CS 4100: Foundations of Artificial Intelligence (Spring 2020) Roi Yehoshua
Student name: Zhiyuan Cao (Due) March 26, 2020

Machine Learning: Classification

1 KNN

Consider the following data set comprised of three numerical features (f_1, f_2, and f_3) and one binary output:

Example	f_1	f_2	f_3	y
$\mathbf{x}_1$	1	4	1	1
$\mathbf{x}_2$	1	2	3	1
$\mathbf{x}_3$	0	0	1	1
$\mathbf{x}_4$	-1	4	0	1
$\mathbf{x}_5$	1	0	-2	0
$\mathbf{x}_6$	-1	-1	1	0
$\mathbf{x}_7$	0	-4	0	0
$\mathbf{x}_8$	1	0	-3	0

Based on this training set, classify a new vector $\mathbf{x} = (1, 0, 1)$ using KNN with $k = 3$ and Manhattan distance. Show your work.

$d(x_1, x) = |0+4+0| = 4$
$d(x_2, x) = |6-2| = 4$
$d(x_3, x) = |0-1| = |-1| = 1$ ✓ $y=1$
$d(x_4, x) = |-2+4-1| = 1$

$d(x_5, x) = |-1-2| = 3$
$d(x_6, x) = |-1-2| = 3$
$d(x_7, x) = |-4-2| = 6$
$d(x_8, x) = |-2-2| = 4$

2 Naive Bayes

The following table shows a set of random samples from a customer database:

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Based on this training set, use Naive Bayes classification to classify the following sample: $\mathbf{x}$ = (age = youth, income = medium, student = yes, credit_rating = fair)

Don't use Laplace smoothing. Show your work.

b= buy   nb= no buy

$P(b) = 9/14 = 0.643$ $P(nb) = 1 - 9/14 = 0.357$

$P(age=youth \mid b) = 0.222$ $P(age=youth \mid nb) = 3/5 = 0.6$

$P(income=m \mid b) = 0.444$ $P(income=m \mid nb) = 0.4$

$P(student=Y \mid b) = 0.667$ $P(student=Y \mid nb) = 0.2$

$P(credit=fair \mid b) = 0.667$ $P(credit=fair \mid nb) = 0.4$

$P(X \mid buy) = 0.222 \cdot 0.444 \cdot 0.667 \cdot 0.667 = 0.044$

$P(X \mid no\ buy) = 0.6 \cdot 0.6 \cdot 0.2 \cdot 0.4 = 0.019$

$P(X \mid buy) \cdot P(b) = 0.028 > P(X \mid nb) \cdot P(nb) = 0.019 \cdot 0.357 = 0.007$

Yes !!

3 Decision Trees

Consider the following data set comprised of three binary input attributes (A_1, A_2, and A_3) and one binary output:

Example	A_1	A_2	A_3	Output y
$\mathbf{x}_1$	1	0	0	0
$\mathbf{x}_2$	1	0	1	0
$\mathbf{x}_3$	0	1	0	0
$\mathbf{x}_4$	1	1	1	1
$\mathbf{x}_5$	1	1	0	1

Use the algorithm in Figure 18.5 to learn a decision tree for these data. Show the computations made to determine the attribute to split at each node.

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** a tree

  if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
  else
    $A \leftarrow \operatorname{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
    tree ← a new decision tree with root test A
    for each value v_k of A **do**
      $exs \leftarrow \{e : e \in examples$ **and** $e.A = v_k\}$
      subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − A, *examples*)
      add a branch to *tree* with label $(A = v_k)$ and subtree *subtree*
    return *tree*

Figure 18.5 The decision-tree learning algorithm. The function IMPORTANCE is described in Section 18.3.4. The function PLURALITY-VALUE selects the most common output value among a set of examples, breaking ties randomly.

Remaindar (A): $\sum_{k=1}^{d} \frac{p_k+n_k}{p+n} B\left(\frac{p_k}{p_k+n_k}\right)$

① A2

A_1: $(\frac{4}{5})(-\frac{2}{4}\lg(\frac{2}{4}) - \frac{2}{4}\lg(\frac{2}{4})) + (\frac{1}{5})(-0 - \frac{1}{1}\lg(\frac{1}{1})) = 0.8$

$A_2 = 0.551$ ✓

$A_3 = 0.951$

② A1

A_1: $\frac{2}{3} * H\langle 1,0\rangle + \frac{1}{3} H\langle 0,1\rangle = 0$

$A_3 = 0.66$

$\boxed{A2 \rightarrow A1}$

split split

4 Classification Metrics

Suppose that you are working on a spam detection system. You formulated the problem as a classification task where "Spam" is the positive class and "not Spam" is the negative class. Your training set contains $n = 1000$ emails, 99% of these are non-Spam and 1% are spam.

1. What is the accuracy of a classifier that always predicts "not Spam"?

2. Suppose you trained a classifier on this training set, and you've got the following confusion matrix:

		Predicted class	
		Spam	not Spam
Actual class	Spam	8	2
	not Spam	16	974

What are the accuracy, precision and recall of the classifier?

① $\frac{990}{1000} = 99\%$ is the accuracy of a classifier

② precision: $\frac{8}{16+8} = \frac{1}{3}$

recall: $\frac{8}{8+2} = \frac{4}{5}$