K-Nearest Neighbor

인공지능 2021년 1학기
HW #1

박준

개요

- K-Nearest Neighbor 알고리즘을 이용하여 Iris classification 문제 해결하기
- K-Nearest Neighbor 알고리즘은 수업시간에 다룬 내용을 기반으로 (수업시간 내용으로 충분함)

Iris Data

- 4가지 특성으로 아이리스 꽃을 분류하는 예제
- Data size
  - 150
- # of features: 4
  - ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
  - * sepal: 꽃받침 / petal: 꽃잎
- # of class: 3
  - 0: Setosa, 1: Versicolor, 2: Virginica

Iris Data: example

{'data': array([[5.1, 3.5, 1.4, 0.2],

[4.9, 3. , 1.4, 0.2],

[4.7, 3.2, 1.3, 0.2],

[4.6, 3.1, 1.5, 0.2],

[5. , 3.6, 1.4, 0.2],

…….

[6.2, 3.4, 5.4, 2.3],

[5.9, 3. , 5.1, 1.8]]),

'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,

2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,

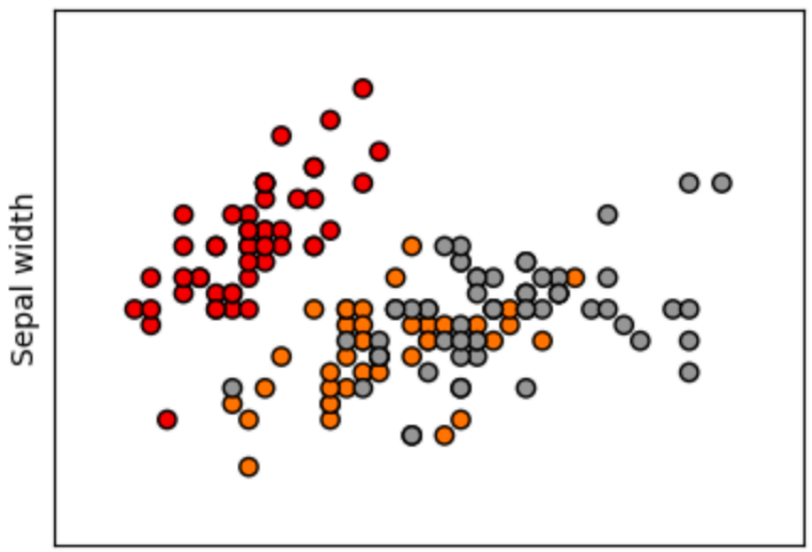
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]),

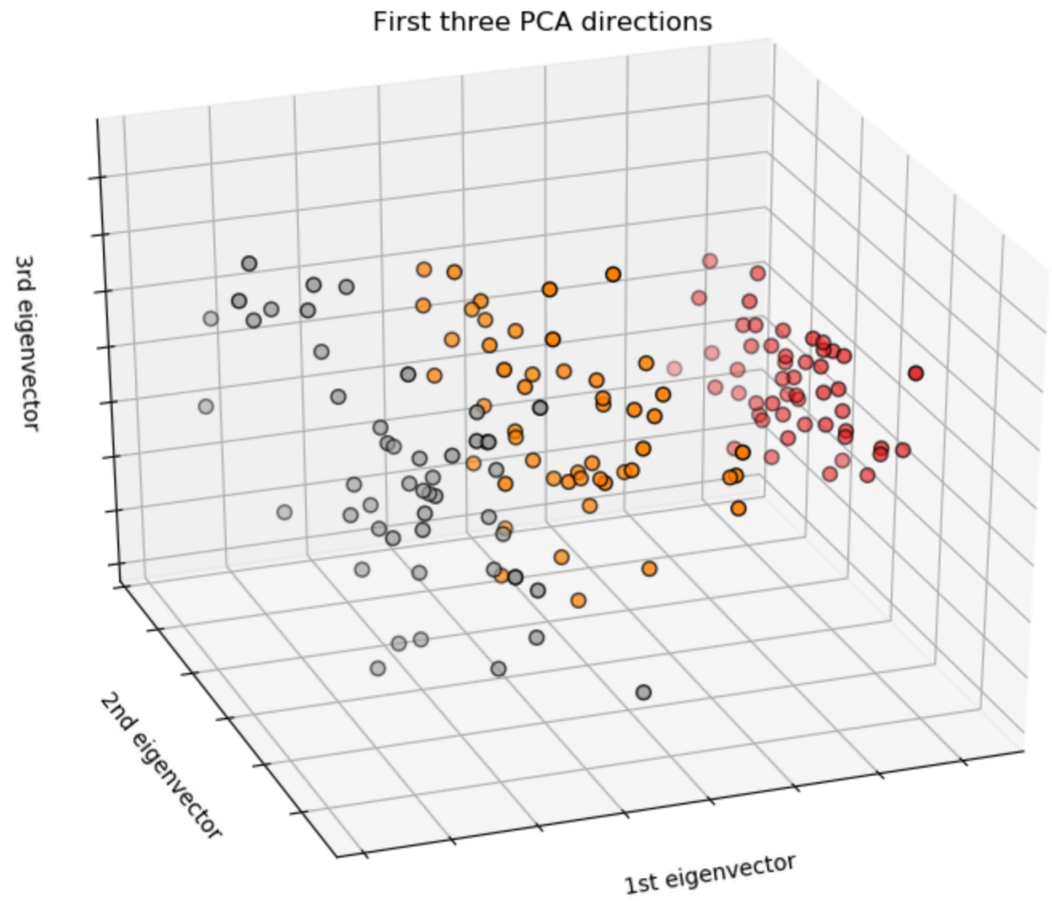
'target_names': array(['setosa', 'versicolor', 'virginica'],

……..}

Iris Data

- Example





Iris Data

- Packages
  - scikit-learn
  - numpy
  - matplotlib

Iris Data

- Data load
- Use scikit-learn

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris

iris = load_iris()
# print(iris)

X = iris.data          # iris data input
y = iris.target        # iris target (label)
y_name = iris.target_names  # iris target name
```

{'data': array([[5.1, 3.5, 1.4, 0.2],
[4.9, 3. , 1.4, 0.2],
[4.7, 3.2, 1.3, 0.2],
[4.6, 3.1, 1.5, 0.2],
[5. , 3.6, 1.4, 0.2], ...), 'target': array([0, 0, 0, 0, 0, 0, 0, 0, ...]), 'target_names': array(['setosa', ...]), ...

Iris Data

```
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris

iris = load_iris()
print(iris)

X = iris.data[:, :2]  # for now, use the first two features.
y = iris.target

x1_min, x1_max = X[:, 0].min() - .5, X[:, 0].max() + .5
x2_min, x2_max = X[:, 1].min() - .5, X[:, 1].max() + .5

plt.figure(figsize=(8, 6))
# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set1,
            edgecolor='k')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')

plt.xlim(x1_min, x1_max)
plt.ylim(x2_min, x2_max)


plt.show()
```

cmap str or `Colormap`

KNN

- Build a KNN class
  - Variables
    - K
    - Features: X
    - Target: y
    - ...
- Methods
  - Calculate distance
  - Obtain K-Nearest Neighbor
  - Obtain majority vote / weighted majority vote

KNN

- Train data
  - Use 14/15 for training
    - Every 1-st, ..., 13-th, 14-th data
      - data[0],..., data[13], data[15],...
- Test data
  - Use the rest for testing
    - Every 15-th data
    - data[14], data[29], ...

KNN

- Test
  - For every test example,
    - Calculate the output using K-Nearest Neighbor algorithm
  - Compare the calculated output and the true output
  - Use K=3, 5, 10

KNN

- Sample output

Test Data Index: 0 Computed class: setosa , True class: setosa

Test Data Index: 1 Computed class: setosa , True class: setosa

Test Data Index: 2 Computed class: setosa , True class: setosa

Test Data Index: 3 Computed class: versicolor , True class: versicolor

Test Data Index: 4 Computed class: versicolor , True class: versicolor

Test Data Index: 5 Computed class: versicolor , True class: versicolor

Test Data Index: 6 Computed class: virginica , True class: virginica

Test Data Index: 7 Computed class: virginica , True class: virginica

Test Data Index: 8 Computed class: virginica , True class: virginica

Test Data Index: 9 Computed class: virginica , True class: virginica

Submission

- Source code (with comments) files
  - KNN class python file
  - Main python file
- Output results (included in the Report)
- Report
  - Use the given template file

- Due
  - 4/12 (Monday) 11pm
  - Late: 20% per day