3

Regression, Observations and Interventions

In this chapter, we’re going to build a link between associations, interventions and regression models We’ll look into the logic of statistical control – a tool used by scientists in hope of making their models more robust. Finally, we’ll look into the connection between regression and structural models.

By the end of this chapter, you should have a solid understanding of statistical control and how it can help in estimating causal effects from observational data. This knowledge will allow us to build more complex non-linear models introduced in Part 2: Causal Inference.

In this chapter, we cover the following topics:

* Starting simple - observational data and linear regression
* Should we always control for all available covariates?
* Regression and structural models

Starting simple - observational data and linear regression

In previous chapters, we discussed the concept of association. In this section, we’ll quantify associations between variables using a regression model. We’ll see the geometrical interpretation of this model and demonstrate that regression can be performed in arbitrary direction. For the sake of simplicity, we’ll focus our attention on linear cases. Let’s start!

Linear regression

Linear regression is a basic data fitting algorithm that can be used to predict the expected value of a dependent (target) variable given values of some predictor(s) . Formally:

Note that can be multidimensional. In such case, is usually represented as a matrix X with shape , where is the number of observations and is the dimensionality of (the number of predictor variables). We call the regression model with multidimensional a multiple regression.

More on conditioning

In the previous chapter, we talked about conditional probabilities. In this chapter we expand the concept of conditioning to expected values. We read as the expected value of given . This notation is (again) slightly imprecise. A better notation would be – the expected value of given that takes value . You might find the fact that we’re talking about the expected value of interesting. Take a look at Figure 3.2. Notice that for each value of , we predict just one value of (that lies precisely on the red line), but there are datapoints (blue dots) above and below the line as well. Technically, these points could be modeled as a distribution. In such a case, the red line would represent the expected value of the distribution of at each value of . Modeling a distribution rather than just a point estimate is a hallmark of Bayesian approach to modeling that can be leveraged to estimate aleatoric uncertainty in linear and non-linear regression models. If this topic is interesting to you you can check my series of blog posts on probabilistic deep learning (Molak, 2021).

An important feature of linear regression is that it allows us to easily quantify the strength of the relationship between predictors and the target variable by computing regression coefficients. Intuitively, regression coefficients can be thought of as the amount of change in the output variable relative to a unit change in the input variable.

Coefficients and multiple regression

In multiple regression with predictors , each predictor has its respective coefficient . Each coefficient represents relative contribution of to the change in the target variable , holding everything else constant.

Let’s take a model with just one predictor . Such model can be described by the following formula:

In the preceding formula,  is a predicted value for observation , is a learned intercept term, is the observed value of , is the regression coefficient for and is the noise term. We call and model parameters.

Let’s build a simple example:

1. First, we’ll define our data generating process. We’ll use the preceding linear regression formula and assign arbitrary values to the parameters and . We’ll choose 1.12 for and 0.93 for (you can use other values if you want). We will also pick to be normally distributed with zero mean and the standard deviation of 1. With these values, our formula becomes:

Non-linear associations

Non-linear associations are also quantifiable. Even linear regression can be used to model some non-linear relationships. This is possible because linear regression has to be linear in parameters, not necessarily in the data. More complex relationships can be quantified using entropy-based metrics like mutual information (Murphy, 2022; pp. 213-218).

1. Next, we’ll translate our data generating formula into code in order to generate some data. Code for this chapter is in the Chapter\_03.ipynb notebook (https://github.com/PacktPublishing/Causal-Inference-and-Discovery-in-Python/blob/main/Chapter\_03.ipynb).

Let’s put it in work!

1. We’ll start with importing libraries that we’re going to use in this chapter. We’re going to use statsmodels for computing our linear regression model:

import numpy as np

import statsmodels.api as sm

import matplotlib.pyplot as plt

plt.style.use('fivethirtyeight')

statsmodels

statsmodels is a popular statistical library in Python that offers support for R-like syntax and R-like model summaries (in case you haven’t heard of R – it is a statistical programming language). statsmodels is a great choice if you want to work with traditional statistical models. Convenient summaries contain computed -values and other useful statistics. If you come from sklearn background you might find statsmodels API a bit confusing. There are several key differences between both libraries. One of them is the .fit() method that in statsmodels returns an instance of a wrapper object that can be further used to generate predictions. For more details on statsmodels, refer to the documentation: https://www.statsmodels.org/stable/index.html

1. Next, we’ll set random seed for reproducibility and define the number of samples that we’re going to generate:

np.random.seed(45)

N\_SAMPLES = 5000

1. Next, we’ll define our model parameters and :

alpha = 1.12

beta = 0.93

epsilon = np.random.randn(N\_SAMPLES)

1. Finally, we’ll use our model formula to generate the data:

X = np.random.randn(N\_SAMPLES)

y = alpha + beta \* X + 0.5 \* epsilon

1. There’s one more step that we need to take before fitting the model. statsmodels requires us to add a constant feature to the data. This is needed to perform the intercept computations. Many libraries performs this step implicitly, nonetheless statsmodels wants us to do it explicitly. To make our lives easier the authors provided us with a convenience method, .add\_constant(). Let’s apply it!

X = sm.add\_constant(X)

Now our X got an extra column of ones at column index 0. Let’s print the first five rows of X to see it:

print(X[:5, :])

The result is as follows:

[[ 1. 0.11530002]

[ 1. -0.43617719]

[ 1. -0.54138887]

[ 1. -1.64773122]

[ 1. -0.32616934]]

1. Now, we’re ready to fit the regression model using statsmodels and print the summary:

model = sm.OLS(y, X)

fitted\_model = model.fit()

print(fitted\_model.summary())

Output of the model summary is presented in Figure 3.1. We marked the estimated coefficients with a red ellipse:

Table

Description automatically generated

Figure 3.1 – A summary of the results of a simple linear regression model.

The coefficient marked const is the estimate of , while the coefficient marked x1 is the estimate of . They are slightly different from their true counterparts (). This is because we made our model noisy by adding the term. You can see that both coefficients are associated with -values below 0.001 (check the column named P>|t|), which indicates that they are statistically significant at the customary level.

-values and statistical significance

Broadly speaking, -value is a statistical device meant to help distinguish between the signal and the noise in statistical comparisons or summaries. More precisely, -value is a probability of observing the data at least as extreme as we observed, given that the null hypothesis is true (note that this statement is actually a counterfactual). -values are broadly used in statistics and science to determine statistical significance of a given result. The last few year brought severe critique of -values and statistical significance, highlighting that they are widely misused and misinterpreted which can lead to detrimental real-world consequences. For an overview, check Wasserstein and Lazar (2016).

Geometric interpretation of linear regression

Linear regression can also be viewed from the geometric point of view. Let’s plot the data we generated alongside the fitted regression line:

Chart, line chart

Description automatically generated

Figure 3.2 – Generated data and fitted regression line.

Each blue dot in Figure 3.2 represents a single observation, while the red line represents the best line fit found by the linear regression algorithm. In case of multiple regression, the line becomes a hyperplane.

Reversing the order

Regression is a purely statistical rung 1 model and we can use it to quantify the association between and as well as between and . In other words regression does not say anything about the data’s causal structure. In particular, there might be no causal link between two variables at all and we can still find a relationship between them using a regression model.

In the first chapter, we discussed the example of association between ice cream sales and drownings. We showed that this association was spurious, but regression model would still quantify it as existing (assuming that the association would be strong enough and possible to express using the model of choice, in our case linear regression). This is the nature of the first rung of The Ladder of Causation and – as we mentioned earlier – it can be very useful in certain cases.

More on regression vocabulary

When we use as a predictor and as a target variable (also called the dependent variable), we say that we regress on . If we use as a predictor of , we say that we regress on . Regressing on can be also expressed in R-style notation as , regressing on as , respectively. We will use this notation across the book to describe various models. We decided to use R-style notation, because of its neatness and simplicity. In some cases, though, it might be clearer to just say that we regress on rather than using a formula. We’ll use this descriptive style where clarity demands it.

To make it more hands on, let’s see how the reversed regression model looks like. We will now regress on . The code to build the reversed model is very similar to the code to build the original model, so we won’t discuss it here. If you want to examine the code for yourself, you can find it in the Chapter\_03.ipynb notebook (https://github.com/PacktPublishing/Causal-Inference-and-Discovery-in-Python/blob/main/Chapter\_03.ipynb).

Let’s take a look at the results summary in Figure 3.3:

Table

Description automatically generated

Figure 3.3 – Results summary for the reversed model.

As we can see, the coefficients have changed. In particular, the intercept is now negative. As we’re now regressing on , the intercept became the point where the fitted line crosses the axis (rather than the axis as in the original model). You can verify that the fitted line crosses axis below zero by looking at Figure 3.4 (the reversed model) and Figure 3.2 (the original model):

Chart, scatter chart

Description automatically generated

Figure 3.4 – Visualization of the reversed model.

The regression model itself cannot help us understand which variable is a cause and which is the effect (if any). To determine this, we need some sort of external knowledge.

In case of multiple regression, things become even more complicated as each additional predictor can modify the relationship between the variables in the model.

Let’s see how this translates to the idea of statistical control.

Should we always control for all available covariates?

Multiple regression offers scientists and analysts a possibility to perform statistical control – a procedure to remove unwanted influence from certain variables in the model. In this section, we’ll discuss different perspectives on statistical control and build the intuition why statistical control can easily lead us astray.

Let’s start with an example. When studying predictors of dyslexia in children, you might want to control for parental education. Parental education might affect how much attention parents devote to their children’s reading and writing and this in turn can impact children’s skills and other characteristics, potentially leading to confounding. But how do we actually know if it does lead to confounding?

In some cases, we can refer to previous research to find the answer or at least a hint. In other cases, we can rely on our intuitions or knowledge about the world (for example, we know that child’s skills cannot cause parents age). However, in many cases we will be left without a clear answer. This inevitable uncertainty led to the development of various heuristics guiding the choice of variables that should be included as statistical controls.

Navigating the maze

One of the existing heuristics is to control for as many variables as possible. This idea is based on “the (…) assumption that adding CVs [control variables] necessarily produces more conservative tests of hypotheses” (Becker et al., 2016). Unfortunately, this is not true. Moreover, controlling for wrong variables can lead to severely distorted results, including spurious effects and reversed effect signs.

Some authors offer more fine-grained heuristics. For example, Becker and colleagues (Becker et al., 2016; https://www.semanticscholar.org/paper/Statistical-control-in-correlational-studies%3A-10-Becker-Atinc/79ef4e505101a050f55418b7c0323eaabc82a95d) shared a set of 10 recommendations on how to approach statistical control. Some of their recommendations are (original ordering in parentheses):

* If you’re not sure about a variable, don’t use it as a control (1)
* Use conceptually meaningful control variables (3)
* Conduct comparative tests of relationships between the independent variables and control variables (7)
* Run results with and without the control variables and contrast the findings (8)

Although their article is titled Statistical control in correlational studies: 10 essential recommendations for organizational researchers, which suggests that it only talks about rung 1 concepts, the authors mention the risk of introducing spuriousness by adding to the model control variables that “[are] a cause of multiple IVs [independent variables], which do not themselves have direct causal connections to the DV [dependent variable]” (Becker et al., 2016). It’s not entirely clear to me why in this passage the authors mention “a cause of multiple IVs” without causal connection to the target variable as a source of confounding. In fact, this scenario would not lead to confounding!

We’ll see a couple examples of scenarios leading to confounding in a while, but first let’s take a look at some of the author’s recommendations.

If you don’t know where you’re going you might end up somewhere else

Recommendations (1) and (3) discourage adding variables to the model. This might sound reasonable – if you’re not sure, don’t add, because you might break something by accident. It seems rational, perhaps because most of us have seen or participated in situations like this in the real life – someone does not understand how something works, they do something that seems sensible to them, but they are not aware of their own blind spots and the thing is now broken.

An important aspect of this story is that the thing is now broken. This suggests that it worked properly before. This is not necessarily a valid assumption from the causal point of view. Not including a variable in the model might also lead to confounding and spuriousness. This is because there are various patterns of independence structure possible between any three variables. Let’s consider the structural causal model (SCM) presented in Figure 3.5:

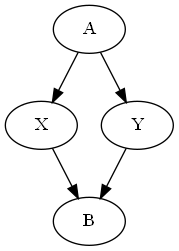


Figure 3.5 – An example SCM with various confounding patterns

More on SCM graphical representations

The representation of the SCM in Figure 3.5 differs slightly from the representations we used in the previous chapter. First, it does not contain noise variables. You can think about this representation as a simplified version with implicit noise variables – clearer and more focused. Second, nodes are represented as ellipses rather than circles. This is because we used the default mode of graphviz library to generate them. The difference in the shapes does not have any particular meaning in our case. That said, it’s a great opportunity to introduce graphviz – a software package and a Python library for graph visualization. It’s a useful tool when you work with causal models and smaller graphs or networks. To learn the basic graphviz syntax, you can refer to the notebooks accompanying this book. For more comprehensive introduction, check https://graphviz.readthedocs.io/en/stable/

From the model structure, we can clearly see that and are causally independent. There’s no arrow between them, nor there’s a directed path that would connect them indirectly (pun unintended). Let’s fit four models to see when confounding appears:

1. First, we’ll start simple with a model that regresses on .
2. Then, we’ll add to this model.
3. Next, we’ll fit a model without , but with .
4. Finally, we’ll build a model with all four variables.

What is your best guess – out of the four models, which ones will correctly capture causal independence between and ? I encourage you to write your hypotheses down on a piece of paper:Ready? Let’s find out!

Code for this experiment is in the Chapter\_03.ipynb notebook (https://github.com/PacktPublishing/Causal-Inference-and-Discovery-in-Python/blob/main/Chapter\_03.ipynb).

1. First, let’s define the SCM:

a = np.random.randn(N\_SAMPLES)

x = 2 \* a + 0.5 \* np.random.randn(N\_SAMPLES)

y = 2 \* a + 0.5 \* np.random.randn(N\_SAMPLES)

b = 1.5 \* x + 0.75 \* y

Note that all the numbers that we use to scale the variables (true coefficients) are arbitrarily chosen.

1. Next, let’s define four model variants and fit the models iteratively:

# Define four model variants

variants = [

[x],

[x, a],

[x, b],

[x, a, b]

]

# Fit models iteratively and store the results

for variant in variants:

X = sm.add\_constant(np.stack(variant).T)

model = sm.OLS(y, X)

fitted\_model = model.fit()

1. Finally, let’s examine the results in Table 3.1:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model |  | |  | |  | |
| -value | Coefficient | -value | Coefficient | -value | Coefficient |
|  | *< .001* | *0.947* | *-* | *-* | *-* | *-* |
|  | *.6565* | *0.014* | *<.001* | *1.967* | *-* | *-* |
|  | *< .001* | *-2.000* | *-* | *-* | *< .001* | *1.333* |
|  | *< .001* | *-2.000* | *0.000* | *< .001* | *< .001* | *1.333* |

Table 3.1 – Summary of the results of four regression models.

What we see in Table 3.1 is that the only model that recognized causal independence of and correctly (large -value for , suggesting the lack of significance) is the second model (). This clearly shows us that all other statistical control schemes led to invalid results, including the model without controlling for any additional variable.

Why does controlling for work while all other schemes do not? There are three elements to the answer:

* First, is a clear confounder between and and we need to control for it in order to remove confounding. This situation is structurally identical to the one in the ice cream example in the first chapter.
* Second, , , and form a pattern that we call a collider or immorality. This pattern has a very interesting behavior – it enables the flow of information between the parent variables ( and in our case) when we control for the child variable (in our example). This is exactly the opposite of what happened when we controlled for !
* Third, not controlling for any variable leads to the same result as controlling for and . This is precisely because the effects of controlling for and are exactly the opposite.

We’ll devote the whole Chapter 5, Forks, Chains, and Immoralities to a detailed discussion on the collider and two other graphical patterns (you already know their names, don’t you?).

Get involved!

Now, let’s get back to the recommendations given by Becker and colleagues. Recommendations (7) and (8) are interesting. Running comparative tests between variables can be immensely helpful in discovering causal relationships between them. Although Becker proposes to run these tests only between independent and control variables, we do not have to (and should not) restrict ourselves to this. In fact, comparative independence tests are the essence of some of the causal discovery methods. We’ll see how they work in Part 3: Causal Discovery.

To control or not to control?

The fact that smart people all over the world create heuristics to decide if a given variable should be included in the model or not only highlights how difficult it is to understand causal relationships between variables in noisy complex real-world scenarios. If we have full knowledge about the causal graph, the task of deciding which variables we should control for becomes relatively easy (and after reading the next couple of chapters you’ll probably find it trivial). If the true causal structure is unknown, the decision is fundamentally difficult.

There’s no one-size-fits-all solution to the control-or-not-to-control question. That said, understanding the realm of causality should help you make much better decisions in this regard. In a sense, causality does not give you a new angle on controlling, it gives you new eyes that allow you to see what’s invisible from the rung 1 of The Ladder of Causation.

Regression and structural models

To finish this chapter, let’s take a look at the connection between regression and structural causal models (SCMs). You might already have an intuitive understanding that they are somehow related. In this section, we’ll discuss the nature of this relationship.

Structural causal models

In the previous chapter, we’ve learned that SCMs are a useful tool for encoding causal models. They consist of a set of variables (exogenous and endogenous) and a set of functions defining the relationships between these variables. We’ve seen that SCMs can be represented as graphs, with nodes representing variables and directed edges representing functions. Finally, we’ve learned that SCMs can produce interventional and counterfactual distributions.

Structural causal models and structural equations

In causal literature, the names structural equation model (SEM) and structural causal model (SCM) are sometimes used interchangeably (e.g. Peters et al., 2017). Others refer to SEMs as a family of specific multivariate modeling technique (e.g. Bollen & Noble, 2011). SEM as a modeling technique is a vast and rich topic. For a good introduction, check the book by Kline (2015), for a detailed account on SEM and causality check Pearl (2012).

Linear regression vs SCMs

As we’ve seen previously, linear regression is a model that allows us to quantify the (relative) strength of a (linear in parameters) relationship between two or more variables. There is no notion of causal directionality in linear regression and in this sense we don’t know in which direction we should perform the regression. This condition is known as observational equivalence (Peters et al., 2017).

Finding the link

Despite all the limitation of the model, we’ve used linear regression to estimate coefficients that we interpreted as causal estimates of the strength of a relationship between variables. When fitting the four models to capture the SCM from Figure 3.5, we’ve seen that in the correct model () the estimate of the coefficient for was equal to 1.967. That’s very close to the true coefficient used in the SCM that was equal to 2. This result shows the direction of our conclusion. Linear regression can be used to estimate causal effects, given that we know the underlying causal structure (that allows us to choose which variables we should control for) and given that the underlying system is linear in parameters. In this sense linear models can be a useful microscope for causal analysis (Pearl, 2013).

To cement our intuitions regarding the link between linear regression and SCMs, let’s build one more SCM that will be linear in parameters, but non-linear in variables and estimate its coefficients with linear regression.

1. As usual, let’s first start with defining the causal structure. Figure 3.7 presents the graphical representation of our model.

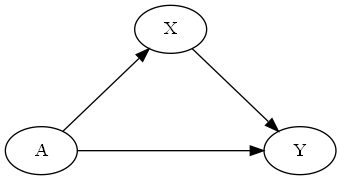


Figure 3.7. Graphical representation of an example SCM.

1. Next, let’s define the functional assignments (we’ll use the same settings for sample size and random seed as previously).

a = np.random.randn(N\_SAMPLES)

x = 2 \* a + .7 \* np.random.randn(N\_SAMPLES)

y = 2 \* a + 3 \* x + .75 \* x\*\*2

1. Let’s add constant, initialize and fit the model.

X = sm.add\_constant(np.stack([x, x\*\*2, a]).T)

model = sm.OLS(y, X)

fitted\_model = model.fit()

print(fitted\_model.summary())

Note that our functional assignment contained not only , but also . We want to make sure, that we add the square term to the model as well. This is the simplest way to introduce non-linearity into a linear regression model. Also, note that the model is still linear in parameters (we only have addition and multiplication).

Another important thing to notice is that we included in the model. The reason for this is that is a confounder in our SCM and – as we learned before – we need to control for a confounder in order to get unbiased estimates.

Great, let’s see the results!

Graphical user interface, text

Description automatically generated

Figure 3.8. The results of the model with a non-linear term.

Figure 3.8 presents the results of our regression analysis. Note that we did not pass variable names to the model, so statsmodels used generic names for them. We’ll translate them for clarity. The coefficient for was marked x1, coefficient for – x2 and the coefficient for – x3. If we compare the coefficient values to the numbers in our SCM, we can notice that they are precisely the same! This is because we modeled as a deterministic function of and , not adding any noise.

We can also see that the model correctly decoded the coefficient for the non-linear term (). Although the relationship between and is non-linear, they are related by a linear functional assignment.

Wrapping it up

It was a lot of material! Congrats on reaching the end of chapter 3!

In this chapter, we learned about the links between regression, observational data and causal models. We started with a review on linear regression. Next, we discussed the concept of statistical control and demonstrated how it can lead as astray. We analyzed selected recommendations regarding statistical control and reviewed them from causal perspective. Finally, we examined the links between linear regression and structural causal models.

Solid understanding of the links between observational data, regression and statistical control will help us move freely in the world of much more complex models that we’ll start introducing in Part 2: Causal Inference.

We’re now ready to introduce yet another perspective on causality – graphical models, which we will cover in the next chapter.

References

Becker, T. E., Atinc, G., Breaugh, J. A., Carlson, K. D., Edwards, J. R., & Spector, P. E. (2016). Statistical control in correlational studies: 10 essential recommendations for organizational researchers. Journal of Organizational Behavior, 37(2), 157–167.

Bollen, K. A., & Noble, M. D. (2011). Structural equation models and the quantification of behavior. PNAS Proceedings of the National Academy of Sciences of the United States of America, 108(Suppl 3), 15639–15646.

Kline, R. B. (2015). *Principles and practice of Structural Equation Modeling*. Guilford Press.

Molak, A. M. (2021, November 12). Modeling uncertainty in neural networks with TensorFlow Probability. Part 2: Aleatoric uncertainty. *Towards Data Science (Medium)*. <https://towardsdatascience.com/modeling-uncertainty-in-neural-networks-with-tensorflow-probability-a706c2274d12>

Murphy, K. P. (2022). *Probabilistic Machine Learning: An introduction*. MIT Press.

Pearl, J. (2012). The causal foundations of structural equation modeling. In R. H. Hoyle (Ed.), Handbook of structural equation modeling (pp. 68–91). The Guilford Press.

Pearl, J. (2013). Linear Models: A Useful “Microscope” for Causal Analysis. *Journal of Causal Inference*, *1*(1), 155-170.

Peters, J., Janzing, & D. Schölkopf, B. (2017). Elements of Causal Inference: Foundations and Learning Algorithms. MIT Press.

Wasserstein, R. L., & Lazar, N. A. (2016) The ASA Statement on p-Values: Context, Process, and Purpose. *The American Statistician*, *70*(2), 129-133