CS 440: Introduction to AI

Homework 5 Solution

Due: November 30 11:59PM, 2010

Your answers must be concise and clear. Explain sufficiently that we can easily determine what you understand. We will give more points for a brief interesting discussion with no answer than for a bluffing answer.

Please email your solution to the TA at cs440ta@cs.illinois.edu.

1 Naive Bayes

MPG	Horsepower	Weight	Acceleration	Displacement
High	LT100	Light	Wow	Small
Low	GT100	Heavy	Poor	Large
Moderate	LT100	Light	Wimpy	Medium
Moderate	LT100	Light	Wimpy	Medium
High	LT100	Light	Wimpy	Small
Low	GT100	Heavy	Wimpy	Medium
Moderate	LT100	Heavy	Wimpy	Medium
Low	GT100	Heavy	Wimpy	Large
Moderate	LT100	Heavy	Poor	Medium
Moderate	GT100	Heavy	Wow	Large
Low	GT100	Heavy	Poor	Large
Moderate	GT100	Light	Wimpy	Medium
Low	GT100	Heavy	Wimpy	Large
Low	GT100	Heavy	Wow	Medium
High	LT100	Light	Wimpy	Small
Moderate	GT100	Heavy	Poor	Medium

Table 1: A data set. LT100: Less than 100 HP. GT100: Greater than 100 HP.

Table 1 shows an automobile dataset. The data consists of five features (MPG, Horsepower, Weight, Acceleration, Displacement). Each is binary or trinary. Assume for your Nave Bayes net that *MPG, Horsepower, Weight, Acceleration* are conditionally independent from each other given *Displacement*. We will learn a Nave Bayes net from the dataset in Table 1 and use it to estimate missing values of new examples

In naive Bayes net, attributes (X) are conditionally independent of each other given a class variable (Y). Y can be infered by the following Naive Bayes classification rule:

$$Y \leftarrow \underset{y_k}{\operatorname{argmax}} P(Y = y_k) \prod_i P(X_i \mid Y = y_k) \tag{1}$$

In order to estimate parameters, we use Maximum Likelihood Estimation (MLE) with *add-one smoothing* as follows:

$$\hat{P}(Y = y_k) = \frac{\#D\{Y = y_k\} + 1}{\mid D \mid + J} \tag{2}$$

$$\hat{P}(X_i = x_{ij} \mid Y = y_k) = \frac{\#D\{X_i = x_{ij} \wedge Y = y_k\} + 1}{\#D\{Y = y_k\} + K} \tag{3}$$

where the $\#D\{x\}$ represents the number of elements in the data set D that satisfy property x, J is the number of distinct values Y can take on, and K is the number of distinct values X_i can take on. In addition to our text, you may find it useful to visit `http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf`.

1. From all the examples in the table 1, estimate parameters.

   (a) [5/20 pts.] What is the estimated probability (2) for each y_k?
   (Solution) $\hat{P}(Y = Small) = (3 + 1)/(16 + 3) = 0.2105$
   $\hat{P}(Y = Medium) = (8 + 1)/(16 + 3) = 0.4737$
   $\hat{P}(Y = Large) = (5 + 1)/(16 + 3) = 0.3158$

   (b) [5/20 pts.] What is the estimated probability (3) for the attribute *Weight* and each y_k? (The number of parameters you need to estimate is 6.)
   (Solution) $\hat{P}(Weight = Light \mid Y = Small) = (3 + 1)/(3 + 2) = 0.8$
   $\hat{P}(Weight = Heavy \mid Y = Small) = 1 - \hat{P}(Weight = Light \mid Y = Small) = 0.2$
   The other probabilities can be estimated similarly.
   $\hat{P}(Weight = Light \mid Y = Medium) = 0.4$
   $\hat{P}(Weight = Heavy \mid Y = Medium) = 0.6$
   $\hat{P}(Weight = Light \mid Y = Large) = 0.1429$
   $\hat{P}(Weight = Heavy \mid Y = Large) = 0.8571$

   (c) [5/20 pts.] Explain in your own words why it is useful to add one to the numerator counts in equation (2) and (3).
   (Solution) The numerators of the equation (2) and (3) can be zero if we do not add one, and it will result in the estimated value of zero. Even though the estimated probability from the sample is zero, it's likely to have a small probability rather than zero in the real data. This zero probability will make the whole multiplication part of the expression (1) zero even if other probabilities are high, resulting in

poor classification. To avoid this, it is common to use a "smoothed" estimate which effectively adds in a number of additional "hallucinated" examples, and which assumes these hallucinated examples are spread evenly over the possible values of X_i or Y.

2. [5/20 pts.] We can classify new examples by Naive Bayes classification (1) with the estimated probabilities. We have a new example (MPG:High, $Horsepower$:LT100, $Weight$:Light, $Acceleration$:Poor). To make the problem simple, we provide a fake conditional probability table of $P(X_{ij} \mid Y_k)$ for the values in the example.

Displacement	MPG:High	Horsepower:LT100	Weight:Light	Acceleration:Poor
Small	0.5	0.6	0.5	0.1
Medium	0.2	0.5	0.4	0.3
Large	0.1	0.3	0.2	0.4

Table 2: Fake conditional probability table

Classify the new example by Naive Bayes classification (1) with the probabilities obtained from 1-(a) and the fake probabilities in table 2. Show your work.

(Solution) $P(Y = Small) \prod_i P(X_i \mid Y = Small)$
$= P(Y = Small) \cdot P(MPG = High \mid Y = Small) \cdot P(Horsepower = LT100 \mid Y = Small) \cdot P(Weight = Light \mid Y = Small) \cdot P(Acceleration = Poor \mid Y = Small)$
$= 0.2105 \cdot 0.5 \cdot 0.6 \cdot 0.5 \cdot 0.1 = 0.0031575$

Other probabilities can be obtained similarly.

$P(Y = Medium) \prod_i P(X_i \mid Y = Medium) = 0.4737 \cdot 0.2 \cdot 0.5 \cdot 0.4 \cdot 0.3 = 0.005684$

$P(Y = Large) \prod_i P(X_i \mid Y = Large) = 0.3158 \cdot 0.1 \cdot 0.3 \cdot 0.2 \cdot 0.4 = 0.00075792$

Therefore, the predicted class of the given example is $\text{argmax}_{y_k}[P(Y = y_k) \prod_i P(X_i \mid Y = y_k)]$ = Medium.

2 Decision Tree

1. [30 pts.] For this question, you will manually induce a decision tree from the training data, which are the first twelve examples in table 1. Then, we will predict *MPG* with the other four attributes in the test data (remaining four examples). *Use the order of the attributes in the table to break the tie.* You can report the decision tree as a series of `if-then` statements, where each indentation is a node level in the tree, as follows:

```
if attribute 1 = x
    if attribute 2 = y
        if attribute 3 = z
            class = +
        if attribute 3 = q
            class = 0
        if attribute 3 = r
            class = -
    if attribute 2 = s
        class = +
if attribute 1 = t
    if attribute 3 = z
        class = 0
    if attribute 3 = r
        class = -
```

   (a) [5/30 pts.] What is the entropy of the training data (first twelve examples) over the class variable (*MPG*)? Show your work.

   (Solution)
   $H(S) = \sum_{v \in Labels} -P(v) \cdot log_2(P(v))$
   $= -P(S_{Low}) \cdot log_2(P(S_{Low})) - P(S_{Moderate}) \cdot log_2(P(S_{Moderate})) - P(S_{High}) \cdot log_2(P(S_{High}))$
   $= 0.3333 \cdot 1.5850 + 0.5 \cdot 1 + 0.1667 \cdot 2.5850 = 1.4592$

   (b) [15/30 pts.] Use the training data to generate a decision tree. Express your decision tree using a series of `if-then` statements.

   (Solution) For each node of the tree, you need to choose which attribute to use to split the data based on Information Gain. If the node has only one kind of class, then stop splitting the data and assign the class to the node. The Information Gain can be computed as follows.
   $Gain(S, Attribute) = H(S_b) - \sum_i H(S_{ai}) \frac{|S_{ai}|}{|S_b|}$

For the root node, you need to choose one attribute that would have the highest information gain by splitting the whole training data.

$Gain(S, Horsepower) = 1.4592 - (6/12 \cdot 0.9183 + 6/12 \cdot 0.9183) = 0.5409$

$Gain(S, Weight) = 1.4592 - (5/12 \cdot 0.9710 + 7/12 \cdot 0.9852) = 0.4799$

$Gain(S, Acceleration) = 1.4592 - (2/12 \cdot 1 + 3/12 \cdot 0.9183 + 7/12 \cdot 1.3788) = 0.2587$

$Gain(S, Displacement) = 1.4592 - (2/12 \cdot 0 + 6/12 \cdot 0.6500 + 4/12 \cdot 0.8113) = 0.8638$

We obtain the highest information gain when we split by Displacement, so we choose it for the root node to split the sample. The splitted samples will be used to compute information gain of child nodes and other attributes and split them.

For example, we now have two examples whose *Displacement* is Small. Since all the two examples have the class High, we assign that class to the node. Also, we have four examples whose *Displacement* is Large. Among them, we have three Low-labeled examples and one Moderate-labeled example. For these four examples, we obtain the highest information gain splitting by *Acceleration*, so we again split those four examples by *Acceleration*. The other nodes can be splitted in a similar way.

The following tree is the resulting decision tree.

```
if attribute Displacement = Small
    class = High
if attribute Displacement = Medium
    if Horsepower = LT100
        class = Moderate
    if Horsepower = GT100
        if Weight = Light
            class = Moderate
        if Weight = Heavy
            class = Low
if attribute Displacement = Large
    if Acceleration = Poor
        class = Low
    if Acceleration = Wimpy
        class = Low
    if Acceleration = Wow
        class = Moderate
```

(c) [10/30 pts.] How well do the tree perform on the test data (remaining four examples)? Report the classification results and the error rate (that is, the ratio of the errors to the number of examples).

(Solution) Applying the decision tree to the test examples, the predicted classes of them are Low, Low, High, and Low, respectively, while the actual classes of them are Low, Low, High, and Moderate, respectively. Therefore, the error rate is $1/4 = 0.25$.

3 Evaluation

1. [10 pts.] Suppose you are running a learning experiment on a new algorithm for Boolean classification. You have a data set consisting of 15 positive and 15 negative examples. You plan to use leave-one-out cross validation and compare your algorithm to a baseline function, a simple majority classifier. (A majority classifier simply outputs the most common class label from the dataset without considering any of the examples attribute values.) What is the accuracy of the majority classifier? Briefly justify your answer.

   (Solution) The accuracy is zero. In the case of equal numbers of two classes, the leave-one-out cross validation for a majority classifier performs very poorly. Each time we remove one example, the majority class in the remaining data will be the opposite class of the removed example. For example, if we remove a positive example, the majority class in the remaining 29 examples (14 positive, 15 negative) will be negative. Each time we leave one example out, the majority classifier fails in predicting the actual class, so the accuracy becomes zero.