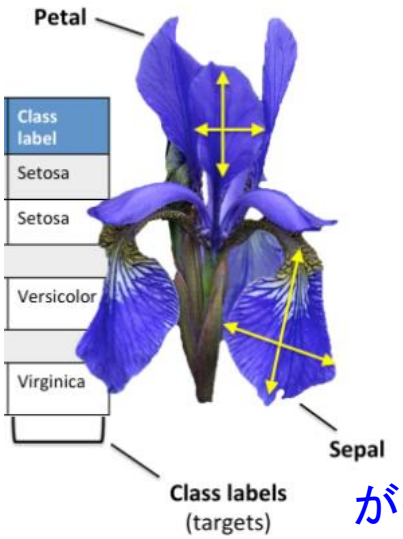
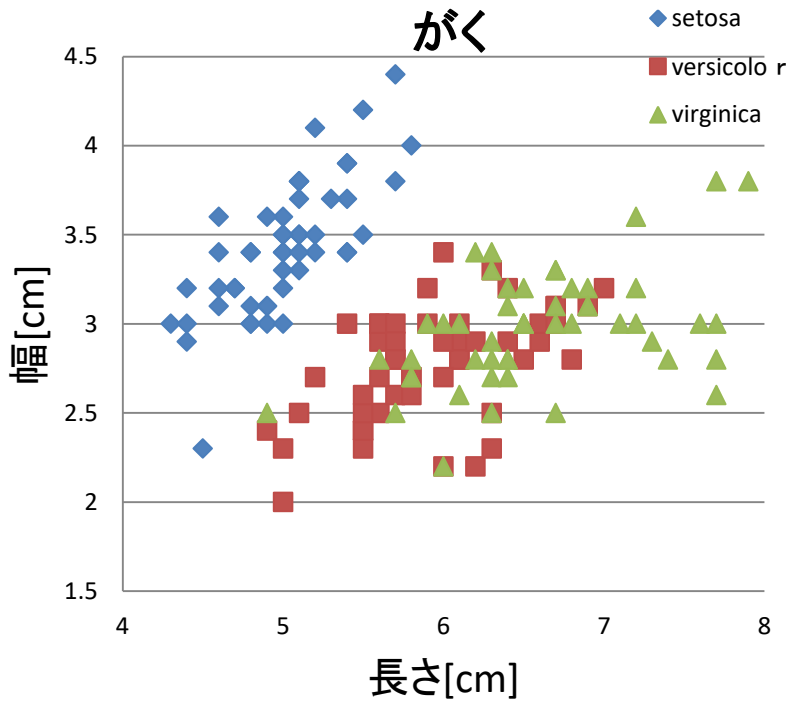
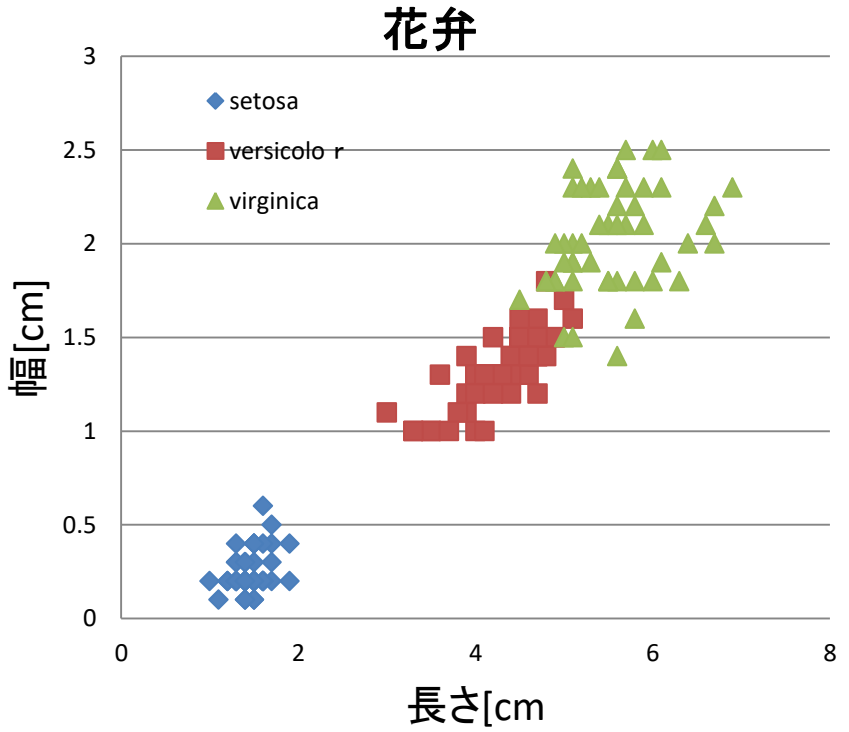
アイリスのデータを機械学習して、品種を予測する

花弁



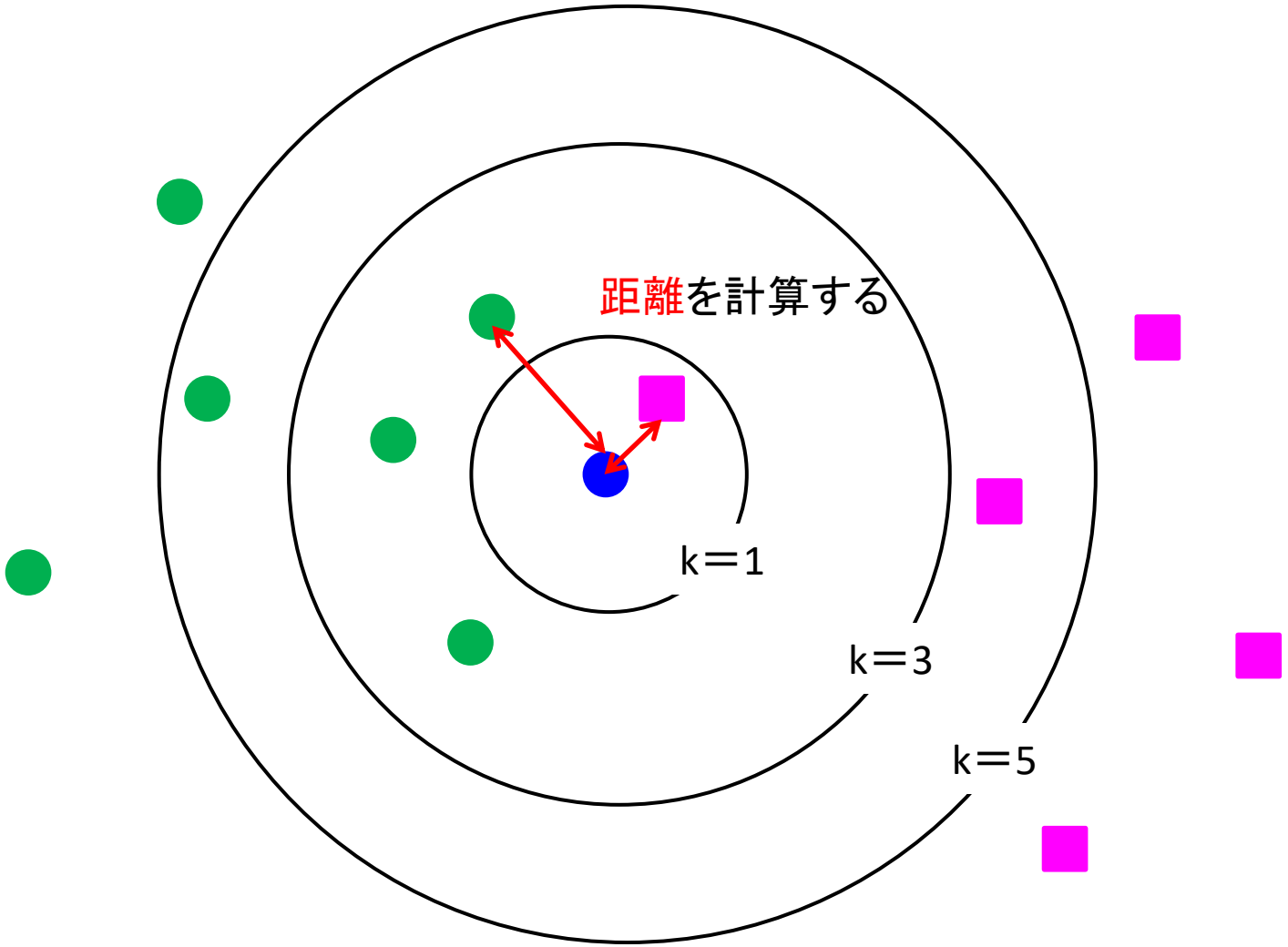
がく





kNN（k-nearest neighbor ）法

●が●の仲間か■の仲間を決めるアルゴリズムの1つ
k＝1では●の近傍に■が1個、k＝3の時は●が3個、■は1個
k＝3で分類分けすると決めると、多数決で●は●の仲間である判定する



①Jupyter を起動して、sklearnをインポートする。

```
import sklearn.datasets as datasets
iris = datasets.load_iris()
```

②irisのデータを読み込む

構造の確認

```
print(iris.feature_names)
print(iris.target_names)
```

③データの形式を表示させる

```
['sepal length*1 (cm)', 'sepal width*2 (cm)', 'petal length*3 (cm)', 'petal width*4 (cm)']
['setosa' 'versicolor' 'virginica']
```

*1 ガクの長さ, *2 ガクの幅, *3 花弁の長さ, *4 花弁の幅

表で表すとアイリスデータは以下のようになる

sepal length (ガクの長さ) [cm]	sepal width (ガクの幅) [cm]	petal length (花弁の長さ) [cm]	petal width (花弁の幅) [cm]	ラベル (setosa/versicolor/virginica)

In [1]:
```
import sklearn.datasets as datasets
iris = datasets.load_iris()
```

In [3]:
```
print(iris.feature_names)
print(iris.target_names)
```

```
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
['setosa' 'versicolor' 'virginica']
```

④データを表示

In [4]:
```
iris.data
```

Out[4]:
```
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],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
```

In [4]:
```
iris.target
```

Out[4]:
```
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, 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])
```

花のデータ　出力値

```
In [5]: #まず訓練データとテストデータを分ける
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.3, random_state=0)

#kNNのモデル作成
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=1)

#学習
knn.fit(X_train, y_train)

#テストデータの予測
y_pred = knn.predict(X_test)

#予測結果と正解の確認
print(y_pred)
print(y_test)

import numpy as np
print(np.mean(y_pred == y_test))
```

kNN

kの値を1

テストに使用する割合

⑤訓練データで学習した後、データの30%をテストして、予測と正解を比較する

X:　入力値（特徴量）
y:　目標値（アヤメの種類）

train:　訓練データ
test　:　テストデータ

Xは、ベクトル量なので大文字
yは、1種類なので小文字

予測値

```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 2 1 1 2 0 2 0 0]
```

正解

```
[2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0 2 1 0 2 2 1 0
 1 1 1 2 0 2 0 0]
```

正解率→ 0.9777777777777777

```
import numpy as np
print(np.mean(y_pred == y_test))
```

正解率の平均値