-- **Quick recap**

We are in a **supervised** classification setting and we want to come up with a **classifier** $h$ which has minimimal classification error, that is: we want to **minimize**

$$\mathbb{E}_{XY}\left[\mathbf{1}_{h(X)\neq Y}\right] = \mathbb{P}(h(X) \neq Y)$$

-- **Bayes classifier**

The **best classifier** we can hope for is the Bayes classifier, which is written as:

$$h_{\text{Bayes}}(x) = \underset{k}{\text{argmax}} \quad \mathbb{P}(Y = k \mid X = x)$$

> Example: suppose we have $Y \in \{0, 1, 2\}$. Then the classification procedure for a given $x_i$ is:
>
> - Calculate $\mathbb{P}(Y = k \mid X = x_i)$ for each possible $k$
> - Check for which $k_{\text{max}}$ the probability is the largest (remembering that they all sum to one)
> - Assign $y_i = k_{\text{max}}$
>
> Note that although the output is always just the class of the data points, we also have **probabilistic information** about the choice of labels. This means that we can have a notion of "degree of certitude" of assigning one class of the other.
>
> For instance, which classification is more certain?
>
> - $\mathbb{P}(Y = 1 \mid X = x_i) = 0.90$ and $\mathbb{P}(Y = 0 \mid X = x_i) = 0.10$
> - $\mathbb{P}(Y = 1 \mid X = x_i) = 0.52$ and $\mathbb{P}(Y = 0 \mid X = x_i) = 0.48$
>
> Having this extra nuance is very helpful for the end user.

We would only have access to the Bayesian classifier **if we knew** the true distribution of the data, which is rarely the case. As such, people have proposed many ways of **approximating** it.

We have seen two of such approximations last time: Logistic Regression and LDA

-- **Why not linear regression?**

Suppose we want to predict medical conditions of a patient based on his/her symptoms. Let's say that based on a set of features $X_1, \ldots, X_p$, the patient can be assigned to one of three diagnoses: stroke, drug overdose, epilepsy. We can translate these conditions into different values for label $Y$:

$$Y = \begin{cases} 1 & \text{if } \texttt{stroke} \\ 2 & \text{if } \texttt{drug overdose} \\ 3 & \text{if } \texttt{epilepsy} \end{cases}$$

If we simply use a linear regression model to classify data, as in

$$\hat{Y} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

then we have an implicit notion of **ordering** between the conditions: the difference between `stroke` and `drug overdose` is exactly the same as the difference between `drug overdose` and `epilepsy`, which does not really make sense in practice. In summary, there is **no natural way to convert** a qualitative response variable with more than two levels into a quantitative response that is ready for linear regression.

When working with **binary** responses we don't have this problem anymore. However, we have to ensure that the values of $\hat{Y}$ will be between 0 and 1, which is not always easy nor natural to do.

---

-- **Logistic regression**

We approximate directly $\mathbb{P}(Y = k \mid X = x)$ and get

$$h_{\text{logreg}}(x) = \underset{k}{\text{argmax}} \quad p_\beta(x) \quad \text{where} \quad p_\beta(x) = \frac{\exp\left(\beta^T x\right)}{1 + \exp\left(\beta^T x\right)}$$

with $\beta = [\beta_0, \ldots, \beta_p]^T$ and $x = [1, x^1, \ldots, x^p]^T$

The parameters $\beta$ are estimated via MLE = Maximum Likelihood Estimation (see TD 2)

-- **LDA**

We use the fact that $\mathbb{P}(Y = k \mid X = x) \propto \mathbb{P}(X = x \mid Y = k) \times \mathbb{P}(Y = k)$ and approximate each factor

$$h_{\text{LDA}}(x) = \underset{k}{\text{argmax}} \quad p_k(x) \times \mathbb{P}(Y = k)$$

Where the $p_k(x) = \mathcal{N}(\mu_k, C_k)$ with $C_k = C$. The parameters in the gaussians are easily estimated from the data and $\mathbb{P}(Y = k)$ is simply the fraction of points belonging to class $k$.

---

Note that $p_k(x)$ could have been modeled as anything we wanted. In LDA, we make a **strong assumption** regarding the distribution of the data and say that they follow a Gaussian. This translates into estimating just a **few parameters** from the dataset using well known estimators: $kp$ parameters for the means and $kp^2$ parameters for the covariances.

I will present now another classifier with different assumptions.

-- **Naïve Bayes classifier**

Instead of making **assumptions** about the **family** of distributions for the $f_k(x)$ we say that:

> For each class $k$ the $p$ predictors are independent.

Mathematically, we have that

$$p(x^1, x^2, \ldots, x^p \mid Y = k) = p(x^1 \mid Y = k) \times \cdots \times p(x^p \mid Y = k) = \prod_{i=1}^{p} p(x^i \mid Y = k)$$

Note that by using this assumption we model only the **marginal** probabilities of the predictors.

We don't care anymore about all of the **joint** information between them.

- Note that if we fix each of the marginals to be a Gaussian distribution, then we have $kp$ parameters for the means and $kp$ parameters for the standard deviations of the predictors. This boils down to $2kp$ parameters to estimate versus the $(1 + p)kp$ parameters from LDA.

Do we actually **believe** that this assumption is true in the data? **Not really**.

**However**, when $n \approx p$ (or even for $n < p$) there is simply not enough data for us to effectively estimate a joint distribution for the predictors.

Once we are OK with using the Naive Bayes assumption, we have that

$$h_{\text{nB}}(x) = \underset{k}{\text{argmax}} \quad \mathbb{P}(Y=k) \times \prod_{i=1}^{p} p_k(x^i)$$

**Remark**: Note that we don't care about the normalization constant here because it is the same for all values of $k$ and we are only interested in taking the maximum among them.

-- **How to implement it?**

Considering a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i \leq n}$ we can easily implement the conditional density approximations by considering different cases for them:

- If $x^j$ is a **quantitative** variable (i.e. $x^j \in \mathbb{R}$) then we can assume $p_k(x^j) = \mathcal{N}(\mu_{jk}, \sigma^2_{jk})$ and the parameters of the gaussian distribution can be easily estimated as

  $$\mu_{jk} = \frac{1}{\sum_i \mathbf{1}_{\{y_i=k\}}} \sum_i \mathbf{1}_{\{y_i=k\}}\, x_i^j \quad \text{and} \quad \sigma^2_{jk} = \frac{1}{\sum_i \mathbf{1}_{\{y_i=k\}}} \sum_i \mathbf{1}_{\{y_i=k\}} (x_i^j - \mu_{jk})^2$$

  When all variables are considered as gaussian, we say that we are using a Gaussian Naive Bayes
- If $x^j \in \{1, \dots, L\}$, then we can assume that $p_k(x^j)$ is a **Categorical distribution** over $L$ choices with

  $$p_k(x^j) = \mathcal{C}(p^j_{k,1}, \dots, p^j_{k,L}) \quad \text{and} \quad p^j_{k,\ell} = \frac{1}{\sum_i \mathbf{1}_{\{y_i=k\}}} \sum_i \mathbf{1}_{\{y_i=k\}} \times \mathbf{1}_{\{x_i^j=\ell\}}$$

It is possible to show that the Naive Bayes classifier **can be directly related** to several other classifiers, including the logistic regression and the LDA. We won't do it here, but I've added some references in the Chamilo for the course.

OK, I've showed you the Naive Bayes classifier and now I'm going to show you an important application of this kind of method: text classification. However, to do that, I will **first setup the problem** for you.

---

-- **Text classification**

When working with a database containing several text documents, we might be interested in:

- Detecting whether an e-mail should be considered as SPAM or not (HAM)
- Detect if a news article is fake news or not
- Sentiment analysis of Tweets about political candidates

We define then the problem of **text classification** as follows:

- We are given a description $d \in \mathbb{X}$ of a document, where $\mathbb{X}$ Is the **document space**
  - $\mathbb{X}$ is some type of high-dimensional space
- A fixed set of classes $\mathbb{C} = \{c_1, \dots, c_J\}$
  - The classes are human defined

We want to build a classifier $\gamma : \mathbb{X} \to \mathbb{C}$ that takes document descriptions into labels.

> For instance, we might have: $d$ = "Beijing joins the World Trade Organization" and $c$ = "China"

As we will see later, the dimensionality of the feature vector $\mathbb{X}$ can be huge, so it is natural to want to use a Naive Bayes approach for building the classifier $\gamma$. With that said, the probability of a given document $d$ to belong to a category $k$ is:

$$\mathbb{P}(c = k \mid d) \propto \mathbb{P}(c = k) \times \prod_{1 \leq k \leq n_d} \mathbb{P}(t_k \mid c)$$

where $\mathbb{P}(t_k \mid c)$ is the probability of term $t_k$ happening in a document of class $c$.

> For instance,
>
> - The probability of the word "Gun" appearing in a document from class "James Bond"
> - Or "Spaguetti" appearing in a document from class "Linear Algebra"

The list $\langle t_1, \ldots, t_{n_d} \rangle$ are the **tokens** representing document $d$, with $n_d$ indicating how many of them we have. We will discuss how to define these tokens in another lecture. For now, just consider them as **given**.

For a given trainind dataset, we estimate the probabilities $\mathbb{P}(t \mid c)$ as the relative frequency of term $t$

$$\mathbb{P}(t \mid c = k) = \frac{T_{kt}}{\sum_{t' \in V} T_{kt'}}$$

where

- $V$ is the **vocabulary of all words** that are present in the training set
- $T_{kt}$ is the number of occurences of $t$ in training documents from class $k$, including **multiple occurences** of a term in a document.

Note that we assume **positional independence** in our analysis, i.e. we count all occurences of $t$ in a document regardless of their position.

The prior distributions are calculated as

$$\mathbb{P}(c = k) = \frac{N_k}{N}$$

where $N_k$ is the number of documents in class $k$ and $N$ is the total number of documents.

Although the estimators look OK, they have an **important problem**.

> Example:
>
> Suppose I want to classify e-mails between SPAM or HAM, and for the sake of simplicity their prior distributions are 50% and 50%. Let's say we have an e-mail whose representation is $\langle w_1, \ldots, w_n \rangle$ and for which our trained classifier gives:
>
> $$\begin{aligned} \mathbb{P}(\text{HAM} \mid w_1, \ldots, w_n) &= 0.5 \times \prod_{i=1}^{n} \mathbb{P}(w_i \mid \text{HAM}) &= 0.90 \\ \mathbb{P}(\text{SPAM} \mid w_1, \ldots, w_n) &= 0.5 \times \prod_{i=1}^{n} \mathbb{P}(w_i \mid \text{SPAM}) &= 0.10 \end{aligned}$$

> Well, in this case it is easy to say that we should classify the e-mail as HAM.
>
> Now suppose that we have another e-mail with $\langle w_1, \ldots, w_n, w_{n+1} \rangle$, where $w_{n+1}$ was not part of the initial vocabulary used to train the classifier. As such, we have that
>
> $$\mathbb{P}(w_{n+1} \mid \mathrm{HAM}) = \mathbb{P}(w_{n+1} \mid \mathrm{SPAM}) = 0$$
>
> and, therefore, the posterior distributions collapse to zero, since
>
> $$\begin{aligned} \mathbb{P}(\mathrm{HAM} \mid w_1, \ldots, w_n) &= 0.5 \times \prod_{i=1}^{n} \mathbb{P}(w_i \mid \mathrm{HAM}) \times \mathbb{P}(w_{n+1} \mid \mathrm{HAM}) = 0 \\ \mathbb{P}(\mathrm{SPAM} \mid w_1, \ldots, w_n) &= 0.5 \times \prod_{i=1}^{n} \mathbb{P}(w_i \mid \mathrm{SPAM}) \times \mathbb{P}(w_{n+1} \mid \mathrm{SPAM}) = 0 \end{aligned}$$
>
> Conclusion: despite the strong classification of the initial e-mail as HAM, the appeareance of just a new word into the game makes the posteriors collapse to zero. **How do we solve this?**

-- **Laplace smoothing**

The way to avoid this collapse when new terms appear is to use a shrinkage procedure (or regularization, depending of how you interpret things) as per

$$\mathbb{P}(t \mid c = k) = \frac{T_{kt} + 1}{\sum_{t' \in V}(T_{kt'} + 1)} = \frac{T_{kt} + 1}{\sum_{t' \in V} T_{kt'} + |V|}$$

where we simply add one on the counting for each term. Although mysterious and rather magical, this smoothing can be seen interpreted under a Bayesian optics:

- We start with a prior that the probabilities $\mathbb{P}(t \mid c = k)$ are uniform among all different elements of $V$
- Given the dataset, we update these probabilities to adapt to what we observe. This is our posterior.

-- **Example with actual text**

| | docID | words in document | in $c = China$? |
|---|---|---|---|
| training set | 1 | Chinese Beijing Chinese | yes |
| | 2 | Chinese Chinese Shanghai | yes |
| | 3 | Chinese Macao | yes |
| | 4 | Tokyo Japan Chinese | no |
| test set | 5 | Chinese Chinese Chinese Tokyo Japan | ? |

> We have two classes: China and not China, so we will represent them as 0 and 1
>
> The prior probabilities are $\mathbb{P}(c = 0) = 1/4$ and $\mathbb{P}(c = 1) = 3/4$
>
> The likelihoods are:

$$\begin{aligned}
\mathbb{P}(\text{Chinese} \mid c=1) &= \frac{5+1}{8+6} = 6/14 \\
\mathbb{P}(\text{Tokyo} \mid c=1) &= \frac{0+1}{8+6} = 1/14 \\
\mathbb{P}(\text{Japan} \mid c=1) &= \frac{0+1}{8+6} = 1/14 \\
\mathbb{P}(\text{Chinese} \mid c=0) &= \frac{1+1}{3+6} = 2/9 \\
\mathbb{P}(\text{Tokyo} \mid c=0) &= \frac{1+1}{3+6} = 2/9 \\
\mathbb{P}(\text{Japan} \mid c=0) &= \frac{1+1}{3+6} = 2/9
\end{aligned}$$

We then have that

$$\begin{aligned}
\mathbb{P}(c=1 \mid d_5) &\propto \mathbb{P}(c=1) \times \Big(\mathbb{P}(\text{Chinese} \mid c=1)\Big)^3 \times \mathbb{P}(\text{Japan} \mid c=1) \times \mathbb{P}(\text{Tokyo} \mid c=1) \\
&\propto 3/4 \ \times \ (3/7)^3 \ \times \ 1/14 \ \times \ 1/14 \approx 0.0003 \\
\mathbb{P}(c=0 \mid d_5) &\propto \mathbb{P}(c=0) \times \Big(\mathbb{P}(\text{Chinese} \mid c=0)\Big)^3 \times \mathbb{P}(\text{Japan} \mid c=0) \times \mathbb{P}(\text{Tokyo} \mid c=0) \\
&\propto 1/4 \ \times \ (2/9)^3 \ \times \ 2/9 \ \times \ 2/9 \approx 0.0001
\end{aligned}$$

The classifier will choose class $c = 1$, i.e. the document is related to China.

The classifier that we have used so far is based on $t_k$, which is the **fractions of tokens** in documents of class $k$ that contain a certain term $t$. This is what people call **multinomial** naive Bayes.

A different approach would be to consider the **fraction of documents** of class $k$ that contain term $t$. This is what we call the **Bernoulli** naive Bayes

-- The Bernoulli model

In this approach, we process data in a **binary** way: each document is represented as a feature vector with $|V|$ dimensions (the size of the vocabulary) and it is filled with zeros and ones. The value 1 means the given token occurs in the document, value 0 means it does not.

We have that

$$\mathbb{P}(c=k \mid d) \propto \mathbb{P}(c=k) \times \prod_{i=1}^{|V|} \mathbb{P}(e_i = 1 \mid x = k)^{e_i} \times \Big(1 - \mathbb{P}(e_i = 1 \mid x = k)\Big)^{1-e_i}$$

where the conditional probabilities are estimated as

$$\mathbb{P}(e_i = 1 \mid x = k) = \frac{D_{ki} + 1}{D_k + 2}$$

where $D_{ki}$ is the **number of documents** in the training set which have the token $i$ and belong to class $k$, $D_k$ is the total number of documents in class $k$, and the +1 and +2 come from Laplace Smoothing.

> When classifying a document, the Bernoulli model uses binary occurence information, **ignoring the number** of occurences. The multinomial model **keeps track** of multiple occurences.

We can apply the Bernoulli model to the same example from before.

> The likelihoods are given by:
>
> $$\begin{aligned}
\mathbb{P}(\text{Chinese} \mid c = 1) &= \frac{3+1}{3+2} = 4/5 \\
\mathbb{P}(\text{Japan} \mid c = 1) &= \frac{0+1}{3+2} = 1/5 \\
\mathbb{P}(\text{Tokyo} \mid c = 1) &= \frac{0+1}{3+2} = 1/5 \\
\mathbb{P}(\text{Macao} \mid c = 1) &= \frac{1+1}{3+2} = 2/5 \\
\mathbb{P}(\text{Beijing} \mid c = 1) &= \frac{1+1}{3+2} = 2/5 \\
\mathbb{P}(\text{Shanghai} \mid c = 1) &= \frac{1+1}{3+2} = 2/5 \\
\mathbb{P}(\text{Chinese} \mid c = 0) &= \frac{1+1}{1+2} = 2/3 \\
\mathbb{P}(\text{Japan} \mid c = 0) &= \frac{1+1}{1+2} = 2/3 \\
\mathbb{P}(\text{Tokyo} \mid c = 0) &= \frac{1+1}{1+2} = 2/3 \\
\mathbb{P}(\text{Macao} \mid c = 0) &= \frac{0+1}{1+2} = 1/3 \\
\mathbb{P}(\text{Beijing} \mid c = 0) &= \frac{0+1}{1+2} = 1/3 \\
\mathbb{P}(\text{Shanghai} \mid c = 0) &= \frac{0+1}{1+2} = 1/3
\end{aligned}$$
>
> We have then that $\mathbb{P}(c = 1 \mid d_5) \approx 0.005$ and $\mathbb{P}(c = 0 \mid d_5) \approx 0.022$.
>
> So the classifier now considers that the document is not from class China.

**Comment**: In general, one can not be sure whether the Bernoulli or the Multinomial model for Naive Bayes will yield better results. Each one of them tackles the problem of text classification in a slightly different manner and can behave differently.

In our next class, I will be talking about **performance metrics for classifiers**, which is naturally an important topic when talking about classification. that and will close our chapter on classification.

Then, we will talk a bit about text mining and how these tokens and vocabulary that we were talking about can be extracted from a set of documents.