Applied Machine Learning (AML)

Naive Bayes

Oisin Mac Aodha • Siddharth N.

Naive Bayes Classification

Generative Classification

- In classification the goal is to a learn a function $\hat{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$
  - $y \in \{1, ..., C\}$ is one of C classes (e.g. spam / ham, digits 0-9)
  - $\boldsymbol{x} = [x_1, ..., x_D]^\top$ are the features (e.g. continuous or discrete)
- In probabilistic classification we choose the most probable class given an observation

$$\hat{y} = \arg\max_c p(y = c|\boldsymbol{x})$$

- We can use **Bayes's rule** to convert the class prior and class-conditionals to a posterior probability for a class

$$p(y = c|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|y = c)p(y = c)}{\sum_{c'} p(\boldsymbol{x}|y = c')p(y = c')} \qquad \text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

Generative Classification - Components

$$p(y = c|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|y = c)p(y = c)}{p(\boldsymbol{x})}$$

- $p(\boldsymbol{x}|y = c)$: **Likelihood**
  - Class conditional density for each class
  - Describes how likely we are to see observation $\boldsymbol{x}$ for a given class
- $p(y = c)$: **Prior probability**
  - The prior for each class
  - Encodes which classes are common and which are rare
- $p(\boldsymbol{x})$: **Evidence**
  - $p(\boldsymbol{x}) = \sum_{c'} p(\boldsymbol{x}|y = c')p(y = c')$
  - Normalises the probabilities across observations
  - Does not impact which class is the most likely

Representing the Class Conditional Density

- Representing the prior $p(y)$ for each class is straight forward, i.e. we can compute frequency of each class
- We need to choose a probabilistic model for our conditional density $p(\boldsymbol{x}|y)$

- For example, for multivariate continuous data i.e. $\boldsymbol{x} \in \mathbb{R}^D$, we can use the multivariate Gaussian with parameters $\boldsymbol{\mu}_c$ and Σ_c
- However, this requires estimating $D(D+1)/2$ parameters for each class covariance matrix Σ_c, which may be problematic as the dimensionality D gets large

Naive Bayes Assumption

- Naive Bayes makes the *simplifying* assumption that the **features are conditionally independent given the class label**

$$p(\boldsymbol{x}|y) = \prod_{d=1}^{D} p(x_d|y)$$

- The model is called "*naive*" since we do not expect the features to be independent, even conditional on the class labels
- Even though this assumption is not typically true, Naive Bayes can still work well in practice

Independence

- **Independence** means that one variable does not affect another, A is (*marginally*) independent of B if

$$p(A|B) = P(A)$$

- Which, from the definition of the conditional probability, is equivalent to saying

$$p(A, B) = P(A)P(B)$$

- A is **conditionally independent** of C given B if

$$p(A|C, B) = p(A|B)$$

i.e. once we know B, knowing C does not provide additional information about A

Conditional Independence - Example

- The probabilities of going to the beach and having a heat stroke are not independent, i.e.

$$p(B, S) > p(B)p(S)$$

- However, they may be independent if we know the weather is hot

$$p(B, S|H) = p(B|H)p(S|H)$$

- Hot weather "explains" all the dependence between the beach and heatstroke
- In classification, the class label explains all the dependence between the features

Conditional Independence of Features - Example

- Suppose we had a feature vector $\boldsymbol{x} = [x_1, x_2, x_3]^\top$, we can write out the conditional probability as

$$
\begin{aligned}
p(\boldsymbol{x}|y) &= p(x_1, x_2, x_3|y) \\
&= p(x_3|x_2, x_1, y)p(x_2, x_1|y) \\
&= p(x_3|x_2, x_1, y)p(x_2|x_1, y)p(x_1|y) \\
&= \prod_{d=1}^{D} p(x_d|x_{d-1}, ..., x_1, y)
\end{aligned}
$$

- In Naive Bayes we make the following simplifying assumption

$$p(\boldsymbol{x}|y) = \prod_{d=1}^{D} p(x_d|y)$$

Naive Bayes with Binary Data

Spam Email Classification Example

- The task is to separate spam from ham (i.e. 'not spam') emails
- We have access to the following dataset containing six emails

id	email	status
1	"send us your password"	spam
2	"send us review"	ham
3	"review your account"	ham
4	"review us"	spam
5	"send your password"	spam
6	"send us your account"	spam

- We can fit a **Naive Bayes classifier** to this data so that we can classify new emails

Representing Text Data

- We need to turn each email into a *vector* $\boldsymbol{x}$
- We can simply use a binary feature $x_d \in \{0, 1\}$ to indicate if a specific word is present or not
- For example, for a vocabulary with the following words:
  { 'password', 'review', 'send', 'us', 'your', 'account' }

  The email containing the text **"send us your password"** would be encoded as $\boldsymbol{x} = [1, 0, 1, 1, 1, 0]$

- We can exclude common words, e.g. 'a', 'the', ...

Representing Text Data

- Given the following vocabulary we can extract features from our data: { 'password', 'review', 'send', 'us', 'your', 'account' }

id	email	feature	status
1	"send us your password"	$[1, 0, 1, 1, 1, 0]$	spam
2	"send us review"	$[0, 1, 1, 1, 0, 0]$	ham
3	"review your account"	$[0, 1, 0, 0, 1, 1]$	ham
4	"review us"	$[0, 1, 0, 1, 0, 0]$	spam
5	"send your password"	$[1, 0, 1, 0, 1, 0]$	spam
6	"send us your account"	$[0, 0, 1, 1, 1, 1]$	spam

informatics

Modelling Binary Features

- As the features are **binary**, i.e. $x_d \in \{0, 1\}$, we can use the Bernoulli distribution to represent the class condition density

$$p(\boldsymbol{x}|y = c; \boldsymbol{\theta}) = \prod_{d=1}^{D} \text{Ber}(x_d|\phi_{dc})$$

- Here, $\phi_{dc} \in [0, 1]$ is the probability that $x_d = 1$ when y is class c

$$\text{Ber}(x_d|\theta_{dc}) = \theta_{dc}^{x_d}(1 - \theta_{dc})^{(1-x_d)}$$

- In the case of binary features, the maximum likelihood estimate is

$$\hat{\phi}_{\text{MLE}} = \frac{N_{dc}}{N_c}$$

i.e. the empirical fraction of times that feature d is present in examples from class c

informatics

Spam Classification Example

id	email	status
1	"send us your password"	s
2	"send us review"	h
3	"review your account"	h
4	"review us"	s
5	"send your password"	s
6	"send us your account"	s

- Class priors:

  $p(\text{spam}) = 4/6 \quad p(\text{ham}) = 2/6$

- Per-class likelihoods:

$p(x_d\|\text{spam})$	$p(x_d\|\text{ham})$	x_d
2/4	0/2	password
1/4	2/2	review
3/4	1/2	send
3/4	1/2	us
3/4	1/2	your
1/4	1/2	account

informatics

Classifying New Data - Spam Likelihood

- Given an *new* email we would like to be able to classify it
- For example, given the test email: "review us now"
- $\boldsymbol{x}_t = [0, 1, 0, 1, 0, 0]^\intercal$

- Class priors:

  $p(\text{spam}) = 4/6 \quad p(\text{ham}) = 2/6$

- Per-class likelihoods:

$p(x_d\|\text{spam})$	$p(x_d\|\text{ham})$	x_d
2/4	0/2	password
1/4	2/2	review
3/4	1/2	send
3/4	1/2	us
3/4	1/2	your
1/4	1/2	account

$$\begin{aligned} p(\boldsymbol{x}_t|\text{spam}) &= p(0, 1, 0, 1, 0, 0|\text{spam}) \\ &= (1 - \frac{2}{4})(\frac{1}{4})(1 - \frac{3}{4})(\frac{3}{4})(1 - \frac{3}{4})(1 - \frac{1}{4}) = 0.004 \end{aligned}$$

informatics

Classifying New Data - Ham Likelihood

- Given an *new* email we would like to be able to classify it
- For example, given the test email: "review us now"
- $\boldsymbol{x}_t = [0, 1, 0, 1, 0, 0]^\intercal$
- Class priors: $p(\text{spam}) = 4/6 \quad p(\text{ham}) = 2/6$
- Per-class likelihoods:

$p(x_d \mid \text{spam})$	$p(x_d \mid \text{ham})$	x_d
2/4	0/2	password
1/4	2/2	review
3/4	1/2	send
3/4	1/2	us
3/4	1/2	your
1/4	1/2	account

$$\begin{aligned} p(\boldsymbol{x}_t \mid \text{ham}) &= p(0, 1, 0, 1, 0, 0 \mid \text{ham}) \\ &= (1 - \frac{0}{2})(\frac{2}{2})(1 - \frac{1}{2})(\frac{1}{2})(1 - \frac{1}{2})(1 - \frac{1}{2}) = 0.0625 \end{aligned}$$

Classifying New Data - Prediction

- From our Bayes classifier, we can obtain our **posterior** probability as

$$\begin{aligned} p(\text{ham} \mid \boldsymbol{x}_t) &= \frac{p(\boldsymbol{x}_t \mid \text{ham})p(\text{ham})}{p(\boldsymbol{x}_t \mid \text{ham})p(\text{ham}) + p(\boldsymbol{x}_t \mid \text{spam})p(\text{spam})} \\ &= \frac{0.0625 \times 2/6}{0.004 \times 4/6 + 0.0625 \times 2/6} \\ &= 0.88 \end{aligned}$$

- Thus, according to our model, the probability that "review us now" is a ham email is $p(\text{ham} \mid \boldsymbol{x}_t) = 0.88$
  and by extension,
  $p(\text{spam} \mid \boldsymbol{x}_t) = 1 - 0.88$

Problems With Naive Bayes

- **Zero-frequency problem**
  - e.g. any email containing the word "password" is spam $p(\text{password} \mid \text{ham}) = 0/2$
  - Solution: never allow *zero* probabilities
  - Laplace smoothing: add a small positive number to all counts

$$p(x_d \mid y) = \frac{N_{dc} + \epsilon}{N_c + 2\epsilon}$$

- **Independence assumption**
  - Every feature contributes independently
  - e.g. you can fool Naive Bayes by adding lots of 'hammy' words

Missing Data

- Suppose we do not have the value for some feature x_j?
  - e.g. some medical test was not performed on the patient
  - How can we compute $p(x_1 = 1, ..., x_j = ?, ...x_d \mid y)$?
- This is easy with Naive Bayes
  - We simply ignore the feature in any instance where the value is *missing*
  - We compute the likelihood based on observed features only

$$p(\boldsymbol{x} \mid y) = \prod_{\substack{d=1 \\ d \neq j}}^{D} p(x_d \mid y)$$

  - No need to 'estimate' or explicitly model missing features

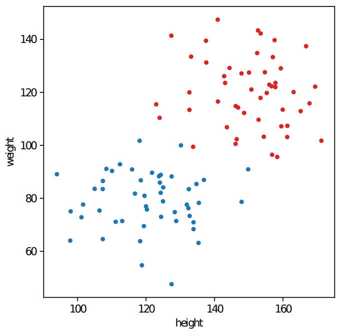
Naive Bayes with Continuous Data

Continuous Feature Example - Task

- Task: Distinguish alpacas from llamas based on size


Continuous Feature Example - Data

- Task: Distinguish alpacas from llamas
  - Classes: $y \in \{a, l\}$
  - Features: height (cm) and weight (kg)
  - Training examples: $\{(h_n, w_n, y_n)\}_{n=1}^{N}$
  - Assume height and weight are independent



Naive Bayes with Continuous Data

- In the case of real-valued features, $x_d \in \mathbb{R}$, we can use the univariate Gaussian distribution

$$p(\boldsymbol{x}|y = c; \boldsymbol{\theta}) = \prod_{d=1}^{D} \mathcal{N}(x_d|\mu_{dc}, \sigma^2_{dc})$$

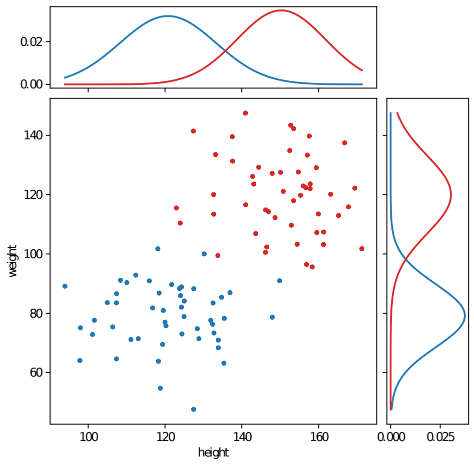
- Here μ_{dc} is the **mean** of feature d when the class label is c and σ^2_{dc} is its **variance**
- This is equivalent to Gaussian discriminant analysis using *diagonal covariance matrices*

Continuous Feature Example - Model

- Task: Distinguish alpacas from llamas
  - Classes: $y \in \{a, l\}$
  - Features: height (cm) and weight (kg)
  - Training examples: $\{(h_n, w_n, y_n)\}_{n=1}^{N}$
  - Assume height and weight are independent
- Class priors: $p(a) = N_a/N$ and $p(l) = N_l/N$
- Class conditionals for alpacas:
  - Height $\sim \mathcal{N}(x_h|\mu_{ha}, \sigma_{ha}^2)$
  - Weight $\sim \mathcal{N}(x_w|\mu_{wa}, \sigma_{wa}^2)$
- Class conditionals for llamas:
  - Height $\sim \mathcal{N}(x_h|\mu_{hl}, \sigma_{hl}^2)$
  - Weight $\sim \mathcal{N}(x_w|\mu_{wl}, \sigma_{wl}^2)$

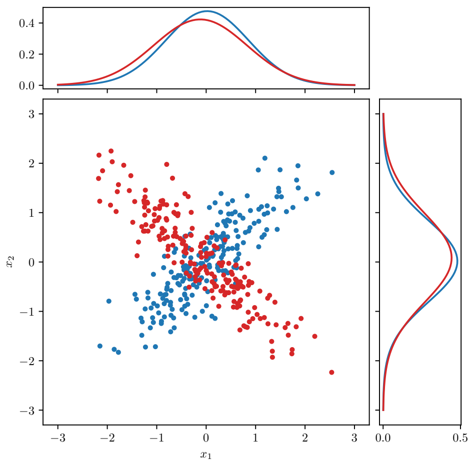
Continuous Feature Example

- Class priors:
  $p(a) = N_a/N$, $p(l) = N_l/N$
- Class conditionals for alpacas:
  - Height $\sim \mathcal{N}(x_h|\mu_{ha}, \sigma_{ha}^2)$
  - Weight $\sim \mathcal{N}(x_w|\mu_{wa}, \sigma_{wa}^2)$
- Class conditionals for llamas:
  - Height $\sim \mathcal{N}(x_h|\mu_{hl}, \sigma_{hl}^2)$
  - Weight $\sim \mathcal{N}(x_w|\mu_{wl}, \sigma_{wl}^2)$



Problems With Naive Bayes

- The conditional independence assumption used by Naive Bayes can fail to capture relationships that may be present in some datasets



Summary

- We presented the **Naive Bayes** classifier
- It assumes that *features* are conditionally independent given the *class*
- This results in a reduction in the number of parameters we need to learn
- We can apply it to both *discrete* and *continuous* data
- This underlying assumption of Naive Bayes is a simplification that will not necessarily work for all datasets