Evaluation of Clustering

CS5154/6054

Yizong Cheng

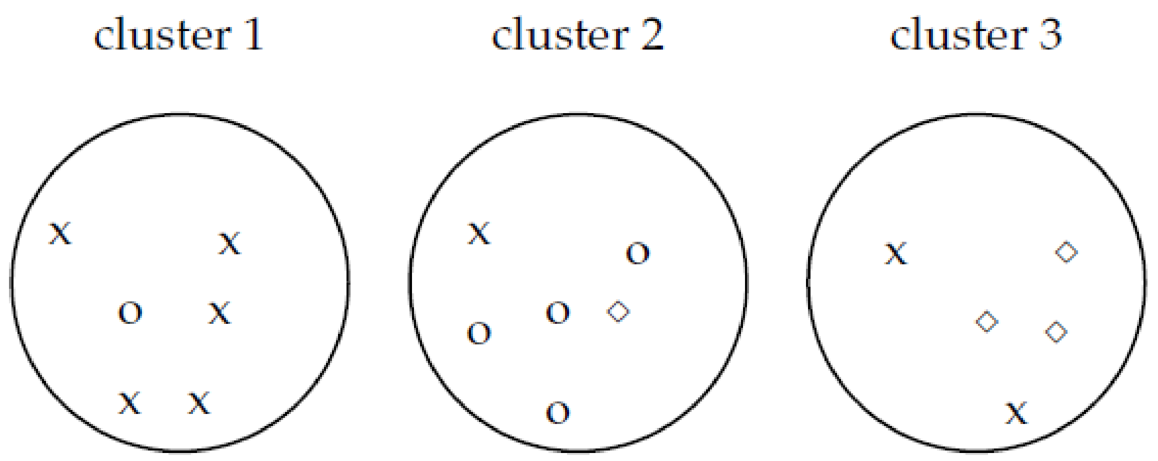
11/3/2022

16.3 Evaluation of clustering

This section introduces four external criteria of clustering quality. *Purity* is a simple and transparent evaluation measure. *Normalized mutual information* can be information-theoretically interpreted. The *Rand index* penalizes both false positive and false negative decisions during clustering. The *F measure* in addition supports differential weighting of these two types of errors.

$$\text{purity}(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j| \tag{16.1}$$

where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$ is the set of classes. We interpret ω_k as the set of documents in ω_k and c_j as the set of documents in c_j in Equation (16.1).



▶ **Figure 16.4** Purity as an external evaluation criterion for cluster quality. Majority class and number of members of the majority class for the three clusters are: x, 5 (cluster 1); o, 4 (cluster 2); and ⋄, 3 (cluster 3). Purity is $(1/17) \times (5+4+3) \approx 0.71$.

	purity	NMI	RI	F_5
lower bound	0.0	0.0	0.0	0.0
maximum	1	1	1	1
value for Figure 16.4	0.71	0.36	0.68	0.46

▶ **Table 16.2** The four external evaluation measures applied to the clustering in Figure 16.4.

mation or *NMI*:

(16.2) $$\text{NMI}(\Omega, \mathbb{C}) = \frac{I(\Omega; \mathbb{C})}{[H(\Omega) + H(\mathbb{C})]/2}$$

I is mutual information (cf. Chapter 13, page 272):

(16.3) $$I(\Omega; \mathbb{C}) = \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)}$$

(16.4) $$= \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|}$$

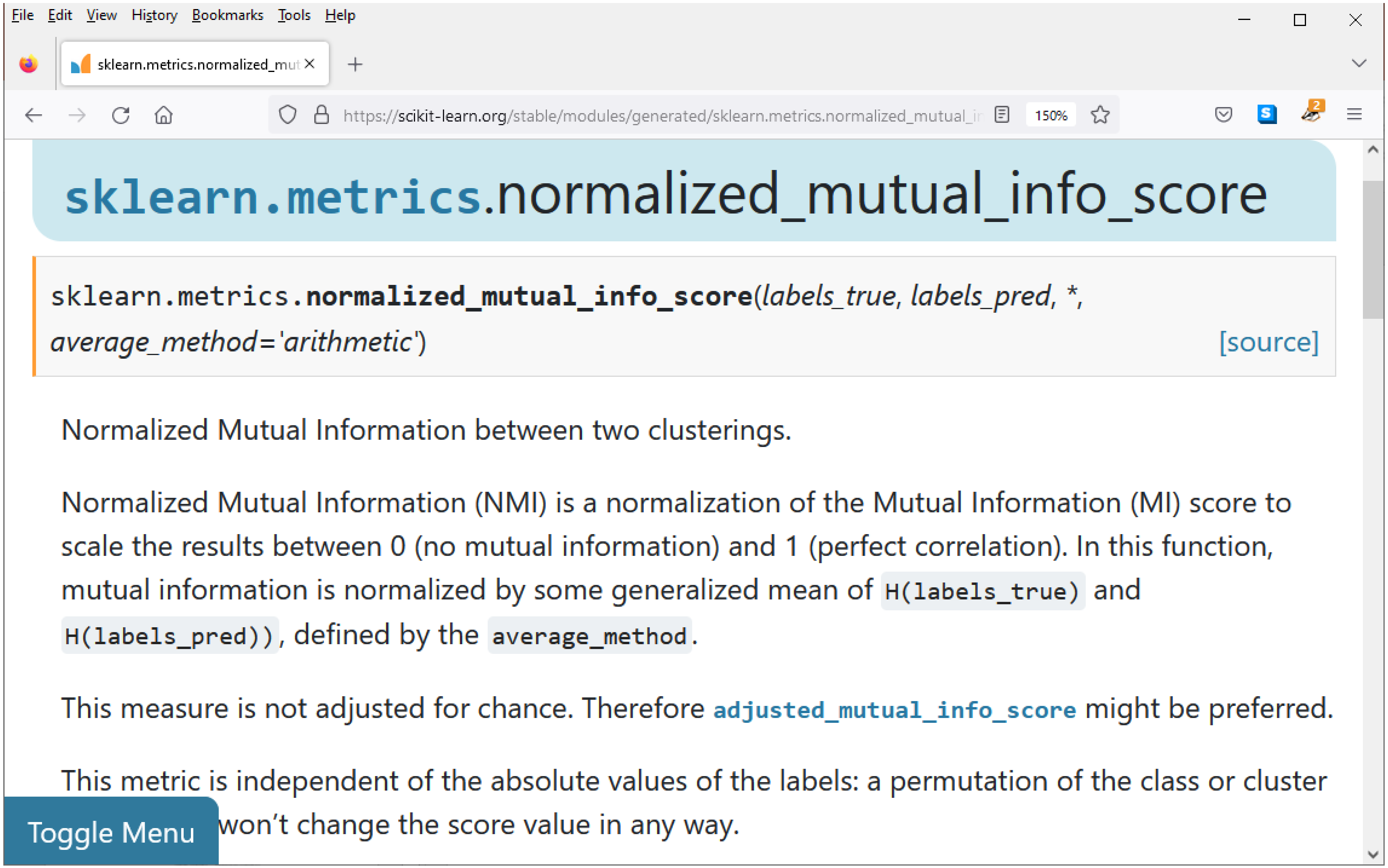
where $P(\omega_k)$, $P(c_j)$, and $P(\omega_k \cap c_j)$ are the probabilities of a document being in cluster ω_k, class c_j, and in the intersection of ω_k and c_j, respectively. Equation (16.4) is equivalent to Equation (16.3) for maximum likelihood estimates of the probabilities (i.e., the estimate of each probability is the corresponding relative frequency).

H is entropy as defined in Chapter 5 (page 99):

(16.5) $$H(\Omega) = -\sum_k P(\omega_k) \log P(\omega_k)$$

(16.6) $$= -\sum_k \frac{|\omega_k|}{N} \log \frac{|\omega_k|}{N}$$

where, again, the second equation is based on maximum likelihood estimates of the probabilities.


File Edit View History Bookmarks Tools Help
sklearn.metrics.normalized_mut
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.normalized_mutual_i
150%
Toggle Menu

sklearn.metrics.normalized_mutual_info_score

sklearn.metrics.**normalized_mutual_info_score**(*labels_true*, *labels_pred*, ***, *average_method='arithmetic'*) [source]

Normalized Mutual Information between two clusterings.

Normalized Mutual Information (NMI) is a normalization of the Mutual Information (MI) score to scale the results between 0 (no mutual information) and 1 (perfect correlation). In this function, mutual information is normalized by some generalized mean of `H(labels_true)` and `H(labels_pred))`, defined by the `average_method`.

This measure is not adjusted for chance. Therefore `adjusted_mutual_info_score` might be preferred.

This metric is independent of the absolute values of the labels: a permutation of the class or cluster won’t change the score value in any way.

An alternative to this information-theoretic interpretation of clustering is to view it as a series of decisions, one for each of the $N(N-1)/2$ pairs of documents in the collection. We want to assign two documents to the same cluster if and only if they are similar. A true positive (TP) decision assigns two similar documents to the same cluster, a true negative (TN) decision assigns two dissimilar documents to different clusters. There are two types of errors we can commit. A false positive (FP) decision assigns two dissimilar documents to the same cluster. A false negative (FN) decision assigns two similar documents to different clusters. RAND INDEX RI The *Rand index* (RI) measures the percentage of decisions that are correct. That is, it is simply accuracy (Section 8.3, page 155).

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}}$$

As an example, we compute RI for Figure 16.4. We first compute TP + FP. The three clusters contain 6, 6, and 5 points, respectively, so the total number of "positives" or pairs of documents that are in the same cluster is:

$$\mathrm{TP} + \mathrm{FP} = \binom{6}{2} + \binom{6}{2} + \binom{5}{2} = 40$$

Of these, the x pairs in cluster 1, the o pairs in cluster 2, the $\diamond$ pairs in cluster 3, and the x pair in cluster 3 are true positives:

$$\mathrm{TP} = \binom{5}{2} + \binom{4}{2} + \binom{3}{2} + \binom{2}{2} = 20$$

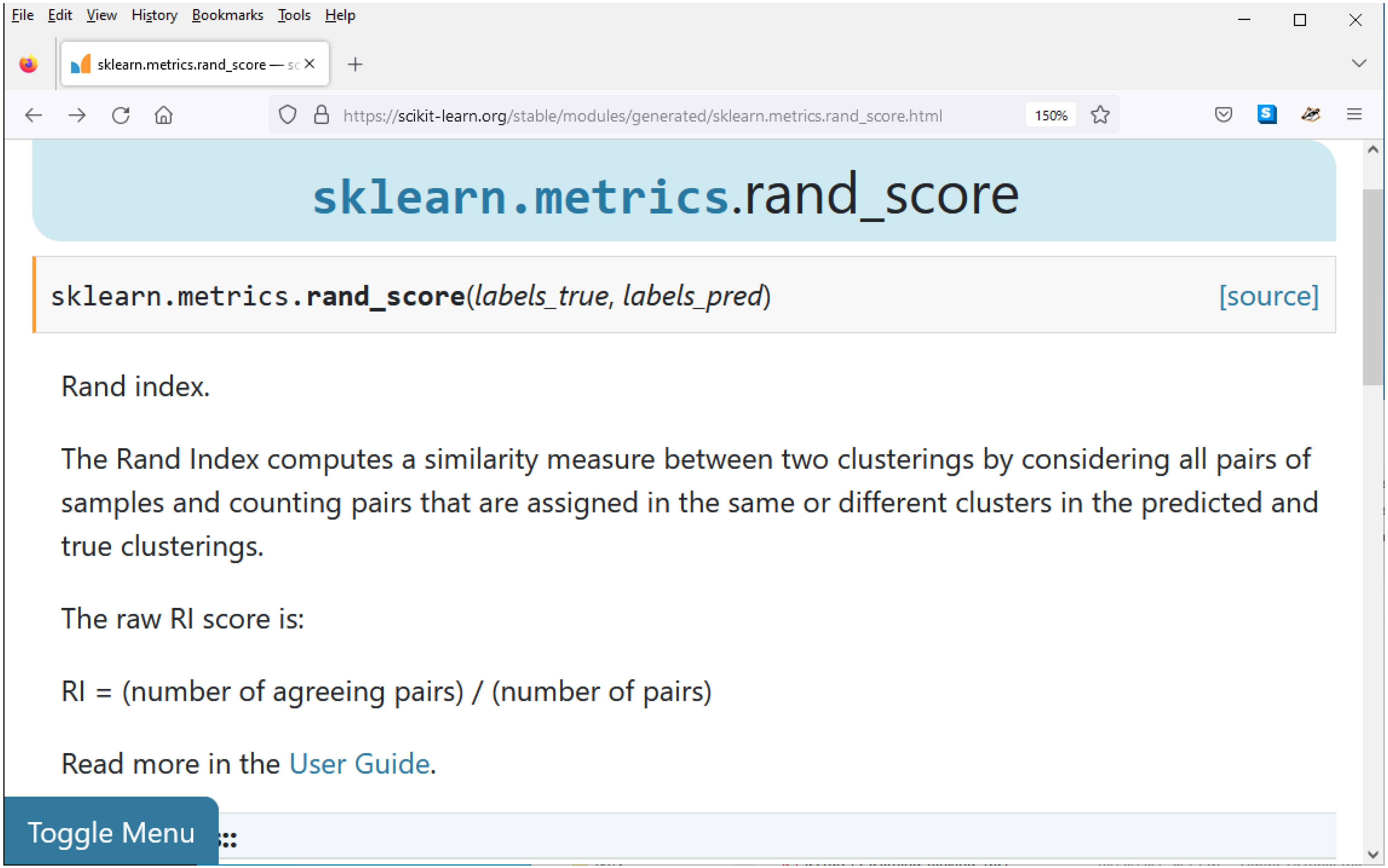
Thus, FP $= 40 - 20 = 20$.

FN and TN are computed similarly, resulting in the following contingency table:

	Same cluster	Different clusters
Same class	TP = 20	FN = 24
Different classes	FP = 20	TN = 72

RI is then $(20 + 72)/(20 + 20 + 24 + 72) \approx 0.68$.

The Rand index gives equal weight to false positives and false negatives.


File Edit View History Bookmarks Tools Help
sklearn.metrics.rand_score — sc
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.rand_score.html
150%

sklearn.metrics.rand_score

sklearn.metrics.**rand_score**(*labels_true*, *labels_pred*) [source]

Rand index.

The Rand Index computes a similarity measure between two clusterings by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusterings.

The raw RI score is:

RI = (number of agreeing pairs) / (number of pairs)

Read more in the User Guide.

F MEASURE

The Rand index gives equal weight to false positives and false negatives. Separating similar documents is sometimes worse than putting pairs of dissimilar documents in the same cluster. We can use the *F measure* (Section 8.3, page 154) to penalize false negatives more strongly than false positives by selecting a value $\beta > 1$, thus giving more weight to recall.

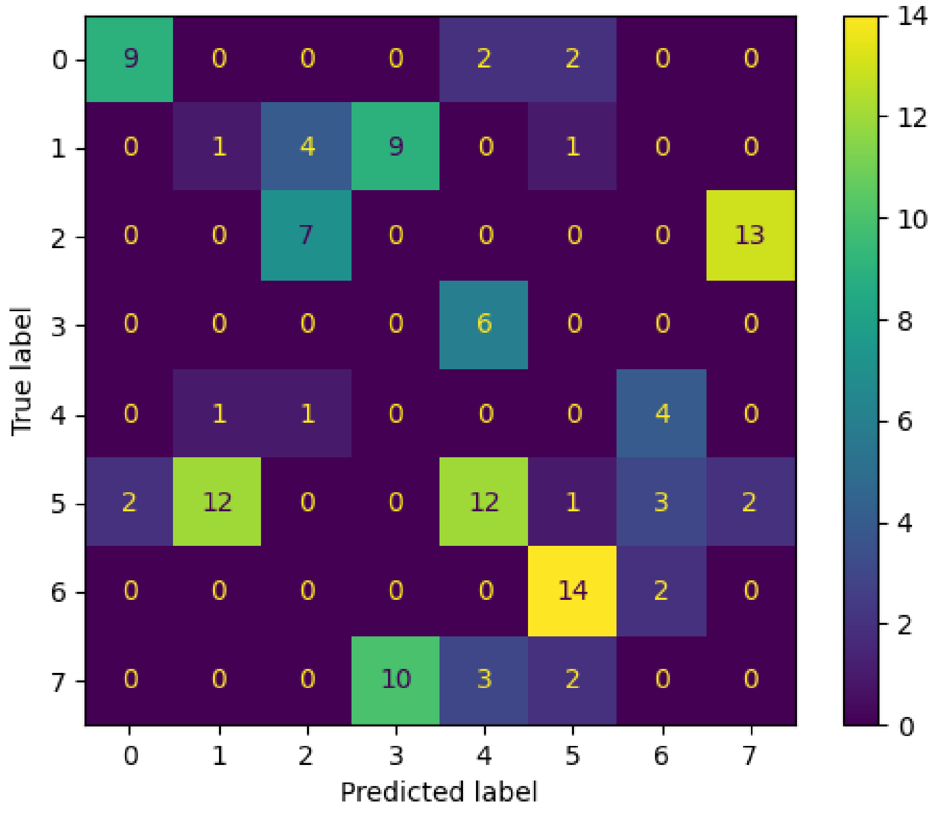
$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad R = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

Based on the numbers in the contingency table, $P = 20/40 = 0.5$ and $R = 20/44 \approx 0.455$. This gives us $F_1 \approx 0.48$ for $\beta = 1$ and $F_5 \approx 0.456$ for $\beta = 5$. In information retrieval, evaluating clustering with F has the advantage that the measure is already familiar to the research community.

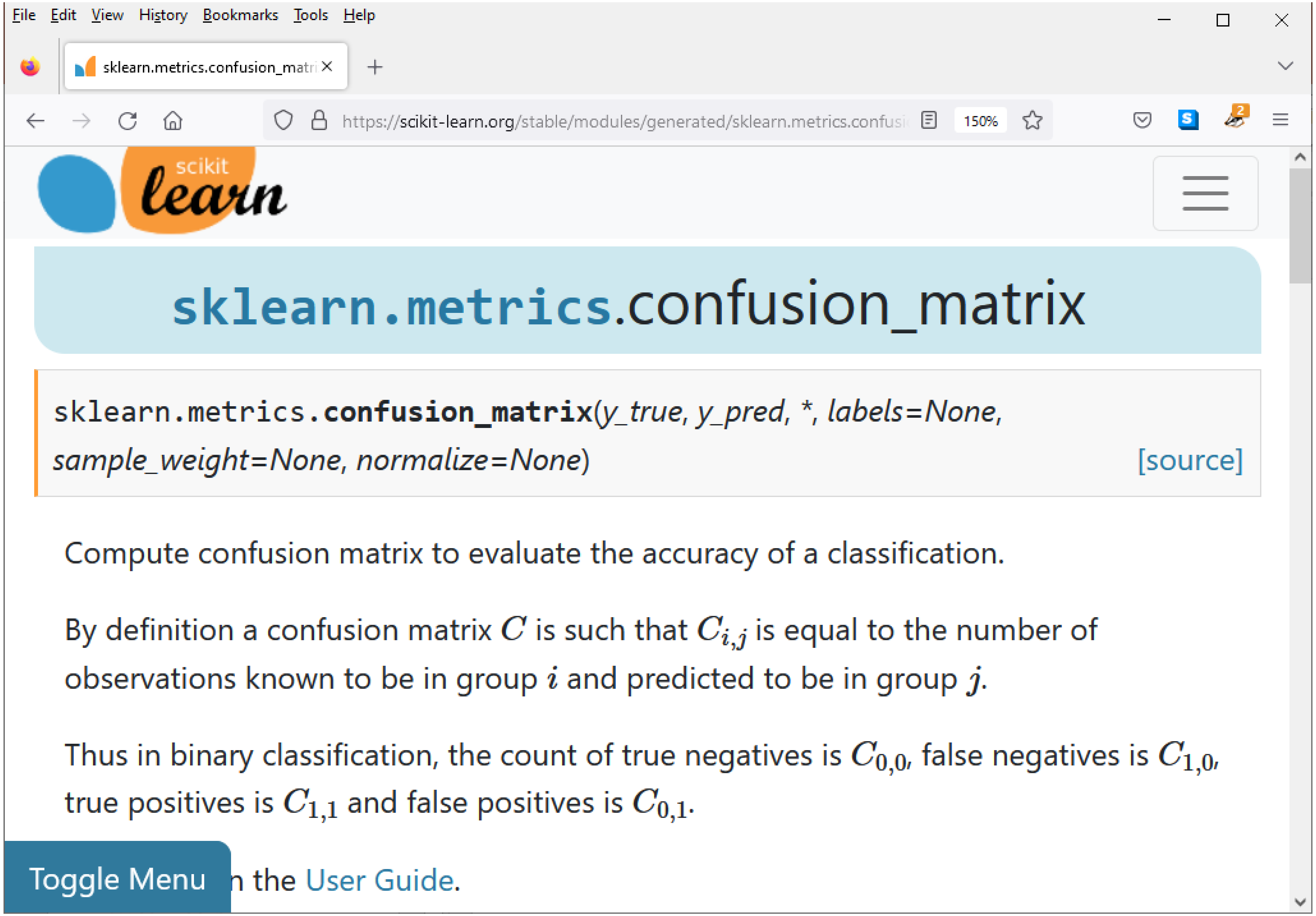
```
# IR19A.py CS5154/6054 cheng 2022
# twice k-means
# confusion matrix
# NMI
# Usage: python IR19A.py

import numpy as np
import random
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay, normalized_mutual_info_score
import matplotlib.pyplot as plt

f = open("gutprotocol.txt", "r", encoding="utf8")
docs = f.readlines()
f.close()
N = len(docs)
```

```
cv = TfidfVectorizer(max_df=0.4, min_df=4)
X = cv.fit_transform(docs)

model = KMeans(n_init=1, max_iter=5)
model.fit_predict(X)
y1 = model.labels_
print(y1)

model.fit_predict(X)
y2 = model.labels_
print(y2)

cm = confusion_matrix(y1, y2)
print(cm)
disp = ConfusionMatrixDisplay(cm)
disp.plot()
plt.show()

print(normalized_mutual_info_score(y1, y2))
```

```
0.5783707218490585
```



	0	1	2	3	4	5	6	7
0	9	0	0	0	2	2	0	0
1	0	1	4	9	0	1	0	0
2	0	0	7	0	0	0	0	13
3	0	0	0	0	6	0	0	0
4	0	1	1	0	0	0	4	0
5	2	12	0	0	12	1	3	2
6	0	0	0	0	0	14	2	0
7	0	0	0	10	3	2	0	0


File Edit View History Bookmarks Tools Help
sklearn.metrics.confusion_matri
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusi
150%
scikit learn

sklearn.metrics.confusion_matrix

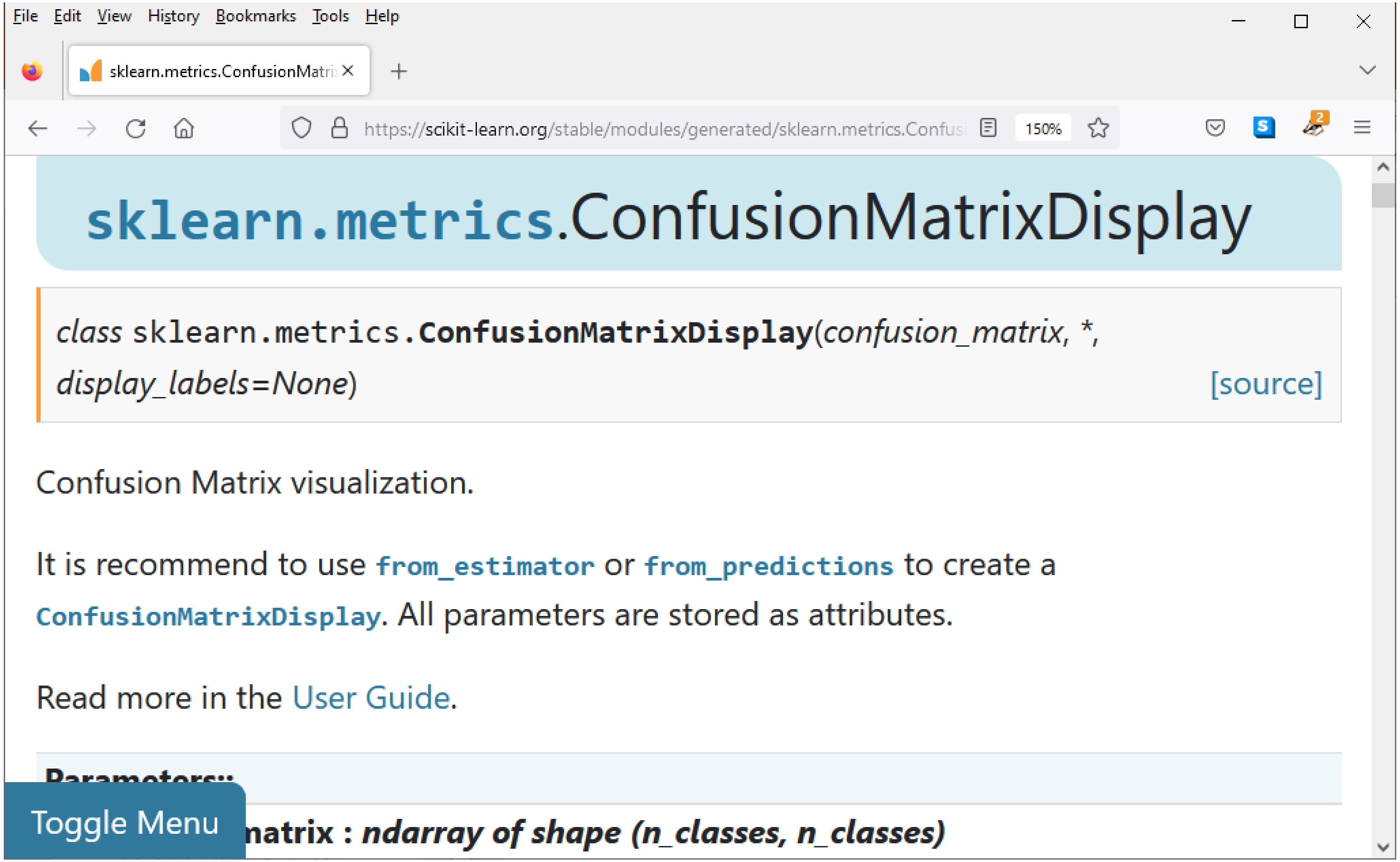
`sklearn.metrics.`**`confusion_matrix`**(*y_true*, *y_pred*, ***, *labels=None*, *sample_weight=None*, *normalize=None*) [source]

Compute confusion matrix to evaluate the accuracy of a classification.

By definition a confusion matrix C is such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j.

Thus in binary classification, the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$.

Toggle Menu n the User Guide.


File Edit View History Bookmarks Tools Help
sklearn.metrics.ConfusionMatri
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.Confusi
150%
Toggle Menu

sklearn.metrics.ConfusionMatrixDisplay

class sklearn.metrics.**ConfusionMatrixDisplay**(*confusion_matrix*, ***, *display_labels=None*) [source]

Confusion Matrix visualization.

It is recommend to use `from_estimator` or `from_predictions` to create a `ConfusionMatrixDisplay`. All parameters are stored as attributes.

Read more in the User Guide.

Parameters:

natrix : *ndarray of shape (n_classes, n_classes)*

```python
inverted = {}
for i in range(N):
    if not y1[i] in inverted:
        s = set()
        s.add(i)
        inverted.update({y1[i]: s})
    else:
        inverted.get(y1[i]).add(i)

for x, y in inverted.items():
    print(x, y)
```

```
[5 2 2 0 5 5 2 5 2 5 6 5 4 0 5 2 0 0 5 5 5 5 0 2 5 0 0 5 0 6 5 4 1 5 2 5
 5 4 4 7 2 2 1 7 7 5 0 5 5 5 5 0 7 7 5 7 7 7 7 1 7 7 0 5 2 2 7 2 2 2 2 1
 1 2 1 1 1 4 7 0 0 6 6 6 6 6 6 5 7 3 3 4 5 2 6 6 6 6 6 6 6 6 3 3 3 3 1 5 1
 5 1 1 5 1 5 1 7 2 5 1 2]
[4 7 7 0 4 7 7 4 7 1 6 0 6 0 4 2 0 0 4 6 6 1 6 0 7 1 0 0 1 0 6 0 6 5 4 7 4
 1 6 6 5 7 7 3 3 4 4 0 7 4 4 1 4 3 3 1 3 3 3 3 3 3 3 5 4 7 7 4 2 2 7 2 2 2
 2 2 3 3 3 2 3 4 5 5 5 5 5 5 5 4 4 4 4 1 1 7 5 5 5 5 5 5 5 5 4 4 4 4 3 1 3
 1 1 3 1 3 1 2 5 7 5 2 2]
```

```
0.5783707218490585
5 {0, 4, 5, 7, 9, 11, 14, 18, 19, 20, 21, 22, 25, 28, 31, 34, 36, 37, 46, 48, 49, 50, 51, 55, 64, 89, 94, 109, 111, 114, 116, 120}
2 {1, 2, 6, 8, 15, 24, 35, 41, 42, 65, 66, 68, 69, 70, 71, 72, 75, 95, 119, 122}
0 {3, 13, 47, 16, 17, 81, 82, 52, 23, 26, 27, 29, 63}
6 {96, 97, 98, 99, 100, 101, 102, 103, 10, 83, 84, 85, 86, 87, 88, 30}
4 {32, 38, 39, 12, 79, 93}
1 {33, 73, 74, 43, 76, 77, 78, 108, 110, 112, 113, 115, 117, 121, 60}
7 {67, 90, 40, 44, 45, 80, 53, 54, 118, 56, 57, 58, 59, 61, 62}
3 {104, 105, 106, 107, 91, 92}
```

Implementing Purity

- cm = confusion_matrix(y1, y2)
- purity = sum(np.amax(cm, axis=0))/N
- But we can also have
- purity = sum(np.amax(cm, axis=1))/N
- Are these the same?