

# 5. Clasificatorul Naive Bayes

---

---

Algoritmi de învățare generativi  
Antrenarea clasificatorului Naive Bayes  
Medierea Laplace  
Predicția datelor

---

---

În lucrările de laborator precedente au fost prezentați algoritmi ce modelează distribuția condiționată  $p(y|x, \theta)$  a lui  $y$ , dându-se caracteristicile  $x$  și parametrii  $\theta$  [? ]. Spre exemplu, regresia logistică modelează  $p(y|x, \theta)$  sub forma  $h_\theta(x) = g(\theta^T x)$ , unde  $g$  este funcția sigmoid.

O posibilă problemă de clasificare este de a face diferența între elefanți ( $y = 1$ ) și câini ( $y = 0$ ) pe baza unui set de caracteristici (eng. features) ce descriu un animal. Dându-se o bază de date de antrenare, un algoritm precum regresia logistică încearcă să determine o linie dreaptă, denumită margine de decizie, ce separă elefanții de câini. Pentru a clasifica un nou animal ca fiind elefant sau câine, metoda verifică de ce parte a marginii de decizie se află caracteristicile animalului.

Abordarea prezentată în această lucrare de laborator construiește mai întâi un model ce descrie modul în care arată elefanții, urmat de construcția unui model separat ce descrie modul în care arată câinii. În final, pentru a clasifica un animal, putem calcula cât de mult se aseamănă el cu modelul ce descrie elefanții, pe de o parte, și cât de mult se aseamănă cu modelul ce descrie câinii, pe de altă parte.

Algoritmii precum regresia logistică, ce modelează  $p(y|x)$  direct, sau algoritmii ce învață să mapeze direct spațiul caracteristicilor de intrare  $\mathbb{X}$  la etichetele (eng. labels)  $\{0, 1\}$ , sunt denumiți *algoritmi de învățare discriminanți*. În cele ce urmează, vom discuta despre o altă clasă de algoritmi, intitulată *algoritmi de învățare generativi*, ce modelează distribuțiile  $p(x|y)$  și  $p(y)$ . Spre exemplu, dacă  $y$  indică faptul că un exemplu este un câine (0) sau un elefant (1), atunci  $p(x|y = 0)$  modelează distribuția caracteristicilor (eng. features) câinilor, iar  $p(x|y = 1)$  modelează distribuția caracteristicilor elefanților.

După modelarea lui  $p(y)$  (denumită *probabilitatea a priori*) și a lui  $p(x|y)$ , algoritmul calculează probabilitatea posterioară  $p(y|x)$  a lui  $y$  dându-se  $x$ , prin regula Bayes:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}, \quad (5.1)$$

unde numitorul are expresia:

$$p(x) = p(x|y = 1)p(y = 1) + p(x|y = 0)p(y = 0). \quad (5.2)$$

Nu este necesar să calculăm numitorul pentru a efectua o predicție utilizând  $p(y|x)$ :

$$\begin{aligned} \arg \max_y p(y|x) &= \arg \max_y \frac{p(x|y)p(y)}{p(x)} \\ &= \arg \max_y p(x|y)p(y). \end{aligned}$$

## 5.1 Clasificatorul Naive Bayes

În studiul clasificatorului Naive Bayes, vom lua în considerare caracteristici  $x_i$  ce iau valori discrete.

În această lucrare de laborator vom construi un filtru de email-uri spam, ce va detecta mesaje nesolicitate și comerciale în limba engleză. Acest algoritm poate filtra mesajele spam și le poate salva, spre exemplu, într-un folder separat. Clasificarea de email-uri face parte dintr-un spectru mai larg de probleme intitulate *clasificare de text*.

Luăm în considerare că avem la dispoziție o bază de date de antrenare ce conține email-uri etichetate ca spam sau non-spam. Primul pas în construcția filtrului de spam este specificarea caracteristicilor  $x_i$  ce descriu un email.

Vom reprezenta un email printr-un vector de caracteristici a cărui lungime este egală cu numărul de cuvinte dintr-un dicționar. În particular, dacă un email conține cuvântul de la poziția  $i$  din dicționar, atunci vom seta  $x_j = 1$ ; altfel  $x_i = 0$ . Spre exemplu, vectorul:

$$x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{2087} \\ \vdots \\ x_{39831} \\ \vdots \\ x_{50000} \end{bmatrix} \begin{matrix} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \text{buy} \\ \vdots \\ \text{price} \\ \vdots \\ \text{zygmurgy} \end{matrix}, \quad (5.3)$$

este utilizat pentru a descrie un email ce conține cuvintele "a" și "buy", dar nu și "aardvark", "aardwolf", sau "zygmurgy". Setul de cuvinte codat în vectorul de caracteristici este denumit *vocabular*. Astfel, dimensiunea lui  $x$  este egală cu dimensiunea vocabularului.

Luând în considerare vectorul de caracteristici definit mai sus, vom construi în continuare modelul generativ  $p(x|y)$ . Dacă am folosi, spre exemplu, un vocabular de 50000 de cuvinte, atunci  $x \in \{0, 1\}^{50000}$  ( $x$  este un vector de 50000 de dimensiuni, format din 0 și 1). Dacă ar fi să modelăm  $x$  explicit cu o distribuție multinomială pe cele  $2^{50000}$  posibile rezultate, atunci am obține un vector de parametri de dimensiune  $2^{50000} - 1$  (în mod evident cu un număr mult prea mare de parametri).

Pentru a modela  $p(x|y)$  se vor considera caracteristicile  $x_i$  ca fiind independente între ele, dându-se  $y$ . Această presupunere este denumită *presupunerea Naive Bayes*, iar algoritmul rezultat poartă numele de *clasificatorul Naive Bayes*. Spre exemplu, considerăm că  $y = 1$  reprezintă un email spam, iar cuvintele "buy" și "price" au indecșii 2087, respectiv 39831. În cazul unui email spam ( $y = 1$ ), informația  $x_{2087}$  (informația că mesajul conține cuvântul "buy") nu va avea niciun efect asupra valorii lui  $x_{39831}$  (informația că mesajul conține cuvântul "price"). Formal, acest lucru se poate scrie ca:

$$p(x_{2087}|y) = p(x_{2087}|y, x_{39831}). \quad (5.4)$$

A se lua în considerare că egalitatea 5.4 *nu* indică faptul că variabilele  $x_{2087}$  și  $x_{39831}$  sunt independente. Independența variabilelor una de cealaltă ar fi scrisă  $p(x_{2087}) = p(x_{2087}|x_{39831})$ . În 5.4 presupunem că  $x_{2087}$  și  $x_{39831}$  sunt condițional independente *dându-se*  $y$ .

Astfel, obținem:

$$\begin{aligned} p(x_1, \dots, x_{50000}|y) &= p(x_1|y) \cdot p(x_2|y, x_1) \cdot p(x_3|y, x_1, x_2) \cdot \dots \cdot p(x_{50000}|y, x_1, \dots, x_{49999}) \\ &= p(x_1|y) \cdot p(x_2|y) \cdot p(x_3|y) \cdot \dots \cdot p(x_{50000}|y) \\ &= \prod_{j=1}^n p(x_j|y). \end{aligned}$$

Prima egalitate este dedusă direct din proprietățile standard ale probabilităților, iar a doua folosește presupunerea Naive Bayes. Algoritmul rezultat funcționează bine în multe aplicații, cu toate că presupunerea Naive Bayes este puternică.

## 5.2 Antrenarea clasificatorului Naive Bayes

Pentru a parametriza modelul, vom lua în considerare distribuția Bernoulli  $\phi$ :

$$p(y = 1, \phi) = \phi. \quad (5.5)$$

Distribuția Bernoulli modelează variabilele aleatoare ce iau doar valorile 0 și 1. Ea poate fi utilizată pentru a modela procesul de "dat cu banul", unde 1 reprezintă "cap", iar 0 reprezintă "pajură" (sau vice versa). În acest caz  $\phi_{y=0} = \phi_{y=1} = 0.5$  (probabilitatea

de a obține ”cap” este egală cu probabilitatea de a obține ”pajură”). În cazul unui ban ”manipulat” (ce tinde, spre exemplu, să cadă în special pe ”cap”)  $\phi_{y=0} \neq \phi_{y=1}$ .

Modelul filtrului de spam este parametrizat astfel:

$$\phi_{j|y=1} = p(x_j = 1|y = 1), \quad (5.6)$$

$$\phi_{j|y=0} = p(x_j = 1|y = 0), \quad (5.7)$$

$$\phi_y = p(y = 1). \quad (5.8)$$

Dându-se o bază de date de antrenare,  $(x^{(i)}, y^{(i)}); i = 1, \dots, m$ , se poate scrie probabilitatea comună a datelor (likelihood):

$$\mathcal{L}(\phi_y, \phi_{j|y=0}, \phi_{j|y=1}) = \prod_{i=1}^m p(x^{(i)}, y^{(i)}). \quad (5.9)$$

Maximizând față de  $\phi_y$ ,  $\phi_{j|y=0}$  și  $\phi_{j|y=1}$ , obținem estimatul probabilității maxime (eng. Maximum Likelihood Estimation):

$$\phi_{j|y=1} = \frac{\sum_{i=1}^m 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 1\}}{\sum_{i=1}^m 1\{y^{(i)} = 1\}}, \quad (5.10)$$

$$\phi_{j|y=0} = \frac{\sum_{i=1}^m 1\{x_j^{(i)} = 1 \wedge y^{(i)} = 0\}}{\sum_{i=1}^m 1\{y^{(i)} = 0\}}, \quad (5.11)$$

$$\phi_y = \frac{\sum_{i=1}^m 1\{y^{(i)} = 1\}}{m}, \quad (5.12)$$

unde, simbolul ” $\wedge$ ” reprezintă operația binară ”ȘI logic”. De asemenea,  $1\{z\}$  reprezintă așa numita *funcție indicator* ce ia valoarea 1 atunci când expresia  $y$  este adevărată:

$$1\{\text{Adevărat}\} = 1, \quad (5.13)$$

$$1\{\text{Fals}\} = 0. \quad (5.14)$$

Spre exemplu,  $1\{2 = 3\} = 0$ , deoarece  $2 \neq 3$ , iar  $1\{1 + 1 = 2\} = 1$ .

Parametrii condiderați au o interpretare naturală. Spre exemplu,  $\phi_{j|y=1}$  este fracțiunea din email-urile spam ( $y = 1$ ) în care apare cuvântul  $j$ .

### 5.3 Medierea Laplace

Statistic, nu este o idee bună estimarea probabilității unui anumit eveniment ca fiind zero, doar pentru că nu a fost observat în baza de date de antrenare. *Medierea Laplace* este utilizată pentru a evita situația în care anumiți parametri  $\phi_j$  ar putea deveni zero, făcând astfel împărțirea imposibilă. Noul estimat, după medierea Laplace, devine:

$$\phi_j = \frac{\sum_{i=1}^m \mathbf{1}\{z^{(i)} = j\} + 1}{m + V}. \quad (5.15)$$

În formula de mai sus a fost adăugat 1 la numărător și  $V$  la numitor. În exemplul de față vom considera valoarea lui  $V$  egală cu numărul de cuvinte din dicționar.

### 5.4 Predicția datelor

Parametrii  $\phi_{j|y=1}$ ,  $\phi_{j|y=0}$  și  $\phi_y$  vor fi calculați din setul de date de antrenare. Pentru a efectua o predicție, vom utiliza acești parametri în comparația cantităților  $p(x|y=1)p(y=1)$  și  $p(x|y=0)p(y=0)$ . Odată ce am obținut estimatele parametrilor, putem efectua predicții pe noi exemple  $x$  astfel:

$$\begin{aligned} p(y=1|x) &= \frac{p(x|y=1) \cdot p(y=1)}{p(x)} \\ &= \frac{\left(\prod_{j=1}^n p(x_j|y=1)\right) \cdot p(y=1)}{\left(\prod_{j=1}^n p(x_j|y=1)\right) \cdot p(y=1) + \left(\prod_{j=1}^n p(x_j|y=0)\right) \cdot p(y=0)}. \end{aligned}$$

Așa cum am menționat în introducere, nu este necesar calculul numitorului din formula precedentă pentru a efectua o predicție.

Computațional este mai eficient să calculăm probabilitățile logaritmice  $\log p(x|y=1)$  și  $\log p(y=1)$ . Astfel, înmulțirile necesare calculului probabilităților vor fi înlocuite de adunări.

### 5.5 Cerințe

Să se scrie un script Python ce va clasifica mesaje email scrise în limba engleză ca fiind spam sau non-spam.

#### 5.5.1 Baza de date de antrenare

Pentru a implementa algoritmul, descărcați datele din arhiva "lab5Data.zip".

Baza de date este împărțită în două subseturi, și anume 700 de email-uri de antrenare și 260 de email-uri de testare. Fiecare subset conține 50% mesaje spam și 50% mesaje non-spam. Adicional, email-urile au fost procesate în felul următor:

- (a) Anumite cuvinte, precum "and", "the" și "of" sunt foarte comune în frazele din limba engleză și nu sunt relevante în decizia de a clasifica un mesaj ca spam sau nu. Aceste cuvinte au fost înlăturate din email-uri.
- (b) Cuvintele care au același înțeles, însă o terminație diferită, au fost ajustate în așa fel încât să aibă aceeași formă. Spre exemplu, "include", "includes" și "included" vor fi reprezentate de "include". Toate literele din corpul email-ului au fost convertite la litere mici.
- (c) Numerele și semnele de punctuație au fost înlăturate. Tab-urile, noile linii și spațiile au fost convertite la un singur spațiu.

Mai jos se găsesc două exemple de mesaje înainte și după procesare.

### **Mesajul non-spam "5-1361msg1" înainte de preprocesare**

Subject: Re: 5.1344 Native speaker intuitions

The discussion on native speaker intuitions has been extremely interesting, but I worry that my brief intervention may have muddied the waters. I take it that there are a number of separable issues. The first is the extent to which a native speaker is likely to judge a lexical string as grammatical or ungrammatical per se. The second is concerned with the relationships between syntax and interpretation (although even here the distinction may not be entirely clear cut).

### **Mesajul non-spam "5-1361msg1" după preprocesare**

re native speaker intuition discussion native speaker intuition  
extremely interest worry brief intervention muddy waters number  
separable issue first extent native speaker likely judge lexical  
string grammatical ungrammatical per se second concern  
relationship between syntax interpretation although even here  
distinction entirely clear cut

### **Mesajul spam "spmsgc19" după preprocesare**

financial freedom follow financial freedom work ethic

extraordinary desire earn least per month work home special  
skills experience required train personal support need ensure  
success legitimate homebased income opportunity put back control  
finance life ve try opportunity past fail live promise

Arhiva "lab5Data.zip" conține fișierul "train-features-full.txt", ce reprezintă matricea de antrenare  $x$ , formată din 700 de linii și 2500 de coloane. Fiecare linie reprezintă un email, iar fiecare coloană reprezintă un cuvânt din dicționar. Un element dintr-o linie a matricei  $x$  reprezintă numărul de apariții în email al cuvântului reprezentat de acel element.

Fișierul "train-labels.txt" conține etichetele atribuite fiecărui email ( $y = 1$  pentru email-urile spam și  $y = 0$  pentru email-urile non-spam) și are dimensiunea  $700 \times 1$ .

### 5.5.2 Antrenarea

Pașii de antrenare ai algoritmului Naive Bayes, odată ce baza de date de antrenare a fost încărcată în Python, sunt următorii:

1. calculul  $\phi_y$ ;
2. calculul fiecărui  $\phi_{j|y=1}$  pentru fiecare cuvânt din dicționar și stocarea rezultatelor într-un vector;
3. calculul fiecărui  $\phi_{j|y=0}$  pentru fiecare cuvânt din dicționar și stocarea rezultatelor într-un vector.

### 5.5.3 Testarea

Odată ce parametrii modelului au fost calculați, ei pot fi utilizați în efectuarea predicțiilor. Pentru aceasta se vor folosi fișierele "test-features-full.txt" și "test-labels.txt", ambele având structura fișierelor "train-features-full.txt", respectiv "train-labels.txt".

Pașii utilizați pentru clasificarea documentelor sunt următorii:

1. pentru fiecare document din baza de date de testare se calculează  $\log p(x|y = 1) + \log p(y = 1)$ ;
2. similar, se calculează  $\log p(x|y = 0) + \log p(y = 0)$ ;
3. se compară cele două cantități de la punctele (1) și (2) și se ia decizia de clasificare a email-ului în spam sau non-spam, luând în considerare cea mai mare valoare.