Data Mining & Knowledge Discovery

Classification - 04

Md. Biplob Hosen

Lecturer, IIT-JU

Email: biplob.hosen@juniv.edu

Instance-Based Learning

- The classification framework shown in decision tree and rule-based classifiers involves a two-step process: (1) an inductive step for constructing a classification model from data, and (2) a deductive step for applying the model to test examples.
- These are examples of **eager learners** because they are designed to learn a model that maps the input attributes to the class label as soon as the training data becomes available.
- An opposite strategy would be to delay the process of modeling the training data until it is needed to classify the test instances. Techniques that employ this strategy are known as **lazy learners**. An example of a lazy learner is the **Rote classifier**, which memorizes the entire training data and performs classification only if the attributes of a test instance match one of the training examples exactly.
- An obvious drawback of this approach is that some test instances may not be classified because they do not match any training example.

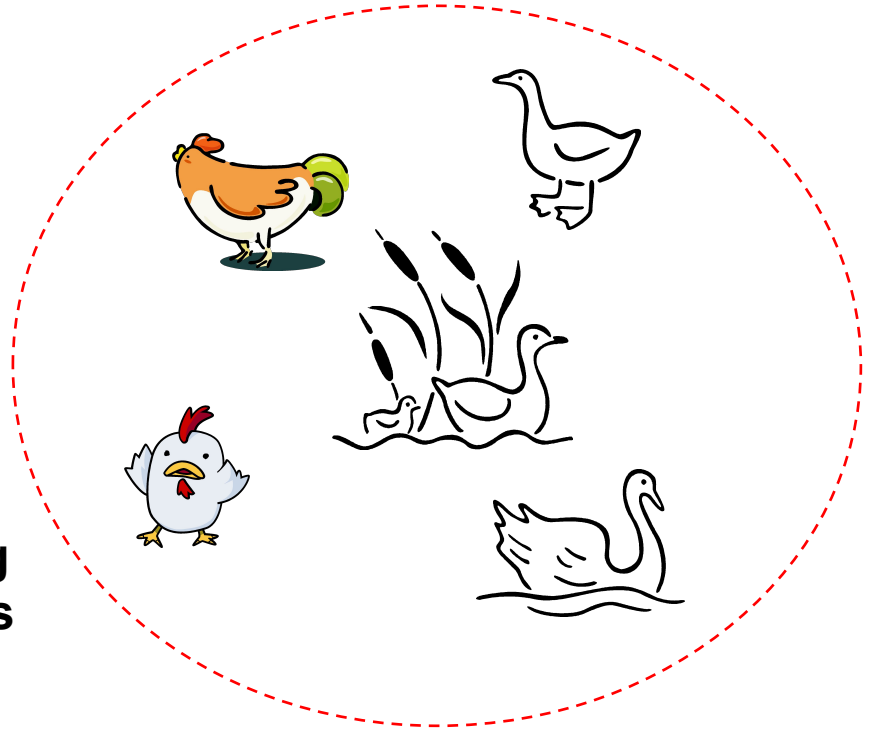
Nearest Neighbor Classifiers

- One way to make this approach more flexible is to find all the training examples that are relatively similar to the attributes of the test instances.
- These examples, which are known as nearest neighbors, can be used to determine the class label of the test instance.
- A nearest neighbor classifier represents each example as a data point in a d-dimensional space, where d is the number of attributes.
- Given a test instance, we compute its proximity to the training instances according to one of the proximity measures.
- The k-nearest neighbors of a given test instance z refer to the k training examples that are closest to z.

Continue…

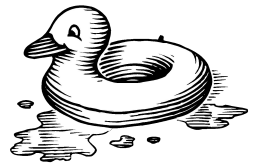
Basic idea:

- If it walks like a duck, quacks like a duck, then it's probably a duck.



Training Records

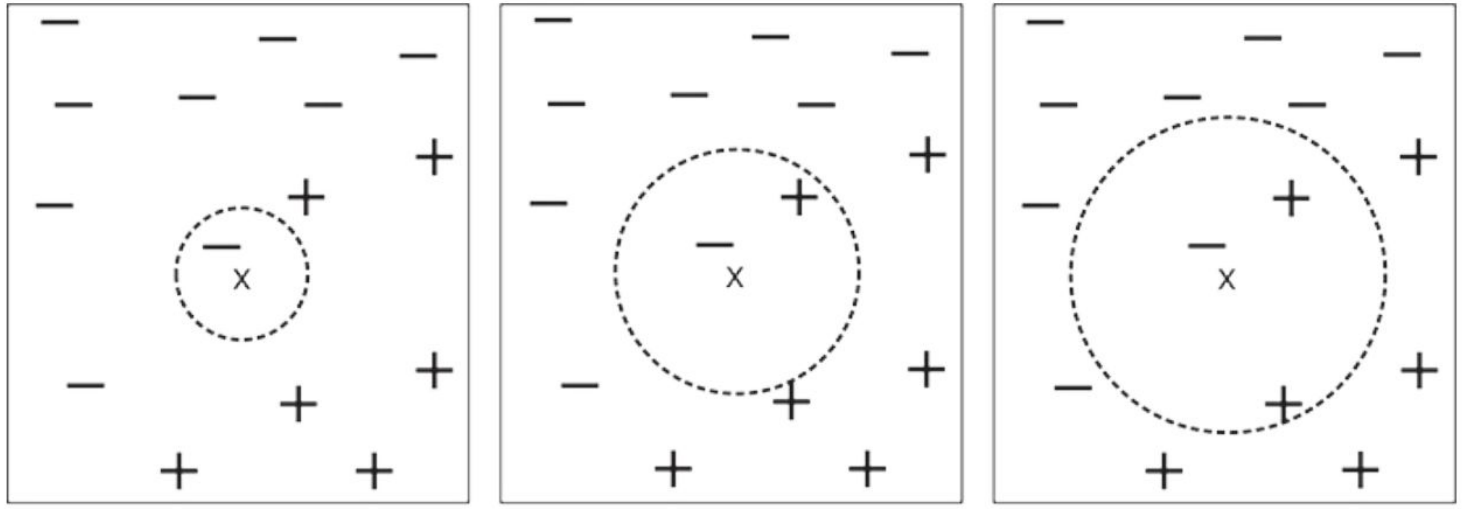
Test Record



Continue…

- Following figure illustrates the 1-, 2-, and 3-nearest neighbors of a test instance located at the center of each circle. The instance is classified based on the class labels of its neighbors.
- In the case where the neighbors have more than one label, the test instance is assigned to the majority class of its nearest neighbors. In figure (a) , the 1-nearest neighbor of the instance is a negative example. Therefore the instance is assigned to the negative class.
- If the number of nearest neighbors is three, as shown in figure (c) , then the neighborhood contains two positive examples and one negative example. Using the majority voting scheme, the instance is assigned to the positive class.
- In the case where there is a tie between the classes (Figure (b)), we may randomly choose one of them to classify the data point.

Continue…



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

- The preceding discussion underscores the importance of choosing the right value for k. If k is too small, then the nearest neighbor classifier may be susceptible to overfitting due to noise, i.e., mislabeled examples in the training data. On the other hand, if k is too large, the nearest neighbor classifier may misclassify the test instance because its list of nearest neighbors includes training examples that are located far away from its neighborhood

Algorithm

- The algorithm computes the distance (or similarity) between each test instance and all the training examples to determine its nearest neighbor list, z=(x′, y′).

1. Load the data
2. Initialise the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points
    i. Calculate the distance between test data and each row of training dataset. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other distance function or metrics that can be used are Manhattan distance, Minkowski distance, Chebyshev, cosine, etc. If there are categorical variables, hamming distance can be used.
    ii. Sort the calculated distances in ascending order based on distance values
    iii. Get top k rows from the sorted array
    iv. Get the most frequent class of these rows
    v. Return the predicted class

Example

- Suppose we have height, weight and T-shirt size of some customers and we need to predict the T-shirt size of a new customer given only height and weight information we have. Data including height, weight and T-shirt size information is shown below -

Height (in cms)	Weight (in kgs)	T Shirt Size
158	58	M
158	59	M
158	63	M
160	59	M
160	60	M
163	60	M
163	61	M
160	64	L
163	64	L
165	61	L

Height (in cms)	Weight (in kgs)	T Shirt Size
165	62	L
165	65	L
168	62	L
168	63	L
168	66	L
170	63	L
170	64	L
170	68	L

Continue…

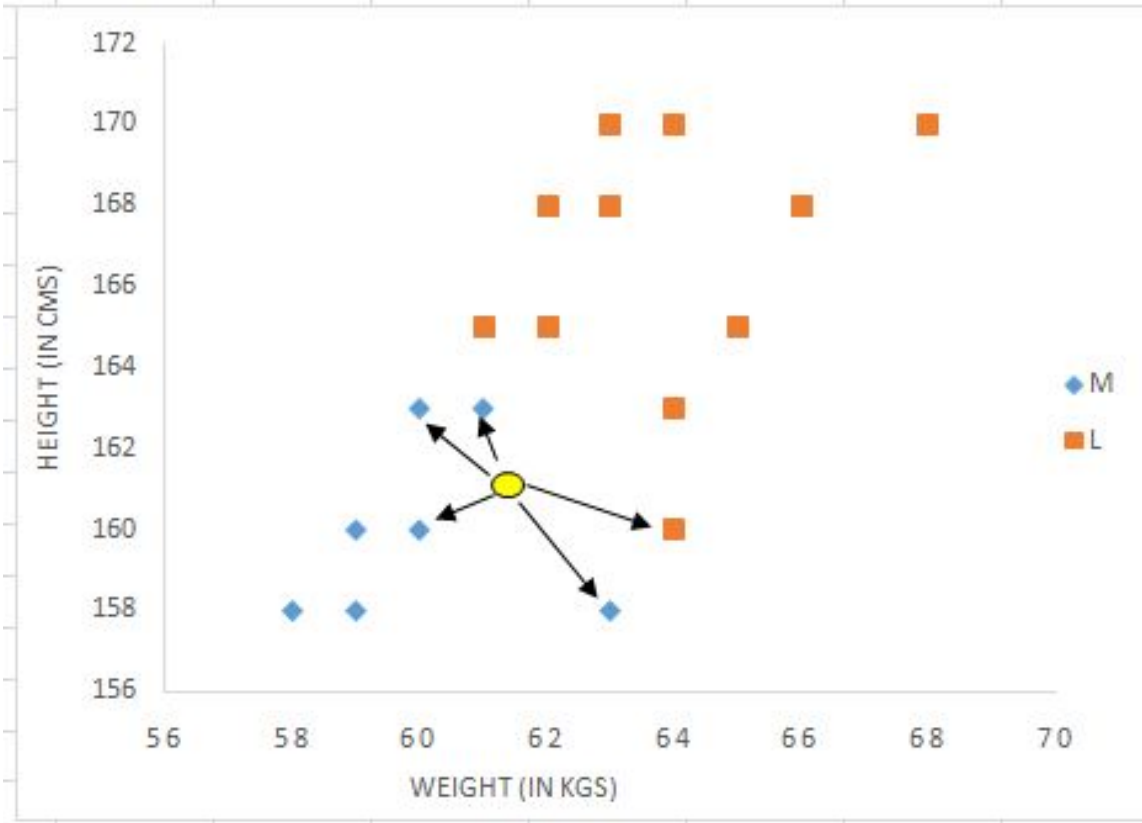
- Step 1 - **Calculate Similarity based on distance function:** There are many distance functions but Euclidean is the most commonly used measure.
- New customer named 'Monica' has height 161cm and weight 61kg.
- Euclidean distance between first observation and new observation (monica) is as follows: =SQRT((161-158)^2+(61-58)^2)
- Similarly, we will calculate distance of all the training cases with new case and calculates the rank in terms of distance. The smallest distance value will be ranked 1 and considered as nearest neighbor.
- Step 2 : **Find K-Nearest Neighbors:** Let k be 5. Then the algorithm searches for the 5 customers closest to Monica, i.e. most similar to Monica in terms of attributes, and see what categories those 5 customers were in. If 4 of them had ‘Medium T shirt sizes’ and 1 had 'Large T shirt size' then your best guess for Monica is ‘Medium T shirt.

Continue…

- See the calculation shown in the snapshot below -

=SQRT((A21-A6)^2+(B21-B6)^2)

	A	B	C	D	E
1	Height (in cms)	Weight (in kgs)	T Shirt Size	Distance	
2	158	58	M	4.2	
3	158	59	M	3.6	
4	158	63	M	3.6	
5	160	59	M	2.2	3
6	160	60	M	1.4	1
7	163	60	M	2.2	3
8	163	61	M	2.0	2
9	160	64	L	3.2	5
10	163	64	L	3.6	
11	165	61	L	4.0	
12	165	62	L	4.1	
13	165	65	L	5.7	
14	168	62	L	7.1	
15	168	63	L	7.3	
16	168	66	L	8.6	
17	170	63	L	9.2	
18	170	64	L	9.5	
19	170	68	L	11.4	
20					
21	161	61			



Naive Bayes Classifier

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naive Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Bayes' Theorem

- Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred.
- Bayes' theorem is stated mathematically as: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$
- Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as **evidence**.
- P(A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
- P(A|B) is a probability of hypothesis A on the observed event B.
- P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

Working of Naive Bayes' Classifier

- Suppose we have a dataset of weather conditions and corresponding target variable "Play". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.
- So to solve this problem, we need to follow the below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.

- **Problem:** If the weather is sunny, then the Player should play or not?
- Solution: To solve this, first consider the following dataset:

Continue…

- Frequency table:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	4

Likelihood table:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

- **P(Yes|Sunny)**= P(Sunny|Yes)*P(Yes)/P(Sunny)
- P(Sunny|Yes)= 3/10= 0.3; P(Sunny)= 0.35; P(Yes)=0.71
- So, P(Yes|Sunny) = 0.3*0.71/0.35= 0.60

- **P(No|Sunny)**= P(Sunny|No)*P(No)/P(Sunny)
- P(Sunny|NO)= 2/4=0.5; P(No)= 0.29; P(Sunny)= 0.35
- So, P(No|Sunny)= 0.5*0.29/0.35 = 0.41
- Here, P(Yes|Sunny)>P(No|Sunny). **Hence on a Sunny day, Player can play the game.**

Day	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Example

Day	Outlook	Temperature	Humidity	Windy	Play Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

Solution

Outlook

	Yes	No	P(yes)	P(no)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rainy	3	2	3/9	2/5
Total	9	5	100%	100%

Temperature

	Yes	No	P(yes)	P(no)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	100%	100%

Humidity

	Yes	No	P(yes)	P(no)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	100%	100%

Wind

	Yes	No	P(yes)	P(no)
False	6	2	6/9	2/5
True	3	3	3/9	3/5
Total	9	5	100%	100%

Play		**P(Yes)/P(No)**
Yes	9	9/14
No	5	5/14
Total	14	100%

Continue…

- Let us test it on a new set of features (let us call it today):
- today = (Sunny, Hot, Normal, False)
- So, probability of playing golf is given by:

$$P(Yes|today) = \frac{P(SunnyOutlook|Yes)P(HotTemperature|Yes)P(NormalHumidity|Yes)P(NoWind|Yes)P(Yes)}{P(today)}$$

- Probability to not play golf is given by:

$$P(No|today) = \frac{P(SunnyOutlook|No)P(HotTemperature|No)P(NormalHumidity|No)P(NoWind|No)P(No)}{P(today)}$$

- Since, P(today) is common in both probabilities, we can ignore P(today) and find proportional probabilities as:

$$P(Yes|today) \propto \frac{2}{9}.\frac{2}{9}.\frac{6}{9}.\frac{6}{9}.\frac{9}{14} \approx 0.0141 \qquad P(No|today) \propto \frac{3}{5}.\frac{2}{5}.\frac{1}{5}.\frac{2}{5}.\frac{5}{14} \approx 0.0068$$

- **So, prediction that golf would be played is ‘Yes’.**

Exercise

- Chapter-4 [6, 7, 8]

Thank You!