MECG15603/MCCG15603 Statistical Learning MEME19903/MECG11103/MCCG11103 Predictive Modelling Topic 2b: Supervised Learning: Naive Bayes

Dr Liew How Hui

May 2024

Outline

Class Arrangement

- Week 9 (Done): Logistic Regression Model. An example of parametric model.
- Week 10: Lecture 1:30pm-3:30pm (Naive Bayes Model). Practical 3:30-4:30pm
- Week 11: Lecture 1:30pm-3:30pm. Practical 3:30-4:30pm
- Week 12: Lecture 1:30pm-3:30pm. Practical 3:30-4:30pm

Generative Models

Naive Bayes Models are generative models

$$\mathbb{P}(Y = y|\mathbf{X} = \mathbf{x}) = \frac{\mathbb{P}(\mathbf{X} = \mathbf{x}|Y = y)\mathbb{P}(Y = y)}{\mathbb{P}(\mathbf{X} = \mathbf{x})} \quad (1)$$

based on the Bayes Theorem

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

Note that '$\mathbb{P}$' is regarded as probability "mass" and "density" function when the variables are discrete and continuous respectively.

Generative Models (cont)

When the response variable Y is categorical and has K distinct values 1, ..., K, the generative model (1) becomes

$$\mathbb{P}(Y = j|\mathbf{X} = \mathbf{x}) = \frac{\mathbb{P}(\mathbf{X} = \mathbf{x}|Y = j)\mathbb{P}(Y = j)}{\sum_{k=1}^{K} \mathbb{P}(\mathbf{X} = \mathbf{x}|Y = k)\mathbb{P}(Y = k)}, \tag{2}$$

where $j \in \{1, \cdots, K\}$.

Generative Models (cont)

From the generative model (2) for categorical response, we can derive the **generative classifier**

$$
\begin{aligned}
h_D(\mathbf{x}) &= \underset{j\in\{1,\dots,K\}}{\operatorname{argmax}} \mathbb{P}(Y=j|\mathbf{X}=\mathbf{x}) \\
&= \underset{j\in\{1,\dots,K\}}{\operatorname{argmax}} \frac{\mathbb{P}(\mathbf{X}=\mathbf{x}|Y=j)\mathbb{P}(Y=j)}{\mathbb{P}(\mathbf{X}=\mathbf{x})} \\
&= \underset{j\in\{1,\dots,K\}}{\operatorname{argmax}} \mathbb{P}(\mathbf{X}=\mathbf{x}|Y=j)\mathbb{P}(Y=j) \\
&= \underset{j\in\{1,\dots,K\}}{\operatorname{argmax}} [\ln \mathbb{P}(\mathbf{X}=\mathbf{x}|Y=j) + \ln \mathbb{P}(Y=j)]
\end{aligned} \tag{3}
$$

Naïve Bayes Classifiers

A **naïve Bayes classifier (NB)** (`https://en.wikipedia.org/wiki/Naive_Bayes_classifier`) is a generative classifier (3) with **strong independence assumptions** on the likelihood function:

$$\begin{aligned}
&\mathbb{P}(\mathbf{X} = \mathbf{x} | Y = j) \\
&= \mathbb{P}(X_1 = x_1, X_2 = x_2, ..., X_p = x_p | Y = j) \\
&= \mathbb{P}(X_1 = x_1 | Y = j) \times \cdots \times \mathbb{P}(X_p = x_p | Y = j) \\
&= \prod_{i=1}^{p} \mathbb{P}(X_i = x_i | Y = j).
\end{aligned}$$

Naïve Bayes Classifiers (cont)

The strong independence assumptions allows the generative classifier (3) to be expressed as

$$
\begin{aligned}
h_D(\mathbf{x}) &= \underset{j\in\{1,\dots,K\}}{\text{argmax}}\, \mathbb{P}(Y=j)\prod_{i=1}^{p}\mathbb{P}(X_i=x_i|Y=j) \\
&= \underset{j\in\{1,\dots,K\}}{\text{argmax}}\, \ln\mathbb{P}(Y=j)+\left[\sum_{i=1}^{p}\ln\mathbb{P}(X_i=x_i|Y=j)\right].
\end{aligned}
\tag{4}
$$

Naïve Bayes Classifiers (cont)

The prior distribution $\mathbb{P}(Y = j)$ is usually estimated using maximum likelihood estimation (MLE) leading to

$$\mathbb{P}\widehat{(Y = j)} = \frac{\#\{i \ : \ y_i = j\}}{n}. \tag{5}$$

If we know the theoretical distribution of the outcome Y to be uniformly distributed, we can use

$$\mathbb{P}(Y = j) = \frac{1}{K}.$$

Naïve Bayes Classifiers (cont)

The features X_i can either be categorical or be numeric:

- One X_i is numeric — Gaussian NB
- One X_i is categorical — Categorical NB
- All X_i are binary — Bernoulli NB
- All X_i are integral — Multinomial NB & Complement NB(?)

Gaussian Naïve Bayes

For continuous inputs X_i in (4), it is assume that X_i is 'normal' and satisfies the Gaussian distribution:

$$\mathbb{P}(X_i = x_{ki} | Y = j) = \frac{1}{\sigma_j\sqrt{2\pi}} \exp(-\frac{(x_{ki} - \mu_j)^2}{2\sigma_j^2}). \quad (6)$$

The theoretical estimations of the mean μ_j and the standard deviation σ_j are

$$\mu_j = \mathbb{E}[X_i | Y = j], \quad \sigma_j^2 = \mathbb{E}[(X_i - \mu_j)^2 | Y = j]. \quad (7)$$

Gaussian Naïve Bayes (cont)

The maximum likelihood estimator for (7) provides us the following estimates:

$$\widehat{\mu}_j = \frac{1}{\sum_{k=1}^n I(y_k = j)} \sum_{k=1}^n x_{ki} I(y_i = j),$$

$$s_j = \sqrt{\frac{1}{(\sum_{k=1}^n I(y_k = j)) - 1} \sum_{k=1}^n (x_{ki} - \widehat{\mu}_j)^2 I(y_i = j)}. \tag{8}$$

Here n is the number of rows, j is one of the class in $\{1, 2, \cdots, K\}$.

Categorical NB

If the feature X_i is **categorical** and takes on d_i possible values in $\{c_1^{(i)}, \cdots, c_{d_i}^{(i)}\} =: \mathscr{C}_i$.

The maximum likelihood estimate of the likelihood function is

$$\widehat{\mathbb{P}(X_i = c | Y = j)} = \frac{\sum_{k=1}^n I(x_{ki} = c \bigwedge y_k = j)}{\sum_{k=1}^n I(y_k = j)} \tag{9}$$

for each $c \in \mathscr{C}_i$.

When there is no k such that $x_{ki} = c \wedge y_k = j$, the probability estimate $\widehat{\mathbb{P}(X_i = c | Y = j)} = 0$ and this may be bad for estimation.

Categorical NB (cont)

Therefore, the **Laplace smoothing** for (9) is introduced:

$$\mathbb{P}(\widehat{X_i = c|Y} = j) = \frac{\alpha + \sum_{k=1}^{n} I(x_{ki} = c \bigwedge y_k = j)}{\alpha d_i + \sum_{k=1}^{n} I(y_k = j)} \quad (10)$$

where α is a **smoothing parameter** and d_i is the number of available categories of feature X_i defined above.

When $\alpha = 0$, (10) is called **no/without Laplace smoothing**.

When $\alpha = 1$, (10) is called **(with) Laplace smoothing**.

When $0 < \alpha < 1$, (10) is called *Lidstone smoothing*,

Multinomial NB

The Naïve Bayes algorithm for multinomially distributed data is called a *multinomial Naïve Bayes classifier*:
Application: **text classification**

$$\begin{aligned} & h_D(\texttt{document}) \\ = & \operatorname*{argmax}_{j=1,\cdots,K} \mathbb{P}(\texttt{document}|Y=j)\mathbb{P}(Y=j) \\ = & \operatorname*{argmax}_{j=1,\cdots,K} \mathbb{P}(wc_1, wc_2, \cdots, wc_p|Y=j)\mathbb{P}(Y=j) \end{aligned}$$

where wc_i is the number of times the word X_i, $i = 1, \cdots, p$, occurred in the $\texttt{document}$, p is the size of the vocabulary.

Multinomial NB (cont)

Possible entries of "classes" for `document` are "scientific", "economic", "management", etc. A naïve estimate for $\mathbb{P}(Y = j)$ is

$$\begin{aligned}
&\mathbb{P}(Y = j) \\
\approx&\frac{\text{number of documents of class } j}{\text{number of documents, } n}; \\
&\mathbb{P}(X_i = wc_i | Y = j) \\
\approx&\frac{\begin{array}{l}\text{total number of the occurrences of} \\ \text{the word } X_i \text{ in documents of class } j\end{array}}{\text{total number of words } X_1, \cdots, X_p \text{ in documents of class } j} =: \theta_{ji}.
\end{aligned}$$

Multinomial NB (cont)

A more robust estimate of the parameters $\theta_j := (\theta_{j1}, \ldots, \theta_{jp})$ is given by a smoothed version of MLE:

$$\mathbb{P}(X_i = wc_i | Y = j) \approx \frac{N_{ji} + \alpha}{N_j + \alpha d_i}$$

where $N_{ji} = \sum_{y_i=j} wc_i$ is the number of times feature i appears in a sample of class j in the training set D, and $N_j = \sum_{i=1}^{n} N_{ji}$ is the total count of all features for class j.
For the smoothing priors $\alpha \geq 0$,
$\alpha < 1$ is called *Lidstone smoothing*,
$\alpha = 1$ is called *Laplace smoothing*.

Multinomial NB (cont)

The conditional probability is

$$\mathbb{P}(\mathbf{X} = \mathbf{x} | Y = j) = \frac{(\sum_{i=1}^{p} wc_i)!}{wc_1! \times \cdots \times wc_p!} \prod_{i=1}^{p} \theta_{ji}^{wc_i}. \qquad (11)$$

According to `https://scikit-learn.org/stable/modules/naive_bayes.html`, the Multinomial Naïve Bayes classifiers is implemented as `MultinomialNB`.

```
from sklearn.naive_bayes import MultinomialNB
MultinomialNB(alpha=1.0, fit_prior=True,
  class_prior=None)
```

Complement Multinomial NB

A *complement Naïve Bayes* (CNB) algorithm is an adaptation of the standard MNB algorithm that is particularly suited for imbalanced data sets.

The procedure for calculating the weights is as follows:

$$\widehat{\theta}_{ji} = \frac{\alpha_i + \sum_{k:y_j \neq j} d_{ij}}{\alpha + \sum_{k:y_j \neq j} \sum_s d_{sj}} \Rightarrow w'_{ji} = \ln \widehat{\theta}_{ji} \Rightarrow w_{ji} = \frac{w'_{ji}}{\sum_k |w'_{jk}|}$$

where the summations are over all documents k not in class j, d_{ij} is either the count or tf-idf value (term frequency-inverse document frequency, see `https://en.wikipedia.org/wiki/Tf-idf`) of term i in document j.

Complement Multinomial NB (cont)

In Python's Sklearn, CNB is implemented as `ComplementNB` and has the form:

```
from sklearn.naive_bayes import *
ComplementNB(alpha=1.0, fit_prior=True,
  class_prior=None, norm=False)
```

There is no CNB in R because it is inspired by text classification rather than having a firm statistical theory.

Bernoulli NB

Bernoulli Naïve Bayes is used when the data is distributed according to multivariate Bernoulli distributions i.e., x_i is a binary value.
The conditional probability is

$$\mathbb{P}(\mathbf{X} = \mathbf{x} | Y = j) = \prod_{i=1}^{p} \theta_{ji}^{x_i} (1 - \theta_{ji}^{x_i})^{1-x_i} \tag{12}$$

It is implemented in Python as `BernoulliNB`:

```
from sklearn.naive_bayes import *
BernoulliNB(alpha=1.0, binarize=0.0,
  fit_prior=True, class_prior=None)
```

R Implementations

Python is more powerful than R when it comes to text processing. Therefore, Python has the variations of multinomial Naive Bayes available in the `sklearn` library but is a bit weak on the mixed categorical and Gaussian naive Bayes model.

In contrast, R has good supports the mixed categorical and Gaussian naïve Bayes models but only supports the standard multinomial NB.

```
library(naivebayes)
naive_bayes(formula, data, prior = NULL, laplace = 0,
  usekernel = FALSE, usepoisson = FALSE,
  subset, na.action = stats::na.pass, ...)
multinomial_naive_bayes(x, y, prior=NULL, laplace=0.5)
```

Other choices: `e1071::naiveBayes` (slow),

`bnlearn::naive.bayes` (which can only handle categorical data),

`klaR::NaiveBayes` (slow), etc.

Outline

Results Interpretation

The generative models are based strongly on the Bayesian statistics philosophy:

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Average Likelihood}}$$

In the formula (1),

- $\mathbb{P}(Y = y)$ is called the **prior probability**;
- $\mathbb{P}(Y = y|\mathbf{X} = \mathbf{x})$ is called the **posterior probability**, i.e. the 'updated' probability based the input $\mathbf{x}$;
- $\mathbb{P}(\mathbf{X} = \mathbf{x}|Y = y)$ is called the **likelihood function**, which encodes human experience on the distribution of inputs associated to the output y.
- $\mathbb{P}(\mathbf{X} = \mathbf{x})$ is a 'scaling' and is constant w.r.t. to the output.

Results Interpretation (cont)

The probabilistic framework that underlie the generative models is the **Maximum a Posteriori (MAP)**:

$$\begin{aligned}\hat{\theta} &= \text{argmax}_\theta P(\theta|D) \\ &= \underset{\theta}{\text{argmax}}\, P((\mathbf{x}_y, y_1), \cdots, (\mathbf{x}_n, y_n) \mid \theta) P(\theta).\end{aligned}$$

The MAP obtains a point estimate of an unobserved quantity θ on the basis of empirical data $(\mathbf{x}_i, y_i)$. It is closely related to the method of MLE, but employs an augmented optimisation objective which incorporates a prior distribution over the quantity one wants to estimate. MAP estimation can therefore be seen as a regularisation of MLE.

Results Interpretation (cont)

As a practical user, the simple differences are:

- *Discriminative learning*: We try to approximate $\mathbb{P}(Y|X)$ using
  - function approximation: (multinomial) logistic regression (LR), ANN;
  - data and distance: kNN
  - information, logic and statistics: decision tree, random forest, etc.
- *Generative learning*: We regard $P(Y|X=x)$ as what happens when the prior $P(Y)$ will change with new data $X=x$ is given. This leads to the modelling of the likelihood $P(X|Y)$:
  - Naive Bayes (NB)
  - Discriminative Analysis, e.g. LDA, QDA

Outline

Models Comparison

LR can only deal with binary classification.

NB can deal with multiclass classification.

When the data with only standardised numeric inputs (i.e. $\sigma_j = 1$) and a binary output, Gaussian NB and logistic regression are both linear classifiers:

$$\frac{P(Y=1)P(X_1|Y=1)\cdots P(X_p|Y=1)}{P(Y=0)P(X_1|Y=0)\cdots P(X_p|Y=0)}$$
$$= \frac{P(Y=1)\exp\{-\frac{1}{2}\left((X_1-\mu_{1,1})^2+\cdots+(X_p-\mu_{p,1})^2\right)\}}{P(Y=0)\exp\{-\frac{1}{2}\left((X_1-\mu_{1,0})^2+\cdots+(X_p-\mu_{p,0})^2\right)\}}$$
$$= \exp\{\beta_0+\beta_1X_1+\cdots+\beta_pX_p\}$$

Models Comparison (cont)

So far, we have been using the frequentist statistics approach to compare models:

- LM models can be compared using F-statistic
- GLM models can be compared using AIC and specific statistics (e.g. χ^2-test for LR)
- LR and NB can't be compared because they do not have a common statistic. Instead, the performance measurements based on the empirical generalisation error are used:
  - cross-validation: when the data size is not large, we can split the data to K blocks and calculate K accuracies (get the average)
  - holdout method: split the historical data to training and testing datasets. Compute the accuracies (and/or sensitivity, etc.) and choose a model based on them.

Models Comparison (cont)

Things become more complex when the class of models has hyperparameters such as kNN (k is the hyperparameter). The frequentist statistics recommends the model selection method below:

- cross-validation: when the data size is not large, we can take out a small portion (e.g. 10%) as an **independent** testing dataset and the remainder for cross-validation on the model by varying the hyperparameter. When the hyperparameter is selected, it is tested againsts the independent testing dataset.
- 3-way holdout: when the data size is large, we can split the historical data into 60% for training, 20% for testing to choose the hyperparameter and 20% for validating the model with the 'best' hyperparameter.

Models Comparison (cont)

In contrast to frequentist statistics's model comparison, in the Bayesian model comparison, prior probabilities are assigned to each of the models, and these probabilities are updated given the data according to Bayes rule.

Given an indexed set of predictive models $M_1, \ldots, M_m$ and associatived prior beliefs in the appropriateness of each model $p(M_i)$, the Bayesian statistics framework uses the model posterior probability

$$p(M_i|D) = \frac{p(D|M_i)p(M_i)}{p(D)}, \quad p(D) = \sum_{i=1}^{m} p(D|M_i)p(M_i)$$

where D is the dataset.

Models Comparison (cont)

When the model M_i is parameterised by θ_i, the model likelihood

$$p(D|M_i)p(M_i) = \int p(D|\theta_i, M_i)p(\theta_i|M_i)d\theta_i.$$

In discrete parameter space, the integral is replaced with summation. Note that the number of parameters $\dim(\theta_i)$ need not be the same for each model.

According to Bayesian statistics, two competing model hypotheses M_i and M_j can be compared using the **Bayes' factor**:

$$\underbrace{\frac{p(M_i|D)}{p(M_j|D)}}_{\text{Posterior odds}} = \underbrace{\frac{p(D|M_i)}{p(D|M_j)}}_{\text{Bayes' factor}} \underbrace{\frac{p(M_i)}{p(M_j)}}_{\text{Prior odds}}.$$

Models Comparison (cont)

For instance, consider a coin tossing problem. We want compare if the coin tossing is biased with M_{biased} and is fair with M_{fair}.

Suppose binomial distribution is used. For $D_1 = 5$ heads & 2 tails,

$$\frac{p(M_{fair}|D_1)}{p(M_{biased}|D_2)} = 1.09$$

Both models are equally OK.

For $D_2 = 50$ heads & 20 tails,

$$\frac{p(M_{fair}|D_2)}{p(M_{biased}|D_2)} = 0.109$$

M_{fair} is only ≈11% of the likelihood of M_{biased}.

Models Comparison (cont)

Computer simulation examples are given at

`https://bookdown.org/kevin_davisross/bayesian-reasoning-and-methods/model-comparison.html`

For theory, see

`https://en.wikipedia.org/wiki/Bayes_factor`

Outline

Case Study 1: Categorical NB

Example 1: Table Q3(d) shows a data set containing 7 observations with 3 categorical predictors, X_1, X_2 and X_3.

Observation	X_1	X_2	X_3	Y
1	C	No	0	Positive
2	A	Yes	1	Positive
3	B	Yes	0	Negative
4	B	Yes	0	Negative
5	A	No	1	Positive
6	C	No	1	Negative
7	B	Yes	1	Positive

Table Q3(d)

Without Laplace smoothing, predict the response, Y for an observation with $X_1 = \text{B}$, $X_2 = \text{Yes}$ and $X_3 = 1$ using Naïve Bayes approach. (5 marks)

Case Study 1 (cont)

Solution: Let '+' denote 'Positive' and '-' denote 'Negative'.

prior, $\mathbb{P}(Y)$	$\mathbb{P}(X_1\|Y)$	$\mathbb{P}(X_2\|Y)$	$\mathbb{P}(X_3\|Y)$	prop, Π	$\hat{Y}$
$\mathbb{P}(+) = \frac{4}{7}$	$\mathbb{P}(\text{B}\|+) = \frac{1}{4}$	$\mathbb{P}(\text{Yes}\|+) = \frac{1}{2}$	$\mathbb{P}(1\|+) = \frac{3}{4}$	0.0536	
$\mathbb{P}(-) = \frac{3}{7}$	$\mathbb{P}(\text{B}\|-) = \frac{2}{3}$	$\mathbb{P}(\text{Yes}\|-) = \frac{2}{3}$	$\mathbb{P}(1\|-) = \frac{1}{3}$	0.0635	$\checkmark$, $-$

Since $\mathbb{P}(Y = -|X) > \mathbb{P}(Y = +|X)$, Y has higher probability to be "Negative".

Case Study 2: Categorical NB

Example 2: Consider the following case given in
`https://machinelearningmastery.com/naive-bayes-tutorial-for-machine-learning/`

Weather	Car	Y
sunny	working	go-out
rainy	broken	go-out
sunny	working	go-out
sunny	working	go-out
sunny	working	go-out
rainy	broken	stay-home
rainy	broken	stay-home
sunny	working	stay-home
sunny	broken	stay-home
rainy	broken	stay-home

Construct the categorical Naïve Bayes model for the above data.

Case Study 2 (cont)

Solution: Let X_1=Weather, X_2=Car. The categorical Naïve Bayes model:

$$\begin{aligned}&\mathbb{P}(Y=j|\mathbf{X}=\mathbf{x})\\ \propto&\mathbb{P}(Y=j)\times\mathbb{P}(X_1=x_1|Y=j)\times\mathbb{P}(X_2=x_2|Y=j)\end{aligned}$$

where Prior, $\mathbb{P}(Y)=\begin{cases}0.5, & Y=out\\ 0.5, & Y=stay\end{cases}$

$$\mathbb{P}(X_1|Y=out)=\begin{cases}\frac{4}{5}, & X_1=sunny\\ \frac{1}{5}, & X_1=rainy\end{cases},$$

$$\mathbb{P}(X_1|Y=stay)=\begin{cases}\frac{2}{5}, & X_1=sunny\\ \frac{3}{5}, & X_1=rainy\end{cases}$$

Case Study 2 (cont)

Solution (cont):

$$\mathbb{P}(X_2|Y = out) = \begin{cases} \frac{4}{5}, & X_2 = working \\ \frac{1}{5}, & X_2 = broken \end{cases},$$

$$\mathbb{P}(X_2|Y = stay) = \begin{cases} \frac{1}{5}, & X_2 = working \\ \frac{4}{5}, & X_2 = broken \end{cases}$$

Case Study 3: Gaussian NB

Example 3: The table below shows the data collected for predicting whether a customer will default on the credit card or not:

customer	balance	student	Default
1	500	No	N
2	1980	Yes	Y
3	60	No	N
4	2810	Yes	Y
5	1400	No	N
6	300	No	N
7	2000	Yes	Y
8	940	No	N
9	1630	No	Y
10	2170	Yes	Y

Case Study 3 (cont)

(a) Compute the probability density of customer with balance 2080, given Default=Y.
(b) Compute the probability of customer who is a student, given Default=Y.
(c) Calculate the “probability density” of default for a student customer with balance 2080 by using the Naïve Bayes assumption.

Case Study 3 (cont)

Note: This question just asks for specific answer without the full model, so we don't need to write the full model.

(a) **Solution**:

$$\mathbb{P}(\texttt{balance} = 2080 \mid \texttt{Default} = Y)$$
$$= \frac{1}{s_Y\sqrt{2\pi}} \exp\left(-\frac{(2080-\mu_Y)^2}{2s_Y^2}\right) = 0.0009162$$

where $\mu_Y = \frac{1980+2810+2000+1630+2170}{5} = 2118$;
$s_Y = 433.7857$

Gaussian Naïve Bayes (cont)

(b) **Solution**: $\mathbb{P}(\texttt{student} = \mathit{Yes} \mid \texttt{Default} = Y) = \dfrac{4}{5}$

(c) **Solution**:

$\mathbb{P}(\texttt{student} = \mathit{Yes} \mid \texttt{Default} = N) = \frac{0}{5} = 0$

$$
\begin{aligned}
&\mathbb{P}(\texttt{Default} = Y \mid \texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes}) \\
=&\frac{\mathbb{P}(\texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes} \mid \texttt{Default} = Y)\mathbb{P}(\texttt{Default} = \texttt{Y})}{\mathbb{P}(\texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes}) =: \mathbb{P}(\dots)} \\
=&\frac{\mathbb{P}(\texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes} \mid \texttt{Default} = Y)\mathbb{P}(\texttt{Default} = \texttt{Y})}{\mathbb{P}(\dots \mid \texttt{Default} = Y)\mathbb{P}(\texttt{Default} = \texttt{Y}) + \mathbb{P}(\dots \mid \texttt{Default} = N)\mathbb{P}(\texttt{Default} = \texttt{N})} \\
=&\frac{0.0009162 \times \frac{4}{5} \times \frac{5}{10}}{0.0009162 \times \frac{4}{5} \times \frac{5}{10} + \mathbb{P}(\texttt{balance} = 2080 | \texttt{Default} = N) \times 0 \times \frac{5}{10}} = 1
\end{aligned}
$$

Case Study 3 (cont)

Remark on **Example** 3: Let x_1=balance, x_2=student, y=Default. The Naive Bayes Model is

$$h_D(x_1, x_2) = \underset{j}{\operatorname{argmax}}\, \mathbb{P}(x_1|y=j)\mathbb{P}(x_2|y=j)\mathbb{P}(y=j).$$

where the prior $\mathbb{P}(y) = \begin{cases} 0.5 & y = N \\ 0.5 & y = Y \end{cases}$

$$\mathbb{P}(x_2|y=N) = \begin{cases} 1 & x_2 = No \\ 0 & x_2 = Yes \end{cases}$$

$$\mathbb{P}(x_2|y=Y) = \begin{cases} 1/5 & x_2 = No \\ 4/5 & x_2 = Yes \end{cases}$$

Case Study 3 (cont)

Remark on **Example** 3 (cont):

$$\mathbb{P}(x_1|y=N) = \frac{1}{\sqrt{2\pi}(533.6666)} \exp\left\{-\frac{(x_1-640)^2}{2(533.6666)^2}\right\}$$

$$\mathbb{P}(x_1|y=Y) = \frac{1}{\sqrt{2\pi}(433.7857)} \exp\left\{-\frac{(x_1-2118)^2}{2(433.7857)^2}\right\}$$

Case Study 4: Gaussian NB

Example 4:
A more efficient marketing strategy can be achieved by targeting the customers who have higher probability to complete a purchase. Hence, you have been asked to predict whether a customer will buy the product based on their demographic data such as age, race, gender and income. Table Q3(c) shows the data collected from previous records.

Case Study 4 (cont)

Cust.	Age	Race	Gender	Income	Result
1	52	Indian	Male	RM 11,500	Not Buy
2	22	Chinese	Female	RM 6,500	Buy
3	30	Chinese	Male	RM 8,000	Buy
4	26	Malay	Male	RM 8,500	Buy
5	27	Indian	Female	RM 6,500	Buy
6	32	Chinese	Female	RM 9,500	Not Buy
7	33	Indian	Male	RM 4,000	Not Buy
8	50	Malay	Female	RM 10,000	Buy
9	31	Chinese	Female	RM 5,500	Buy
10	27	Malay	Male	RM 9,200	Not Buy

Table Q3(c)

Case Study 4 (cont)

(i) State an assumption used in Naïve Bayes approach. (1 mark)

(ii) Using Naïve Bayes approach without Laplace smoothing, predict whether a Malay female customer, aged 29, with income RM7,800, will buy the product. (9 marks)

Case Study 5: Software Support

Gaussian Naïve Bayes (Classifier) is available in Python as `GaussianNB` of the form:

```
from sklearn.naive_bayes import GaussianNB
GaussianNB(priors=None, var_smoothing=1e-09)
```

All the above mentioned naïve Bayes models are available in R except the complement NB. R provides unified functions such as `naivebayes::naive_bayes`, `e1071::naiveBayes`, `bnlearn::naive.bayes` (which can only handle categorical data), `klaR::NaiveBayes`.

```
naive_bayes(formula, data, prior = NULL, laplace = 0,
  usekernel = FALSE, usepoisson = FALSE,
  subset, na.action = stats::na.pass, ...)
```

Case Study 6: Categorical NB with Laplace Smoothing

An issue faced by a Naïve Bayes classifier with "discrete" data is the numerator in (9) being zero, i.e. $n_{X=c,Y=j} = 0$. In this case, the posterior probability will become zero regardless of the value of other density functions and the Naïve Bayes classifier will fail.

Case Study 6 (cont)

Example 5: By using the data from **Example** 3, perform the following tasks.

(a) Compute the probability density of customer with balance 2080, given Default=N.

(b) Compute the probability of customer who is a student, given Default=N.

(c) Calculate the "probability density" of non-default for a student customer with balance 2080 by using NB model without Laplace smoothing.

(d) Redo part (b) and (c) with Laplace smoothing.

Case Study 6 (cont)

(a) Solution: $\mathbb{P}(\texttt{balance} = 2080 \mid \texttt{Default} = N) = \frac{1}{s_N\sqrt{2\pi}} \exp(-\frac{(2080-\mu_N)^2}{2s_N^2}) = 1.9616 \times 10^{-5}$
where $\mu_N = \frac{500+60+1400+300+940}{5} = 640; \quad s_N = 533.6666$
(b) Solution:
$\mathbb{P}(\texttt{student} = \mathit{Yes} \mid \texttt{Default} = N) = \frac{0}{5} = 0$
(c) Solution:

$$\begin{aligned}&\mathbb{P}(\texttt{Default} = N \mid \texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes})\\ &= \frac{1.9616 \times 10^{-5} \times 0 \times \frac{5}{10}}{\mathbb{P}(\texttt{balance} = 2080,\ \texttt{student} = \mathit{Yes})} = 0.\end{aligned}$$

Case Study 6 (cont)

Remark: This situation is normally happened to the categorical variable. To avoid the stated problem, *Laplace smoothing* or `https://en.wikipedia.org/wiki/Additive_smoothing` is applied to the Naïve Bayes classifier. Laplace smoothing modified the density function of categorical variable by adding α (by default, $\alpha = 1$) to each variable per class:

$$\mathbb{P}(X = x_i | Y = k) = \frac{n_{X=x_i;Y=k} + \alpha}{n_{Y=k} + d\alpha} \tag{13}$$

where d is the number of classes in the categorical variable X.

Case Study 6 (cont)

(d) Redo part (b) and (b) by applying Laplace smoothing:
The "continuous" variable is the same:

$$\mathbb{P}(\texttt{balance} = 2080 \mid \texttt{Default} = N) = 1.9616 \times 10^{-5}$$

The categorical variable needs the Laplace smoothing ($\alpha = 1$, $d = 2$ for $\texttt{student}$=Yes or No)

$$\mathbb{P}(\texttt{student} = Yes \mid \texttt{Default} = N) = \frac{0+1}{5+2}$$

Case Study 6 (cont)

Therefore,

$$
\begin{aligned}
&\mathbb{P}(\texttt{Default} = N \mid \texttt{balance} = 2080,\ \texttt{student} = Yes)\\
=&\frac{\mathbb{P}(2080, Yes \mid N)\mathbb{P}(N)}{\mathbb{P}(2080, Yes \mid N)\mathbb{P}(N) + \mathbb{P}(2080, Yes \mid Y)\mathbb{P}(Y)}\\
=&\frac{1.9616 \times 10^{-5} \times \frac{1}{7} \times \frac{5}{10}}{1.9616 \times 10^{-5} \times \frac{1}{7} \times \frac{5}{10} + 0.000916154 \times \frac{5}{7} \times \frac{5}{10}}\\
=&\frac{1.401151 \times 10^{-6}}{1.401151 \times 10^{-6} + 0.0003271979} = 0.004264014
\end{aligned}
$$

Outline

Lab Practice on Classification Method

`prac_cls2.R` (Naive Bayes)

No programming in Final but you need to interpret and to use the output of the trained Naive Bayes model given by R.