به نام خدا

پروژه SVM(ماشین های بردار پشتیبان)　　نام درس: هوش مصنوعی و سیستم های خبره

نام و نام خانوادگی:سروش میرشکاری　　کد دانشجویی : ۴۰۲۳۰۰۰۳۲

پاسخ‌های پارت اول: آشنایی با SVM خطی

سوال ۱: چرا الان جداساز خطی جواب نمیده؟

جواب ساده‌اش اینه که جداساز خطی، همونطور که از اسمش پیداست، فقط بلده یک خط راست برای جدا کردن کلاس‌ها بکشه. وقتی ما از make_blobs استفاده می‌کردیم، داده‌ها مثل دو تا خوشه یا توده بودن که به راحتی می‌شد یک خط صاف بینشون کشید و از هم جداشون کرد. برای همین دقت مدل نزدیک به ۱۰۰٪ بود.

اما وقتی دیتاست رو به make_moons تغییر دادیم، شکل داده‌ها کاملاً عوض شد. الان داده‌ها شبیه دو تا هلال ماه در هم تنیده هستن. شما هر چقدر هم تلاش کنی، نمی‌تونی یک خط راست پیدا کنی که این دو تا هلال رو به درستی از هم جدا کنه. خط راست نهایتاً می‌تونه یک بخش کوچکی از هر هلال رو درست طبقه‌بندی کنه و بخش بزرگی رو اشتباه می‌گیره. به همین دلیله که دقت مدل به شدت افت می‌کنه (مثلاً نزدیک ۵۰٪ یا حتی کمتر)، چون مدل خطی اساساً برای حل چنین مسئله غیرخطی طراحی نشده و ابزار لازم برای کشیدن یک مرز منحنی رو نداره.

سوال ۲: به نظرت چه جداسازی برای این دیتا مناسب‌تره؟ چرا؟

برای داده‌های غیرخطی مثل make_moons، ما به یک جداساز غیرخطی احتیاج داریم؛ یعنی به جای یک خط راست، به یک مرز تصمیم منحنی نیاز داریم که بتونه خودش رو با شکل پیچیده‌ی داده‌ها وفق بده.

در SVM، این کار رو با استفاده از کرنل‌ها (Kernels) انجام می‌دیم. به جای کرنل linear، باید از کرنل‌های قوی‌تری مثل:

RBF (Radial Basis Function): این معروف‌ترین و پرکاربردترین کرنل برای مسائل غیرخطی هست. کارش اینه که انگار داده‌ها رو به یک بُعد بالاتر می‌بره؛ جایی که جداسازی اونها با یک صفحه (که معادل خط در ابعاد پایین‌تره) ممکن میشه. نتیجه‌اش در فضای اصلی ما، یک مرز تصمیم نرم و کاملاً انعطاف‌پذیره که می‌تونه هر شکلی، از دایره گرفته تا شکل‌های خیلی پیچیده‌تر، به خودش بگیره. برای داده‌های هلالی ما، این کرنل عالی عمل می‌کنه.

چندجمله‌ای (Polynomial): این کرنل هم می‌تونه مرزهای منحنی ایجاد کنه. انگار که به مدل اجازه می‌دیم از معادلات درجه ۲، ۳ یا بالاتر برای کشیدن مرز استفاده کنه.

پس دلیل اینکه این جداسازها مناسب‌ترن اینه که انعطاف‌پذیری لازم برای یادگیری الگوهای پیچیده و غیرخطی رو دارن، کاری که از دست یک مدل خطی ساده برنمیاد.

پاسخ‌های پارت دوم: بازی با کرنل‌ها

سوال ۳: هر کدوم از این فیچرها چه چیزی رو نشون میدن؟

دو ویژگی‌ای که از دیتاست سرطان سینه انتخاب کردیم این‌ها هستن:

(mean smoothness) میانگین نرمی: این ویژگی میگه که کانتور یا حاشیه سلول‌های تومور چقدر صاف و یکنواخته. اگه تغییرات شعاع سلول در نقاط مختلف کم باشه، یعنی سلول صاف‌تر و گردتره. معمولاً سلول‌های سرطانی حاشیه‌های نامنظم‌تر و ناهموارتری دارن.

(mean compactness)میانگین فشردگی: این یک معیار برای سنجش "گرد بودن" شکل سلوله. هر چی سلول به دایره نزدیک‌تر باشه، فشردگی‌اش بیشتره. این ویژگی از طریق فرمولی شامل محیط و مساحت سلول محاسبه میشه و نشون میده که شکل سلول چقدر پیچیده و نامنظمه.

سوال ۴: کدام کرنل بهترین عملکرد را داشت؟ چرا؟

روی دیتاست سرطان سینه (با این دو ویژگی)، کرنل RBF به طور واضح بهترین عملکرد رو داشت. دلیلش اینه که وقتی ما این دو ویژگی رو روی نمودار رسم می‌کنیم، می‌بینیم که مرز بین تومورهای خوش‌خیم و بدخیم یک خط راست و ساده نیست. بلکه یک رابطه پیچیده‌تر و غیرخطی بین نرمی و فشردگی سلول‌ها برای تشخیص بیماری وجود داره.

کرنل خطی چون فقط می‌تونه یک خط راست بکشه، نمی‌تونه این دو گروه رو که کمی در هم تنیده شدن، به خوبی جدا کنه و دقتش پایین‌تره.

کرنل RBF با انعطاف‌پذیری بالایی که داره، می‌تونه یک مرز تصمیم منحنی و نرم ایجاد کنه که به بهترین شکل ممکن دور خوشه‌های هر کلاس می‌پیچه و اون‌ها رو از هم جدا می‌کنه. برای همین دقت بالاتری به دست میاره.

سوال ۵: آیا مدل خطی کافی بود؛ اگر نه شکل داده چه چیزی را نشان می‌دهد؟

قطعاً نه، مدل خطی کافی نبود. دقت پایین‌ترش این رو ثابت می‌کنه. اما مهم‌تر از اون، شکل پراکندگی داده‌ها روی نمودار به ما نشون می‌ده که چرا مدل خطی کافی نیست. وقتی به نمودار نگاه می‌کنیم، می‌بینیم که نقاط آبی (خوش‌خیم) و قرمز (بدخیم) طوری پخش شدن که نمی‌شه با یک خط‌کش یک خط صاف بینشون کشید که کمترین خطا رو داشته باشه. داده‌ها یک الگوی غیرخطی دارن. این یعنی برای جداسازی موفق، ما به یک مرز تصمیم نیاز داریم که بتونه بپیچه و خم بشه، کاری که فقط از عهده کرنل‌های غیرخطی مثل RBF برمیاد.

سوال ۶: اگر تعداد ویژگی‌ها بیشتر بود چه تغییری در انتخاب کرنل رخ می‌داد؟

این سوال خیلی مهمیه. اگه به جای ۲ ویژگی، از تمام ۳۰ ویژگی دیتاست سرطان سینه استفاده می‌کردیم، انتخاب ما ممکن بود تغییر کنه. با افزایش تعداد ابعاد (ویژگی‌ها)، یک پدیده جالب به نام "طلسم ابعاد" (Curse of Dimensionality)رخ می‌ده. در فضاهای با ابعاد خیلی بالا، داده‌ها خیلی پراکنده‌تر میشن و به طرز عجیبی، احتمال اینکه بشه اون‌ها رو با یک صفحه (یعنی یک مرز خطی) از هم جدا کرد، بیشتر می‌شه.

بنابراین:

برای تعداد ویژگی‌های بسیار بالا (مثلاً هزاران ویژگی مثل مسائل تحلیل متن): اغلب کرنل خطی سریع‌تر و حتی گاهی دقیق‌تر عمل می‌کنه. چون هم محاسباتش خیلی سبُک‌تره و هم در اون ابعاد بالا، معمولاً یک جداساز خطی پیدا می‌شه.

برای تعداد ویژگی‌های متوسط (مثل ۳۰ ویژگی دیتاست ما): هنوز هم کرنل RBF یک انتخاب خیلی قوی و مطمئنه، چون روابط بین این ۳۰ ویژگی ممکنه پیچیده و غیرخطی باشه. اما همیشه ارزش داره که کرنل خطی رو هم به عنوان یک گزینه سریع و ساده امتحان کنیم.

پس به طور خلاصه، هر چی تعداد ویژگی‌ها بیشتر بشه، تمایل ما به استفاده از کرنل خطی (به دلیل سرعت و کارایی) بیشتر می‌شه.

پاسخ‌های پارت سوم: بازی با نویز

سوال ۶ و ۷: دقت مدل را با حالت بدون نویز مقایسه کن و تاثیر افزایش نویز را بررسی کن.

همونطور که انتظار می‌رفت، اضافه کردن نویز به داده‌ها مثل اینه که عینک‌مون رو کثیف کنیم؛ دیدن الگوها سخت‌تر می‌شه.

مقایسه با حالت بدون نویز: وقتی نویز (حتی به مقدار کم مثل ۰/۳) اضافه کردیم، دقت تمام مدل‌ها کم شد. دلیلش اینه که نویز بعضی از نقاط رو از جای اصلی‌شون هل می‌ده و ممکنه اون‌ها رو به سمت مرز یا حتی اون طرف مرز کلاس مقابل ببره. این کار مرز بین کلاس‌ها رو مبهم می‌کنه و کار مدل رو برای پیدا کردن یک جداساز تمیز سخت می‌کنه.

تاثیر افزایش نویز: وقتی سطح نویز رو از ۰/۳ به ۰/۵ افزایش دادیم، دقت مدل‌ها باز هم کمتر شد. نویز بیشتر یعنی هرج‌ومرج و همپوشانی بیشتر بین کلاس‌ها. این کار باعث می‌شه مدل در یادگیری الگوی اصلی ضعیف‌تر عمل کنه و خطای بیشتری داشته باشه.

سوال ۸: کدام کرنل بهتر با نویز کنار آمد؟ چرا؟

در این آزمایش، معمولاً کرنل RBF بهترین مقاومت رو در برابر نویز نشون می‌ده. دلیلش به ماهیت انعطاف‌پذیر اما هوشمندش برمی‌گرده:

کرنل خطی چون انعطاف نداره، با اضافه شدن نویز و غیرخطی‌تر شدن داده‌ها، به شدت عملکردش افت می‌کنه.

کرنل چندجمله‌ای (Poly) می‌تونه خیلی حساس باشه. اگه درجه‌اش بالا باشه، ممکنه سعی کنه خودش رو به تک‌تک نقاط نویزی هم Fit کنه و یک مرز خیلی عجیب و غریب و پر پیچ‌وخم بکشه که به این حالت بیش‌برازش (Overfitting)روی نویز میگن و باعث می‌شه روی داده‌های جدید خیلی بد عمل کنه.

کرنل RBF یک تعادل خوب داره. به اندازه کافی انعطاف‌پذیره که بتونه یک مرز غیرخطی مناسب پیدا کنه، اما اونقدر هم حساس نیست که بخواد خودش رو به هر نقطه نویزی بچسبونه. انگار که سعی می‌کنه یک نمای کلی و نرم از مرز رو یاد بگیره، که همین باعث می‌شه در حضور نویز، استوارتر و قابل اعتمادتر از بقیه باشه.

پاسخ‌های پارت چهارم: داده‌های درهم‌تنیده

سوال ۹: کدام کرنل بهتر با داده‌ای در هم کنار آمد؟

وقتی داده‌ها به شدت در هم تنیده و دارای همپوشانی هستن (مثل وقتی که class_sep=0.8 بود)، باز هم کرنل RBF بهترین عملکرد رو داره. در چنین شرایطی، پیدا کردن یک مرز عالی که هیچ خطایی نداشته باشه غیرممکنه. هدف اینه که مرزی پیدا کنیم که بهترین توازن رو بین طبقه‌بندی درست داده‌ها برقرار کنه.

کرنل خطی که کاملاً شکست می‌خوره. اما کرنل RBF می‌تونه یک مرز نرم و غیرخطی ایجاد کنه که سعی می‌کنه تا جای ممکن خودش رو از وسط ناحیه همپوشانی عبور بده و بهترین جداسازی ممکن رو انجام بده.

سوال ۱۰: آیا با تغییر class_sep یا n_informative عملکرد مدل تغییر کرد؟

بله، این پارامترها به شدت روی عملکرد مدل تأثیر دارن:

(class_sep) میزان جدایی کلاس‌ها: این پارامتر مستقیماً سختی مسئله رو تعیین می‌کنه. هر چی این عدد بزرگتر باشه، کلاس‌ها از هم دورتر و جداسازی‌شون راحت‌تره و در نتیجه دقت مدل بالاتر میره. هر چی کوچکتر باشه، همپوشانی بیشتر و کار مدل سخت‌تر می‌شه و دقت پایین میاد.

(n_informative) تعداد ویژگی‌های اطلاعاتی: این پارامتر میگه چند تا از ویژگی‌ها واقعاً برای تشخیص کلاس مفید هستن. اگه این عدد رو کم کنیم، یعنی اطلاعات مفید کمتری به مدل می‌دیم و کارش برای پیدا کردن الگو سخت می‌شه و دقتش کم می‌شه. اگه زیادش کنیم، اطلاعات بیشتری در اختیار مدل قرار می‌گیره و می‌تونه بهتر یاد بگیره (البته تا جایی که تعداد ویژگی‌ها از حد معقولی بیشتر نشه).

پاسخ‌های پارت پنجم: کنترل پیچیدگی با C

سوال ۱۱: بهترین مقدار C کدام بود؟ چرا؟

پارامتر C در SVM مثل یک اهرم کنترل بین دو هدف متضاد عمل می‌کنه:

داشتن یک مرز ساده و با حاشیه امنیت بالا: این باعث می‌شه مدل روی داده‌های جدید خوب عمل کنه (یعنی تعمیم‌پذیری بالایی داشته باشه).

طبقه‌بندی درست تمام نقاط آموزشی: این باعث می‌شه خطای مدل روی داده‌هایی که دیده، کم باشه.

C خیلی کوچک (مثلاً ۰/۰۱): مدل به هدف اول اهمیت بیشتری می‌ده. یعنی یک مرز خیلی ساده و صاف می‌کشه و حاضره چند تا از نقاط آموزشی رو اشتباه طبقه‌بندی کنه. به این حالت کم‌برازش (Underfitting) میگن.

C خیلی بزرگ (مثلاً ۱۰۰۰): مدل به هدف دوم وسواس نشون می‌ده. سعی می‌کنه به هر قیمتی شده تمام نقاط آموزشی رو درست جا بده، حتی اگه مجبور بشه یک مرز خیلی پیچیده و پر پیچ‌وخم بکشه. این مرز روی داده‌های آموزشی عالی عمل می‌کنه، اما روی داده‌های جدید افتضاح خواهد بود. به این حالت بیش‌برازش (Overfitting) میگن.

بهترین مقدار C، مقداریه که بهترین توازن رو بین این دو هدف برقرار کنه و در نهایت بالاترین دقت رو روی داده‌های تست (Test) به ما بده. در این آزمایش، معمولاً یک مقدار میانی مثل C=1 یا C=10 بهترین عملکرد رو داشت چون نه اونقدر ساده بود که دچار کم‌برازش بشه و نه اونقدر پیچیده بود که دچار بیش‌برازش بشه.

سوال ۱۲: نوع غالب خطای مدل در Confusion Matrix چه بود؟ آیا قابل بهبود است؟

برای جواب دادن به این سوال، باید به ماتریس درهم‌ریختگی (Confusion Matrix) نگاه کنیم. این ماتریس به ما میگه مدل چه نوع اشتباهاتی کرده. دو نوع خطای اصلی داریم:

(False Positive) مثبت کاذب: نمونه در واقعیت منفی (مثلاً خوش‌خیم) بوده، اما مدل به اشتباه گفته مثبته (بدخیم).

(False Negative) منفی کاذب: نمونه در واقعیت مثبت (بدخیم) بوده، اما مدل به اشتباه گفته منفیه (خوش‌خیم). این خطا در مسائل پزشکی معمولاً خطرناک‌تره.

اینکه کدام خطا غالب بود، به مقدار C و ماهیت داده‌ها بستگی داره. اما آیا قابل بهبود است؟ بله، قطعاً. راه‌های بهبودش این‌هاست:

تنظیم دقیق‌تر هایپرپارامترها: می‌تونیم با GridSearchCV بهترین مقادیر C و gamma رو پیدا کنیم.

مهندسی ویژگی: شاید با اضافه کردن ویژگی‌های بهتر یا تغییر ویژگی‌های موجود، بشه به مدل کمک کرد.

استفاده از :'class_weight='balanced اگه یک نوع خطا (مثلاً منفی کاذب) برامون مهم‌تره، می‌تونیم با این پارامتر به مدل بگیم که برای اون کلاس اهمیت بیشتری قائل بشه و جریمه اشتباه کردن روی اون کلاس رو سنگین‌تر کنه. این کار باعث می‌شه مدل در تشخیص اون کلاس خاص، دقت بیشتری به خرج بده.

پاسخ‌های پارت ششم: SVM چندکلاسه

سوال ۱۳: مرز بین کدام دو کلاس سخت‌تر بود؟ چرا؟

برای فهمیدن این موضوع، باید به نمودار مرز تصمیم نگاه کنیم. معمولاً مرز بین دو کلاسی که از نظر ویژگی‌ها به هم نزدیک‌تر هستن و نقاطشون در نمودار بیشترین همپوشانی رو با هم دارن، سخت‌ترین مرز برای یادگیریه. در دیتاست Wine، معمولاً کلاس ۱ و ۲ ویژگی‌های شبیه‌تری نسبت به کلاس ۰ دارن. در نتیجه، مدل برای کشیدن یک خط مرزی تمیز و قاطع بین این دو کلاس نزدیک به هم، با چالش بیشتری روبرو می‌شه و احتمالاً ناحیه مرزی بینشون کمی مبهم‌تر و باریک‌تر از مرزهای دیگه است.

سوال ۱۴: آیا مدل توانست سه کلاس را به درستی از هم جدا کند؟ چه کرنلی بهتر جواب داد؟

بله، SVM با استفاده از استراتژی‌های داخلی خودش مثل One-vs-Rest (OvR) یا One-vs-One (OvO) به خوبی می‌تونه مسائل چندکلاسه رو هم مدیریت کنه. در استراتژی OvR، مدل سه تا طبقه‌بند باینری یاد می‌گیره: یکی برای "کلاس ۰ در مقابل بقیه"، یکی برای "کلاس ۱ در مقابل بقیه" و یکی برای "کلاس ۲ در مقابل بقیه". در نهایت برای یک نقطه جدید، هر سه طبقه‌بند نظر می‌دن و هر کدوم که با اطمینان بیشتری گفت این نقطه مال منه، برنده می‌شه.

در این مسئله هم، مثل اکثر مسائل پیچیده، کرنل RBF به دلیل انعطاف‌پذیری بالایی که در ایجاد مرزهای غیرخطی داره، بهترین گزینه بود و تونست با دقت خوبی سه کلاس رو از هم جدا کنه.