Wong Man Yiu 1155160210

BMEG3105 Data analytics for personalized genomics and precision medicine

Lecture 8 Classification

Lecturer: Yu LI Friday, 29 September, 20223

Outline of lecture:

1.Classication

- Why classification?
- What is classification?
- How to do classification?

2.K-nearest neighbor classification

3. Clustering VS classification

Why classification?

- Characteristics of each class

- Classify items
  - Better organization

  - Where to put the new items?

- Classify people
  - Patients: different treatment for different groups (i.e., children, elder)
  - Customers: Is the person within the targeting group?

In biology:

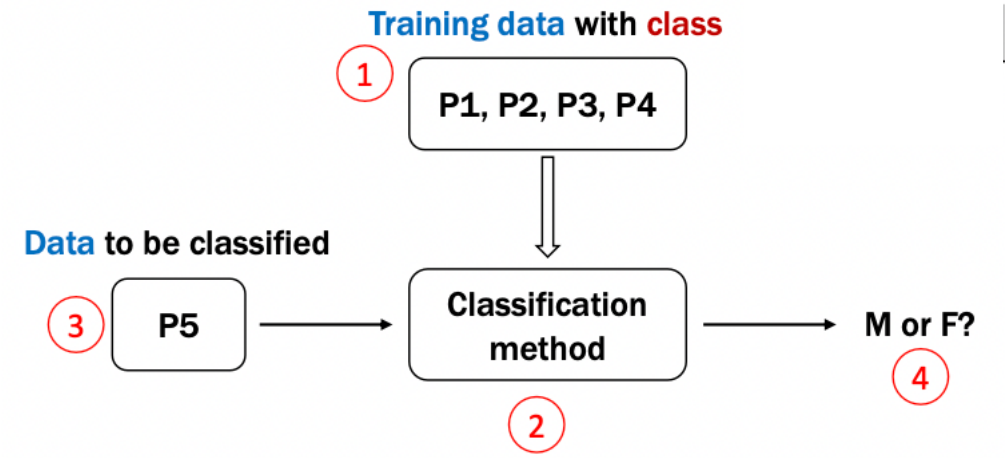
- Given a new gene expression profile, we can do prediction (i.e., Normal or Tumor)

Wong Man Yiu 1155160210

What is classification?

- Given a collection of records (training set)
  - Each record contains a set of attributes, one of the attributes is the class
- Find a method to assign the class of previously unseen records based on their other attributes and the training set as accurately as possible

How to do classification?

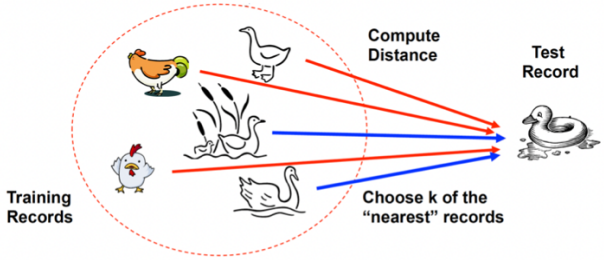


Elements needed to do classification

- Training data with class
- Classification method
- Data to be classified

K-nearest neighbor classification

- Basic idea: If it walks like a duck, quacks like a duck, then it's probably a duck



Wong Man Yiu 1155160210

- KNN is a simple algorithm that stores all available instances and classifies new instances based on a distance metric to the available ones

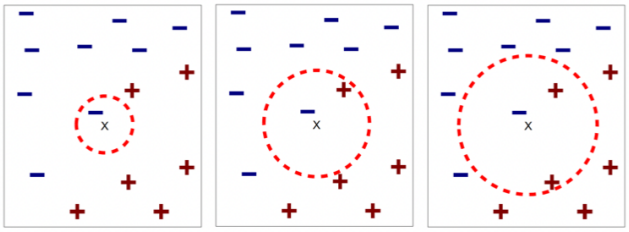
- Training process:
  - Store the available training instances

- Predicting process:
  - Find the K training instances that are closest to the query instance
  - Return the most frequent class label among those K instances
- Data should be normalized

Factors needed to determine when using KNN?

- A distance metric
  - Cosine similarity
  - Correlation
  - Euclidean distance
  - Manhattan distance
  - Mahalanobis distance

- How many neighbors to look at
  - K (different Ks can lead to different results)



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

- A weighing function (closer the distance, more the contribution) (optional)

How should we choose K?

- In practice, using a value of K somewhere between 5 and 10 gives good results for most low-dimensional data sets

- A good K can also be using cross-validation

Wong Man Yiu 1155160210

The standard procedure of KNN

- Supoose we have chosen the distance metric and K
  1. Normalization
  2. Compute distances
  3. Identify the K most similar data
  4. Take their class out and find the mode class

A running example of KNN

- Suppose we have chosen the **Euclidean distance** and **K=2**

1. Normalization

Person	Height(m)	Weight(kg)	Gender
P1	1.79	75	M
P2	1.64	54	F
P3	1.70	63	M
P4	1.88	78	M
P5	1.75	70	??

Person	Height	Weight	Gender
P1	0.625	0.875	M
P2	0	0	F
P3	0.25	0.375	M
P4	1	1	M
P5	0.4583	0.6667	??

2. Compute distance (Euclidean distance)

Person	P5	Gender
P1	0.267	M
P2	0.809	F
P3	0.358	M
P4	0.636	M
P5	0	??

3. Identify the K most similar data (K=2)

Person	P5	Gender
P1	0.267	M
P2	0.809	F
P3	0.358	M
P4	0.636	M
P5	0	??

4. Take their class out and find the mode class
   - M

Wong Man Yiu 1155160210

Clustering VS Classification

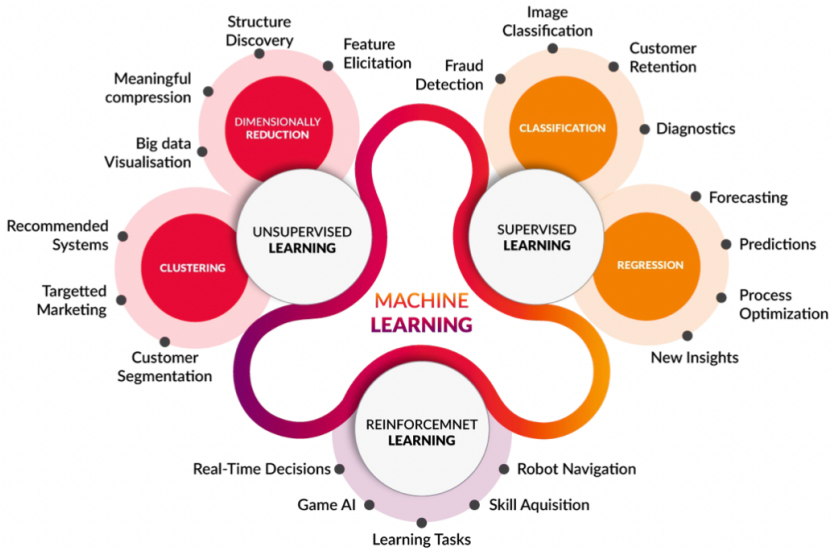
	Clustering	Classification
Goal	Find similarity (clusters) in the data	Assign class to the new data
Data	Data without class	Training data with class and testing data without class
Classes	Unknown number of classes	Known number of classes
Output	The cluster index for each point	The class assignment of the testing data
Algorithm	One phase	Two phases (training and application)

Unsupervised learning and supervised learning

- Unsupervised learning
  - Machine learning algorithms to analyze and cluster unlabeled data
  - Example: clustering and dimension reduction

- Supervised learning
  - Machine learning algorithms to classify and predict outcomes, trained on labelled data
  - Example: classification and regression

Machine learning



Wong Man Yiu 1155160210

KNN in Python

- Example:

```
>>> X = [[0], [1], [2], [3]]
>>> y = [0, 0, 1, 1]
>>> from sklearn.neighbors import KNeighborsClassifier
>>> neigh = KNeighborsClassifier(n_neighbors=3)
>>> neigh.fit(X, y)
KNeighborsClassifier(...)
>>> print(neigh.predict([[1.1]]))
[0]
>>> print(neigh.predict_proba([[0.9]]))
[[0.666... 0.333...]]
```