How to convert a Scikit-learn dataset to a Pandas dataset?

How do I convert data from a Scikit-learn Bunch object to a Pandas DataFrame?

```
from sklearn.datasets import load_iris
import pandas as pd
data = load_iris()
print(type(data))
data1 = pd. # Is there a Pandas method to accomplish this?
```

dataset scikit-learn pandas

asked Jun 27 '16 at 7:28

SANBI samples
299 1 3 15

migrated from stats.stackexchange.com Jun 29 '16 at 16:42

This question came from our site for people interested in statistics, machine learning, data analysis, data mining, and data visualization.

7 Answers

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris

# save load_iris() sklearn dataset to iris
# if you'd like to check dataset type use: type(load_iris())
# if you'd like to view list of attributes use: dir(load_iris())
iris = load_iris()

# np.c_ is the numpy concatenate function
# which is used to concat iris['data'] and iris['target'] arrays
# for pandas column argument: concat iris['feature_names'] list
# and string list (in this case one string); you can make this anything you'd
like..
# the original dataset would probably call this ['Species']
data1 = pd.DataFrame(data= np.c_[iris['data'], iris['target']],
                     columns= iris['feature_names'] + ['target'])
```

edited Dec 7 '16 at 23:57
rolyat
310 2 8

answered Jun 29 '16 at 13:26
TomDLT
934 5 14

3 Can you add a little text to explain this code? This is somewhat brief by our standards. – gung Jun 29 '16 at 14:09

1 Some bunches have the feature_names as a ndarray which will break the columns parameter. – user1969453 Jul 10 '17 at 1:17

Missing "Species" key and values for dataframe. – mastash3ff Jul 11 '17 at 15:24

Species is no longer available in the latest iris data frame, as far as I can tell. They are replaced by target_names. – Kingz Jul 13 '17 at 5:14

1 This code didn't work as-is for me. For the columns parameter, I needed to pass in columns=np.append(iris['feature_names'], 'target). Did I do something wrong, or does this answer need an edit? – Josh Davis Oct 2 '17 at 1:35

This tutorial maybe of interest: http://www.neural.cz/dataset-exploration-boston-house-pricing.html

answered Apr 21 '17 at 22:40
justin4480
121 1 3

2 Need to concatenate the data with target: df = pd.DataFrame(np.concatenate((iris.data, np.array([iris.target]).T), axis=1), columns=iris.feature_names + ['target']) – Tyler Apr 26 '17 at 7:06

TOMDLt's solution is not generic enough for all the datasets in scikit-learn. For example it does not work for the boston housing dataset. I propose a different solution which is more universal. No need to use numpy as well.

```
from sklearn import datasets
import pandas as pd

boston_data = datasets.load_boston()
df_boston = pd.DataFrame(boston_data.data,columns=boston_data.feature_names)
df_boston['target'] = pd.Series(boston_data.target)
df_boston.head()
```

edited Sep 26 '17 at 19:37

answered Sep 23 '17 at 13:03
Nilav Baran Ghosh
73 2 8

This works for me.

```
dataFrame = pd.dataFrame(data = np.c_[ [iris['data'],iris['target'] ],
columns=iris['feature_names'].tolist() + ['target'])
```

Just as an alternative that I could wrap my head around much easier:

```
data = load_iris()
df = pd.DataFrame(data['data'], columns=data['feature_names'])
df['target'] = data['target']
df.head()
```

Basically instead of concatenating from the get go, just make a data frame with the matrix of features and then just add the target column with data['whatvername'] and grab the target values from the dataset

answered Oct 7 '17 at 18:48

daguito81
21 1

Working off the best answer and addressing my comment, here is a function for the conversion

```
def bunch_to_dataframe(bunch):
  fnames = bunch.feature_names
  features = fnames.tolist() if isinstance(fnames, np.ndarray) else fnames
  features += ['target']
  return pd.DataFrame(data= np.c_[bunch['data'], bunch['target']],
                 columns=features)
```

answered Jul 10 '17 at 2:09

user1969453

There might be a better way but here is what I have done in the past and it works quite well:

```
items = data.items()                          #Gets all the data from this Bunch
```

```
the dataframe
```

Now mydata will have everything you need - attributes, target variable and columnnames

answered Jun 29 '16 at 17:09
Shanbhag
102 10

1 The solution by TomDLT is much superior than what I am suggesting above. It does the same thing but is very elegant and easy to understand. Use that! – Shanbhag Jun 29 '16 at 17:22

`mydata = pd.DataFrame(items[1][1])` throws `TypeError: 'dict_items' object does not support indexing` – SANBI samples Jul 1 '16 at 7:31