Evaluating Clusters

Learn how to evaluate the performance of clustering algorithms.

Chapter Goals:

- Learn how to evaluate clustering algorithms

A. Evaluation metrics

When we don't have access to any true cluster assignments (labels), the best we can do to evaluate clusters is to just take a look at them and see if they make sense with respect to the dataset and domain. However, if we do have access to the true cluster labels for the data observations, we can apply a number of metrics to evaluate our clustering algorithm.

One popular evaluation metric is the adjusted Rand index. The regular Rand index gives a measurement of similarity between the true clustering assignments (true labels) and the predicted clustering assignments (predicted labels). The adjusted Rand index (ARI) is a corrected-for-chance version of the regular one, meaning that the score is adjusted so that random clustering assignments will not have a good score.

The ARI value ranges from -1 to 1, inclusive. Negative scores represent bad labelings, random labelings will get a score near 0, and perfect labelings get a score of 1.

In scikit-learn, ARI is implemented through the `adjusted_rand_score` function (part of the `metrics` module). It takes in two required arguments, the true cluster labels and the predicted cluster labels, and returns the ARI score.

```python
from sklearn.metrics import adjusted_rand_score
true_labels = np.array([0, 0, 0, 1, 1, 1])
pred_labels = np.array([0, 0, 1, 1, 2, 2])

ari = adjusted_rand_score(true_labels, pred_labels)
print('{}\n'.format(ari))

# symmetric
ari = adjusted_rand_score(pred_labels, true_labels)
print('{}\n'.format(ari))

# Perfect labeling
perf_labels = np.array([0, 0, 0, 1, 1, 1])
ari = adjusted_rand_score(true_labels, perf_labels)
print('{}\n'.format(ari))

# Perfect labeling, permuted
permuted_labels = np.array([1, 1, 1, 0, 0, 0])
ari = adjusted_rand_score(true_labels, permuted_labels)
print('{}\n'.format(ari))

renamed_labels = np.array([1, 1, 1, 3, 3, 3])
# Renamed labels to 1, 3
ari = adjusted_rand_score(true_labels, renamed_labels)
print('{}\n'.format(ari))

true_labels2 = np.array([0, 1, 2, 0, 3, 4, 5, 1])
# Bad labeling
pred_labels2 = np.array([1, 1, 0, 0, 2, 2, 2, 2])
ari = adjusted_rand_score(true_labels2, pred_labels2)
print('{}\n'.format(ari))
```

RUN SAVE RESET

Close

Output 1.179s

```
0.24242424242424246

0.24242424242424246

1.0

1.0

1.0
```

Note that the `adjusted_rand_score` function is symmetric. This means that changing the order of the arguments will not affect the score. Furthermore, permutations in the labeling or changing the label names (i.e. `0` and `1` vs. `1` and `3`) does not affect the score.

Another common clustering evaluation metric is adjusted mutual information (AMI). It is implemented in scikit-learn with the `adjusted_mutual_info_score` function (also part of the `cluster` module). Like `adjusted_rand_score`, the function is symmetric and oblivious to permutations and renamed labels.

```
from sklearn.metrics import adjusted_mutual_info_score
true_labels = np.array([0, 0, 0, 1, 1, 1])
pred_labels = np.array([0, 0, 1, 1, 2, 2])

ami = adjusted_mutual_info_score(true_labels, pred_labels)
print('{}\n'.format(ami))

# symmetric
ami = adjusted_mutual_info_score(pred_labels, true_labels)
print('{}\n'.format(ami))

# Perfect labeling
perf_labels = np.array([0, 0, 0, 1, 1, 1])
ami = adjusted_mutual_info_score(true_labels, perf_labels)
print('{}\n'.format(ami))

# Perfect labeling, permuted
permuted_labels = np.array([1, 1, 1, 0, 0, 0])
ami = adjusted_mutual_info_score(true_labels, permuted_labels)
print('{}\n'.format(ami))

renamed_labels = np.array([1, 1, 1, 3, 3, 3])
# Renamed labels to 1, 3
ami = adjusted_mutual_info_score(true_labels, renamed_labels)
print('{}\n'.format(ami))

true_labels2 = np.array([0, 1, 2, 0, 3, 4, 5, 1])
# Bad labeling
pred_labels2 = np.array([1, 1, 0, 0, 2, 2, 2, 2])
ami = adjusted_mutual_info_score(true_labels2, pred_labels2)
print('{}\n'.format(ami))
```

RUN SAVE RESET

Close

Output 1.166s

```
0.2250422831983088

0.2250422831983088

1.0

1.0

1.0
```

The ARI and AMI metrics are very similar. They both assign a score of 1.0 to perfect labelings, a score near 0.0 to random labelings, and negative scores to poor labelings.

A general rule of thumb of when to use which: ARI is used when the true clusters are large and approximately equal sized, while AMI is used when the true clusters are unbalanced in size and there exist small clusters.