Creating Synthetic Datasets with Generative AI

Synthetic Datasets

Synthetic Data Use Cases

Hey, Pluralsight. This is long-time data professional Russ Thomas, and this course is Creating Synthetic Datasets with Generative AI. Now there's a lot of reasons you might want synthetic data, testing your data pipelines, validating stored procedures or other algorithms, training models, producing proof of concept work, code demos, new employee training, any number of reasons. Once upon a time, if we needed data to do something like test our code, we'd just make a copy of what was in prod and call it a certification environment. But increased scrutiny and liability over data privacy and security concerns makes that a less appealing option than it once was and probably with good reason. None of us like the idea of our personal data being used around the world for any number of analytics, profiles, tests, demos, or other what have yous. That said, none of us are really interested in typing out thousands of records by hand to meet a specific dataset need either. In our fake company Carved Rock Fitness, one of our most popular devices is our spin cycles or stationary bicycles. We might, for example, want to build a system that tracks and analyzes telemetry data from that spin equipment. How many miles, kilowatts, or calories has a client burned through or generated? How much simulated elevation gain or average simulated speed did a workout entail? What was the heart rate and oxygen levels during the class? With all that type of data, we could do some really interesting analytics for our clients. It would be nice to have some data to start with for testing, development, initial proof of concept, building our models, so let's first build a sample table and then list out some expectations for our synthetic data. Off the top of my head, I think a sample dataset would look something like this. We'd have a customer ID, a start time and end time to the workout, what the average heart rate was, max heart rate, average MPH, max MPH, average oxygen level in our blood, elevation gain of the workout on the bike, kilocalories burnt, the weight and the age of the customer. Now this is just off the top of my head, but it's probably a good place to start. So generating some synthetic data to begin with, I'm going to use Claude. Claude from Anthropic is one of my favorite LLMs. Now, you could do this exact same thing with ChatGPT or any other number of LLMs, and honestly, they're all going to give you pretty similar results.

Generate a Sample Dataset

All right, Claude, here's my prompt. Please create a sample table with 10 records that are similar to this example. And in my example, I have a JSON block with those data elements we talked about. The output that Claude is generating is pretty promising. I mean, I'm looking at these numbers and even the ages, the weights, the ranges are fairly believable. I'm not seeing like an age of 200 years old or a weight of 3000 pounds or anything like that. One problem I notice is that it just kind of stops. So obviously, I have a token limit here that I forgot to modify. So I'm going to switch this token limit out to its maximum available value, and I'm also going to turn up the temperature all the way to 1 so that we get maybe a little bit more variety in what we are producing. Run that again and okay, this time I do get all 10 records. So 10 records is a fun little start, but I'm interested in more like 10,000 records and that's just to begin with. As we start modeling for some machine learning, we're probably going to want maybe records in the millions. So let me change my prompt here and I'm going to do a couple of things. I'm going to tell Claude that I want 1,000 records that are similar to the example given. And this time, I'm going to give it some constraints. StartTime should be between 5:00 in the morning and 11:00 at night. EndTime is usually between 10 and 90 minutes later, most sessions around 30 minutes. AvgHR is usually between 135 and 172. MaxHR should never be more than 180, we don't want anyone having a heart attack. AvgMPH is around 10 to 20. Max should never be more than 50. I mean, the stationary bikes do go downhill sometimes, so I guess 50 is possible, but unlikely. The AvgOxygen, we're going to put that between 94 and 99 with the occasional outlier closer to 90. But again, if it drops below that, we're going to have people passing out and that is not valid data. ElevationGain, that is going to be between 0 and 4,000 ft. And KCal is something that's going to be highly dependent on the StartTime and EndTime, as well as the AvgMPH and the ElevationGain, not to mention the weight of the individual on the stationary cycle. So let's give an example, a 30 minute session for a 200LB person with 500 feet of elevation gain is usually around 380. Higher duration, speeds, or elevation gains should relate to higher KCal numbers. And finally, ages are going to range from 12 to 76 with ages between 18 and 28 having the highest MPH in ElevationGain number. All right, let's see what this generates. Right away, we get some results. And one thing I notice is that in my sample dataset, it's given me about four records, but says the rest are going to be represented by this ellipsis. My problem is I need actual records. I don't need an ellipsis. So have we hit a limit here? Are we not able to do that with generative AI? Well, there is always a way around things. And if we think about this, what we actually need is not for Claude to generate it, but maybe for Claude to write us some code that could generate it.

Dataset Assessment

So what we want to try now is to prompt Claude not to generate synthetic data, but to generate code that can produce synthetic data. Please write some Python code that will generate a synthetic dataset of 10,000 records that are similar to this example and follow the given constraints, dependencies, and relationships. We're going to leave everything else the same. And as a final step, we'll say the final step of that Python code should export the dataset to a CSV file. And just like that, we get some pretty good sample code. It's taken an interesting approach to developing random numbers between the boundaries that we've set, but also establishing a relationship between calories burnt and elevation gain, weight of the person, just like we asked. Now, I don't know if this is exactly on point, but it's really cool that it figured out a way to come up with something, and we can fine-tune this as we go along. So pulling this into SQL Server, let's kind of take a look at our data. One of the things I'm interested in is I asked for 10,000 records, and I got 10,000 records, but it is 10,000 unique customer IDs. Well, as I'm tracking data and as I'm doing some of my proof of concept tests, I kind of want multiple records related to an individual customer. So how would I go about generating data in that fashion? Because if I go that route, I'm going to need the age and the weight of the customer to kind of remain constant in relation to the customer ID. Thinking long term, a customer could have a birthday and their age could go up by one or their weight could slowly decrease over time. But for our purposes, we're going to want the age and the weight to stay the same for each individual customer.

Adding Complexity through Iteration

If you’ve looked into prompt engineering at all, prompt engineering is all about guiding your LLM to produce output, but there’s a lot of iteration involved. The best way to approach something like this is to iterate, just like you would through your own code development if you were writing all this code. So the first thing I would probably do is instead of trying to do this all in one blast, let’s go step by step. What is the first thing I need? Probably a sample of synthetic data of individual customers. So I might do something like this, Please generate some Python code that will produce a table of 1,000 records that are similar to the following example. When selecting synthetic names, they should reflect a large variety of cultures, backgrounds, and ages. Now, one of the amazing things about LLMs is that they have been trained on real-world data. So again, I gave an example age, but it was able to generate all kinds of ages and weights and names that are all very realistic. As we look through this sample data, there’s nothing here that jumps out at me as outlandish or unrealistic. All the ages are very typical. The weights are typical, and the names are a good variety of various cultures and backgrounds. So now what I want to do is use Claude to generate a synthetic dataset that has a foreign key relationship to my customer data, creating workout sessions that still maintain all the other relevant things and restrictions that we identified. So I’m going to use my original prompt, but I’m going to modify it a bit. Please write some Python code that will generate a synthetic dataset of 10,000 records. All the same, all of my restrictions on individual data is the same, but I’m going to remove the final line, the one that talks about age and instead put in the Age, Weight, and CustomerID for each record will be pulled randomly from a CSV file that will be provided by the end user. The CSV file has the following format, and I give it the format of the CSV that Claude itself generated for me just a few minutes ago. The Python code should then export all records to a CSV file named synthetic_data. As I run this, I get my sample code. We open a customer data file, reads in the CSV, randomly selects a record, and then pulls the age and weight from that record and then produces a workout session with the other random data that we specified, including a calculation for calories that takes into consideration the duration of the workout, the weight of the user, the average MPH, and the elevation gain. Now again, I’m not so concerned how spot on accurate this is right now because I can easily come back and modify this one line of code. The whole point of using an LLM in this way is to iterate and get a head start. This is all about reducing toil.

Prompt Engineering

As we identified, using LLMs to generate synthetic data is really just a specialized type of prompt engineering that requires the ability to communicate what type of synthetic data you want into a concept that the LLM can code around. The best practices of prompt engineering then all apply, namely be clear and concise, use examples, create a domain, use JSON and XML, break into steps, and leverage conversation chains. Just picking one of these best practices to align to an example from our course, all the different parameters that we put around each of our columns related to average heart rate, elevation gain, oxygen levels that were all written out in paragraph form. If you had a really large number of parameters for a large dataset, it might have been easier to create instead a JSON document that had each column with a description of its parameters and restrictions. Another one that deserves some attention is domain creation. What is the purpose of your dataset? If you are creating a large synthetic dataset that you want to match a real dataset just with synthetic data, you want to make sure that the data profile aligns, especially if you’re going to use it to create a model. The model wouldn’t be very good if it didn’t match the realistic dataset you are trying to model it for. For example, if I was creating a synthetic dataset to aid in modeling something for the area where I grew up, I would want a dataset that accurately represents the realistic demographic of rural southern Colorado. If the LLM doesn’t get it quite right, iterate again, provide it some more hints. Lastly, when creating datasets, don’t forget that sometimes to be realistic, maybe you actually do want a certain amount of bad data. If you are creating a dataset for testing pipelines, you may want to create data that is intentionally dirty or has missing elements. Ask the LLM to randomly insert bad records once or twice in every 100,000 rows and a few null values for every 1,000 rows. Most of us that have worked with data know that this is probably more realistic. Now we covered a lot of ground in a really short period of time. I will include all of the generated code, as well as my prompts that generated that code in the project assets. I will also include a copy of the CSVs as they were created that you can import and play with. It’s worth pointing out that while I was iterating through the samples for this course, Claude did change its mind from time to time on how it would go about creating random data, and there was quite a bit of variety in the code that it generated. So you will notice in my course assets that I will include various approaches to the same problem that were generated by Claude. All right, I really appreciate your time, and I hope you join me again in the future on another exploration.