هفته دوم

برای طبقه بندی، علاوه بر روش قبلی یک روش دیگه به اسم naive bayes ارائه شده است.

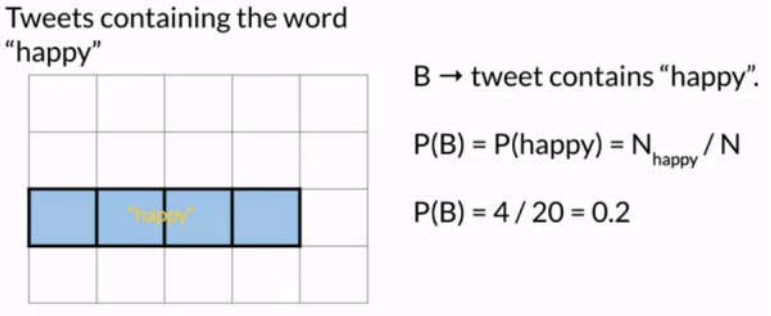
naive bayes classification:

احتمالات، بیس خیلی از اپلیکیشن های NLP است. مثلا برای طبقه بندی توییت به کلاس مثبت و منفی استفاده میشه.

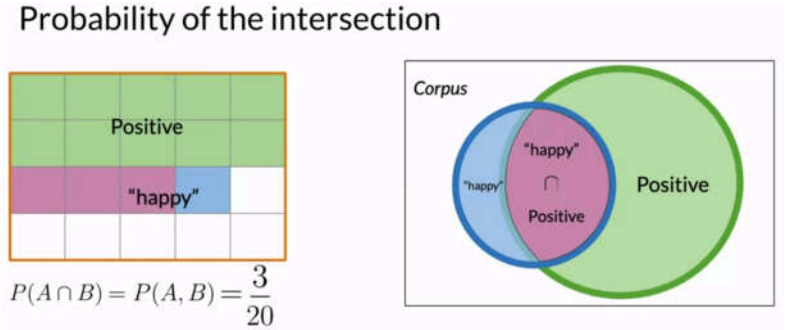
تصور کن یک مجموعه corpus داریم.

فرمول های احتمال:

- احتمال توییت هایی که شامل کلمه happy باشه.



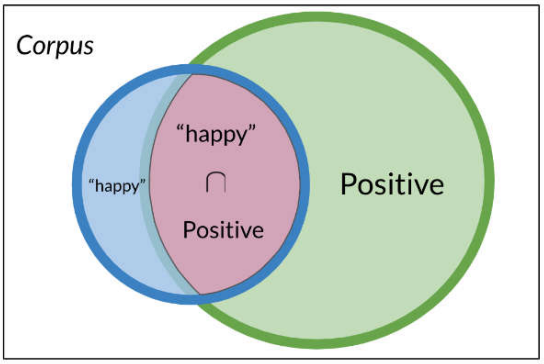
- احتمال تقاطع (intersection): احتمال توییت هایی که کلاس مثبت باشند همچنین دارای کلمه happy است. (overlap events)



- conditional probabilities: اگه من بگم تو کالیفرنیا هستیم و زمستان هست بهتر میتونی درجه هوا را حدس بزنی تا اینکه هیچی نگم.

برای درک Bayes rule ابتدا احتمالات شرطی توضیح بدیم.

احتمال happy بشرط positive بودن، یعنی مطمئنی که positive هست و داخل اون باید احتمال happy رو بدست بیاری.



$$P(\text{Positive}|\text{“happy”}) = \frac{P(\text{Positive} \cap \text{“happy”})}{P(\text{“happy”})}$$

- فرمول احتمال شرطی به صورت زیر نتیجه گیری میشود.

$$P(\text{Positive}|\text{“happy”}) = \frac{P(\text{Positive} \cap \text{“happy”})}{P(\text{“happy”})}$$

- Bayes rule به صورت زیر نتیجه گیری میشود.

$$P(\text{Positive}|\text{“happy”}) = P(\text{“happy”}|\text{Positive}) \times \frac{P(\text{Positive})}{P(\text{“happy”})}$$

هدف گذشته طبقه بندی میکردیم توییت ها را با استفاده از لجستیک رگرسیون، این هفته، همون مسئله را با Naive Bayes حل کنیم.

Naive Bayes یک supervised machine learning است.

چرا بهش naives میگن، چون ویژگی ها را به صورت مستقل از هم در نظر می گیرد. و در واقعیت در اکثر مواقع ویژگی ها مستقل نیستند، با این حال، هنوز هم به عنوان یک روش ساده برای analysis sentiment به خوبی کار می کند.

Analysis sentiment with naive bayes

مراحل:

1. یونیک ورد ها رو جدا میکنیم. که 8 کلمه است.

Vocabulary
I
am
happy
because
learning
NLP
sad
not

- از این مجموعه corpus تعداد 2 تاش کلاس مثبت و دوتاش کلاس منفی است. در هر کلاس منفی و مثبت به صورت جداگانه تعداد دفعاتی که کلمات تکرار شده است را بدست میاریم. در نهایت ما تعداد دفعات تکرار کلمات در هر کلاس به صورت جدول زیر داریم.

word	Pos	Neg
I	3	3
am	3	3
happy	2	1
because	1	0
learning	1	1
NLP	1	1
sad	1	2
not	1	2

2. جمع کلمات در هر کلاس را به صورت مجزا محاسبه میکنیم.

3. احتمال هر کلمه به شرط هر کلاس به صورت جدا محاسبه میشود. مثلا احتمال کلمه I به شرط کلاس منفی

$$p(I|Neg)$$

دقت کن برای محاسبه احتمال، مخرج برابر با Npos و Nneg هست. Npos و Nneg یعنی freq **همه کلمات** در هر کلاس .

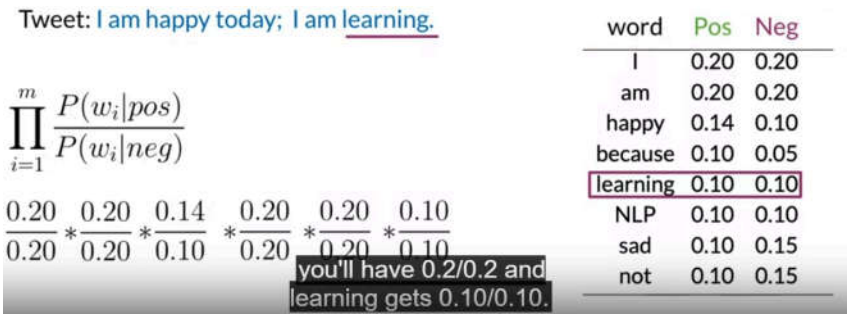
word	Pos	Neg
I	0.24	0.25
am	0.24	0.25
happy	0.15	0.08
because	0.08	0.01
learning	0.08	0.08
NLP	0.08	0.08
sad	0.08	0.17
not	0.08	0.17
Sum	**1**	**1**

4. حالا برای توییت جدید، فرمول زیر محاسبه میشه .

$$\prod_{i=1}^{m} \frac{P(w_i|pos)}{P(w_i|neg)}$$

دقت کن، اگر کلمه ای در توییت جدید چندبار تکرار بشه براش احتمال اش چند بار محاسبه میکنیم.

مثال:



5. **اگه حاصل بزرگتر از یک بشه به کلاس 1 در غیر این صورت به کلاس 0 تعلق دارد.**

نکته) اگر احتمال یک کلمه 0 بشه، حاصل فرمول صفر میشه برای رفع این مشکل چکار کنیم: استفاده از laplacian smooth

Laplacian smooth : یعنی میخوای قسمت هایی از فرمول اصلی را تغییر بدی.

فرمول برای هر کلمه به صورت زیر شد:

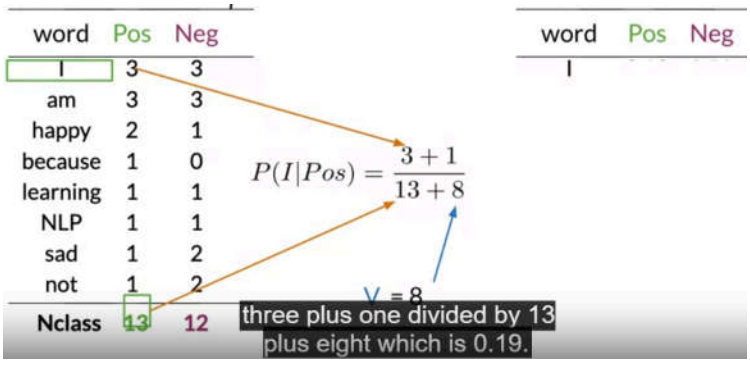
$$P(\mathrm{w_i}|\mathrm{class}) = \frac{\mathrm{freq}(\mathrm{w_i}, \mathrm{class}) + 1}{\mathrm{N_{class}} + \mathrm{V_{class}}}$$

1. از این به بعد همه با عدد یک جمع میشه.
2. Vclass = تعداد کل **کلمات** موجود در corpus برای نرمال شدن اضافه میشه.
3. دقت کن اول صورت و مخرج محاسبه میشه، بعد بر هم تقسیم میشه.

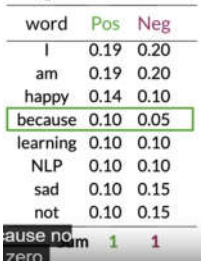
توجه) Nclass کل freq کلمات در کلاس مدنظر (کلاس مثبت جدا، کلاس منفی جدا)

در فرمول قبلی این تغییرات اعمال شده.

مثال:



همچنان جمع احتمال محاسبه شده در هر کلاس در فرمول جدید هم برابر یک میباشد.



word	Pos	Neg
I	0.19	0.20
am	0.19	0.20
happy	0.14	0.10
because	0.10	0.05
learning	0.10	0.10
NLP	0.10	0.10
sad	0.10	0.15
not	0.10	0.15
ause no zero	m	1	1

log likelihood :

فرمول رو میتونیم به صورت لایکلیهود بنویسم،

$$\frac{P(w_i|pos)}{P(w_i|neg)}$$

برای نسبت بالا قانون naive bayes مینویسیم.

و دو طرف معادل رو در (p(neg) / p(pos)) ضرب میکنیم. و به فرمول زیر میرسیم.

$$\frac{P(pos)}{P(neg)}\prod_{i=1}^{m}\frac{P(w_i|pos)}{P(w_i|neg)} > 1$$

و بعد log اضافه میکنیم.

$$log(\frac{P(pos)}{P(neg)}\prod_{i=1}^{n}\frac{P(w_i|pos)}{P(w_i|neg)})$$

حالا هنگامی که داده تست میاد، کلمات داده تست likelihood اگه وجود داشته باشه محاسبه میشه و بخاطر log طبق فرمول پایین تک تک کلمات تست به جای اینکه با هم ضرب بشه، جمع میشه.

$$log\frac{P(pos)}{P(neg)}+\sum_{i=1}^{n}log\frac{P(w_i|pos)}{P(w_i|neg)}$$

log prior + log likelihood

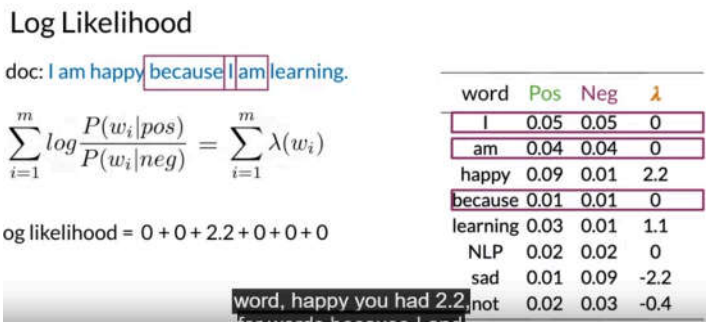
لازم به ذکر است به این قسمت فرمول که پایین نشون داده شده، لامبدا یا log likelihood میگویند.

$$\lambda(w) = log\frac{P(w|pos)}{P(w|neg)}$$

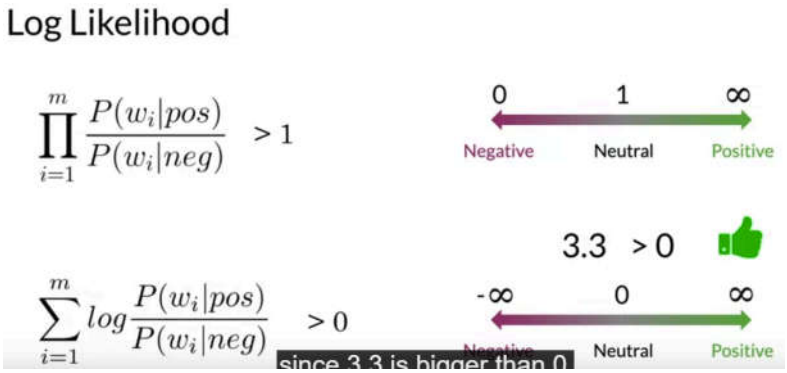
و به این قسمت دیگه فرمول log prior میگویند.

$$\frac{P(pos)}{P(neg)}$$

مثال) نحوه inference گرفتن:



چه مقدار میشد میگفتیم متعلق به کلاس مثبت یا منفی است:



ترین کردن naive bayes classifier :

روش ترین کردن این طبقه بند با naive bayes و دیپ لرنینگ ها متفاوت است. در این روش گرادیان ندارد، مراحل:

1. مجموعه داده (corpus) جمع اوری میشه (تقسیم به دو گروه مثبت و منفی میشه)
2. انجام مراحل پیش پردازش که در هفته اول به ان اشاره شد.
3. محاسبه تعداد تکرار هر کلمه در هر کلاس به صورت جدا. مطابق هفته اول (word frequence)
4. حالا فرمول laplacian smooth رو برای هر کلمه محاسبه میکنیم.

$$P(\mathrm{w_i}|\mathrm{class}) = \frac{\mathrm{freq}(\mathrm{w_i}, \mathrm{class}) + 1}{\mathrm{N_{class}} + \mathrm{V_{class}}}$$

5. سپس log likelihood یا لامبدا را محاسبه میکنیم.

$$\lambda(w) = log \frac{P(\mathrm{w}|\mathrm{pos})}{P(\mathrm{w}|\mathrm{neg})}$$

نکته)

$$log \frac{P(tweet|pos)}{P(tweet|neg)} = log(P(tweet|pos)) - log(P(tweet|neg))$$

6. سپس log prior محاسبه میکنیم.

$$\frac{P(pos)}{P(neg)}$$

خلاصه مطالب گفته شده:

0. Get or annotate a dataset with positive and negative tweets
1. Preprocess the tweets: process_tweet(tweet) → [w_1, w_2, w_3, ...]
2. Compute freq(w, class)
3. Get P(w | pos), P(w | neg)
4. Get λ(w)
5. Compute logprior = log(P(pos) / P(neg))

تست naive bayes classifier : مراحل:

1. دریافت توییت جدید:

Tweet: [I, pass, the, NLP, interview]

2. انجام مراحل پیش پردازش مطابق هفته گذشته
3. این جدول لامبدا از موقع ترین داریم.

word	λ
I	-0.01
the	-0.01
happi	0.63
because	0.01
pass	0.5
NLP	0
sad	-0.75
not	-0.75

4. مقدار لامبدا کلمات توییت جدید به صورت زیر جمع میشه.

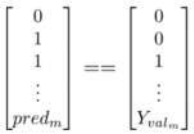
$$score = -0.01 + 0.5 - 0.01 + 0 + logprior = 0.48$$

5. و چون حاصل بیشتر از 0 است پس متعلق به کلاس مثبت است.

برای محاسبه صحت:

1. مجموعه داده تست (X و Y) داریم
2. برای X لامبدا را محاسبه میکنیم
3. اگه مقدار لامبدا برای هر X بزرگتر از 0 بود کلاس مثبت (Y = 1) اگه کوچیک تر از صفر بود کلاس منفی (= Y 0) را مشخص میکنیم.
4. سپس با فرمول زیر صحت را محاسبه میکنیم. (مشابه محاسبه صحت در هفته اول)

$$\frac{1}{m}\sum_{i=1}^{m}(pred_i == Y_{val_i})$$



naive bayes چه کاربردهایی دارد:

1. شناسایی دوتا نویسنده (لازم به ذکر است که از هر نویسنده باید یک مجموعه corpus داشته باشیم.)
2. تشخیص هرز بودن و نبودن ایمیل
3. داکیومنت مربوط یا نامربوط در دیتابیس

فرضیه های naive bayes:

1. در naive bayes کلمات در جملات مستقل هستند. (اما در واقعیت اینجوری نیست)
2. naive bayes به توزیع مجموعه داده های train وابسته است. (دو کلاس باید بالانس باشه در واقعیت اینجوری نیست مثلا توییت مثبت بیشتر از اسپم هست)

آنالیز ارور: از هر روش nlp که استفاده کنی ممکنه بعضی داده ها رو به اشتباه طبقه بندی کند، خوبه به مسائل زیر توجه کنی:

1. ممکنه در مرحله پیش پردازش یک داده مهم به اشتباه حذف بشه. مثلا استیکر غمگین از توییت حذف بشه معنی جمله متفاوت میشه. پس همیشه داده اصلی و پیش پردازش رو بررسی کن.
2. گاهی وقتا نگارش متن خوب نیست و از مدل نمی تونیم انتظار داشته باشیم به خوبی تشخیص دهد. پس داده اصلی رو بررسی کن
3. مدل، طعنه و کنایه رو متوجه نمیشه.