# Scikit Learn CountVectorizer



I'm trying to compute a simple word frequency using scikit-learn's `CountVectorizer` .

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer

texts=["dog cat fish","dog cat cat","fish bird","bird"]
cv = CountVectorizer()
cv_fit=cv.fit_transform(texts)

print cv.vocabulary_
{u'bird': 0, u'cat': 1, u'dog': 2, u'fish': 3}
```

I was expecting it to return `{u'bird': 2, u'cat': 3, u'dog': 2, u'fish': 2}` .

python scikit-learn

edited Dec 15 '14 at 16:28
matsjoyce
**4,554** 6 19 37

asked Dec 15 '14 at 16:20
Adrien
**53** 1 6

`CountVectorizer` creates *"A mapping of terms to feature indices"* - if you just want the frequency, why not use `collections.Counter` ? – jonrsharpe Dec 15 '14 at 16:25

## 1 Answer

`cv.vocabulary_` in this instance is a dict, where the keys are the words (features) that you've found and the values are indices, which is why they're `0, 1, 2, 3` . It's just bad luck that it looked similar to your counts :)

You need to work with the `cv_fit` object to get the counts

```
from sklearn.feature_extraction.text import CountVectorizer

texts=["dog cat fish","dog cat cat","fish bird", 'bird']
cv = CountVectorizer()
cv_fit=cv.fit_transform(texts)

print(cv.get_feature_names())
print(cv_fit.toarray())
#['bird', 'cat', 'dog', 'fish']
#[[0 1 1 1]
# [0 2 1 0]
# [1 0 0 1]
# [1 0 0 0]]
```

Each row in the array is one of your original documents (strings), each column is a feature (word), and the element is the count for that particular word and document. You can see that if

you sum each column you'll get the correct number

```
print(cv_fit.toarray().sum(axis=0))
#[2 3 2 2]
```

Honestly though, I'd suggest using `collections.Counter` or something from NLTK, unless you have some specific reason to use scikit-learn, as it'll be simpler.

edited Dec 15 '14 at 16:43

answered Dec 15 '14 at 16:37

Ffisegydd

**19.7k** 4 55 79