CS 412: Spring'21
Introduction To Data Mining

Assignment 5

Xiangcan Li

1. (a)
   - The expected information needed to classify a tuple in D is given by

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

where p_i is the nonzero probability that an arbitrary tuple in D belongs to class C_i and is estimated by $|Ci, D|/|D|$.
The expected information required to classify a tuple from D based on the partitioning by A is given by

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j),$$

where the term $\frac{|D_j|}{|D|}$ acts as the weight of the jth partition.
Information gain is defined as the difference between the original information requirement and the new requirement. That is,

$$Gain(A) = Info(D) - Info_A(D).$$

The potential information generated by splitting the training data set, D, into v partitions, corresponding to the v outcomes of a test on attribute A is given by

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

Then the gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}.$$

The attribute with the maximum gain ratio is selected as the splitting attribute.

   - The Gini measures the impurity of D, a data partition or set of training tuples, as

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

where p_i is the probability that a tuple in D belongs to class C_i and is estimated by $|C_i, D|/|D|$. The sum is computed over m classes.

If we split the training data set, D, into v partitions on attribute A, a weighted sum of the impurity of such partition is given by

$$Gini_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} Gini(D_j).$$

For a discrete-valued attribute, the subset that gives the minimum Gini impurity for that attribute is selected as its splitting subset.
For continuous-valued attributes, each possible split point must be considered. The point giving the minimum Gini impurity for a given (continuousvalued) attribute is taken as the split point of that attribute.

(b) Attribute 'Patrons?' is used to split the restaurant dataset into 3 partitions of subsets, $\{\text{None}, \text{Some}, \text{Full}\}$.
Then we have

$$Gini(\text{None}) = 1 - \left(\frac{0}{2} \times \frac{0}{2} + \frac{2}{2} \times \frac{2}{2}\right) = 0$$
$$Gini(\text{Some}) = 1 - \left(\frac{4}{4} \times \frac{4}{4} + \frac{0}{4} \times \frac{0}{4}\right) = 0$$
$$Gini(\text{Full}) = 1 - \left(\frac{2}{6} \times \frac{2}{6} + \frac{4}{6} \times \frac{4}{6}\right) = 0.44$$
$$Gini_{\text{Patrons}}(\text{restaurant}) = \frac{2}{12} \times 0 + \frac{4}{12} \times 0 + \frac{6}{12} \times 0.44 = 0.22.$$

Attribute 'Type?' is used to split the restaurant dataset into 4 partitions of subsets, $\{\text{French}, \text{Italian}, \text{Thai}, \text{Burger}\}$.
Then we have

$$Gini(\text{French}) = 1 - \left(\frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2}\right) = \frac{1}{2}$$
$$Gini(\text{Italian}) = 1 - \left(\frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2}\right) = \frac{1}{2}$$
$$Gini(\text{Thai}) = 1 - \left(\frac{2}{4} \times \frac{2}{4} + \frac{2}{4} \times \frac{2}{4}\right) = \frac{1}{2}$$
$$Gini(\text{Burger}) = 1 - \left(\frac{2}{4} \times \frac{2}{4} + \frac{2}{4} \times \frac{2}{4}\right) = \frac{1}{2}$$
$$Gini_{\text{Type}}(\text{restaurant}) = \frac{2}{12} \times \frac{1}{2} + \frac{2}{12} \times \frac{1}{2} + \frac{4}{12} \times \frac{1}{2} + \frac{4}{12} \times \frac{1}{2} = 0.5.$$

Since $0.22 < 0.5$, the Gini impurity for the split on 'Patrons?' is less. Hence, I will split on 'Patrons?' at the root.

(c) Let D represents the restaurant dataset. We first compute the expected information needed to classify a tuple D:

$$Info(D) = -\frac{6}{12}\log_2\frac{6}{12} - \frac{6}{12}\log_2\frac{6}{12} = 1.$$

Attribute 'Patrons?' is used to split D into 3 partitions of subsets, $\{\text{None}, \text{Some}, \text{Full}\}$. Let A represents attribute 'Patrons?' and D_1, D_2, D_3 represent the None dataset, Some dataset, Full dataset, respectively.
Then we have

$$\begin{aligned}
Info(D_1) &= -\frac{0}{2}\log_2\frac{0}{2} - \frac{2}{2}\log_2\frac{2}{2} = 0 \\
Info(D_2) &= -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0 \\
Info(D_3) &= -\frac{2}{6}\log_2\frac{2}{6} - \frac{4}{6}\log_2\frac{4}{6} \approx 0.92 \\
Info_A(D) &= \frac{2}{12}\times 0 + \frac{4}{12}\times 0 + \frac{6}{12}\times 0.92 = 0.46
\end{aligned}$$

Hence, the gain in information from such a partitioning would be

$$Gain(A) = Info(D) - Info_A(D) = 1 - 0.46 = 0.54.$$

The potential information generated by splitting the training data set, D, into 3 partitions is

$$SplitInfo_A(D) = -\frac{2}{12}\log_2\frac{2}{12} - \frac{4}{12}\log_2\frac{4}{12} - \frac{6}{12}\log_2\frac{6}{12} \approx 1.46.$$

Hence, the gain ratio is

$$GainRatio(A) = \frac{0.54}{1.46} = 0.37.$$

Attribute 'Type?' is used to split D into 4 partitions of subsets, $\{\text{French}, \text{Italian}, \text{Thai}, \text{Burger}\}$. Let B represents attribute 'Type?' and D_1, D_2, D_3, D_4 represent the French dataset, Italian dataset, Thai dataset, Burger dataset, respectively.
Then we have

$$\begin{aligned}
Info(D_1) &= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1 \\
Info(D_2) &= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = 1 \\
Info(D_3) &= -\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = 1 \\
Info(D_4) &= -\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = 1 \\
Info_B(D) &= \frac{2}{12}\times 1 + \frac{2}{12}\times 1 + \frac{4}{12}\times 1 + \frac{4}{12}\times 1 = 1
\end{aligned}$$

Hence, the gain in information from such a partitioning would be

$$Gain(B) = Info(D) - Info_B(D) = 1 - 1 = 0.$$

The potential information generated by splitting the training data set, D, into 3 partitions is

$$SplitInfo_B(D) = -\frac{2}{12}\log_2\frac{2}{12} - \frac{2}{12}\log_2\frac{2}{12} - \frac{4}{12}\log_2\frac{4}{12} - \frac{4}{12}\log_2\frac{4}{12} \approx 1.92.$$

Hence, the gain ratio is

$$GainRatio(B) = \frac{0}{1.92} = 0.$$

Since $0.37 > 0$, the GainRatio for the split on 'Patrons?' is larger. Hence, I will split on 'Patrons?' at the root.

2. From Bayes's theorem, we have $P(h_k|\mathbf{X}) = \alpha P(\mathbf{X}|h_k)P(h_k)$, where α is some constant.

   (a) Then we have

$$\begin{aligned}
P(h_1|\text{Candy}_1 = \text{lime}) &= \alpha P(\text{Candy}_1 = \text{lime}|h_1)P(h_1) = \alpha \cdot 0 \cdot 0.25 = 0 \\
P(h_2|\text{Candy}_1 = \text{lime}) &= \alpha P(\text{Candy}_1 = \text{lime}|h_2)P(h_2) = \alpha \cdot 0.5 \cdot 0.5 = 0.25\alpha \\
P(h_3|\text{Candy}_1 = \text{lime}) &= \alpha P(\text{Candy}_1 = \text{lime}|h_3)P(h_3) = \alpha \cdot 1 \cdot 0.25 = 0.25\alpha
\end{aligned}$$

   Hence, $0.25\alpha + 0.25\alpha = 1$, which indicates that $\alpha = 2$. Then

$$\begin{aligned}
P(h_1|\text{Candy}_1 = \text{lime}) &= 0 \\
P(h_2|\text{Candy}_1 = \text{lime}) &= 0.5 \\
P(h_3|\text{Candy}_1 = \text{lime}) &= 0.5
\end{aligned}$$

   (b) From the Bayes's theorem, We have

$$\begin{aligned}
p(h_1|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= \alpha P(\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}|h_1)P(h_1) \\
&= \alpha \cdot 0 \cdot 1 \cdot 0.25 \\
&= 0 \\
P(h_2|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= \alpha P(\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}|h_2)P(h_2) \\
&= \alpha \cdot 0.5 \cdot 0.5 \cdot 0.5 \\
&= 0.125\alpha \\
P(h_3|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= \alpha P(\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}|h_3)P(h_3) \\
&= \alpha \cdot 1 \cdot 0 \cdot 0.25 \\
&= 0
\end{aligned}$$

   Hence, $0.125\alpha = 1$, which indicates that $\alpha = 8$. Then

$$\begin{aligned}
P(h_1|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= 0 \\
P(h_2|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= 1 \\
P(h_3|\text{Candy}_1 = \text{lime}, \text{Candy}_2 = \text{cherry}) &= 0
\end{aligned}$$

3. (a) Note that there are only two values for size, color and shape. Hence, given the class label, which is either Yes or No, if we know the probability of Small, we could also know the probability of Big. This holds color and shape class. Furthermore, for the class label, if we know the probability of yes, then we also know the probability of no.
   Hence, the number of independent parameters are

$$2 \cdot (2-1) + 2 \cdot (2-1) + 2 \cdot (2-1) + (2-1) = 7.$$

   Those independent parameters are

   $P(Small|Yes), P(Small|No), P(Red|Yes), P(Red|No), P(Circle|Yes), P(Circle|No), P(Yes)$.

Size	Yes	No	P(Size | Yes)	P(Size | No)	
Small	1	3	1/4	3/6	4/10
Large	3	3	3/4	3/6	6/10
			4/10	6/10	
Color	Yes	No	P(Color | Yes)	P(Color | No)	
Red	4	1	4/4	1/6	5/10
Green	0	5	0/4	5/6	5/10
			4/10	6/10	
Shape	Yes	No	P(Shape | Yes)	P(Shape | No)	
Circle	3	2	3/4	2/6	5/10
Irregular	1	4	1/4	4/6	5/10
			4/10	6/10	
Good Apple	Yes	No	P(Yes)	P(No)	
	4	6	4/10	6/10	

Table 1: Likelihhod Table.

(b) We first draw the likelihood table as above.
From the table above, we obtain the values of independent parameters as follow:

$$P(Small|Yes) = 0.25, P(Small|No) = 0.5, P(Red|Yes) = 1, P(Red|No) = 0.17$$
$$P(Circle|Yes) = 0.75, P(Circle|No) = 0.33, P(Yes) = 0.4$$

(c) The tuple we wish to classify is

$$\mathbf{X} = (Size = Small, Color = Red, Shape = Circle)$$

We need to find out which class maximizes $P(\mathbf{X}|C_i)P(C_i)$, for $i = 1, 2$. $P(C_i)$, the prior probability of each class, can be computed based on the training tuples:

$$P(good\_apple = Yes) = 4/10 = 0.4$$
$$P(good\_apple = No) \ = 6/10 = 0.6$$

To compute $P(\mathbf{X}|C_i)$, for $i = 1, 2$, we compute the following conditional probabilities:

$$P(Size = Small \mid good\_apple = Yes) = 1/4 = 0.25$$
$$P(Size = Small \mid good\_apple = No) = 3/6 = 0.5$$
$$P(Color = Red \mid good\_apple = Yes) = 4/4 = 1$$
$$P(Color = Red \mid good\_apple = No) = 1/6$$
$$P(Shape = Circle \mid good\_apple = Yes) = 3/4 = 0.75$$
$$P(Shape = Circle \mid good\_apple = No) = 2/6 = 1/3$$

Using these probabilities, we obtain

$$\begin{aligned} P(\mathbf{X} \mid good\_apple = Yes) =& P(Size = Small \mid good\_apple = Yes) \\ & \times P(Color = Red \mid good\_apple = Yes) \\ & \times P(Shape = Circle \mid good\_apple = Yes) \\ =& 0.25 \times 1 \times 0.75 = 0.1875 \end{aligned}$$

Similarly,

$$\begin{aligned} P(\mathbf{X} \mid good\_apple = No) =& P(Size = Small \mid good\_apple = No) \\ & \times P(Color = Red \mid good\_apple = No) \\ & \times P(Shape = Circle \mid good\_apple = No) \\ =& 0.5 \times 1/6 \times 1/3 = 1/36 \end{aligned}$$

To find the class, C_i, that maximizes $P(\mathbf{X}|C_i)P(C_i)$, we compute

$$P(\mathbf{X} \mid good\_apple = Yes)P(good\_apple = Yes) = 0.1875 \times 0.4 = 0.075$$
$$P(\mathbf{X} \mid good\_apple = No)P(good\_apple = No) = 1/36 \times 0.6 = 0.0167$$

Since $0.075 > 0.0167$, the naive Bayes classifier predicts $good\_apple = Yes$ for tuple $\mathbf{X}$.

4. (a) Given a set, D, of d tuples, bagging works as follows. For iteration $i(i = 1, 2, ..., k)$, a training set, D_i, of d tuples is sampled with replacement from the original set of tuples, D.

   The bootstrap method samples the given training tuples uniformly with replacement. That is, each time a tuple is selected, it is equally likely to be selected again and re-added to the training set.

   RFs can be built using bagging in tandem with random attribute selection. A training set, D, of d tuples is given. The general procedure to generate k decision trees for the ensemble is as follows. For each iteration, $i(i = 1, 2, ..., k)$, a training set, D_i, of d tuples is sampled with replacement from D. That is, each D_i is a bootstrap sample of D, so that some tuples may occur more than once in D_i , while others may be excluded. Let F be the number of attributes to be used to determine the split at each node, where F is much smaller than the number of available attributes. To construct a decision tree classifier, M_i, randomly select, at each node, F attributes as candidates for the split at the node. The CART methodology is used to grow the trees. The trees are grown to maximum size and are not pruned.

   Precision is the number of correctly-identified members of a class divided by all the times the model predicted that class. In the context of classification, the precision is done by such division.

   d: Tree depth represents depth of each tree in the forest. The deeper the tree, the more splits it has and it captures more information about the data.

   m: RFs randomly select m attributes from available attributes to be used to determine the split at each node. RFs are built over a random extraction of the observations from the dataset and a random extraction of the features. Not every tree sees all the features, which guarantees that the trees are decorrelated.

(b) The bootstrap method samples the given training tuples uniformly with replacement. Suppose we are given a data set of n tuples. The data set is sampled n times, with replacement, resulting in a training set of n samples.

Each tuple has a probability of $1/n$ of being selected, so the probability of not being chosen is $(1 - 1/n)$. We have to select n times, so the probability that a tuple will not be chosen during this whole time is $(1 - 1/n)^n$.

Note that

$$\lim_{n\to\infty} (1 - 1/n)^n = 1/e.$$

Hence, If n is large, the probability approaches $1/e = 0.368$. Thus, 36.8% of tuples will not be selected for training and the remaining 63.2% will form the training set.

We conclude the expected number of unique samples from the original set of n samples in the bootstrapped sample is $63.2\%n$ given n is large.

(c) The idea of random forests is basically to build many decision trees decorrelated. Note that if there are few dominating features, these features will be selected in many trees even for different subsamples, making the trees in the forest correlated again. Hence, subsampling features can further decorate trees.

The lower the number of sampled features, the higher the decorrelation effect. On the other hand, the random forest bias is the same as the bias of any of the sampled trees. Still, the randomization of random forests restricts the model so that the bias is usually higher than a fully-grown tree. Hence, if we sample fewer feature, we can expect a higher bias.

We can use mean squared error (MSE) to measure accuracy. We have

$$\mathrm{MSE}(\theta) = \mathrm{Var}(\theta) + \mathrm{Bias}^2(\theta).$$

Although we can reduce the variance, we may get a higher bias. Hence, $\mathrm{MSE}(\theta)$ may not decrease and thus that will not improve accuracy.

Extra Credit.

1. (a) Sensitivity is the true positive rate.
   Sensitivity $= a/(a+b) = 2588/(2588+412) = 86.27\%$.

   (b) Specificity is the true negative rate
   Specificity $= d/(b+d) = 6954/(46+6954) = 99.34\%$.

   (c) Accuracy is the recognition rate
   Accuracy $= (a+d)/(a+b+c+d) = (2588+6954)/(2588+412+46+6954) = 95.42\%$.

   (d) Precision is $TP/(TP+FP)$.
   Precision $= a/(a+c) = 2588/(2588+46) = 98.25\%$.

   (e) Recall is TP/P.
   Recall $= a/(a+b) = 2588/(2588+412) = 86.27\%$.

   (f)

$$\begin{aligned}\text{F1 score} &= 2 \cdot (\text{Recall} \cdot \text{Precision})/(\text{Recall} + \text{Precision}) \\ &= 2 \cdot (0.9825 \cdot 0.8627)/(0.9825 + 0.8627) \\ &= 0.9187\end{aligned}$$

   I disagree with Professor Griffin. If there are some essential emails like scheduling for the meetings with your boss deleted by the spam detector, it will cost us a lot. However, if the spam detector fails to recognize some spam emails, then we can view them and then delete them manually, which will only cost us several minutes. Hence, I believe the ideal spam detector should have high precision at the cost of having a low recall.