Difference Between fit(), transform(), fit_transform() methods in Scikit-Learn (with Python Code)

Mayur Badole — Published On April 30, 2021
Beginner Libraries Programming Python

This article was published as a part of the Data Science Blogathon.

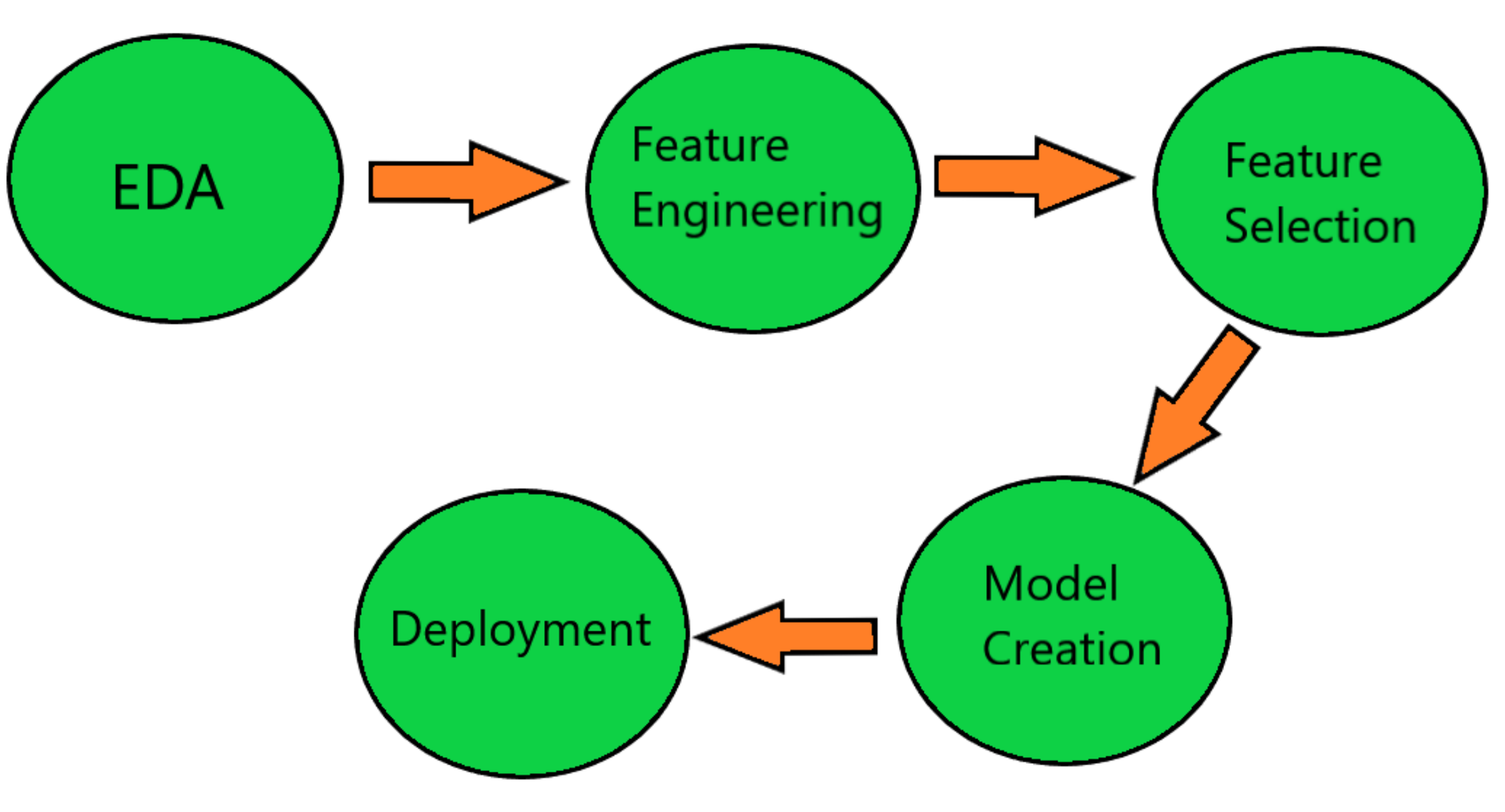
"Consumer data will be the biggest differentiator in the next two to three years. Whoever unlocks the reams of data and uses it strategically will win"

Table of contents

Introduction

Before going ahead, if we considered the life cycle of any data science project, then we know that there are certain steps that help us to develop any data science projects. We will discuss them in points:

1. **Exploratory Data Analysis (EDA)** is used to analyze the datasets and by this, we summarize their main importance.
2. **Feature Engineering** is the process of extract features from raw data with some domain knowledge.
3. **Feature Selection** where we select those features that will give a high impact on the model.
4. **Model creation** in this we create a machine learning model using suitable algorithms.
5. **Deployment** where we deploy our ML model on the web.



If we considered the first 3 steps then it will probably more towards Data Preprocessing and Model Creation is more towards Model Training. So these are the two most important steps whenever we wanted to deployment any machine learning application.

Transformer In Sklearn

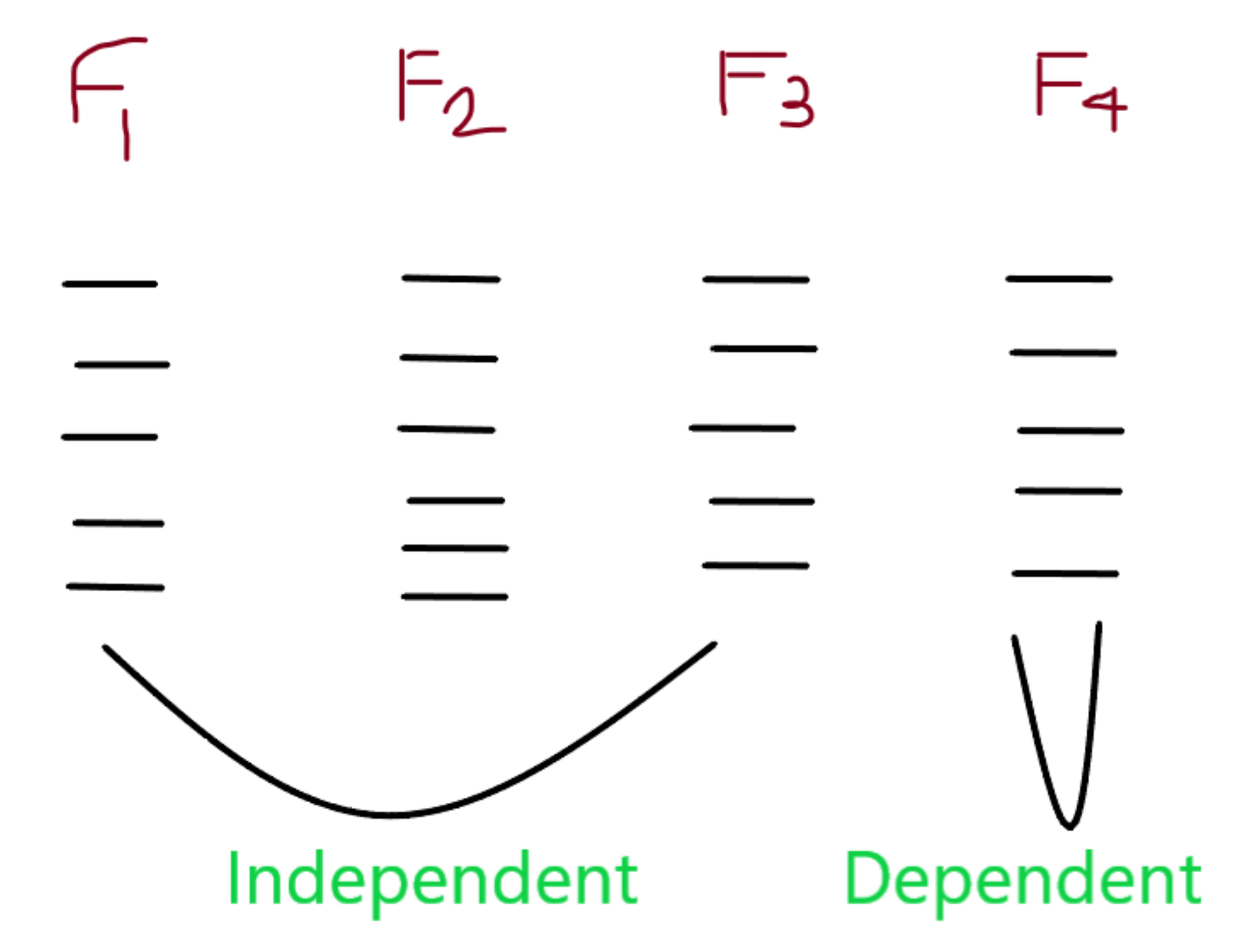
Scikit-learn has an object usually something called a **Transformer.** The use of a transformer is that it will be performing data preprocessing and feature transformation but in the case of model training, we have objects called models like linear regression, classification, etc… if we talk about the examples of Transformer-like **StandardScaler** which helps us to do feature transformation where it converts the feature with mean =**0** and standard deviation =**1**, **PCA**, **Imputer**, **MinMaxScaler,** etc… then all these particular techniques have seen that we are doing some preprocessing on the input data will change the formate of data and that data will be used for model training

Suppose we take **f1, f2, f3** and **f4** feature where f1,f2,f3 are independent features and f4 is our dependent feature and we apply a standardization process in which it takes a feature **F** and converts into **F'** by applying a formula of standardization, If you notice at this stage we take one input feature F and convert it into other input feature F' itself So, in this condition we do Three difference operation:

1. fit()

2. transform()

3. fit_transform()



Now, we will discuss how those following operations are different from each other.

Why they differ from each other

fit() :

In the **fit()** method, where we use the required formula and perform the calculation on the feature values of input data and fit this calculation to the transformer. For applying the fit() method we have to use **.fit()** in front of the transformer object.

Suppose we initialize the StandardScaler object **O** and we do **.fit()** then what will it do that, it takes the feature **F** and it will just compute the **mean (μ)** and **standard deviation (σ)** of feature **F.** That has happened in the fit method.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# split training and testing data
xtrain,xtest,ytrain,ytest= train_test_split(
                                            x,y,
                                            test_size=0.3,
                                            random_state=42
                                           )

# creating object
stand= StandardScaler()
# fit data
Fit= stand.fit(xtrain)
```

First, we have to split the dataset into training and testing subsets and after that, we apply a transformer to that data.

In the next step, we basically perform transform because it was the second operation on the transformer:

transform() :

For changing the data we probably do transform, in the transform() method, where we apply the calculations that we have calculated in fit() to every data point in feature F. We have to use **.transform()** in front of a fit object because we transform the fit calculations.

We use the example that is used above section when we create an object of the fit method then we just put it in front of the .transform and transform method uses those calculations to transform the scale of the data points, and the output will we get is always in the form of sparse matrix or array.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# split training and testing data
xtrain,xtest,ytrain,ytest= train_test_split(
                                            x,y,
                                            test_size=0.3,
                                            random_state=42
                                           )

# creating object
stand= StandardScaler()
# fit data
Fit= stand.fit(xtrain)
# transform data
x_scaled = Fit.transform(xtrain)
```

```
array([[-0.82955914, -0.84844726,  0.76004081, ...,  0.42452065,
         1.71511025, -0.28720569],
       [ 0.38933126, -0.84844726, -0.82003838, ...,  1.15770723,
         0.13623772, -0.64327699],
       [ 0.3313334 ,  1.64017602, -0.82003838, ..., -1.57425478,
         1.38751668, -0.71110009],
       ...,
       [-0.87071891, -0.84844726,  1.14837868, ...,  1.36496845,
         0.34050194,  0.93361021],
       [ 1.79250517,  0.54283955, -0.82003838, ..., -1.57425478,
         0.11696751, -0.28720569],
       [ 0.28643184, -0.84844726,  0.95656807, ..., -0.62992083,
         0.13623772, -0.28720569]])
```

As you can see that the output of the transform is in the form of an array in which data points vary from 0 to 1.

notice: It will only perform when we want to do some kind of transformation on the input data.

fit_transform():

This fit_transform() method is basically the combination of fit method and transform method, it is equivalent to **fit().transform().** This method performs fit and transform on the input data at a single time and converts the data points. If we use fit and transform separate when we need both then it will decrease the efficiency of the model so we use fit_transform() which will do both the work.

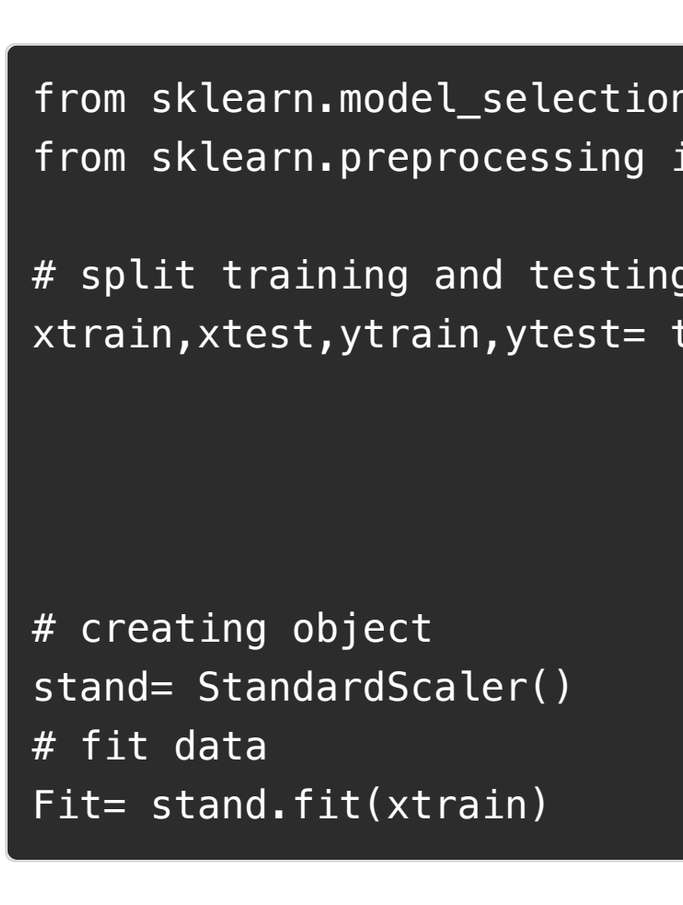
Suppose, we create the StandarScaler object, and then we perform **.fit_transform()** then it will calculate the mean(μ) and standard deviation(σ) of the feature **F** at a time it will transform the data points of the feature F.

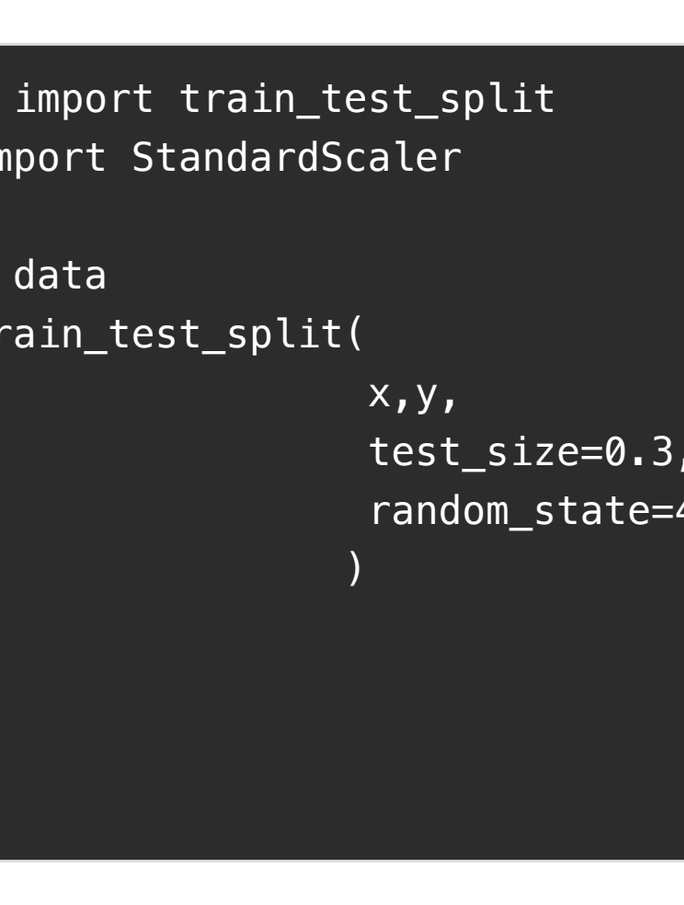
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
# split training and testing data
xtrain,xtest,ytrain,ytest= train_test_split(
                                            x,y,
                                            test_size=0.3,
                                            random_state=42
                                           )
stand= StandardScaler()
Fit_Transform = stand.fit_transform(xtrain)
Fit_Transform
```

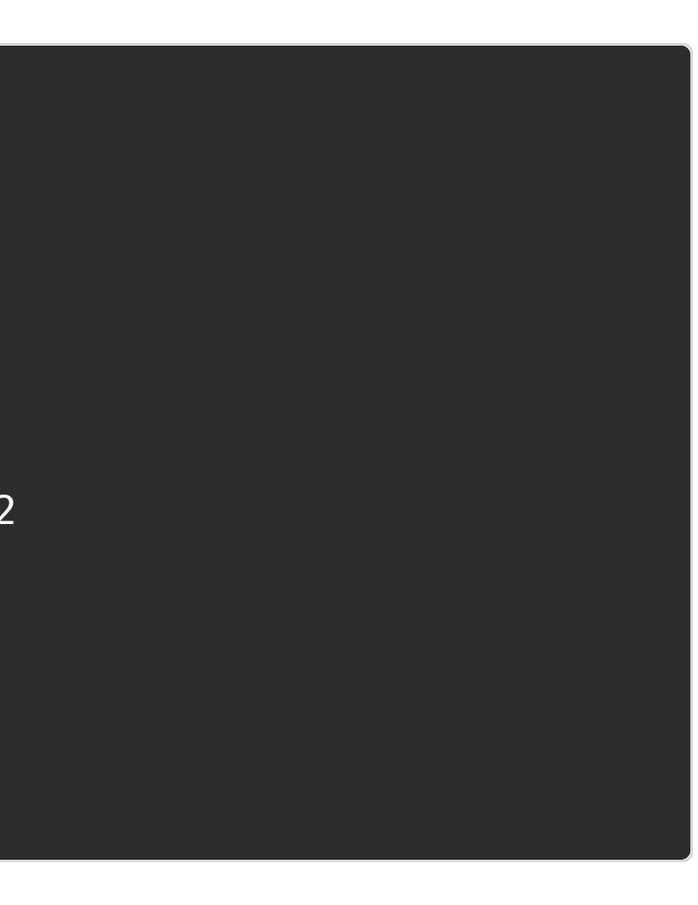
```
array([[-0.82955914, -0.84844726,  0.76004081, ...,  0.42452065,
         1.71511025, -0.28720569],
       [ 0.38933126, -0.84844726, -0.82003838, ...,  1.15770723,
         0.13623772, -0.64327699],
       [ 0.3313334 ,  1.64017602, -0.82003838, ..., -1.57425478,
         1.38751668, -0.71110009],
       ...,
       [-0.87071891, -0.84844726,  1.14837868, ...,  1.36496845,
         0.34050194,  0.93361021],
       [ 1.79250517,  0.54283955, -0.82003838, ..., -1.57425478,
         0.11696751, -0.28720569],
       [ 0.28643184, -0.84844726,  0.95656807, ..., -0.62992083,
         0.13623772, -0.28720569]])
```

This method output is the same as the output we obtain after applying the separate fit() and transform() method.

Related


Want to Build a Career in Data Science? Learn from these 5 Data Science Videos!


10 Resources to Successfully navigate a career in Data Science!


What Data Science Future Looks Like?

blogathon python scikit-learn sklearn

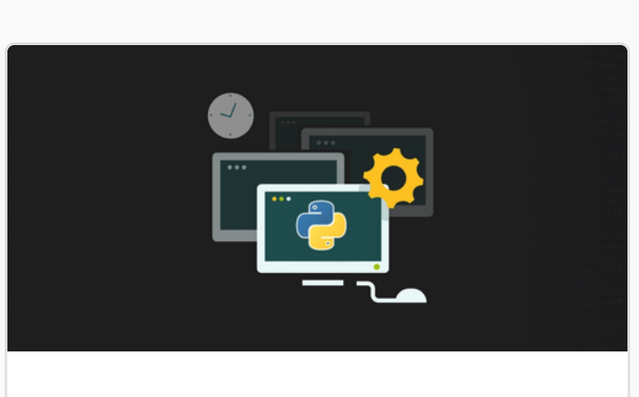
About the Author

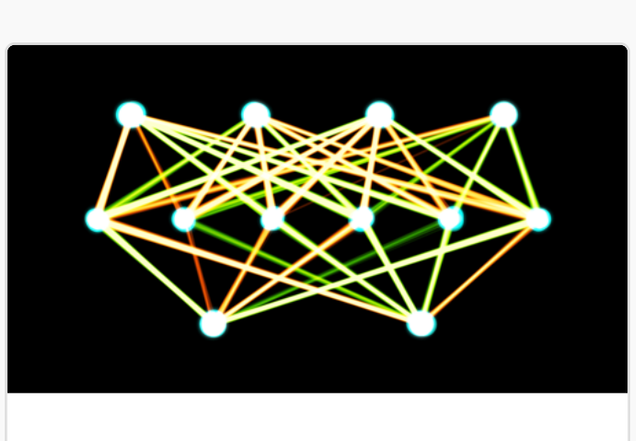
Mayur Badole

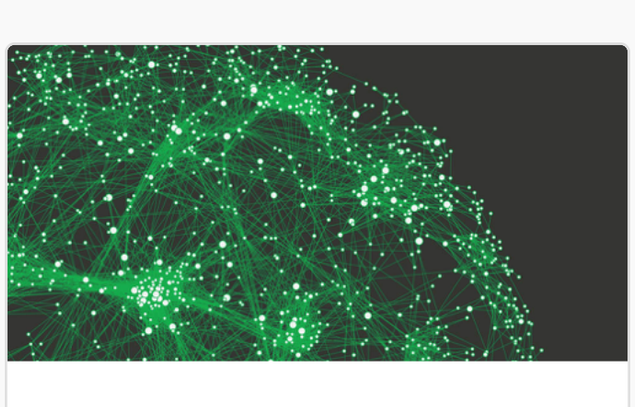
Previous Post
Make Your Tableau Visuals More Effective – Tips And Tricks

Next Post
How to Download Kaggle Datasets using Jupyter Notebook

Top Resources


Python Tutorial: Working with CSV file for Data Science
Harika Bonthu -
AUG 21, 2021


Boost Model Accuracy of Imbalanced COVID-19 Mortality Prediction Using GAN-based..
Bala Gangadhar Thilak Adiboina -
OCT 07, 2020


The Most Comprehensive Guide to K-Means Clustering You'll Ever Need
Pulkit Sharma - AUG 19, 2019


Understanding Random Forest
Sruthi E R - JUN 17, 2021