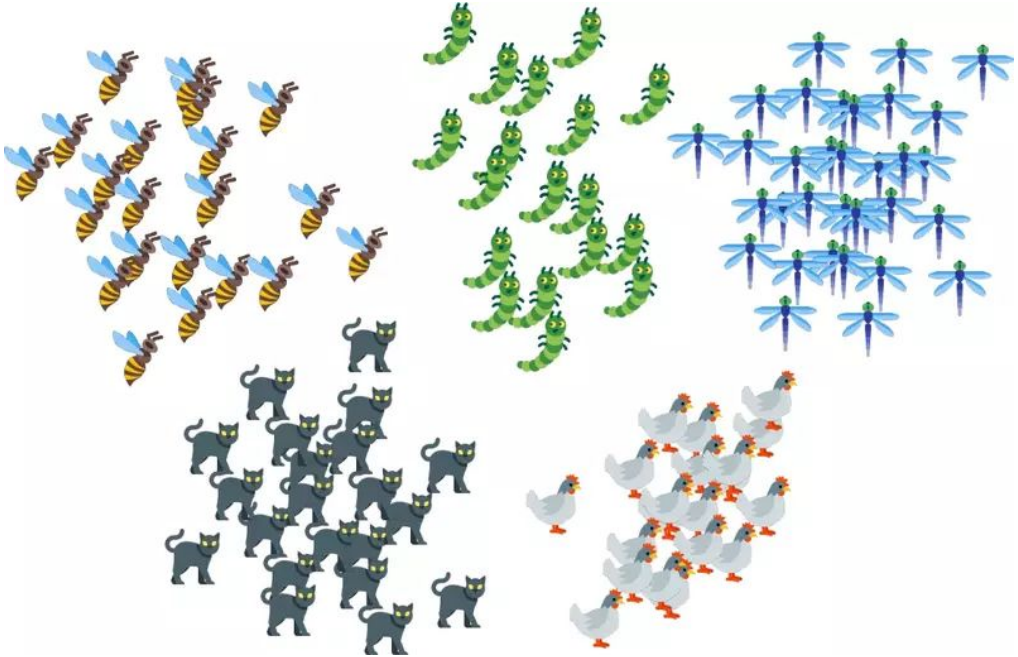
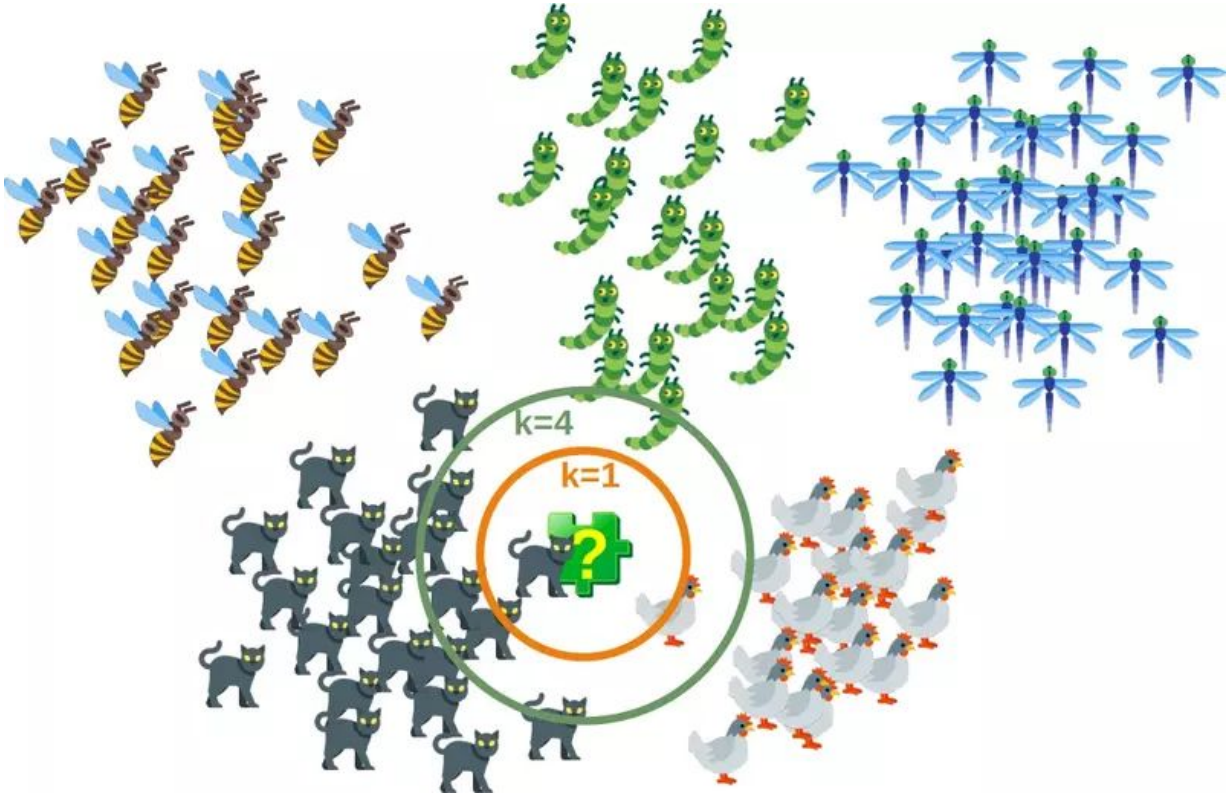
k-NN - k-Найближчі сусіди

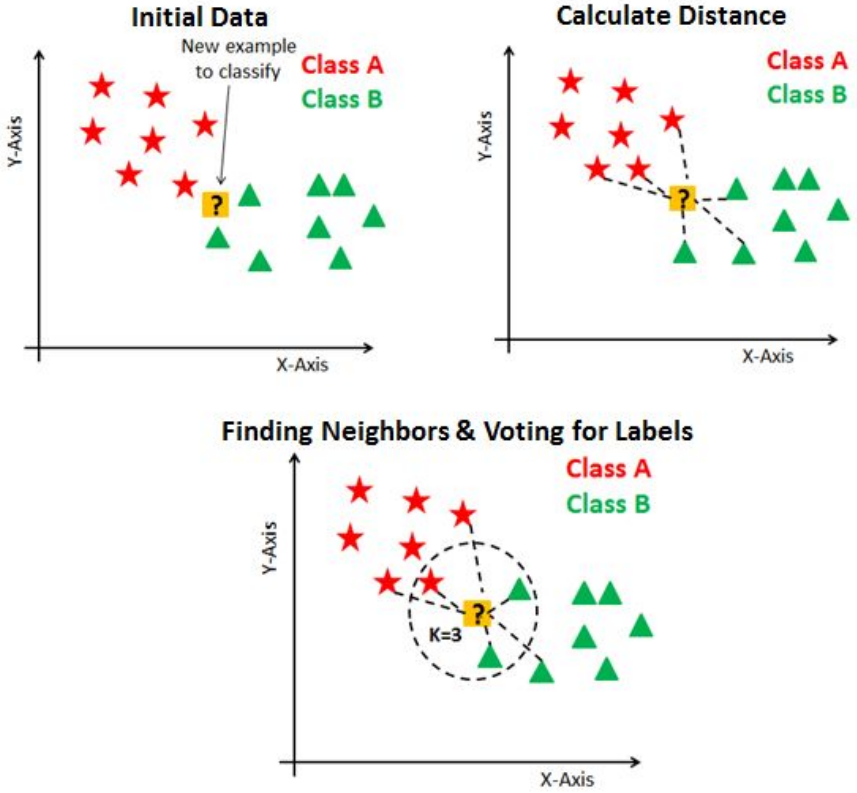
Ідея: схожі дані в своєму оточенні мають більшість даних того ж класу.




k=4
k=1
?

Робота алгоритму

1. Завантажуємо всі навчальні дані з мітками класів в пам'ять алгоритму.
2. Вибираємо кількість K сусідів для аналізу.
3. Для кожного прикладу з тестових даних обчислюємо відстань до точок навчального набору даних та знаходимо K найближчих. Клас, до якого належить більшість з них, буде класом нової точки.



k-NN: переваги та недоліки

Переваги KNN

- Просто втілити
- Гнучкість у виборі характеристик / метрики відстані
- Природно обробляє випадки багатокласової класифікації
- Може давати хороші результати за умови використання репрезентативних даних

Недоліки KNN

- Необхідно визначити значення параметра K (кількість найближчих сусідів)
- Високі обчислювальні витрати, оскільки нам потрібно обчислити відстань від кожного екземпляра запиту до всіх навчальних
- Зберігання даних
- Ми повинні бути впевнені, що вибрали відповідну метрику відстані

Приклад використання kNN

```
# Import necessary modules
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

# Loading data
irisData = load_iris()

# Create feature and target arrays
X = irisData.data
y = irisData.target

# Split into training and test set
X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size = 0.2, random_state=42)

knn = KNeighborsClassifier(n_neighbors=7)

knn.fit(X_train, y_train)

# Predict on dataset which model has not seen before
print(knn.predict(X_test))
```