

kNN

Made Satria Wibawa, M.Eng.
2020

PENDAHULUAN

kNN

kNN merupakan singkatan dari k-nearest neighbour.

kNN termasuk ke dalam *instance based learning* atau yang sering disebut *memory based learning*.

Instance based learning tidak membentuk model hasil dari generalisasi data, melainkan membandingkan instance baru dengan dataset yang tersimpan di memory.

Algoritma klasifikasi ini adalah yang paling mudah dari algoritma yang ada.

ALGORITMA

Algoritma kNN

Algoritma :

kNN

Input :

k = jumlah cluster

D = dataset yang memiliki n buah objek

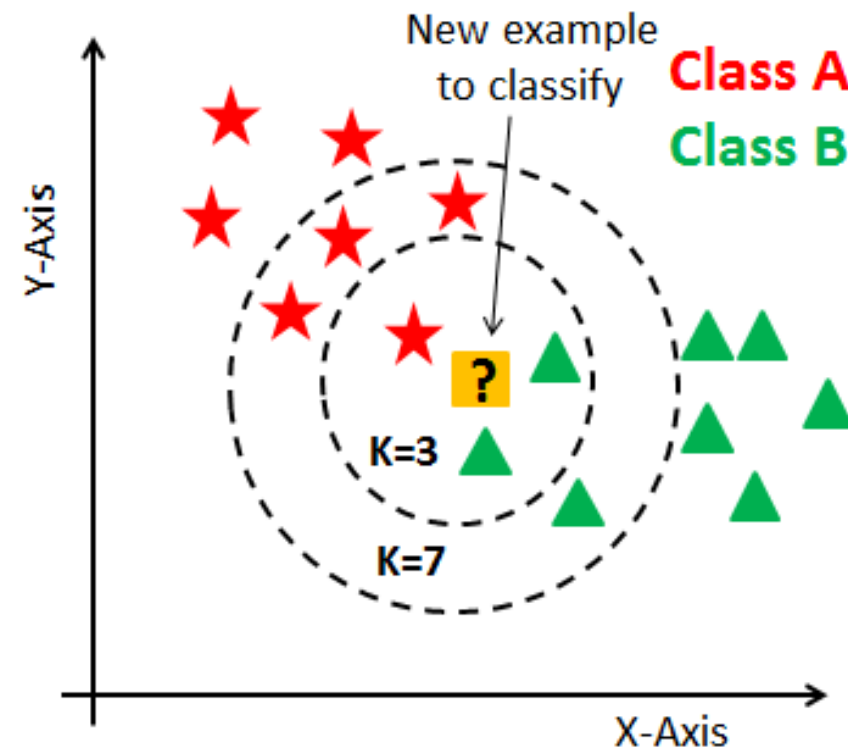
d = data tanpa kelas label

Output :

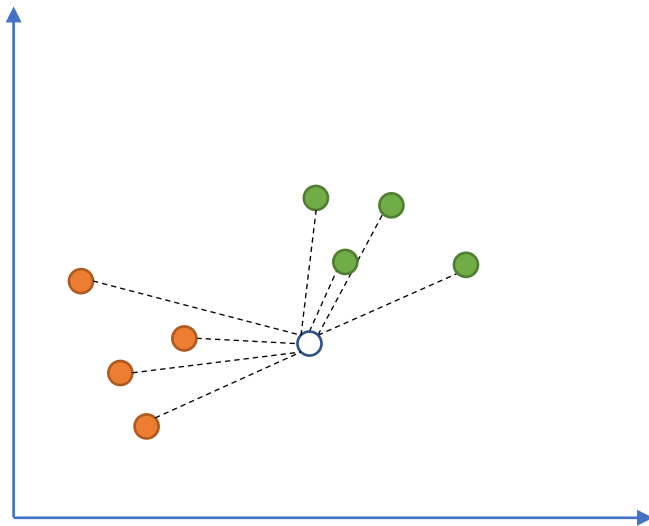
kelas label dari data

Metode :

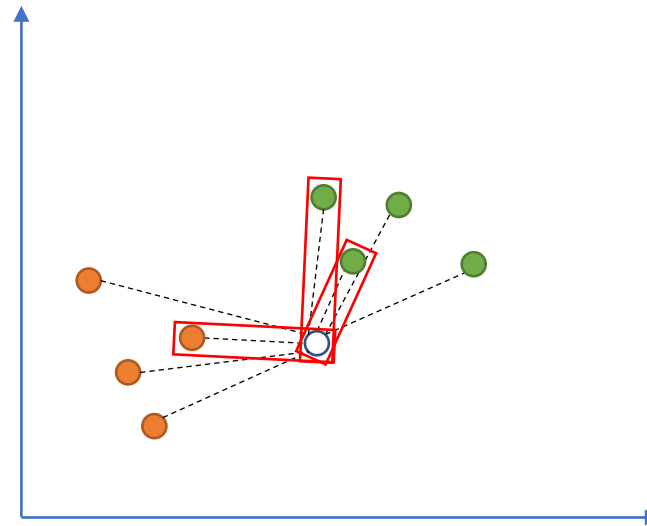
- 1) hitung jarak d dengan semua data pada D
- 2) urutkan berdasarkan jarak terkecil
- 3) voting kelas label sebanyak k , kelas terbanyak adalah kelas dari d



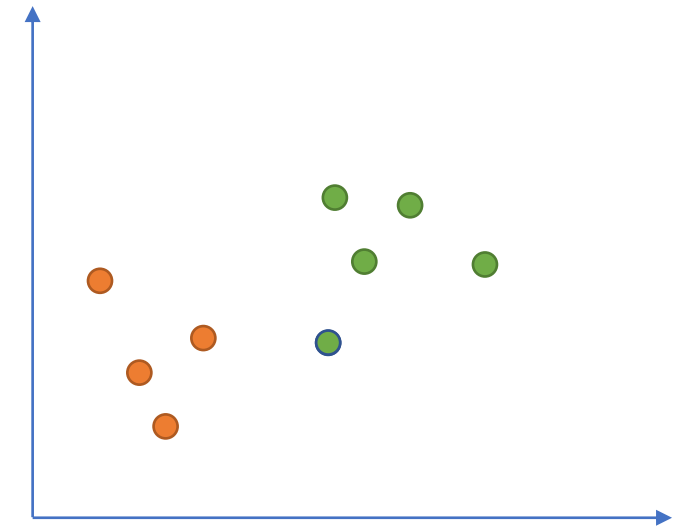
Tahapan kNN



hitung jarak object baru dengan dataset



tentukan k tetangga terdekat (nearest neighbour)



tentukan kelas berdasarkan voting

Perhitungan Jarak/Disimilaritas

- Jarak pada atribut numerik

$$\text{Euclidean : } D(X_1, X_2) = \sqrt{\sum_{i=1}^n (x_{1i} - x_{2i})^2}$$

$$\text{Manhattan : } D(X_1, X_2) = \sum_{i=1}^n |x_{1i} - x_{2i}|$$

$$\text{Minkowski : } D(X_1, X_2) = \left(\sum_{i=1}^n (|x_{1i} - x_{2i}|)^q \right)^{\frac{1}{q}}$$

- Jarak pada atribut nominal

$$\text{Hamming : } D(X_1, X_2) = \sum_{i=1}^n |x_{1i} - x_{2i}|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

PERHITUNGAN

kNN pada Data Numerik

Jika unit pada atribut berbeda, lakukan normalisasi tiap atribut dengan z-score

$$x' = \frac{x - \mu}{\sigma}$$

contoh:

index	umur	berat	tinggi	kelas
0	27	90	165	obesitas
1	26	65	158	normal
2	20	98	170	obesitas
3	23	75	178	normal

maka:

$$\mu_{umur} = 24.00 \quad \sigma_{umur} = 2.738$$

$$\mu_{berat} = 82.00 \quad \sigma_{berat} = 12.82$$

$$\mu_{tinggi} = 167.8 \quad \sigma_{tinggi} = 7.292$$

sehingga:

index	umur	berat badan	tinggi badan	kelas
0	1.095	0.623	-0.377	obesitas
1	0.730	-1.325	-1.336	normal
2	-1.460	1.247	0.308	obesitas
3	-0.365	-0.545	1.405	normal

kNN pada Data Numerik

dataset:

index	umur	berat badan	tinggi badan	kelas
0	1.095	0.623	-0.377	obesitas
1	0.730	-1.325	-1.336	normal
2	-1.460	1.247	0.308	obesitas
3	-0.365	-0.545	1.405	normal

objek baru

umur	berat badan	tinggi badan	kelas
22	72	158	?

umur	berat badan	tinggi badan	kelas
-0.730	-0.779	-1.336	?

hitung jarak

index	jarak	kelas	ranking
0	2.494	obesitas	2
1	1.559	normal	1
2	2.711	obesitas	3
3	2.776	normal	4

jika $k = 3$, maka

voting kelas

index	jarak	kelas	ranking
0	2.494	obesitas	2
1	1.559	normal	1
2	2.711	obesitas	3
3	2.776	normal	4

2 kelas obesitas

1 kelas normal

OBESITAS

kNN pada Data Nominal

Gunakan Hamming distance

index	food	chat	fast	price	bar	tip
0	great	yes	yes	normal	no	yes
1	great	no	yes	normal	no	yes
2	mediocre	yes	no	high	yes	no
3	mediocre	no	yes	normal	yes	no

Misal kita akan klasifikasi dengan kNN dengan $k = 3$

Contoh (great, no, no, normal, no)

index	food	chat	fast	price	bar	jarak	tip
0	0	1	1	0	0	2	yes
1	0	0	1	0	0	1	yes
2	1	1	0	1	1	4	no
3	1	0	1	0	1	3	no

2 yes

1 no

YES

IMPLEMENTASI

kNN di Scikit-learn

 [Install](#) [User Guide](#) [API](#) [Examples](#) [More](#)

[Prev](#) [Up](#) [Next](#)

scikit-learn 0.23.2
[Other versions](#)

Please [cite us](#) if you use the software.

[sklearn.neighbors.KNeighborsClassifier](#)
Examples using
`sklearn.neighbors.KNeighborsClassifier`

sklearn.neighbors.KNeighborsClassifier

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, *, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)
```

[\[source\]](#)


Classifier implementing the k-nearest neighbors vote.

Read more in the [User Guide](#).

Parameters:
n_neighbors : *int*, **default=5**
Number of neighbors to use by default for `kneighbors` queries.
weights : *{'uniform', 'distance'}* or *callable*, **default='uniform'**
weight function used in prediction. Possible values:

- 'uniform' : uniform weights. All points in each neighborhood are weighted equally.
- 'distance' : weight points by the inverse of their distance. in this case, closer neighbors of a query point will have a greater influence than neighbors which are further away.
- [callable] : a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

algorithm : *{'auto', 'ball_tree', 'kd_tree', 'brute'}*, **default='auto'**
Algorithm used to compute the nearest neighbors:
... ..

 [Install](#) [User Guide](#) [API](#) [Examples](#) [More](#)

[Prev](#) [Up](#) [Next](#)

scikit-learn 0.23.2
[Other versions](#)

Please [cite us](#) if you use the software.

[sklearn.preprocessing.StandardScaler](#)
Examples using
`sklearn.preprocessing.StandardScaler`

sklearn.preprocessing.StandardScaler

```
class sklearn.preprocessing.StandardScaler(*, copy=True, with_mean=True, with_std=True)
```

[\[source\]](#)

Standardize features by removing the mean and scaling to unit variance

The standard score of a sample \mathbf{x} is calculated as:

$$z = (\mathbf{x} - \mathbf{\bar{u}}) / s$$

where $\mathbf{\bar{u}}$ is the mean of the training samples or zero if `with_mean=False`, and s is the standard deviation of the training samples or one if `with_std=False`.

Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using [transform](#).

Standardization of a dataset is a common requirement for many machine learning estimators: they might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance).

For instance many elements used in the objective function of a learning algorithm (such as the RBF kernel of Support Vector Machines or the L1 and L2 regularizers of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

