به نام خدا

علم داده – تمرین چهارم **نیایش خانی ۹۹۵۲۱۲۳۵**

سوال ۰۱ Regularization یک تکنیک در یادگیری ماشین است که با اضافه کردن یک سری محدودیت یا جریمه‌ها (penalties) به تابع خطا در حین آموزش مدل، از پیچیدگی بیش از حد مدل جلوگیری می‌کند و باعث می‌شود مدل برای داده‌های جدید بهتر تعمیم دهد. بنابراین از بیش‌برازش (Overfitting) جلوگیری می‌کند.

- **چگونگی کارکرد:**

همانطور که ذکر شد، Regularization تابع ضرر یا تابع هدف را که مدل ما سعی می‌کند در طول فرآیند آموزش به حداقل برساند، اصلاح می کند. تابع ضرر اندازه گیری می کند که مدل ما چقدر با داده های آموزشی مطابقت دارد و چقدر از مقادیر واقعی انحراف دارد. Regularization یک ترم به تابع ضرر اضافه می کند که به اندازه یا بزرگی پارامترهای مدل بستگی دارد. این ترم مدل را به دلیل داشتن مقادیر زیاد یا شدید پارامترها جریمه می کند و آن را به داشتن مقادیر کوچکتر یا ساده تر تشویق می کند. به این ترتیب، مدل ما باید بین برازش داده های آموزشی و کوچک نگه داشتن پارامترها تعادل برقرار کند.
اگر این ترم جریمه را $R(w)$ در نظر بگیریم که پیچیدگی مدل (وزن‌ها w) را محدود می‌کند، تابع خطای جدید به شکل زیر خواهد بود:

$$\lambda R(w) + L(w) = L_{reg}(w)$$

λ پارامتری است که تعادل بین به حداقل رساندن خطا و ساده نگه داشتن مدل را تنظیم می‌کند.

- **چگونه از بیش‌برازش جلوگیری می‌کند؟**

۱. کنترل پیچیدگی مدل : همانطور که پیش‌تر گفته شد، Regularization از بزرگ شدن وزن‌ها جلوگیری می‌کند. وزن‌های بزرگ معمولاً نشان‌دهنده تلاش مدل برای حفظ داده‌های آموزشی (یادگیری نویز) هستند. با محدود کردن وزن‌ها، مدل روی الگوهای اصلی داده تمرکز می‌کند.

۲. همچنین با جلوگیری از پیچیدگی بیش از حد مدل، مدل را ساده‌تر شده و عملکرد آن روی داده‌های دیده‌نشده بهبود می‌یابد.

(در L1 Regularization، وزن‌های مرتبط با ویژگی‌های غیرضروری به صفر می‌رسند و این باعث ساده‌تر شدن مدل و کاهش خطر بیش‌برازش می‌شود.)

سوال ۲. برای L1 Feature Selection مناسب‌تر است زیرا باعث حذف ویژگی‌های غیرضروری می‌شود.

L1 جریمه‌ای را بر اساس مقادیر مطلق ضرایب مدل اضافه می‌کند که باعث ایجاد sparsity در مدل می‌شود، به این معنی که بسیاری از ضرایب دقیقا به صفر می‌رسند. فقط ویژگی‌هایی که تأثیر قابل‌توجهی بر پیش‌بینی متغیر هدف دارند، ضرایب غیرصفر خواهند داشت. ویژگی‌هایی که ضرایبشان صفر می‌شود، عملاً از مدل حذف می‌شوند و ضرایب غیرصفر می‌تواند معیاری برای اهمیت ویژگی‌ها باشد و به اولویت‌بندی آن‌ها کمک کند. این خاصیت L1 را برای انتخاب ویژگی بسیار مؤثر می‌سازد.

از طرفی L2 جریمه‌ای بر اساس مقادیر مربعی ضرایب اضافه می‌کند و اگرچه ضرایب بزرگ را جریمه می‌کند، اما هیچ ضریبی را دقیقاً به صفر نمی‌رساند. در نتیجه ضرایب کاهش می‌یابند اما همه ویژگی‌ها در مدل باقی می‌مانند.

(البته نوع سوم Regularization یعنی Elastic Net مانند L1 ویژگی‌ها را انتخاب می‌کند و در عین حال مانند L2 به همبستگی بین ویژگی‌ها رسیدگی می‌کند بنابراین اگر داده ما دارای ویژگی‌های همبسته باشد، این مدل با ترکیب خاصیت دو مدل قبلی، می‌تواند موثر باشد.)

سوال ۳. هر دو روش‌هایی هستند که به استفاده از دانش یادگرفته‌شده در یک زمینه (منبع) برای بهبود عملکرد در یک زمینه دیگر (هدف) کمک می‌کنند.

Transfer Learning یک مفهوم کلی است که تمرکز آن روی انتقال دانش از یک وظیفه یا دامنه (منبع) به یک وظیفه یا دامنه دیگر (هدف) است برای مثال استفاده از مدلی که روی مجموعه داده‌های ImageNet آموزش دیده است برای دسته‌بندی تصاویر پزشکی.

Domain Adaption نوع خاصی از Transfer Learning است که در آن وظایف منبع و هدف یکسان هستند، اما توزیع داده‌ها در دامنه منبع و هدف متفاوت است مثلا آموزش مدلی روی تصاویر گرفته شده در روز و تطبیق آن برای عملکرد خوب روی تصاویر شب.

- در Transfer Learning تمرکز بر شباهت وظایف است و چالش اصلی انتقال دانش بین وظایفی است که ممکن است ماهیت متفاوتی داشته باشند مثل انتقال دانش از شناسایی اشیاء به تشخیص چهره. همچنین می‌تواند در دامنه‌ها، وظایف یا هر دو تفاوت وجود داشته باشد. اما در Domain Adaption تمرکز بر شباهت وظیفه است یعنی وظیفه یکسان باقی می‌ماند، اما چالش اصلی غلبه بر تفاوت در توزیع داده‌ها بین دامنه‌ها است مانند تطبیق یک مدل تحلیل احساسات که روی نظرات فیلم آموزش دیده است برای استفاده روی نظرات محصولات.
- همچنین می‌توان تکنیک‌های مورد استفاده در هرکدام هم مورد مقایسه قرار داد. در Transfer Learning تنظیم مجدد مدل‌های از پیش آموزش‌دیده یا Fine Tunning استفاده می‌شود که کار آن استفاده از مدل آموزش‌دیده روی یک مجموعه داده بزرگ و تنظیم آن برای یک وظیفه خاص اما در Domain Adaption مثلا از Adversarial Training استفاده می‌شود.
- Transfer Learning معمولا به داده‌های لیبل دار در هر دو وظیفه منبع و هدف نیاز دارد، اما در هدف می‌تواند کمتر باشد و اغلب از یک دامنه منبع با مجموعه داده بزرگ و لیبل دار استفاده می‌کند. اما Domain Adaption فرض می‌کند که داده‌های لیبل دار فقط در دامنه منبع در دسترس هستند، در حالی که دامنه هدف ممکن است بدون لیبل یا با تعداد کمی لیبل باشد.
- با توجه به این تفاوت‌ها، کاربرد آنها نیز متفاوت است. کاربرد Transfer Learning در وظایف مختلف (مانند تشخیص اشیاء به بخش‌بندی تصاویر) و پردازش زبان طبیعی (مانند استفاده از مدل از پیش آموزش‌دیده برای وظایف مختلف متنی) است و کاربرد Domain Adaption در دامنه‌های مختلف (مانند تطبیق سیستم‌های ترجمه ماشینی بین زبان‌های مختلف) و بینایی کامپیوتر (مانند تطبیق بین شرایط مختلف آب و هوایی یا روشنایی).

سوال ۴. پارامتر k در الگوریتم K-Means تعداد خوشه‌ها را مشخص می‌کند و روش‌های مختلفی برای تعیین مقدار بهینه آن وجود دارد.

۱. Elbow Method: در این متد مجموع مربعات فاصله‌ها درون خوشه‌ها را برای مقادیر مختلف k محاسبه می‌شود. سپس نموداری از k(محور افقی) در برابر WCSS (محور عمودی) رسم می‌کند. Elbow point در نمودار جایی است که کاهش WCSS کندتر می‌شود و این نقطه مقدار بهینه k را نشان می‌دهد. البته در برخی داده‌ها elbow point ممکن است به وضوح مشخص نباشد.

۲. Silhouette Score: این روش شباهت یک نقطه به خوشه خودش (انسجام) را نسبت به نزدیک‌ترین خوشه دیگر اندازه‌گیری می‌کند. این امتیاز بین ۱- (خوشه‌بندی ضعیف) و ۱+ (خوشه‌بندی خوب) است. K با بالاترین میانگین این امتیاز انتخاب می‌شود.

۳. Gap Static: تغییرات درون خوشه‌ها را برای مقادیر مختلف k با توزیع مرجع مقایسه می‌کندو k را طوری انتخاب می‌کند که بیشترین گپ بین تغییرات مشاهده‌شده و پیش‌بینی‌شده را داشته باشد.

همچنین متدهای دیگری نیز مانند Domain Knowledge و Cross validation برای دیتاهایی که لیبل دارند استفاده می‌شوند.

- **چگونه در این الگوریتم overfitting رخ می‌دهد؟**

این اتفاق زمانی رخ می‌دهد که خوشه‌ها بیش از حد خاص یا تطبیقی باشند و نتوانند به خوبی داده‌های جدید را تعمیم دهند. رایج ترین دلیل این اتفاق انتخاب K بسیار بزرگ است. وقتی k بیش از حد بزرگ باشد، هر نقطه داده می‌تواند خوشه خودش را تشکیل دهد و واریانس درون خوشه‌ها به صفر می‌رسد. اگرچه این کار تابع هدف را به حداقل می‌رساند، اما باعث اورفیت داده‌ها شده و نویز یا نقاط پرت هم به عنوان خوشه‌های جداگانه در نظر گرفته شده و خوشه‌ها به جای تعمیم‌پذیری، به نویز داده‌ها حساس می‌شوند.

سوال ۵. در یادگیری ماشین، Bias-Variance Tradeoff یک مفهوم اساسی است که تعادل بین دو نوع خطا را توصیف می کند که بر عملکرد یک مدل تاثیر می گذارد:

Bias خطایی است که ناشی از فرضیات بسیار ساده در الگوریتم یادگیری است. بایاس بالا می‌تواند باعث شود که مدل روابط مرتبط بین ویژگی‌ها و خروجی‌های هدف را از دست بدهد و منجر به underfitting شود.

Variance نیز خطایی ناشی از حساسیت مدل به نوسانات کوچک در داده های آموزشی است. واریانس بالا می‌تواند باعث شود مدل نویز تصادفی در داده‌های آموزشی را مدل‌سازی کند که منجر به overfitting می‌شود.

و tradeoff بین این دو مربوط به یافتن بالانس مناسب بین بایاس و واریانس برای به حداقل رساندن خطای کل است. افزایش پیچیدگی مدل باعث کاهش بایس اما افزایش واریانس می‌شود، در حالی که کاهش پیچیدگی باعث افزایش بایاس و کاهش واریانس می‌شود. هدف ما انتخاب مدلی است که پیچیدگی بهینه‌ای داشته باشد و به خوبی به داده‌های جدید و تعمیم پیدا کند.

مثال:

(MAE میانگین تفاوت مطلق بین مقادیر پیش بینی شده و واقعی را محاسبه می کند و MSE میانگین جذر اختلاف بین مقادیر پیش بینی شده و واقعی را محاسبه می کند. MSE با مربع کردن خطاها به خطاهای بزرگتر وزن بیشتری می‌دهد و آن را به خطاهای پرت حساس تر می‌کند.)

اگر فرض کنیم یک مجموعه داده با مقادیر واقعی و پیش بینی از دو مدل داریم که مدل A مدلی ساده با بایاس بالا و واریانس کم و مدل B یک مدل پیچیده با بایاس کم و واریانس بالا باشد و داشته باشیم:

مقادیر واقعی: [3, -0.5, 2, 7]

پیش‌بینی‌های مدل A: [2.5, 0, 2, 8]

پیش‌بینی‌های مدل B: [3.1, -0.6, 1.9, 6.8]

در محاسبات مدل A خواهیم داشت:

$$MAE = \frac{|3-2.5| + |-0.5-0| + |2-2| + |7-8|}{4} = 0.5$$

$$MSE = \frac{(3-2.5)^2 + (-0.5\ -0)^2 + (2-2)^2 + (7-8)^2}{4} = 0.375$$

در محاسبات مدل B خواهیم داشت:

$$MAE = \frac{|3-3.1| + |-0.5-(-0.6)| + |2-1.9| + |7-6.8|}{4} = 0.125$$

$$MSE = \frac{(3-3.1)^2 + (-0.5\ -(-0.6))^2 + (2-1.9)^2 + (7-6.8)^2}{4} = 0.0175$$

همانطور که مشاهده می‌شود در مدل A با MAE و MSE بالاتر، عملکرد نشان‌دهنده بایاس بالاتر است که نشان‌دهنده Underfitting است. در مدل B مقادیر کمتر MAE و MSE نشان می‌دهد که مدل B با داده‌های آموزشی بهتر مطابقت دارد، که نشان‌دهنده بایاس کمتر است. با این حال، بدون ارزیابی روی یک مجموعه اعتبارسنجی جداگانه، نمی‌توانیم تعیین کنیم که آیا مدل B واریانس بالا (اورفیت) داشته است یا خیر.

محدودیت‌های هر متریک:

اگرچه MAE تفسیر ساده‌ای از میانگین بزرگی خطا ارائه می دهد اما خطاهای بزرگتر را به اندازه MSE جریمه نمی‌کند. همچنین در MSE خطاهای بزرگتر به میزان قابل توجهی مجازات می‌شوند، اما به دلیل به توان دو رسیدن خطاها بسیار حساس به outliar ها است.

سوال ۶. خیر، نمی‌توانیم پیچیدگی یک مدل یادگیری ماشین را به‌طور نامحدود افزایش دهیم تا دقت را بالا ببریم. افزایش پیچیدگی مدل ممکن است توانایی مدل در یادگیری داده‌های آموزشی را بهبود بخشد، اما می‌تواند منجر به اورفیت شود. برای دستیابی به دقت مطلوب، تعادل بین پیچیدگی مدل با قابلیت تعمیم ضروری است. عواملی که باید در نظر بگیریم عبارتند از:

۱. Bias-Variance Tradeoff : همانطور که پیش‌تر ذکر شد، بایاس به خطاهای ناشی از مدل‌های بیش از حد ساده اشاره می‌کند و واریانس به خطاهای ناشی از مدل های بسیار پیچیده و حساس به نوسانات داده‌های آموزشی اشاره دارد (بیش از حد) و ایجاد تعادل مناسب بین بایاس و واریانس برای عملکرد بهینه مدل بسیار مهم است.
۲. کیفیت و کمیت داده: داده‌های با کیفیت بالا و کافی برای آموزش مدل‌ها مهم هستند. داده های بیشتر می تواند به تعمیم بهتر مدل های پیچیده کمک کند، اما داده ها باید مرتبط و تمیز باشند.
۳. Feature Engineering: انتخاب ویژگی های مرتبط و تبدیل آنها به طور مناسب می تواند عملکرد مدل را به طور قابل توجهی افزایش دهد. ویژگی‌های نامربوط یا اضافی می‌توانند نویز ایجاد کنند و منجر به اورفیت شوند.
۴. تکنیک‌های Regularization: روش‌هایی مانند منظم‌سازی L1 و L2 جریمه‌ای برای ضرایب بزرگ اضافه می‌کنند.
۵. Cross-Validation: استفاده از متدهای این روش، مانند K-fold، به ارزیابی نحوه تعمیم مدل به مجموعه داده‌های مستقل کمک می‌کند و به تشخیص اورفیت نیز کمک می‌کند.
۶. تنظیم هایپرپارامترها: تنظیم هایپرپارامترها مانند نرخ یادگیری، عمق درخت و ... برای کنترل پیچیدگی مدل و دستیابی به عملکردی خوب، مهم است.

۷. جلوگیری از اورفیت: مانیتور کردن پرفورمنس مدل در دیتای ولیدیشن و داشتن early stop می‌تواند از اورفیت جلوگیری کند. دلیل اهمیت این مورد این است که مدل داده‌های آموزشی را به خاطر نمی‌سپارد، اما یاد می‌گیرد که الگوها را تعمیم دهد.

سوال ۷. Precision ، دقت پیش بینی‌های مثبت را اندازه‌گیری می‌کند و از فرمول زیر به دست می‌آید:

$$Precision = \frac{TP}{TP + FP}$$

Recall هم توانایی مدل را برای شناسایی همه موارد مثبت واقعی ارزیابی می کند و از این فرمول به دست می‌آید:

$$Recall = \frac{TP}{TP + FN}$$

اگر فرض کنیم که یک تست برای شناسایی کرونا گرفته شده است:

مثبت واقعی (TP): افراد بیمار به درستی به عنوان بیمار شناسایی می‌شوند.

مثبت کاذب (FP): افراد سالم به اشتباه به عنوان بیمار شناسایی شده‌اند.

منفی واقعی (TN): افراد سالم به درستی به عنوان سالم شناخته می‌شوند.

منفی کاذب (FN): افراد بیمار به اشتباه به عنوان سالم شناخته شده‌اند.

فرض کنیم ۱۵۰ بیمار داریم که ۱۰۰ تای آنها مبتلا به کرونا و ۵۰ تای آنها سالم هستند. classifier ما ۱۵۰ پیش‌بینی انجام داده است که در مجموع ۱۰۰ پاسخ مثبت و ۵۰ پاسخ منفی بوده است. Confusion Matrix ما نیز به صورت زیر است:

	Predicted No	Predicted Yes
Actual No	TN = 45	FP = 5
Actual Yes	FN = 5	TP = 95

همانطور که مشاهده می‌شود، ۴۵ بیمار به درستی تستشان منفی شده است (TN)؛ تست ۵ بیمار به غلط مثبت شده است (FP)؛ ۵ بیمار کرونا داشته اند اما شناسایی نشدند (FN) و ۹۵ بیمار نیز به درستی شناسایی شده‌اند (TP).

برای محاسبه Precision و Recall خواهیم داشت:

$$Precision = \frac{TP}{TP + FP} = \frac{95}{95 + 5} = 95\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{95}{95 + 5} = 95\%$$

همچنین میزان Accuracy این مدل را نیز می‌توان حساب کرد:

$$Accuracy = \frac{TN + TP}{TN + FP + FN + TP} = \frac{45 + 95}{45 + 5 + 5 + 95} = 93.33\%$$

سوال ۸. الف)

TP: مدل هایی که فروشنده پیشنهاد می کند و خریدار دوست دارد : ۴

FP: مدل هایی را که فروشنده ارائه می دهد اما خریدار دوست ندارد : ۹۶ = ۴ – ۱۰۰

FN: مدل هایی که خریدار دوست دارد اما فروشنده پیشنهاد نمی کند : ۱ = ۴ – ۵

TN: مدل هایی که نه توسط فروشنده ارائه می شود و نه مورد پسند خریدار است. از آنجایی که مدل های بی نهایت کفشی وجود دارد که مورد پسند یا ارائه نشده اند، TN در این مورد تعریف نشده است.

بنابراین ماتریس ما به این صورت خواهد بود:

	Predicted No	Predicted Yes
Actual No	TN	FP=96
Actual Yes	FN=1	TP=4

برای محاسبه Precision و Recall خواهیم داشت:

$$Precision = \frac{TP}{TP + FP} = \frac{4}{4 + 96} = 4\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{4}{4 + 1} = 80\%$$

ب)

TP: مدل هایی که فروشنده پیشنهاد می کند و خریدار دوست دارد: ۳

FP: مدل هایی را که فروشنده پیشنهاد می دهد اما خریدار دوست ندارد : ۱ = ۳ - ۴

FN: مدل هایی که خریدار دوست دارد اما فروشنده پیشنهاد نمی کند = ۰ = ۳ - ۳

TN: مانند مورد اول، TN تعریف نشده است.

بنابراین ماتریس ما به این صورت خواهد بود:

	Predicted No	Predicted Yes
Actual No	TN	FP=1
Actual Yes	FN=0	TP=3

برای محاسبه Precision و Recall خواهیم داشت:

$$Precision = \frac{TP}{TP + FP} = \frac{3}{3 + 1} = 75\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{3}{3 + 0} = 100\%$$

سوال ۹. داشتن یک مدل با این سیستم به دلیل اینکه دیتای ما بسیار imbalanced است، سخت است. برای آموزش و ارزیابی چنین سیستمی، مراحل زیر را انجام می‌دهیم:

۱. Presprocessing: برای هندل کردن دیتایی که imbalanced است، می‌توان دو کار انجام داد. یک oversample کردن از کلاسی است که در اقلیت است. برای این کار می‌توان از تکنیک‌هایی مانند SMOT (Synthetic Minority Over-sampling Technique) استفاده کرد که می‌تواند نمونه‌های Fraud فیک برای متعادل کردن دیتا تولید کند.

راه دوم undersample کردن کلاس اکثریت است که با کاهش تصادفی تعداد تراکنش‌هایی که قانونی هستند، به تعادل کلاس‌ها کمک می‌کند اما باید توجه داشت که ممکن است منجر به از دست رفتن اطلاعات مهم شود.

برای جلوگیری از این اتفاق، راخ حل استفاده از ترکیب این دو روش است.

۲. Feature Engineering: می‌توان الگوهای تراکنش مانند مقادیر تراکنش‌ها، فرکانس و زمان آنها را برای شناسایی anomalyها آنالیز کرد. همچنین می‌توان رفتار کاربر را بررسی کرد و رفتارهای غیرمعمول کاربر مانند مکان‌های غیرمعمول ورود به سیستم و ... را کنترل کنیم.

۳. آموزش مدل: انتخاب من برای آموزش مدل Decision Tree است زیرا هم در هندل کردن دیتاهایی که متعادل نیستند خوب عمل می‌کند و هم در دیتاست‌های بزرگ. الگوریتمی مانند random forest می‌تواند انتخاب خوبی باشد. در طول آموزش باید وزن‌های بالاتری به طبقه اقلیت اختصاص بدهیدم.

۴. ارزیابی مدل: برای ارزیابی مدل accuracy نمی‌تواند معیار مناسبی باشد زیرا مدلی که ۱۰۰٪ مواقع No Fraud را پیش‌بینی می‌کند، accuracy ۹۹.۹۹ درصد را بدون شناسایی هیچ fraudای به دست می‌آورد. در عوض می‌توان از Precision استفاده کرد که نسبت تقلب‌های به درستی شناسایی شده در میان تقلب های پیش بینی شده را به ما می‌دهد. و یا recall که نسبت کلاهبرداری‌های به درستی شناسایی شده در بین تمام کلاهبرداری های واقعی را محاسبه می‌کند. همچنین می‌توان از F1 Score استفاده کرد که میانگین Precision و recall و برای بلنس کردن این دو است.

برای انجام Cross validaion نیز می‌توانیم از K-fold استفاده کنیم. همچنین با بررسی Confusion Matrix می‌توانیم نحوه کارکرد مدل را بررسی کنیم و در صورت نیاز آن را بهبود دهیم.

سوال ۱۰. منحنی ROC یک نمایش گرافیکی است که برای ارزیابی عملکرد یک مدل binary classification با Thresholdهای مختلف استفاده می‌شود. این نمودار نرخ مثبت واقعی (TPR) یا به عبارت دیگر recall را در برابر نرخ مثبت کاذب (FPR) ترسیم می کند.

منحنی ROC، trdaoff بین TPR و FPR را در سطوح Thresholdهای مختلف نشان می‌دهد. مدلی که کاملاً بین کلاس‌ها تمایز قائل شود، یک نقطه در گوشه سمت چپ بالای نمودار خواهد بود (TPR=1، FPR=0) و مدلی که قابلیت تفکیک ندارد، یک خط مورب از پایین به سمت چپ به گوشه بالا سمت راست تولید می‌کند که نشان‌دهنده حدس‌زدن تصادفی است.

AUC یا مساحت زیر منحنی ROC، توانایی کلی مدل را برای تمایز بین طبقات مثبت و منفی quantify می‌کند. مقدار AUC هرچقدر بالاتر باشد، نشان دهنده عملکرد بهتر مدل است.

الف) تفسیر نمرات AUC:

اگر AUC = 0.5 عملکرد مدل معادل حدس زدن تصادفی است و هیچ توانایی تشخیصی را نشان نمی دهد. همچنین در صورتی که

$0.5 < AUC < 1.0$، مدل مقداری توانایی تشخیص دارد. هر چه به ۱ نزدیک تر باشد بهتر است. اگر AUC = 1.0، این مدل کاملا بین کلاس‌های مثبت و منفی تمایز قائل می‌شود.

AUC = 0.7 نشان می‌دهد که مدل می تواند در ۷۰٪ مواقع بین کلاس های مثبت و منفی تمایز قائل شود. اگرچه بهتر از حدس زدن تصادفی است، اما نشان می‌دهد که این مدل ممکن است هنوز تعداد قابل توجهی از نمونه‌ها را اشتباه طبقه‌بندی کند. در مقابل، AUC = 0.9 نشان‌دهنده سطح بالایی از دقت است و مدل به درستی بین کلاس ها در ۹۰٪ مواقع تمایز قائل می شود. چنین مدلی برای اهداف تصمیم گیری قابل اعتماد است و احتمال خطاهای طبقه بندی نادرست کمتر است.

ب) همانطور که در قسمت الف ذکر شد، در منحنی ROC محورها نرخ مثبت واقعی (TPR) در برابر نرخ مثبت کاذب (FPR) هستند که tradeoff بین حساسیت و ویژگی را نشان می دهد و دید جامعی از عملکرد مدل در تمام آستانه‌های طبقه بندی ارائه می دهد.

در طرف دیگر در منحنی PR محورها نمودارهای Precison (نسبت پیش‌بینی‌های‌مثبت واقعی در بین همه پیش‌بینی‌های‌مثبت) را در برابر Recall (یا TPR) در آستانه‌های مختلف ترسیم می‌کند که تعادل بین precision و recall را برجسته می‌کند و بر عملکرد مربوط به کلاس مثبت تمرکز می‌کند.

تفاوت آنها :

- اگر کلاس ما imbalamced باشد، منحنی ROC دیدگاه بسیار خوش بینانه‌ای خواهند داشت زیرا هم موارد منفی واقعی و هم مثبت کاذب را در نظر می‌گیرند. در مقابل، PR صرفا بر روی طبقه مثبت تمرکز می‌کند و هنگام برخورد با چنین دیتایی، اطلاعات بیشتری را به همراه دارند.
- منحنی ROC یک دید کلی از عملکرد مدل ارائه می‌دهند، PR آنالیز دقیق‌تری از اینکه مدل چگونه کلاس مثبت را شناسایی می‌کند، ارائه می دهد.

مثال عملکرد بهتر PR:

به عنوان مثال برای تشخیص Fraud در سیستم‌های مالی، که دیتا بسیار imbalamced است، منحنی PR درک بهتری از مدل در شناسایی Fraud خواهد داشت زیرا براساس precision و recall است و به فراوانی منفی‌های واقعی بی توجه است.

سوال ۱۱. K-fold cross-validation یک تکنیک قوی برای ارزیابی عملکرد مدل های یادگیری ماشینی است که اطمینان حاصل می‌کند که آنها به خوبی به داده‌های دیده نشده تعمیم می‌یابند.

روند آن به این صورت است که ابتدا دیتاها پارتیشن‌بندی می‌شوند یعنی مجموعه داده به k تا زیرمجموعه با اندازه مساوی تقسیم می‌شود که به عنوان "folds" شناخته می شوند.

سپس آموزش و validation آغاز می‌شود به این صورت که این مدل هر بار با استفاده از k-1 فولد برای آموزش و فولد باقی مانده برای validation ، k بار آموزش داده می‌شود. این فرآیند طوری می‌چرخد که هر فولد یک بار به عنوان مجموعه validation عمل کند.

و در مرحله آخر، معیارهای ارزیابی هر دور iteration میانگین گرفته می‌شوند تا عملکرد کلی محاسبه شود.

مزایا نسبت به تقسیم ساده: تمام نقاط دیتا هم برای آموزش و هم برای validation استفاده می‌شود و استفاده از دیتا را به حداکثر می‌رساند. از طرفی با میانگین‌گیری نتایج، تخمین عملکرد دقیق‌تر و قابل اعتمادتری ارائه می‌دهد. همچنین تضمین می‌کند که عملکرد مدل بیش از حد به یک تقسیم‌بندی خاص وابسته نیست.

معایب: افزایش بار محاسباتی دارد زیرا آموزش مدل باید k بار تکرار شود و می‌تواند مخصوصا با دیتاست‌های بزرگ یا مدل های پیچیده، سنگین باشد.همچنین اگر مراحل پیش پردازش داده‌ها به دقت در هر قسمت مدیریت نشود، اطلاعات مجموعه validation می تواند به طور ناخواسته بر روند آموزش تأثیر بگذارد.

انتخاب مقدار k: فرمولی برای محاسبه k وجود ندارد و مقدار آن بستگی به دیتاست دارد. اما رایج‌ترین مقدار k=10 است که می‌تواند بایاس و واریانس را متعادل می کند. برای دیتاست‌های کوچکتر k بزرگتر (مثلا ۲۰) می‌تواند مفید باشد و داده‌های آموزشی بیشتری را در هر iteration ارائه دهد. برای دیتاست‌های بزرگتر هم یک k کوچکتر (مثلا ۵) ممکن است کافی باشد و بار محاسباتی را کاهش دهد.
یک روشی نیز به نام(Leave-One-Out Cross-Validation (LOOCV وجود دارد که k را برابر با تعداد نقاط داده (n) قرار می‌دهد. و هر نقطه داده را به عنوان یک مجموعه validation جداگانه در نظر می‌گیرد. البته با اینکه این کار بایاس را به حداقل می رساند، می‌تواند منجر به واریانس بالا و هزینه های محاسباتی قابل توجهی شود.

سوال ۱۲. برای نشان دادن اینکه سیستم توصیه ما بهتر از رویکرد دستی قبلی است، باید معیارهای زیر را در نظر بگیریم:

قطعا precision و recall اولین معیاری است که در نظر می‌گیریم. زیرا precision نسبت موارد توصیه شده مربوطه را از همه موارد توصیه شده اندازه گیری می کند و recall نیز نسبت اقلام مرتبط توصیه شده از همه موارد مرتبط موجود را اندازه می گیرد.

معیار بعدی می‌تواند Click-Through Rate(CTR) که درصد موارد پیشنهادی که کاربران روی آن کلیک می کنند را اندازه‌گیری می‌کند و نشان‌دهنده میزان مرتبط بودن پیشنهادات است.

همچنین Conversion Rate نیز می‌تواند مفید باشد چراکه درصد کاربرانی که پس از تعامل با محصولات پیشنهادی اقدام به خرید می‌کنند محاسبه می‌کند و این کار پرفورمنس سیستم را در افزایش فروش می‌سنجد.

معیار دیگر RPV است یعنی درآمد به ازای هر بازدید که معیاری کلیدی برای کسب و کار است و کار آن اندازه‌گیری میزان درآمد حاصل از هر بازدید کاربر هنگام تعامل با توصیه‌ها است.

تخمین ROI : برای برآورد مدت زمانی که طول می‌کشد تا هزینه پروژه با سود آن برگردد، متدهای متعددی وجود دارد.

متد اول مشخص کردن میزان هزینه است. برای این کار ابتدا باید سرمایه گذاری اولیه (I) را مشخص کنیم که برای بدست آوردن آن می‌توان از داده‌های مالی تاریخی برای برآورد هزینه توسعه ساخت مدل توصیه استفاده کرد، از جمله: هزینه های نرم افزار و سخت افزار، حقوق دولوپر و دیتاساینتیست‌ها، جمع‌آوری دیتا و هزینه‌های تمیز کردن و ... . هزینه بعدی نیز هزینه‌های عملیاتی (C) یا جاری است مانند هزینه‌های سرور برای هاستینگ سیستم توصیه، هزینه‌های نگهداری منظم و به روزرسانی مدل و ... .

متد بعدی تخمین افزایش درآمد (R) است که برای آن باید درآمد حاصل از فروش اضافی پس از اجرای سیستم جدید را اندازه‌گیری کنیم که این را می توان از AOV(average order value) ، افزایش Conversion Rate و ... بدست آورد.

برای محاسبه مقدار زمان ROI باید محاسبه کنیم که چقدر طول می کشد تا درآمد افزایشی هزینه‌ها را پوشش دهد که از فرمول زیر به دست می‌آید:

$$\text{Payback Period (in months)} = \frac{\text{Initial Investment (I)}}{\text{Monthly Incremental Revenue (R) - Monthly Operational Costs (C)}}$$

به عنوان مثال، اگر سرمایه گذاری اولیه ۵۰۰۰۰ دلار باشد، درآمد افزایشی ماهانه ۱۰۰۰۰ دلار و هزینه های عملیاتی ماهانه ۲۰۰۰ دلار:

$$Playback\ Period = \frac{50000}{10000 - 2000} = 6.25\ months$$