the simplest possible rating first, even if it's just a moving average of recent finishing positions. Don't add complexity until the simple version is logged and tracked.
- [ ] **Week 3 - Calibration baseline:** Note the favourite strike rate at your chosen venues. If your model can't beat ~33%Strike rate of favourites in graded UK greyhound races (ranging 31.6%–42% by venue), used to calibrate whether the bookmaker's implied probability is fair.Source: How to Win Greyhound Betting on favourites in graded races, something is wrong with the inputs or the maths.
- [ ] **Week 4 - Paper betting:** Place a minimum of fifty "paper" bets through your model without real money. Record the price you would have taken, the stake, and the reasoning in one line.
- [ ] **Week 4 - Stake rules:** Decide your unit size (often 1-2% of bank) and write it down. Do not change this during the first three months of real betting.
- [ ] **Week 4 - Logging:** Set up a bet log: date, race, selection, price, stake, model probability, market probability, result, profit and loss.

By day thirty, you should have: a dataset, a working stack, a logged model, a paper-betting trail and a stake plan in writing. If you don't have all five, don't go live yet. The book isn't going anywhere.

A Weekly Review That Actually Gets Done

The biggest threat to a quantitative system isn't a bad model. It's a missed weekend. Three unlogged weeks in a row and you lose the thread entirely. The trick is to make the review small enough that it survives a busy life.

Set aside roughly ninety minutes across seven days. The plan below assumes you're betting primarily in the evenings and reviewing in shorter chunks. Adjust the order to suit your own week, but keep the structure.

Weekly review routine

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Pull the weekend's results into your dataset. Run the three validation rules and flag anything odd.	Generate the next seven days' cards. For each meeting, list the races your system flags as value, with model price vs market price.	Place midweek bets. Log every bet before placing it, even if the price moves by a tick or two.	Settle midweek bets. Update running profit and loss and a simple moving strike rate for the past four weeks.	Refresh cards for the weekend. Look for races where your edge appears largest, not just where the races are biggest.	Place weekend bets. Keep stakes identical to the plan - no "this feels bigger" upsize.	Settle the week. One page of notes: what the model got right, what it missed, one thing to investigate next week.

Sunday's note is the most important part. It's how you avoid the trap of remembering only the winners and forgetting the quiet losses that tell you where the model is failing.

The Monthly Audit and Reset

Weekly reviews keep you current. Monthly audits keep you honest. A month is long enough that noise has a chance to settle, but short enough that bad habits don't get cemented.

Run the audit on the first weekend of each new month, ideally before that month's cards are priced. Twelve months of the same audit creates a calendar you can actually plan your year around.

Monthly model audit and reset

January Review overall profit and loss for the year. Recalibrate stake size if bank has materially changed.	**February** Check feature importance: which inputs are pulling their weight, which are dead weight?	**March** Audit trap biases. Confirm Trap 1 is still running at roughly 18–19%Win rate for Trap 1 in UK greyhound races versus the theoretical 16.6% baseline, quantifying the trap bias signal that can be fed into a value-betting model.Source: How to Win Greyhound Betting across your dataset, not just at one venue.	**April** Compare your model's implied probabilities to the market's implied probabilities. Look for systematic gaps.
May Re-run a small backtest on the past four weeks only. Did the model behave as the backtest predicted?	**June** Review data quality: missing trap draws, missing sectional times, late results. Patch anything noisy.	**July** Stress-test stake rules against the worst losing run in the dataset. Would you have stayed the course?	**August** Check for overfitting by hiding the most recent month and seeing if the model still flags the same races.
September Audit venue-level performance. Are some tracks genuinely harder for your model than others?	**October** Re-read Chapters 12 and 15. Force yourself to question any "improvement" you added in the last ninety days.	**November** Compare realised strike rate to model probability. A 20% shot should win around 20% of the time across a sample, give or take.	**December** Pause live betting for two days. Plan next year's data budget, tools and learning goals.

If a particular month keeps flagging the same problem - say, your strike rate is off at one track - that becomes a focused piece of work for the following month rather than a vague worry.

When to Pause, Refine or Scale Back

This is the chapter where the book gets uncomfortable, because the right answer to "should I keep betting?" is sometimes "no, not this week." A system that won't be paused is a system that will eventually be destroyed by a bad run it couldn't ride out.

The three signals below aren't the same. Pause is a short, sharp stop. Refine is a planned, documented change. Scale back is a permanent reduction in risk while you rebuild confidence. Treating them as interchangeable is how bettors talk themselves into ruin.

Signs you should pause, refine or scale back

Signal	Likely cause	Action	How long
Three losing sessions in a row, but model probabilities and market probabilities still line up as expected	Normal variance	Pause	24-48 hours, then resume at plan stakes
Strike rate well below model probability for two full weeks	Possible overfitting or stale weights	Refine	Switch to paper betting for one week, retest, then return live
Bank has dropped by more than 20% from peak	Stake plan too aggressive, or model broken	Scale back	Halve unit size for one full month, regardless of results
Validation rules failing regularly (missing data, late results)	Feed or pipeline issue	Pause	Until three clean weeks of data in a row
You can't explain why a feature is in the model	Added complexity without justification	Refine	Remove the feature and retest over a month before reinstating
Most bets are clustered on short-priced favourites	Model is overweighting the market	Scale back	Reduce stakes until value bets at longer prices are also being taken
You start looking for "extra" bets outside the model	Erosion of discipline	Pause	Two full weekends off betting, then restart cold

The column on the right is doing a lot of work. A pause without a defined length becomes a drift. A refinement without a test becomes a tweak. A scale-back without a return plan becomes a retirement. Pin this table somewhere you'll actually see it on a bad day.

Building the Habit

The hardest part of a quantitative system isn't the maths. It's the discipline to do the boring review on the Sunday after a bad Saturday, when the temptation is to chase, abandon or invent a new rule. The thirty-day setup, the weekly review and the monthly audit aren't there to make you a better mathematician. They're there to make sure that when your edge does appear, you're still standing to take it.

A few practical notes before you start. First, keep the system boring on purpose. If it feels exciting, you've probably added something that won't survive contact with a quiet February. Second, log everything. The most useful discoveries tend to come from notes you wish you'd written six months earlier. Fourth, accept that you will lose weeks. A system that never has a bad week is a system that simply hasn't been traded for long enough.

Finally, remember the line from the start of the book. This isn't about finding the best dog. It's about building a system for deciding what the numbers are actually worth. That system is now in your hands, in seven parts: a thirty-day setup, a weekly review, a monthly audit, three pause-and-pivot triggers, a stake plan and a log. Run them, in that order, for a year. The greyhounds will still be running. The market will still be mispricing some of them. The only question is whether you'll be the one with a process in place when it does.

KEY TAKEAWAYS

- The first thirty days are for setup, not betting. No live money until data, stack, model, paper trail and stake rules are all in place.
- A weekly review of about ninety minutes keeps the system current; a Sunday note is the single highest-leverage habit.
- A monthly audit turns vague unease into specific, fixable problems and stops small issues from drifting into big ones.
- Pause is short and sharp. Refine is planned and tested. Scale back is a deliberate reduction in risk. Keep them separate.
- The discipline to follow the routine on a bad week is what eventually separates a working edge from a fondly remembered spreadsheet.