آمار و احتمال مهندسی

اساتید: دکتر توسلی‌پور، دکتر وهابی
دانشکده مهندسی برق و کامپیوتر، دانشکدگان فنی، دانشگاه تهران

تمرین کامپیوتری صفر ـ NumPy، قانون بیز
طراح: فرشته باقری
سوپروایزر: مهدی جمال‌خواه
تاریخ تحویل: ۲۷ مهر ۱۴۰۳

نکات

- هدف تمرین درک عمیق‌تر مفاهیم درس می‌باشد، در نتیجه زمان کافی برای تحلیل کردن نتایج اختصاص دهید.
- در ابتدای همه‌ی سوالات seed را سه رقم آخر شماره دانشجویی‌تان قرار دهید.
- پاسخ تمرین باید به صورت یک فایل زیپ با نام CA0_[Last-Name]_[Student-Id].zip بارگذاری شود. پاسخ سوالات تئوری و تحلیل نتایج‌ها باید به صورت Markdown در فایل Notebook یا در یک فایل pdf که شامل نمودارها و نتایج نیز هست، باشد.

بیشتر بدانیم: طبقه‌بندی

فرض کنید یک سری داده داریم که هر نمونه از آن شامل برداری از ویژگی‌ها (feature) و یک برچسب (label) است. حال هدف ما این است که رابطه‌ی (pattern) بین ویژگی‌ها و برچسب داده‌ها را بدست آوریم تا بتوانیم از روی ویژگی‌های هر نمونه برچسب آن را پیش‌بینی کنیم؛ به بیان دیگر داده‌ها را به دسته‌های مختلف طبقه‌بندی کنیم. به صورت ریاضی:

$$x = [x_1, x_2, \dots x_d] \quad (feature) \quad \Rightarrow \mathbf{f(x)} \Rightarrow \quad y \quad (label)$$

برای مثال، ویژگی می‌تواند قد و وزن فرد باشد و برچسب، مرد یا زن بودن آن فرد باشد. در طبقه‌بندی، هدف یافتن تابع f است. روش‌های گوناگونی برای پیدا کردن این تابع بدست آمده است اما همگی این تابع را تخمین می‌زنند . فرض کنید تابع تخمین زده شده را با $\hat{f}$ نشان می‌دهیم. برای ارزیابی روش‌های متفاوت، ما نیاز داریم به گونه‌ای بررسی کنیم که $\hat{f}$ چقدر به f نزدیک است. یک راه، این است که داده‌هایی را به تابع $\hat{f}$ بدهیم و خروجی آن را با برچسب واقعی که از قبل می‌دانیم مقایسه کنیم و ببینیم که چند درصد از داده‌های ما را به درستی پیش‌بینی شده‌اند. به این پارامتر بدست آمده **دقت (accuracy)** می‌گویند. در ادامه با یک الگوریتم ساده برای تخمین زدن این تابع آشنا خواهیم شد که مفاهیم پایه‌ی آن را در درس فراگرفته‌اید.

طبقه‌بند بیز ساده Naive Bayes Classifier

برای سادگی فرض کنید که برچسب ما باینری باشد ($y \in \{0, 1\}$). طبقه‌بند بیز می‌گوید احتمال تعلق داده به هر کلاس به شرط ویژگی‌های داده شده را بدست بیاور. در نتیجه، داده متعلق به کلاسی است که احتمال آن بیشتر باشد. به صورت ریاضی:

$$P(y=1|x) \overset{?}{>} P(y=0|x) \Rightarrow \begin{cases} Yes: & label = 1 \\ No: & label = 0 \end{cases}$$

برای محاسبه $p(y|x)$ می‌توانیم از رابطه بیز کمک بگیریم و آن را به صورت زیر بازنویسی کنیم:

$$\frac{P(x|y=1)P(y=1)}{P(x)} \overset{?}{>} \frac{P(x|y=0)P(y=0)}{P(x)}$$

$$P(x|y=1)P(y=1) \overset{?}{>} P(x|y=0)P(y=0)$$

طبقه‌بند بیز ساده یک فرض ساده‌ساز را هم به این مساله اضافه می‌کند و فرض می‌کند که ویژگی‌های یک داده از هم مستقل هستند: (دقت کنید که x یک بردار به طول D است.)

$$\prod_{i=۱}^{D}[P(x_i|y=۱)]P(y=۱) \overset{?}{>} \prod_{i=۱}^{D}[P(x_i|y=۰)]P(y=۰)$$

در نتیجه کافیست که $P(y)$ و $P(x_i|y)$ را بدست آوریم. یک روش ساده برای بدست آوردن این دو احتمال برای یک مساله خاص را در سوال ۳ شرح خواهیم داد.

۱. NumPy ۲۵ نمره

در این سوال با مفاهیم ابتدایی NumPy آشنا می‌شوید. فایل numpy_basic.ipynb را دنبال کنید؛ بخش‌های مشخص شده در فایل numpy_basic.py را کامل کنید و توابع آن را در نوت‌بوک تست کنید. (برای این سوال تنها کافیست بخش‌های مشخص شده در فایل پایتون را پیاده‌سازی کنید.)

۲. رای اکثریت ۲۵ نمره

رای اکثریت یک فرآیند تصمیم‌گیری است که در آن انتخابی که بیش از نیمی از آرا را دریافت کند، به عنوان تصمیم نهایی انتخاب می‌شود. این روش به طور گسترده در سیستم‌های دموکراتیک، تصمیم‌گیری گروهی و الگوریتم‌های مختلف یادگیری ماشین مورد استفاده قرار می‌گیرد.

در این سوال می‌خواهیم بررسی کنیم که آیا رای اکثریت منجر به افزایش دقت تصمیم‌گیری می‌شود یا خیر. برای سادگی یک مسئله باینری را در نظر می‌گیریم. (یعنی در آن رای‌دهنده تصمیمی که می‌گیرد یا درست است یا غلط)

فرض کنید هر رای‌دهنده به صورت مستقل و با دقت مشخصی (p) تصمیم **درستی** می‌گیرد.

الف) با نوشتن کد مناسب، در هر کدام از سناریوهای زیر، احتمال درست بودن رای اکثریت را بدست آورید. چه نتیجه‌ای می‌گیرید؟ (فرض کنید که هیچ دانش قبلی‌ای نداریم.)

سناریو	دقت (p)	تعداد آراء "۱"	تعداد آراء "۰"
۱	۰/۷	۸	۴
۲	۰/۷	۱۰	۲
۳	۰/۳	۸	۴
۴	۰/۵	۹	۳
۵	۰/۵	۵	۷

ب) در این بخش از پروژه، شما باید تأثیر افزایش دقت رای‌دهنده (p) بر دقت کلی رای اکثریت را با استفاده از شبیه‌سازی بررسی کنید. فرض کنید ۱۲ رای‌دهنده داریم و هر نفر با دقت p تصمیم می‌گیرد. شما باید شبیه‌سازی‌هایی انجام دهید که در آن‌ها p را از ۰ تا ۱ (با افزایش‌های ۰/۱) تغییر داده، accuracy را محاسبه کنید و نتایج را روی نمودار نشان دهید.

ج) مقدار optimal دقت فردی را مشخص کنید. (منظور از مقدار optimal کمترین مقدار p است که منجر به دقت ٪۱۰۰ برای رای اکثریت می‌شود.)

د) با استفاده از heatmap، تاثیر p و تعداد افراد رای‌دهنده (n) را روی دقت تصمیم‌گیری نشان دهید. محور افقی و عمودی را به ترتیب تعداد افراد و p قرار دهید. همچنین میزان رنگ heatmap نشان دهنده‌ی دقت رای اکثریت خواهد بود. محدوده‌ی مقادیر باید به صورت زیر باشند:

	p	n
شروع	۰	۰
پایان	۱	۵۰
فاصله	۰/۱	۱

۳. تشخیص ایمیل اسپم ۵۰ نمره

در این بخش، هدف ما شناسایی ایمیل‌های اسپم از ایمیل‌های عادی است. مجموعه داده‌ای که استفاده می‌کنیم، شامل دو دسته از ایمیل‌ها است که با ۰ (غیر اسپم) و ۱ (اسپم) مشخص شده‌اند. شما باید از قانون بیز برای تعیین اینکه آیا یک ایمیل جدید اسپم است یا خیر، استفاده کنید. برای این کار از طبقه‌بند بیز ساده استفاده می‌کنیم.

در این مسئله یک فایل emails.csv در اختیار شما قرار داده شده است که شامل محتوای ایمیل‌ها و برچسب آن‌ها است. همان طور که گفته شد، ما باید به ازای هر داده (هر ایمیل) بردار ویژگی را بدست آوریم. یک راه ساده این است که هر کلمه موجود در دیکشنری، که شامل تمام کلمات است، را یک ویژگی در نظر بگیریم که هر ایمیل یا شامل آن کلمه هست یا نیست.

گام اول: پیش‌پردازش داده‌ها (۱۰ نمره)

در گام اول باید متن ایمیل‌های فایل را پیش‌پردازش کنید. برای این کار می توانید از کتابخانه nltk استفاده کنید یا خودتان موارد مورد نیازتان را پیاده‌سازی کنید. در این مرحله باید سعی کنید اطلاعات پیام‌ها را به نحوی مدیریت کنید که به بهترین حالت در پروژه استفاده کنید. به طور مثال، یکی از پیشنهادهای اولیه در این مرحله می‌تواند lowercase کردن و حذف علائم نگارشی و اعداد از هر پیام باشد؛ زیرا این علائم اطلاعات خاصی در مورد نوع پیام به ما نخواهند داد و قابل حذف هستند.

گام دوم: تقسیم به داده آموزش و آزمایش (۵ نمره)

این فرآیند به طور کلی شامل تقسیم داده‌های موجود به دو بخش جداگانه است:

مجموعه آموزش: این بخش از داده‌ها برای پیدا کردن $P(y)$ و $P(x_i|y)$ استفاده می‌شود.

مجموعه آزمایش: این بخش از داده‌ها برای ارزیابی عملکرد مدل استفاده می‌شود. مدل با استفاده از این داده‌ها آزمایش می‌شود تا ببیند چقدر خوب می‌تواند داده‌های جدید را پیش‌بینی کند.

هدف از تقسیم داده‌ها به این دو مجموعه، ارزیابی صحیح عملکرد مدل است. با استفاده از داده‌های آزمایش، می‌توان فهمید که مدل در پیش‌بینی داده‌های جدید چقدر دقیق و معتبر است.

برای اینکار می‌توانید از تابع `train_test_split` از کتابخانه scikit-learn استفاده کنید.

گام سوم: ساخت مدل BoW (۲۰ نمره)

همانطور که گفتیم برای سادگی در اینجا هر کلمه را یک ویژگی در نظر می‌گیریم. به این روش **B**ag **o**f **W**ords می‌گویند . همانطور که از نام این روش مشخص است، فرض می‌کنیم مجموعه‌ای از کلمات داریم که بدون توجه به دستور زبان کنار هم قرار گرفته‌اند. به عنوان مثال به دو جمله زیر دقت کنید:

- جمله‌ی اول: من از غذای این رستوران خوشم آمد.
- جمله‌ی دوم: غذای رستوران خیلی خوب بود ولی رفتار پرسنل خوب نبود.

	من	از	غذای	این	رستوران	خوشم	آمد	خیلی	خوب	بود	ولی	رفتار	پرسنل	نبود
جمله ۱	۱	۱	۱	۱	۱	۱	۱	۰	۰	۰	۰	۰	۰	۰
جمله ۲	۰	۰	۱	۰	۱	۰	۰	۱	۲	۱	۱	۱	۱	۱

همانطور که در بالا مشاهده می‌شود یک BoW تشکیل شد که نشان می‌دهد هر واژه در جمله وجود دارد یا خیر. اگر تعداد زیادی نمونه از این جملات متعلق به دسته‌بندی "اسپم" و "غیر اسپم" را داشته باشیم، می‌توانیم ماتریس BoW را طوری تشکیل دهیم که بعداً بتوانیم از آن برای پیش‌بینی کلاس یا برچسب پیام‌های جدید استفاده کنیم.

در این پروژه BoW بر اساس تعداد تکرار کلمات و بر اساس دسته‌بندی پیام مشخص می‌شود. یعنی در نهایت ابعاد ماتریس BoW حاصل به صورت تعداد کلمات یکتا × ۲ خواهد بود، که تعداد تکرار هر کلمه در هر دسته را به صورت مجزا نشان می‌دهد.

با توجه به این توضیحات ماتریس BoW را برای **مجموعه آموزش** بدست آورید و با استفاده از آن احتمال رخ دادن هر کلمه به شرط اسپم بودن و نبودن ($P(x_i|y)$) و احتمال رخداد ایمیل اسپم و غیر اسپم ($P(y)$) را بدست آورید.

گام چهارم: پیش‌بینی با استفاده از قانون بیز (۲۰ نمره)

حال برای هر داده موجود در **مجموعه آزمایش** با استفاده از احتمالاتی که در مرحله قبل بدست آوردید، کلمات آن ایمیل را بررسی کنید و رابطه زیر را محاسبه کنید و کلاس پیش‌بینی شده را بدست آورید.

$$\prod_{i=۱}^{D}[P(x_i|y=۱)]P(y=۱) \overset{?}{>} \prod_{i=۱}^{D}[P(x_i|y=۰)]P(y=۰) \Rightarrow \begin{cases} Yes: & label = ۱ \\ No: & label = ۰ \end{cases}$$

اینکار را برای تمام ایمیل های موجود در مجموعه آزمایش انجام دهید و نتیجه پیش بینی خود را با برچسب‌های اصلی مقایسه کرده و دقت مدل خود را به دست آورید.

پرسش‌ها

۱. اگر در متن ایمیل کلمه‌ای باشد که در ماتریس BoW وجود ندارد باید چکار کرد؟ صفر در نظر گرفتن احتمال آن یا در نظر نگرفتن آن کلمه چه پیامدی دارد؟ در مورد روش Laplace Smoothing تحقیق کنید و آن را در پروژه خود استفاده کنید. (۱۰ نمره)

۲. اگر متن پیام طولانی باشد، با ضرب شدن احتمال کلمات در هم چه اتفاقی می افتد؟ راه حل شما برای این مشکل چیست؟ نتایج استفاده از این روش را در پروژه‌ی خود گزارش کنید. (۱۰ نمره)

- راهنمایی:

$$\log(P(c \mid X)) \propto \log(P(c)) + \sum_{i=۱}^{n} \log(P(x_i \mid c))$$

۳. یکی از مشکلاتی که باعث می‌شود دقت پیش‌بینی ما کاهش پیدا کند، وجود کلماتی است که در هر متنی ممکن است وجود داشته باشند. کلماتی نظیر حروف اضافه، ضمایر ملکی و ... که به آنها stop words می گویند. به عبارت دیگر، برخی از کلمات به طور مکرر در تمام جملات از دو کلاس و برچسب مختلف تکرار می‌شوند. یعنی با اینکه احتمال وقوع آنها بالاست، اطلاعاتی در مورد برچسب آن جمله به ما اضافه نمی‌کنند. در نتیجه برای افزایش دقت پیش‌بینی، یکی از راه حل ها می تواند حذف این کلمات از BoW باشد. این راه را پیاده‌سازی کرده و نتیجه را با قسمت‌های قبل مقایسه کنید. (۱۰ نمره)