**فصل 18- بیز ساده[[1]](#footnote-1)**

**18.0 مقدمه**

قضیه بیز، روشی برتر برای درک احتمال یک رویداد P(A|B)، با توجه به برخی اطلاعات جدید P(B|A)، و یک باور قبلی در مورد احتمال رویداد، P(A) است:

محبوبیت روش بیز در دهه گذشته سر به فلک کشیده است، به طوری که هر روز بیشتر در کاربردهای متداول سنتی در دانشگاه، دولت و تجارت مورد استفاده قرار می‌گیرد. در یادگیری ماشین، یکی از کاربردهای قضیه بیز برای دسته‌بندی به شکل دسته‌بند ساده بیز ارائه می‌شود. دسته‌بند ساده بیز، تعدادی از ویژگی‌های مطلوب را در یک دسته‌بند واحد ترکیب می‌کند. از ویژگی‌های این دسته‌بند می‌توان به موارد زیر اشاره کرد:

1. یک رویکرد شهودی
2. توانایی کار با داده‌های کوچک
3. هزینه‌های محاسباتی کم برای آموزش و پیش‌بینی
4. نتایج خوب در کاربردهای مختلف در اغلب موارد

به طور خاص، یک دسته‌بند ساده بیز بر اساس تابع زیر است:

که:

* احتمال پسین[[2]](#footnote-2) نامیده می‌شود و برابر با احتمال مشاهده‌ی کلاس با توجه به مقادیر مشاهده شده برای ویژگی‌های است. .
* احتمال[[3]](#footnote-3) نامیده می‌شود و برابر با احتمال مقادیر یک مشاهده برای ویژگی‌های ، با توجه به کلاس آنها، یعنی است.
* احتمال پیشین[[4]](#footnote-4) نامیده می‌شود و باور ما به احتمال وجود و وقوع کلاس قبل از مشاهده داده‌ها است.
* احتمال حاشیه‌ای[[5]](#footnote-5) نامیده می‌شود.

در بیز ساده، ما مقادیر پسین یک مشاهده را برای هر کلاس ممکن مقایسه می‌کنیم. به طور خاص، از آنجایی که احتمال حاشیه‌ای در این مقایسه‌ها ثابت است، ما اعداد پسین را برای هر کلاس، مقایسه می‌کنیم. برای هر مشاهده، کلاسی که بیشترین عدد پسین را دارد، کلاس پیش‌بینی شده‌ی می‌شود.

دو نکته مهم در مورد دسته‌بند ساده‌ی بیز وجود دارد. ابتدا، برای هر ویژگی در داده‌ها، باید توزیع آماری احتمال را فرض کنیم. توزیع‌های رایج عبارتند از توزیع‌های نرمال (گاوسی)، چند جمله‌ای و برنولی. توزیع انتخاب شده اغلب بر اساس ماهیت ویژگی‌‌ها (پیوسته، دودویی و غیره) تعیین می‌شود. دوم اینکه، بیز ساده نام خود را به این دلیل گرفته است که ما فرض می‌کنیم هر ویژگی و احتمال نتیجه‌ی آن مستقل است. این فرض «ساده لوحانه[[6]](#footnote-6)» اغلب اشتباه است، اما در عمل زیاد کیفیت دسته‌بند را کاهش نمی‌دهد.

در این فصل ما استفاده از scikit-learn را برای آموزش سه نوع دسته‌بندی کننده ساده بیز با استفاده از سه توزیع احتمال متفاوت پوشش خواهیم داد.

**18.1 آموزش یک دسته‌بند برای ویژگی‌های پیوسته**

**مسئله**

شما فقط ویژگی‌های پیوسته دارید و می‌خواهید یک دسته‌بند ساده‌ی بیز را آموزش دهید.

**راه‌حل**

از یک دسته‌بند بیز ساده گوسی در scikit-learn استفاده کنید:



**بحث**

رایج ترین نوع دسته‌بندی کننده ساده‌ی بیز، بیز ساده گوسی است. در بیز ساده‌ی گاوسی، فرض می‌کنیم که احتمال مقادیر ویژگی ، با توجه به مشاهده از کلاس است که از توزیع نرمال پیروی می‌کند:

که و واریانس و مقادیر میانگین ویژگی برای کلاس هستند. به دلیل فرض توزیع نرمال، بیز ساده گوسی در مواردی که همه ویژگی‌های ما پیوسته هستند بهترین استفاده را دارد.

در scikit-learn، ما یک بیز ساده‌ی گاوسی را مانند هر مدل دیگری با استفاده از fit، آموزش می‌دهیم، و سپس می‌توانیم در مورد کلاس مشاهده، پیش‌بینی کنیم:





یکی از جنبه‌های جالب دسته‌بندی‌کننده‌های ساده‌ی بیز این است که به ما اجازه می‌دهند یک باور قبلی[[7]](#footnote-7) را به کلاس‌های هدف اختصاص دهیم. ما می‌توانیم این کار را با استفاده از پارامتر priors در کلاس GaussianNB انجام دهیم، که فهرستی از احتمالات اختصاص داده شده به هر کلاس از بردار هدف را به عنوان ورودی می‌گیرد:



اگر هیچ آرگومانی به پارامتر priors اضافه نکنیم، prior بر اساس داده‌ها تنظیم می‌شود.

در نهایت، توجه داشته باشید که احتمالات خام پیش‌بینی شده از بیز ساده‌ی گاوسی (که خروجی دریافت شده با استفاده از predict\_proba هستند) کالیبره نشده‌اند. یعنی نباید آنها را باور کرد. اگر بخواهیم احتمالات پیش‌بینی شده‌ی مفیدی ایجاد کنیم، باید آنها را با استفاده از رگرسیون ایزوتونیک یا یک روش مرتبط کالیبره کنیم.

**همچنین ببینید:**

* [چگونه دسته‌بندی کننده ساده بیز در یادگیری ماشین کار می‌کند.](https://dataaspirant.com/naive-bayes-classifier-machine-learning)

**18.2 آموزش یک دسته‌بند برای ویژگی‌های گسسته و شمارشی**

**مسئله**

با توجه به داده‌های گسسته یا شمارشی، باید یک دسته‌بندی کننده ساده بیز را آموزش دهید.

**راه‌حل**

از یک دسته‌بندی کننده ساده چند جمله ای Bayes استفاده کنید:



**بحث**

بیز ساده چند جمله‌ای مشابه بیز ساده گوسی کار می‌کند، اما فرض بر این است که ویژگی‌ها به صورت چند جمله‌ای توزیع شده‌اند. در عمل، این بدان معنی است که این دسته‌بندی معمولاً زمانی استفاده می‌شود که داده‌های مجزا داشته باشیم (به عنوان مثال، رتبه‌بندی فیلم‌ها از 1 تا 5). یکی از رایج‌ترین کاربردهای بیز ساده‌ی چند جمله‌ای، دسته‌بندی متن با استفاده از کیسه‌های کلمات[[8]](#footnote-8) یا رویکردهای tf-idf است (به دستورالعمل‌های 6.8 و 6.9 مراجعه کنید).

در راه حل خود، ما یک مجموعه داده متن اسباب‌بازی[[9]](#footnote-9) از سه مشاهده ایجاد کردیم و رشته‌های متن را به یک ماتریس ویژگی کیسه‌ای از کلمات و یک بردار هدف همراه تبدیل کردیم. سپس از MultinomialNB برای آموزش یک مدل در حالی که احتمالات قبلی را برای دو کلاس (طرفدار برزیل و طرفدار آلمان) تعریف می‌کنیم، استفاده کردیم.

MultinomialNB مشابه GaussianNB عمل می‌کند؛ مدل‌ها با استفاده از fit آموزش داده می‌شوند و مشاهدات را می‌توان با استفاده از predict، پیش‌بینی کرد:





اگر class\_prior مشخص نشده باشد، احتمالات قبلی با استفاده از داده‌ها آموخته می‌شوند. با این حال، اگر بخواهیم یک توزیع یکنواخت به عنوان قبلی استفاده شود، می‌توانیم fit\_prior=False را تنظیم کنیم.

در نهایت، MultinomialNB حاوی یک فراپارامتر هموارکننده‌ی[[10]](#footnote-10) افزودنی، به نام alpha است که باید تنظیم شود. مقدار پیش‌فرض 1.0 است، و مقدار 0.0 به این معنی است که هیچ هموارسازی صورت نمی‌گیرد.

**18.3 آموزش یک دسته‌بند ساده بیز برای ویژگی‌های دودویی**

**مسئله**

شما داده‌های ویژگی‌های دودویی دارید و باید یک دسته‌بندی کننده ساده‌ی بیز را آموزش دهید.

**راه‌حل**

از یک دسته‌بندی کننده ساده برنولی بیز استفاده کنید:



**بحث**

دسته‌بندی‌کننده بیز ساده برنولی فرض می‌کند که همه ویژگی‌های ما دودویی هستند به طوری که فقط دو مقدار را می‌گیرند (به عنوان مثال، یک ویژگی دسته‌بندی اسمی‌که یک‌ بار کدگذاری شده است). مانند پسر عموی چندجمله‌ای[[11]](#footnote-11) خود، بیز ساده‌ی برنولی اغلب در دسته‌بندی متن استفاده می‌شود، زمانی که ماتریس ویژگی ما صرفاً وجود یا عدم وجود یک کلمه در یک سند است. علاوه بر این، BernoulliNB نیز مانند MultinomialNB، دارای یک هایپرپارامتر هموار کننده‌ی افزودنی، به نام alpha است که ما می‌خواهیم با استفاده از تکنیک‌های انتخاب مدل، آن را تنظیم کنیم. در نهایت، اگر بخواهیم از priors استفاده کنیم، می‌توانیم از پارامتر class\_prior با لیستی حاوی احتمالات قبلی برای هر کلاس استفاده کنیم. اگر بخواهیم یک احتمال قبلی یکنواخت را مشخص کنیم، می‌توانیم fit\_prior=False را تنظیم کنیم:



**18.4 کالیبره کردن احتمالات پیش‌بینی شده**

**مسئله**

شما می‌خواهید احتمالات پیش‌بینی شده را از دسته‌بندی کننده‌های ساده‌ی بیز کالیبره کنید تا قابل تفسیر باشند.

**راه‌حل**

از CalibratedClassifierCV استفاده کنید:





**بحث**

احتمالات کلاس، بخش مشترک و مفیدی از مدل‌های یادگیری ماشین هستند. در scikit-learn، اکثر الگوریتم‌های یادگیری به ما امکان می‌دهند تا احتمالات پیش‌بینی شده عضو یک کلاس را با استفاده از predict\_proba ببینیم. این مورد می‌تواند بسیار مفید باشد اگر برای مثال، بخواهیم فقط یک کلاس خاص را پیش‌بینی کنیم در صورتی که مدل احتمال، این را پیش‌بینی کند که آن کلاس بیش از 90٪ است. با این حال، برخی از مدل‌ها، از جمله دسته‌بندی‌کننده‌های ساده‌ی بیز، احتمالاتی را به دست می‌دهند که بر اساس دنیای واقعی نیستند. یعنی، predict\_proba ممکن است پیش‌بینی کند که یک مشاهده 0.70 شانس دارد که یک کلاس خاص باشد، در حالی که واقعیت این است که احتمال واقعی 0.10 یا 0.99 است. به طور خاص در بیز ساده، در حالی که رتبه‌بندی احتمالات پیش‌بینی‌شده برای کلاس‌های هدف مختلف، معتبر است، احتمالات پیش‌بینی‌شده خام تمایل به گرفتن مقادیر شدید نزدیک به ۰ و ۱ دارند.

برای به دست آوردن احتمالات پیش‌بینی شده معنادار، ما نیاز به انجام آنچه کالیبراسیون نامیده می‌شود داریم. در scikit-learn می‌توانیم از کلاس CalibratedClassifierCV برای ایجاد احتمالات پیش‌بینی‌شده با کالیبراسیون خوب با استفاده از اعتبارسنجی متقاطع[[12]](#footnote-12) k-fold استفاده کنیم. در CalibratedClassifierCV از مجموعه‌های آموزشی برای آموزش مدل و از مجموعه داده‌های تست برای کالیبره کردن احتمالات پیش‌بینی شده استفاده می‌شود. احتمالات پیش‌بینی‌شده‌ی به دست آمده از این کالیبراسیون، میانگین k-folds هستند.

با استفاده از این راه حل ما می‌توانیم تفاوت بین احتمالات پیش‌بینی شده خام و احتمالات به خوبی کالیبره شده را ببینیم. در راه حل خود، ما یک دسته‌بندی کننده‌ی ساده‌ی گاوسی بیز ایجاد کردیم. اگر این دسته‌بندی‌کننده را آموزش دهیم و سپس احتمالات کلاس را برای یک مشاهده جدید پیش‌بینی کنیم، می‌توانیم تخمین‌های احتمال بسیار شدید را ببینیم:





با این حال، پس از کالیبره کردن احتمالات پیش‌بینی شده (که در راه حل خود انجام دادیم)، نتایج بسیار متفاوتی دریافت می‌کنیم:





CalibratedClassifierCV دو روش کالیبراسیون ارائه می‌دهد - مدل سیگموئید پلات[[13]](#footnote-13) و رگرسیون ایزوتونیک[[14]](#footnote-14) - که توسط پارامتر method، تعریف شده‌اند. در حالی که فضایی برای پرداختن به جزئیات نداریم؛ زیرا رگرسیون ایزوتونیک ناپارامتریک[[15]](#footnote-15) است، با این حال زمانی که اندازه نمونه بسیار کوچک است (به عنوان مثال، 100 مشاهده) رگرسیون ایزوتونیک تمایل دارد که بیش از حد برازش[[16]](#footnote-16) کند. در این راه حل، از مجموعه داده iris با 150 مشاهدات استفاده کردیم و بنابراین از مدل سیگموئید پلات استفاده کردیم.

1. - Naive Bayes [↑](#footnote-ref-1)
2. - posterior [↑](#footnote-ref-2)
3. - likelihood [↑](#footnote-ref-3)
4. - prior [↑](#footnote-ref-4)
5. - marginal probability [↑](#footnote-ref-5)
6. - naïve [↑](#footnote-ref-6)
7. - prior belief [↑](#footnote-ref-7)
8. Bag of words [↑](#footnote-ref-8)
9. - toy text [↑](#footnote-ref-9)
10. smoothing [↑](#footnote-ref-10)
11. - multinomial cousin [↑](#footnote-ref-11)
12. - cross-validation [↑](#footnote-ref-12)
13. - Platt’s sigmoid model [↑](#footnote-ref-13)
14. - Isotonic regression [↑](#footnote-ref-14)
15. Nonparametric [↑](#footnote-ref-15)
16. - overfit [↑](#footnote-ref-16)