The Naïve Bayes Classifier

Introduction

- *Naïve Bayes* is a probabilistic classification method based on Bayes' theorem (or Bayes' law) with a few tweaks.

- Bayes' theorem gives the relationship between the probabilities of two events and their conditional probabilities.

Thomas Bayes

- Bayes' theorem is named after the English mathematician Thomas Bayes.
- `https://en.wikipedia.org/wiki/Thomas_Bayes`
- We'll introduce Bayes' Theorem and Naive Bayes classifier.

Conditional Probability

Sometimes, knowing event B has occurred gives us more information about A.

Definition (Conditional Probability)

- Suppose we have two events A and B within a sample space S.
- The **conditional probability** of A given B is defined to be

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

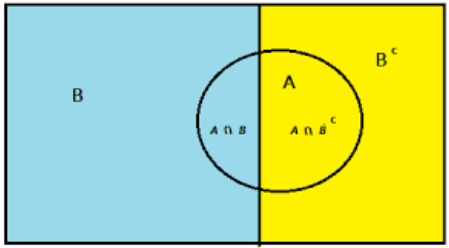
when $P(B) > 0$.

Conditional Probability

- From a well shuffled deck of 52 card, the probability to have a heart from a random draw is $P(A) = 1/4$. However, if we know the color of the withdrawn card (B), then the probability that the card is a heart, $P(A|B)$, will be different from 1/4.

- In the population, the probability of a random person to have diabetes, $P(A)$, is 8%. However, if we know further that the person is in the old range of age (B), then the probability of having diabetes $P(A|B)$ now might be different from 8%.

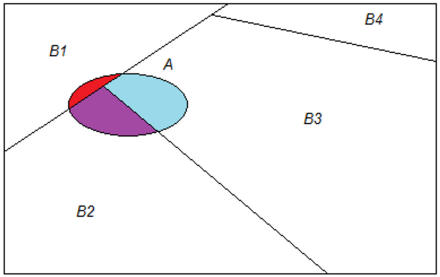
Law of Total Probability



- Law of total probability for the simple case: Let B denote an event and B^c denote the complement of B. For any event A we then have:

$$P(A) = P(A \cap B) + P(A \cap B^c).$$

Law of Total Probability



- For general: If $B_1, B_2, ..., B_k$ are mutually exclusive where the union of them fulfil the whole sample space, then for any event A, we have:

$$P(A) = P(A \cap B_1) + P(A \cap B_2) + ... + P(A \cap B_k).$$

Bayes' Theorem

- The conditional probability of event A occurring, given that event B has already occurred, is denoted as $P(A|B)$, which is defined as

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A) \times P(A)}{P(B)},$$

  where $P(B \cap A) = P(A \cap B)$ is the probability that both events B and A occur.

- Bayes' theorem is significant because quite often $P(A|B)$ is much more difficult to compute than $P(B|A)$ and $P(A)$ from the training data.

- We will illustrate the use of this theorem with a few examples.

Example 1: Lab Test

- Suppose, for a certain disease, 1% of the entire population has this disease (*prevelance* is 1%).
- A test returns a positive result in 95% of the cases in which the disease is actually present (known as *sensitivity*); and it returns a positive result in 6% of the cases in which the disease is not present (known as *false positive*).
- Suppose that a patient takes a lab test for that disease and the test shows positive result. With positive result given, what is the probability that the patient actually has the disease?

Example 1: Lab Test

- Define the events $A = \{\text{having the disease}\}$ and $B = \{\text{positive test result}\}$.
  From the information given, we have:

$$P(A) = 0.01; \quad P(A^c) = 0.99; \quad P(B \mid A) = 0.95; \quad P(B \mid A^c) = 0.06.$$

- Our aim: compute the conditional probability $P(A|B)$.
- Using Bayes' theorem, conditional probability and the law of total probability:

$$\begin{aligned} P(A|B) &= \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{P(B \cap A) + P(B \cap A^c)} \\ &= \frac{P(B|A)P(A)}{P(A) \times P(B|A) + P(A^c)P(B|A^c)} \\ &= \frac{0.95 \times 0.01}{0.01 \times 0.95 + 0.99 \times 0.06} \approx 0.1379 \end{aligned}$$

Example 1: Lab Test

- The result means that the probability of the patient actually having the disease given a positive test result is 13.79%.
- Without any test result, the probability of the patient actually having the disease is only 1%.
- The probability of being labelled as having the disease does increase after incorporating the feature variable of test result.

Example 2: Email Filter

- Suppose that 5% of all emails are spams. The phrase “you are a winner" occurs in 50% of spam emails, and in 10% of non-spam emails.

- Given an email with the phrase “you are a winner" in it, what is the probability that it is a spam email?

Example 2: Email Filter

- Define the events $A = \{\text{email is spam}\}$ and $B = \{\text{email contains the phrase "you are a winner"}\}$.

$$P(A) = 0.05, \quad P(A^c) = 0.95, \quad P(B \mid A) = 0.50, \quad P(B \mid A^c) = 0.10.$$

- Our aim: compute $P(A|B)$.

$$\begin{aligned} P(A|B) &= \frac{P(B|A)P(A)}{P(B)} = \frac{P(B|A)P(A)}{P(B \cap A) + P(B \cap A^c)} \\ &= \frac{P(B|A)P(A)}{P(A) \times P(B|A) + P(A^c)P(B|A^c)} \\ &= \frac{0.50 \times 0.05}{0.05 \times 0.50 + 0.95 \times 0.10} \approx 0.208 \end{aligned}$$

Example 2: Email Filter

- Without any knowledge of occurrence of the phrase in an email, the probability of that email being a spam is only 5%.
- The result means that the probability of an email being a spam given that it contains the phrase “you are a winner" is about 20.8%.
- Note that we often use more than one feature variable in making predictions.
- For example, on top of the phrase “you are a winner”, the occurrence of the phrase “transfer bank account" can be another feature in predicting spam.
- The more general form of Bayes' theorem allows us to incorporate multiple feature variables or attributes.

Naïve Bayes Classifier - Notes

- A *naïve Bayes* classifier assumes that the effect of features to the response is independent of each other.

- It means, the *naïve Bayes* classifier considers each feature contributes independently to the probability that the object belongs to a class label.

- For example, an object can be classified based on its attributes such as shape, color, and weight. The response has two categories: tennis ball (y_1) and soccer ball (y_2). Then, the contribution of shape in probability $P(Y = y_j|X)$ is independent of the color's contribution, and also is independent of the weight's contribution.

Naïve Bayes Classifier - Notes

- The input variables are generally categorical, but variations of the algorithm can accept continuous variables.
- There are also ways to convert a continuous variable into a categorical one. This process is often referred to as the discretization of continuous variables.
- For example, weight can be discretized to the categories $\leq 1kg$, $1kg$ to $\leq 5kg$, and $> 5kg$.
- The output typically includes a class label and its corresponding probability score.

Naïve Bayes Classifier - Applications

- *Naïve Bayes* classifiers are easy to implement and can execute efficiently even without prior knowledge of the data, they are among the most popular algorithms for classifying text documents.
- Spam filtering is a classic use case of *naïve Bayes* text classification.
- *Naïve Bayes* classifiers can also be used for fraud detection.
- In the domain of auto insurance, for example, based on a training set with attributes such as driver's rating, vehicle age, vehicle price, historical claims by the policy holder, police report status, and claim genuineness, *naïve Bayes* can provide probability-based classification of whether a new claim is genuine.

Bayes' Theorem for Naïve Bayes Classifier

- Suppose the categorical outcome variable Y takes on values in the set $\{y_1, y_2, ..., y_k\}$. For example, binary response Y has $k = 2$.

- Given m features $X = \{X_1, X_2, ..., X_m\}$, we would need to calculate the probability that the object belong to each category of the response, $P(Y = y_j|X)$, $j = 1, 2, ..., k$.

- Then, the object will be classified to the category y_j with the largest probability $P(Y = y_j|X)$.

Bayes' Theorem for Naïve Bayes Classifier

- With Bayes theorem, we have

$$P(Y = y_j|X) = \frac{P(X_1 = x_1, X_2 = x_2, ..., X_m = x_m|Y = y_j) \times P(Y = y_j)}{P(X_1 = x_1, X_2 = x_2, ..., X_m = x_m)}, \quad j = 1, 2, ..., k$$

- With two simplifications, Bayes' theorem can be extended to become a Naïve Bayes classifier.

Bayes' Theorem for Naïve Bayes Classifier

- The first simplification is to use the conditional independence assumption between features X's which simplifies the computation of the numerator term,

$$
\begin{aligned}
&P(X_1 = x_1, X_2 = x_2, ..., X_m = x_m | Y = y_j) \\
&= P(X_1 = x_1 | Y = y_j) P(X_2 = x_2 | Y = y_j) ... P(X_m = x_m | Y = y_j) \\
&= \prod_{i=1}^{m} P(X_i = x_i | Y = y_j).
\end{aligned}
$$

Bayes' Theorem for Naïve Bayes Classifier

- The second simplification is to ignore the term in the denominator,

$$P(X_1 = x_1, X_2 = x_2, ..., X_m = x_m)$$

since it is constant (the same) for all response categories $j = 1, ..., k$ in the set $\{y_1, y_2, ..., y_k\}$.

- Hence, for $j = 1, 2, ..., k$, we have

$$P(Y = y_j|X) \propto P(Y = y_j) \times \prod_{i=1}^{m} P(X_i = x_i|Y = y_j)$$

where each $P(Y = y_j)$ is known from the data, and each $P(X_i = x_i|Y = y_j)$ is calculate-able given the data.

Naïve Bayes Classifier: Fruit Example

- Suppose we wish to predict the class of fruits, Y, that takes on the values *Banana, Orange, Other*. Hence, $k = 3$ here.
- The binary feature variables X are whether the fruit is long, sweet and yellow. Hence, we have 3 features here.
- The tabulation on 1000 pieces of fruit is as follows:

Y	Long	Not Long	Sweet	Not Sweet	Yellow	Not Yellow	Total
Banana	200	300	100	400	200	300	500
Orange	20	280	100	200	180	120	300
Other	100	100	50	150	50	150	200

Fruit Example

Y	Long	Not Long	Sweet	Not Sweet	Yellow	Not Yellow	Total
Banana	200	300	100	400	200	300	500
Orange	20	280	100	200	180	120	300
Other	100	100	50	150	50	150	200

- We can get $P(Y = y_j)$ easily:

$$P(Y = Banana) = \frac{500}{1000} = 0.5,$$

$$P(Y = Orange) = \frac{300}{1000} = 0.3,$$

$$P(Y = Other) = \frac{200}{1000} = 0.2.$$

Fruit Example

Y	Long	Not Long	Sweet	Not Sweet	Yellow	Not Yellow	Total
Banana	200	300	100	400	200	300	500
Orange	20	280	100	200	180	120	300
Other	100	100	50	150	50	150	200

The conditional probabilities are

i	x_i	$P(x_i \mid Y = \text{Banana})$	$P(x_i \mid Y = \text{Orange})$	$P(x_i \mid Y = \text{Others})$
1	Long	$200/500$	$20/300$	$100/200$
2	Sweet	$100/500$	$100/300$	$50/200$
3	Yellow	$200/500$	$180/300$	$50/200$

Fruit Example: Prediction

- Suppose we want to predict the identity for a new piece of fruit which is **long**, **sweet** but **not yellow**, then we need to calculate three probabilities below:

$$P(Y = Banana|X_1 = long, X_2 = sweet, X_3 = not\ yellow)$$

$$P(Y = Orange|X_1 = long, X_2 = sweet, X_3 = not\ yellow)$$

$$P(Y = Others|X_1 = long, X_2 = sweet, X_3 = not\ yellow)$$

Fruit Example: Prediction

Given the information of three features, the probability that the fruit is a **banana** is:

$$
\begin{aligned}
&P(Y = Banana | X_1 = long, X_2 = sweet, X_3 = not\ yellow) \\
&\propto P(Y = Banana) \times P(X_1 = Long | Y = Banana) \\
&\times P(X_2 = Sweet | Y = Banana) \times P(X_3 = Not\ Yellow | Y = Banana) \\
&= 0.5 \times \frac{200}{500} \times \frac{100}{500} \times \left(1 - \frac{200}{500}\right) \\
&= 0.024
\end{aligned}
$$

Fruit Example: Prediction

Given the information of three features, the probability that the fruit is an **orange** is:

$$
\begin{aligned}
& P(Y = Orange|X) \\
& \propto P(Y = Orange) \times P(X_1 = Long|Y = Orange) \\
& \times P(X_2 = Sweet|Y = Orange) \times P(X_3 = Not\ Yellow|Y = Orange) \\
& = 0.3 \times \frac{20}{300} \times \frac{100}{300} \times \left(1 - \frac{180}{300}\right) \\
& \approx 0.0027
\end{aligned}
$$

Fruit Example: Prediction

Given the information of three features, the probability that the fruit belongs to "**Others**" is

$$
\begin{aligned}
& P(Y = Others|X) \\
& \propto P(Y = Others) \times P(X_1 = Long|Y = Others) \\
& \times P(X_2 = Sweet|Y = Others) \times P(X_3 = Not\ Yellow|Y = Others) \\
& = 0.2 \times \frac{100}{200} \times \frac{50}{200} \times \left(1 - \frac{50}{200}\right) \\
& \approx 0.0188
\end{aligned}
$$

$P(Y = Banana|X) \propto 0.024$ is **largest** among three, we predict the fruit to be a banana.

Naïve Bayes Classifier with Log Probabilities

- When looking at problems with a large number of feature values, or outcome with many categories, the conditional probability can become very small in magnitude (close to zero).
- This is the problem of numerical underflow, caused by multiplying several probability values that are close to zero.
- A way to alleviate the problem is to compute the logarithm of the probability scores:

$$\log P(Y = y_j) + \sum_{i=1}^{m} \log P(X_i = x_i | Y = y_j),$$

for $j = 1, 2, ..., k$.

Naïve Bayes Classifier: Example 2

- Aim: predict whether employees would enroll in an onsite educational program based on feature variables such as `Age`, `Income`, `JobStatisfaction` and `Desire`.

- We will illustrate with both manual calculation and using the `naiveBayes` function in the package 'e1071' in R.

Data file is `sample1.csv`. The last row (15th) has no outcome and is for prediction.

Example 2

```
> sample <- read.table("C:/Data/sample1.csv",header=TRUE,sep=",")
> # Enrolls = RESPONSE with 2 categories
> sample
       Age Income JobSatisfaction    Desire Enrolls
1      <=30   High              No      Fair      No
2      <=30   High              No Excellent      No
3  31 to 40   High              No      Fair     Yes
4       >40 Medium              No      Fair     Yes
5       >40    Low             Yes      Fair     Yes
6       >40    Low             Yes Excellent      No
7  31 to 40    Low             Yes Excellent     Yes
8      <=30 Medium              No      Fair      No
9      <=30    Low             Yes      Fair     Yes
10      >40 Medium             Yes      Fair     Yes
11     <=30 Medium             Yes Excellent     Yes
12 31 to 40 Medium              No Excellent     Yes
```

Example 2

- Two data frame objects called `traindata` and `testdata` are created for the naïve Bayes Classifier.

- We will train the classifier using `traindata`, then make predictions for the single record in `testdata`.

```
> traindata <- as.data.frame(sample[1:14,]) # first 14 rows
> testdata <- as.data.frame(sample[15,]) # the 15th row
> testdata
    Age Income JobSatisfaction Desire Enrolls
15 <=30 Medium             Yes   Fair
```

Example 2: $P(Y = y_j)$

- We will first illustrate the naïve Bayes classifier via manual computation.
- Response 'Enrolls' has 2 categories, hence, we need to compute the probabilities $P(Y = Yes)$ and $P(Y = No)$.

```
> tprior <- table(traindata$Enrolls);tprior
 No Yes
  5   9
> tprior <- tprior/sum(tprior); tprior
       No       Yes
0.3571429 0.6428571
```

Example 2: $P(X_i = x_i|Y = y_j)$

- Next, we need to compute the conditional probabilities $P(X_i = x_i|Y = 1)$ and $P(X_i = x_i|Y = 0)$, where $i = 1, 2, 3, 4$ for the feature variables $X = \{\text{Age, Income, JobSatisfaction, Desire}\}$.
- First, compute the conditional probabilities for Age:

```
> ageCounts <- table(traindata[,c("Enrolls", "Age")]);ageCounts
       Age
Enrolls <=30 >40 31 to 40
    No     3   2        0
    Yes    2   3        4
> ageCounts <- ageCounts/rowSums(ageCounts); ageCounts
       Age
Enrolls      <=30       >40  31 to 40
    No  0.6000000 0.4000000 0.0000000
    Yes 0.2222222 0.3333333 0.4444444
```

It means $P(\text{Age} \leq 30|Y = 1) = 0.2222$; $P(\text{Age is 31 to 40}|Y = 1) = 0.4444$, etc.

Example 2: $P(X_i = x_i | Y = y_j)$

- We perform similar operations for `Income`:

```
> incomeCounts <- table(traindata[,c("Enrolls", "Income")])
> incomeCounts <- incomeCounts/rowSums(incomeCounts);incomeCounts
```

```
        Income
Enrolls      High       Low    Medium
    No  0.4000000 0.2000000 0.4000000
    Yes 0.2222222 0.3333333 0.4444444
```

- This means, $P(\text{Income} = High | Y = No) = 0.4$; $P(\text{Income} = Low | Y = No) = 0.2$ and $P(\text{Income} = Medium | Y = No) = 0.4$.
- Similar for the category $Y = Yes$.

Example 2: $P(X_i = x_i|Y = y_j)$

- We perform similar operations for JobSatisfaction:

```
> jsCounts <- table(traindata[,c("Enrolls", "JobSatisfaction")])
> jsCounts <- jsCounts/rowSums(jsCounts);jsCounts
```

```
        JobSatisfaction
Enrolls        No       Yes
    No  0.8000000 0.2000000
    Yes 0.3333333 0.6666667
```

- This means, $P(\text{Desire} = Excellent|Y = No) = 0.6$; $P(\text{Desire} = Fair|Y = No) = 0.4$ and
  $P(\text{Desire} = Excellent|Y = Yes) = 0.3333$ and $P(\text{Desire} = Fair|Y = Yes) = 0.0.6667$

Example 2: $P(X_i = x_i | Y = y_j)$

- We perform similar operations for `Desire`:

```
> desireCounts <- table(traindata[,c("Enrolls", "Desire")])
> desireCounts <- desireCounts/rowSums(desireCounts);desireCounts
       Desire
Enrolls Excellent      Fair
    No  0.6000000 0.4000000
    Yes 0.3333333 0.6666667
```

Example 2: $P(Y = y_j|X)$

- For the test point, we'll compute the probability scores

$$P(Y = 1|X) \propto P(Y = 1) \times \prod_{i=1}^{4} P(X_i = x_i|Y = 1)$$

and

$$P(Y = 0|X) \propto P(Y = 0) \times \prod_{i=1}^{4} P(X_i = x_i|Y = 0)$$

where $X = (\text{Age} \leq 30,\ \text{Income} = \text{Medium},\ \text{JabSatisfaction} = Yes,\ \text{Desire} = \text{Fair})$.

Example 2: $P(Y = y_j|X)$

- $P(Y = 1|X)$ or $P(\text{Enrolls} = \text{Yes}|X)$ is proportional to

```
> prob_yes <-
+ ageCounts["Yes",testdata[,c("Age")]]*
+ incomeCounts["Yes",testdata[,c("Income")]]*
+ jsCounts["Yes",testdata[,c("JobSatisfaction")]]*
+ desireCounts["Yes",testdata[,c("Desire")]]*
+ tprior["Yes"]
> prob_yes
      Yes
0.02821869
```

Example 2: $P(Y = y_j|X)$

- $P(Y = 0|X)$ or $P(\text{Enrolls} = \text{No}|X)$ is proportional to

```
> prob_no <-
+ ageCounts["No",testdata[,c("Age")]]*
+ incomeCounts["No",testdata[,c("Income")]]*
+ jsCounts["No",testdata[,c("JobSatisfaction")]]*
+ desireCounts["No",testdata[,c("Desire")]]*
+ tprior["No"]
> prob_no
          No
0.006857143
```

- Take the ratio $P(Y = 1|X)/P(Y = 0|X)$, we have

```
> prob_yes/prob_no
     Yes
4.115226
```

- The predicted result for the test point is 'Yes'.

Example 2: Using built in package

- Alternatively, we can use the naiveBayes function in the R package 'e1071' to perform naïve Bayes classification:

```
> library(e1071)
> model <- naiveBayes(Enrolls ~ Age+Income+JobSatisfaction+Desire, traindat
> results <- predict(model,testdata,"raw")
> # use "raw" to get probabilities;
> #use "class" to get the category's name.
>
> results
           No       Yes
[1,] 0.1954948 0.8045052
> results[2]/results[1] # ratio of two probabilities = 4.115226
[1] 4.115226
```

Diagnotics for Naïve Bayes Classifier

- Recall that for diagnostics of a classifier, we have learnt about the confusion matrix as well as measures such as accuracy, precision, TPR, FPR (type I error rate), FNR (type II error rate) since Topic 4. These metrics could be used to measure how good a Naive Bayes classifier is.

- We now will familiarize ourselves with one additional diagnostics tool, the Receiver Operating Characteristic (**ROC**) **curve**, which is used for the case when the response is of binary outcome.

ROC Curve

- Recall that the False Positive Rate (FPR) and True Positive Rate (TPR) are calculated as

$$\text{FPR} = \frac{FP}{FP + TN} \quad \text{and} \quad \text{TPR} = \frac{TP}{TP + FN}.$$

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

- Recall that for classification using the majority rule, normally the response is predicted to be 1 if $\hat{Y} > 0.5$ and 0 otherwise. Here, 0.5 is used as the threshold for majority rule. This threshold could be changed which then will change the goodness of a classifier. (Please revisit Tutorial 5, Q3 for an example)

ROC Curve

- Let Y denote the response where 1 is for positive outcome and 0 is for negative outcome.
- If the **threshold is increased**, then less test objects will be predicted to be 1, and so TP will be either constant or decreases. However, the sum (TP + FN) is still constant because the number of objects with actual label $Y = 1$ is a constant in the test data set, so **TPR** will either be **constant or decreases**.
- Similarly, if the threshold is increased, FP will be either constant or decreases, while the sum (FP + TN) is a constant, so FPR will either be constant or decreases.
- Thus, in summary, if the **threshold is increased, both TPR and FPR generally decrease**.
- A good classifier has large TPR (close to 1) and small FPR (close to 0).

AUC

- A useful metric is to compute the Area Under the ROC Curve, **AUC**.
- Higher AUC scores mean the classifier performs better.
- AUC scores can be computed with the R package 'ROCR'.