CSE517A Machine Learning **Spring 2024**

Lecture 5: Naïve Bayes Classifier

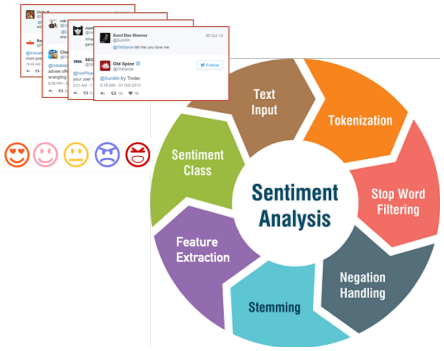
Instructor: Marion Neumann

Reading: FCML 5.2.1 (Bayes Classifier, Naïve Bayes, Classifying Text, and Smoothing)

Application

Sentiment analysis aims at discovering people's opinions, emotions, feelings about a subject matter, product, or service from text.

Take some time to answer the following warm-up questions: (1) Is this a regression or classification problem? (2) What are the features and how would you represent them? (3) What is the prediction task? (4) How well do you think a linear model will perform?



1 Introduction

Thought: Can we model $p(y \mid \mathbf{x})$ without model assumptions, e.g. Gaussian/Bernoulli distribution etc.?

Idea: Estimate $p(y \mid \mathbf{x})$ from the data *directly*, then use Bayes classifier: $y^* = \arg\max_{c \in C} p(y = c \mid \mathbf{x}^*, D)$.

Let y be discrete (i.e., $c \in C$), then we can get an estimate by **counting** occurrences in the training data:

$$p(y = c \mid \mathbf{x}) = \frac{\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x} \cap y_i = c)}{\sum_{i=1}^{n} I(\mathbf{x}_i = \mathbf{x})}, \tag{1}$$

where $I(\cdot)$ is the indicator function that is 1 if $\cdot$ is true and 0 if $\cdot$ is false. Note that the numerator counts all examples with input $\mathbf{x}$ that have label c and the denominator counts *all* examples with input $\mathbf{x}$. Hence, we get a probability like so:

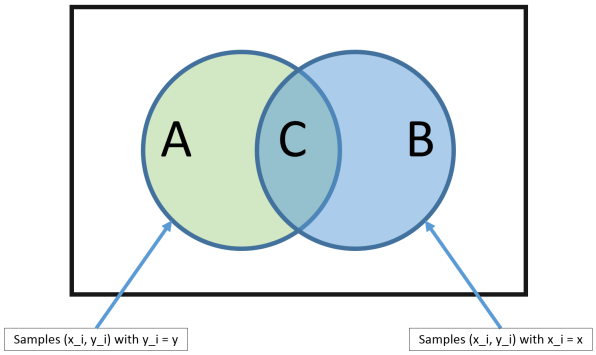


Figure 1: Illustration of estimating $\hat{p}(y \mid \mathbf{x})$

From Figure 1, it is clear that we can estimate $p(y \mid \mathbf{x})$ *directly from the data* as

$$\hat{p}(y \mid \mathbf{x}) = \frac{|C|}{|B|}. \tag{2}$$

This is equivalent to the *MLE method* in our coin flipping example! However, there is a big **problem** with this method:

The MLE estimate is only good if there are many training vectors with the **same identical** features as $\mathbf{x}$, but this **never** happens for high-dimensional or continuous feature spaces.

Solution: Bayes Rule.

$$p(y \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y)\, p(y)}{p(\mathbf{x})} \tag{3}$$

Let's estimate $p(\mathbf{x} \mid y)$ and $p(y)$ instead!
More specifically, we estimate the *class conditional likelihood* $p(\mathbf{x} \mid y, D)$ and the *prior class distribution given the data* $p(y \mid D)$ instead! Note that we don't care for $p(\mathbf{x}) = p(\mathbf{x} \mid D)$, since the Bayes classifier takes the argmax with respect to c and this part does not depend on c.

2 Naïve Bayes

We have a **discrete label space** C that can either be *binary* $\{+1, -1\}$ or *multi-class* $\{1, ..., K\}$. The **feature space** $\mathcal{X}$ may be *categorical* (binary or with more than two categories), *multi-nomial*, or *continuous*.

2.1 Estimate $p(y)$

Estimating $p(y)$ is easy. If y takes on discrete binary values, for example, this just becomes coin tossing.

Let $\hat{\pi}$ be our estimator for $p(y)$ derived via counting similar as illustrated for $p(y \mid \mathbf{x})$ above.

Then, for *binary classification* we have:

$$\hat{\pi}_{y=+1} = p(y = +1)$$
$$\hat{\pi}_{y=-1} = p(y = -1)$$

and for *multi-class classification* we get:

$$\hat{\pi}_c = p(y = c) = \frac{1}{n} \sum_{i=1}^{n} I(y_i = c) \qquad \text{for all } c \in \{1, ..., K\}.$$

2.2 Estimate $p(\mathbf{x} \mid y)$

Estimating $p(\mathbf{x} \mid y)$ is not easy. **Can you guess why?**

So, we have to make an assumption, the so-called *Naïve Bayes assumption*.

Naïve Bayes Assumption: The features are **independent given the label**, hence

$$p(\mathbf{x} \mid y) = \prod_{\alpha=1}^{d} p([\mathbf{x}]_\alpha \mid y) \tag{4}$$

where $[\mathbf{x}]_\alpha$ is the value for feature α.

Assume Equation (4) holds, then the **Naïve Bayes classifier** can be derived from the **Bayes classifier** as:

$$
\begin{aligned}
y = h(\mathbf{x}) &= \arg\max_y p(y \mid \mathbf{x}) \\
&= \arg\max_y \frac{p(\mathbf{x} \mid y)p(y)}{p(\mathbf{x})} \\
&= \arg\max_y p(\mathbf{x} \mid y)\, p(y) \\
&= \arg\max_y \prod_{\alpha=1}^{d} p([\mathbf{x}]_\alpha \mid y)\, p(y)
\end{aligned}
\tag{5}
$$

Estimating $\log p([\mathbf{x}]_\alpha \mid y)$ is easy as we only need to consider one dimension. And estimating $p(y)$ is not affected by the assumption.

Case 1: Categorical Features

Features: $[\mathbf{x}]_\alpha \in \{1, 2, ..., K_\alpha\}$ (no ordering, K_α = number of categories for feature α).

Model: We use the categorical distribution (also sometimes referred to *multinoulli* distribution):

$$
p([\mathbf{x}]_\alpha \mid y = c) = \prod_{j=1}^{K_\alpha} [\boldsymbol{\theta}_{\alpha c}]_j^{I([\mathbf{x}]_\alpha = j)} \tag{6}
$$

with parameters $[\boldsymbol{\theta}_{\alpha c}]_j = p([\mathbf{x}]_\alpha = j \mid y = c)$.

$\boldsymbol{\theta}_{\alpha c}$ is the vector of category probabilities of input feature α, given that the class label is c. That is, the j-th entry in this probability vector $[\boldsymbol{\theta}_{\alpha c}]_j$ is the probability of feature α having the value j, given that the label is c. Note that these probabilities sum to one:

$$
\sum_{j=1}^{K_\alpha} [\boldsymbol{\theta}_{\alpha c}]_j = 1.
$$

Parameter Estimation:

$$
[\hat{\boldsymbol{\theta}}_{\alpha c}]_j = \frac{\sum_{i=1}^{n} I(y_i = c) I([\mathbf{x}_i]_\alpha = j) + l}{\sum_{i=1}^{n} I(y_i = c) + l K_\alpha}, \text{ where } l \geq 0 \text{ is a } \textit{smoothing} \text{ parameter.} \tag{7}
$$

Training the Naïve Bayes classifier corresponds to estimating $[\hat{\boldsymbol{\theta}}_{\alpha c}]_j$ for all α, j and c and storing them in the respective *conditional probability tables* (CPT). Since we use those parameter estimates to get $p([\mathbf{x}]_\alpha \mid y = c)$, we get one CPT per feature dimension α with one column per class label c and one row per feature category j. Also note that by setting $l = 0$, we get an MLE estimator; $l > 0$ leads to MAP. Setting $l = 1$ is referred to as *Laplace smoothing*.

Prediction:

$$
\begin{aligned}
y &= \arg\max_c p(y = c \mid \mathbf{x}) \\
&= \arg\max_c \hat{\pi}_c \prod_{\alpha=1}^{d} [\hat{\boldsymbol{\theta}}_{\alpha c}]_j
\end{aligned}
\tag{8}
$$

Exercise 2.1. Should you play tennis or not?

Consider the test input $\mathbf{x}_* = [Overcast, Hot, High, Strong]$, work out the CPTs and $\hat{\pi}$, then predict y_* (y_* =yes or y_* =no) for the following data:

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
Day1	Sunny	Hot	High	Weak	*No*
Day2	Sunny	Hot	High	Strong	*No*
Day3	Overcast	Hot	High	Weak	Yes
Day4	Rain	Mild	High	Weak	Yes
Day5	Rain	Cool	Normal	Weak	Yes
Day6	Rain	Cool	Normal	Strong	*No*
Day7	Overcast	Cool	Normal	Strong	Yes
Day8	Sunny	Mild	High	Weak	*No*
Day9	Sunny	Cool	Normal	Weak	Yes
Day10	Rain	Mild	Normal	Weak	Yes
Day11	Sunny	Mild	Normal	Strong	Yes
Day12	Overcast	Mild	High	Strong	Yes
Day13	Overcast	Hot	Normal	Weak	Yes
Day14	Rain	Mild	High	Strong	*No*

Case 2: Multinomial Features

Features: $[\mathbf{x}]_\alpha \in \{0, 1, 2, ..., m\}$ (count features).
Each feature $[\mathbf{x}]_\alpha$ are counts of occurrences in a sequence of length $m = \sum_{\alpha=1}^{d} [\mathbf{x}]_\alpha$. An example of this is the representation of text, where the sequence is the document and the features are counts of each specific word α in the document of length m. The feature dimensionality d is the size of the vocabulary.

Model: Use the multinomial distribution (cf. FCML 2.3.3):

$$p(\mathbf{x} \mid m, y = c) = \frac{m!}{[\mathbf{x}]_1![\mathbf{x}]_2!...[\mathbf{x}]_d!} \prod_{\alpha=1}^{d} \theta_{\alpha c}^{[\mathbf{x}]_\alpha} \tag{9}$$

where $\theta_{\alpha c}$ is the probability of selecting $[\mathbf{x}]_\alpha$ and $\sum_{\alpha=1}^{d} \theta_{\alpha c} = 1$.

For example, we can use this to generate a spam email, $\mathbf{x}$ of class y = spam by by picking m words independently at random from the vocabulary of d word using $p(\mathbf{x} \mid y = spam)$.

Parameter Estimation:

$$\hat{\theta}_{\alpha c} = \frac{\sum_{i=1}^{n} I(y_i = c)[\mathbf{x}_i]_\alpha + l}{\sum_{i=1}^{n} I(y_i = c) \sum_{\beta=1}^{d} [\mathbf{x}_i]_\beta + ld} \tag{10}$$

where the numerator sums up all counts for feature α given the class label c and the denominator sums up all counts of all features given the class label. E.g.:

$$\frac{\text{\# of times word } \alpha \text{ appears in all spam emails}}{\text{length of all spam emails}}$$

Note that $\sum_{\beta=1}^{d} [\mathbf{x}_i]_\beta$ corresponds to m_i and l is a *smoothing* parameter.

Prediction:

$$\begin{aligned} y &= \arg\max_c p(y = c \mid \mathbf{x}) \\ &= \arg\max_c \hat{\pi}_c \prod_{\alpha=1}^{d} \hat{\theta}_{\alpha c}^{[\mathbf{x}]_\alpha} \end{aligned} \tag{11}$$

Exercise 2.2. Consider the following **sentiment analysis** data:

"this book is awesome"	*positive*
"harry potter books suck"	*negative*
"these pretzles are making me thirsty"	*negative*
"they choppin my fingers off Ira"	*negative*
"supreme beings of leisure rock"	*positive*
"cheeto jesus is a tyrant"	*negative*

(a) Train a multinomial Naïve Bayes classifier for binary sentiment prediction ($y \in \{positive, negative\}$). HINT: Perform some text preprocessing, like stopword filtering.

(b) Using your trained Naïve Bayes model, predict the sentiment for the following test input x^* = "just had my first cheeto ever it was awesome".

(c) Add a smoothing term of $l = 1$ to the probability of observing each word. Recalculate your Naïve Bayes prediction.

Case 3: Continuous Features (Gaussian Naïve Bayes)

Features: $[\mathbf{x}]_\alpha \in \mathbb{R}$ (the feature takes on a real value).

Model: Use Gaussian distribution (cf. FCML 2.5.3):

$$p([\mathbf{x}]_\alpha \mid y = c) = \mathcal{N}(\mu_{\alpha c}, \sigma^2_{\alpha c}) = \frac{1}{\sqrt{2\pi\sigma^2_{\alpha c}}} e^{-\frac{1}{2}\left(\frac{[\mathbf{x}]_\alpha - \mu_{\alpha c}}{\sigma_{\alpha c}}\right)^2} \tag{12}$$

Note that the model specified above is based on our assumption about the data - that each feature α comes from a class-conditional Gaussian distribution. This model choice leads to *Gaussian Naïve Bayes* (GNB). Other specification could be used as well.

Parameter Estimation:

$$\hat{\mu}_{\alpha c} \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)[\mathbf{x}_i]_\alpha \tag{13}$$

$$\hat{\sigma}^2_{\alpha c} \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)\left([\mathbf{x}_i]_\alpha - \mu_{\alpha c}\right)^2 \tag{14}$$

where $n_c = \sum_{i=1}^{n} I(y_i = c)$.

Exercise 2.3. Consider the following data:

	Gender	Height	Weight	Foot_Size
0	male	6.00	180	12
1	male	5.92	190	11
2	male	5.58	170	12
3	male	5.92	165	10
4	female	5.00	100	6
5	female	5.50	150	8
6	female	5.42	130	7
7	female	5.75	150	9

(a) Train a Gaussian Naïve Bayes classifier for a binary gender prediction task ($y \in \{male, female\}$).

(b) Using your trained Naïve Bayes model, predict the gender for the following test input $x^* = [6, 130, 8]$.

(c) Why do we not need to add smoothing terms for continuous features?

Understanding the Naïve Assumption

If we were to **not** use the naïve assumption the parameters to estimate for a Gaussian likelihood $p(\mathbf{x} \mid y)$ are given as:

$$\hat{\boldsymbol{\mu}}_c \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)\, \mathbf{x}_i \tag{15}$$

$$\hat{\Sigma}_c \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)\, (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)\, (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)^\top \tag{16}$$

Note that Eq. (13) and Eq. (15) are **equivalent** and that the covariance matrix $\hat{\Sigma}_c$ is **symmetric**.

Example $\mathbf{x} \in \mathbb{R}^2$:
Without naïve assumption:

$$\hat{\boldsymbol{\mu}}_c \leftarrow \begin{bmatrix} \hat{\mu}_{1c} \\ \hat{\mu}_{2c} \end{bmatrix} \quad \text{(from Eq. (13) or Eq. (15))}$$

$$\hat{\Sigma}_c \leftarrow \begin{bmatrix} \hat{\sigma}^2_{1c} & \hat{\sigma}_{12c} \\ \hat{\sigma}_{21c} & \hat{\sigma}^2_{2c} \end{bmatrix} \quad \text{(from Eq. (16))}$$

With naïve assumption:

$$\hat{\boldsymbol{\mu}}_c \leftarrow \begin{bmatrix} \hat{\mu}_{1c} \\ \hat{\mu}_{2c} \end{bmatrix} \quad \text{(from Eq. (13) or Eq. (15))}$$

$$\hat{\Sigma}_c \leftarrow \begin{bmatrix} \hat{\sigma}^2_{1c} & 0 \\ 0 & \hat{\sigma}^2_{2c} \end{bmatrix} \quad \text{(from Eq. (14))}$$

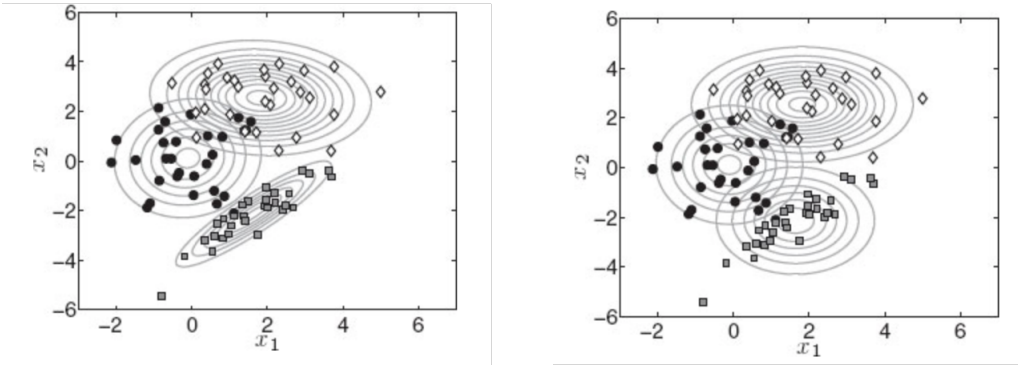


Figure 2: Gaussian likelihood models for $d = 2$ **without** (*left*) and **with** (*right*) **naïve assumption**.

Exercise 2.4. Consider the example datasetshown in Figure 2 with $\mathbf{x} \in \mathbb{R}^{10}$ and $K = 3$.

(a) What is the total **number of parameters** you will need to estimate for a Naïve Bayes model using the Gaussian likelihood *with* and *without* the naïve assumption?

(b) Can you derive a general formula for both cases for any d and K?

Reminder: the parameters for the Gaussian likelihood with naïve assumption for $d = 10$ are:

$$\hat{\boldsymbol{\mu}}_c \leftarrow \frac{1}{n_c} \sum_{i=1}^{n} I(y_i = c)\, \mathbf{x}_i$$

$$\hat{\Sigma}_c \leftarrow \begin{bmatrix} \hat{\sigma}_{1c}^2 & & 0 \\ & \ddots & \\ 0 & & \hat{\sigma}_{10c}^2 \end{bmatrix}$$

2.3 Naïve Bayes is a Linear Classifier

If the likelihood factors $p([\mathbf{x}]_\alpha \mid y)$ are from the *exponential family* (e.g. Gaussian, Bernoulli (i.e., categorical for binary features), Dirichlet, beta, multinomial), Naïve Bayes is a linear classifier. In general, however, this is not true.

In this case we can a linear classifier with the respective parameters $\mathbf{w}$ and b:

$$h(\mathbf{x}) = +1 \iff \mathbf{w}^\top \mathbf{x} + b > 0 \tag{17}$$

(1) Suppose $y_i \in \{-1, +1\}$ and features are binary (*Bernoulli Naïve Bayes*). We can show that

$$\begin{aligned} h(\mathbf{x}) &= \arg\max_c \; p(y = c) \prod_{\alpha=1}^{d} p([\mathbf{x}]_\alpha \mid y = c) \\ &= sign(\mathbf{w}_{\text{BER}}^\top \mathbf{x} + b_{\text{BER}}) \end{aligned} \tag{18}$$

(2) Suppose $y_i \in \{-1, +1\}$ and features are multinomial. We can show that

$$\begin{aligned} h(\mathbf{x}) &= \arg\max_c \; p(y = c)\, p(\mathbf{x} \mid m, y = c) \\ &= sign(\mathbf{w}_{\text{MULT}}^\top \mathbf{x} + b_{\text{MULT}}) \end{aligned} \tag{19}$$

(3) Suppose $y_i \in \{-1, +1\}$ and features are continuous (GNB), let $\sigma_{\alpha,c=-1} = \sigma_{\alpha,c=+1}$ (standard deviation does not depend on label), we can show that

$$p(y \mid \mathbf{x}) = \frac{1}{1 + e^{-y(\mathbf{w}_{\text{GNB}}^\top \mathbf{x} + b_{\text{GNB}})}} \tag{20}$$

Note: This model is also known as logistic regression. NB and LR produce *asymptotically* the same model. But NB and LR are different learning algorithms since the parameters are estimated differently (i.e., $\mathbf{w}_{\text{GNB}} \neq \mathbf{w}_{\text{LR}}$ and $b_{\text{GNB}} \neq b_{\text{LR}}$).

Exercise 2.5. This will be a homework problem.

(a) Derive $\mathbf{w}_{\text{BER}}$ and b_{BER} for (1) in terms of the Bernoulli Naïve Bayes model parameters. Hint: The Bernoulli case is just the categorical case with binary outcomes!

(b) Derive $\mathbf{w}_{\text{MULT}}$ and b_{MULT} for (2) in terms of the multinomial Naïve Bayes model parameters.

(c) Derive $\mathbf{w}_{\text{GNB}}$ and b_{GNB} for (3) in terms of Gaussian Naïve Bayes model parameters.

2.4 [optional] "True" Bayesian Naïve Bayes

Note that even though Naïve Bayes uses Bayes rule, it is **not** a Bayesian classifier. We can make it Bayesian by incorporating a *prior distribution over our parameters*. As an example we will look at *multi-class* classification with *binary* features here.

Similar to the Bayesian approach to coin flipping, we can incorporate a prior distribution over $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ and utilize the posterior distribution for predictions to achieve "true" Bayesian Naïve Bayes. Common choices for the prior distributions are the Dirichlet distribution for the class probability (multivariate generalization of the beta distribution) and the factored beta distribution for $\boldsymbol{\theta}$:

$$\begin{aligned} p(\boldsymbol{\pi}) &= Dir(\boldsymbol{\alpha}) \\ p(\boldsymbol{\theta}) &= \prod_{\alpha} \prod_{c} p(\theta_{\alpha c}), \text{ where} \\ p(\theta_{\alpha c}) &= Beta(\beta_0, \beta_1) \end{aligned}$$

where α, β_0, β_1 are the *hyperparameters*. If we set those to 1, we get Laplace smoothing which is equivalent to MAP estimation. With our Bayesian framework, we can now use the posterior means $\bar{\boldsymbol{\pi}}$ and $\bar{\theta}_{\alpha c}\ \forall \alpha, c$ to get estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$. Those posterior means can again be computed via counting and their derivation is left as an exercise for the interested reader. Note that for categorical features $\boldsymbol{\theta}_{\alpha c}$ would be a vector again and we would need the Dirichlet distribution instead of the beta distribution.

Prediction:

$$\begin{aligned} p(y = c \mid \mathbf{x}, D) &\propto \int p(y = c \mid \pi)\, p(\pi \mid D)\, \mathrm{d}\pi \prod_{\alpha=1}^{d} \int p([\mathbf{x}]_\alpha \mid y = c, \theta_{\alpha c})\, p(\theta_{\alpha c} \mid D)\, \mathrm{d}\theta_{\alpha c} \\ &\propto \bar{\pi}_c \prod_{\alpha=1}^{d} (\bar{\theta}_{\alpha c})^{I([\mathbf{x}]_\alpha = j)} (1 - \bar{\theta}_{\alpha c})^{I([\mathbf{x}]_\alpha \neq j)} \end{aligned} \tag{21}$$

2.5 Mixed and Missing Features

Naïve Bayes can handle features of *mixed types* and even *missing features* very naturally. To see the former, recall that the Naïve Bayes classifier is given by $h(\mathbf{x}) = \arg\max_y \prod_{\alpha=1}^{d} p([\mathbf{x}]_\alpha \mid y)\, p(y)$ (cf. Eq. (5)) and that we can estimate $p([\mathbf{x}]_\alpha \mid y)$ according to each feature type independently.

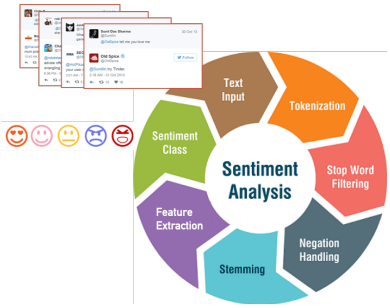
For missing features we can simply use:

$$y = \arg\max_c p(y = c \mid \mathbf{x}) = \arg\max_c \hat{\pi}_c \prod_{\alpha \in \text{OBS}} p([\mathbf{x}]_\alpha \mid y = c), \tag{22}$$

where OBS is the set of features that are observed for test input $\mathbf{x}$.

Exercise 2.6. To see how the Naïve Bayes classifier can deal with unobserved (missing) features we will look at an example: **Suzie's date** (in-class exercise; also available on Canvas).

Application



Reconsider our sentiment analysis problem.

If you were to use Naïve Bayes as your sentiment predictor, what feature representation would make sense: *categorical*, *multinomial*, *continuous*, or mixed? (There is no correct/incorrect answer, so be creative!)

Is your Naïve Bayes sentiment predictor a linear classifier?

We will implement a Naïve Bayes classifier in our second implementation project.

2.6 Summary

- We need different models for different feature types!
- MLE overfits and does not work for small n (number of samples)
- use smoothing → MAP solution
- Naïve Bayes is not a Bayesian method
- Naïve Bayes can deal with mixed and missing features (this is not the case for most other ML methods!)

Exercise 2.7. Practice Retrieving!
For this summary exercise, it is intended that your answers are based on **your own** (current) understanding of the concepts (and not on the definitions you read and copy from these notes or from elsewhere). Don't hesitate to **say it out loud** to your seat neighbor, your pet or stuffed animal, or to yourself before **writing it down**. After writing it down, check your answers with your (lecture)notes and the provided reading. Correct any mistakes you made. Research studies show that this practice of retrieval and phrasing out loud will help you retain the knowledge!

(a) Using *your own words*, recap each of the above summary points in 2-3 sentences by retrieving the knowledge from the top of your head.

(b) Name one example for an application with *categorical* features. How do we model categorical features for NB?

(c) Name one example for an application with *multinomial* features. How do we model multinomial features for NB?

(d) Name one example for an application with *continuous* features. How do we model continuous features for NB?

(e) Name one example for an application that has mixed features (*categorical*, *multinomial*, and *continuous*) features.

(f) How can we implement MAP estimation instead of MLE estimation in NB?

(g) Critically think about the multinomial Naïve Bayes. Why does is make sense to say that "multinomial Naïve Bayes" is a misnomer? Hint: think about the naïve assumption and whether it actually applies.

And always remember: It's not bad to get it wrong. *Getting it wrong is part of learning!* Use your notes or

3 Discussion: Generative v.s. Discriminative Learning

All models use $p(y \mid \mathbf{x})$ for predictions, but

- *generative* models estimate $p(\mathbf{x} \mid y)\, p(y)$
- *discriminative* models estimate $p(y \mid \mathbf{x})$ directly

Here we give some comparisons between generative and discriminative learning.

(1) Training efficiency: generative models are usually faster to train

(2) Predicition quality: discriminative models are usually more accurate and give better probability estimates

(3) Generating data: generative models can be used to generate data (e.g. spam email)

(4) Assumptions: generative models make strong independence assumptions which are often not valid; discriminative models make stronger model assumptions