What is random_state?

random_state = 0 or 42 or none

Kishan Modasiya · Follow

5 min read · Jun 25, 2022

422 1

Image from Unsplash

Maybe you all have used `random_state` before if you are in **Machine Learning** while splitting dataset into training and testing or in any hyperparameter. Most people use the value of `random_state` as **0** or **42**. But do you really know what is exactly it? Then let's get started and get to know what is it.

What is it?

In **Scikit-learn**, it controls the shuffling applied to the data before applying the split. We use it in `train_test_split` for splitting data into training and testing dataset. It takes one of the following values.

- **None (Default)**
  It uses the global random state instance from `numpy.random`. If we call the same function with `random_state=none` then it will produce different results in every execution.
- **An integer**
  If we use any integer value for `random_state` then it will produce the same result for an integer value. If we change the value of `random_state`, then only the result will be different.

> ***`random_state`*** *can not be negative!!!*

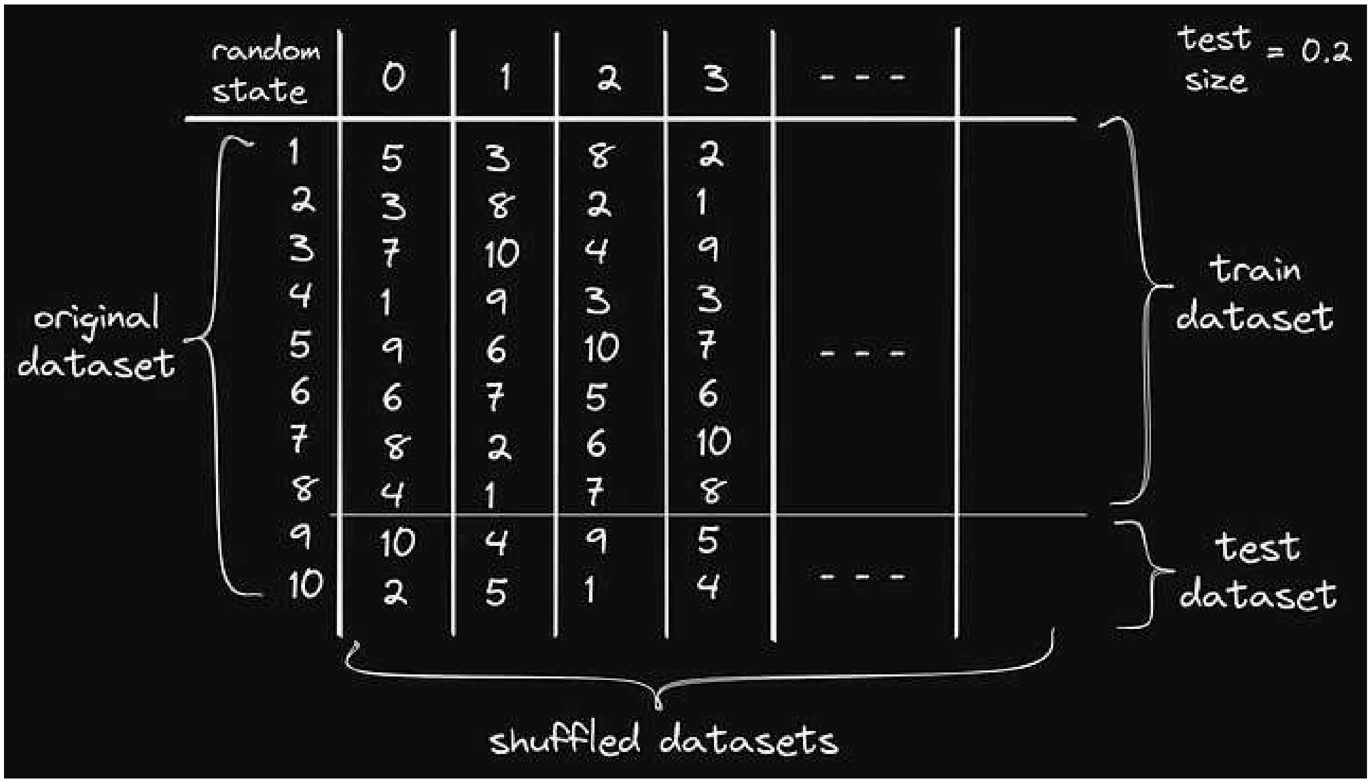
Let's see how it works

Suppose we have dataset of 10 numbers from 1 to 10, and now if we want to split it into training dataset and testing dataset and size of testing dataset is 20% of the whole dataset.

Training dataset has 8 and testing dataset has 2 data samples. So we ensure that a random process will output the same result every time, which makes code reproducible. Because if we do not shuffle dataset then it will produce different datasets every time and it is not good to train model with different data every time.

For all random datasets, each assign with a `random_state` value. It means one `random_state` value has a fixed dataset. It means every time we run code with `random_state` value 1, it will produce the same splitting datasets.

See the below image for better intuition.



random state	0	1	2	3	- - -
1	5	3	8	2	
2	3	8	2	1	
3	7	10	4	9	
4	1	9	3	3	
5	9	6	10	7	- - -
6	6	7	5	6	
7	8	2	6	10	
8	4	1	7	8	
9	10	4	9	5	
10	2	5	1	4	- - -

Image of how random_state works

- Here I wanna clear you about one thing. I see most of the people are using `random_state = 42`, even I have used too. As per the above image, there's one fixed shuffled dataset for `random_state` value 42. It means whenever we use 42 as `random_state`, it'll return a shuffled dataset.

So, **42 is not a special number for `random_state`.**

Let's see how it can be used in splitting the dataset.

Here, we use a dataset of Wine Quality and Linear Regression model. Keep it simple because our main is `random_state`, not accuracy.

Use of random_state in splitting

In the above code, for `random_state` 0, `mean_squared_error` is 0.384471197820124. If we try different values for `random_state`, then error will be different every time.

- For `random_state = 1`, get `mean_squared_error` 0.38307198158142
- For `random_state = 69`, get `mean_squared_error` 0.47013897077423

- For `random_state = 143`, get `mean_squared_error` 0.42062134425032

But how many random states are possible? 🤔

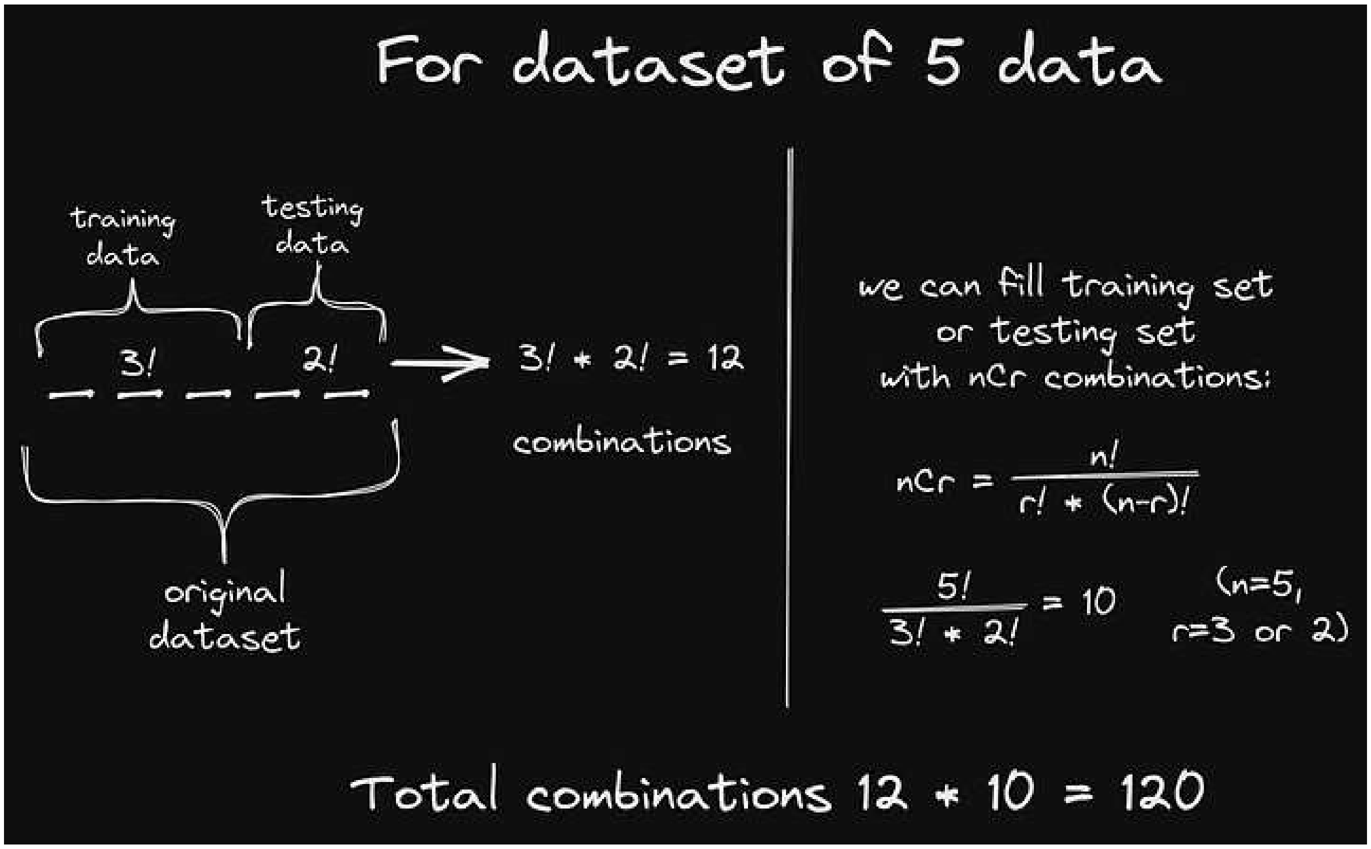
I have done an experiment on how many **unique datasets** do we get by shuffling the original dataset?

From GIPHY

I took a dataset of 5 data (Just simple 1, 2, 3, 4, 5) and split into in training and testing dataset 2000 times, with `random_state` 1 to 2000. I stored these 2000 shuffled datasets in the list. And in that list I found **120 unique dataset.**

It means we can assume that for the dataset of 5 data, there's a total 120 unique combination of dataset we got, it means we can set `random_state` in between 0 to 119. You can set `random_state` greater than 119 then it will give dataset from one of that 120 unique datasets.

As per my understanding, I can explain it in this way:



Total unique combination of random_state

Why do we need it?

From GIPHY

- Imagine we are working with house price prediction. In the dataset, as we go from up to bottom, the number of bedrooms is increasing or the area of the apartment is increasing. This is called bias data.
  If we just split data without shuffling, it will give good performance on the training dataset but poor performance on testing. For that, we need to do shuffling of data. So, if we use `random_state` then we can avoid this problem.
- When we split data, we want consistence in the result of the dataset every time. This means if we rerun the code, then the training and testing dataset will remain the same every time. Suppose I want you to get the same results that I got when you try out my code. For that `random_state` is required.
- For different `random_state` we can see the difference in performance. As per, we see above in the example, different `random_state` give different `mean_squared_error`. It means if you select randomly the value of

`random_state` and if you are lucky, then the error score will be minimized for that `random_state`.

Other uses of random state

- **K means**
  In KMeans, `random_state` determines random number generation for centroid initialization. We can use an Integer value to make the randomness deterministic. Also, it is useful when we want to produce the same clusters every time.

- **Random Forest**
  In Random Forest Classifier and Regression, `random_state` controls the randomness of the bootstrapping of the samples used when building trees and the sampling of the features to consider when looking for the best split at each node.

- **Decision Tree**
  In Decision Tree Classifier or Regression, when we want to find the best features that controls the randomness of splitting nodes, `random_state` is helpful. It will describe the structure of the tree.

- **Randomized Search CV**

- **Stratified KFold**

`random_state` is used in many other places in Machine Learning, but most of the time it uses for the randomness of data or shuffling.

So that's it for `random_state`!

Check out my other blog on Machine Learning here 👉 **Medium**.

Thanks for reading it. If you like it, then give it a clap and share it.

From GIPHY

Follow for more on **Medium**, I'll share more Machine Learning stuff here. Here's my **Twitter**, follow and connect with me there and feel free to DM.

*More content at **PlainEnglish.io**. Sign up for our **free weekly newsletter**. Follow us on **Twitter** and **LinkedIn**. Check out our **Community Discord** and join our **Talent Collective**.*

Mlearning.ai Submission Suggestions

How to become a writer on Mlearning.ai