گزارش کار مینی پروژه شماره دوم

استاد درس: دکترمهدی علیاری

دانشجو: سمانه اعلائی

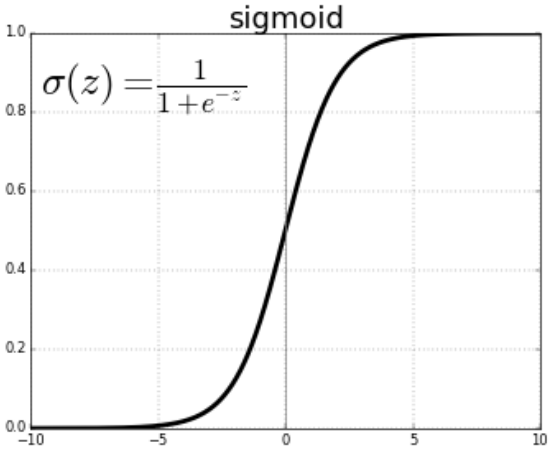

بسم الله الرحمن الرحيم

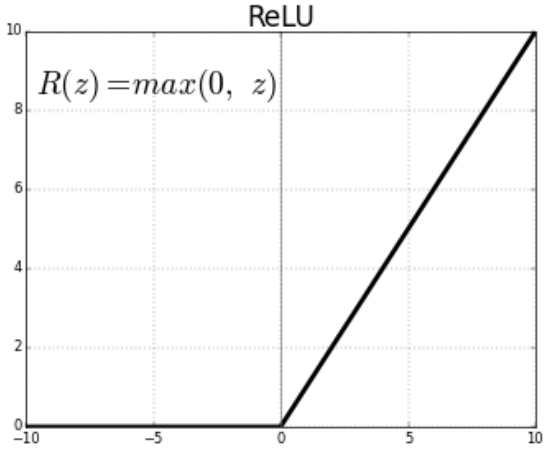
سوال اول:

۱.

دلیل اصلی استفاده از تابع سیگموید این است که بین (۰ تا ۱) وجود دارد. بنابراین، به ویژه برای مدل هایی استفاده می شود که باید احتمال را به عنوان خروجی پیش بینی کنیم. از آنجایی که احتمال هر چیزی فقط بین محدوده ۰ و ۱ وجود دارد، سیگموید انتخاب مناسبی است. تابع قابل تمایز است.

فعال‌سازی سیگموید، در حالی که برای طبقه‌بندی باینری مفید است، زمانی که ورودی‌های سیگموید بسیار بزرگ هستند (مثبت یا منفی) می‌توانند از فیدینگ گرادیان‌های رنج ببرند. این به این دلیل است که گرادیان تابع سیگموید در هر دو انتهای محدوده ورودی بسیار کوچک می شود.





ReLU برای شبکه‌های عصبی مدرن نسبت به سیگموید ترجیح داده می‌شود، زیرا بسیار سریع‌تر و بدون به خطر انداختن عملکرد اجرا می‌شود. با این حال، سیگموییدها به دلیل توانایی آنها در خروجی خروجی های گسسته بین ۰ و ۱ که به راحتی توسط انسان قابل تفسیر هستند، هنگام برخورد با وظایف طبقه بندی باینری مفید باقی می مانند. ReLU می تواند بر هر دو معایب موجود در توابع سیگموئید غلبه کند. از مشکل فیدینگ گرادیان جلوگیری می کند زیرا برای همه ورودی های مثبت گرادیان ثابت ۱ دارد. جریان گرادیان در فرایند برگشتی آسان‌تر می‌شود و آموزش موثرتر می‌شود.

ReLu بهترین و پیشرفته ترین تابع فعال سازی در حال حاضر در مقایسه با سیگموئید و TanH است زیرا تمام ایراداتی مانند مشکل فیدینگ گرادیان در این تابع فعال سازی کاملاً حذف شده است که این عملکرد فعال سازی را در مقایسه با سایر عملکردهای فعال سازی پیشرفته تر می کند.

در طول انتشار، فعال سازی ReLU اطمینان حاصل می کند که گرادیان ها برای مقادیر مثبت منتشر می شوند، که می تواند تا حدودی به کاهش مشکل فیدینگ گرادیان کمک کند. با این حال، اگر بسیاری از ورودی‌ها منفی باشند، همچنان می‌تواند به «نرون‌های مرده» منجر شود و باعث می‌شود که گرادیان‌ها برای آن نورون‌ها صفر شود.

در یک مشکل طبقه بندی دو کلاسه، اگر دو لایه پایین شبکه شما ReLU (واحد خطی اصلاح شده) و فعال کننده های Sigmoid باشند، رفتار شبکه عصبی شما به صورت زیر خواهد بود:

فعال سازی ReLU :

تابع فعال سازی ReLU به صورت $f(x) = max(0, x)$ تعریف می شود. این بدان معناست که برای هر ورودی تابع ReLU، خروجی در صورت مثبت بودن خود ورودی و در غیر این صورت صفر خواهد بود. هدف ReLU معرفی غیر خطی بودن مدل است که به آن امکان می دهد توابع پیچیده تری را بیاموزد. همچنین به کاهش مشکل گرادیان ناپدید شدن در حین انتشار پس زمینه کمک می کند.

فعال سازی سیگموید:

تابع فعال سازی سیگموید به صورت $f = \frac{1}{1+e^{-x}}$ تعریف می شود. ورودی را به مقداری بین ۰ و ۱ خرد می کند. در زمینه یک مسئله طبقه بندی دو کلاسه، خروجی تابع Sigmoid را می توان به عنوان احتمال کلاس مثبت (کلاس ۱) تفسیر کرد.

ورودی ها از لایه ReLU عبور کنند:

هر نورون در این لایه اگر مثبت باشد مستقیماً ورودی یا اگر منفی باشد صفر است. این باعث پراکندگی می شود و به شبکه کمک می کند تا غیرخطی ها را مدیریت کند.

خروجی های لایه ReLU از لایه سیگموید عبور کنند:

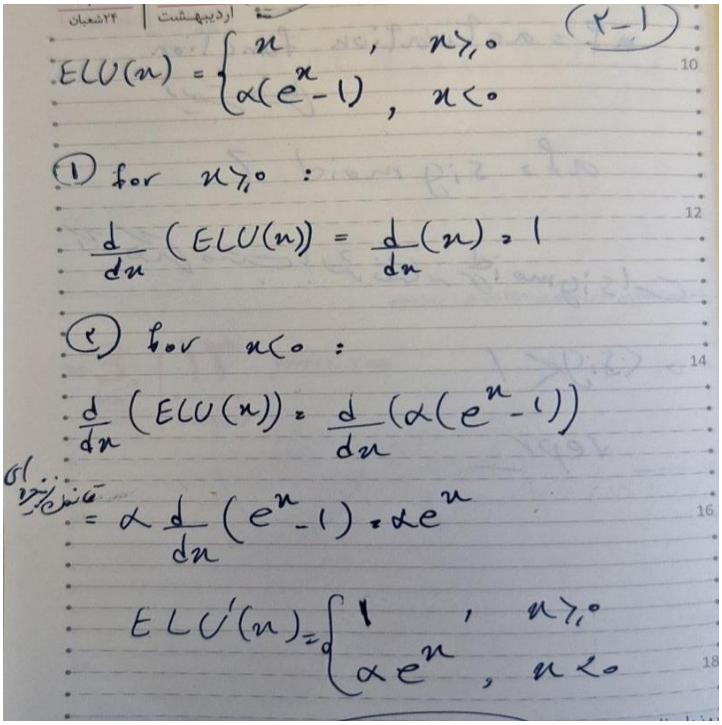
تابع سیگموید خروجی ها را از لایه ReLU (که اکنون به دلیل ماهیت ReLU غیر منفی هستند) می گیرد و آنها را در محدوده (۰، ۱) خرد می کند. سپس این مقادیر به عنوان احتمالات کلاس مثبت تفسیر می شوند.

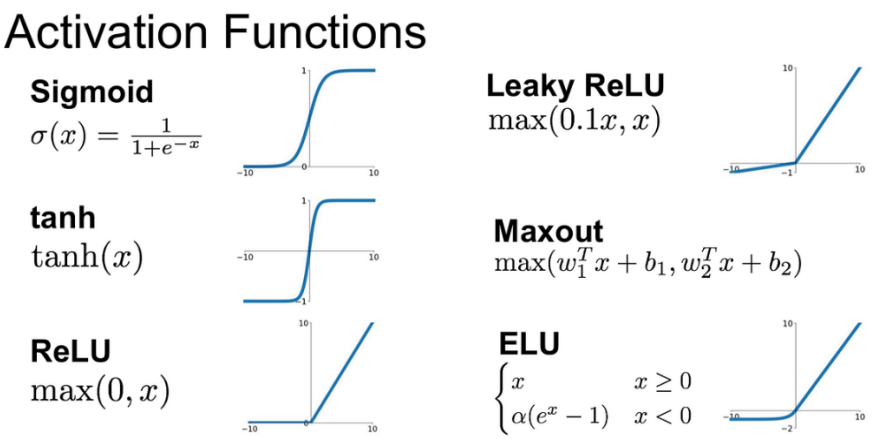
برای مثال:

اگر قبل از ReLU $[2.5, 0, -1, 3.5]$ باشد بعد از ReLU، $[2.5, 0, 0, 3.5]$ خواهد بود. زیرا ReLU مقادیر منفی را صفر می کند. سپس این مقادیر از تابع سیگوید عبور می کنند و در نتیجه احتمالات تقریباً $[0.924 , 0.5 , 0.5, 0.97]$حاصل می شود. اگر این لایه خروجی باشد و آستانه هدف ۰.۵ باشد، شبکه کلاس مثبت (کلاس ۱) را برای همه خروجی ها به جز خروجی دوم که آن را به عنوان کلاس ۰ پیش بینی می کند، درنظر میگیرد.

۲.

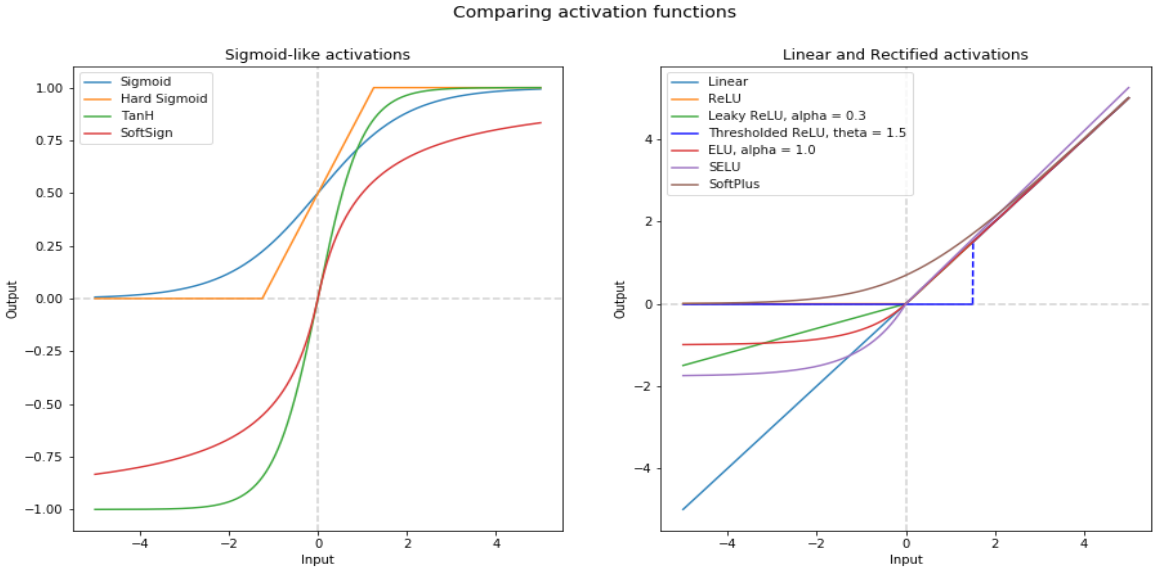
توابع فعال‌سازی توابعی هستند که در شبکه‌های عصبی برای محاسبه مجموع وزن‌های ورودی و بایاس استفاده می‌شوند، که برای تصمیم‌گیری درباره اینکه آیا یک نورون می‌تواند فعال شود یا خیر، استفاده می‌شود. این توابع اطلاعات ورودی را تحت یک پردازش گرادیان، معمولاً گرادیان نزولی، تغییر می‌دهند و سپس خروجی را برای شبکه عصبی تولید می‌کنند که شامل پارامترهای موجود در داده است. این توابع فعال‌سازی در برخی از منابع به عنوان تابع انتقال معروف هستند. توابع فعال‌سازی می‌توانند خطی یا غیرخطی باشند، به تبعیت از نوع تابعی که نماینده آن هستند، و برای کنترل خروجی شبکه‌های عصبی ما در دامنه‌های مختلف، از تشخیص و دسته‌بندی اشیاء تا تشخیص گفتار، تقسیم‌بندی، درک صحنه و توصیف، ترجمه ماشینی تست به سیستم‌های گفتاری، سیستم‌های تشخیص سرطان، تشخیص اثر انگشت، پیش‌بینی آب‌وهوا، خودروهای خودران و سایر حوزه‌ها، برای نام بردن تنها بخشی از آن‌ها، با نتایج پژوهش‌های اولیه که به تایید نتایج موثق‌تر در محاسبات شبکه‌های عصبی منجر شده‌اند.





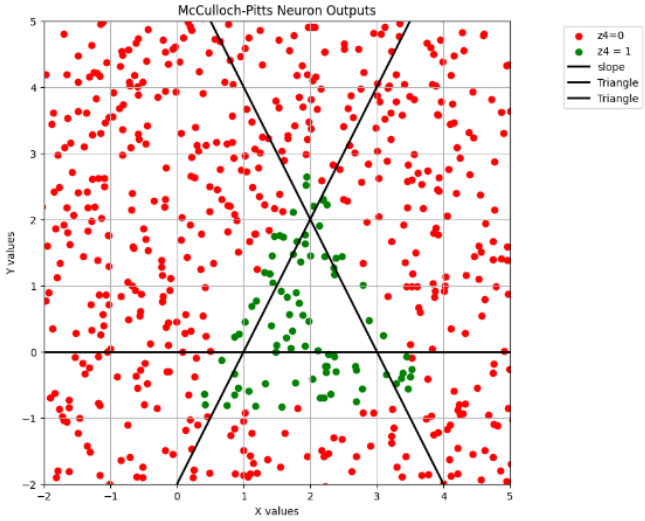
واحد خطی نمایی یا نام شناخته شده آن ELU تابعی است که تمایل دارد مقدار را سریعتر به صفر برساند و نتایج دقیق تری تولید کند. متفاوت از سایر توابع فعال‌سازی، ELU یک ثابت آلفای اضافی دارد که باید عدد مثبت باشد.

ELU به جز ورودی های منفی بسیار شبیه به RELU است. هر دو در فرم تابع شناسایی برای ورودی های غیر منفی هستند. از سوی دیگر، ELU به آرامی صاف می شود تا زمانی که خروجی آن برابر با $-\alpha$ باشد در حالی که RELU به سرعت صاف می شود. ELU به آرامی صاف می شود تا زمانی که خروجی آن برابر با $-\alpha$ باشد در حالی که RELU به شدت صاف می شود.

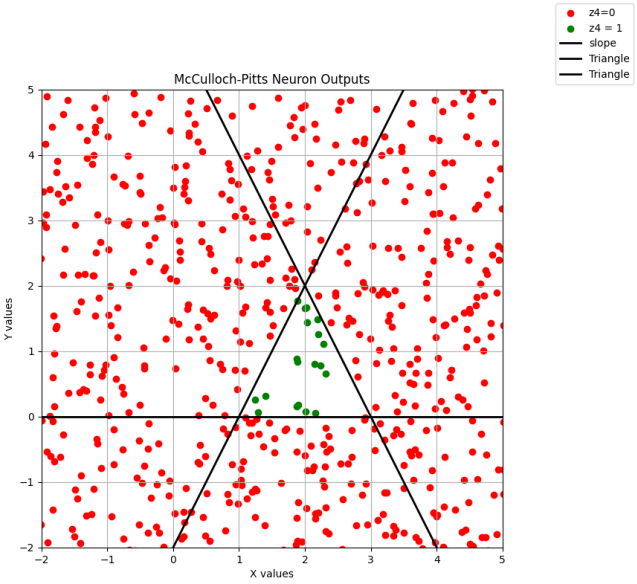


.۳

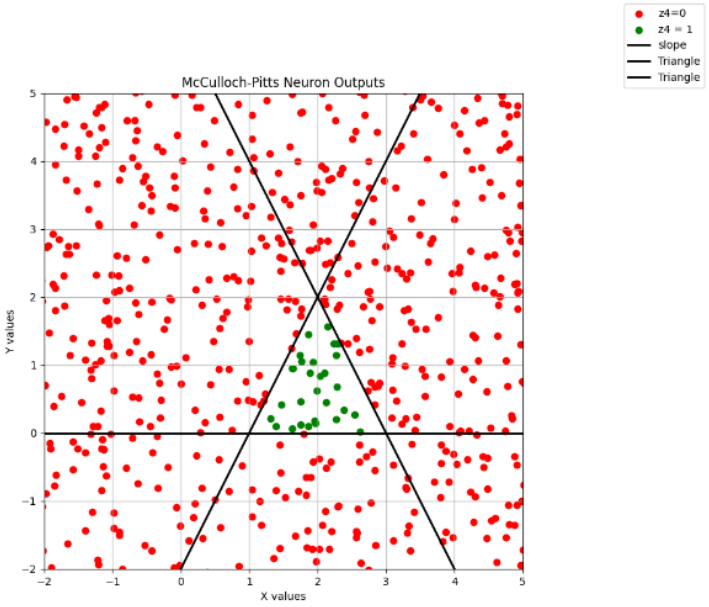
Sigmoid(threshold=0.3)



Sigmoid(threshold=0.5)



Leky_RELU(threshold=0)



توابع فعال سازی مختلف ویژگی های متفاوتی دارند که به مدل اجازه می دهد انواع مختلفی از روابط را در داده ها ثبت کند. به عنوان مثال، ReLU(واحد خطی اصلاح شده) پراکندگی را با صفر کردن مقادیر منفی معرفی می کند، در حالی که سیگموئید خروجی را بین ۰ و ۱ تقسیم می کند و آن را برای مسائل طبقه بندی باینری مناسب می کند. توابع فعال سازی مختلف می توانند به تفاسیر متفاوتی از مدل منجر شوند. برای مثال، فعال‌سازی سیگموئید را می‌توان به عنوان احتمال در مسائل طبقه‌بندی باینری تفسیر کرد، در حالی که فعال‌سازی softmax اغلب برای طبقه‌بندی چند کلاسه برای تفسیر خروجی‌ها به عنوان احتمالات کلاس استفاده می‌شود.

در توابع سیگموید ما با تغییر سطخ آستانه ناحیه کلاس بندی را کمتر یا بیش تر می کنیم با تعیین مقدار سطح آستانه به مقدار ۰.۵ مشاهده می شود که داده های سبز رنگ در ناحیه هاشور خورده تجمیع می شوند. با کاهش مقدار آستانه به ۰.۳ ناحیه کلاس بندی و نقاط سبز رنگ از ناحیه ی هاشور خورده فراتر می رود. دلیل آن این است که تابع سیگوید بین ۰ و۱ کلاس بندی شده اند و با تغییر مقدار آستانه داده های بیشتری را را نقاط سبز شامل می شود. نتیجه حاصل می شود که با استفاده از توابع فعال مختلف باید سطح آستانه را تغییر بدهیم تا در منطقه هاشور خورده قرار گیرد.

سوال دوم

۱.

داده‌های خطای بلبرینگ برای تشخیص و نظارت بر وضعیت یاتاقان‌ها در ماشین‌های دوار ضروری است. یاتاقان ها می توانند انواع مختلفی از عیوب را تجربه کنند که بر عملکرد و طول عمر ماشین آلات تأثیر می گذارد. داده‌های جمع‌آوری‌شده شامل سیگنال‌های ارتعاش، خوانش دما و سایر پارامترهای مرتبط از سنسورهای روی یا نزدیک یاتاقان‌ها است.

انواع رایج عیوب بلبرینگ:

: (B007_2) خطای ساچمه

این خطا شامل نقصی در توپ بلبرینگ است که باعث ایجاد ارتعاش و نویز قابل توجهی می‌شود به دلیل سطوح تماس نامنظم بین توپ و مسیرهای حرکت است و داده‌های ارتعاشی نشان‌دهنده پیک‌های دوره‌ای است که با فرکانس چرخش توپ مطابقت دارند. فرکانس و الگوی این پیک‌ها به شناسایی خطا کمک می‌کنند.

خرابی رینگ خارجی (OR007@6_2)

این خطا در حلقه بیرونی بلبرینگ رخ می‌دهد. حلقه بیرونی، حلقه‌ای است که در بلبرینگ ثابت است و نقص‌ها در اینجا باعث لرزش می‌شوند زمانی که عناصر غلتش بر روی نقص عبور می‌کنند.

سیگنال لرزش دارای ضربه‌های دوره‌ای است که با محل نقص در حلقه بیرونی مرتبط است. موقعیت منطقه بار (در این حالت در نقطه ۶:۰۰ قرار دارد) شدت و ویژگی‌های لرزش را تحت تأثیر قرار می‌دهد.

خطای رینگ داخلی

عبارت «تحلیل خرابی رینگ داخلی در داده های سر متحرک بلبرینگ با اطلاعات ۱۲ کیلوهرتز به حوزه خاصی در آنالیز خرابی بلبرینگ برای ماشین آلات دوار اشاره می کند .در اینجا اجزای کلیدی این عبارت شرح داده شده است. در یک بلبرینگ، رینگ داخلی درونی‌ترین جزء است که مستقیماً با محور چرخان در تماس است .این بخشی است که همراه با محور می چرخد.

در ماشین آلات دوار، سلامت بلبرینگ ها برای عملکرد قابل اعتماد ضروری است .بلبرینگ ها به طور معمول از یک رینگ داخلی، یک رینگ خارجی و ساچمه هایی تشکیل شده اند که بین این رینگ ها می چرخند .هنگام تجزیه و تحلیل خرابی های بلبرینگ، درک موقعیت نسبی این اجزا نسبت به منطقه بارگذاری ضروری است .این گزارش بر روی موقعیت های رینگ داخلی، ساچمه و رینگ خارجی نسبت به منطقه بارگذاری در مرکز ساعت ۶ تمرکز دارد، به ویژه در متن مجموعه داده خطای بلبرینگ سر متحرک ۱۲ کیلو دور در دقیقه . متمرکز می شود.

منطقه بارگذاری و اجزای بلبرینگ

منطقه بارگذاری ناحیه ای در بلبرینگ است که حداکثر بار در آن اعمال می شود .هنگامی که منطقه بارگذاری در مرکز ساعت ۶ قرار دارد، به این معنی است که حداکثر بار شعاعی به طور مستقیم به سمت پایین اعمال می شود .موقعیت های رینگ داخلی، ساچمه و رینگ خارجی نسبت به این منطقه بارگذاری می تواند ماهیت و شدت خرابی های بلبرینگ را نشان دهد.

موقعیت رینگ داخلی نسبت به منطقه بارگذاری

رینگ داخلی بخشی از بلبرینگ است که روی محور چرخان نصب می شود .هنگام تجزیه و تحلیل خرابی ها:

- **موقعیت مرکزی :**اگر رینگ داخلی در مرکز منطقه بارگذاری (6: 00 ساعت) قرار داشته باشد، این حالت نشان دهنده شرایط بارگذاری عادی است.
- **خارج از مرکز :**اگر رینگ داخلی نسبت به منطقه بارگذاری خارج از مرکز باشد، این امر می تواند ناشی از عدم تراز صحیح یا سایش نامناسب باشد و می تواند منجر به خرابی زودرس بلبرینگ شود.

در بخش های بعدی این گزارش، موقعیت های ساچمه و رینگ خارجی نسبت به منطقه بارگذاری در مجموعه داده خطای بلبرینگ سر متحرک ۱۲ کیلو دور در دقیقه مورد بحث قرار خواهد گرفت:

موقعیت نسبت به منطقه بارگذاری (مرکز در ساعت ۶:۰۰)

رینگ داخلی: رینگ داخلی همراه با محور می چرخد و در شرایط عادی، هرگونه ایرادی به صورت دوره ای از منطقه بارگذاری در ساعت ۶:۰۰ عبور می کند. در صورت وجود خرابی در رینگ داخلی، این خرابی زمانی که عیب درون منطقه بارگذاری قرار گیرد، به وضوح مشخص خواهد شد. بار باعث ایجاد تغییر شکل جزئی در عیب می شود که منجر به افزایش لرزش و صدا می شود.

شاخص های خرابی:

- سیگنال های لرزش با فرکانس بالا
- فرکانس های مشخصه مرتبط با سرعت چرخش رینگ داخلی (به عنوان مثال، BPFI - فرکانس عبور ساچمه از روی رینگ داخلی)

ساچمه: ساچمه ها عناصر غلتشی هستند که بار را بین رینگ داخلی و خارجی منتقل می کنند.

موقعیت نسبت به منطقه بارگذاری (مرکز در ساعت ۶:۰۰): همانطور که ساچمه ها از منطقه بارگذاری در ساعت ۶:۰۰ عبور می کنند، بیشترین بار را تحمل می کنند. هرگونه نقص یا خرابی در ساچمه ها منجر به افزایش لرزش و صدا به خصوص زمانی که این ساچمه ها در موقعیت ساعت ۶:۰۰ تحت بار قرار دارند، می شود.

شاخص های خرابی:

- فرکانس های مشخصه مربوط به فرکانس چرخش ساچمه، که معمولا توسط سرعت محور و فرکانس قفس بلبرینگ تعدیل می شود.

رینگ خارجی: رینگ خارجی ثابت بوده و در محفظه بلبرینگ نصب می شود.

موقعیت نسبت به منطقه بارگذاری (مرکز در ساعت ۶:۰۰): خرابی های رینگ خارجی در مقایسه با رینگ داخلی و ساچمه ها ایستا (استاتیک) تر هستند. هرگونه ایرادی روی رینگ خارجی، زمانی که منطقه بارگذاری در مرکز ساعت ۶:۰۰ قرار دارد، همیشه تحت بار خواهد بود.

شاخص های خرابی:

- فرکانس های مشخصه مرتبط با رینگ خارجی (به عنوان مثال، BPFO - فرکانس عبور ساچمه از روی رینگ خارجی)، که تمایل دارند ثابت باشند و به طور مستقیم تحت تاثیر سرعت محور مانند فرکانس های رینگ داخلی تعدیل نشوند.

تحلیل داده های خرابی بلبرینگ

تحلیل داده های خرابی بلبرینگ سر متحرک ۱۲ کیلو دور در دقیقه شامل استفاده از تکنیک های تحلیل ارتعاشات برای شناسایی فرکانس های مشخصه خرابی است .در اینجا خلاصه ای از فرکانس های مورد انتظار برای هر جزء در رابطه با منطقه بارگذاری در مرکز ساعت ۶ 00:ارائه شده است:

خرابی های رینگ داخلی:

- **BPFI فرکانس عبور ساچمه از روی رینگ داخل :**این فرکانس بر اساس تعداد ساچمه ها و سرعت چرخش رینگ داخلی محاسبه می شود .این نشان دهنده سرعت عبور یک عیب روی رینگ داخلی از زیر منطقه بارگذاری است.
- **علائم :**افزایش دامنه ارتعاش درBPFI ، که توسط چرخش محور تعدیل می شود.

خرابی های ساچمه:

- **فرکانس چرخش ساچمه :**فرکانسی که یک ساچمه به تنهایی حول محور خود می چرخد .این فرکانس پیچیده است زیرا می تواند تحت تاثیر سرعت هر دو رینگ داخلی و خارجی قرار گیرد.
- **علائم :**ارتعاش در فرکانس چرخش ساچمه، که اغلب توسط فرکانس قفس تعدیل می شود.

خرابی های رینگ خارجی:

- **BPFO فرکانس عبور ساچمه از روی رینگ خارجی :** این فرکانس بر اساس تعداد ساچمه ها و سرعت چرخش بلبرینگ محاسبه می شود .این نشان می دهد که یک عیب روی رینگ خارجی چند بار از منطقه بارگذاری عبور می کند.
- **علائم :**ارتعاش ثابت درBPFO ، با تعدیل کمتر نسبت به خرابی های رینگ داخلی.

ملاحظات عملی:

- **جمع آوری داده :**جمع آوری دقیق داده ها بسیار مهم است .سنسورهای ارتعاش باید به صورت استراتژیک برای دریافت سیگنال از رینگ داخلی، ساچمه و رینگ خارجی قرار گیرند.
- **پردازش سیگنال :**از تکنیک هایی مانند تبدیل فوریه سریع (FFT) برای تبدیل سیگنال های حوزه زمان به حوزه فرکانس جهت شناسایی فرکانس های مشخصه خرابی استفاده کنید.
- **پایش :**پایش مداوم می تواند به تشخیص زودهنگام خرابی ها کمک کند و امکان نگهداری به موقع و جلوگیری از خرابی های فاجعه آمیز را فراهم نماید.

درک موقعیت رینگ داخلی، ساچمه و رینگ خارجی نسبت به منطقه بارگذاری در مرکز ساعت ۶:۰۰ برای تشخیص خرابی بلبرینگ ضروری است. تیم های نگهداری با تمرکز بر فرکانس های مشخصه مرتبط با هر جزء، می توانند به طور مؤثر عیوب بلبرینگ را در ماشین آلات دوار شناسایی و برطرف کنند. تحلیل داده های خرابی بلبرینگ سر متحرک ۱۲ کیلو دور در دقیقه رویکردی جامع برای اطمینان از قابلیت اطمینان و طول عمر سیستم های مکانیکی ارائه می دهد.

اعتبارسنجی

در یادگیری ماشین، مجموعه داده به طور معمول به سه قسمت تقسیم می شود :مجموعه های آموزشی، اعتبارسنجی و تست .این تقسیم بندی برای ساخت یک مدل قوی و قابل اعتماد، به ویژه در وظایف طبقه بندی خرابی، بسیار مهم است . در اینجا توضیحاتی در مورد نقش هر مجموعه و اهمیت مجموعه اعتبارسنجی ارائه شده است.

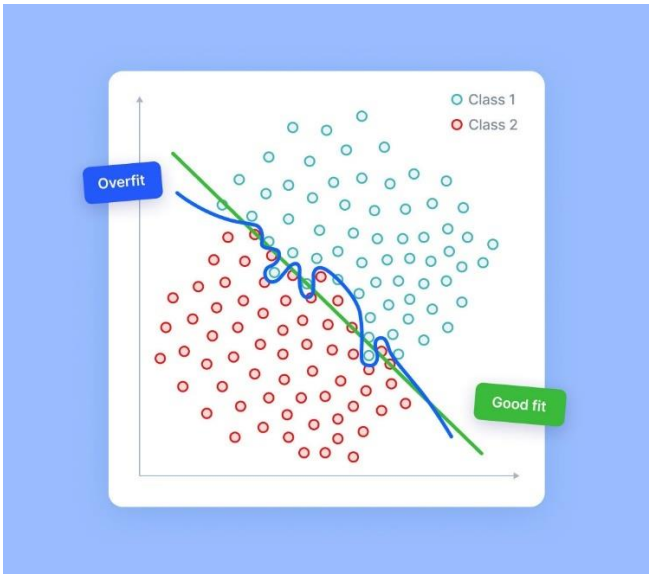
کدی که نوشتیم، بهم ریختگی و تقسیم بندی داده را انجام می دهد تا اطمینان حاصل کنیم که مدل بر روی زیرمجموعه های متمایز از داده ها آموزش داده شده، اعتبارسنجی شده و مورد آزمایش قرار بگیرد.

تنظیم مدل

مجموعه اعتبارسنجی برای تنظیم ابرپارامترها (تنظیمات مدل) و تصمیم گیری در مورد معماری مدل استفاده می شود. این فرآیند به عنوان بهینه سازی ابرپارامتر یا انتخاب مدل شناخته می شود. با ارزیابی مدل های مختلف یا ابرپارامترها روی مجموعه اعتبارسنجی، می توانیم بهترین مدل را قبل از آزمایش روی مجموعه تست (داده های جدید) انتخاب کنیم.

جلوگیری از overfitting برازش بیش از حد

استفاده از مجموعه اعتبارسنجی به نظارت بر عملکرد مدل روی داده های جدید در حین آموزش کمک می کند. اگر عملکرد مدل روی مجموعه آموزشی خوب باشد ولی روی مجموعه اعتبارسنجی ضعیف باشد، نشان دهنده overfitting است overfitting .زمانی اتفاق می افتد که مدل داده های آموزشی را بیش از حد خوب یاد می گیرد، که شامل نویز و داده های پرت (outlier) نیز می شود و باعث می شود مدل بر روی داده های جدید قابلیت تعمیم پذیری کمتری داشته باشد.



توقف زودهنگام

در طول آموزش، تکنیک های توقف زودهنگام قابل اجرا هستند، جایی که آموزش مدل زمانی متوقف می شود که عملکرد مجموعه اعتبارسنجی شروع به کاهش کند. این کار تضمین می کند که مدل بر روی داده های آموزشی overfitting نشود و قابلیت تعمیم پذیری بهتری را حفظ کند.

ارزیابی مدل

مجموعه اعتبارسنجی ارزیابی بی طرفانه ای از عملکرد مدل در طول آموزش ارائه می دهد. این مجموعه به عنوان جانشینی برای درک اینکه مدل چگونه ممکن است بر روی مجموعه تست (داده های جدید و ندیده) عمل کند، عمل می نماید.

تاثیر استفاده از مجموعه اعتبارسنجی در طبقه بندی خرابی

در طبقه بندی خرابی، به خصوص زمانی که با چندین کلاس خرابی سر و کار داریم (در کدی که نوشتیم :عادی، خرابی، خرابی۱، خرابی۲)این ۴ حالت را داریم، مجموعه اعتبارسنجی موارد زیر را تضمین می کند:

عملکرد متوازن در بین کلاس ها :مدل نه تنها بر اساس توانایی خود در تشخیص بین داده های عادی و داده های خرابی، بلکه بر اساس عملکرد آن در انواع مختلف خرابی ارزیابی می شود .این به ساخت مدلی کمک می کند که قوی باشد و بتواند انواع خرابی ها را به طور دقیق طبقه بندی کند.

شناسایی عدم تعادل کلاس ها :مجموعه اعتبارسنجی می تواند برجسته کند که آیا مدل نسبت به یک کلاس خاص سوگیری (bias) دارد یا خیر .این امر برای تشخیص خرابی بسیار مهم است زیرا ممکن است برخی از انواع خرابی در داده های آموزشی به اندازه کافی نشان داده نشده باشند.

۲.

پرسپترون‌های چند لایه (MLPs) با یک تابع فعال‌سازی مناسب در لایه خروجی (مانند سیگموئید یا سافت‌مکس) می‌توانند ابزارهای قدرتمندی برای طبقه‌بندی باشند.تعداد لایه‌های پنهان و نرون‌های موجود در آن لایه‌ها، پیچیدگی مدل و توانایی آن در یادگیری روابط پیچیده بین ویژگی‌ها و برچسب‌های کلاس را تعیین می‌کند. درسته که در scikit-learn و کلاس MLPClassifier امکان تغییر مستقیم تابع افت (loss function) در حین آموزش وجود نداره .تابع افت پیش‌فرض این کلاس، افت کراس‌انتروپی (یا همون افت لگاریتمی) هست که برای وظایف طبقه‌بندی باMLP ها کاملاً مناسبه و به طور کلی نیازی به تغییرش نیست.

تحلیل ماتریس اشفتگی

در ماتریس اشفتگی محور افقی داده های پیش بینی شده و محور عمودی داده های واقعی می باشند.

```
Accuracy: 0.975
Confusion Matrix:
 [[12  0]
 [ 1 27]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.92      1.00      0.96        12
         1.0       1.00      0.96      0.98        28

    accuracy                           0.97        40
   macro avg       0.96      0.98      0.97        40
weighted avg       0.98      0.97      0.98        40
```

ماتریس اشفتگی و ارزیابی عملکرد مدل طبقه بندی

ماتریس اشفتگی نمای کلی از پیش بینی های مدل و برچسب های واقعی برای هر کلاس ارائه می دهد .این ماتریس می تواند به ما در درک نقاط قوت و ضعف توانایی های طبقه بندی مدل کمک کند .

کلاس ۰:

عملکرد مدل برای کلاس ۰ بسیار خوب است، به طوری که همه ۱۲ نمونه به درستی به عنوان مثبت (مثبت واقعی) پیش بینی شده اند. این دقت بالا نشان می دهد که مدل ویژگی های متمایز کلاس ۰ را یاد گرفته است و می تواند نمونه های متعلق به این دسته را به طور قابل اعتمادی طبقه بندی کند. دقت ۰.۹۲ نشان می دهد که حدود ۹۲ درصد از نمونه های پیش بینی شده به عنوان کلاس ۰ صحیح هستند. Recall =۰.۱ نشان می دهد که مدل ۱۰۰ درصد از نمونه های واقعی کلاس ۱ را شناسایی کرده است. F1-score که برابر با ۹۶ است. دقت و recall را ترکیب می کند، نشان دهنده تعادل کلی بین توانایی مدل برای شناسایی موارد مثبت واقعی و تمایل آن به ایجاد موارد مثبت اشتباه است.

کلاس ۱:

برای کلاس۱، مدل به ۲۷ مورد مثبت واقعی دست می یابد، که نشان می دهد ۲۷ نمونه از این کلاس را به درستی پیش بینی کرده است. با این حال، ۱ مورد منفی اشتباه نیز وجود دارد که نشان می دهد مدل یک نمونه ای را که واقعاً به کلاس۱ تعلق دارند را شناسایی نکرده است. دقت ۰.۱ نشان می دهد که حدود ۱۰۰ درصد از نمونه های پیش بینی شده به عنوان کلاس ۱ صحیح هستند. Recall =۰.۹۶ نشان می دهد که مدل ۹۶ درصد از نمونه های واقعی کلاس ۱ را شناسایی کرده است. F1-score که برابر با ۹۸ است نشان دهنده تعادل کلی بین توانایی مدل برای شناسایی موارد مثبت واقعی و تمایل آن به ایجاد موارد مثبت اشتباه است.

در نتیجه، مدل عملکرد قوی را برای کلاس ۰و۱ نشان می‌دهد که نشان‌دهنده درک خوبی از ویژگی‌های متمایز آن است. برای بهبود بیشتر میتوانیم جمع‌آوری داده‌های آموزشی بیشتر برای این کلاس‌ها، تنظیم دقیق پارامترهای مدل، یا کاوش الگوریتم‌ها یا معماری‌های مختلف برای ثبت بهتر ویژگی‌های متمایز آنها باشد.

۳.

در این سوال ما از دو مدل متفاوت تصمیم به مقایسه گرفتیم همانطور که مشاهده میکنید برای MLPClassifier بهینه ساز sgd همچنین به صورت خودکار تابع اتلاف hinge تعریف شده است. اما برای MLPClassifier بهینه ساز خود را adam و تابع اتلاف به طور خودکار cross_entropy در نظر گرفته شده است.

```
mlp    =    MLPClassifier(hidden_layer_sizes=(64,    32),    alpha=0.0001,
                          activation='relu', solver='sgd', batch_size=32,
                       learning_rate='adaptive', learning_rate_init=0.001,
                            max_iter=2000, shuffle=True, random_state=94,
                              tol=1e-4,  verbose=False,  warm_start=False,
                           early_stopping=False, validation_fraction=0.1,
                                                    n_iter_no_change=10)
```

```
mlp    =    MLPClassifier((64,    32),    alpha=0.0001,    activation='relu',
                                           solver='adam', batch_size=32,
                       learning_rate='adaptive', learning_rate_init=0.001,
                            max_iter=2000, shuffle=True, random_state=94,
                              tol=1e-4,  verbose=False,  warm_start=False,
                           early_stopping=False, validation_fraction=0.1,
                                                    n_iter_no_change=10)
```

```
Accuracy: 0.975
Confusion Matrix:
 [[12  0]
 [ 1 27]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.92      1.00      0.96        12
         1.0       1.00      0.96      0.98        28

    accuracy                           0.97        40
   macro avg       0.96      0.98      0.97        40
weighted avg       0.98      0.97      0.98        40
```

```
Accuracy: 0.975
Confusion Matrix:
 [[23  2]
 [ 0 55]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      0.92      0.96        25
         1.0       0.96      1.00      0.98        55

    accuracy                           0.97        80
   macro avg       0.98      0.96      0.97        80
weighted avg       0.98      0.97      0.97        80
```

تجزیه و تحلیل منحنی‌های آموزش و اعتبارسنجی:

۱- تابع خطا bce + Adam:

- همگرایی سریع با منحنی یادگیری تند.
- بیش‌برازش کم، زیرا نمودارهای تلفات و دقت آموزش و اعتبارسنجی به هم نزدیک هستند و در نزدیکی ۱۰۰درصد استقرار می‌یابند.

۲- تابع خطا Hinge + SGD:

- افزایش پایدار دقت آموزش، نزدیک به ۱۰۰ درصد.
- دقت اعتبارسنجی مطابق با دقت آموزش و در نزدیکی ۱۰۰درصد استقرار می‌یابد.
- کاهش تند تلفات آموزش و استقرار در مقدار کمی.
- تلفات اعتبارسنجی همراه با تلفات آموزش، استقراری کمی بالاتر از تلفات آموزش.

تجزیه و تحلیل دقیق و ارزیابی:

۱- دقت و ماتریس سردرگمی:

هر دو پیکربندی به دقت ۰،۹۷۵ دست یافتند. با این حال، ماتریس‌های سردرگمی نشان می‌دهند که توزیع خطاها متفاوت است:

۲- تابع خطا bce + Adam:

اندازه مجموعه داده: ۴۰ نمونه.

دقت و بازخوانی بالا برای هر دو کلاس.

یک خطا در طبقه‌بندی ۱

۳- تابع خطا Hinge + SGD:

اندازه بزرگتر مجموعه داده: ۸۰ نمونه.

بازخوانی کامل در کلاس ۱ و دقت کمی پایین‌تر در کلاس ۰

دو خطا در طبقه‌بندی ۰

اندازه بزرگتر مجموعه داده در پیکربندی تابع خطا Hinge، ارزیابی قوی‌تری را فراهم کرده است، اما همچنین افزایش احتمال خطاها در کلاس کمیتها به دلیل تعادل نکردن مجموعه داده را نیز افزایش می‌دهد.

تجزیه و تحلیل منحنی‌های آموزش و اعتبارسنجی:

۱- تابع خطا bce + Adam:

- همگرایی سریع با منحنی یادگیری تند.
- بیش‌برازش کم، زیرا نمودارهای تلفات و دقت آموزش و اعتبارسنجی به هم نزدیک هستند.

۲- تابع خطا Hinge + SGD:

- بهبود پایدار با منحنی‌های یادگیری پایدار.
- بیش‌برازش کمی، زیرا تلفات اعتبارسنجی کمی بالاتر از تلفات آموزش است.

تأثیر تابع خطا:

۱- تابع خطا bce:

- برای مدل‌های احتمالاتی مناسب است.
- بهبوددهنده Adam نرخ یادگیری را تطبیق می‌دهد که منجر به همگرایی سریع‌تر و پایداری می‌شود.
- برای وظایف طبقه‌بندی دودویی مناسب است و کاهش تلفات را به صورت صاف و پیوسته فراهم می‌کند.

۲- تابع خطا Hinge:

- برای طبقه‌بندی‌کننده‌های مبتنی بر حاشیه مانند SVM مناسب است.
- برجسته‌سازی بیشینه کردن حاشیه بین کلاس‌ها که می‌تواند منجر به تعمیم بهتر شود.
- بهینه‌ساز SGD مسیر بهینه‌سازی ساده‌ای را فراهم می‌کند، اگرچه ممکن است همگرایی کندتری داشته باشد.

هر دو پیکربندی عملکرد بسیار خوبی داشته‌اند و دقت بالایی را به دست آورده‌اند. انتخاب بین تابع خطا **تابع خطا bce + Adam**

و تابع خطا Hinge + SGD بستگی به نیازهای وظیفه خاص و ویژگی‌های مجموعه داده دارد.

تحلیل دقیق و بررسی عملکرد ضعیف:

اگرچه مدل‌ها در این حالت عملکرد خوبی داشتند، اما در برخی حالات عملکرد ممکن است ضعیف شود. در زیر توضیحاتی در مورد برخی از دلایل ممکن برای عملکرد ضعیف و تحلیل مرتبط آنها آورده شده است:

بیش‌برازش:

- نشانه‌ها: دقت آموزش بالا و دقت اعتبارسنجی پایین.
- علت: مدل برای اندازه مجموعه داده یا تعداد دوره‌های آموزشی زیاد، پیچیده است.
- راهکار: استفاده از تکنیک‌های مقرراتی، کاهش پیچیدگی مدل یا استفاده از موقف زودهنگام.

تعادل کلاس‌ها:

- نشانه‌ها: دقت/بازخوانی بالا برای کلاس اکثریت و پایین برای کلاس اقلیت.
- علت: تراز نبودن نمونه‌های کلاس‌ها در مجموعه آموزشی.
- راهکار: استفاده از تکنیک‌هایی مانند بیش‌نمونه‌گیری، کم‌نمونه‌گیری یا وزن‌دهی به کلاس‌ها.

کیفیت ناکافی داده:

- نشانه‌ها: پیش‌بینی‌های نامنظم یا نویزی.
- علت: ویژگی‌های نویزی، ناقص یا غیرمرتبط.
- راهکار: پاکسازی داده، انتخاب ویژگی و مهندسی ویژگی.

پارامترهای نامناسب:

- نشانه‌ها: عملکرد ضعیف در مقابل کیفیت و پیش‌پردازش خوب داده.
- علت: انتخاب نامناسب پارامترها.
- راهکار: تنظیم پارامترها با استفاده از جستجوی شبکه یا جستجوی تصادفی.

پیش‌پردازش نامناسب:

- نشانه‌ها: عملکرد غیرقابل پیش‌بینی در اجراهای مختلف.
- علت: مقیاس‌بندی یا تغییر نامناسب ویژگی‌ها.
- راهکار: استانداردسازی یا نرمال‌سازی داده، اموازنه‌بندی پیش‌پردازش مطمئن، تأمین کنید.

.۴

اعتبارسنجی متقاطع (Cross-Validation) یکی از روش‌های اساسی در ارزیابی مدل‌های یادگیری ماشین است .این روش به ما کمک می‌کند تا عملکرد مدل را در داده‌های جدید و ناشناخته تخمین بزنیم و از برازش بیش از حد (Overfitting)مدل جلوگیری کنیم .دو نوع رایج از اعتبارسنجی متقاطع عبارتند از:

- **اعتبارسنجی متقاطعK-Fold** : در این روش، مجموعه داده به K زیرمجموعه یا فولد تقسیم می‌شود .مدل K بار آموزش و ارزیابی می‌شود و هر بار از یک فولد متفاوت به عنوان مجموعه اعتبارسنجی استفاده می‌شود . معیارهای عملکرد از هر طبقه برای تخمین عملکرد تعمیم مدل به طور میانگین محاسبه می‌شوند.
- **اعتبارسنجی متقاطع K-Fold طبقه‌بندی شده :**این روش مشابه K-Fold است، با این تفاوت که هنگام تقسیم داده‌ها به فولدها برای ارزیابی مدل، عدم تعادل کلاس‌ها را در نظر می‌گیرد .به این معنی که هر فولد باید همان نسبت نمونه‌ها را برای هر کلاس هدف به عنوان کل مجموعه داده داشته باشد.

مزایای اعتبارسنجی متقاطعK-Fold

- **دقت پایدار :**این روش دقت تصادفی را حل می‌کند و تخمین دقیق‌تری از عملکرد مدل در داده‌های جدید ارائه می‌دهد.
- **جلوگیری از برازش بیش از حد :**با قرار دادن مدل در معرض زیرمجموعه‌های مختلف داده، از برازش بیش از حد مجموعه داده‌های آموزشی جلوگیری می‌کند.
- **اعتبارسنجی تعمیم مدل :**اعتبارسنجی متقابل بینشی را در مورد نحوه تعمیم مدل به مجموعه داده‌های ناشناخته ارائه می‌دهد.

مزایای اعتبارسنجی متقاطع K-Fold طبقه‌بندی شده

علاوه بر مزایایK-Fold ، اعتبارسنجی متقاطع K-Fold طبقه‌بندی شده مزایای دیگری نیز دارد:

- **غلبه بر عدم تعادل کلاس :**این روش برای مجموعه داده‌هایی که کلاس‌های نامتعادل دارند، مانند مجموعه داده‌هایی که در آنها یک کلاس به طور قابل توجهی کمتر از کلاس‌های دیگر وجود دارد، مناسب است.

- **ارزیابی عادلانه مدل :**با اطمینان از اینکه هر فولد دارای همان نسبت نمونه‌ها برای هر کلاس است، ارزیابی عادلانه‌تری از عملکرد مدل را ارائه می‌دهد.

انتخاب بین K-Fold و K-Fold طبقه‌بندی شده

انتخاب بین K-Fold و K-Fold طبقه‌بندی شده به نوع مجموعه داده و اهداف مدل‌سازی ما بستگی دارد. اگر مجموعه داده متعادل است، K-Fold به طور کلی کافی است .با این حال، اگر مجموعه داده نامتعادل است، K-Foldطبقه‌بندی شده انتخاب بهتری است.

مقایسه عملکرد مدل با و بدون اعتبارسنجی K-Fold

عملکرد یک مدل یادگیری ماشین با استفاده از اعتبارسنجی K-Fold و بدون آن را مقایسه می‌کنیم.

تحلیل نمودارهای دقت

با اعتبارسنجی K-Fold

نمودار اول دقت آموزش و اعتبارسنجی را در طول اپوک‌ها نشان می‌دهد:

- دقت آموزش (خط قرمز) و دقت اعتبارسنجی (خط سبز) بسیار نزدیک و صاف هستند.
- هر دو دقت به سرعت به دقت ۱ می‌رسند و ثابت می‌مانند که نشان دهنده تعمیم خوب مدل به داده‌های ناشناخته است.

بدون اعتبارسنجی K-Fold

نمودار دوم دقت آموزش و اعتبارسنجی برای مدل آموزش‌دیده بدون اعتبارسنجی K-Foldرا نشان می‌دهد:

- دقت آموزش (خط آبی) و دقت اعتبارسنجی (خط نارنجی) نیز به سرعت به مقادیر بالایی می‌رسند.
- در ابتدای کار نوسانات قابل توجهی در دقت اعتبارسنجی وجود دارد و بعداً ثابت می‌شود، اما روند کلی حاکی از بیش‌برازش است، زیرا دقت اعتبارسنجی همیشه با دقت آموزش مطابقت ندارد.

نتایج و گزارش‌های طبقه‌بندی

با اعتبارسنجی K-Fold

- دقت تست: ۱.۰
- گزارش طبقه‌بندی:

- دقت: ۱.۰۰
- بازیابی: ۱.۰۰
- امتیاز :F1 ۱.۰۰

بدون اعتبارسنجی K-Fold

- دقت: ۰.۹۷۵
- گزارش طبقه‌بندی:
  - دقت: ۰.۹۸
  - بازیابی: ۰.۹۷
  - امتیاز F1 :۰.۹۸

تحلیل و مقایسه

دقت

- مدل با اعتبارسنجی K-Fold به دقت کامل ۱ در مجموعه تست دست یافته است، در حالی که مدل بدون اعتبارسنجی K-Foldبه دقت ۰.۹۷۵ دست یافته است.
- دقت بالاتر با اعتبارسنجی متقاطع K-Fold نشان‌دهنده تعمیم بهتر است.

گزارش طبقه‌بندی

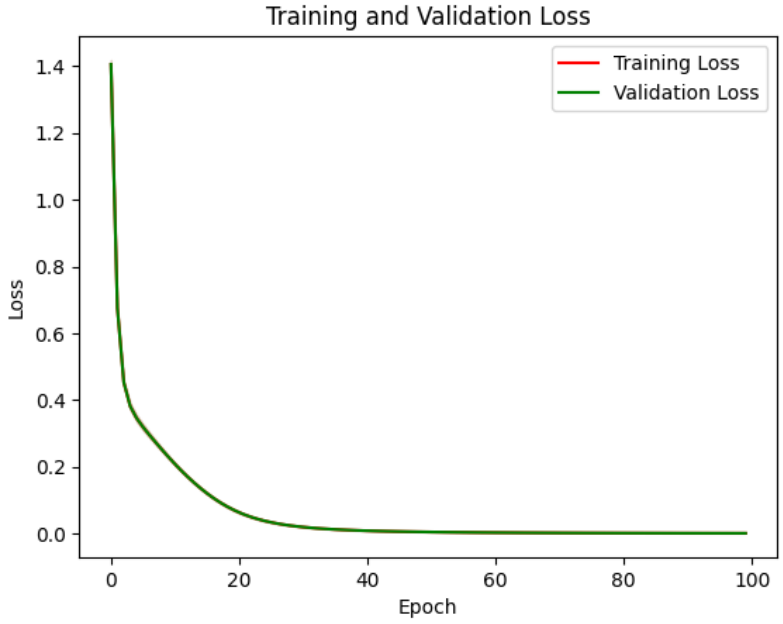
- مدل با اعتبارسنجی K-Foldدارای دقت، بازیابی و امتیاز F1 کامل برای هر دو کلاس است.
- مدل بدون اعتبارسنجی K-Foldنیز عملکرد خوبی دارد اما دقت کمی پایین‌تر ۰.۹۲ برای کلاس ۰ و بازیابی کمی پایین‌تر ۰.۹۶برای کلاس ۱ دارد.

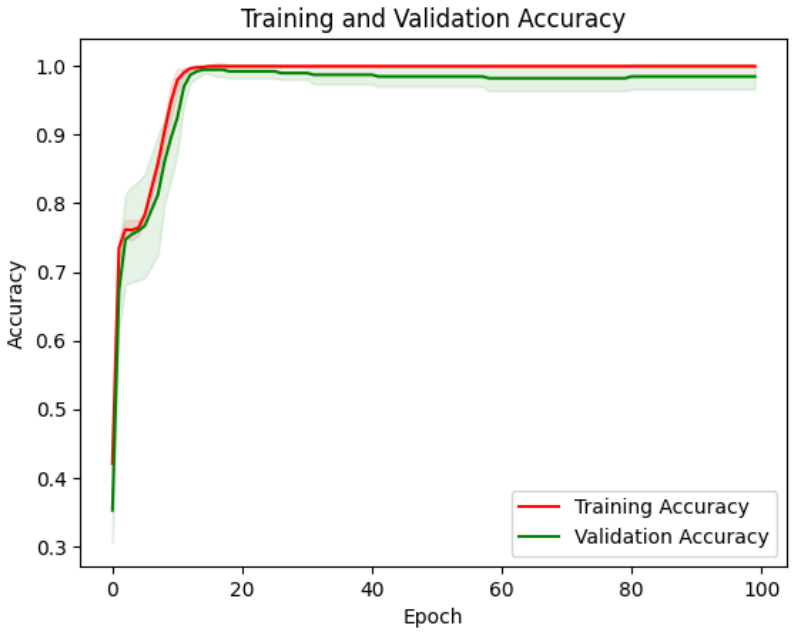
بیش‌برازش

- نمودارهای دقت نشان می‌دهند که مدل بدون اعتبارسنجی K-Foldتا حدی دچار بیش‌برازش است، زیرا نوساناتی در دقت اعتبارسنجی مشاهده می‌شود.
- مدل با اعتبارسنجی K-Fold نمودارهای دقت صاف و ثابتی دارد که نشان‌دهنده عملکرد قوی‌تر و تعمیم‌یافته‌تر است.

استفاده از اعتبارسنجیK-Fold به طور قابل توجهی عملکرد مدل را با ارائه تعمیم بهتر و کاهش بیش‌برازش بهبود می‌بخشد. نمودارهای دقت صاف‌تر و گزارش طبقه‌بندی ایده‌آل نشان می‌دهند که اعتبارسنجی K-Foldرویکرد برتری برای آموزش مدل‌های یادگیری ماشین نسبت به روش تقسیم تک سنتی است.

با استفاده از اعتبارسنجی K-Fold، اطمینان حاصل می‌شود که مدل بر روی زیرمجموعه‌های مختلف داده‌ها ارزیابی می‌شود که منجر به ارزیابی قابل اعتمادتر و قوی‌تر از عملکرد آن می‌شود. این روش نه تنها دقت بیشتری ارائه می‌دهد، بلکه به کاهش ریسک بیش‌برازش نیز کمک می‌کند و در نتیجه مدل‌هایی با قابلیت تعمیم بهتر تولید می‌کند.





سوال ۳

۱.

در این کد ما از Train-Test Split استفاده نمودیم

تقسیم بندی مجموعه داده برای آموزش و تست (تقسیم بندی تصادفی با حفظ توزیع کلاس)

کد از روش رایج تقسیم بندی مجموعه داده برای آموزش و تست استفاده می‌کند. در اینجا توضیحی در مورد این روش آمده است:

روش استفاده شده در کد (تقسیم‌بندی تصادفی با حفظ توزیع کلاس):

- از کتابخانه‌ی scikit-learn و تابع train_test_split برای تقسیم‌بندی استفاده می‌شود.
- آرگومان test_size=0.15 مشخص می‌کند که ۱۵ درصد از داده‌ها برای تست اختصاص یابد و ۸۵ درصد باقی‌مانده برای آموزش استفاده شود.

- آرگومان random_state=94 برای تولید اعداد تصادفی قابل تکرار استفاده می‌شود (برای اطمینان از اینکه هر بار اجرای کد نتایج یکسانی به دست آورید).
- آرگومان مهم دیگر $stratify = y$ است. این آرگومان اطمینان می‌دهد که توزیع کلاس‌ها در مجموعه داده‌ی تست، مشابه توزیع کلاس‌ها در کل مجموعه داده باشد. این کار برای طبقه‌بندی چند کلاسه‌ای که کلاس‌ها ممکن است تعداد متفاوتی داشته باشند، اهمیت زیادی دارد.

روش بهترk-fold cross-validation :

اگرچه روش تقسیم‌بندی تصادفی با حفظ توزیع کلاس، یک روش رایج است، اما روشی به نام k-fold cross-validationمی‌تواند عملکرد مدل را دقیق‌تر ارزیابی کند. در این روش:

- مجموعه داده‌ی اصلی به k قسمت مساوی تقسیم می‌شود مثلا k=10
- در هر تکرار (iteration)
    - یک قسمت به عنوان مجموعه داده‌ی تست و k-1 قسمت باقی‌مانده به عنوان مجموعه داده‌ی آموزش در نظر گرفته می‌شود.
    - مدل روی مجموعه داده‌ی آموزش ساخته می‌شود.
    - عملکرد مدل روی مجموعه داده‌ی تست ارزیابی می‌گردد.
- این فرایند برای هر k تکرار انجام می‌شود و در نهایت میانگین عملکرد مدل در تمام تکرارها به عنوان ارزیابی نهایی در نظر گرفته می‌شود.

مزایای:k-fold cross-validation

- از کل مجموعه داده برای ارزیابی مدل استفاده می‌شود (در حالی که در تقسیم‌بندی تصادفی، بخشی از داده برای تست کنار گذاشته می‌شود).
- ارزیابی مدل با ثبات‌تر است، زیرا عملکرد مدل روی زیرمجموعه‌های مختلفی از داده‌ها سنجیده می‌شود.

۲.

درخت تصمیم (Decision Tree) یک الگوریتم یادگیری با نظارت است که برای هر دو کار طبقه‌بندی (classification) و رگرسیون (regression) به کار می‌رود .این الگوریتم داده‌ها را بر اساس مقادیر ویژگی‌های ورودی (input features) به زیرمجموعه‌هایی تقسیم می‌کند و بدین ترتیب یک مدل درختی از تصمیمات و نتایج احتمالی آن‌ها ایجاد می‌کند.

۱- انتخاب بهترین ویژگی

در گره‌ی ریشه(root node) ، الگوریتم تمام ویژگی‌ها را ارزیابی می‌کند و ویژگی‌ای را انتخاب می‌کند که داده‌ها را به بهترین شکل به طبقات مجزا تقسیم کند. این کار اغلب با استفاده از معیارهایی مانند ناخالصی (Gini Impurity)یا افزایش اطلاعات (Entropy) انجام می‌شود.

بهترین ویژگی، ویژگی‌ای است که در هنگام استفاده برای تقسیم داده‌ها، منجر به بالاترین افزایش اطلاعات (یا پایین‌ترین ناخالصی) شود.

۲- تقسیم مجموعه داده

پس از انتخاب بهترین ویژگی برای تقسیم، مجموعه داده بر اساس مقادیر آن ویژگی به زیرمجموعه‌هایی تقسیم می‌شود:

- برای ویژگی‌های پیوسته :(Continuous Features) در این حالت، الگوریتم یک مقدار آستانه (threshold) را انتخاب می‌کند و داده‌ها را بر اساس اینکه مقدار ویژگی برای هر نمونه، بزرگتر یا کوچکتر از آستانه باشد، به دو زیرمجموعه تقسیم می‌کند.
- برای ویژگی‌های اسمی :(Categorical Features) این ویژگی‌ها، مقادیر مجزا و از پیش تعریف شده‌ای دارند برای این نوع ویژگی‌ها، الگوریتم داده‌ها را بر اساس دسته‌بندی‌های مختلف آن ویژگی تقسیم می‌کند .

این فرایند تقسیم، در هر گره‌ی داخلی درخت (به جز گره‌ی نهایی) بر اساس بهترین ویژگی انتخاب‌شده در آن گره، تکرار می‌شود .در نهایت، مجموعه داده‌ی اولیه به زیرمجموعه‌های کوچکتر و همگن‌تری از نظر خروجی (کلاس) تقسیم می‌شود.

۳- ایجاد گره‌های تصمیم و برگ:

- گره تصمیم (Decision Node)

زمانی ایجاد می‌شود که داده‌ها قابلیت تقسیم بیشتر بر اساس ویژگی‌ها را داشته باشند.

- برگ درخت(Leaf Node)

زمانی ایجاد می‌شود که دیگر امکان تقسیم داده‌ها وجود نداشته باشد (داده‌های زیرمجموعه همگی متعلق به یک کلاس هستند یا دیگر ویژگی‌ای برای تقسیم باقی نمانده است).

۴- تکرار فرایند(Repeat the Process)

الگوریتم به صورت بازگشتی (recursively) مراحل ۱ تا ۳ را برای زیرمجموعه‌های ایجاد شده در تقسیم قبلی تکرار می‌کند. این فرایند تا زمانی که یکی از شرایط توقف زیر رخ دهد، ادامه می‌یابد:

داده‌های یک گره همگی به یک کلاس تعلق داشته باشند: دیگر نیازی به تقسیم بیشتر نیست.

ویژگی باقیمانده‌ای برای تقسیم وجود نداشته باشد: دیگر نمی‌توان داده‌ها را بر اساس ویژگی‌ها تقسیم کرد.

به عمق از پیش تعیین شده‌ی درخت برسد: برای جلوگیری ازoverfitting ، ممکن است حداکثر عمق مجاز برای درخت در نظر گرفته شود.

تعداد نمونه‌های کافی برای تقسیم وجود نداشته باشد: تعداد نمونه‌های موجود برای تقسیم یک گره به زیرمجموعه‌های فرزند، به کمتر از حداقل مقدار مورد نیاز برسد.

۵- پیش‌بینی:(Prediction)

برای طبقه‌بندی یک نمونه‌ی جدید (داده‌ی تست):

- نمونه از ریشه‌ی درخت به سمت پایین هدایت می‌شود.
- در هر گره‌ی تصمیم، مقدار ویژگی مربوط به آن گره برای نمونه بررسی می‌شود.
- بر اساس این مقدار، نمونه مسیر (شاخه) خروجیِ مرتبط با آن مقدار را دنبال می‌کند (مثلا برای یک ویژگی پیوسته، مسیر مربوط به بزرگتر یا کوچکتر بودن از آستانه‌ی مشخص).
- این فرایند تا رسیدن به یک گره‌ی نهایی (برگ) ادامه می‌یابد.
- کلاس مرتبط با گره‌ی نهایی به عنوان کلاس پیش‌بینی‌شده برای نمونه‌ی جدید در نظر گرفته می‌شود.

ساخت گام به گام درخت تصمیم:

۱. گره ریشه (Root Node):

- الگوریتم هر دو ویژگی Feature1 و Feature2 را ارزیابی می‌کند تا مشخص کند کدام ویژگی بهترین تقسیم را بر اساس داده‌ها ایجاد می‌کند (مثلا با استفاده از معیار ناخالصی gini).
- فرض کنید Feature1 با آستانه‌ی ۵ انتخاب شود.

۲. اولین تقسیم (First Split):

بر اساس آستانه‌ی انتخاب‌شده برای Feature1، داده‌ها به دو زیرمجموعه تقسیم می‌شوند:

- زیرمجموعه‌ی اول: شامل نمونه‌هایی که مقدار Feature1 آن‌ها کوچک‌تر یا مساوی ۵ است (Feature1 <= ۵).

- زیرمجموعه‌ی دوم: شامل نمونه‌هایی که مقدار Feature1 آن‌ها بزرگتر از ۵ است (Feature1 > ۵).

برای نشان دادن این تقسیم، یک گره‌ی تصمیم برای Feature1 با آستانه‌ی ۵ در درخت ایجاد می‌شود.

۳. گره‌های بعدی (Subsequent Nodes):

برای هر یک از زیرمجموعه‌های ایجاد شده در مرحله‌ی قبل، فرایند به صورت زیر تکرار می‌شود:

- الگوریتم ویژگی‌های باقیمانده را ارزیابی می‌کند.
- بهترین ویژگی برای تقسیم بیشتر آن زیرمجموعه‌ی خاص انتخاب می‌شود.
- زیرمجموعه بر اساس ویژگی انتخاب‌شده، تقسیم می‌شود.
- بر اساس اینکه آیا امکان تقسیم بیشتر وجود دارد یا خیر، یک گره‌ی تصمیم یا یک برگ (leaf node) در درخت ایجاد می‌شود.

۴. برگ‌های درخت (Leaf Nodes):

- زمانی که دیگر امکان تقسیم یک زیرمجموعه وجود نداشته باشد (مثلا همه‌ی نمونه‌های موجود در آن زیرمجموعه به Class0 تعلق داشته باشند)، یک برگ در انتهای مسیر آن زیرمجموعه در درخت ایجاد می‌شود. برچسب این برگ، کلاس مربوط به نمونه‌های آن زیرمجموعه است (مثلا Class0).

به این ترتیب، با تکرار این فرایند برای تمام زیرمجموعه‌های ایجاد شده، یک درخت تصمیم با گره‌های تصمیم و برگ‌ها شکل می‌گیرد که برای طبقه‌بندی نمونه‌های جدید قابل استفاده است.

تعیین گره‌ها و افزایش اطلاعات در درخت‌های تصمیم (تعیین گره تقسیم و محاسبه‌ی افزایش اطلاعات)

برای انتخاب بهترین ویژگی جهت تقسیم داده‌ها در یک درخت تصمیم، از معیاری به نام افزایش اطلاعات (Information Gain) استفاده می‌کنیم. افزایش اطلاعات به ما کمک می‌کند تا میزان کاهش در ناخالصی (Entropy) یا عدم اطمینان را که با تقسیم داده‌ها بر اساس یک ویژگی خاص به دست می‌آید، کمیت‌سازی کنیم (مقدار آن را به صورت عددی نشان دهیم).

۱ .محاسبه‌ی ناخالصی (Entropy) گره‌ی والد:

ناخالصی، میزان ناهمگنی یا درهم‌ریختگی (عدم اطمینان) یک مجموعه داده را اندازه‌گیری می‌کند .برای محاسبه‌ی ناخالصی از فرمول آنتروپی (Entropy) استفاده می‌شود.

فرمول آنتروپی H(D) برای یک مجموعه داده‌ی D با n کلاس به صورت زیر است:

$$H(D) = -\Sigma(pi * log2(pi))$$

که در آن:

pi نسبت نمونه‌ها (احتمال) متعلق به کلاس i ام است.

۲. انتخاب ویژگی برای تقسیم:

برای هر ویژگی در مجموعه داده:

- داده‌ها را بر اساس مقادیر آن ویژگی به زیرمجموعه‌هایی تقسیم کنید.
- برای ویژگی‌های پیوسته: یک آستانه انتخاب کنید و داده‌ها را به دو زیرمجموعه تقسیم کنید، یک زیرمجموعه برای مقادیر کمتر از آستانه و دیگری برای مقادیر بزرگتر یا مساوی آستانه.
- برای ویژگی‌های اسمی: داده‌ها را بر اساس هر دسته‌ی مجزا از آن ویژگی به زیرمجموعه‌های مجزا تقسیم کنید.

۳. محاسبه‌ی آنتروپی وزن‌دار فرزندان (Calculate the Weighted Entropy of the Children)

پس از تقسیم داده‌ها بر اساس یک ویژگی خاص، برای ارزیابی میزان کاهش در ناخالصی (افزایش اطلاعات)، آنتروپی وزن‌دار فرزندان (مجموعه‌های حاصل از تقسیم) را محاسبه می‌کنیم.

آنتروپی وزن‌دار، مجموع آنتروپی هر زیرمجموعه است که با نسبت نمونه‌های موجود در آن زیرمجموعه وزن‌دهی شده است. به عبارت دیگر، میزان ناخالصی هر زیرمجموعه در کل محاسبه در نظر گرفته می‌شود.

فرمول آنتروپی وزن‌دار (Hweighted) پس از یک تقسیم به صورت زیر است:

$$Hweighted = \Sigma (|Dj| / |D|) * H(Dj)$$

در این فرمول:

- |Dj|: تعداد نمونه‌ها در زیرمجموعه‌ی Dj است.
- |D|: تعداد کل نمونه‌ها در مجموعه داده‌ی اصلی است.
- H(Dj): آنتروپی (ناخالصی) زیرمجموعه‌ی Dj است.
- m: تعداد کل زیرمجموعه‌های ایجاد شده پس از تقسیم (معمولاً برای ویژگی‌های دودویی m=2 است).

۴ .محاسبه‌ی افزایش اطلاعات(Calculate Information Gain)

افزایش اطلاعات (Information Gain) نشان‌دهنده‌ی کاهش ناخالصی (Entropy) است که با تقسیم داده‌ها بر اساس یک ویژگی خاص به دست می‌آید .به عبارت دیگر، میزان مفید بودن یک ویژگی برای طبقه‌بندی را اندازه‌گیری می‌کند .

هرچه افزایش اطلاعات برای یک ویژگی بیشتر باشد، آن ویژگی برای بهبود درخت تصمیم و جداسازی بهتر نمونه‌ها مفیدتر است.

فرمول محاسبه‌ی افزایش اطلاعات (IG) به صورت زیر است:

$$\boldsymbol{IG(D,A) = H(D) - Hweighted(D|A)}$$

در این فرمول:

- IG(D, A): افزایش اطلاعات حاصل از تقسیم مجموعه داده‌ی D بر اساس ویژگی A
- H(D): آنتروپی مجموعه داده‌ی اصلی D ناخالصی قبل از تقسیم
- Hweighted(D|A): آنتروپی وزن‌دار فرزندان

۵ .انتخاب ویژگی با بالاترین افزایش اطلاعات:

پس از محاسبه‌ی افزایش اطلاعات برای هر یک از ویژگی‌ها، آن‌ها را با هم مقایسه می‌کنیم.

ویژگی‌ای که بالاترین افزایش اطلاعات را به دست دهد، به عنوان معیار تقسیم در گره‌ی جاری انتخاب می‌شود. به عبارت دیگر، این ویژگی، بهترین ویژگی برای تقسیم داده‌ها در آن گره‌ی خاص است، زیرا بیشترین میزان کاهش در ناخالصی (افزایش اطلاعات) را ایجاد می‌کند.

۶ .ایجاد گره‌ی تصمیم یا برگ:(Leaf Node)

بعد از انتخاب ویژگی با بالاترین افزایش اطلاعات، دو حالت ممکن است وجود داشته باشد:

- کسب اطلاعات ناچیز (Maximum Information Gain = 0) اگر حداکثر افزایش اطلاعات برای تمام ویژگی‌ها صفر باشد، به این معنی است که تقسیم بیشتر داده‌ها بر اساس هیچ ویژگی‌ای منجر به بهبود نمی‌شود . در این حالت، یک برگ در انتهای مسیر گره‌ی جاری ایجاد می‌شود و برچسب آن، رایج‌ترین کلاس در مجموعه داده‌ی فعلی در نظر گرفته می‌شود.
- تقسیم با ویژگی منتخب (Create Decision Node) در صورتی که حداکثر افزایش اطلاعات برای یک ویژگی فرض کنید A بزرگتر از صفر باشد، به این معنی است که با تقسیم داده‌ها بر اساس ویژگیA ، می‌توان میزان ناخالصی را کاهش داد .بنابراین، در گره‌ی جاری یک گره‌ی تصمیم ایجاد می‌کنیم و آن را با ویژگی A و آستانه‌ی تقسیم (برای ویژگی‌های پیوسته) مرتبط می‌سازیم .سپس برای هر زیرمجموعه‌ی ایجاد شده پس از تقسیم بر

اساسA ، فرآیند را به صورت بازگشتی (recursive) از مرحله‌ی ۱ (محاسبه‌ی آنتروپی گره‌ی والد) تکرار می‌کنیم.

با تکرار این فرایند برای تمام گره‌های داخلی درخت، در نهایت یک درخت تصمیم با گره‌های تصمیم و برگ‌ها به دست می‌آید که برای طبقه‌بندی نمونه‌های جدید قابل استفاده است.

معیار ناخالصی (Gini)

معیار ناخالصی جینی (Gini Impurity) یکی دیگر از معیارهای اندازه‌گیری میزان درهم‌ریختگی (ناخالصی) یک مجموعه داده است که در درخت‌های تصمیم مشابه با آنتروپی عمل می‌کند. در حالی که آنتروپی بر اساس تئوری اطلاعات محاسبه می‌شود، از معیار جینی در الگوریتم‌هایی مانند CART (Classification and Regression Tree) استفاده می‌شود. هر دوی این معیارها به دنبال کمیت‌سازی میزان بی‌نظمی یا ناخالصی یک مجموعه داده هستند، اما روش محاسبه‌ی آن‌ها متفاوت است.

فرمول محاسبه‌ی ناخالصی جینی

ناخالصی جینی برای یک مجموعه داده‌ی D با n کلاس به صورت زیر محاسبه می‌شود:

$$\boldsymbol{Gini(D) = 1 - \Sigma(pi\text{^}2)}$$

که در آن:

- pi: نسبت نمونه‌ها (احتمال) متعلق به کلاس iام است. هرس با آلفای پیچیدگی هزینه (Cost Complexity Pruning - CCP)

هرس (Pruning) تکنیکی است که برای جلوگیری از overfitting در درخت‌های تصمیم با کاهش پیچیدگی آن‌ها به کار می‌رود. یکی از روش‌های موثر هرس، هرس با پیچیدگی هزینه (Cost Complexity Pruning - CCP) است که از پارامتری به نام آلفا (α) برای کنترل تعادل بین پیچیدگی درخت و عملکرد آن استفاده می‌کند.

مراحل هرس با پیچیدگی هزینه (CCP)

در ادامه مراحل هرس با پیچیدگی هزینه (CCP) آورده شده است:

۱. ایجاد یک درخت تصمیم کامل (Full Decision Tree): ابتدا یک درخت تصمیم کامل با حداقل محدودیت برای تقسیم گره‌ها (مثلاً حداقل تعداد نمونه در هر زیرمجموعه) ساخته می‌شود. این درخت حداکثر توانایی برای جداسازی داده‌ها را بر اساس ویژگی‌ها در اختیار دارد، اما ممکن است مستعد overfitting باشد.

۲. محاسبه‌ی هزینه‌ی پیچیدگی (Cost Complexity): برای هر زیرمجموعه (گره داخلی) در درخت، یک مقدار هزینه بر اساس تعداد برگ‌های آن زیرمجموعه و یک پارامتر جریمه‌دهی (ضریب جریمه) محاسبه می‌شود. این پارامتر جریمه‌دهی، میزان پیچیدگی مدل را در نظر می‌گیرد و به عنوان یک عامل بازدارنده در برابر ایجاد زیرمجموعه‌های بیش از حد در نظر گرفته می‌شود.

۳. هرس با آلفای از پیش تعیین‌شده (Pruning with Predefined Alpha): پارامتر آلفا (α) یک مقدار غیرمنفی است که میزان هرس را کنترل می‌کند. با افزایش آلفا، تعداد زیرمجموعه‌های هرس‌شده (حذف‌شده) افزایش می‌یابد و در نتیجه، درخت تصمیم ساده‌تر می‌شود.

۴. انتخاب زیرمجموعه‌ی بهینه (Choosing the Optimal Subtree): با توجه به مقادیر هزینه‌های محاسبه‌شده در مرحله‌ی ۲، زیرمجموعه‌ای به عنوان زیرمجموعه‌ی بهینه انتخاب می‌شود که کمترین هزینه را بر اساس معیار خاصی (مثلاً خطای اعتبارسنجی) به همراه جریمه‌ی پیچیدگی در نظر گرفته شده توسط آلفا به دست می‌دهد.

۷. جایگزینی زیرمجموعه‌ی هرس‌شده (Replacing the Pruned Subtree): زیرمجموعه‌ی بهینه (که ممکن است یک برگ باشد) جایگزین کل زیرمجموعه‌ی هرس‌شده در درخت تصمیم می‌شود.

۸. تکرار مراحل ۳ تا ۵ (Iteration of Steps 3-5): مراحل ۳ تا ۵ به صورت بازگشتی (recursive) برای تمام زیرمجموعه‌های باقی‌مانده در درخت تکرار می‌شوند تا زمانی که دیگر زیرمجموعه‌ی قابل هرس با توجه به آلفای از پیش تعیین‌شده وجود نداشته باشد.

در نهایت، این فرایند منجر به ایجاد یک درخت تصمیم هرس‌شده می‌شود که تعادل بهتری بین دقت و پیچیدگی مدل برقرار می‌کند.

۹. ایجاد درخت تصمیم کامل (Generate the Tree)

در مرحله‌ی اول، یک درخت تصمیم کامل با حداقل محدودیت برای تقسیم گره‌ها (مثلاً حداقل تعداد نمونه در هر زیرمجموعه) بر اساس داده‌های آموزشی ساخته می‌شود. این درخت با حداکثر توانایی برای جداسازی داده‌ها بر اساس ویژگی‌ها شکل می‌گیرد.

۱۰. محاسبه‌ی هزینه‌ی پیچیدگی برای زیرمجموعه‌ها (Calculate the Cost Complexity for Subtrees)

برای هر زیرمجموعه‌ی داخلی (گره) در درخت تصمیم کامل که با Tt نشان داده می‌شود، هزینه‌ی پیچیدگی (Cost Complexity) با فرمول زیر محاسبه می‌شود:

$$R\alpha(Tt) = R(Tt) + \alpha \times |Tt|$$

در این فرمول:

- R(Tt): میزان خطای طبقه‌بندی (Misclassification Error) برای زیرمجموعه‌ی Tt است. این مقدار نشان می‌دهد که چه تعداد نمونه در آن زیرمجموعه به اشتباه طبقه‌بندی شده‌اند.
- α (alpha): پارامتر پیچیدگی است که میزان جریمه در نظر گرفته شده برای افزایش تعداد برگ‌ها در درخت را کنترل می‌کند. هرچه α بزرگتر باشد، جریمه‌ی بیشتری برای پیچیدگی در نظر گرفته می‌شود و در نتیجه تمایل به ایجاد زیرمجموعه‌های کمتر (درخت ساده‌تر) وجود دارد.
- |Tt|: تعداد برگ‌های موجود در زیرمجموعه‌ی Tt است.

۱۱. هرس زیرمجموعه‌ها (Prune Subtrees)

پس از محاسبه‌ی هزینه‌ی پیچیدگی برای تمام زیرمجموعه‌های درخت (شامل ریشه‌ی درخت)، فرایند هرس با هدف ساده‌سازی درخت و جلوگیری از overfitting آغاز می‌شود:

- حذف زیرمجموعه‌ی با کمترین هزینه‌ی پیچیدگی: زیرمجموعه‌ی داخلی Tt (که ممکن است ریشه‌ی درخت نیز باشد) که کمترین مقدار Rα(Tt) را در بین همه‌ی زیرمجموعه‌ها دارد، برای حذف انتخاب می‌شود. این زیرمجموعه به احتمال زیاد بیش از حد پیچیده است و با حذف آن، تعادل بهتری بین دقت و پیچیدگی مدل برقرار می‌شود.

۱۲. انتخاب آلفای بهینه (Select the Optimal Alpha)

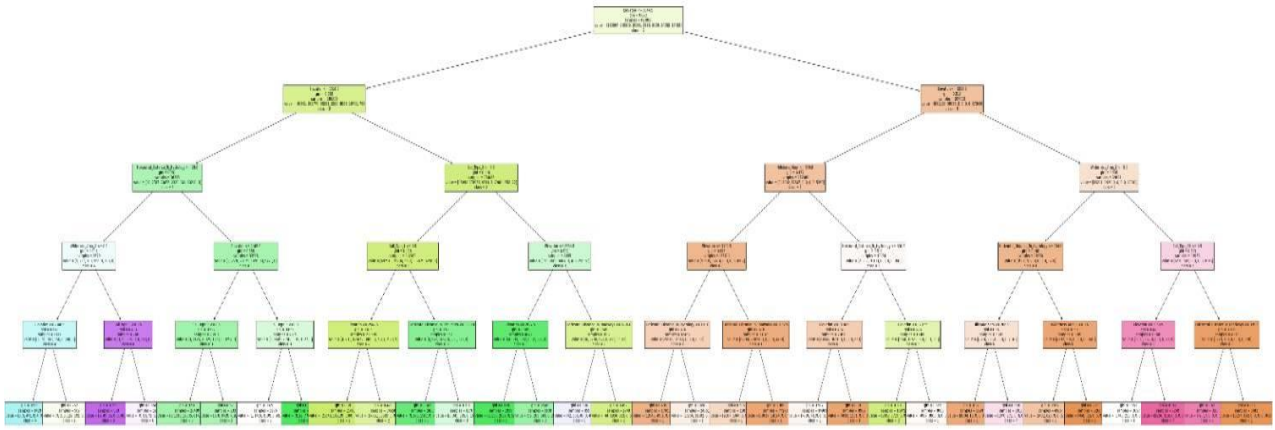
پارامتر آلفا (α) نقش مهمی در میزان هرس و در نتیجه، پیچیدگی نهایی درخت تصمیم دارد. مقدار بهینه برای α لزوماً مشخص نیست و باید با استفاده از اعتبارسنجی(Cross-Validation) تعیین شود.

- اعتبارسنجی: داده‌های آموزشی به مجموعه‌های آموزشی و اعتبارسنجی تقسیم می‌شوند. برای هر مقدار احتمالی α، یک درخت تصمیم با CCP ساخته می‌شود و عملکرد آن (مثلاً دقت) بر روی مجموعه‌ی اعتبارسنجی اندازه‌گیری می‌شود.
- بهترین آلفا: مقدار آلفای بهینه، مقداری است که بهترین تعادل را بین دقت مدل بر روی مجموعه‌ی اعتبارسنجی و سادگی درخت (تعداد برگ‌ها) برقرار می‌کند. به عبارت دیگر، آلفای بهینه، کمترین خطای اعتبارسنجی را با کمترین پیچیدگی درخت به دست می‌دهد.

۱۳. هرس نهایی درخت (Prune the Tree)

پس از انتخاب آلفای بهینه با استفاده از اعتبارسنجی، از این مقدار برای هرس نهایی درخت تصمیم استفاده می‌شود. به این ترتیب، زیرمجموعه‌هایی که هزینه‌ی پیچیدگی آن‌ها (با توجه به αی به دست آمده) از یک آستانه‌ی مشخص (مثلاً

میانگین هزینه‌ی پیچیدگی کل درخت) بیشتر باشد، حذف می‌شوند. در نهایت، درخت تصمیم هرس‌شده با پیچیدگی مناسب و عملکرد قابل قبول بر اساس داده‌های آموزشی به دست می‌آید.



طبق توضیحات گفته شده، ابتدا براسا بهره اطلاعاتی یک گره رشیه را به دست می آوریم سپس شروع به مقایسه کردن میکنیم برای مثال در گره ریشه شرط تصمیم گیری ما Elevation<=2510.5 می باشد.

در اینجا Elevation براساس بهره اطلاعات انتخاب میشود و برای انتخاب فرزند یک زیر مجموعه از مقادیر یکتاElevation نگاه میسازیم سپس برای زیر مجموعه آن هم همینکار را انجام می دهیم. در زیرمجموعه ها اگر لیبل ها برابر بود گره برگ ساخته خواهد شد. اگر برابر نبودند باید یک گره تصمیم بسازیم و مراحل مرحله اول را دوباره انجام دهیم و بهره اطلاعاتی ان زیرمجموع را به دست بیاوریم.

همچنین ما ۷ کلاس داریم که باتوجه به مقدار بیشتر سمپل ها در ان کلاس، کلاس ما برگزیده خواهد شد. با درست بودن شرط به گره سمت چپی گره یا فرزند می رویم. گره فرزند سمت چپی Horizontal_distance_to_Hydrology<=15 اگر درست بود به گره سمت چپ می رویم. تعدا نمونه ها ۳۶۳۱۵ می باشد که تعدا بیشتری از نمونه ها به کلاس ۳ تعلق دارد و مقدار gini مقدار ناخالصی را به دست می آورد. در اینجا سعی کردیم که با افزایش که عمق درخت بتوانیم مقدار ناخالصی را کاهش دهیم. با تنظیم پارامتر ها میتوانیم به ناخالصی کم در گره های برگ بشویم. و تعدا سمپل هارا متعلق به یک کلاس کنیم.

سپس به ویژگی دیگر wildrenes_area_2 می رسیم که wildrenes_area_2<=0 به سمت شاخه سمت چپ میریم. تعداد نمونه ها ۲۶۷۹ و متعلق به کلاس ۴ هست اما ناخالصی بسیار زیادی دارد.

گره بعدی گره Elvation<=2340.5 می باشد در صورت صحیح بودن به گره سمت چپ ریخته خواهد شد. انقدر این کار را ادامه می دهیم تا به عمق ۵ و به گره برگ برسیم.

باید توجه داشت که مسیری که انتخاب کردیم تا انتها می رویم تا تکلیف مسیر و گره برگ را مشخص کنیم سپس شاخه های آن را بررسی میکنیم و بعد از اتمام و مشخص کردن گره برگ تمام شاخه ها به مسیر جدیدی میرویم.

.۲

دقت (Accuracy) در یادگیری ماشین، نسبت تعداد نمونه‌های به‌درستی پیش‌بینی‌شده به کل نمونه‌ها است .این معیار نشان می‌دهد که یک مدل طبقه‌بندی تا چه حد در پیش‌بینی کلاس نمونه‌های جدید موفق عمل کرده است.

برای محاسبه‌ی دقت، فرمول زیر استفاده می‌شود:

دقت = (تعداد نمونه‌های درست طبقه‌بندی‌شده) / (کل تعداد نمونه‌ها)

این معیار به صورت درصد نیز بیان می‌شود (بین ۰ تا ۱۰۰ درصد) .هرچه مقدار دقت بالاتر باشد، عملکرد مدل در طبقه‌بندی نمونه‌ها بهتر است.

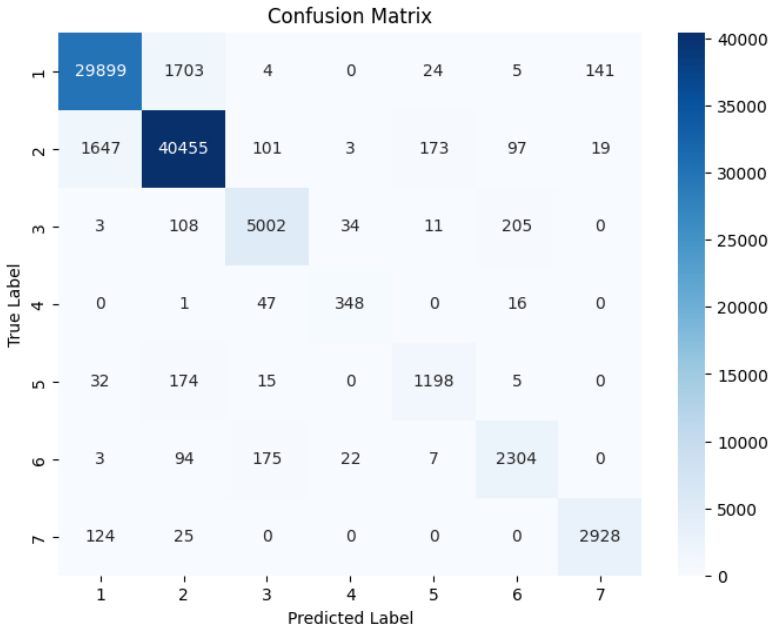
نتیجه:

$$Accuracy = 0.9424224343675418$$

ماتریس اشتباه(Confusion Matrix)

تعریف :ماتریس اشتباه (Confusion Matrix) در یادگیری ماشین، جدول خلاصه‌ای از نتایج پیش‌بینی در مسائل طبقه‌بندی است .این جدول، تعداد پیش‌بینی‌های صحیح و نادرست را به تفکیک برای هر کلاس، با اعداد نشان می‌دهد.

ماتریس اشتباه به تجسم عملکرد یک مدل طبقه‌بندی در تمایز قائل شدن بین کلاس‌های مختلف کمک می‌کند .با بررسی این جدول، می‌توان دریافت که مدل تا چه حد در تشخیص نمونه‌های متعلق به هر کلاس موفق عمل کرده است.



دقت (Precision)، فراخوان (Recall)، و نمره‌ی F1 (F1-Score)

در مسائل طبقه‌بندی با دو یا چند کلاس، ارزیابی عملکرد مدل تنها با استفاده از دقت (Accuracy) همیشه کافی نیست. معیارهای دیگری مانند دقت (Precision)، فراخوان (Recall)، و نمره‌ی F1 (F1-Score) برای تحلیل جزئی‌تر نتایج پیش‌بینی به کار می‌روند.

۱. دقت (Precision):

دقت، نسبت نمونه‌های مثبتِ به‌درستی پیش‌بینی‌شده به کل نمونه‌هایی که توسط مدل مثبت پیش‌بینی شده‌اند را نشان می‌دهد. به عبارت دیگر، دقت بیانگر این است که از میان نمونه‌هایی که مدل آن‌ها را مثبت تشخیص داده است، چه تعداد واقعا مثبت بوده‌اند.

فرمول محاسبه‌ی دقت به صورت زیر است:

$$Precision = True\ Positives/True\ Positives\ +\ False\ Positives$$

۲. فراخوان (Recall):

فراخوان، نسبت نمونه‌های مثبتِ واقعی که توسط مدل به‌درستی پیش‌بینی شده‌اند به کل نمونه‌های مثبت واقعی را نشان می‌دهد. به عبارت دیگر، فراخوان بیانگر این است که از کل نمونه‌های مثبت واقعی در مجموعه داده، مدل چه تعداد از آن‌ها را به درستی مثبت تشخیص داده است.

فرمول محاسبه‌ی فراخوان به صورت زیر است:

$$Recall = \frac{True\ Positives}{True\ Positives} + False\ Negatives$$

F1-Score

در برخی موارد، تمایز قائل شدن بین دقت و فراخوان اهمیت زیادی دارد .برای مثال، در تشخیص یک بیماری خطرناک، ممکن است ترجیح دهیم تمام موارد مثبت واقعی را شناسایی کنیم، حتی اگر این امر منجر به افزایش تعداد موارد مثبت کاذب شود (کاهش دقت) .نمره‌ی F1 (F1-Score) معیاری است که به طور متعادل دقت و فراخوان را در نظر می‌گیرد و به همین دلیل در چنین سناریوهایی مفید است.

$$F1 - Score = 2 \times Precision \times Recall/Precision\ +\ Recall$$

Classification Report:

	precision	recall	f1-score	support
1	0.94	0.94	0.94	31776
2	0.95	0.95	0.95	42495
3	0.94	0.93	0.93	5363
4	0.86	0.84	0.85	412
5	0.85	0.84	0.84	1424
6	0.88	0.88	0.88	2605
7	0.95	0.95	0.95	3077
accuracy			0.94	87152
macro avg	0.91	0.91	0.91	87152
weighted avg	0.94	0.94	0.94	87152

تفسیر دقت(Accuracy)

دقت کلی مدل، ۲۴.۹۴ درصد است، که نشان می‌دهد تقریباً ۲۴.۹۴ درصد از کل پیش‌بینی‌ها صحیح هستند .این دقت بالا حاکی از عملکرد خوب مدل بر روی مجموعه داده‌ی داده‌شده است.

ماتریس اشتباه(Confusion Matrix)

ماتریس اشتباه، نمایشی دقیق از نتایج پیش‌بینی برای هر کلاس ارائه می‌کند. این جدول، تفکیکی از موارد زیر را برای هر کلاس نشان می‌دهد:

مثبت‌های واقعی (True Positives) تعداد نمونه‌هایی که در واقعیت به کلاس موردنظر تعلق دارند و توسط مدل نیز به درستی تشخیص داده شده‌اند.

مثبت‌های کاذب (False Positives) تعداد نمونه‌هایی که در واقعیت به کلاس موردنظر تعلق ندارند، اما مدل آن‌ها را به اشتباه به عنوان این کلاس پیش‌بینی کرده است.

منفی‌های واقعی (True Negatives) تعداد نمونه‌هایی که در واقعیت به کلاس موردنظر تعلق ندارند و توسط مدل نیز به درستی تشخیص داده شده‌اند.

منفی‌های کاذب (False Negatives) تعداد نمونه‌هایی که در واقعیت به کلاس موردنظر تعلق دارند، اما مدل آن‌ها را به اشتباه به عنوان کلاسی دیگر پیش‌بینی کرده است.

برای مثال، در کلاس ۱، تعداد ۲۹۸۹۹ نمونه‌ی مثبت واقعی (True Positive) و ۱۷۰۳ نمونه‌ی منفی کاذب (False Negative)وجود دارد.

ماتریس اشتباه نشان می‌دهد که مدل در تشخیص نمونه‌های متعلق به کلاس‌های مختلف عملکرد خوبی داشته و نرخ خطای طبقه‌بندی (Misclassification Rate) پایین است.

دقت، فراخوان، و نمره‌ی :F1 عملکرد بر اساس کلاس

عملکرد بر اساس کلاس(Class-wise Performance)

این بخش به ارزیابی عملکرد مدل برای هر کلاس به صورت جداگانه می‌پردازد.

High Precision برای کلاس های ۲ و ۷: نشان می دهد که وقتی مدل این کلاس ها را پیش بینی می کند، احتمال درستی آن بسیار زیاد است.

دقت کمتر برای کلاس ۴ و ۵: نشان می دهد که مثبت کاذب بیشتری برای این کلاس ها نسبت به سایرین وجود دارد.میانگین کل (Macro Average) در مقابل میانگین وزنی (Weighted Average)

تفسیر عملکرد مدل درخت تصمیم

عملکرد بر اساس کلاس:

- کلاس‌های ۱، ۲ و ۷: مدل می‌تواند اکثر نمونه‌های این کلاس‌ها را با موفقیت شناسایی کند. این موضوع به دلیل مقادیر بالای دقت و فراخوان برای این کلاس‌ها است.
- کلاس ۴ و ۵: مدل در تشخیص نمونه‌های این کلاس‌ها عملکرد ضعیف‌تری دارد. Recall کمتر برای این کلاس‌ها نشان می‌دهد که مدل تعداد بیشتری از نمونه‌های واقعی این کلاس‌ها را به اشتباه به عنوان کلاس‌های دیگر پیش‌بینی می‌کند.

امتیاز F1:

- امتیاز F1 میانگین وزنی Precision و Recall است. این امتیاز هنگام جستجوی تعادل بین دقت و یادآوری مفید است.
- کلاس‌های ۱، ۲، ۳، ۶ و ۷: امتیاز F1 بالا برای این کلاس‌ها نشان می‌دهد که مدل تعادل خوبی را در قابلیت پیش‌بینی برای این کلاس‌ها دارد. به عبارت دیگر، مدل می‌تواند نمونه‌های این کلاس‌ها را با دقت بالا و نرخ خطای پایین شناسایی کند.
- کلاس ۴ و ۵: امتیاز F1 پایین‌تر برای این کلاس‌ها نشان می‌دهد که مدل در ایجاد تعادل در دقت و یادآوری بیشتر با این کلاس‌ها مبارزه می‌کند. به عبارت دیگر، مدل در تشخیص نمونه‌های این کلاس‌ها با چالش بیشتری روبرو است.

دقت، یادآوری، امتیاز F1 در حدود ۰.۹۱: این مقادیر میانگین این معیارها را در همه کلاس‌ها بدون در نظر گرفتن نسبت هر کلاس به دست می‌آورند. به طور کلی، این مقادیر نشان می‌دهند که مدل عملکرد خوبی در تشخیص نمونه‌های همهی کلاس‌ها دارد.

میانگین وزنی:

- دقت، یادآوری، امتیاز F1 در حدود ۰.۹۴: این مقادیر پشتیبانی (تعداد نمونه‌های واقعی برای هر کلاس) را در نظر می‌گیرند و معیار واقعی‌تری از عملکرد کلی مدل ارائه می‌دهند. به طور کلی، این مقادیر نشان می‌دهند که با در نظر گرفتن توزیع نمونه‌ها در کلاس‌های مختلف، عملکرد کلی مدل همچنان بسیار خوب است.

  - مدل درخت تصمیم عملکرد خوبی در طبقه‌بندی نمونه‌ها در این مجموعه داده دارد.
  - مدل می‌تواند اکثر نمونه‌های کلاس‌های ۱، ۲ و ۷ را به درستی شناسایی کند.
  - مدل در تشخیص نمونه‌های کلاس‌های ۴ و ۵ با چالش بیشتری روبرو است.
  - به طور کلی، مدل عملکردی قوی و قابل اعتماد برای این کار طبقه‌بندی ارائه می‌دهد.

میانگین کل (Macro Average):

میانگین کل، عملکرد مدل را بر روی تمام کلاس‌ها به صورت مساوی در نظر می‌گیرد، بدون توجه به تعداد نمونه‌های موجود در هر کلاس. در این خروجی، دقت، فراخوان، و نمره‌ی F1 میانگین کل، همگی ۰.۹۱ هستند. این نتیجه نشان می‌دهد که به طور متوسط، مدل در تشخیص نمونه‌های همهی کلاس‌ها عملکرد خوبی دارد.

میانگین وزنی (Weighted Average):

میانگین وزنی، عملکرد مدل را بر روی تمام کلاس‌ها در نظر می‌گیرد، اما تعداد نمونه‌های موجود در هر کلاس را نیز لحاظ می‌کند. به عبارت دیگر، کلاس‌هایی با تعداد نمونه‌ی بیشتر، تأثیر بیشتری بر روی میانگین نهایی خواهند داشت. در این خروجی، دقت، فراخوان، و نمره‌ی F1 میانگین وزنی، همگی ۰.۹۴ هستند. این نتیجه نشان می‌دهد که با در نظر گرفتن توزیع نمونه‌ها در کلاس‌های مختلف، عملکرد کلی مدل همچنان بسیار خوب است.

- هر دو میانگین کل (۰.۹۱) و میانگین وزنی (۰.۹۴) نشان می‌دهند که مدل عملکرد خوبی دارد.
- میانگین وزنی (۰.۹۴) به دلیل در نظر گرفتن تعداد نمونه‌ها در هر کلاس، تصویری واقعی‌تر از عملکرد کلی مدل ارائه می‌دهد.

مدل درخت تصمیم دقت بالایی در حدود ۹۴.۲۴ درصد را نشان می‌دهد، با مقادیر قوی برای دقت، فراخوان، و نمره‌ی F1 در اکثر کلاس‌ها می باشد.

کلاس ۱:

واقعی مثبت (TP): 29899

کاذب مثبت (FP): $1703 + 4 + 0 + 24 + 5 + 141 = 1877$

کاذب های منفی (FN): 1647

واقعی منفی (TN): دیگر مقادیر همه

کلاس ۲:

TP: 40455

FP: $1647 + 101 + 3 + 173 + 97 + 19 = 2040$

FN: 1703

TN: دیگر مقادیر همه

کلاس ۳:

TP: 5002

FP: $3 + 108 + 34 + 11 + 205 = 361$

FN: 101

TN: دیگر مقادیر تمام

کلاس ۴:

TP: 348

FP: $0 + 1 + 47 + 16 = 64$

FN: 34

TN: دیگر مقادیر تمام

کلاس ۵:

TP: 1198

FP: $32 + 174 + 15 + 5 = 226$

FN: 173

TN: دیگر مقادیر همه

۶ کلاس:

TP: 2304

FP: $3 + 94 + 175 + 22 + 7 = 301$

FN: 205

TN: دیگر مقادیر همه

۷ کلاس:

TP: 2928

FP: $124 + 25 = 149$

FN: 19

TN: دیگر مقادیر همه

عملکرد مدل درخت تصمیم روی مجموعه داده

مدل درخت تصمیم با دستیابی به دقت کلی بالای ۹۴.۲۴ درصد، عملکرد خوبی را در این مجموعه داده نشان می‌دهد. مدل به طور کلی به مقادیر بالای دقت، فراخوان و نمره‌ی F1 دست می‌یابد، به ویژه برای کلاس‌های ۱، ۲ و ۷. با این حال، در کلاس‌های ۴ و ۵ جای بهبود وجود دارد، جایی که دقت و فراخوان مدل نسبتاً پایین‌تر هستند و نشان‌دهنده‌ی طبقه‌بندی اشتباه بیشتر در این کلاس‌ها است.

به طور کلی، درخت تصمیم برای این کار طبقه‌بندی مؤثر بوده و استحکام و قابلیت اطمینان را به خصوص در کلاس‌های با فراوانی بالاتر نشان می‌دهد. معیارهای ارزیابی پیشنهاد می‌کنند که تنظیم بیشتر و احتمالاً رسیدگی به عدم تعادل کلاس‌ها (توزیع نابرابر نمونه‌ها در هر کلاس) ممکن است عملکرد را به خصوص برای کلاس‌های با عملکرد پایین‌تر بهبود بخشد.

قسمت دوم

۱. max_depth (حداکثر عمق)

- مقدار کوچک (مثلاً ۳):

- اثر: درخت با تعداد کمتری از سطوح ایجاد می‌شود که ممکن است منجر به دقت کمتر شود زیرا درخت به اندازه کافی جزئیات داده‌ها را نمی‌بیند.
- **مزیت:**
  - کاهش پیچیدگی مدل: احتمال بیش‌برازش را کاهش می‌دهد.
  - سرعت آموزش و پیش‌بینی بیشتر می‌شود.

- مقدار بزرگ (مثلاً None یا مقدار بسیار زیاد):

- اثر: درخت تا زمانی که تمامی برگ‌ها به یک کلاس برسند یا هیچ ویژگی دیگری برای تقسیم وجود نداشته باشد، رشد می‌کند. این ممکن است منجر به بیش‌برازش شود زیرا درخت بسیار پیچیده و دقیق به داده‌های آموزش شده خواهد بود.
- مزیت:
  - توانایی مدل برای یادگیری تمامی جزئیات داده‌های آموزش: ممکن است دقت مدل را بر روی داده‌های آموزش افزایش دهد.

۲. min_samples_split (حداقل نمونه برای تقسیم)

- مقدار کوچک (مثلاً ۲):

- اثر: درخت می‌تواند به سرعت تقسیم شود حتی اگر تعداد نمونه‌ها در هر گره بسیار کم باشد، که می‌تواند منجر به بیش‌برازش شود.

- مزیت:
  - مدل بسیار حساس به تغییرات داده‌ها می‌شود و می‌تواند ساختارهای پیچیده‌تری را یاد بگیرد.

- مقدار بزرگ (مثلاً ۵۰):

- اثر: درخت نمی‌تواند تقسیم‌های کوچکی انجام دهد مگر اینکه تعداد نمونه‌ها در یک گره به حداقل مقدار تعیین شده برسد، که می‌تواند منجر به کم‌برازش شود زیرا مدل قادر به یادگیری تمامی جزئیات داده‌ها نخواهد بود.
- مزیت:
  - کاهش پیچیدگی مدل و کاهش احتمال بیش‌برازش: مدل‌های ساده‌تر و پایدارتر می‌شوند که احتمال کمتری برای تنظیم بیش از حد بر روی داده‌های آموزش دارند.

۳. تاثیر تغییر پارامترهای هرس کردن (Pruning) بر نتایج و مزایای آن

- ccp_alpha (هزینه پیچیدگی هرس):

- مقدار کوچک (مثلاً ۰.۰۰۰۱):

- اثر: درخت تصمیم ممکن است تقریباً به طور کامل رشد کند، که می‌تواند منجر به بیش‌برازش شود زیرا تعداد گره‌های برگ زیادی خواهد داشت.
- مزیت:
  - درخت قادر خواهد بود تمامی جزئیات موجود در داده‌های آموزش را بیاموزد.

- مقدار بزرگ (مثلاً ۰.۱):

- اثر: بسیاری از گره‌های برگ حذف می‌شوند که می‌تواند منجر به کم‌برازش شود زیرا مدل ساده‌تر و قادر به یادگیری تمامی جزئیات داده‌های آموزش نخواهد بود.
- مزیت:
  - کاهش پیچیدگی مدل، افزایش تعمیم‌پذیری و کاهش احتمال بیش‌برازش: دقت مدل در پیش‌بینی داده‌های ناشناخته افزایش می‌یابد.

تغییر مقادیر پارامترهای max_depth و min_samples_split تاثیر قابل توجهی بر پیچیدگی و عملکرد مدل دارد.

پارامتر ccp_alpha با کنترل هرس کردن، می‌تواند به بهینه‌سازی تعادل بین بیش‌برازش و کم‌برازش کمک کند.

استفاده مناسب از این پارامترها می‌تواند منجر به ایجاد مدل‌هایی شود که هم دقت بالایی دارند و هم به خوبی تعمیم می‌یابند.

۳. چگونه روش‌های جنگل تصادفی و AdaBoost می‌توانند نتایج را بهبود بخشند؟

جنگل تصادفی و AdaBoost دو روش یادگیری ماشینی قدرتمند هستند که به عنوان روش‌های یادگیری گروهی شناخته می‌شوند. این روش‌ها با ترکیب چندین مدل یادگیری ضعیف‌تر (معمولاً درختان تصمیم) به طور قابل توجهی می‌توانند عملکرد مدل‌های طبقه‌بندی را ارتقا دهند.

۱. جنگل تصادفی:

- کاهش بیش‌برازش: درختان تصمیم به دلیل ماهیتشان، به خصوص زمانی که عمیق و پیچیده باشند، مستعد بیش‌برازش بر روی داده‌های آموزشی هستند. جنگل تصادفی با میانگین‌گیری پیش‌بینی‌های تعداد زیادی درخت تصمیم که هر کدام بر روی زیرمجموعه‌ای تصادفی از داده‌ها آموزش دیده‌اند، این مشکل را حل می‌کند. این روش منجر به پیش‌بینی‌های پایدارتر و دقیق‌تر می‌شود و از وابستگی مدل به نویز موجود در داده‌های آموزشی می‌کاهد.
- مدیریت تنوع: در جنگل تصادفی، هر درخت تصمیم از طریق نمونه‌گیری تصادفی زیرمجموعه‌ای از داده‌ها (شامل ردیف‌ها و ستون‌ها) را برای آموزش دریافت می‌کند. این تصادفی بودن به کاهش واریانس در پیش‌بینی‌ها و جلوگیری از وابستگی مدل به یک زیرمجموعه خاص از داده‌ها کمک می‌کند. تنوع ایجاد شده توسط این روش، قدرت تعمیم مدل را افزایش می‌دهد و از بیش‌برازش آن جلوگیری می‌کند.
- بهبود دقت: با ترکیب پیش‌بینی‌های چندین درخت تصمیم که هر کدام دیدگاه متفاوتی از داده‌ها دارند، جنگل تصادفی می‌تواند به دقت بیشتری نسبت به یک درخت تصمیم واحد دست یابد. این روش با جمع‌آوری آراء درختان مختلف، پیش‌بینی نهایی را انجام می‌دهد که به طور معمول دقیق‌تر و قابل اعتمادتر است.
- اهمیت ویژگی‌ها: جنگل تصادفی ابزاری ارزشمند برای محاسبه اهمیت هر ویژگی در مجموعه داده‌ها ارائه می‌دهد. این روش با تجزیه و تحلیل نحوه‌ی عملکرد هر درخت تصمیم بر روی ویژگی‌های مختلف، بینش مفیدی در مورد نقش هر ویژگی در پیش‌بینی نهایی ارائه می‌دهد. این اطلاعات می‌تواند برای انتخاب ویژگی‌های مهم و کاهش ابعاد مجموعه داده‌ها مفید باشد.

۲. AdaBoost:

- تمرکز بر نمونه‌های دشوار AdaBoost با تمرکز بر نمونه‌هایی که در طبقه‌بندی توسط مدل‌های قبلی با مشکل مواجه شده‌اند، عملکرد کلی را ارتقا می‌دهد. این روش به نمونه‌های اشتباه طبقه‌بندی شده وزن بیشتری اختصاص می‌دهد و مدل‌های بعدی را مجبور می‌کند تا بیشتر بر روی این موارد دشوار تمرکز کنند. این فرآیند تکراری به بهبود عملکرد مدل بر روی نمونه‌های چالش‌برانگیز کمک می‌کند و از نادیده گرفتن این نمونه‌ها توسط مدل جلوگیری می‌کند.
- کاهش سوگیری AdaBoost با تنظیم وزن نمونه‌ها به طور متوالی بر اساس عملکرد مدل در هر مرحله، سوگیری مدل را کاهش می‌دهد. این روش با تمرکز بیشتر بر روی خطاها، مدل را به یادگیری از اشتباهات خود

و تصحیح آن‌ها در مراحل بعدی آموزش ترغیب می‌کند. این امر منجر به عملکرد بهتر در مجموعه داده‌هایی می‌شود که ممکن است یک درخت تصمیم واحد در آن‌ها با سوگیری مواجه شود.

- ترکیب یادگیرهای ضعیف AdaBoost با ترکیب چندین یادگیر ضعیف (معمولاً درختان تصمیم کم عمق) به یک طبقه‌بند قوی دست می‌یابد. هر یادگیر ضعیف بر روی داده‌های وزنی آموزش دیده و بر اساس دقت خود به مدل نهایی کمک می‌کند. این روش یادگیری گروهی می‌تواند به طور قابل توجهی دقت کلی طبقه‌بندی را به طور قابل توجهی بهبود بخشد.
- مقاومت در برابر بیش‌برازش: در حالی که AdaBoost هنوز ممکن است بر روی داده‌های پر نویز بیش‌برازش کند، به طور کلی در مقایسه با یک درخت تصمیم واحد مقاومت بیشتری در برابر بیش‌برازش دارد. تمرکز بر بهبود تکراری و تصحیح خطاها به ایجاد یک مدل قوی‌تر کمک می‌کند.
- جنگل تصادفی با کاهش بیش‌برازش، مدیریت تنوع از طریق نمونه‌گیری تصادفی، و جمع‌آوری پیش‌بینی‌های چندین درخت برای دقت و پایداری بهتر، نتایج را بهبود می‌بخشد.
- AdaBoost عملکرد را با تمرکز بر نمونه‌های دشوار، کاهش سوگیری از طریق تنظیم وزن‌های تکراری، و ترکیب یادگیرهای ضعیف به یک طبقه‌بند قوی بهبود می‌بخشد.
- هر دو روش از قدرت چندین مدل برای غلبه بر محدودیت‌های یک درخت تصمیم واحد استفاده می‌کنند و منجر به طبقه‌بندی‌های دقیق‌تر، پایدارتر و قوی‌تر می‌شوند.

سوال ۴

میانگین گیری Micro و Macro در طبقه بندی چند کلاسه باsklearn

هنگامی که تعداد زیادی کلاس دارید یا به یک خلاصه کلی تر از عملکرد کلی نیاز دارید، استفاده از میانگین های میکرو یا ماکرو گزینه بهتری می تواند باشد.

در طبقه بندی چند کلاسه، برای ارزیابی عملکرد یک الگوریتم طبقه بندی از معیارهایی مانند دقت(Accuracy) ، فراخوان (Recall)و F1-Score استفاده می شود .نحوه میانگین گیری این معیارها بر تفسیر نتایج تاثیرگذار است .در کتابخانه scikit-learn، اصطلاحات Micro و Macro به روش های مختلفی برای میانگین گیری معیارهای عملکرد اشاره دارد.

تفاوت های کلیدی بین میانگین گیری Macro و Micro:

دقت مدل Gaussian Naive Bayes در مجموعه تست ۸۲.۹۳٪ است که نشان می دهد مدل به درستی تقریباً ۸۳٪ از نمونه های مجموعه آزمایش را طبقه بندی کرده است. دقت یک معیار رایج برای ارزیابی عملکرد کلی یک مدل طبقه بندی است، اما ممکن است به خودی خود کافی نباشد، به خصوص زمانی که با مجموعه داده های نامتعادل سروکار داریم.

ماتریس درهم ریختگی (Confusion Matrix) جزئیات پیش بینی های مدل را به صورت خلاصه ارائه می دهد:

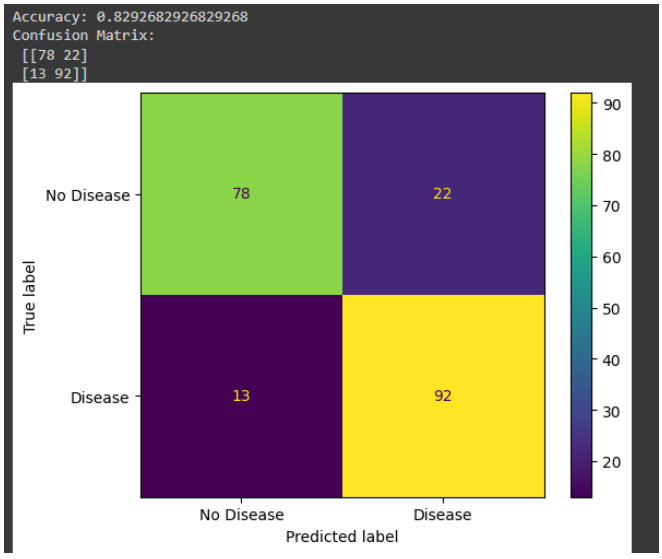
تحقیق های منفی صحیح (TN) ۷۸ :این مدل برای ۷۸ بیمار به درستی "عدم بیماری" را پیش بینی کرد.

تحقیق های مثبت غلط (FP) ۲۲: این مدل به اشتباه برای ۲۲ بیمار که در واقع بیماری ندارند، "بیماری" را پیش بینی کرد.

تحقیق های منفی غلط (FN) ۱۳: این مدل به اشتباه برای ۱۳ بیمار که در واقع بیماری دارند، "عدم بیماری" را پیش بینی کرد.

تحقیق های مثبت صحیح (TP) ۹۲ : این مدل به درستی برای ۹۲ بیمار، "بیماری" را پیش بینی کرد.

همانطور که از ماتریس درهم ریختگی مشاهده می کنید، مدل تعداد کمی بیشتر از موارد تحقیق های مثبت و منفی صحیح را دارد که نشان دهنده عملکرد خوب است. با این حال، همچنان تعداد قابل توجهی از تحقیق های مثبت و منفی غلط وجود دارد که به ویژه در زمینه مراقبت های بهداشتی که چنین اشتباهاتی می تواند حیاتی باشد، باید مورد توجه قرار گیرد.



گزارش طبقه بندی مدل

دقت(Precision)

- نسبت تشخیص های مثبت که واقعا صحیح بوده اند.
  - برای بدون بیماری(0) :0.86
  - برای بیماری(1) : 0.81
- دقت برای بدون بیماری بالاتر است، به این معنی که زمانی که مدل بدون بیماری را پیش بینی می کند، در ۸۶ درصد موارد درست است.

فراخوان (Recall) یا حساسیت(Sensitivity)

- نسبت موارد مثبت واقعی که به درستی شناسایی شده اند.
  - برای بدون بیماری(0) : 0.78
  - برای بیماری :0.88
- فراخوان برای بیماری بالاتر است، یعنی مدل در شناسایی موارد واقعی بیماری (۸۸٪) عملکرد بهتری دارد.

نمره F1

- میانگین هارمونیک دقت و فراخوان، که تعادلی بین این دو معیار ایجاد می کند.
  - برای بدون بیماری(0) : 0.82
  - برای بیماری(1) :0.84
- نمره F1 یک معیار واحد ارائه می دهد که دقت و فراخوان را برای هر کلاس متعادل می کند.

فراوانی

- تعداد وقوع واقعی هر کلاس در مجموعه تست.
  - برای بدون بیماری100 :(0)
  - برای بیماری105 :(1)
- این نشان می دهد که مجموعه داده نسبتاً متعادل است.

میانگین گیری میکرو و ماکرو

- **میانگین گیری میکرو:**
  - سهم همه کلاس ها را برای محاسبه معیار میانگین جمع می کند.
  - مجموع موارد مثبت واقعی، مثبت کاذب، منفی کاذب و منفی واقعی را در کل محاسبه می کند و سپس دقت، فراخوان و نمره F1 را محاسبه می کند.
  - زمانی مفید است که بخواهید عملکرد کلی مدل را در همه طبقات بدون اهمیت دادن به هیچ کلاس خاصی ارزیابی کنیم.
- **میانگین گیری ماکرو:**
  - این معیار را برای هر کلاس به طور مستقل محاسبه می کند و سپس میانگین آن را می گیرد.
  - با همه کلاس ها به طور مساوی رفتار می کند، صرف نظر از فراوانی آنها.

- زمانی مفید است که بخواهید عملکرد مدل را در هر کلاس به طور مستقل ارزیابی کنید و سپس نتایج را میانگین بگیرید.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.78      0.82       100
           1       0.81      0.88      0.84       105

    accuracy                           0.83       205
   macro avg       0.83      0.83      0.83       205
weighted avg       0.83      0.83      0.83       205
```

میانگین گیری میکرو(Micro-Averaging)

- **تعریف :**در این روش، عملکرد همه کلاس ها در نظر گرفته شده و با وزن مساوی برای هر نمونه، میانگین کل محاسبه می شود.
- **نحوه محاسبه :**برای محاسبه این نوع میانگین، مجموع کل موارد مثبت واقعی(True Positives) ، منفی کاذب (False Negatives) و مثبت کاذب (False Positives) در نظر گرفته می شود و سپس با استفاده از این مقادیر کل، میانگین معیار مورد نظر محاسبه می شود.
- **کاربرد :**زمانی که می خواهید عملکرد کلی طبقه بندی کننده را بدون در نظر گرفتن توزیع نامتعادل کلاس ها ارزیابی کنید، مفید است.
- **مزایا :**در سناریوهایی که توزیع کلاس ها بسیار نامتوازن است، معیار متعادل تری ارائه می دهد.
- **معایب :**می تواند تحت تاثیر عملکرد کلاس های غالب (کلاس هایی که نمونه بیشتری دارند) قرار گیرد.

۲ .میانگین گیری ماکرو(Macro-Averaging)

- **تعریف :**در این روش، ابتدا برای هر کلاس به طور جداگانه میانگین معیار مورد نظر محاسبه شده و سپس میانگین بدون وزن این مقادیر برای همه کلاس ها بدست می آید.
- **نحوه محاسبه :**برای هر کلاس به طور جداگانه دقت، فراخوان یا F1 محاسبه شده و سپس میانگین حسابی این مقادیر برای همه کلاس ها بدست می آید.

- **کاربرد :**زمانی که می خواهید عملکرد هر کلاس را به طور مساوی ارزیابی کنید، صرف نظر از تعداد نمونه های آن کلاس، مفید است.
- **مزایا :**با وزن دهی برابر به همه کلاس ها، معیاری از عملکرد ارائه می دهد که به عملکرد همه کلاس ها حساس است.
- **معایب :**در صورتی که توزیع کلاس ها به شدت نامتعادل باشد، می تواند گمراه کننده باشد، زیرا فراوانی هر کلاس را در نظر نمی گیرد.

میکرو :عملکرد همه کلاس ها را در نظر می گیرد و برای سناریوهایی با توزیع نامتعادل و تمرکز بر عملکرد کلی مناسب است.

ماکرو :برای هر کلاس به طور جداگانه میانگین گیری می کند و سپس میانگین کل را محاسبه می کند .برای زمانی که می خواهید همه کلاس ها را به طور مساوی در نظر بگیرید، مناسب است.