CS340 Machine learning
Naïve Bayes classifiers
Document classification

• Let $Y \in \{1, \ldots, C\}$ be the class label and $x \in \{0,1\}^d$

• eg $Y \in \{\text{spam, urgent, normal}\}$,
  
  $x_i = I(\text{word i is present in message})$

• Bag of words model

$$\begin{bmatrix}
1 & 2 & 3 & 4 & 5 & 6 & 7
\end{bmatrix}$$

Words = \{john, mary, sex, money, send, meeting, unk\}

“John sent money to Mary after the meeting about money”
  \hspace{1cm} \text{Stop word removal}

“john sent money mary after meeting about money”
  \hspace{1cm} \text{Tokenization}

$$\begin{bmatrix}
1 & 7 & 4 & 2 & 7 & 6 & 7 & 4
\end{bmatrix}$$

\text{Word counting}

$$[1, 1, 0, 2, 0, 1, 3]$$

\hspace{1cm} \text{Thresholding (binarization)}

$$[1, 1, 0, 1, 0, 1, 1]$$
Bayes rule for classifiers

\[ p(y = c | x) = \frac{p(x | y = c) p(y = c)}{\sum_{c'} p(x | y = c') p(y = c')} \]

- Class posterior
- Class-conditional density
- Class prior
- Normalization constant
Class conditional density $p(x|y=c)$

- What is the probability of generating a $d$-dimensional feature vector for each class $c$?
- Let us assume we generate each feature independently (naive Bayes assumption)

$$p(x|y = c) = \prod_{i=1}^{d} p(x_i|y = c)$$

- E.g., prob of seeing “send” is assumed to be independent of seeing “money” given that we know this is a spam email
- Allows us to use 1 dimensional density models $p(x_i|y)$. Can combine features of different types.
Count features (multivariate Poisson)

- Suppose $X_i \in \{0,1,2,\ldots\}$ counts the number of times word $i$ occurs.

- A suitable class-conditional density is

$$X_i \mid y = c \sim Poi(\lambda_{ic})$$

- The likelihood is

$$p(x \mid y = c) \propto \prod_{i=1}^{d} e^{-\lambda_{ic}} \lambda_{ic}^{x_i}$$

\[\text{Poi}(\lambda=0.100)\]

\[\text{Poi}(\lambda=1.000)\]

\[\text{Poi}(\lambda=10.000)\]

\[\text{Poi}(\lambda=20.000)\]
Count features (multinomial model)

- Let \((X_1, \ldots, X_d) \mid y = c, N \sim \text{Mult}(\theta_c, N)\)

\[
P(x_1, \ldots, x_d \mid \theta_c, N) = \binom{N}{x_1 \ldots x_d} \prod_{i=1}^{d} \theta_{ic}^{x_i}
\]

\(X_i\)'s no longer conditionally independent since \(\sum_i x_i = N\)

\[
= \frac{N!}{x_1!x_2!\ldots x_d!} \prod_{i=1}^{d} \theta_{ic}^{x_i}
\]

We also require \(\sum_i \theta_i = 1\).

\[
= (\sum_i x_i)! \prod_i \frac{\theta_{ic}^{x_i}}{x_i!}
\]

where \(N = \sum_i x_i\) is the number of words in the document (assumed independent of \(Y = c\)).
Binary features (multivariate Bernoulli)

- Let $X_i|y=c \sim \text{Ber}(\theta_{ic})$ so $p(X_i=1|y=c) = \theta_{ic}$

\[
p(x|y = c) = \prod_{i=1}^{d} \theta_{ic}^{I(x_i=1)} (1 - \theta_{ic})^{I(x_i=0)}
\]
Which class-conditional density?

• For document classification, the multinomial model is found to work best. However, we will mostly focus on the multivariate Bernoulli (binary features) model, for simplicity.

• We can easily handle features of different types, eg $x_1 \in \{0,1\}$, $x_2 \in \mathbb{R}$, $x_3 \in \mathbb{R}^+$, $x_4 \in \{0,1,2,\ldots\}$

• We can use mixtures of Gaussians/ Gammas/ Bernoullis etc. to get more accurate models (see later).
Class prior

- Let \((Y_1, \ldots, Y_C) \sim \text{Mult}(\pi, 1)\) be the class prior.

\[
P(y_1, \ldots, y_C | \pi) = \prod_{c=1}^{C} \pi_c^{I(y_c=1)} \quad \sum_{c=1}^{C} \pi_c = 1
\]

- Since \(\sum_c Y_c = 1\), only one bit can be on. This is called a 1-of-C encoding. We can write \(Y = c\) instead.

\[Y = 2 \equiv (Y_1, Y_2, Y_3) = (0, 1, 0)\]

\[
P(y | \pi) = \prod_{c=1}^{C} \pi_c^{I(y=c)} = \pi_y
\]

- e.g., \(p(\text{spam}) = 0.7, p(\text{urgent}) = 0.1, p(\text{normal}) = 0.2\)
Class posterior

- **Bayes rule**
  \[
  p(y = c|x) = \frac{p(y = c)p(x|y = c)}{p(x)} = \frac{\pi_c \prod_{i=1}^{d} \theta_{ic}^{I(x_i = 1)} (1 - \theta_{ic})^{I(x_i = 0)}}{p(x)}
  \]

- Since numerator and denominator are very small number, use logs to avoid underflow
  \[
  \log p(y = c, x) = \log \pi_c + \sum_{i=1}^{d} I(x_i = 1) \log \theta_{ic} + I(x_i = 0) \log (1 - \theta_{ic}) - \log p(x)
  \]

- **How compute the normalization constant?**
  \[
  \log p(x) = \log \left[ \sum_c p(y = c, x) \right] = \log \left[ \sum_c \pi_c f_c \right]
  \]
Log-sum-exp trick

- Define

\[
\log p(x) = \log \left[ \sum_c \pi_c f_c \right]
\]

\[
b_c = \log \pi_c + \log f_c
\]

\[
\log p(x) = \log \sum_c e^{b_c} = \log \left[ (\sum_c e^{b_c}) e^{-B} e^B \right]
\]

\[
= \log \left[ (\sum_c e^{b_c-B}) e^B \right] = \left[ \log(\sum_c e^{b_c-B}) \right] + B
\]

\[
B = \max_c b_c
\]

\[
\log(e^{-120} + e^{-121}) = \log \left( e^{-120}(e^0 + e^{-1}) \right) = \log(e^0 + e^{-1}) - 120
\]

- In Matlab, use Minka’s function \( S = \text{logsumexp}(b) \)

\[
\text{logjoint} = \log(\text{prior}) + \text{counts} \times \log(\theta) + (1-\text{counts}) \times \log(1-\theta);
\]

\[
\log p(y=c, x) = \log p(y=c/\infty)
\]

\[
\log p(y=c/\infty)
\]
• Suppose the value of $x_1$ is unknown
• We can simply drop the term $p(x_1|y=c)$.

\[ p(y = c|x_{2:d}) \propto p(y = c, x_{2:d}) \]
\[ = \int p(y = c, x_1, x_{2:d}) dx_1 \]
\[ = p(y = c)[\int p(x_1|y = c)dx_1]\prod_{i=2}^{d} p(x_i|y = c) \]
\[ = p(y = c)\prod_{i=2}^{d} p(x_i|y = c) \]

• This is a big advantage of generative classifiers (which specify $p(x|y=c)$) over discriminative classifiers (that learn $p(y=c|x)$ directly).
Parameter estimation

• So far we have assumed that the parameters of $p(x|y=c)$ and $p(y=c)$ are known.

• To estimate $p(y=c)$, we can use MLE or MAP or fully Bayesian estimation of a multinomial, eg

$$\hat{\pi}_{c}^{MAP} = \frac{N_c + \alpha_c - 1}{\sum_{c'}(N_c + \alpha_{c'} - 1)}$$

• We can then use the plug-in approximation

$$p(y|D) \approx \prod_{c} \hat{\pi}_{c}^{I(y=c)}$$

or the posterior predictive

$$p(y|D) = \prod_{c} \bar{\pi}_{c}^{I(y=c)}$$
Posterior predictive for a multinomial

- Recall that, for the Dirichlet-multinomial model, the posterior predictive is equivalent to plugging in the posterior mean parameters, since

\[
p(y = c|D) = \int p(y = c|\pi_c)p(\pi_c|D)d\pi_c \\
= \int \pi_c \text{Dir}(\pi|\alpha'_c, \ldots, \alpha'_c) d\pi_c \\
= \bar{\pi}_c = \frac{N_c + \alpha_c}{N + \alpha}
\]
MLE for Bernoulli features

• We will assume the params for $p(x|y=c)$ are independent for each class.

• Since we treat each feature separately, we just count how many times word $j$ occurred in documents of class $c$, and divide by the number of documents of class $c$

\[
\hat{\theta}_{jc} = \frac{\sum_{i: y_i = c} \sum_{w \in i} I(w = j)}{\sum_{i: y_i = c} 1} = \frac{N_{jc}}{N_c}
\]

• We can easily add priors to regularize this.
Class conditional densities

- At test time, we can either use a plug-in approximation
  
  \[
  p(x|y = c, D) \approx \prod_j \hat{\theta}_{jc}^{I(x_j=1)} (1 - \hat{\theta}_{jc})^I(x_j=0)
  \]

  or the exact posterior predictive

  \[
  p(x|y = c, D) = \prod_j \bar{\theta}_{jc}^{I(x_j=1)} (1 - \bar{\theta}_{jc})^I(x_j=0)
  \]
Naïve Bayes with real-valued features

- If $X_j \in \mathbb{R}$, we can use Gaussian class conditional densities $X_j | y = c \sim \mathcal{N}(\mu_{jc}, \sigma_{jc})$

$$p(x | y = c) = \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma_{jc}^2}} \exp\left(-\frac{1}{2\sigma_{jc}^2}(x_j - \mu_{jc})^2\right)$$
Plug-in approximation

- We can compute MLEs for each feature $j$ and class $c$ separately

\[
\hat{\theta}_{jc} = (\hat{\mu}_{jc}, \hat{\sigma}_{jc}^2)
\]

\[
\hat{\mu}_{jc} = \frac{1}{n_c} \sum_{i:y_i=c} x_{ij} = \bar{x}_{jc}
\]

\[
\hat{\sigma}_{jc}^2 = \frac{1}{n_c} \sum_{i:y_i=c} (x_{ij} - \bar{x}_{jc})^2
\]

- Then we can use a plug-in approximation

\[
p(y = c|x_1:d, D) \propto p(y = c|D) \prod_{j=1}^{d} p(x_j|y = c, D)
\]

\[
\approx p(y = c|\hat{\pi}) \prod_{j=1}^{d} p(x_j|y = c, \hat{\theta}_{jc})
\]

\[
= \hat{\pi}_c \prod_j \mathcal{N}(x_j|\hat{\mu}_{jc}, \hat{\sigma}_{jc}^2)
\]
• If we use conjugate priors, it is simple to derive a fully Bayesian solution: we just update the hyperparameters for each feature \( j \) and class \( c \), and then use the predictive distribution, which is a student \( T \) distribution.

\[
p(y = c|x_1:d, D) \propto p(y = c|D) \prod_{j=1}^{d} p(x_j|y = c, D)
\]

\[
= \pi_c \prod_j t_{2\alpha_{jcn}} \left( x_j | \mu_{jcn}, \frac{\beta_{jcn}(\kappa_{jcn} + 1)}{\alpha_{jcn}\kappa_{jcn}} \right)
\]