به نام خدا

نام و نام خانوادگی: امیرحسین عبدیان

شماره دانشجویی:4021661210

پرسش1 بخش1 :

طبقه‌بندی (Classification)، همان‌طور که از نامش مشخص است، یعنی «دسته‌بندی چیزها» به گروه‌های کوچک‌تر؛ ولی این بار توسط یک ماشین! اگر فکر می‌کنید خیلی جذاب نیست، فقط تصور کنید کامپیوتر می‌تواند تفاوت شما را با یک ناشناس تشخیص دهد یا تمایز یک سیب‌زمینی را از یک گوجه‌فرنگی بفهمد یا فرق بین نمره‌ی A و F را بشناسد.

طبقه‌بندی جزو انواع یادگیری ماشین تحت نظارت است که در آن از داده‌هایی با برچسب برای آموزش استفاده می‌شود. در یادگیری ماشین و آمار، موضوع طبقه‌بندی این است که بر اساس داده‌های آموزشی که قبلاً دسته‌بندی شده‌اند، بفهمیم یک داده جدید به کدام دسته یا گروه از میان مجموعه‌ها متعلق است.

طبقه‌بندی، یک فرایند برای قرار دادن داده‌ها یا اشیاء در دسته‌های معینی است که بر اساس ویژگی‌های آن‌ها تعیین می‌شود. در یادگیری ماشین، طبقه‌بندی یک روش از یادگیری‌ نظارت شده است، جایی که الگوریتم بر روی مجموعه‌ای از داده‌های دارای برچسب آموزش می‌بیند تا بتواند دسته داده‌های جدید و ناشناخته را پیش‌بینی کند. هدف از طبقه‌بندی این است که مدلی بسازیم که قادر باشد به مشاهدات جدید، بر اساس ویژگی‌های آنها، برچسب مناسبی اختصاص دهد. به عنوان مثال، فرض کنید مدلی را بر روی تصاویری که به عنوان سگ یا گربه برچسب‌گذاری شده‌اند آموزش داده‌ایم؛ سپس از این مدل استفاده می‌شود تا تصاویر جدیدی که تا به حال ندیده‌ است را بر اساس ویژگی‌هایی مثل رنگ، بافت و شکل طبقه‌بندی کند.

در طبقه بندی دو کلاسه هدف این است که ورودی را در یکی از دو دسته قرار دهیم. مثلاً، بر اساس وضعیت سلامت فرد، تصمیم می‌گیریم که آیا او به یک بیماری خاص مبتلا است یا نه.

در طبقه بندی چند کلاسه هدف این است که ورودی را در یکی از چند دسته مختلف قرار دهیم. به عنوان مثال، با داشتن اطلاعات مختلف درباره گونه‌های گل‌ها، می‌خواهیم بفهمیم مشاهده ما به کدام گونه متعلق است.

در مدل طبقه بندی خطی یک مرز تصمیم ساده و خطی بین دسته‌ها ترسیم می‌کنند. به خاطر سادگی خود، از نظر محاسباتی بسیار سریع عمل می‌کنند. مثال‌هایی از این نوع:

رگرسیون لجستیک

ماشین‌های بردار پشتیبان با کرنل خطی

پرسپترون تک‌لایه

طبقه‌بندی‌کننده گرادیان تصادفی (SGD)

ارزیابی مدل‌ طبقه‌بندی

ارزیابی مدل طبقه‌بندی گام مهمی در یادگیری ماشین است چرا که کمک می‌کند تا کارایی و توانایی تعمیم مدل روی داده‌های جدید و ندیده‌شده را سنجید. چندین معیار و روش برای ارزیابی یک مدل طبقه‌بندی وجود دارد که بسته به مسئله و نیازهای خاص مورد استفاده قرار می‌گیرند. معیارهای متداول ارزیابی به شرح زیر هستند:

دقت طبقه‌بندی (accuracy)

نسبت تعداد نمونه‌هایی که به درستی طبقه‌بندی شده‌اند به تعداد کل نمونه‌ها در مجموعه آزمون. این معیار ساده و قابل درک است اما در مجموعه‌ داده‌های نامتوازن، که کلاس اکثریت بر دقت حاکم است، ممکن است گمراه کننده باشد.

ماتریس درهم‌ریختگی

جدولی که تعداد مثبت‌های صحیح، منفی‌های صحیح، مثبت‌های غلط و منفی‌های غلط را برای هر کلاس نشان می‌دهد و می‌توان با استفاده از آن، معیارهای ارزیابی مختلفی را محاسبه کرد.

دقت (تشابه) و بازخوانی (Precision and Recall)

دقت نسبت تعداد مثبت‌های صحیح به تعداد کل مثبت‌های پیش‌بینی شده را اندازه‌گیری می‌کند، در حالی که بازخوانی، نسبت مثبت‌های صحیح به تعداد کل مثبت‌های واقعی را اندازه‌گیری می‌کند. این معیارها در مواردی که یک کلاس از دیگری مهم‌تر است یا زمانی که میان مثبت‌های غلط و منفی‌های غلط تعادلی وجود دارد، مفید هستند.

امتیاز F1

میانگین هارمونیکی از دقت و بازخوانی، که به صورت محاسبه می‌شود. این معیار برای مجموعه‌های داده نامتوازن که هر دوی دقت و بازخوانی مهم هستند، مفید است.

منحنی ROC و AUC

منحنی مشخصه عملکرد (ROC) نموداری از نرخ مثبت‌های صحیح (بازخوانی) در مقابل نرخ مثبت‌های غلط (۱-خصوصیت) برای مقادیر آستانه مختلف تابع تصمیم طبقه‌بندی است. مساحت زیر منحنی (AUC) کارایی کلی طبقه‌بندی را اندازه‌گیری می‌کند، با مقادیری که از ۰.۵ (حدس تصادفی) تا ۱ (طبقه‌بندی کامل) متغیر می‌شوند.

اعتبارسنجی متقابل (Cross-Validation)

روشی که داده‌ها را به چندین قسمت تقسیم می‌کند و مدل را برای هر قسمت آموزش می‌دهد، در حالی که روی بقیه آنها آزمون انجام می‌شود، تا بتوان به تخمین موثق‌تری از کارایی مدل دست یافت.

طبقه‌بندی چگونه کار می‌کند؟

در طبقه‌بندی، هدف اصلی این است که یک مدل روی داده‌هایی با برچسب آموزش داده شود تا الگوها و ارتباطات بین ورودی‌ها و برچسب‌های آن‌ها را فهمیده و بیاموزد. وقتی مدل آموزش داده شود، می‌توانیم از آن برای پیش‌بینی برچسب‌ها برای داده‌های جدید و ندیده استفاده کنیم.

فرآیند طبقه‌بندی شامل گام های زیر است:

۱- فهم مسئله

قبل از شروع طبقه‌بندی، باید به درستی مسئله‌ای که می‌خواهید حل کنید را متوجه شوید. کدام برچسب‌ها را می‌خواهید پیش‌بینی کنید؟ چطور داده‌های ورودی با این برچسب‌ها ارتباط دارند؟

به عنوان مثال، فرض کنید می‌خواهیم بر اساس ۷ ویژگی مستقل پیش‌بینی کنیم که آیا یک بیمار، بیماری خاصی دارد یا نه. این به این معناست که دو نتیجه ممکن داریم:

بیمار مبتلا به بیماری است، یعنی “صحیح True”.

بیمار مبتلا نیست، یعنی “غلط False”.

این یک مسئله طبقه‌بندی دودویی است.

۲- آماده‌ سازی داده

وقتی با مسئله آشنا شدید، مرحله بعد آماده‌کردن داده‌هاست. این مرحله شامل جمع‌آوری، پیش‌ پردازش داده ها و تقسیم داده‌ ها به دسته‌های آموزش، اعتبارسنجی و تست می‌شود. اینجا، داده‌ها تمیز و به شکل مناسب برای الگوریتم طبقه‌بندی تبدیل می‌شوند.

X : ویژگی مستقلی است که به صورت ماتریسی با ابعاد N\*M می‌باشد که N تعداد مشاهدات و M تعداد ویژگی‌ها را نمایان می‌کند.

Y : برداری با طول N است که برای هر مشاهده، کلاس پیش‌بینی شده را نشان می‌دهد.

۳- استخراج ویژگی‌ها

ویژگی‌های مؤثر و مرتبط از داده‌ها استخراج می‌شوند تا بین کلاس‌ها تمایز بیافرینیم.

به عنوان مثال، فرض کنید داده‌های ورودی X، شامل ۷ ویژگی مستقل است، اما فقط ۵ ویژگی از آن‌ها بر روی نتیجه تاثیرگذارند و ۲ ویژگی باقی‌مانده همبستگی کم یا بی‌معنی دارند. در این حالت، ما فقط از این ۵ ویژگی برای آموزش مدل استفاده می‌کنیم.

۴- انتخاب مدل

برای طبقه‌بندی، انواع مختلفی از مدل‌ها مانند رگرسیون لجستیک، درخت تصمیم، ماشین‌های بردار پشتیبان و شبکه‌های عصبی در دسترس هستند. انتخاب مدل مناسب برای مسئله شما، با توجه به حجم و پیچیدگی داده‌ها و منابع محاسباتی که دارید، اهمیت دارد.

۵- آموزش مدل

پس از انتخاب یک مدل، مرحله بعدی آموزش آن با داده‌های آموزشی است. در این فرآیند، پارامترهای مدل به نحوی تغییر می‌یابند تا اختلاف بین برچسب‌های پیش‌بینی شده و برچسب‌های واقعی کلاس در داده‌های آموزشی کاهش یابد.

۶- ارزیابی مدل

پس از آموزش مدل، لازم است که عملکرد آن روی یک مجموعه اعتبارسنجی بررسی شود. این ارزیابی به شما دید خوبی از عملکرد مدل بر روی داده‌های جدید و ناشناخته می‌دهد.

لگاریتم خطا یا خطای انتروپی متقابل، ماتریس درهم‌ریختگی، دقت، بازخوانی و منحنی AUC-ROC از معیارهای کیفیتی هستند که برای اندازه‌گیری عملکرد مدل استفاده می‌شوند.

۷- تیونینگ مناسب مدل

اگر عملکرد مدل مطلوب نبود، می‌توانید با تنظیم پارامترها یا امتحان یک مدل متفاوت، آن را بهینه‌سازی کنید.

۸- استقرار مدل

در نهایت، هنگامی که از عملکرد مدل راضی شدیم، می‌توانیم آن را استقرار دهیم تا پیش‌بینی‌هایی روی داده‌های جدید انجام دهد. این مدل می‌تواند برای حل مشکلات واقعی استفاده شود.

حال برای تعیین نوع طبقه بندی از حالت دو کلاسه به چند کلاسه در مرحله فهم مسئله باید مسئله را درست فهمید و مشخص نمایید برای حل مسئله به چند کلاس نیاز دارید و در قسمت آماده سازی داده ها و استخراج ویژگی ها مسئله را با با توجه به فهم حل آن آماده سازی نمایید.

**پرسش1 بخش2 :**

A screenshot of a computer

Description automatically generatedکدی که ارائه کردم از تابع make\_classification از ماژول sklearn.datasets برای ایجاد یک مجموعه داده مصنوعی برای یک مشکل طبقه بندی باینری استفاده می کند. در اینجا یک تفکیک از پارامترهای تابع است:

n\_samples=1000: تعداد کل نمونه هایی که باید تولید شوند.

n\_features=2: تعداد ویژگی ها (ابعاد) در مجموعه داده.

n\_redundant=0: تعداد ویژگی های اضافی که باید اضافه شوند.

n\_clusters\_per\_class=1: تعداد خوشه ها در هر کلاس.

class\_sep=2: ضریب ضرب انحراف استاندارد خوشه برای جداسازی کلاسها.

n\_classes=2: تعداد کلاس ها (طبقه بندی باینری).

random\_state=27: بذری برای مولد اعداد تصادفی که تکرارپذیری را تضمین می کند.

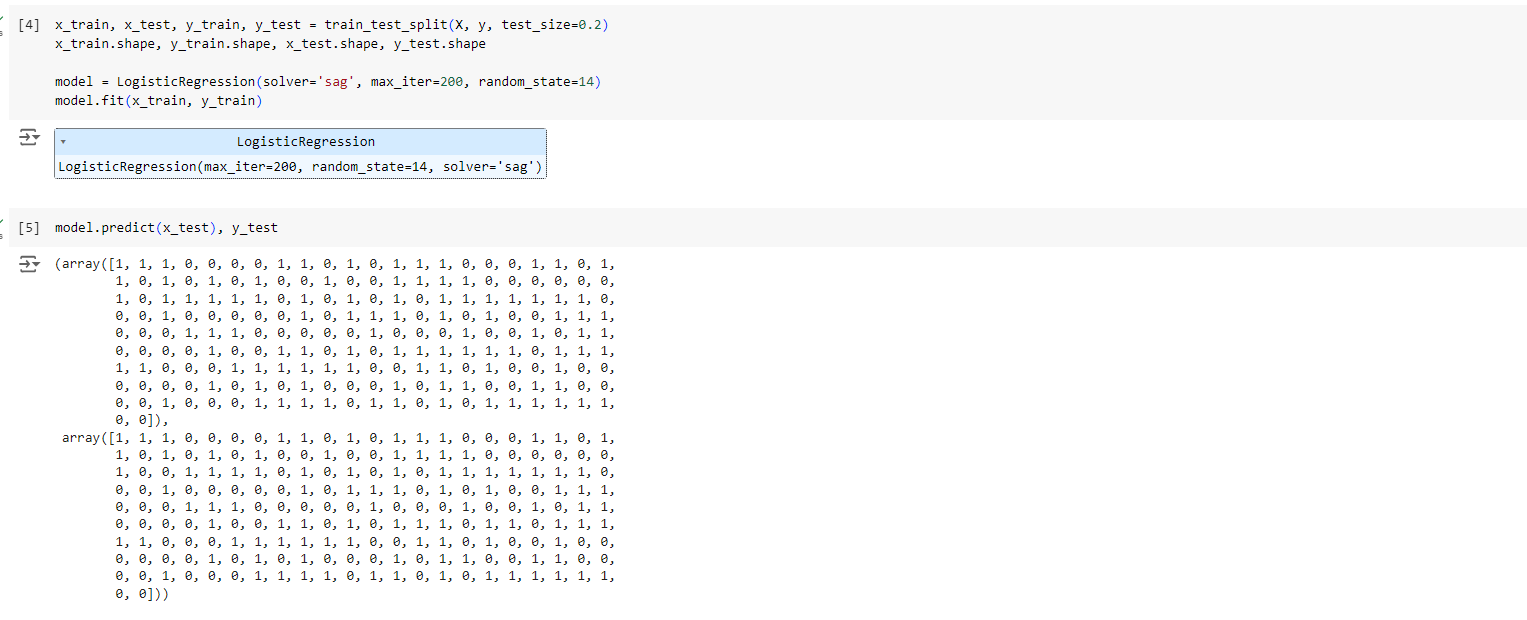
این تابع دو آرایه را برمی گرداند: X (ماتریس ویژگی) و y (برچسب های کلاس هدف).

سپس کد یک رنگ آرایه رنگی با دو رنگ ("آبی" و "قرمز") مربوط به دو کلاس ایجاد می کند. از تابع plt.scatter از کتابخانه matplotlib.pyplot برای ایجاد نمودار پراکندگی استفاده می کند، که در آن محور x نشان دهنده اولین ویژگی (X[:, 0]) و محور y نشان دهنده ویژگی دوم است (X[: ، 1]). پارامتر c برای رنگ آمیزی نقاط داده بر اساس برچسب کلاس آنها (colors[y]) استفاده می شود.

در نهایت، ()plt.show برای نمایش نمودار پراکندگی فراخوانی می شود.

در نگاه اول مدل ما از سختی زیادی برخوردار نیست ولی برای چالش برانگیزتر شدن مسئله میتوان بر روی فاکتورهایی نظیر features، class\_sep و ... تغییراتی اعمال نمود تا مدل سخت تر گردد.

**پرسش1 بخش3 :**



کدی که ارائه کردم مربوط به آموزش و ارزیابی مدل یادگیری ماشین با استفاده از کتابخانه scikit-learn و از طبقه بندی کننده LogisticRegression میباشد. داده ها را به مجموعه های آموزشی (80٪) و تست (20٪) تقسیم کرده ام. کد model.predict(x\_test) برای ایجاد پیش‌بینی در داده‌های آزمون x\_test با استفاده از مدل آموزش‌دیده استفاده می‌شود. y\_test حاوی برچسب های واقعی برای داده های تست است.

A white background with a black border

Description automatically generated with medium confidence A white background with black text

Description automatically generated

model.predict\_proba(x\_test) و model.predict\_log\_proba(x\_test) روشی است که در یادگیری ماشین برای بدست آوردن احتمالات کلاس پیش بینی شده برای داده های آزمون x\_test از یک مدل طبقه بندی آموزش دیده استفاده می شود. در اینجا توضیح مختصری از تفاوت های کلیدی بین model.predict\_proba (x\_test) و model.predict\_log\_proba (x\_test) آورده شده است:

predict\_proba(x\_test) احتمالات پیش بینی شده را برای هر کلاس به طور مستقیم برمی گرداند. خروجی یک آرایه دو بعدی با شکل است (n\_samples، n\_classes)، که در آن هر سطر حاوی احتمالات برای هر کلاس است.

predict\_log\_proba(x\_test) لگاریتم احتمالات پیش بینی شده را برای هر کلاس برمی گرداند. شکل خروجی مانند predict\_proba است.

predict\_log\_proba هنگام برخورد با احتمالات بسیار کوچک ثبات عددی بهتری را فراهم می کند، زیرا لگاریتم یک عدد کوچک یک مقدار منفی بزرگتر است که می تواند با دقت بیشتری نمایش داده شود.

احتمالات لاگ از predict\_log\_proba اغلب در محاسبه تلفات لاگ (از دست دادن آنتروپی متقابل)، یک معیار ارزیابی رایج برای مدل‌های طبقه‌بندی استفاده می‌شود.

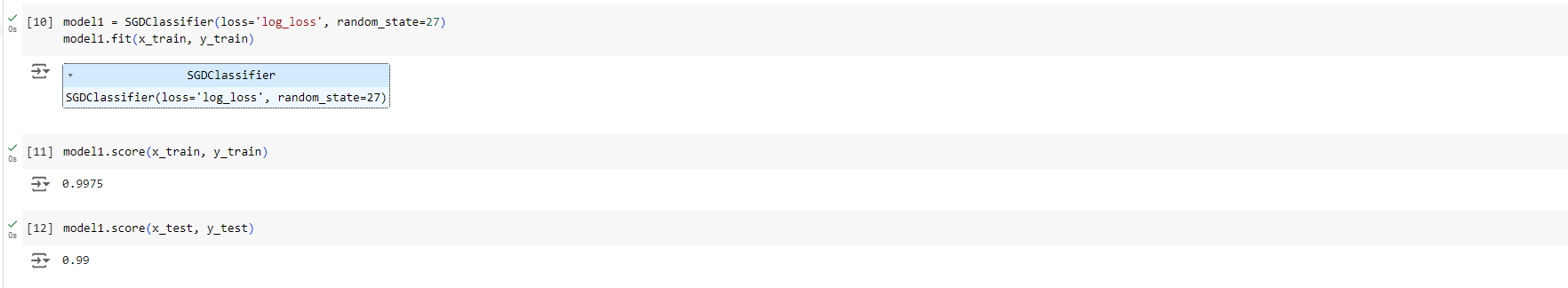
predict\_proba قابل تفسیرتر است زیرا به طور مستقیم احتمالات پیش بینی شده را نشان می دهد، در حالی که predict\_log\_proba برای به دست آوردن احتمالات واقعی نیاز به یک مرحله اضافی (افزایش احتمالات log) دارد.

به طور خلاصه، predict\_proba احتمالات خام را برمی‌گرداند، در حالی که predict\_log\_proba احتمالات گزارش را برمی‌گرداند، که می‌تواند برای ثبات عددی و محاسبه ضرر گزارش مفید باشد، اما به یک مرحله اضافی برای تفسیر نیاز دارد.



ر اینجا توضیح مختصری از model.score() در scikit-learn آورده شده است:

برای مدل‌های طبقه‌بندی، model.score(X\_test, y\_test) امتیاز دقت را محاسبه می‌کند - کسری از پیش‌بینی‌های صحیح انجام شده توسط مدل در داده‌های آزمون X\_test در مقایسه با برچسب‌های واقعی y\_test و بالعکس داده‌های تمرین x\_train در مقایسه با برچسب‌های واقعی y\_train .



کدی که ارائه کردم مربوط به آموزش و ارزیابی طبقه‌بندی‌کننده گرادیان تصادفی (SGD) در scikit-learn است. این خط نمونه ای از SGDClassifier را از scikit-learn ایجاد می کند که یک طبقه بندی خطی است که از الگوریتم Stochastic Gradient Descent برای بهینه سازی استفاده می کند. پارامتر ضرر تابع ضرر را مشخص می کند که باید بهینه شود، در این مورد، 'log\_loss' (همچنین به عنوان از دست دادن رگرسیون لجستیک یا از دست دادن آنتروپی متقابل شناخته می شود). پارامتر random\_state برای تکرارپذیری نتایج روی 27 تنظیم شده است.

**پرسش1 بخش4 :**

کدی که ارائه کردم برای آموزش یک مدل یادگیری ماشین، ایجاد یک شبکه شبکه ای از نقاط در فضای ویژگی، ارزیابی عملکرد تصمیم گیری مدل در شبکه مش، و تجسم داده های ورودی و مرز تصمیم استفاده می شود.

به طور خلاصه، داده های ورودی، مرز تصمیم و نحوه طبقه بندی مدل نقاط در فضای ویژگی را به تصویر می کشد.

A screenshot of a computer

Description automatically generated

A white background with black and white clouds

Description automatically generated

این خطوط یک شبکه مشبک به ابعاد 500×500 نقطه در فضای ویژگی ایجاد می‌کنند که برای تجسم مرز تصمیم‌گیری مدل آموزش‌دیده استفاده می‌شود.

این خطوط تابع تصمیم مدل آموزش دیده را روی نقاط شبکه مش ارزیابی می کنند.

این خطوط یک نمودار پراکنده از داده ورودی X ایجاد می کنند، جایی که نقاط با توجه به برچسب های هدف مربوطه خود y (آبی یا قرمز) رنگ می شوند. تابع plt.contour مرز تصمیم مدل آموزش دیده را به صورت خطوط کانتور در سطوح -1، 0 و 1 ترسیم می کند. توری. در نهایت، ()plt.show نمودار را نمایش می دهد.

تفاوت اصلی با کد قبلی، پارامتر سطوح در تابع plt.contour است. به جای ترسیم یک خط منفرد در سطح 0، اکنون سه خط کانتور در سطوح 1-، 0 و 1 ترسیم می کند. از مدل آموزش دیده

با تجسم خطوط کانتور چندگانه، می‌توانید مرزهای تصمیم‌گیری و اینکه چگونه مدل کلاس‌ها یا مناطق مختلف را در فضای ویژگی جدا می‌کند را بهتر درک کنید. تفسیر خاص خطوط کانتور به نوع مدل یادگیری ماشین و تابع تصمیم مورد استفاده بستگی دارد.

A screenshot of a computer

Description automatically generated

از تابع plot\_decision\_regions از کتابخانه mlxtend برای تجسم مرزهای تصمیم یک مدل یادگیری ماشین آموزش دیده استفاده می کند.

در اینجا خلاصه ای از آنچه این کد انجام می دهد آورده شده است:

از mlxtend.plotting import plot\_decision\_regions تابع plot\_decision\_regions را از ماژول mlxtend.plotting وارد می کند.

plot\_decision\_regions(X, y, clf=model) تابع plot\_decision\_regions را با آرگومان های زیر فراخوانی می کند:

X: داده های ورودی یا ماتریس ویژگی.

y: برچسب ها یا کلاس های هدف.

clf=model: شی مدل یادگیری ماشین آموزش دیده.

تابع plot\_decision\_regions یک ابزار مناسب است که توسط کتابخانه mlxtend برای تجسم مرزهای تصمیم یک طبقه‌بندی کننده یا مدل رگرسیون ارائه شده است. به طور خودکار ایجاد یک شبکه مش را کنترل می کند، پیش بینی های مدل یا عملکرد تصمیم گیری را در شبکه مش ارزیابی می کند، و مناطق تصمیم را با استفاده از رنگ ها یا سایه های مختلف ترسیم می کند.

این تابع می تواند به ویژه هنگام کار با داده های دو بعدی (یعنی دو ویژگی) مفید باشد، زیرا یک نمایش بصری واضح از نحوه جداسازی یا طبقه بندی مناطق مختلف در فضای ویژگی مدل را ارائه می دهد.

با استفاده از این تابع، نیازی به نوشتن کد ایجاد شبکه مش، ارزیابی مدل در شبکه مش و ترسیم مرزهای تصمیم به صورت دستی ندارید. تابع plot\_decision\_regions این مراحل را برای شما انجام می دهد و تجسم و درک رفتار مدل آموزش دیده شما را آسان تر می کند.

**پرسش1 بخش5 :**

با توجه به راهنمای داخل github ابتدا شروع به نصب کتابخانه drawdata در گوگل کولب مینماییم. سپس شروع به نوشتن کدها مینماییم.

A screenshot of a computer

Description automatically generated