Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

CHAPTER

4

The Model Development Process

In the previous chapter, you saw how an algorithm works. In this chapter, I will review how an algorithm is developed; this obviously is hugely helpful in understanding the many ways biases can creep into algorithms. Also, seasoned data scientists may want to briefly glance at this chapter so that they are aware of my mental frame and terminology since I will be referencing both frequently going forward. One note on terminology: with the advent of machine learning, a whole new vocabulary has been introduced (e.g., *observations* have become *instances*, *dependent variables* have become *labels*, and *predictive variables* have become *features*), which unfortunately makes it really hard to write something that all generations of data scientists can understand. At least the new job title of *data scientist* is a lot fancier than *model developer* or *modeler*, which is what data scientists used to be called in ancient times (ca. anno 2010)! Apart from the title, I will generally use more traditional terms, mostly for the benefit of those who may have had just a tiny bit of exposure to statistics in other fields of study and for whom it will be easier to connect the dots if I use familiar terms.

© Tobias Baer 2019
T. Baer, *Understand, Manage, and Prevent Algorithmic Bias*,
https://doi.org/10.1007/978-1-4842-4885-0_4

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

You may wonder why the title of this chapter talks about developing a *model* as opposed to an *algorithm*. When data scientists estimate the parameters of an equation to predict a specific outcome in a particular context (e.g., predict the probability of default for a bank's Canadian small business retail portfolio), we typically call the outcome a *model*. Terminology often varies a bit by industry, geography, function (e.g., risk management or marketing), and even organization; some readers (e.g., folks working in banking) will have heard the term *model* (e.g., US banking regulators use this term to regulate banks' algorithms) while others may be used to the term *algorithm* (e.g., folks dealing with specific functions on a website such as a recommendation engine). For our purposes, there is no need to differentiate between the terms *model* and *algorithm*.

Overview of the Model Development Process

On a high level, one can differentiate five major steps in model development: model design, data engineering, model assembly, model validation, and model implementation.

1. *Model design* defines the overall structure of the model, such as what shall go in and what shall come out of it—not unlike the plan an architect makes for the construction of a new house.
2. *Data engineering* prepares the data used to estimate the coefficients of the algorithm. It covers all activities from identifying the data you want to collect (in our architecture analogy, this first substep is placing an order for construction materials) to putting all data neatly into one or more large tables—with most challenges hiding behind the notion of "neatly." (Just think of bathroom tiles—if you want to have that perfect bathroom, the tiler should carefully inspect each tile, dispose of the broken ones, and cut some tiles just to the right length to fit into corners and crevices.)
3. *Model assembly* is the heart of model development. Here the raw data is transformed into an equation, with coefficients derived through statistical techniques.
4. *Model validation* is an independent review and assertion of the model's fitness for use.
5. *Model implementation* is the deployment of the model in actual business operations.

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

Let's discuss each step in a bit more detail, in particular the two steps most important for taming biases: data engineering and model assembly.

Step 1: Model Design

Model design determines the big, fundamental questions about a model. We can call them the "four whats:"

- What outcome is predicted…
- for what kind of business problems…
- based on what kind of data…
- with what kind of methodology?

The answers to these questions are really driven by the needs of the business users and the way they intend to use the model; just as an architect may design a useless building if there is insufficient understanding of the customer's needs, a lot of biases can creep into a model if there is insufficient communication between the data scientist and the business users.

Step 2: Data Engineering

Just as some of the best Parisian chefs pride themselves on shopping the Marché International de Rungis at some ungodly hour to get the best pick of the very best local produce (the market opens at 1 a.m. and closes at 11 a.m.), much of the value created by the data scientist happens in the data engineering phase. While different data scientists use slightly different terminology and groupings for their activities, I find it most useful to distinguish five major elements of data engineering: sample definition, data collection, splitting of samples, (addressing of) data quality, and data aggregation.

- *Sample definition* determines exactly which historical reference cases to collect data on. In your hair example, you picked 200 reference cases—should they all be in your neighborhood or shall you include some data points from a different city or even different country? Should you make sure to include Caucasians, Blacks, and Latinos in certain ratios—a process called *stratification*? What about Pacific Islanders? Should you also stratify by age and gender? And would it be better to collect data on 500, 50,000 or maybe 5 million people? Sample definition quickly gets complex, and any trade-off you make can quickly come back to bite you! Only sales people of some

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

overpriced software tool will claim that this is easy—in my experience, many of the gravest problems of algorithms are rooted in poor sampling.

- *Data collection* is the process of getting the actual data for your sample. In the old days, it often involved the IT department writing queries in COBOL for mainframe computers or retrieving data tapes from the dusty archive in the basement; nowadays the data scientist may just type up a simple query into the data lake. Data may have to be collated from multiple sources; sometimes it even may have to be captured manually (e.g., looked up in paper files and typed into a spreadsheet).
- *Splitting of the sample into development, test, and validation samples* is critical to allow a proper validation of the model (a key technique to ensure that the model functions properly). You are in trouble if you forget this or make a poor choice in the way you split (e.g., leaving you with too little or inappropriate data for validation). The model coefficients are estimated based on the development sample; if it fails to show a similar predictive power on the test sample, the data scientist knows that the model is overfitted to the development sample and hence unstable, and can adjust the model. If there are many iterations, however, the model may become overfitted to the test sample as well, and therefore a separate validation sample that is not touched before the final validation is the ultimate test of the model's stability. A paranoid user may not share the development sample with the data scientist before the model is finished (this is how modeling competitions are run). By contrast, if your data scientist forgets to split the sample at the very beginning, the integrity of the thought process is ruined even if later a validation sample is set aside. This is a bit like reading your child's diary: once you have read it, you have broken her trust, even if you put back the diary on the shelf *exactly* how it was.
- *Data quality* needs to be first assessed and, once the specific issues of the data collected have been identified (e.g., missing data or non-sensical values), achieved through a detested activity called *data cleaning*. For example, if you look at a sample of individuals on November 1, 2018 and find that half of them are exactly 118 years and 10 months old, this is suspicious—and

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

closer investigation might find that for many people in the database the date of birth was not collected but some really old computer system three mergers ago entered January 1, 1900 as a default date of birth in such cases. You then either need to overwrite every incidence of January 1, 1900 with a "missing" indicator, or if you are really lucky, you realize that these are all Chinese citizens for which you also have their ID number where the date of birth happens to be encoded—so you can find out the correct age for each of them and thus *clean up* this data quality issue by overwriting January 1, 1900 with the correct date of birth taken from the ID number. Why is this activity detested, though? In doing your clean-up, you might stumble on the fact that for many other people (with more reasonable dates of birth) the ID shows a different date of birth as well. Annoyed, you might decide to replace the date of birth for *all* people based on their ID. At this point, however, you find several people born in 2058, which at the time of writing of this book is still in the future. An investigation of their ID numbers reveals that the check digit of the ID number is wrong and hence the ID must have been incorrectly captured. You sigh and realize that this will never end and that you *really* hate data cleaning.

- *Data aggregation* combines multiple data items (e.g., individual transactions you have made with a credit card or items in your browser's search history) into new variables. This is a key step that at the same time can create more insightful variables (e.g., it might be more meaningful to know that on average you spend $1,287 per month on food than knowing that yesterday you spent $1.99 at Wholefood Markets—forgot the whipping cream, huh?) and lose information (the fact that in three out of four cases you will return to the same supermarket within five hours after a major purchase with a transaction value above $50 to make another small purchase worth less than $10 is actually a very important insight that may suggest to a bank that you are a lot more likely to forget paying your credit card than the average customer—so if I aggregate all food purchases to a total amount for each day or month, I lose the information that you're a real scatterbrain).

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

In most model development efforts, data engineering is the most time-consuming work step; as you will see soon, it's also ripe with opportunity to create bias.

Step 3: Model Assembly

Once the data is prepared, data scientists can start to assemble the algorithm. This entails a lot more than just running a statistical software package to estimate coefficients of an equation—in fact, it entails seven substeps. Steps 2, 4, and 5 are the fun part that data scientists typically enjoy the most; it therefore may be no coincidence that they sometimes get carried away with these steps and then run out of time for some of the other steps.

I also should note that *model assembly* is not a very common term; usually you will hear *model estimation*—but my whole point here is that model estimation is just one of seven important steps, and biases often creep into models if the other steps are forgotten or shortchanged.

- *Exclusion of records* based on logical criteria is a key step to prevent bias. Many samples contain hidden garbage that slips through if data scientists don't spend enough time on this step. For example, you naively might assume that to build a credit risk model, you simply look at past loans, classify them as either repaid or defaulted, and build a model. Big mistake! Because of rounding issues, many banks have loan accounts in their books that the customer believed to have completely repaid but that technically still have a cent or so outstanding. If that cent is more than 90 days overdue, the naïve approach will label the account as "defaulted." At the same time, banks might have a sensible operational rule that if a defaulted customer owes them less than a dollar, they don't do anything about it because the cost of chasing a few cents is a lot more than the money owed; instead, they will periodically write off these accounts. Now let's further assume that these rounding issues only occur if the loan amount contains a fraction of $12,000—this is because the bank sets interest rates with three digits after the comma, and 0.001% on $12,000 is exactly $0.01 per month (= $12,000 * 0.001% / 12). Can you see what will happen? A clever algorithm might figure out that the risk of default is much lower than normal if the loan amount is a multiple of $12,000—and thereby create a loophole where some risky customers might slip in and get a loan although for any other loan amount the algorithm would

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

reject them. The data scientist therefore must review the distribution of defaulted balance amounts, identify this issue with immaterial values, and exclude any record with a balance below the operational threshold where the bank would commence collections efforts. It's important to note that unlike data cleaning (which deals with factually wrong data), this step deals with conceptual issues caused by factually perfectly correct data. This step therefore hinges a lot more on domain knowledge and judgment by the data scientist.

- *Feature development* is the process where new variables are created to tease out insights from raw data that can be used as inputs in an algorithm. In the hair example, you encountered the coding of a dummy indicating males (i.e., male = 1, female = 0) as a very simple example. The distance of a mobile phone's location right now from the closest of the top three locations where in the past 12 months it has spent most of its time is an example of a very complex feature. Maybe you are trying to combat online fraud—in this case, you might have the idea that if the mobile phone is far away from its typical location, it is more likely that the phone was stolen and a thief is trying to use it. What is a typical location? Here you have decided to define the top three locations. For that, you first need to create a log of the phone's location for, say, the past 12 months, so you need some form of location data (e.g., from log-in events when the phone connects with a cell phone tower, or possibly more precise locations recorded by an app or sent with search queries), process it to assign a location to each unit of time, and make assumptions for how you deal with gaps in the data (e.g., how you deal with a situation where for two weeks no data was sent at all—it is possible that the person did stay at home (maybe tending to a sick family member) but it is more likely that the phone was broken and sent to a repair shop, or that the person went hiking in the Himalayas—hence you may decide that 12 or 24 hours after the last signal you set the location to "unknown" and exclude time spans with unknown location from the analysis). You then need to aggregate total time spent per location, choose the top three, and calculate the distance of the phone's current location to each of the top three locations to derive the variable you had in mind. And this isn't even the most complicated variable I

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

have ever seen! But it shows three things: it is fun, it is complicated (and hence time-consuming), and it involves a lot of assumptions and judgmental decisions. Remember that last bit—this is obviously where biases come in!

- *Short-listing of variables* is to the columns of the huge data table that data scientists work with what exclusion of records is to the rows: we delete individual variables (i.e., columns—in order to build an algorithm, we usually need to arrange all data in a huge table where each observation (e.g., a person in the sample for which you have observed the amount of hair as well as the predictive attributes) is a row and each predictive variable (either a raw attribute you have collected such as age or a feature you have calculated based on other data) is a column). It is also a step that often is skipped at the data scientist's peril. Many features considered are absolutely useless (i.e., have no predictive value at all) while others might be predictive but redundant because they are very similar to (in the statistician's language: highly correlated with) other features (as an extreme example, a person's weight in kilograms and the weight in pounds are exactly the same information, just expressed in different units). If these variables are kept in the sample, they don't have any benefit but they can wreak all kind of havoc—in extreme cases (such as two perfectly correlated variables like in the weight example) they actually can crash the model estimation procedure (which in a way is the lucky case because then at least the data scientist will notice) but usually they just play all kinds of naughty tricks on the data scientist, and some of that will cause a biased model.
- *Model estimation* is the step of actually estimating the model's coefficient—this is where we might do some matrix algebra for OLS regressions or run scripts in a statistics package to build a gradient-boosted decision tree with XGBoost (i.e., build a very complex model using a tool somebody else has developed so that we *even* could do this without knowing exactly what we're doing…). This can be a lot of fun, especially if we try out some fancy new algorithm we have never used before and the resulting model's performance is 0.0001% better than the standard algorithm we normally use!
- *Model tuning* is a series of iterative steps where the data scientist looks at the initial results, assesses them (e.g., detects an unwanted bias or other unpleasant behavior

of the model), and tries to fix the problem in one of three ways: the data scientist can either select a different set of rows of the data (e.g., belatedly remove accounts with one-cent-balances), or change the columns with the predictive features (e.g., fix a logical flaw in one of the variables), or change some of the parameter settings of the model estimation procedure. The latter are also called *hyperparameters*—just how a baker can adjust the temperature and the humidity in the oven and adjust the baking time, you can think of a model estimation procedure as a machine with a couple of dials and buttons you can play around with to get better baking results. I'll illustrate this with the estimation of decision trees: decision trees can have a tendency to overfit the data (e.g., if your sample for the hair model contains three bald people who all happen to be born on March 1, an overzealous decision tree might conclude that being born on March 1 is an excellent predictor of being bald). One option to counteract this is the so-called Bonferroni correction (it basically "dials up" the number of people with the same attribute it wants to see before it believes that this is *not* random). The Bonferroni correction can be very conservative, however, and you might decide to instead try the Holm–Bonferroni method or the Šidák correction.[1] For many types of models, the possible variations to such hyperparameters are almost limitless, and sadly there is no one way to set these hyperparameters that can be considered universally correct or better than all other settings. That means that the data scientist's judgment in choosing these hyperparameters is very important—and yet another potential source for bias!

- *Calibration of model outputs and decision rules* is the step where the raw model output (e.g., a probability of default) is converted into a decision rule for business applications (e.g., whether to approve or reject a loan application).

[1]This book is not meant to be a text book about multiple comparisons in statistics; I merely want to illustrate that even apparently simple statistical methods entail myriad little decisions that can affect outcomes. It's the same with electricians—it sounds straightforward to ask for a power outlet to be installed in the garage but the electrician faces myriad little decisions such as whether to use the same fuse as your freezer or a different one, what rating the fuse should have (i.e., how much amperage it can carry), etc. If the fuse blows and your ice cream in the freezer melts every time your father-in-law operates a power tool in the garage, your electrician clearly has made a bad choice with the hyperparameters!

Here a lot of additional considerations and judgments come into play (e.g., approval decisions often consider some sort of profitability criterion, and that requires the allocation of cost) which entails some very philosophical and, in the end, arbitrary judgment calls. And by now you know that where there is judgment, there is bias…

- *Model documentation* is the step where the data scientist writes down what he or she has done so that others can form an understanding and independent opinion of the model. The conceptual validation and proper use of the model by others are critical to prevent biases—and hinge on an appropriate model documentation. If there are gaps or misrepresentations in the model documentation, the reader's biases will kick in (e.g., if the model documentation boasts a high predictive power, this will trigger an anchoring effect and confirmation bias that can prevent a reader from asking clarifying questions to uncover some of the hidden problems of the model).

As a result, model assembly derives an algorithm that now can be implemented in decision processes such as approving a loan, alerting airport security of a potential terrorist, or suggesting you to buy my latest book.

Step 4: Model Validation

Model validation may be informal or, as in the case of regulated financial institutions, a formal governance process executed by a dedicated unit of the organization. It is inspired by technical inspections that have proven their value in many other realms of life—for example, in many countries cars require a regular technical inspection to validate that they are still safe for use on public roads. As you will hear in Chapter 7, one root cause for algorithmic biases are the data scientist's biases, and the independent challenger function created through model validation can be an effective counterbalance to such biases.

Step 5: Model Implementation

In most cases, a model is developed on a different computer system than the system where actual business transactions are processed. In order to use the model in "real life" to make business decisions (this often is called "in production"), additional work steps are required, which are called *implementation*. For example, once a data scientist has developed a new scorecard for approving credit card applications, the bank needs to upload the

Excerpt from Understand, Manage, and Prevent Algorithmic Bias by Tobias Baer (Apress, 2019)

algorithm in its credit decisioning (IT) system and create a process to collect the data required as inputs in the scorecard. As you will see in subsequent chapters, the way this data is created or collected from other sources—and how the system deals with missing or nonsensical values in live operations—also can introduce biased decisions; therefore, model implementation needs to be considered part of the scope of fighting algorithmic bias.

Summary

In this chapter, you reviewed the entire process of developing models; on a high level, you explored the five major steps involved in building a model:

- *Model design* ensures that the model supports its strategic objective by defining the outcome to be predicted, on what kind of population it is developed, which predictive data is used, and what modeling methodology is applied.
- *Data engineering* prepares appropriate data for the model development by defining a suitable sample; collecting raw data; splitting the sample in three parts for development, testing, and validation; ensuring high data quality by identifying and cleaning issues with the data; and aggregating granular data.
- *Model assembly* produces the actual algorithm. This step involves seven substeps, namely the exclusion of inappropriate records based on logical criteria, the development of new features, the elimination of features that are useless or redundant, an initial estimate of the model coefficients, their iterative tuning, the calibration of model outputs and decision-rules around them, and the documentation of the model.
- *Model validation* is a governance process to independently ascertain the model's fitness for use.
- *Model implementation* deploys the model in actual business operations; this involves in particular feeding data inputs into the model and linking model outputs to business decisions.

The discussion of these five work steps has introduced you to the work of the data scientist and the process required to develop an algorithm with statistical techniques. I have not yet differentiated different modeling techniques, apart from alluding to different levels of complexity. A class of modeling techniques you certainly will have heard a lot about recently is called machine learning. In the next chapter, I will unravel facts and myths about machine learning.