Imbalanced-Learn module in Python

Read Courses Practice

Imbalanced-Learn is a Python module that helps in balancing the datasets which are highly skewed or biased towards some classes. Thus, it helps in resampling the classes which are otherwise oversampled or undesampled. If there is a greater imbalance ratio, the output is biased to the class which has a higher number of examples. The following dependencies need to be installed to use imbalanced-learn:

- scipy(>=0.19.1)
- numpy(>=1.13.3)
- scikit-learn(>=0.23)
- joblib(>=0.11)
- keras 2 (optional)
- tensorflow (optional)

To install imbalanced-learn just type in :

```
pip install imbalanced-learn
```

The resampling of data is done in 2 parts:

Estimator: It implements a fit method which is derived from *scikit-learn*. The data and targets are both in the form of a 2D array

```
estimator = obj.fit(data, targets)
```

Resampler: The *fit_resample* method resample the data and targets into a dictionary with a key-value pair of *data_resampled* and *targets_resampled*.

```
data_resampled, targets_resampled = obj.fit_resample(data,
targets)
```

The Imbalanced Learn module has different algorithms for oversampling and undersampling:

We will use the built-in dataset called the *make_classification* dataset which return

- **x:** a matrix of n_samples*n_features and
- **y:** an array of integer labels.

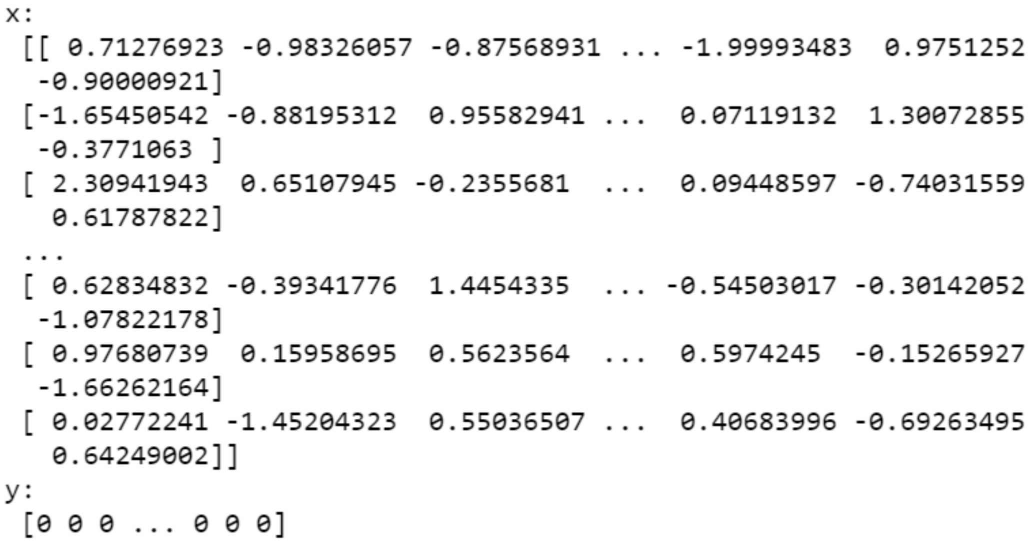
Click dataset to get the dataset used.

Python3

```python
# import required modules
from sklearn.datasets import make_classification

# define dataset
x, y = make_classification(n_samples=10000,
                           weights=[0.99],
                           flip_y=0)
print('x:\n', X)
print('y:\n', y)
```

Output:



```
x:
 [[ 0.71276923 -0.98326057 -0.87568931 ... -1.99993483  0.9751252
  -0.90000921]
 [-1.65450542 -0.88195312  0.95582941 ...  0.07119132  1.30072855
  -0.3771063 ]
 [ 2.30941943  0.65107945 -0.2355681  ...  0.09448597 -0.74031559
   0.61787822]
 ...
 [ 0.62834832 -0.39341776  1.4454335  ... -0.54503017 -0.30142052
  -1.07822178]
 [ 0.97680739  0.15958695  0.5623564  ...  0.5974245  -0.15265927
  -1.66262164]
 [ 0.02772241 -1.45204323  0.55036507 ...  0.40683996 -0.69263495
   0.64249002]]
y:
 [0 0 0 ... 0 0 0]
```

Below are some programs in which depict how to apply oversampling and undersampling to the dataset:

Oversampling

- **Random Over Sampler:** It is a naive method where classes that have low examples are generated and randomly resampled.

Syntax:

from imblearn.over_sampling import RandomOverSampler

Parameters(optional): *sampling_strategy='auto', return_indices=False, random_state=None, ratio=None*

Implementation:
oversample = RandomOverSampler(sampling_strategy='minority')
X_oversample,Y_oversample=oversample.fit_resample(X,Y)

Return Type:*a matrix with the shape of n_samples*n_features*

Example:

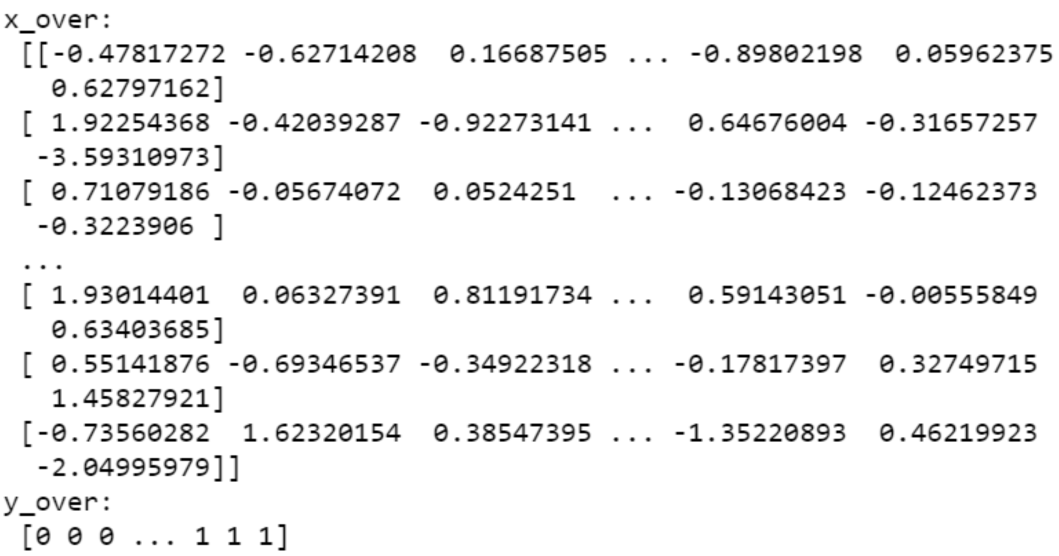
```
# import required modules
from sklearn.datasets import make_classification
from imblearn.over_sampling import RandomOverSampler

# define dataset
x, y = make_classification(n_samples=10000,
                           weights=[0.99],
                           flip_y=0)

oversample = RandomOverSampler(sampling_strategy='minority')
x_over, y_over = oversample.fit_resample(x, y)

# print the features and the labels
print('x_over:\n', x_over)
print('y_over:\n', y_over)
```

Output:



```
x_over:
 [[-0.47817272 -0.62714208  0.16687505 ... -0.89802198  0.05962375
   0.62797162]
 [ 1.92254368 -0.42039287 -0.92273141 ...  0.64676004 -0.31657257
  -3.59310973]
 [ 0.71079186 -0.05674072  0.0524251  ... -0.13068423 -0.12462373
  -0.3223906 ]
 ...
 [ 1.93014401  0.06327391  0.81191734 ...  0.59143051 -0.00555849
   0.63403685]
 [ 0.55141876 -0.69346537 -0.34922318 ... -0.17817397  0.32749715
   1.45827921]
 [-0.73560282  1.62320154  0.38547395 ... -1.35220893  0.46219923
  -2.04995979]]
y_over:
 [0 0 0 ... 1 1 1]
```

- **SMOTE, ADASYN:** Synthetic Minority Oversampling Technique (SMOTE) and the Adaptive Synthetic (ADASYN) are 2 methods used in oversampling. These also generate low examples but ADASYN takes into account the density of distribution to distribute the data points evenly.

Syntax:

from imblearn.over_sampling import SMOTE, ADASYN

Parameters(optional):*, sampling_strategy=’auto’, random_state=None, n_neighbors=5, n_jobs=None*

Implementation:

smote = SMOTE(ratio=’minority’)

X_smote,Y_smote=smote.fit_resample(X,Y)

Return Type:*a matrix with the shape of n_samples*n_features*

Example:

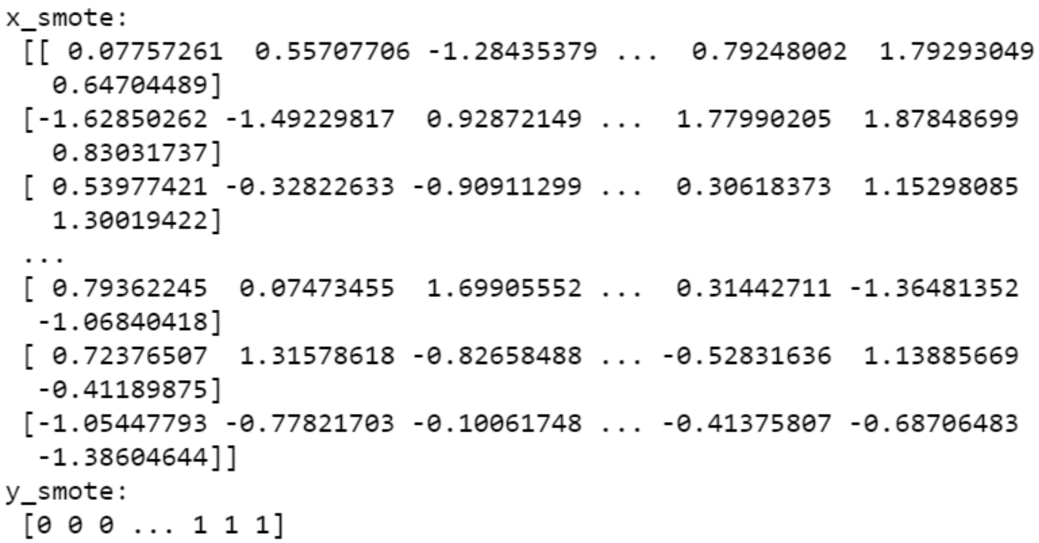
Python3

```
# import required modules
from sklearn.datasets import make_classification
from imblearn.over_sampling import SMOTE

# define dataset
x, y = make_classification(n_samples=10000, weights=[0.99], flip_y=0)
smote = SMOTE()
x_smote, y_smote = smote.fit_resample(x, y)

# print the features and the labels
print('x_smote:\n', x_smote)
print('y_smote:\n', y_smote)
```

Output:



```
x_smote:
 [[ 0.07757261  0.55707706 -1.28435379 ...  0.79248002  1.79293049
   0.64704489]
 [-1.62850262 -1.49229817  0.92872149 ...  1.77990205  1.87848699
   0.83031737]
 [ 0.53977421 -0.32822633 -0.90911299 ...  0.30618373  1.15298085
   1.30019422]
 ...
 [ 0.79362245  0.07473455  1.69905552 ...  0.31442711 -1.36481352
  -1.06840418]
 [ 0.72376507  1.31578618 -0.82658488 ... -0.52831636  1.13885669
  -0.41189875]
 [-1.05447793 -0.77821703 -0.10061748 ... -0.41375807 -0.68706483
  -1.38604644]]
y_smote:
 [0 0 0 ... 1 1 1]
```

Undersampling

- **Edited Nearest Neighbours:** This algorithm removes any sample which has labels different from those of its adjoining classes.

Syntax:

from imblearn.under_sampling import EditedNearestNeighbours

Parameters(optional): *sampling_strategy='auto', return_indices=False, random_state=None, n_neighbors=3, kind_sel='all', n_jobs=1, ratio=None*

Implementation:
en = EditedNearestNeighbours()
X_en,Y_en=en.fit_resample(X, y)

Return Type:*a matrix with the shape of n_samples*n_features*

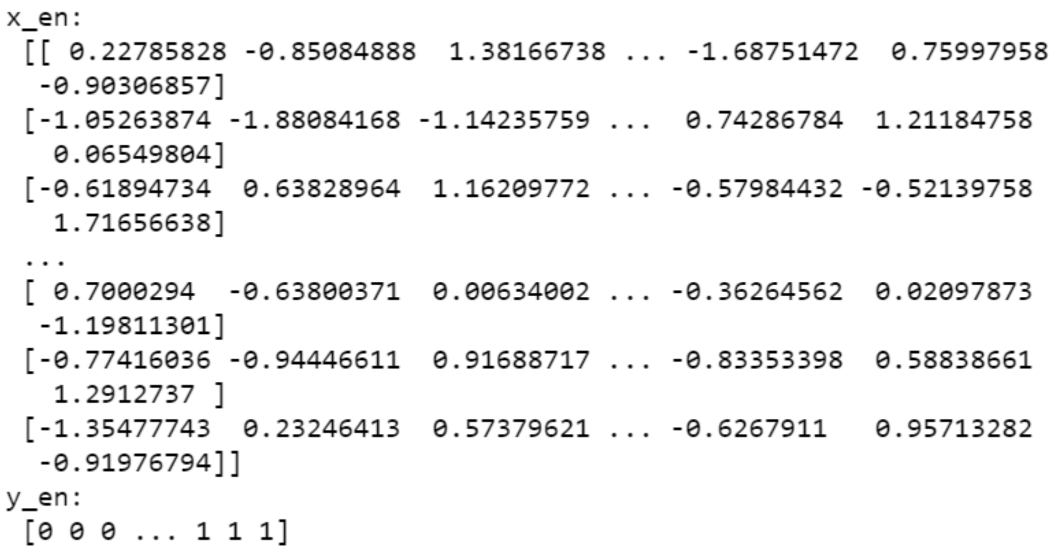
Example:

Python3

```
# import required modules
# define dataset
x, y = make_classification(n_samples=10000, weights=[0.99], flip_y=0)
en = EditedNearestNeighbours()
x_en, y_en = en.fit_resample(x, y)

# print the features and the labels
print('x_en:\n', x_en)
print('y_en:\n', y_en)
```

Output:



```
x_en:
 [[ 0.22785828 -0.85084888  1.38166738 ... -1.68751472  0.75997958
  -0.90306857]
 [-1.05263874 -1.88084168 -1.14235759 ...  0.74286784  1.21184758
   0.06549804]
 [-0.61894734  0.63828964  1.16209772 ... -0.57984432 -0.52139758
   1.71656638]
 ...
 [ 0.7000294  -0.63800371  0.00634002 ... -0.36264562  0.02097873
  -1.19811301]
 [-0.77416036 -0.94446611  0.91688717 ... -0.83353398  0.58838661
   1.2912737 ]
 [-1.35477743  0.23246413  0.57379621 ... -0.6267911   0.95713282
  -0.91976794]]
y_en:
 [0 0 0 ... 1 1 1]
```

- **Random Under Sampler:** It involves sampling any random class with or without any replacement.

Syntax:

from imblearn.under_sampling import RandomUnderSampler

Parameters(optional): *sampling_strategy='auto', return_indices=False, random_state=None, replacement=False, ratio=None*

Implementation:

undersample = RandomUnderSampler()

X_under, y_under = undersample.fit_resample(X, y)

Return Type: *a matrix with the shape of n_samples*n_features*

Example:

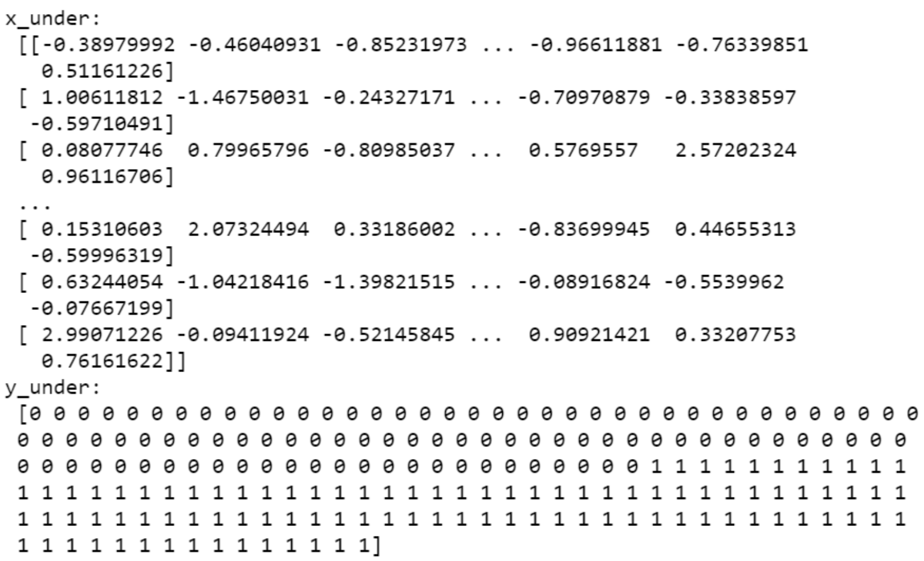
Python3

```
# import required modules
from sklearn.datasets import make_classification
from imblearn.under_sampling import RandomUnderSampler

# define dataset
x, y = make_classification(n_samples=10000,
                           weights=[0.99],
                           flip_y=0)
undersample = RandomUnderSampler()
x_under, y_under = undersample.fit_resample(x, y)

# print the features and the labels
print('x_under:\n', x_under)
print('y_under:\n', y_under)
```

Output:



```
x_under:
 [[-0.38979992 -0.46040931 -0.85231973 ... -0.96611881 -0.76339851
   0.51161226]
 [ 1.00611812 -1.46750031 -0.24327171 ... -0.70970879 -0.33838597
  -0.59710491]
 [ 0.08077746  0.79965796 -0.80985037 ...  0.5769557   2.57202324
   0.96116706]
 ...
 [ 0.15310603  2.07324494  0.33186002 ... -0.83699945  0.44655313
  -0.59996319]
 [ 0.63244054 -1.04218416 -1.39821515 ... -0.08916824 -0.5539962
  -0.07667199]
 [ 2.99071226 -0.09411924 -0.52145845 ...  0.90921421  0.33207753
   0.76161622]]
y_under:
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

Don't miss your chance to ride the wave of the data revolution! Every industry is scaling new heights by tapping into the power of data. Sharpen your skills and become a part of the hottest trend in the 21st century.

Last Updated : 11 Dec, 2020

1

Previous

Python | Calendar Module

Next

Reading and Writing lists to a file in Python

Share your thoughts in the comments

Add Your Comment

Similar Reads

ML | Handling Imbalanced Data with SMOTE and Near Miss Algorithm in Python

How to Handle Imbalanced Classes in Machine Learning

GeeksforGeeks Python Foundation Course - Learn Python in Hindi!

MySQL-Connector-Python module in Python

twitter-text-python (ttp) module - Python

10 reasons to learn Python in 2019

Implementing PCA in Python with scikit-learn

12 Reasons Why You Should Learn Python (2023)

Is it Worth to Learn Python in 2020?

Pipelines - Python and scikit-learn

Complete Tutorials

Advanced Python Tutorials

Python Automation Tutorial

OpenAI Python API - Complete Guide

S sangy987

Article Tags : python-modules , Python

Practice Tags : python

Additional Information

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305

Company

About Us

Legal

Careers

In Media

Contact Us

Advertise with us

GFG Corporate Solution

Placement Training Program

Apply for Mentor

Explore

Job-A-Thon Hiring Challenge

Hack-A-Thon

GfG Weekly Contest

Offline Classes (Delhi/NCR)

DSA in JAVA/C++

Master System Design

Master CP

GeeksforGeeks Videos

Geeks Community

Languages

Python

Java

C++

PHP

GoLang

SQL

R Language

Android Tutorial

Tutorials Archive

DSA

Data Structures

Algorithms

DSA for Beginners

Basic DSA Problems

DSA Roadmap

Top 100 DSA Interview Problems

DSA Roadmap by Sandeep Jain

All Cheat Sheets

Data Science & ML

Data Science With Python

Data Science For Beginner

Machine Learning Tutorial

ML Maths

Data Visualisation Tutorial

Pandas Tutorial

NumPy Tutorial

NLP Tutorial

Deep Learning Tutorial

HTML & CSS

HTML

CSS

Web Templates

CSS Frameworks

Bootstrap

Tailwind CSS

SASS

LESS

Web Design

Python Programming Examples
Django Tutorial
Python Projects
Python Tkinter
Web Scraping
OpenCV Python Tutorial
Python Interview Question

GATE CS Notes
Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths

DevOps

Git
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

Competitive Programming

Top DS or Algo for CP
Top 50 Tree
Top 50 Graph
Top 50 Array
Top 50 String
Top 50 DP
Top 15 Websites for CP

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

JavaScript

JavaScript Examples
TypeScript
ReactJS
NextJS
AngularJS
NodeJS
Lodash
Web Browser

NCERT Solutions

Class 12
Class 11
Class 10
Class 9
Class 8

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science

Commerce

Accountancy

Business Studies

Indian Economics

Macroeconomics

Microeconimics

Statistics for Economics

Management & Finance

Management

HR Managament

Income Tax

Finance

Economics

UPSC Study Material

Polity Notes

Geography Notes

History Notes

Science and Technology Notes

Economy Notes

Ethics Notes

Previous Year Papers

SSC/ BANKING

SSC CGL Syllabus

SBI PO Syllabus

SBI Clerk Syllabus

IBPS PO Syllabus

IBPS Clerk Syllabus

SSC CGL Practice Papers

Colleges

Indian Colleges Admission & Campus Experiences

List of Central Universities - In India

Colleges in Delhi University

IIT Colleges

NIT Colleges

IIIT Colleges

Companies

IT Companies

Software Development Companies

Artificial Intelligence(AI) Companies

CyberSecurity Companies

Service Based Companies

Product Based Companies

PSUs for CS Engineers

Preparation Corner

Company Wise Preparation

Preparation for SDE

Experienced Interviews

Internship Interviews

Competitive Programming

Aptitude Preparation

Exams

JEE Mains

JEE Advanced

GATE CS

NEET

UGC NET

More Tutorials

Software Development

Software Testing

Product Management

SAP

SEO - Search Engine Optimization

Linux

Excel

Write & Earn

Write an Article

Improve an Article

Pick Topics to Write

Share your Experiences

Internships