به نام خدا

عنوان:

تکلیف یازدهم یادگیری ماشین

استاد:

دکتر پدرام

دانشجو:

محمدعلی مجتهدسلیمانی

٤٠٣٣٩٠٤٥٠٤

تاریخ:

١٤٠٣/١٠/٩

# سوال ۱

یادگیری تجمعی یک رویکردی است که با چندین مدل که اصطلاحا به آنها weak learner یا base model گفته میشود به نوعی با همدیگر ادغام شده اند برای حل یک مسئله خاص، استفاده از یادگیری تجمعی مزایای مختلفی دارد:

✓ کاهش variance (کاهش احتمال بیش پردازش):

مدل های پیچیده به طور کلی مستعد overfitting هستند به این معنی که روی داده های آموزشی خیلی خوب هستند اما روی داده های از قبل دیده نشده ضعیف عمل میکنند.

روش های جمعی مثل bagging میانگین پیشبینی از مدل های مختلف را که روی بخش های متفاوتی از dataset آموزش دیده اند میگیرد. این فرآیند میانگین گیری سبب میشود که خطای هر مدل کمتر شود و باعث میشود به طور کلی واریانس کاهش پیدا کند و مدل تعمیم پذیر خواهد بود.

✓ کاهش bias:

روش های جمعی مثل boosting سعی میکنند مدل ها را به صورت توالی قرار دهند تا هر مدل تمرکز کند تا خطای مدل قبلی را کاهش بدهد. این رویکرد باعث میشود که bias کاهش پیدا کند و accuracy بالا برود.

این رویکرد به دلیل تنوعی که از مدل های مختلف و خطا های مختلف دارد باعث میشود که دقت (accuracy) افزایش پیدا میکند. ۲ روش bagging, boosting از روش های مرسوم در این رویکرد هستند.

این رویکرد قابلیت اطمینان و مقاومت بیشتری نسبت به داده های نویزی و outliers دارد به این دلیل که اگر یک مدل به سبب داده های outlier خطا کند مدل های دیگر آن را جبران میکنند و منجر به پیشبینی های پایدار تر و قابل اطمینان تر میشود. همچنین تنوع مدل ها باعث میشود آنها کمتر با یک مجموعه مشابه outlier دچار مشکل شوند.

با کاهش واریانس و bias این مدل ها تعمیم پذیری بهتری خواهند داشت. و میتوانند داده های پیچیده را مدیریت بکنند.

## سوال ۲

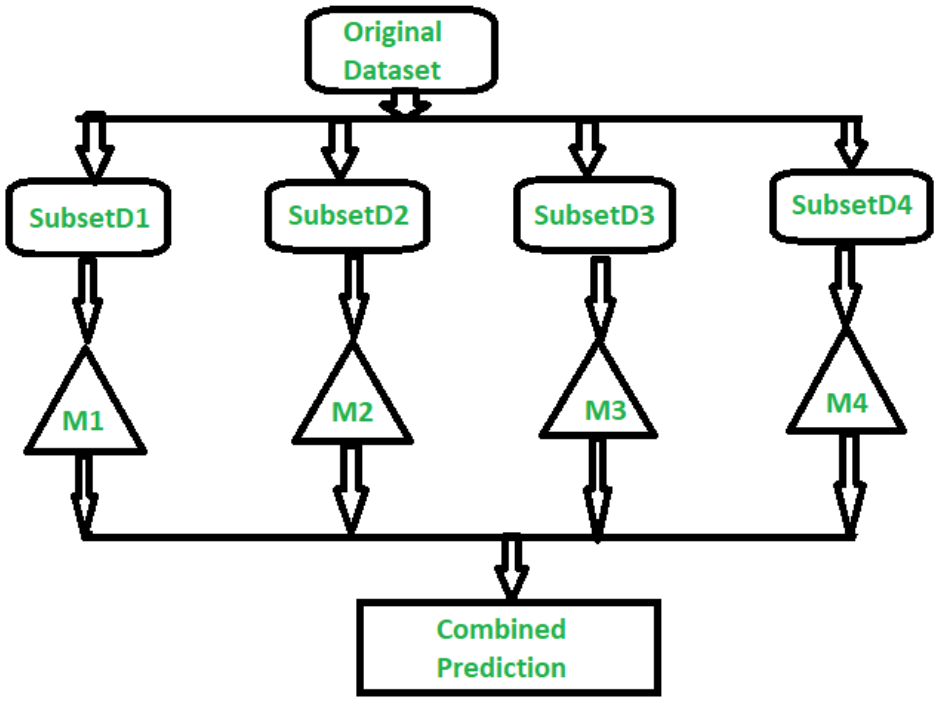
درخت های تصمیم به طور ذاتی دارای واریانس بالا و bias پایین هستند. یعنی میتوانند الگوهای پیچیده را به خوبی یاد بگیرند. اما به شدت به داده های آموزشی حساس هستند و ممکن است بیش از حد روی داده های آموزش بیش پردازش شوند و دچار overfit شویم و دارای واریانس بالا هستند. یعنی با یک تغییر کوچک در مجموعه آموزشی، ساختار درخت تغییر چشمگیری میکند.

در روش bagging ما با استفاده از روش bootstrap چندین مجموعه داده آموزشی میسازیم که با انتخاب و جایگزینی دوباره همراه است این روش. هر درخت روی یکی از این مجموعه ها آموزش میبینند.

به وسیله میانگین گیری از تخمین هر کدام از این درخت ها (یا استفاده از رای گیری برای کلاس بندی) این روش میتواند باعث کاهش واریانس شود. این رویکرد تجمعی باعث میشود که مدل کمتر به یک قسمت خاص از مجموعه داده آموزشی حساس شود.

نکته مهم دیگر این است که برای بیشینه کارآمدی درخت ها باید خطای هر درخت مستقل از بقیه درخت ها باشد تا بتوانیم تنوع خوبی روی داده های مختلف داشته باشیم برای پروسه میانگین گیری.

نکته مهم دیگری که باید اشاره بکنم این است که درخت های تصمیم به عنوان مدل های ناپایدار (unstable learners) شناخته میشوند. به این معنی که وقتی دارند روی مجموعه داده های مختلف آموزش میبینند به علت اینکه با یک تغییر کوچک ساختار درخت تغییر پیدا میکند، درخت ها به هم مشابه نیستند و ما تنوع داریم در مدل های خودمان و همه از یک جنس نخواهند بود.



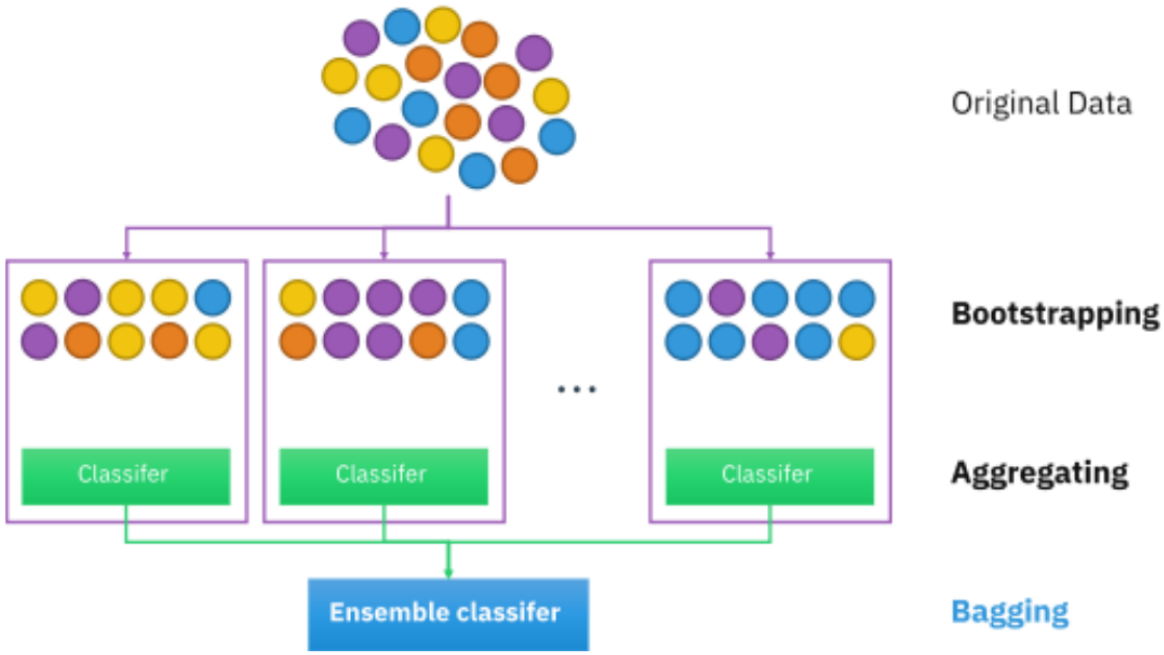

## سوال ۳

وقتی روش bagging ما بهترین عملکرد ممکن را دارد که این شرایط را داشته باشیم: ۱. مدل های base learner ما واریانس بالا داشته باشند به این معنی که حساس هستند به تنوع داده های آموزشی و مستعد بیش پردازش هستند. ۲. مدل های ما دارای تنوع باشند به این معنی که خطای ایجاد شده توسط هر کدام کاملا مستقل از سایر درخت ها باشد. وقتی این وضعیت برقرار

باشد میانگین گیری از پیشبینی به صورت موثر واریانس را کاهش میدهد و باعث افزایش دقت میشود.

پس تنوع بین مدل های پایه بسیار مهم است برای عملکرد خوب روش bagging حالا چگونه این تنوع را بدست بیاوریم؟ :

Bootstrap sampling: مکانیزم مرکزی که باعث تنوع در این روش میشود bootstrap است به این معنی که ما در این روش یک سری مجموعه های آموزشی میسازیم که هر کدام از دیتاست اصلی انتخاب میشوند قرار داده میشوند و بعد دوباره به دیتاست اصلی برمیگردند و در دور دوم دوباره امکان انتخاب آنها است، که با احتمال ۶۳ درصد هر داده میتواند در زیر مجموعه های آموزشی حضور پیدا کنند. این مکانیزم باعث میشود ما اطمینان پیدا کنیم که برداشت های مختلفی از توزیع داده های مجموعه اصلی داشته باشیم. این باعث میشود که تنوع ایجاد شود مخصوصا برای مدل های ناپایدار (unstable) که به ازای کوچکترین تغییر در مجموعه آموزشی ساختار آنها تغییر میکند.



# سوال ٤

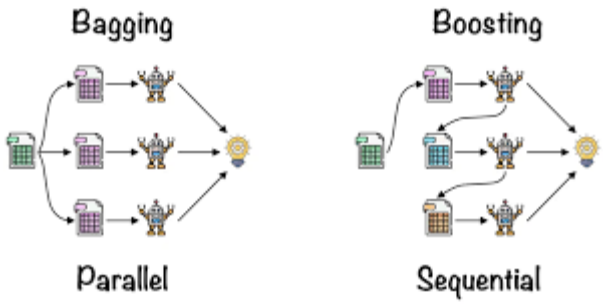
دیدگاه boosting به عنوان یک تکنیک یادگیری تجمعی برای بدست آوردن دقت بالا همراه با یک trade-offیی بین bias و واریانس معرفی شد. این دیدگاه از یک سری مدل های به اصطلاح weak learner استفاده میکند. این مدل ها بسیار ضعیف هستند به طوری که جواب آنها فقط کمی نسبت به یک حدس تصادفی بهتر است، یک strong learner کلاس بندی است که میتواند تصمیم درستی نسبت به کلاس واقعی داده بگیرد.

Boosting ادعا میکند که میتواند با ترکیب weak learner ها خروجی خوبی تولید کند. این دیدگاه طراحی شده است تا bias کاهش دهد. این دیدگاه یک دنباله ای از مدل های آموزشی میسازد و هر مدل روی بخشی تمرکز میکند که مدل قبلی در آن حوزه ضعف داشت. این رویکرد باعث میشود که مدل به آرامی دقت بالاتری بدست بیاورد و الگو های پیچیده را تری پیدا بکند.

در حالی که bagging برای کاهش واریانس استفاده میشود اما شاید در کاهش bias موفق نباشد بخصوص وقتی که مدل های پایه (base learners) از قبل پیچیده باشند. اما boosting ادعا میکند که میتواند bias را کاهش بدهد بخصوص وقتی که دلیل خطای زیاد ما bias باشد.

Boosting رویکرد یادگیری تطبیقی را ارائه میکند که باعث میشود که وزن بیشتری به داده های اشتباه کلاس بندی شده (misclassified) در هر دور داده شود، با این شیوه boosting مدل های بعدی را مجبور میکند که توجه بیشتری به این داده های اشتباه داشته باشد و یادگیری موثر تری نسبت به اشتباه مدل های قبلی داشته باشد.

عکس زیر به صورت دقیق میتواند فرق میان boosting و bagging را به نمایش بگذارد:




در boosting ما دنباله ای از مدل ها را داریم برخلاف bagging که به صورت موازی و مستقل از هم مدل ها را آموزش میداد. در این رویکرد ما داده های اشتباه دسته بندی وزن بیشتری خواهیم داد. در شروع وزن همه به یک اندازه است. این رویکرد نسبت به خطا حساس است و مدل های بعدی را سعی میکند به اشتباهات مدل های قبلی متمرکز کند. پس به طور کلی مدل های weak learner را ترکیب میکند تا یک strong learner بسازد، باعث کاهش bias میشود و نسبت به داده های پیچیده و سخت تطبیق پیدا میکند. XGBoost نمونه ای از این الگوریتم ها خواهد بود.

## سوال ۵

در رویکرد AdaBoost، یک دنباله ای از مدل های base learner آموزش میبینند. در هر iteration، وزن نمونه های آموزشی را تنظیم میکند و داده های اشتباه وزن بیشتری پیدا میکنند. این امر باعث میشود مدل های بعدی تمرکزی بیشتری بر روی داده های اشتباه دسته بندی شده داشته باشند. در نهایت با استفاده از وزن هر مدل ما یک voting برای پیشبینی نهایی خواهیم داشت این وزن هر چه قدر مدل دقیق تر باشد بیشتر خواهد بود.

انتخاب مدل های weak learner که فقط کمی از یک حدس تصادفی بهتر عمل میکنند باعث میشود که کمتر میل به overfitting داشته باشند بخاطر ساده بودن آنها و میتوان به وسیله آنها مدل های پیچیده ساخت، همچنین چون این مدل ها خطا های مختلفی تولید میکنند

باعث تنوع در وزن ها میشود که برای پیاده سازی رویکرد یادگیری تجمعی بسیار مناسب است. اگر همه مدل ها یک نوع خطا تولید بکنند زیاد مناسب نخواهد بود. اگر ما از strong learner ها استفاده بکنیم مانند شبکه های عمیق عصبی به عنوان یک base learner به سرعت دچار overfitting میشویم چون خطای خیلی کمی روی مجموعه های آموزشی داریم. همچنین استفاده از این مدل ها برخلاف weak learner ها تنوع چندانی نخواهند داشت زیرا این مدل ها الگو های مختلف را یاد گرفته اند و باعث نمیشود خطا های مختلفی تولید شود.

انتخاب درخت های تصمیم میتواند گزینه خوبی باشد زیرا جز مدل های ضعیف شناخته میشوند که میتوانند الگو های ساده را شناسایی کنند. سریع آموزش میبینند و میتوانند خطا های مختلفی را تولید بکنند با هر بار تغییر وزن های داده‌ها در مجموعه آموزشی.

پس به طور کلی سعی باید در انتخاب weak learner ها باشد نه انتخاب strong learner ها.

## سوال ٦

مقاومت در برابر overfitting رویکرد AdaBoost دلایل مختلفی دارد:

✓ بیشینه کردن margin:

این رویکرد میتواند margin را بیشینه کند. حاشیه (margin) به فاصله ی بین آخرین داده تا مرز تصمیم گیری گفته میشود. بیشینه کردن این فاصله باعث افزایش تعمیم پذیری و کاهش overfitting میشود.

✓ ماهیت تصادفی (stochastic):

تغییر وزن های آموزشی که در هر iteration انجام میشود، یک ماهیت تصادفی به فرآیند آموزشی اضافه میکند. این ماهیت باعث میشود که مدل داده های آموزشی را حفظ نکند و فقط به آنها حساس نشود و تعمیم پذیری بهتری پیدا میکند.

✓ تمرکز روی داده های اشتباه کلاس بندی شده:

با تمرکز کردن روی داده های اشتباه کلاس بندی شده در هر iteration این رویکرد به تدریج روی داده های سخت متمرکز میشوند و باعث میشود مدل نسبت به داده های درست دسته بندی شده یا به اصطلاح آسان overfit نشود و باعث میشود مرز تصمیم قابل اطمینان باشد.

❖ اگر overfitting اتفاق افتاد میتوانیم مراحل زیر داشته باشیم:

✓ کاهش تعداد iteration:

ساده ترین روش همین کم کردن تعداد دور ها خواهد بود. میتوانیم از validation set یا cross-validation استفاده بکنیم تا تعداد بهینه دور ها را بدست بیاوریم.

✓ افزایش پیچیدگی مدل های base:

اگه از مدل های بسیار ساده استفاده میکنیم میتوانیم از مدل های کمی پیچیده تر (مثلا درخت هایی با عمق ۲ یا ۳) استفاده بکنیم این امر سبب کاهش overfitting در مواقعی میشود.

همچنین از اقداماتی مثل early stopping که عملکرد مدل را روی validation set بررسی میکنیم و اگر داشت در حین کاهش خطا روی مجموعه آموزشی، صعودی میشد عملکرد مدل را متوقف کنیم. یک اقدام دیگر کاهش نرخ یادگیری خواهد بود که باعث میشود

میزان مشارکت هر weak learner را کنترل کنیم تا اگر یک مدل مستعد بیش پردازش بود جلوی آن را بگیریم. یک اقدام دیگر میتواند افزایش اندازه مجموعه آموزشی باشد.