Home › Tutorials › Python

Creating Synthetic Data with Python Faker Tutorial

Generating synthetic data using Python Faker to supplement real-world data for application testing and data privacy.

Aug 2022 · 13 min read

Abid Ali Awan

Certified data scientist, passionate about building ML apps, blogging on data science, and editing.

TOPICS

Python

Data Science

What is Synthetic Data?

Synthetic data is computer-generated data that is similar to real-world data. The primary purpose of synthetics data is to increase the privacy and integrity of systems. For example, to protect the Personally Identifiable Information (PII) or Personal Health Information (PHI) of the users, companies have to implement data protection strategies. Using synthetic data can help companies test new applications and protect user privacy.

In the case of machine learning, we use synthetic data to improve model performance. It is also valid for situations where data is scarce and unbalanced. The typical use of synthetics data in machine learning is self-driving vehicles, security, robotics, fraud protection, and healthcare.

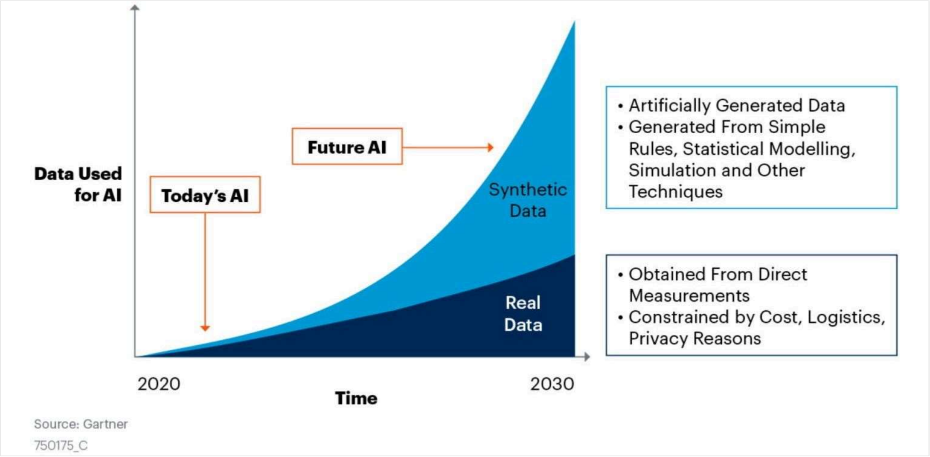


Image from Nvidia

According to data from Gartner, by 2024, 60% of data used to develop machine learning and analytical applications will be synthetically generated. But why are we seeing an upward trend of synthetics data?

It is costly to collect and clean real-world data, and in some cases, it is rare. For example, bank fraud, breast cancer, self-driving cars, and malware attack data are rare to find in the real world. Even if you get the data, it will take time and resources to clean and process it for machine learning tasks.

In the first part of the tutorial, we will learn about why we need synthetic data, its applications, and how to generate it. In the final part, we will explore the Python Faker library and use it to create synthetic data for testing and maintaining user privacy.

Why Do We Need to Generate Synthetic Data?

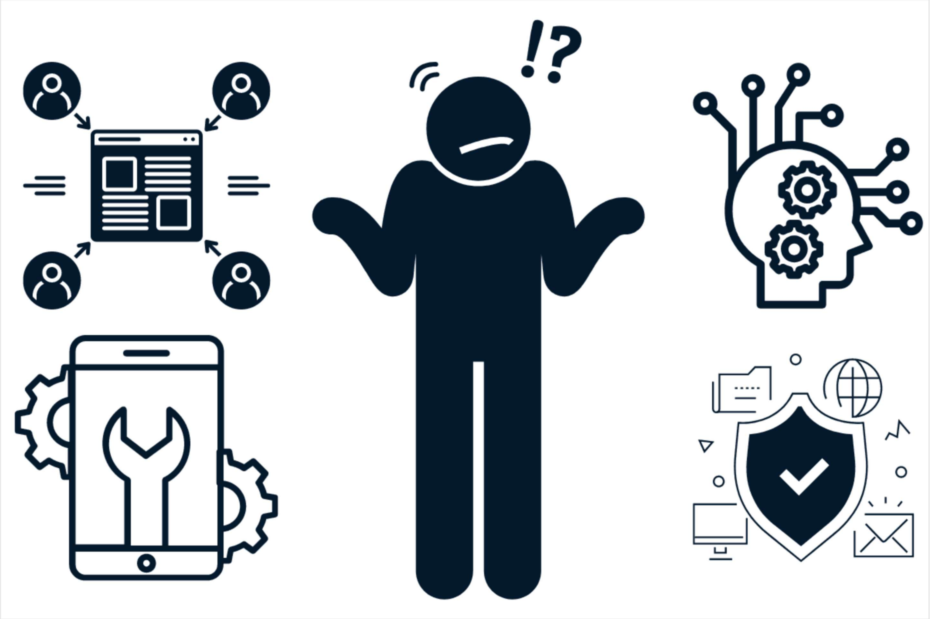


Image by Author

We need synthetic data for user privacy, application testing, improving model performance, representing rare cases, and reducing the cost of operation.

- **Privacy:** to protect users' data. You can replace names, emails, and address with synthetic data. It will help us avoid cyber and black-box attacks where models infer the details of training data.
- **Testing:** application testing on real-world data is expensive. Testing database, UI, and AI applications on synthetics data is more cost-efficient and secure.
- **Model Performance:** generated synthetics data can improve model performance. For example: in image classifiers, we use the shearing, shifting, and rotating of images to increase the size of the dataset and improve model accuracy.
- **Rare Cases:** we cannot wait for the rare event to occur and collect real-world data. Examples: credit fraud detection, car crashes, and cancer data.
- **Cost:** the data collection takes time and resources. It is costly to acquire real-world data, clean it, label it, and prepare it for testing or training models.

What are Synthetic Data Applications?

In this section, we will learn how companies use synthetics data to build cost-effective, privacy-friendly, high-performance applications.

1. **Data Sharing:** synthetic data enables enterprises to share sensitive data internally and with third parties. It also helps move private data to the cloud and retains data for analytics.
2. **Financial service:** synthetics data is generated to mimic rare events such as fraudulent transactions, anomaly detection, and economic recession. It is also used to understand customer behaviors using analytics tools.
3. **Quality Assurance:** maintaining and testing the quality of application or data systems. The synthetic data is rendered to test systems on rarer anomalies and improve performance.
4. **Healthcare:** allow us to share medical records internally and externally while maintaining patient confidentiality. You can also use it for clinical trials and detecting rare diseases. Learn to process sensitive information by taking a data privacy and anonymization course with Python or R.
5. **Automotive:** it is difficult and slow to get real-world data for robots, drones, and self-driving cars. Companies test and train their systems on synthetic simulation data and save cost on building solutions without compromising performance.
6. **Machine Learning:** we can use synthetic data to increase training dataset size, solve imbalance data problems, and test models to ensure performance and accuracy. It is also used to reduce biases in the existing image and text data. It will help us test systems and maintain user privacy. For example, DeepFake is used to test facial recognition systems.

How to Generate Synthetic Data

We can use fake data generators, statistical tools, neural networks, and generative adversarial networks to generate synthetic data.

Generating fake databases using Faker library to test databases and systems. It can generate fake user profiles with addresses and all the essential information. You can also use it to generate random text and paragraphs. It helps companies protect users' privacy in the testing phase and save money in acquiring real-world datasets.

Understanding data distribution to generate a completely new dataset using statistical tools such as Gaussian, Exponential, Chi-square, t, lognormal, and Uniform. You must have subject knowledge to generate distribution-based synthetics data.

Variational Autoencoder is an unsupervised learning method that uses an encoder and decoder to compress the original dataset and generate a representation of the original dataset. It is designed to optimize the correlation between the input and output datasets.

Generative Adversarial Network is the most popular way of generating data. You can use it to render synthetic images, sound, tabular data, and simulation data. It uses generator and discriminator deep learning model architecture to generate synthetic data by comparing random samples with actual data. Read our Demystifying Generative Adversarial Nets tutorial to create your own synthetic data using Keras.

What is Python Faker?

Python Faker is an open-source Python package used to create a fake dataset for application testing, bootstrapping the database, and maintaining user anonymity.

Image by Author

You can install Faker using:

```
pip install faker
```

Explain code

POWERED BY datalab

Faker comes with command line support, Pytest fixtures, Localization (support different regions), reproducibility, and dynamic provider (customizing it to your needs).

You can also use Faker's basic functionalities to create a quick dataset and customize it to your needs. In the table below, you can check various Faker functions and the purpose.

Faker Function	Purpose
name()	Generates fake full name
credit_card_full()	Generates credit card number with expiry and CVV
email()	Generates fake email address
url()	Generate fake URL
phone_number()	Generates fake phone number with country code
address()	Generates fake full address
license_plate()	Generates fake license plate

currency()	Generate tuple of currency code and full form
color_name()	Generate random color name
local_latlng()	Generate latitude, longitude, area, country, and states
domain_name()	Generate the fake website based fake person name
text()	Generate the fake small text
company()	Generate fake company name

To learn about more advanced functions, check out the **Faker documentation**.

It is important to review these functions as we will use them to create various examples and dataframes.

Synthetic Data Generation With Python Faker

In this section, we will use Python Faker to generate synthetics data. It consists of 5 examples of how you can use Faker for various tasks. The main goal is to develop a privacy-centric approach for testing systems. In the last part, we will generate fake data to complement the original data using Faker's localized provider.

You can find all the code for this tutorial in **this DataLab workbook**; you can easily create your own workbook copy to run all of the code in the browser, without installing anything on your computer.

First, we will initiate a fake generator using `Faker()`. By default, it is using the “en_US” locale.

```
from faker import Faker
fake = Faker()
```

Example 1

The “fake” object can generate data by using property names. For example, `fake.name()` is used for generating a random person's full name.

```
print(fake.name())
>>> Jessica Robinson
```

Similarly, we can generate a fake email address, country name, text, geolocation, and URL, as shown below.

```
print(fake.email())
print(fake.country())
print(fake.name())
print(fake.text())
print(fake.latitude(), fake.longitude())
print(fake.url())
```

Explain code POWERED BY datalab

Output

```
ybanks@example.com
Mayotte
Mr. Jose Browning DDS
Dog might bank dog total life financial. Dark view doctor time just.
Stay second treatment language theory. Space seek adult create matter imagine lay
51.7514185 -148.802970
http://fischer.info/
```

Explain code POWERED BY datalab

Example 2

You can use different locales to generate data in diverse languages and for distinct regions.

In the example below, we will generate data in Spanish and the region in Spain.

```
fake = Faker("es_ES")
print(fake.email())
print(fake.country())
print(fake.name())
print(fake.text())
print(fake.latitude(), fake.longitude())
print(fake.url())
```

Explain code POWERED BY datalab

Output

As we can see, the name of the individual has changed, and the text is in Spanish.

```
casandrahierro@example.com
Tonga
Juan Solera-Mancebo
Cumque adipisci eligendi aperiam. Quas laboriosam amet at dignissimos. Excepturi
89.180798 -2.274117
https://corbacho-galan.net/
```

Explain code POWERED BY datalab

Let’s try again with the German language and Germany as the country. To generate a full user profile, we will use the `profile()` function.

```
fake = Faker("de_DE")
fake.profile()
```

Explain code POWERED BY datalab

Output

It is quite clear how we can use Faker to generate data in various languages for various countries. Changing locale will change name, job, address, company, and other user identification data based on language and country.

```
{'job': 'Erzieher',
'company': 'Stadelmann Thanel GmbH',
'ssn': '631-64-0521',
'residence': 'Leo-Schinke-Allee 298\n26224 Altötting',
'current_location': (Decimal('51.5788595'), Decimal('29.780659')),
'blood_group': 'B+',
'website': ['https://www.schmidtke.de/',
  'https://roskoth.com/',
  'http://www.textor.de/',
  'https://www.zirme.com/'],
'username': 'vdoerr',
'name': 'Francesca Fröhlich',
'sex': 'F',
'address': 'Steinbergallee 13\n84765 Saarbrücken',
'mail': 'smuehle@gmail.com',
'birthdate': datetime.date(1998, 3, 19)}
```

Example 3

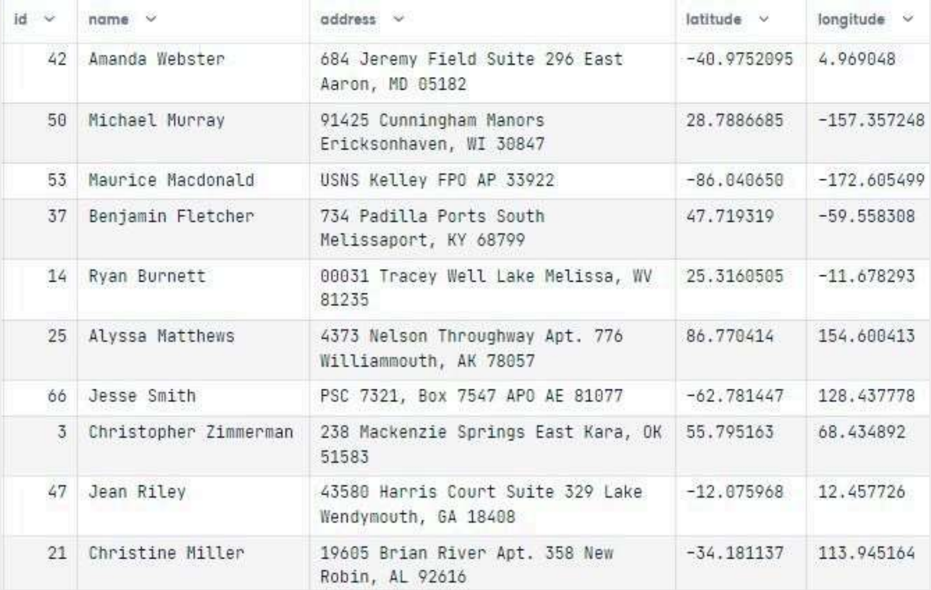
In this example, we will create a pandas dataframe using Faker.

1. Create empty pandas dataframe (data)
2. Pass it through x number of loops to create multiple rows
3. Use `randint()` to generate unique id
4. Use Faker to create a name, address, and geo-location
5. Run the `input_data()` function with x=10

```
from random import randint
import pandas as pd

fake = Faker()

def input_data(x):

    # pandas dataframe
    data = pd.DataFrame()
    for i in range(0, x):
        data.loc[i,'id']= randint(1, 100)
        data.loc[i,'name']= fake.name()
        data.loc[i,'address']= fake.address()
        data.loc[i,'latitude']= str(fake.latitude())
        data.loc[i,'longitude']= str(fake.longitude())
    return data


input_data(10)
```

Explain code

The output looks incredible. We have id, name, address, latitude, and longitude columns with unique user data.



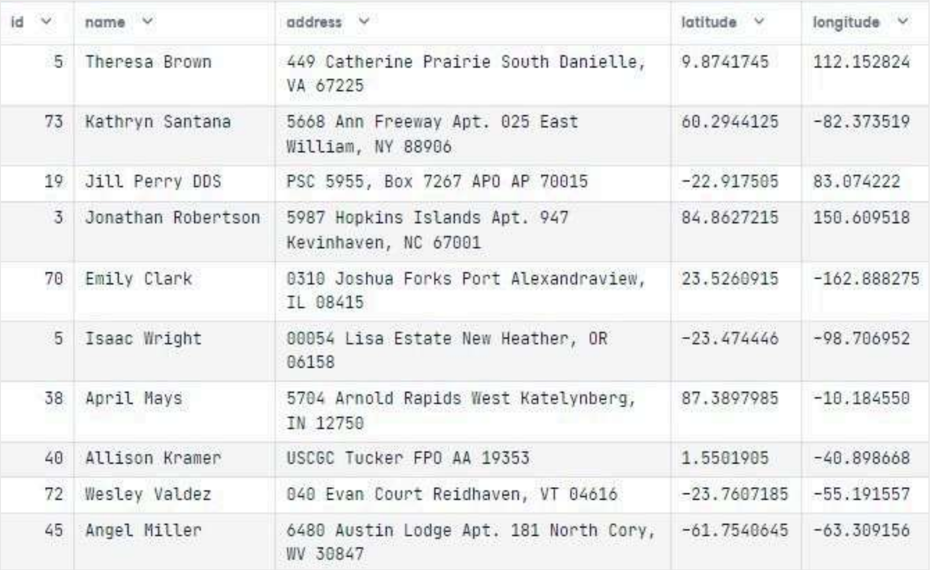
id	name	address	latitude	longitude
42	Amanda Webster	684 Jeremy Field Suite 296 East Aaron, MD 05182	-40.9752095	4.969048
50	Michael Murray	91425 Cunningham Manors Ericksonhaven, WI 30847	28.7886685	-157.357248
53	Maurice Macdonald	USNS Kelley FPO AP 33922	-86.040650	-172.605499
37	Benjamin Fletcher	734 Padilla Ports South Melissaport, KY 68799	47.719319	-59.558308
14	Ryan Burnett	00031 Tracey Well Lake Melissa, WV 81235	25.3160505	-11.678293
25	Alyssa Matthews	4373 Nelson Throughway Apt. 776 Williammouth, AK 78057	86.770414	154.600413
66	Jesse Smith	PSC 7321, Box 7547 APO AE 81077	-62.781447	128.437778
3	Christopher Zimmerman	238 Mackenzie Springs East Kara, OK 51583	55.795163	68.434892
47	Jean Riley	43580 Harris Court Suite 329 Lake Wendymouth, GA 18408	-12.075968	12.457726
21	Christine Miller	19605 Brian River Apt. 358 New Robin, AL 92616	-34.181137	113.945164

To reproduce the result, we have to set the seed. So whenever we run the code cell again, we will get similar results.

```python
Faker.seed(2)
input_data(10)
```

Explain code

POWERED BY datalab



id	name	address	latitude	longitude
5	Theresa Brown	449 Catherine Prairie South Danielle, VA 67225	9.8741745	112.152824
73	Kathryn Santana	5668 Ann Freeway Apt. 025 East William, NY 88906	60.2944125	-82.373519
19	Jill Perry DDS	PSC 5955, Box 7267 APO AP 70015	-22.917505	83.074222
3	Jonathan Robertson	5987 Hopkins Islands Apt. 947 Kevinhaven, NC 67001	84.8627215	150.609518
70	Emily Clark	0310 Joshua Forks Port Alexandraview, IL 08415	23.5260915	-162.888275
5	Isaac Wright	00054 Lisa Estate New Heather, OR 06158	-23.474446	-98.706952
38	April Mays	5704 Arnold Rapids West Katelynberg, IN 12750	87.3897985	-10.184550
40	Allison Kramer	USCGC Tucker FPO AA 19353	1.5501905	-40.898668
72	Wesley Valdez	040 Evan Court Reidhaven, VT 04616	-23.7607185	-55.191557
45	Angel Miller	6480 Austin Lodge Apt. 181 North Cory, WV 30847	-61.7540645	-63.309156

Example 4

We can also generate a sentence that contains keywords of our choice. Similar to `text(), texts(), paragraph(), word(), and words()`. You can increase the number of words in a sentence by setting the nb_words argument.

In the example below, we generate five sentences using a word list. It is not perfect, but it can help us test applications that require tons of text data.

```python
word_list = ["DataCamp", "says", "great", "loves", "tutorial", "workplace"]

for i in range(0, 5):
    print(fake.sentence(ext_word_list=word_list))
```

Explain code

POWERED BY datalab

Output

```
Loves great says.
Says workplace workplace tutorial great loves.
Loves workplace workplace loves workplace loves great DataCamp.
Loves says workplace great.
Workplace great DataCamp.
```

Explain code

POWERED BY datalab

Blending the synthetics and real data

In this part, we will use E-Commerce Data from DataCamp's dataset repository and add fake user data using **CustomerID** and **Country** columns. It will help us maintain user privacy and test the system on additional parameters.

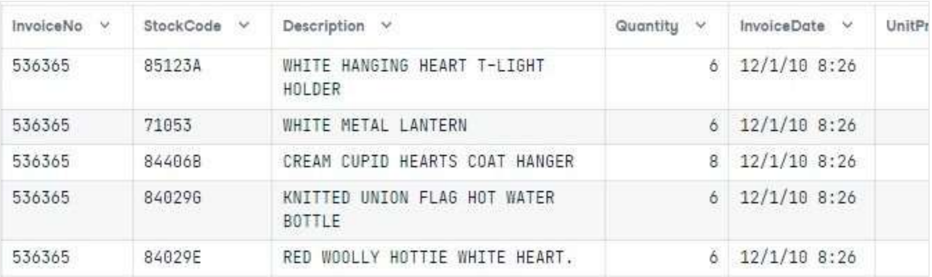
To load CSV file, we will use and pandas \`read_csv()\` function and display top five rows using \`head()\`

```
# Loading CSV file
Ecommerce = pd.read_csv("e-commerce.csv")
Ecommerce.head()
```

Explain code

POWERED BY datalab

The dataset consists of InvoiceNo, StockCode (product ID), Description, Product (item) name, Quantity (per transaction), InvoiceDate, UnitPrice (in Pounds), CustomerID, and Country columns.



InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPr
536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/10 8:26	
536365	71053	WHITE METAL LANTERN	6	12/1/10 8:26	
536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/10 8:26	
536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/10 8:26	
536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/10 8:26	

To create a localized faker object, we will display unique country names and use them to create a dictionary. As we can observe, we have seven countries, and we will be creating seven faker localized generators in the next part.

```
Ecommerce.Country.dropna().unique()

>>> array(['United Kingdom', 'France', 'Australia', 'Netherlands', 'Germany',
       'Norway', 'EIRE'], dtype=object)
```

Explain code

POWERED BY datalab

In the \`anonymous\` function, we have:

1. Extracted unique customer id and dropped missing values
2. Created dictionary of country names and locale short form
3. Extracted row index and country name for unique customer id
4. Used dictionary and counter name to initialize localized faker generator
5. Adding customer name, address, and geo-location to existing dataframe.

The function below uses the dataframe and adds four new columns with user data based on CustomerID and Country columns.

```
def anonymous(df):

    # Extracting unique CustomerID
    unique_id = df.CustomerID.dropna().unique()

    # Creating the dictionary for Faker localized providers
    local = {
        "United Kingdom": "en_GB",
        "France": "fr_FR",
        "Australia": "en_AU",
        "Netherlands": "nl_NL",
        "Germany": "de_DE",
        "Norway": "no_NO",
        "EIRE": "ga_IE",
    }

    for i in unique_id:

        # Extracting row index
        row_id = df[df["CustomerID"] == i].index

        # Extracting country name for faker locale
        CountryName = Ecommerce.loc[
            Ecommerce["CustomerID"] == i, "Country"
        ].to_numpy()[0]

        # Using locale dictionary to create faker locale generator
        code = local[CountryName]
        fake = Faker(code)

        # Generating fake data and adding it to dataframe
        CustomerName = fake.name()
        Address = fake.address()
        Latitude = str(fake.latitude())
        Longitude = str(fake.longitude())

        for x in row_id:
            df.loc[x, "CustomerName"] = CustomerName
            df.loc[x, "Address"] = Address
            df.loc[x, "Latitude"] = Latitude
            df.loc[x, "Longitude"] = Longitude

    return df
```

Explain code

POWERED BY datalab

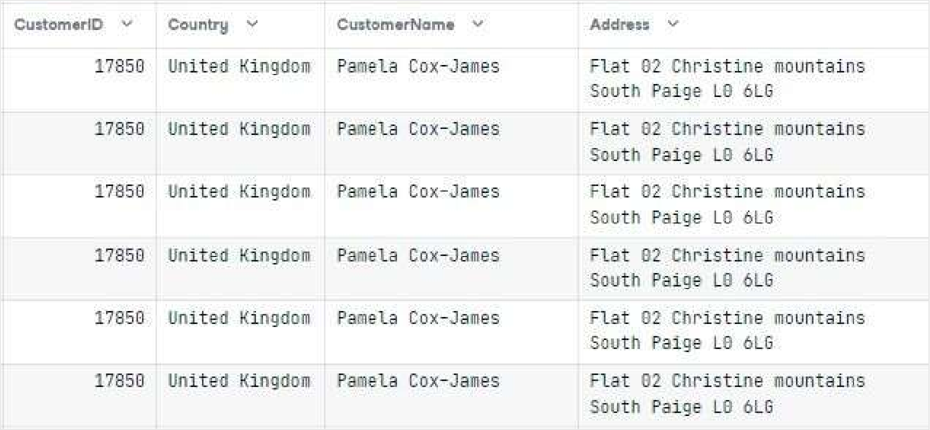
We will use seed(5) for reproducibility and run an `anonymous` function on the Ecommerce dataframe.

```
# Using seed for reproducibility
Faker.seed(5)
```

```
secure_db = anonymous(Ecommerce)
secure_db
```

Explain code POWERED BY datalab

The first few results show the data of 17850, United Kingdom, and Pamela Cox-James.



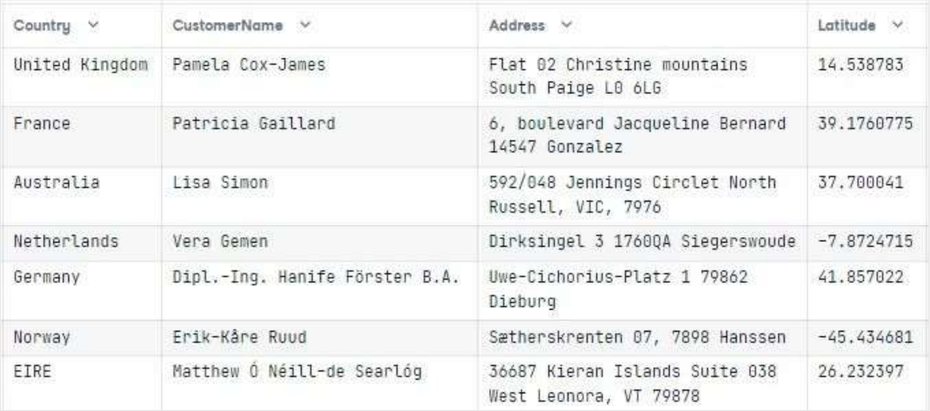
CustomerID	Country	CustomerName	Address
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG
17850	United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG

To visualize the localized results of the function, we need to view data for a unique Country column.

```
display_db = []
for i in Ecommerce.Country.dropna().unique():
    display_db.append(secure_db[secure_db["Country"] == i].to_numpy()[0])
pd.DataFrame(display_db, columns=Ecommerce.columns)
```

Explain code POWERED BY datalab

The result is promising. You can see different names per region and language. The addresses are matched with the country. With this method, you can drop personal identification data and create localized data using Faker to protect user privacy and save money.



Country	CustomerName	Address	Latitude
United Kingdom	Pamela Cox-James	Flat 02 Christine mountains South Paige L0 6LG	14.538783
France	Patricia Gaillard	6, boulevard Jacqueline Bernard 14547 Gonzalez	39.1760775
Australia	Lisa Simon	592/048 Jennings Circlet North Russell, VIC, 7976	37.700041
Netherlands	Vera Gemen	Dirksingel 3 1760QA Siegerswoude	-7.8724715
Germany	Dipl.-Ing. Hanife Förster B.A.	Uwe-Cichorius-Platz 1 79862 Dieburg	41.857022
Norway	Erik-Kåre Ruud	Sætherskrenten 07, 7898 Hanssen	-45.434681
EIRE	Matthew Ó Néill-de Searlóg	36687 Kieran Islands Suite 038 West Leonora, VT 79878	26.232397

The code source is available in this DataLab workbook.

Conclusion

One of the drawbacks of using Python Faker is that it provides poor data quality. It can work for application testing, but it lacks data accuracy. For example, names do not match email, domain name, or username.

You can customize the provider or create a new one based on your preference, but it will take you extra time to perfect the system. In that time, you can create your Python package using `random.choice()`.

In big tech, data scientists are using various tools to process sensitive data and maintain data privacy. They are also using synthetics data to improve model performance, reduce basis, test applications, and save cost in developing cutting-edge AI solutions.

If you are interested in learning more, check out the Data Scientist with Python career track to begin your journey of becoming a confident data scientist.

In this tutorial, we have learned the importance of synthetics data and its applications. We have also generated fake data from scratch using Python Faker to test data systems and maintain users' privacy.

TOPICS

Python Data Science

Courses for Python

COURSE

Introduction to Natural Language Processing in Python

4 hr 115.8K

Learn fundamental natural language processing techniques using Python and how to apply them to extract insights from real-world text data.

See Details → Start Course

See More →

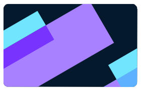
Related

BLOG

What is Synthetic Data?

TUTORIAL

Generating Realistic Random Datasets with Trumania



TUTORIAL

A Complete Guide to Data Augmentation

See More →

LEARN

Learn Python

Learn R

Learn AI

Learn SQL

Learn Power BI

Learn Tableau

Learn Data Engineering

Assessments

Career Tracks

Skill Tracks

Courses

Data Science Roadmap

DATA COURSES

Python Courses

R Courses

SQL Courses

Power BI Courses

Tableau Courses

Alteryx Courses

Azure Courses

Google Sheets Courses

AI Courses

Data Analysis Courses

Data Visualization Courses

Machine Learning Courses

Data Engineering Courses

Probability & Statistics Courses

DATALAB

Get Started

Pricing

Security

Documentation

CERTIFICATION

Certifications

Data Scientist

Data Analyst

Data Engineer

SQL Associate

Power BI Data Analyst

Tableau Certified Data Analyst

Azure Fundamentals

AI Fundamentals

RESOURCES

Resource Center

Upcoming Events

Blog

Code-Alongs

Tutorials

Open Source

RDocumentation

Course Editor

Book a Demo with DataCamp for Business

Data Portfolio

Portfolio Leaderboard

PLANS

Pricing

For Business

For Universities

Discounts, Promos & Sales

DataCamp Donates

FOR BUSINESS

Business Pricing

Teams Plan