Лекция 10. Бейсов класификатор

10.1. Оптимален Бейсов класификатор

До сега ние се интересували от въпроса "каква е най-вероятната хипотеза при зададени обучаващи данни". Фактически, много често е по-важен въпросът "каква е най-вероятна *класификация на нов пример* при зададени обучаващи данни?" Макар изглежда, че отговорът на този въпрос може да бъде получен просто чрез прилагане на MAP хипотеза към новия пример, на практика е възможно да постъпим по-добре.

За да разберем защо е така, да предположим, че имаме пространство от хипотези, съдържащо три хипотези h_1, h_2 и h_3. Нека техните апостериорни вероятности при зададени обучаващи данни са, съответно, 0.4, 0.3 и 0.3. Следователно h_1 е MAP хипотеза. Да предположим, че постъпва нов пример, който се класифицира като положителен от h_1 и като отрицателен от h_2 и h_3. Отчитайки всички хипотези, вероятността че примерът е положителен е 0.4, а че отрицателен – 0.6. Най-вероятната класификация в този случай е отлична от класификацията, генерирана от MAP хипотеза.

В общия случай, най-вероятната класификация на нов пример се получава от комбиниране на предсказанията на всички хипотези, претеглени от техни апостериорни вероятности. Ако възможната класификация на нов пример може да приеме една от стойностите v_j от множеството V, то вероятността $P(v_j|D)$ за коректната класификация на новия пример като v_j е

$$P(v_i \mid D) = \sum_{h_j \in H} P(v_j \mid h_j) P(h_j \mid D)$$

Оптималната (Бейсова) класификация на нов пример е стойността v_j за която $P(v_j|D)$ е максимална:

$$v_j \equiv \arg\max_{v_j \in V} \sum_{h_j \in H} P(v_j \mid h_j) P(h_j \mid D) \quad (10.1)$$

За разгледаме за илюстрация по-горния пример - множеството от възможни класификации на нов пример е V = {+, -} и

$$P(h_1 \mid D) = 0.4 \qquad P(- \mid h_1) = 0 \qquad P(+ \mid h_1) = 1$$
$$P(h_2 \mid D) = 0.3 \qquad P(- \mid h_2) = 1 \qquad P(+ \mid h_2) = 0$$
$$P(h_3 \mid D) = 0.3 \qquad P(- \mid h_3) = 1 \qquad P(+ \mid h_3) = 0$$

Следователно: $\sum_{j=1}^{3} P(+ \mid h_j) P(h_j \mid D) = 0.4$ и $\sum_{j=1}^{3} P(- \mid h_j) P(h_j \mid D) = 0.6$

и $\arg\max_{v_j \in \{+,-\}} \sum_{h_j \in H} P(v_j \mid h_j) P(h_j \mid D) = -$

Всяка система, която класифицира в съгласие с ф-ла (10.1) се нарича *оптимален Бейсов класификатор*. Никакъв друг метод за класификация, използващ същото пространство на хипотези и същите априорни знания, *не може да подобри*, в средно, този метод. Той максимизира вероятността, че новия пример е класифициран правилно при зададени налични данни, пространството от хипотези и априорните вероятности на хипотези.

Например при научаване на двоични понятия с използване на пространството на версии оптималната Бейсова класификация на нов пример се получава чрез претеглено гласуване на всички членове на пространството на версии, където всяка кандидат-хипотеза е претеглена чрез своята апостериорна вероятност.

Една интересна особеност на оптималния Бейсов класификатор е, че неговите предсказания могат да съответстват на хипотеза, която изобщо не се съдържа в H! При използване на уравнение (10.1) класифицирането на примери по този начин не отговаря на начина за класифициране на примери от нито една от хипотезите в H. Един от подходи за третиране на тази ситуация, е да гледаме на оптималния Бейсов класификатор като на един нов класификатор, който ефективно преглежда пространството на хипотези H', отличаващо се от пространството на хипотези H, към което се прилага теоремата на Бейс. В частност, H' включва хипотези, които изпълняват сравнения между линейните комбинации на предсказания от множеството на хипотези в H.

10.2. Алгоритъм на Гибс

Макар, че оптималният Бейсов класификатор постига най-доброто поведение, което може да бъде постигнато при налични обучаващи данни, неговото прилагане е много скъпо от изчислителната гледна точка. Това произлиза от необходимостта да бъдат изчислени апостериорните вероятности на всички възможни хипотези в H, а след това да бъдат комбинирани предсказания на всички хипотези, за да бъде класифициран всеки един нов пример.

Алтернативният, по-малко оптимален метод, е алгоритмът на Гибс (Gibbs), който може да се определи по следния начин:

1. Избери случайно някоя хипотеза h от H съгласно разпределение на апостериорната вероятност в H.
2. Използвай h за да предскажеш класификация на следващия пример x.

Доказано е, че класификационната грешка на алгоритъма на Гибс е, в най-лошия случай, равна на двойна грешка на оптималния Бейсов класификатор (Haussler 1994).

Този резултат има едно интересно следствие за задачата за научаване на понятия. В частност от него следва, че ако се предполага равномерното разпределение на априорните вероятности на хипотези в H, и ако целевите понятия са действително се избират от такова разпределение при своето представяне на самообучаваща се система, то *класификацията на следващият пример съгласно хипотезата, взета по случаен начин от текущото пространство на версиите (съгласно равномерното разпределение) ще доведе до очакваната грешка, която е най-много два пъти по-голяма от тази на оптималният Бейсов класификатор*. Това е отново един пример, как Бейсовият анализ на един не-Бейсов алгоритъм позволява да бъде оценено поведение на този алгоритъм.

10.3. Наивен Бейсов класификатор

Да разгледаме случая, когато примерът x на научаваното понятие се описва с конюнкция от атрибутните стойности, а целевата функция $f(x)$ приема дискретни стойности от крайното множество V. Зададено е множеството от обучаващите примери D и нов пример, подлежащ на класификацията $x = <a_1,...,a_n>$. Алгоритмът за обучение трябва да предскаже стойността на целевата функция, т.е. да класифицира този пример.

Съгласно Бейсовия подход класифицирането на примера се свежда до избор на най-вероятното значение на целевата функция v_{MAP} при зададени атрибутните стойности $<a_1,...,a_n>$, описващи примера:

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1,...,a_n)$$

Използвайки теоремата на Бейс, получаваме:

$$v_{MAP} = \arg\max_{v_j \in V} \frac{P(a_1,...,a_n \mid v_j)P(v_j)}{P(a_1,...,a_n)} = \arg\max_{v_j \in V} P(a_1,...,a_n \mid v_j)P(v_j) \quad (10.2)$$

Сега трябва да опитаме да оценим и двата терма в уравнение (10.2) на основата на обучаващите данни. $P(v_j)$ е лесно да се оцени чрез изчисляване на честотата на срещане на всяка стойност v_j на целевата функция в множеството от обучаващи данни. Обаче, количествената оценка на другия терм по същия начин е практически невъзможна, освен ако не разполагаме с много голямо множество от обучаващите данни. Проблемът е в това, че броят на подобни термове е равен на броя на всички възможни примери (комбинации от атрибутните стойности) умножен по броя на възможните стойности на целевата функция. Следователно, за да получим едно надеждно приближение за подобна оценка трябва да прегледаме няколко пъти всеки възможен пример в пространството от примери.

Един възможен начин да избегнем тези сложности, е да предположим, че атрибутните стойности са *условно независими при зададената стойност на целевия атрибут*, т.е. че при известната стойност на целевата функция вероятността от съществуване на конюнкцията на атрибутните стойностите $a_1,...,a_n$ е равна на произведението на вероятностите на отделните атрибутни стойности:

Условна независимост на атрибути: $P(a_1,...,a_n \mid v_j) = \prod_{i=1}^{n} P(a_i \mid v_j)$

Подставяйки този терм в (10.2) получаваме подход, известен като *наивен Бейсов класификатор:*

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i=1}^{n} P(a_i \mid v_j) \quad (10.3)$$

Обърнете внимание, че в случая на наивния Бейсов класификатор броят на различни терми $P(a_i|v_j)$, чиито стойности трябва да бъдат оценени чрез обучаващите данни, е равен на броя на възможни атрибутни стойности умножен по броя на възможни стойности на целевия атрибут – това е значително по-малко от количество, което е необходимо за оценяването на $P(a_1,..., a_n|v_j)$.

И така, наивният Бейсов подход за обучение съдържа обучаващата фаза, при която се изчисляват приближения за всички $P(v_j)$ и $P(a_i|v_j)$, използвайки честотите на срещане сред обучаващите примери. Например, $P(v_j)$ – като отношение на всички примери с класификацията v_j към броя на всички примери, а $P(a_i|v_j)$ – като отношение на всички примери от класа v_j, имащи стойност на атрибута a_i, към общия брой на примери в класа v_j. Това множество от приближения съответства на научените хипотези. Множеството от научените хипотези се използва за класификация на неизвестен пример съгласно ф-ла (10.3). Ако предположение за условната независимост на атрибути е изпълнено, то класификацията на този наивен Бейсов алгоритъм v_{NB} съвпада с MAP класификацията.

Една интересна особеност на наивния Бейсов метод за обучение, отличаващ го от други обучаващи методи, разгледани до сега, е, че в този метод *няма явно търсене в пространството от хипотези*. Вместо това, хипотезата се формира без търсене, само чрез изчисляване на честоти на срещане на различни комбинации от данни в обучаващите примери.

10.3.1. Илюстративен пример

За илюстрация, да се върнем към вече разгледан по-рано пример за класификацията на дни, когато Христо обича да играе тенис. Обучаващите примери на целевото понятие *Игра-на-тенис* са зададени в следващата таблица:

Ден	*Небе*	*Температура*	*Влажност*	*Вятър*	*Игра-на-тенис*
D1	Слънце	Горещо	Висока	Слаб	не
D2	Слънце	Горещо	Висока	Силен	не
D3	Облаци	Горещо	Висока	Слаб	да
D4	Дъжд	Топло	Висока	Слаб	да
D5	Дъжд	Студено	Нормална	Слаб	да
D6	Дъжд	Студено	Нормална	Силен	не
D7	Облаци	Студено	Нормална	Силен	да

D8	Слънце	Топло	Висока	Слаб	не
D9	Слънце	Студено	Нормална	Слаб	да
D10	Дъжд	Топло	Нормална	Слаб	да
D11	Слънце	Топло	Нормална	Силен	да
D12	Облаци	Топло	Висока	Силен	да
D13	Облаци	Горещо	Нормална	Слаб	да
D14	Дъжд	Топло	Висока	Силен	не

Да разгледаме, как наивният Бейсов класификатор ще класифицира примера

<Небе=Слънце, Температура=Топло, Влажност=Висока, Вятър=Силен>

Нашата задача е да предскажем стойността (*да* или *не*) на целевия атрибут *Игра-на-тенис*.

Прилагането на ф-ла (10.3) към конкретния пример дава:

$$v_{NB} = \arg\max_{v_j \in \{да, не\}} P(v_j) \prod_{i=1}^{4} P(a_i \mid v_j) = \arg\max_{v_j \in \{да, не\}} P(v_j) P(Небе = Слънце \mid v_j)$$
$$P(Температура = топло \mid v_j) P(Влажност = Висока \mid v_j) P(Вятър = силен \mid v_j)$$

За да получим желаната класификация на нас ни са необходими 10 вероятности, които трябва да бъдат оценени на базата на обучаващите данни:

P(Игра-на-тенис=да) = 9/14 = 0.64 *P(Игра-на-тенис=не) = 5/14 = 0.36*

P(Вятър=силен|Игра-на-тенис=да)=3/9 = 0.33
P(Вятър=силен|Игра-на-тенис=не)=3/5 = 0.60
...

И така, пропускайки за краткост имената на атрибути, получаваме:

P(да)P(Слънце|да)P(Топло|да)P(Висока|да)P(Силен|да) = 0.0053

P(не)P(Слънце|не)P(Топло|не)P(Висока|не)P(Силен|не) = 0.0206

Следователно, наивният Бейсов класификатор назначава на примера класификацията *Игра-на-тенис = не*. За да получим условната вероятност на тази класификация, можем просто да нормализираме описаните по-горе стойности до 1:

$$\mathrm{P}(не \mid \text{Слънце}, \text{Топло}, \text{Висока}, \text{Силен}) = \frac{0.0206}{0.0206 + 0.0053} = 0.795$$

10.3.2. Изчисляване на вероятностите

Досега ние изчислявахме вероятностите като пропорцията на броя на сбъдвания на наблюдаваното събитие относително общия брой на възможните събития. Например, в описания по-горе пример ние изчислявахме *P(Вятър = Силен|Игра-на-тенис = не)* чрез пропорцията n_c/n, където $n = 5$ е общия брой на обучаващите примери за които *Игра-на-тенис = не*, а $n_c = 3$ е броят на тези от тях, за които *Вятър = Силен*.

В много случаи използване на пропорцията за оценяване на вероятности води до добри резултати. Обаче това не е така, когато n_c е много малък. За да разберем проблема, да предположим, че истинската (не известна за нас) стойност на *P(Вятър = Силен|Игра-на-тенис = не) = 0.08* и че имаме извадка, съдържаща само 5 обучаващи примера, за които *Игра-на-тенис = не.* Очевидно е, че най-вероятната стойност за n_c трябва да бъде 0. Обаче това води до следните проблеми. Първо, използването на пропорцията n_c/n води до надценяването на истинската стойност на вероятност (защото дори $n_c = 1$ води до приближение *1/5 = 0.2*, което е много над *0.08*). Второ, ако тази оценка е нула, то и общата оценка за вероятността, изчислявана от наивния Бейсов класификатор ще бъде нула за всеки пример, който ще съдържа *Вятър = Силен,* тъй като общата вероятност се изчислява като произведение на отделни условни вероятности.

За да избегнем подобни затруднение се използва така наречено *m-приближение на вероятността:*

$$\frac{n_c + mp}{n + m} \quad (10.4)$$

където n_c и n имат същия смисъл, като и по-рано, p – е априорната оценка на вероятността, която искаме да преценим, а m – е константа, наречена *размер на еквивалентната извадка,* която определя "тежестта" на p относително наблюдаваните данни. Един типичен метод за избор на p при липса на някаква друга информация е предположение за равномерното разпределение на априорната вероятност, т.е. ако един атрибут има k различни стойности, то установяваме $p = 1/k$. Например, при оценяването на *P(Вятър = Силен|Игра-на-тенис = не)* знаем, че атрибутът *Вятър* има две възможни стойности, следователно равномерното разпределения на априорната вероятност води до избора на $p = 0.5$. Обърнете внимание, че ако $m = 0$, то m-приближение е еквивалентно на обикновената пропорция n_c/n. m се нарича размер на еквивалентната извадка защото уравнение (10.4) може да се интерпретира като разширяване на n действителни примера с m виртуални примера, в които наблюдаваната величина е разпределена с вероятността p.

10.4. Пример: Научаване на класификацията на текст

Като една илюстрация за важността на Бейсовите методи за обучение, да разгледаме задачата, в която примерите са текстови документи. Например, искаме

да научим понятието “статии от електронни новини, които смятам за интересни” или “страници от Глобалната мрежа, които обсъждат въпроси на МС”. И в двата случая, ако компютърът научава това целево понятие правилно, той ще може автоматично да филтрира огромен обем от реални текстови документи, за да предложи на потребителя само релевантни документи.

Ще разгледаме един общ алгоритъм за научаване да класифицираме текста, базиран на наивния Бейсов класификатор. Описаният подход е сред най-ефективните съществуващи подходи към този проблем.

Да разгледаме пространството на примери *X*, съдържащо всички възможни *текстови документи* (т.е. всички възможни низове от думи и знаци за пунктуацията с произволна дължина). На нас са дадени обучаващите примери на някоя неизвестна целева функция *f(x),* която приема стойности от едно крайно множество *V*. Задачата е да научим от тези обучаващи примери да предсказваме стойността на целевия атрибут за нови текстови документи. За илюстрация ще разгледаме класификацията на документи само на две групи – *интересни* и *неинтересни*.

Две неща трябва да бъдат решени:

1. Как да представим произволен текстов документ в термините на атрибутните стойности.
2. Как да бъдат оценени вероятностите, необходими за работата на наивния Бейсов класификатор.

Използваният подход е много прост: при зададения текстов документ (като, например параграф на английски, показан долу), се дефинира по един атрибут за всяка позиция на думата в текста, а като неговата стойност – съответната английска дума, намираща се в тази позиция. Така, цитираният параграф може да се опише с 111 позиции на думите. Стойността на първия атрибут е думата ”our”, на втория – “approach”, и т.н. Обърнете внимание, че по-дълги текстови документи изискват по-голям брой атрибути от по-къси документи. Обаче това няма да ни пречи.

> Our approach to representing arbitrary text documents is disturbingly simple: Given a text document, such as this paragraph, we define an attribute for each word position in the document and define the value of that attribute to be the English word found in that position. Thus, the current paragraph would be described by 111 attribute values, corresponding to the 111 word positions. The value of the first attribute is the word “our,” the value of the second attribute is the word “approach,” and so on. Notice that long text documents will require a larger number of attributes than short documents. As we shall see, this will not cause us any trouble.

И така, кодирайки по този начин текстови документи, ние вече можем да приложим наивния Бейсов класификатор. Да предположим, че имаме набор от 700 документа, които са класифицирани като “*неинтересни*”, и други 300, които са класифицирани като “*интересни*”. На нас е предложен някой нов документ, който

трябва да класифицираме като “интересен” или “неинтересен”. Нека този документ е същият параграф, за който вече ставаше дума по-рано. В този случай уравнение (10.3) ни дава:

$$v_{NB} = \underset{v_j \in \{интересни, неинтересни\}}{\arg\max} P(v_j) \prod_{i=1}^{111} P(a_i \mid v_j) = \underset{v_j \in \{интересни, неинтересни\}}{\arg\max} P(v_j) P(a_1 = "our" \mid v_j)$$
$$P(a_2 = "approach" \mid v_j) ... P(a_{111} = "trouble" \mid v_j)$$

Използваното в наивния Бейсов класификатор предположение за условната независимост на атрибути $P(a_1,...,a_{111} \mid v_j) = \prod_{i=1}^{111} P(a_i \mid v_j)$ в нашия контекст означава, че вероятностите на думи, намиращи се в определени позиции, не зависят от думи, намиращи се в други позиции. Очевидно е, че това предположение *е невярно*. Например, вероятността да намерим в някоя позиция дума “learning” (самообучение) е по-голяма, ако преди нея се намира дума “machine” (машинно). Обаче, макар че указаното предположение е невярно, ние нямаме голям избор, тъй като без него броят на вероятности, които трябва да бъдат изчислени, е невероятно голям. За щастие наивният Бейсов класификатор работи учудващо добре и в случаи на очевидното нарушаване на предположение за условната независимост (Domingos & Pazzani 1996).

За да изчислим получения израз, трябва да имаме приближения за стойностите на $P(v_j)$ и $P(a_i = w_k|v_j)$, като чрез w_k тук ще означаваме k-та дума в английския речник. Първият терм може да бъде лесно оценен, като пропорцията на всеки клас в обучаващите данни – $P(интересни) = 0.3$; $P(неинтересни) = 0.7$. Както винаги оценката на условните вероятности (например $P(a_1 = “our”|неинтересни))$ е по-сложна, тъй като трябва да преценим по един такъв терм за всяка комбинация от текстови позиции, английски думи и целевата стойност). В речника на английския език има около 50000 различни думи; имаме 2 стойности на целевия атрибут и 111 текстови позиции в текста, подлежащ на класифициране – от тук следва, че трябва да оценим $2 * 111 * 50000 \approx 10\,000\,000$ комбинации от обучаващите данни.

За щастие, можем да направим още едно разумно предположение, значително намаляващо броя на вероятностите, които трябва да оценим. Ще предположим, че вероятността да срещнем някоя конкретна дума w_k (например “chocolate”) не зависи от конкретната позиция на думата (т.е. a_{23} или a_{96}). Формално това означава, че предполагаме, че атрибутите са независими и равномерно разпределени при зададена целева класификация, т.е., че $P(a_i = w_k|v_j) = P(a_m = w_k|v_j)$ за всички i, j, k, m. По този начин, можем да оценим цялото множество от вероятности $P(a_1 = w_k|v_j)$, $P(a_2 = w_k|v_j)$… чрез една единствена, независима от позиция вероятност $P(w_k|v_j)$. Това означава, че се нуждаем от оценката на само $2 * 50000$ различни стойности за различни $P(w_k|v_j)$. Обърнете внимание, че в случаите, когато обучаващите данни са ограничени по обем, това предположение увеличава броя на наличните примери за оценяване на необходимите вероятности и по този начин увеличава надеждността на тази оценка.

За да завършим проектиране на нашия алгоритъм за самообучение трябва да изберем метода за оценяване на вероятности. Ще използваме *m*-приближение (ф-ла 10.4) с равномерното разпределение на априорните вероятности, а *m* ще изберем равна на размера на речника с думи. Т.е.

$P(w_u \mid v_j) = \frac{n_k + 1}{n + \mid Речник \mid}$, където n – е общият брой позиции на думи във всички обучаващи примери, които имат стойност на целевия атрибут v_j. n_k – е броят на срещания на дума w_k сред тези n позиции, а $|Речник|$ - е общият брой на различни думи, намерени в обучаващите данни.

И така, нашият финален алгоритъм използва наивния Бейсов класификатор заедно с предположението, че вероятността за срещане на дума не зависи от нейната позиция в текста. Пълният текст на алгоритъма е приведен по-долу. При обучение процедурата LEARN_NAÏVE_BAYES_TEXT проверява всички обучаващи документи, за да извлече от тях речника с всички думи и други знаци за пунктуацията, които се появяват в текста. След това тя брои техните честоти за срещане в различни целеви класове, за да изчисли необходимите приближения за вероятностите. По-късно, при постъпване на новия пример, процедурата CLASSIFY_NAÏVE_BAYES използва тези вероятности за изчисляване на v_{NB} съгласно уравнение (10.3). Обърнете внимание, че всяка дума, която се появява в новия документ, но не участва в обучаващите документи, просто се игнорира. Кодът на алгоритъма, заедно с използвани обучаващите данни, можете да намерите в http://www.cs.cmu.edu/~tom/book.html.

LEARN_NAÏVE_BAYES_TEXT(*Примери, V*)

Примери е множество от текстови документи заедно с техните класификации. *V* - е множеството от възможни класификации (стойности на целевия атрибут). Процедурата научава вероятностите $P(w_k|v_j)$, описващи вероятност, че случайно избрана дума от документа с клас v_j ще бъде дума w_k. Тя също така научава и априорните вероятности на класове $P(v_j)$.

1. **Събери всички думи и знаци за пунктуация, намиращи се в *Примери***
   - *Речник* ← множеството от всички различни думи и знаци за пунктуация, срещани в текстови документи на *Примери*
2. **Изчисли необходимите $P(v_j)$ и $P(w_k|v_j)$**
   - За всяка стойност на целевия атрибут v_j направи:
     - $docs_j$ ← подмножество на документи от *Примери*, за които стойността на целевия атрибут е v_j.
     - $P(v_j) \leftarrow \frac{\mid docs_j \mid}{\mid Примери \mid}$
     - $Text_j$ ← един общ документ, получен чрез обединение на всички членове на $docs_j$
     - n ← общия брой на различни позиции на думи в $Text_j$
     - за всяка дума w_k от *Речник* направи:
       - n_k ← броя на срещане на думата w_k в $Text_j$

- $P(w_u \mid v_j) = \dfrac{n_k + 1}{n + \mid Речник \mid}$

CLASSIFY_NAÏVE_BAYES_TEXT(*Документ*)

Предсказва значение на целевия атрибут на некласифициран *Документ*. a_i означава думата, намерена в i-та позиция на *Документ*а.

- *позиции* ← всички позиции на думи от *Документа*, които се срещат в *Речник*.
- Върни v_{NB}, където

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_{i \in позиции} P(a_i \mid v_j)$$

10.5. Резултати от експерименти

Представеният по-горе алгоритъм (с малки изменения) е бил приложен към решаване на задачата за класифициране на статиите в групи по-интереси в Интернет (Joachims 1996). В този случай целевата класификация на една статия е името на групата, в която статията се появява. Тази задача може да се разглежда като задача за създаване на обслужващата програма по поддръжка на групите по интереси, която научава как да разпределя новопостъпващи документи по съществуващите групи. В експериментите са били използвани 20 различни групи. За всяка група са били събрани по 1000 статии, формирайки по този начин колекцията от 20000 документа. 2/3 от тези документи са използвани като обучаващи примери, а останалите 1/3 – като тестови. При 20 групи вероятността за случайно налучкване на правилната класификация е около 5%. Точността, постигната от програмата и измерена върху тестовите примери, е 89%. Използваният алгоритъм е имал само едно отличие от представения по-горе: като стойността на променливата *Речник* в алгоритъма е бил използван размер само на едно подмножество от думи, присъстващи в документите. По-точно, са били изключени 100 най-често срещани в тях думи (такива като “the” и “of”), както и всички думи, които се срещали по-рядко от 3 пъти. Като резултат, полученият *Речник* е съдържал около 38500 думи.

Резюме

- За да изчисли най-вероятната класификация на всеки нов пример оптималният Бейсов класификатор комбинира предсказанията на всички алтернативни хипотези, претеглени от техните апостериорни вероятности.
- Наивният Бейсов класификатор е метод за Бейсово обучение, който е доказал своята приложимост за множество практически задачи. Той се нарича “наивен” защото използва улесняващи предположения, че атрибутните стойности са условно независими при зададена класификация на примера. Когато това предположение е изпълнено, наивният Бейсов класификатор извежда MAP-класификацията. Дори когато това предположение не е изпълнено, използваният класификационен метод остава много ефективен.