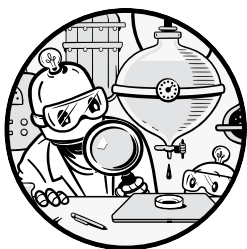


# 2

## THE 80/20 PRINCIPLE



In this chapter, you'll learn about the profound impact of the *80/20 principle* on your life as a programmer. It has many names, including the *Pareto principle*, named after its discoverer Vilfredo Pareto. So, how does the principle work, and why should you care? The 80/20 principle refers to the idea that a majority of effects (80 percent) come from a minority of causes (20 percent). It shows you a path to achieve many more results as a professional coder by focusing your efforts on a few important things and ignoring the many things that hardly move the needle.

### 80/20 Principle Basics

The principle says that the majority of effects come from the minority of causes. For example, the majority of income is earned by the minority of people, the majority of innovations come from the minority of researchers, the majority of books are written by the minority of authors, and so on.

You may have heard about the 80/20 principle—it's everywhere in personal productivity literature. The reason for its popularity is two-fold. First, the principle allows you to be relaxed and productive at the same time, as long as you can figure out the things that matter, which make up the 20 percent of activities that lead to 80 percent of the results, and focus on those relentlessly. Second, we can observe the principle in a huge variety of situations, giving it considerable credibility. It's even tough to come up with a counterexample, where the effects come equally from the causes. Try to find some examples of 50/50 distributions where 50 percent of the effects come from 50 percent of causes! Sure, the distribution is not always 80/20—the concrete numbers can change to 70/30, 90/10, or even 95/5—but the distribution is always heavily skewed toward the minority producing the majority of effects.

We represent the Pareto principle in a Pareto distribution, shown in Figure 2-1.

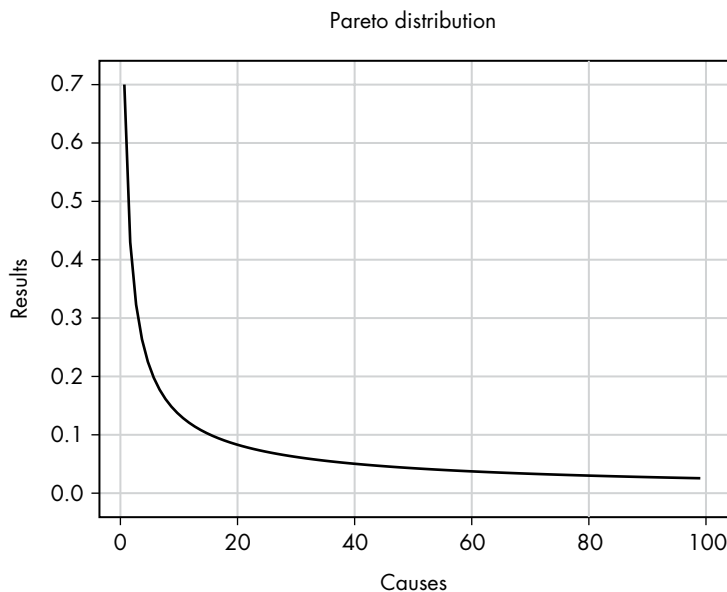


Figure 2-1: Example of a general Pareto distribution

The Pareto distribution plots the results (y-axis) against the causes (x-axis). The results can be any measure of success or failure, like income, productivity, or the number of bugs in a software project. The causes can be any entity these results may be associated with, such as employees, businesses, or software projects, respectively. To obtain the characteristic Pareto curve, we order the causes according to the results they produce. For example, the person with the highest income comes first on the x-axis, then comes the person with the second-highest income, and so on.

Let's look at a practical example.

## Application Software Optimization

Figure 2-2 shows the Pareto principle in action in an imaginary software project: the minority of the code is responsible for the majority of the runtime. The x-axis shows code functions sorted by the runtime they incur. The y-axis shows the runtime of those code functions. The shaded area that dominates the overall area under the plot shows that most code functions contribute much less to the overall runtime than a few selected code functions. Joseph Juran, one of the early discoverers of the Pareto principle, calls the latter the *vital few* and the former the *trivial many*. Spending a lot of time optimizing the *trivial many* barely improves the overall runtime. The existence of Pareto distributions in software projects is well supported by scientific evidence such as in “Power Laws in Software” by Louridas, Spinellis, and Vlachos (2008).

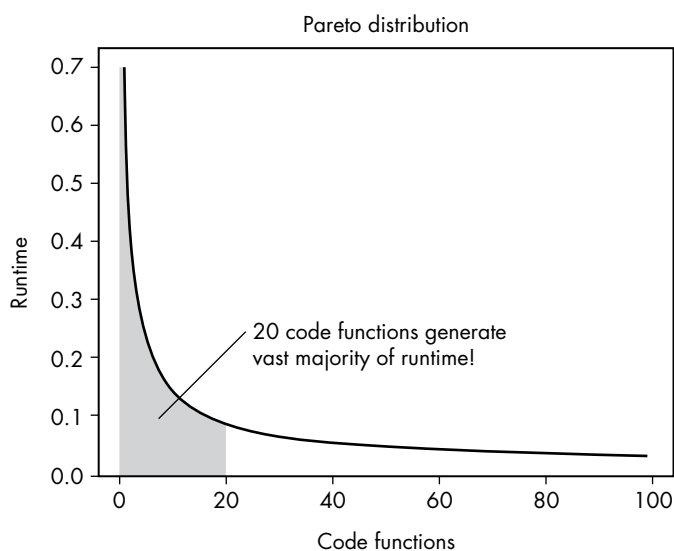


Figure 2-2: Example of a Pareto distribution in software engineering

Large companies like International Business Machines Corporation (IBM), Microsoft, and Apple employ the Pareto principle to build faster, more user-friendly computers by channeling their focus on the vital few; that is, by repeatedly optimizing the 20 percent of the code that was executed most often by the average user. Not all code is created equal. A minority of code has a dominating impact on the user experience, while much of the code has little impact. You might double-click the File Explorer icon multiple times per day, but you seldom change the access rights of a file. The 80/20 principle tells you where to focus your optimization efforts!

The principle is easy to understand, but it can be harder to know how you can use the principle in your own life.

## Productivity

By focusing on the vital few rather than the trivial many, you can increase your productivity by a factor of 10, even 100. Don't believe me? Let's calculate where these numbers come from, assuming an underlying 80/20 distribution.

We'll use the conservative 80/20 parameters (80 percent of the results come from 20 percent of the people) and then calculate the rate of production for each group. In some fields (like programming), the distribution is probably much more skewed.

Figure 2-3 shows that in a company of 10 employees, just 2 employees produce 80 percent of the results, while 8 employees produce 20 percent of the results. We divide 80 percent by two employees for an average output of 40 percent per top-performing employee in the company. If we divide the 20 percent of the results generated by the eight employees, we get an average of 2.5 percent of output per bottom-performing employee. The difference in performance is 16 times!

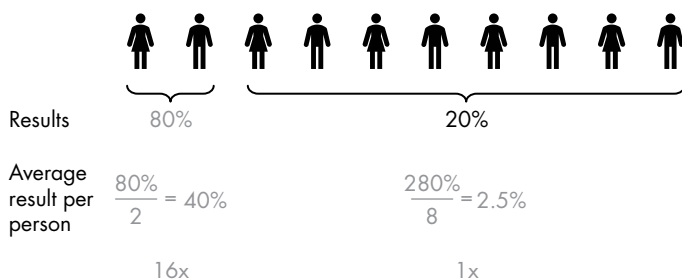


Figure 2-3: The average output of the 20 percent top performers is 16 times the average output of the 80 percent bottom performers.

This 16-times difference in average performance is a fact in millions of organizations throughout the world. The Pareto distribution is also fractal, which means that the top 20 percent of the top 20 percent generate 80 percent of 80 percent of the results, which causes many more significant performance differences in large organizations with thousands of employees.

The differences in results cannot be explained by intelligence alone—a person cannot be 1000 times more intelligent than another person. Instead, the differences in results come from the specific behavior of the individual or the organization. If you did the same things, you could get the same results. However, before you change your behavior, you must be clear about what result you want to accomplish, as research shows an extreme inequality of results in almost any metric you can imagine.

**Income** Ten percent of the people earn almost 50 percent of the income in the United States.

**Happiness** Less than 25 percent of the people in North America rate themselves at 9 or 10 in a happiness scale that ranges 0–10 points in “a 0 to 10 scale, with the worst possible life as a 0 and the best possible life as a 10.”

**Monthly active users** Just two websites of the top 10 websites get 48 percent of the cumulative traffic, as shown in Table 2-1 (based on information from *www.ahrefs.com*).

**Book sales** Just 20 percent of the authors may receive as much as 97 percent of sales.

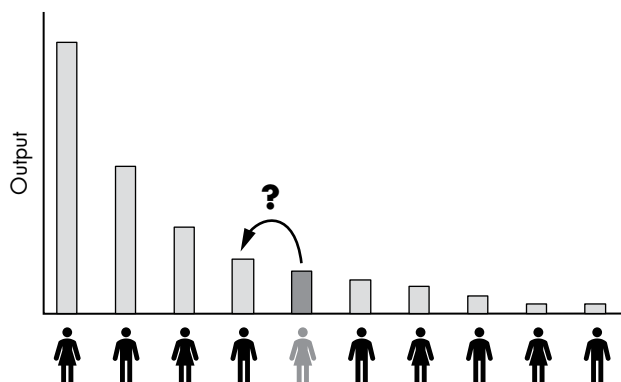
**Scientific productivity** For example, 5.2 percent of scientists account for 38 percent of published articles.

The resources section at the end of the chapter links to some articles to support this data. The inequality of results is a well-established phenomenon in social science, and it is commonly measured in a metric called *Gini coefficient*.

**Table 2-1:** Cumulative Traffic of the Top 10 Most-Trafficked Websites in the United States

#	Domain	Monthly traffic	Cumulative
1	en.wikipedia.org	1.134.008.294	26%
2	youtube.com	935.537.251	48%
3	amazon.com	585.497.848	62%
4	facebook.com	467.339.001	72%
5	twitter.com	285.460.434	79%
6	fandom.com	228.808.284	84%
7	pinterest.com	203.270.264	89%
8	imdb.com	168.810.268	93%
9	reddit.com	166.277.100	97%
10	yelp.com	139.979.616	100%
		4.314.988.360	

So how can you become one of the top performers? Or, to formulate it more generally: how can you *move to the left* on the Pareto distribution curve in your organization (see Figure 2-4)?



**Figure 2-4:** To create more output, you need to move to the left of the curve.

## Success Metrics

Let's say you want to optimize for income. How can you move to the left in the Pareto curve? We're leaving exact science behind here, because you need to find the reasons some people succeed in your specific industry and develop actionable *success metrics* you can control and implement. We define the term *success metric* as measurements of behavior that lead to more success in your field. The tricky thing is that the most crucial success metrics are different in most fields. The 80/20 principle also applies to success metrics: some success metrics have a dominating impact on your performance in a field, while others barely matter at all.

For example, when working as a doctoral researcher, I soon realized that success was all about getting cited by other researchers. The more citations you have as a researcher, the more credibility, visibility, and opportunities you'll have. However, *increasing the number of citations* is hardly an actionable success metric that you can optimize daily. The number of citations is a *lacking indicator*, because it is based on actions you took in the past. The problem with lacking indicators is that they record only the consequence of past actions. They don't tell you the right actions to take daily for success.

To obtain a measure for doing the right actions, the notion of leading indicators was introduced. A *leading indicator* is a metric that predicts a change in the lacking indicator before it occurs. If you do more of the leading indicator, the lacking indicator is likely to improve as a result. As a researcher, then, you'll receive more citations (lacking indicator) by publishing more high-class research papers (leading indicator). That means writing high-class papers is the most important activity for most scientists, not secondary activities such as preparing presentations, organizing, teaching, or drinking coffee. The success metric for researchers is therefore generating a maximal number of high-quality papers, as shown in Figure 2-5.

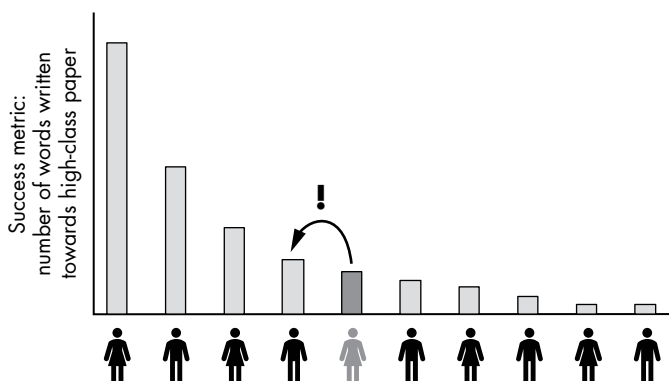


Figure 2-5: Success metric in research: number of words written toward a high-class paper

To push to the left in research, you must write more words today, publish your next high-class paper sooner, receive more citations faster, grow your scientific footprint, and become a more successful scientist. Roughly speaking, many different success metrics can serve as a proxy for “being successful in science.” For instance, when ordering them on a scale from lack to lead measures, you may get *number of citations*, *number of high-class papers written*, *number of total words written in your life*, and *number of words written today*.

The 80/20 approach allows you to identify the activities on which you must focus. Doing more of the success metrics, preferably actionable lead measures, will increase your professional success, and that’s all that should matter. Spend less time on all the different tasks. Refuse to die the death of a thousand cuts. Be lazy with all activities but one: *writing more words per day*.

Say you work 8 hours per day, and you spread your day into eight 1-hour activities. After completing the success metric exercise, you realize that you can skip two 1-hour activities per day and complete four other activities in half the time by being less perfectionistic. You have saved 4 hours per day, but you still accomplish 80 percent of your results. Now you can invest 2 hours into writing more words toward high-class papers per day. Within a few months, you’ll have submitted an extra paper, and over time, you’ll submit more papers than any of your colleagues. You work only 6 hours per day, and you generate imperfect quality in most of your work tasks. But you shine on where it matters: you submit more research papers than anyone else in your environment. As a result, you’ll soon be one of the top 20 percent of researchers. You generate more with less.

Instead of becoming a “Jack of all trades, master of none,” you gain expertise in the area that is most important to you. You heavily focus on the vital few and ignore the trivial many. You lead a less stressful life, but you enjoy more fruits from your invested labor, efforts, time, and money.

## Focus and the Pareto Distribution

A closely related topic I want to discuss is *focus*. We’ll discuss focus in many places in this book—for example, Chapter 10 discusses the power of focus in detail—but the 80/20 principle explains *why* focus is so powerful. Let’s dive into the argument!

Consider the Pareto distribution in Figure 2-6 that shows the percentage improvement of moving toward the top of the distribution. Alice is the fifth most productive person in the organization. If she just overtakes one person in the organization, thereby becoming the fourth most productive person, she’d increase her output (salary) by 10 percent. One step further than that, and her output increases by an *additional* 20 percent. In a Pareto distribution, the growth per rank explodes exponentially, so even small increases in productivity can result in big increases in income. Increasing your productivity leads to superlinear improvements in your income, happiness, and joy at work. Some call this phenomenon “the winner takes all.”

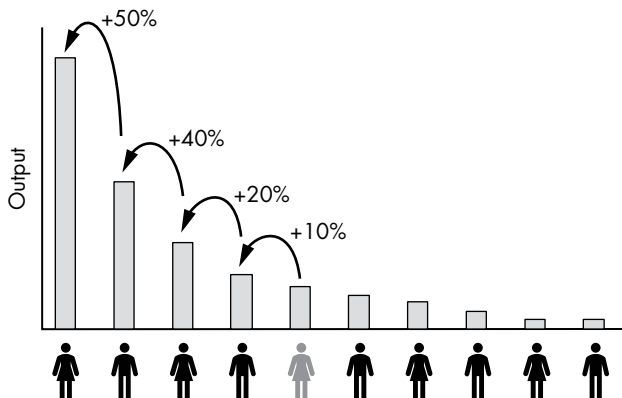


Figure 2-6: Disproportional benefit of improving your rank in a Pareto distribution

That’s why it doesn’t pay to spread your attention: *if you don’t focus, you participate in many Pareto distributions.* Consider Figure 2-7: Alice and Bob can each invest three units of learning efforts every day. Alice focuses on one thing: programming. She just spends her three units of effort in learning to code. Bob spreads his focus to multiple disciplines: one unit of time polishing his chess skills, one unit improving his programming skills, and one unit improving his political skills. He’s reached average skills and output in each of the three areas. But the Pareto distribution disproportionately rewards the top performers, so Alice collects more total output reward.

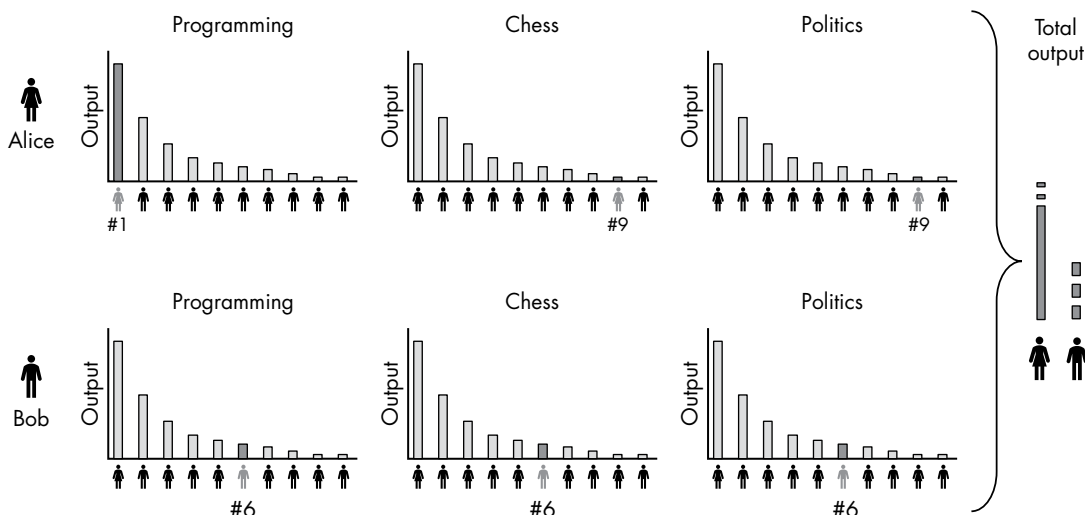


Figure 2-7: Non-linearity of rank output—a strategic explanation attempt for the power of focus

The disproportional rewards hold within each area, too. For instance, Bob may spend his time reading three general books (let’s call them



*Introduction to Python*, *Introduction to C++*, and *Introduction to Java*) while Alice reads three books diving deep into machine learning with Python (let's call them *Introduction to Python*, *Introduction to Machine Learning with Python*, and *Machine Learning for Experts*). As a result, Alice will focus on becoming a machine learning expert and can demand a higher salary for her specialized skill set.

## Implications for Coders

In programming, the results tend to be much more heavily skewed toward the top than in most other fields. Instead of 80/20, the distribution often looks more like 90/10 or 95/5. Bill Gates said a “*great lathe operator commands several times the wage of an average lathe operator, but a great writer of software code is worth 10,000 times the price of an average software writer.*” Gates argues that the difference between a great and an average software writer is not 16 times, but 10,000 times! Here are several reasons why the software world is prone to such extreme Pareto distributions:

- A great programmer can solve some problems that the average programmer simply cannot solve. In some instances, this makes him infinitely times more productive.
- A great programmer can write code that is 10,000 times faster than the code of an average programmer.
- A great programmer writes code with fewer bugs. Think about the effect of a single security bug on Microsoft's reputation and brand! Moreover, every additional bug costs time, energy, and money for subsequent modifications of the codebase and feature additions—the detrimental, compounding effect of bugs.
- A great programmer writes code that is easier to extend, which may improve the productivity of thousands of developers that work on his code at a later stage of the software development process.
- A great programmer thinks out of the box and finds creative solutions to circumvent costly development efforts and help focus on the most important things.

In practice, a combination of these factors is at play, so the difference can be even higher.

So, for you, the key question may be this: *How do you become a great programmer?*

### **A Success Metric for Programmers**

Unfortunately, the statement “become a great programmer” is not a success metric you can directly optimize—it's a multi-dimensional problem. A great programmer understands code quickly, knows algorithms and data

structures, knows different technologies and their strengths and weaknesses, can collaborate with other people, is communicative and creative, stays educated and knows about ways to organize the software development process, and possesses hundreds of soft and hard skills. But you can't be a master of all of those! If you don't focus on the vital few, you'll become washed away by the trivial many. To become a great programmer, you must focus on the vital few.

One of the vital few activities to focus on is to write more lines of code. The more lines you write, the better coder you'll become. It's a simplification of the multi-dimensional problem: by optimizing the proxy metric (write more lines of code), you increase your odds of success at the target metric (become a great writer of software code). See Figure 2-8.

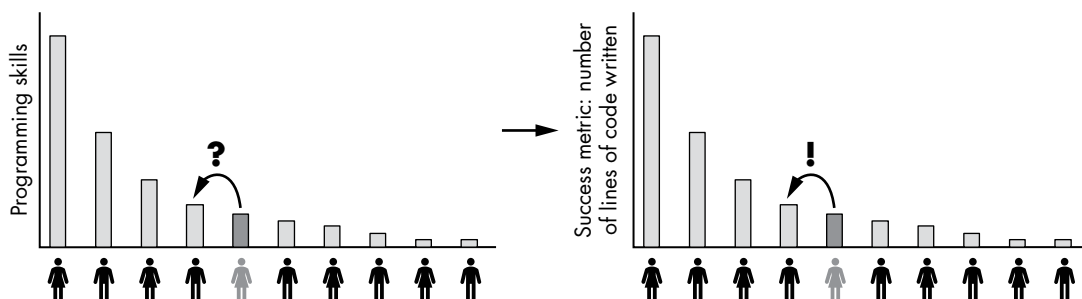


Figure 2-8: Success metric in programming: number of lines of code written

By writing more code, you'll understand code better, and talk and behave more like an expert coder. You attract better coders and find more challenging programming tasks, so you write more code and become even better. You get paid more and more per line of code you write. You or your company can outsource the trivial many tasks.

Here's an 80/20 activity you can follow every day: *track the number of lines you code every day and optimize it*. Make it a game to at least match your average every day.

## ***Pareto Distributions in the Real World***

We'll take a quick look at some real-world examples of the Pareto distribution in action.

### **GitHub Repository TensorFlow Contributions**

We can see an extreme example of a Pareto distribution in contributions to GitHub repositories. Let's consider a wildly popular repository for machine learning computations in Python: *TensorFlow*. Figure 2-9 shows the top seven contributors to this GitHub repository. Table 2-2 shows the same data numerically.

## GitHub TensorFlow Repository Commits

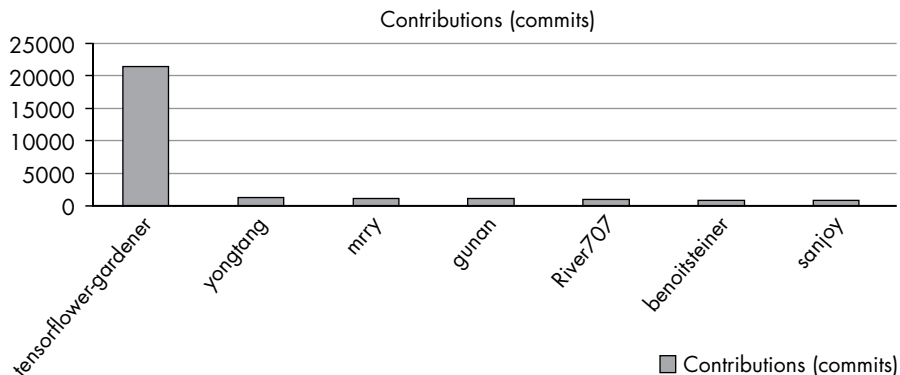


Figure 2-9: GitHub TensorFlow repository commit distribution

**Table 2-2:** Number of TensorFlow Commits and their Contributors

Contributor	Commits
tensorflow-gardener	21426
yongtang	1251
mrry	1120
gunan	1091
River707	868
benoitsteiner	838
sanjoy	795

The user *tensorflow-gardener* contributed more than 20 percent of the 93,000 commits to this repository. Given that there are thousands of contributors, the distribution is much more extreme than the 80/20 distribution. The reason is that the contributor *tensorflow-gardener* consists of a team of coders at Google that created and maintains this repository. Yet, even when this team is filtered out, the remaining individual top contributors are hugely successful programmers with impressive track records. You can check them out on the public GitHub page. Many of them have landed exciting jobs working for very attractive companies. Whether they became successful before or after they generated a large number of commits to the open-source repository is a mere theoretical discussion. For all practical matters, you should start your success habit: write more lines of code every day now. Nothing is stopping you from becoming number 2 on the TensorFlow repository—by committing valuable code to the TensorFlow repository two to three times per day for the next 2–3 years. If you persist, you can join the ranks of the most successful coders on earth just by choosing one powerful habit and sticking to it for a few short years!

## Programmer Net Worth

Sure enough, the net worth of programmers is also Pareto distributed. For privacy reasons, it's hard to get data about an individual's net worth, but the website [www.networthshare.com](http://www.networthshare.com) does show the self-reported net worth of various professions, including programmers. The data is a bit noisy, but it shows the idiosyncratic skewness of real-world Pareto distributions (Figure 2-10).

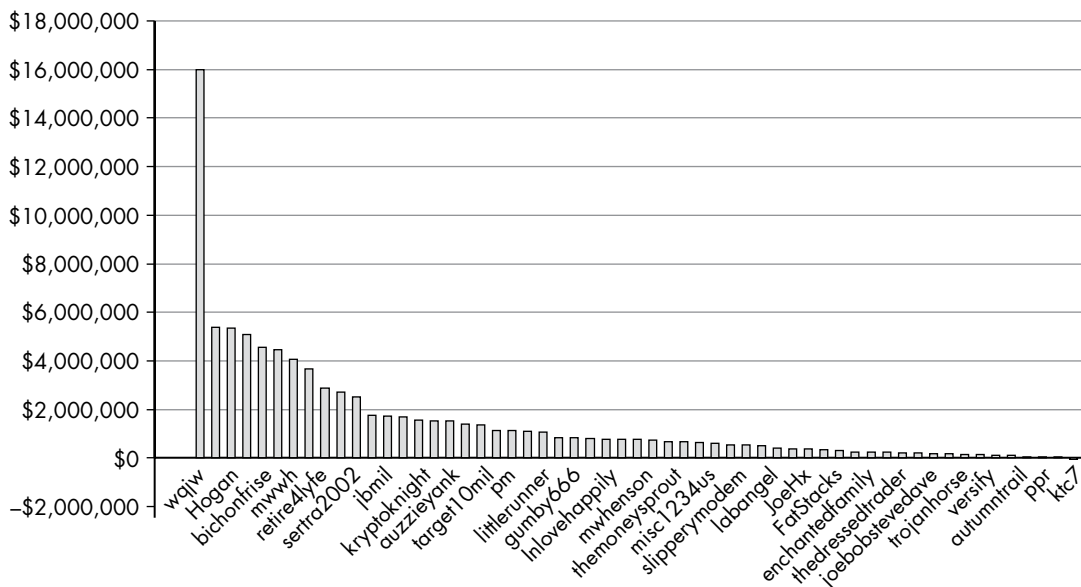


Figure 2-10: Self-reported net worth of 60 programmers

Quite a few software millionaires in our small sample of 29 data points! But the curve is likely to be even more skewed in the real world because there are also many billionaire programmers—Mark Zuckerberg, Bill Gates, Elon Musk, and Steve Wozniak come to mind. Each of those tech geniuses created the prototypes of their services themselves, laying a hand on the source code. Lately, we've seen many more of those software zillionaires in the Blockchain space.

## Freelance Gigs

The freelance developing space is dominated by two marketplaces where freelancers can offer their services and clients can hire freelancers: Upwork and Fiverr. Both platforms grow double digits per year in terms of users and revenues, and both platforms are committed to disrupting the organization of the world's talents.

The average income of a freelance developer is \$51 per hour. But this is only the average rate—the top 10% of freelance developers reach much higher hourly rates. In more or less open markets, income resembles a Pareto distribution.

I have observed this skewed income distribution in my own experience from three perspectives: (1) as a freelancer, (2) as a client hiring hundreds of freelancers, and (3) as a course creator offering Python freelancing education. Most students fail to reach even the average earning potential because they don't stay in the game for more than a month or so. The ones who keep working on their freelancing business for several months daily usually reach the average \$51 per hour earning target. A minority of very ambitious and dedicated students reach \$100 per hour and more.

But why do some students fail while others thrive? Let's plot the number of successful gigs completed by freelance developers on the Fiverr platform with an average rating of at least 4 out of 5. I focused on the popular area of machine learning in Figure 2-11. I collected the data from the Fiverr website and tracked the number of completed gigs for 71 freelancers on the two top search results for the category *Machine Learning Gigs*. Not surprisingly, for us, the distribution resembles a Pareto distribution.

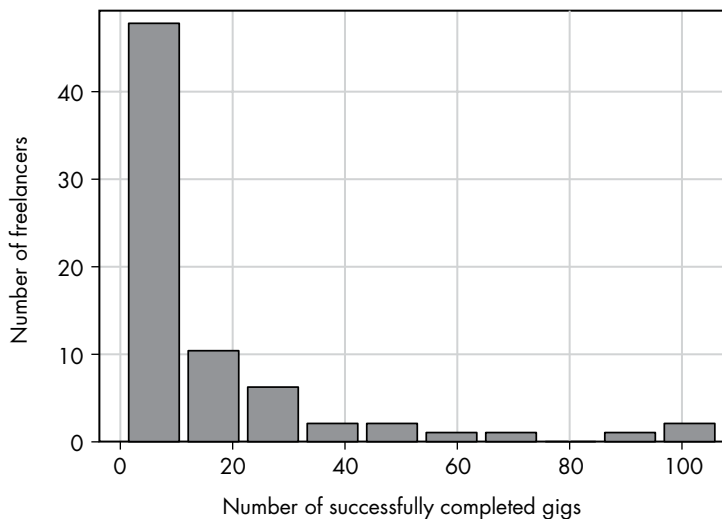


Figure 2-11: Histogram of Fiverr freelancers and the number of gigs they completed

From my own experience as a teacher of thousands of freelancing students, I'm fascinated to see that the vast majority of students have completed fewer than ten gigs. I'm pretty sure that many of those students will later proclaim, "Freelancing doesn't work." To me, this statement is an oxymoron like "work doesn't work" or "business doesn't work." These freelancing students fail because they don't try hard and long enough. They assume that they can make money easily, and when they realize that they must work persistently to join the freelancing winners, they're quick to give up.

This lack of freelancing persistence actually provides an excellent opportunity for you to move up the Pareto distribution. The simple success metric that virtually ensures you eventually join the top 1–3 percent of freelancers is this: *complete more gigs*. Stay in the game longer. Anyone can do this. The fact that you're reading this book shows that you have the commitment, ambition,

and motivation to become a top 1–3 percent freelance coder and coding professional. Most players suffer from a lack of focus, and even if they are very skilled, intelligent, and well-connected, they have no chance of competing against a focused, dedicated, and Pareto-knowledgeable programmer.

## Pareto is Fractal

The Pareto distribution is fractal. If you zoom in, observing only a part of the whole distribution, there's another Pareto distribution! This works as long as the data is not too sparse; in that case, it loses its fractal nature. A single data point, for example, cannot be considered a Pareto distribution. Let's look at this property in Figure 2-12.

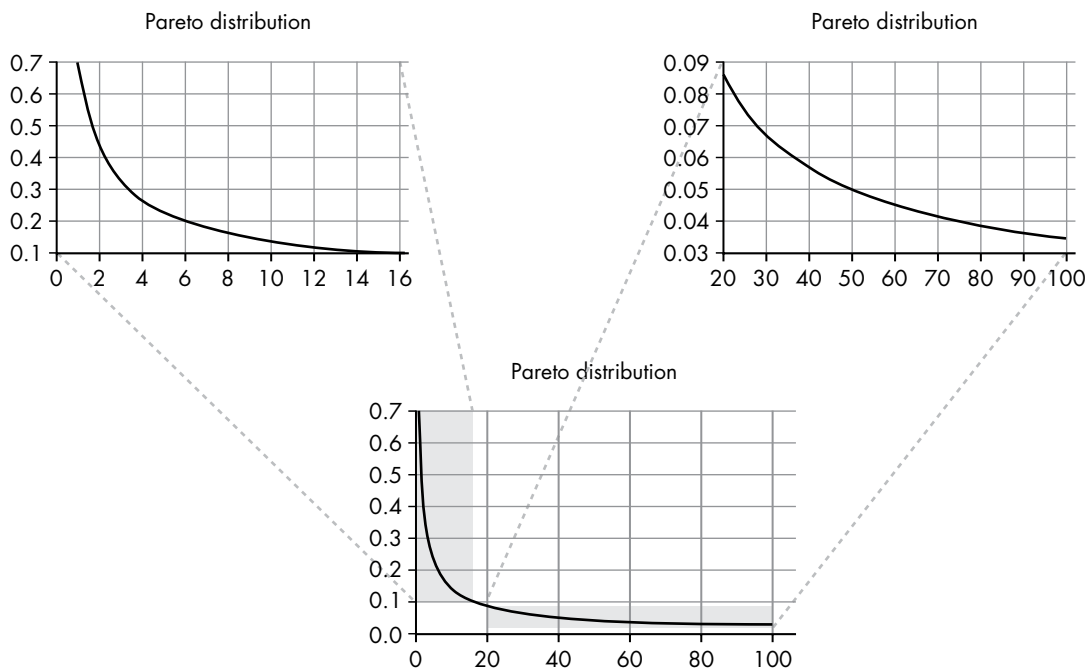


Figure 2-12: The fractal nature of a Pareto distribution

In the center of Figure 2-12 is the Pareto distribution from Figure 2-1. I used the simple Python script in Listing 2-1 to zoom into this Pareto distribution:

---

```
import numpy as np
import matplotlib.pyplot as plt

alpha = 0.7

x = np.arange(100)
y = alpha * x / x**(alpha+1)
```

```
plt.plot(x, y)

plt.grid()
plt.title('Pareto Distribution')
plt.show()
```

---

*Listing 2-1: An interactive script for you to zoom into a Pareto distribution*

You can play with the code yourself; just copy it into your Python shell and run the code. If you do this in your Python shell, you'll be able to zoom into different areas of the Pareto distribution.

The Pareto distribution has various practical applications in life and programming, and I'll discuss some of them in this book, but, in my experience, the most transformative application for you will be to become an *80/20 thinker*, that is, you constantly try to find ways to accomplish much more with much less. Please note that while the concrete Pareto numbers—80/20, 70/30, or 90/10—may vary in your own life, you may draw some value from the fractal nature of productivity and output distributions. For instance, it is always true that not only do a few programmers earn much more than the rest but also a few from those top earners earn more than the rest of the top earners. The pattern stops only when the data gets too sparse. Here are some examples:

**Income** Twenty percent of the 20 percent of coders will earn 80 percent of the 80 percent of income. In other words, 4 percent of the coders will earn 64 percent of the income! This implies that you're never stuck with your current financial situation, even if you already belong to the top 20 percent of coders. (This paper is just one of many that show the fractal nature of income distributions: <http://journalarticle.ukm.my/12411/1/29%20Fatimah%20Abdul%20Razak.pdf>.)

**Activities** Twenty percent of the 20 percent of the 20 percent of the activities you have done this week are often responsible for 80 percent of the 80 percent of the 80 percent of your results. In this scenario, 0.8 percent of the activities will lead to 51 percent of the results. Roughly speaking, if you're working 40 hours per week, 20 minutes may account for half of the results in your work week! An example of such a 20-minute activity would be writing a script that automates a business task and saves you a couple of hours every few weeks that you can invest in other activities. If you're a programmer, deciding to skip the implementation of an unnecessary feature can save you tens of hours of unnecessary work. If you start to apply some 80/20 thinking, you'll quickly find many of those leveraged activities in your own work.

**Progress** No matter where you reside on any Pareto distribution, you can increase your output exponentially by “moving to the left” using your success habit and the power of focus. As long as the optimum hasn't been reached, there's always room for progress, for reaching more with less—even if you're already a highly developed individual, company, or economy.

The activities that can move you up the Pareto curve are not always obvious, but they are never random. Many people give up searching for the success metrics in their fields because they argue that the probabilistic nature of the outcomes makes it completely random. What a wrong conclusion! To become a master coder, writing less code per day won't get you there just as practicing less chess every day cannot lead you to becoming a professional chess player. Other factors will come into play, but that doesn't make success a game of chance. By focusing on the success metrics in your industry, you will manipulate the probabilities in your favor. As an 80/20 thinker, you are the house—and the house *mostly* wins.

## 80/20 Practice Tips

Let's finish this chapter with ten practice tips to leverage the power of the Pareto principle.

### **Figure out your success metrics.**

Define your industry first. Identify what the most successful professionals in your industry are doing exceptionally well and which tasks you can do every day to push you closer toward the top 20 percent. If you're a coder, your success metric may be the number of lines of code written. If you're an author, your success metric may be the number of words written toward the next book. Create a spreadsheet and track your success metric every day. Make it a game to stick to it and surpass yourself. Set a minimum threshold, and don't end the day until you've accomplished the minimal threshold each day. Better yet, don't start the day until you have!

### **Figure out your big goals in life.**

Write them down. Without clearly defined big goals (think: 10-year goals), you won't stick to one thing for a sufficiently long time. You have seen that a critical strategy for moving up the Pareto curve is to stay in the game longer while participating in fewer games.

### **Look for ways to achieve the same things with fewer resources.**

How can you accomplish 80 percent of the result in 20 percent of the time? Can you remove the remaining activities that take 80 percent of the time but lead only to 20 percent of the results? If not, can you outsource them? Fiverr and Upwork are cheap ways to find talent, and it pays to leverage the skills of other people.

### **Reflect on your own successes.**

What did you do that led to great results? How can you do more of those things?



### **Reflect on your own failures.**

How can you do less of the things that are responsible for the failure?

### **Read more books in your industry.**

By reading more books, you simulate practical experience without the massive time and energy investment of actually experiencing it. You learn from the mistakes of others. You learn about new ways of doing things. You acquire more skills in your field. A highly educated expert coder can solve a problem 10–100 times quicker than a beginner can. Reading books in your field is likely to be one of the success metrics in your field that will catapult you to success.

### **Spend much of your time improving and tweaking existing products.**

Do this rather than inventing new products. Again, this comes from the Pareto distribution. If you have one product in your business, you can invest all your energy pushing this one product up the Pareto curve, generating exponentially increasing results for you and your company. But if you create new products all the time without improving and optimizing the old ones, you'll always have subaverage products. Never forget: the big results are found on the left of the Pareto distribution.

### **Smile.**

It's surprising how simple some consequences are. If you're a positive person, many things will be easier. More people will collaborate with you. You'll experience more positivity, happiness, and support. Smiling is a highly leveraged activity with massive impact and little cost.

### **Don't do things that reduce value.**

These are things like smoking, eating unhealthily, sleeping little, drinking alcohol, and watching too much Netflix. Avoiding things that drag you down is one of your biggest leverage points. If you skip doing things that harm you, you'll become healthier, happier, and more successful. And you'll have more time and money to enjoy the good things in life: relationships, nature, and positive experiences.

In the next chapter, you'll learn a key concept that helps you focus on the vital few features of your software: you'll learn how to build a minimum viable product.

## **Resources**

Let's have a look at the sources used in this chapter—feel free to explore them further to find more applications of the Pareto principle!

Panagiotis Louridas, Diomidis Spinellis, and Vasileios Vlachos, "Power Laws in Software," *ACM Transactions on Software Engineering and Methodology* 18, no.1 (September 2008), <https://doi.org/10.1145/1391984.1391986>.

Scientific evidence that contributions to open-source projects are Pareto distributed:

Mathieu Goeminne and Tom Mens, “Evidence for the Pareto Principle in Open Source Software Activity,” Conference: CSMR 2011 Workshop on Software Quality and Maintainability (SQM), (January 2011), [https://www.researchgate.net/publication/228728263\\_Evidence\\_for\\_the\\_Pareto\\_principle\\_in\\_Open\\_Source\\_Software\\_Activity](https://www.researchgate.net/publication/228728263_Evidence_for_the_Pareto_principle_in_Open_Source_Software_Activity).

Source for the commit distribution in the GitHub repository TensorFlow:

<https://github.com/tensorflow/tensorflow/graphs/contributors>.

My blog article on the income distribution of freelance developers:

Christian Mayer, “What’s the Hourly Rate of a Python Freelancer?” *Finxter*, <https://blog.finxter.com/whats-the-hourly-rate-of-a-python-freelancer/>.

Scientific evidence that open markets adhere to the Pareto principles:

William J. Reed, “The Pareto Law of Incomes—an Explanation and an Extension,” *Physica A: Statistical Mechanics and its Applications* 319 (March 2003), [https://doi.org/10.1016/S0378-4371\(02\)01507-8](https://doi.org/10.1016/S0378-4371(02)01507-8).

A paper that shows the fractal nature of income distributions:

Fatimah Abdul Razak and Faridatulazna Ahmad Shahabuddin, “Malaysian Household Income Distribution: A Fractal Point of View,” *Sains Malaysianna* 47, no. 9 (2018), <http://dx.doi.org/10.17576/jsm-2018-4709-29>.

Information about how you can build your side income as a freelance developer with Python:

Christian Mayer, “How to Build Your High-Income Skill Python.” Video, <https://blog.finxter.com/webinar-freelancer/>.

Python Freelancer resource page, *Finxter* (blog), <https://blog.finxter.com/python-freelancing/>.

A deeper dive into the power of 80/20 thinking:

Richard Koch, *The 80/20 Principle: The Secret to Achieving More with Less*, London: Nicholas Brealey, 1997.

Ten percent of the people earn almost 50 percent of the income in the United States.

Facundo Alvaredo, Lucas Chancel, Thomas Piketty, Emmanuel Saez, and Gabriel Zucman, *World Inequality Report 2018*, World Inequality Lab, <https://wir2018.wid.world/files/download/wir2018-summary-english.pdf>.

Less than 25 percent of the people in North America rate themselves with 9 or 10 in a happiness scale that ranges 0–10 points in “a 0 to 10 scale, with the worst possible life as a 0 and the best possible life as a 10.”

John Helliwell, Richard Layard, and Jeffrey Sachs, eds., *World Happiness Report 2016, Update* (Vol. 1). New York: Sustainable Development Solutions Network, ISBN 978-0-9968513-3-6, <https://worldhappiness.report/ed/2016/>.

Twenty percent of the authors may even receive 97 percent of the sales.

Xindi Wang, Burcu Yucesoy, Onur Varol, Tina Eliassi-Rad, and Albert-László Barabási, “Success in books: predicting book sales before publication,” *EPJ Data Sci.* 8, no. 31 (October 2019), <https://doi.org/10.1140/epjds/s13688-019-0208-6>.

Jordi Prats, “Harry Potter and Pareto’s fat tail,” *Significance* (August 2011) <https://www.significancemagazine.com/14-the-statistics-dictionary/105-harry-potter-and-pareto-s-fat-tail>.

Of scientists, 5.2 percent account for 38 percent of the articles.

Javier Ruiz-Castillo and Rodrigo Costas, “Individual and field citation distributions in 29 broad scientific fields,” *Journal of Informetrics* 12, no. 3 (August 2018), <https://doi.org/10.1016/j.joi.2018.07.002>.

