پیچیدگی محاسباتی الگوریتم‌های یادگیری ماشین

فاطمه رحیمی [*۱]، محمد قاسم زاده[۲]

۱- دانشجوی کارشناسی ارشد هوش مصنوعی، دانشکده مهندسی کامپیوتر، دانشگاه یزد

۲- استاد تمام علوم و مهندسی کامپیوتر، دانشکده مهندسی کامپیوتر، دانشگاه یزد

چکیده

در این مقاله روش‌های اساسی هوش مصنوعی از جنبه درجه پیچیدگی محاسباتی مجانبی مورد بررسی قرار می‌گیرند. در این راستا روش‌های درخت تصمیم، جنگل تصادفی، رگرسیون خطی، بیز ساده، ماشین بردار پشتیبان و K همسایه نزدیک مورد توجه قرار گرفته‌اند. ابتدا عملکرد این الگوریتم‌ها به صورت مختصر مورد بحث قرار گرفته و سپس پیچیدگی محاسباتی آن‌ها بررسی شده است. مقایسه تطبیقی نتایج نشان می‌دهد که روش‌های درخت تصمیم، K همسایه نزدیک و بیز ساده کمترین درجه پیچیدگی زمانی برای مرحله آموزش را دارند، در حالی که روش‌های درخت تصمیم و رگرسیون خطی در خصوص مرحله آزمون از همه روش‌های دیگر درجه پیچیدگی زمانی کمتری دارند. همچنین در این مقاله نشان داده می‌شود که با افزایش تعداد نمونه‌ها و تعداد ویژگی‌ها به ترتیب استفاده از الگوریتم‌های ماشین بردار پشتیبان و رگرسیون خطی توصیه نمی‌شود.

کلمات کلیدی: یادگیری ماشین، پیچیدگی محاسباتی، مرتبه زمانی، مرحله آزمون، مرحله آموزش

۱. مقدمه

یادگیری ماشین شاخه‌ای از هوش مصنوعی است که به سیستم‌های رایانه‌ای اجازه می‌دهد بدون واسطه از مثال‌ها، داده‌ها و تجربیات بیاموزند. الگوریتم‌های یادگیری ماشین از طریق توانمندسازی رایانه‌ها برای انجام وظایف خاص به صورت هوشمند، می‌توانند فرایندهای پیچیده‌ای را به دلیل یادگیری از داده‌ها به جای پیروی از قوانین از پیش برنامه‌ریزی شده انجام دهند این حوزه از دانش به طور اخص در دهه اخیر به طور روزافزون مورد توجه محققین بوده است[1,2].

برای انتخاب الگوریتم مناسب عوامل متعددی باید بررسی شود. از جمله این عوامل می‌توان نوع مسئله، تعداد داده‌ها و ویژگی‌ها، طبقه‌بندی یا رگرسیون بودن مسئله و پیچیدگی محاسباتی الگوریتم را نام برد. الگوریتم‌های یادگیری ماشین معمولاً مجموعه‌های داده بزرگ و پیچیده‌ای را پردازش می‌کنند که حاوی تعداد قابل توجهی از ویژگی‌ها هستند تا اطلاعات معناداری درباره مفهوم هدف یا همان کلاس‌ها استخراج کنند. در بیشتر موارد، این الگوریتم‌ها دچار مشکلات تأخیر و پیچیدگی محاسباتی در هنگام پردازش چنین مجموعه داده‌های پیچیده به دلیل وجود ویژگی‌ها و داده‌های زیاد می‌شوند[3].

[*] **Corresponding author: فاطمه رحیمی، دانشجوی کارشناسی ارشد هوش مصنوعی**
Email: fateme.rahimi@stu.yazd.ac.ir

جهت رفع این مشکل باید در هنگام انتخاب الگوریتم به پیچیدگی محاسباتی آن توجه کرد. انتظار می‌رود که الگوریتم‌ها حتی برای مسائل پیچیده، مجموعه داده بسیار بزرگ و تعداد ویژگی‌های زیاد که باعث پیچیدگی محاسباتی از مرتبه نمایی می‌شوند نیز به طور مؤثر به یادگیری ادامه دهند. این موارد به این دلیل حائز اهمیت‌اند که پیچیدگی محاسباتی از مرتبه نامناسب می‌تواند موفقیت یا شکست یک مسئله یادگیری ماشین را تعیین کند[4].

بر خلاف سابقه طولانی استفاده از الگوریتم‌های یادگیری ماشین و کاربرد بسیار آن‌ها در حوزه‌های مختلف علوم و همچنین اهمیت پیچیدگی محاسباتی جهت انتخاب یک الگوریتم برای حل مسئله مورد نظر، کار کمی در مورد پیچیدگی محاسباتی این الگوریتم‌ها به صورت واحد انجام شده است. در این مقاله سعی بر آن بوده که پس از توضیح کلی عملکرد چند الگوریتم یادگیری ماشین، پیچیدگی محاسباتی آن‌ها ارائه و با یکدیگر مقایسه شود. این کار برای مرحله آموزش و آزمون به تفکیک انجام شده است. سپس ضمن این بررسی تطبیقی، نشان داده خواهد شد که برای مسائلی با داده‌های بسیار بزرگ یا تعداد ویژگی زیاد، بهتر است از چه الگوریتم‌هایی استفاده شود.

۲. مروری بر روشهای اساسی یادگیری ماشین

یادگیری ماشین رشته‌ای از علوم و محاسبات کامپیوتر است که به کامپیوترها قابلیت یادگیری بدون برنامه‌نویسی صریح را می‌دهد[2,5]. این حوزه شامل الگوریتم‌هایی است که از آن‌ها معمولاً در زمینه‌های مختلف برای حل مسائل دشواری استفاده می‌شود که بر اساس رویکردهای کامپیوتری به راحتی قابل حل نیستند. این الگوریتم‌ها تلاش می‌کنند که نظم و الگوی خاصی را در داده‌های موجود بیابند و از آن، جهت پیش‌بینی وضعیت داده‌های جدید استفاده کنند.

دانشمندان داده معتقدند که هیچ نوع الگوریتم واحدی وجود ندارد که برای حل یک مسئله بهترین باشد؛ چرا که الگوریتم به کار گرفته شده بسته به نوع مسئله‌ای که باید حل شود، تعداد متغیرها یا ویژگی‌ها و نوع مدلی که برای آن مناسب است انتخاب می‌شود[5]. بنابراین قبل از اینکه درجه پیچیدگی روش‌های مبنایی یادگیری ماشین را مورد بررسی قرار دهیم، مروری بر عملکرد آن‌ها خواهیم داشت. در این رابطه الگوریتم‌های درخت تصمیم، جنگل تصادفی، رگرسیون خطی، بیز ساده، ماشین بردار پشتیبان و K همسایه نزدیک مورد توجه قرار می‌گیرند.

۲-۱. درخت تصمیم

شاید به عنوان ساده‌ترین الگوریتم یادگیری ماشین، بتوان از روش درخت تصمیم نام برد. از درخت تصمیم به عنوان روشی جهت ایجاد سیستم‌های طبقه‌بندی بر اساس متغیرهای کمکی و یا برای توسعه الگوریتم‌های پیش‌بینی یک متغیر هدف استفاده می‌شود. عملکرد این الگوریتم به این صورت است که به طور خودکار از یک مجموعه داده معین یک درخت تصمیم می‌سازد و به طور معمول هدف آن یافتن درخت تصمیم کمینه با به حداقل رساندن خطای تعمیم است[6]. این روش داده‌ها را به وسیله بخش‌های شاخه‌مانند طبقه‌بندی می‌کند و یک درخت وارونه با گره ریشه در بالا، گره‌های داخلی جهت نمایش ویژگی‌ها در میان و گره‌های برگ جهت نمایش یک کلاس یا یک برچسب در پایین می‌سازد.

روش کار درخت تصمیم به این صورت است که داده از گره ریشه وارد مسیر پردازش می‌شود و از هر گره میانی که با ویژگی آن منطبق باشد وارد شاخه مرتبط شده و به گره بعدی می‌رود. این روند تا زمانی که داده به یکی از گره‌های برگ برسد و کلاسش تعیین بشود، ادامه دارد. از دیگر موارد استفاده درخت تصمیم می‌توان به انتخاب متغیر، ارزیابی اهمیت نسبی متغیرها، مدیریت مقادیر گم شده و پیش‌بینی اشاره کرد.

۲-۲. جنگل تصادفی

الگوریتم جنگل تصادفی یکی از الگوریتم‌های مشهور یادگیری ماشین است که استفاده از آن آسان است و منجر به دریافت نتایج خوبی می‌شود. به همین دلایل از این الگوریتم هم برای طبقه‌بندی و هم برای رگرسیون استفاده می‌شود. این الگوریتم متشکل از طبقه‌بندهایی با ساختار درختی است که خروجی هر درخت به ازای هر نمونه ورودی، یک رای به محبوب‌ترین یا پرتکرارترین کلاس است[7]. جنگل تصادفی از چندین درخت تصمیم عمیق تشکیل شده است. هریک از درخت‌های تصمیم در این الگوریتم کاملاً رشد خواهد کرد و نیازی به هرس نخواهد داشت. در نهایت نتایج به دست آمده از این درخت‌ها با یکدیگر ادغام می‌شود و جوابی با اطمینان بالاتر به دست می‌آید. هرچه تعداد درخت‌های موجود در جنگل تصادفی بیشتر باشد، نتایج دقیق‌تری حاصل می‌شود و الگوریتم روی داده‌ها دچار بیش‌برازش نمی‌شود.

۲-۳. رگرسیون خطی

شاید یکی از رایج‌ترین و جامع‌ترین الگوریتم‌های آماری و یادگیری ماشینی رگرسیون خطی باشد. رگرسیون خطی برای یافتن رابطه خطی بین یک یا چند پیش‌بینی کننده استفاده می‌شود. از این الگوریتم معمولاً جهت تخمین مقدار واقعی بر اساس متغیرهای پیوسته استفاده می‌شود. به عبارتی این الگوریتم جایی مورد استفاده قرار می‌گیرد که بتوان مقادیر پیش‌بینی شده را در برابر متغیرهای ورودی چندگانه اندازه‌گیری و مدل‌سازی کرد. رگرسیون خطی روشی است که روابط خطی را بین متغیرهای وابسته و مستقل برقرار می‌کند و جهت ارزیابی و مدل‌سازی داده استفاده می‌شود. این الگوریتم روابط بین متغیرهای وابسته و متغیرهای مستقل را از تحلیل و یادگیری تا نتایج آموزشی فعلی مدل می‌کند[8].

۲-۴. بیز ساده

الگوریتم بیز ساده از مشهورترین و مورداستفاده‌ترین الگوریتم‌های یادگیری ماشین است که از قضیه بیز جهت حل مسائل طبقه‌بندی استفاده می‌کند. ماهیت این الگوریتم احتمالاتی است و نتایج آن نیز برحسب احتمالات بیان می‌شود. همچنین در این الگوریتم یک فرض اساسی مبنی بر این که ویژگی‌ها هیچ ارتباطی با یکدیگر ندارند، در نظر گرفته می‌شود که ممکن است در دنیای واقعی چنین نباشد. از این رو به این الگوریتم بیز ساده می‌گویند. این الگوریتم به دلیل سادگی عملکرد و دقت بالا جایگاه مهمی در بین الگوریتم‌های طبقه‌بندی دارد و از اثربخشی و استحکام بالایی برخوردار است[9].

قاعده بیز به زبان ساده به این صورت مطرح می‌شود: «احتمال وقوع یک پیشامد A با توجه به این که رخداد B قبلا رخ داده است، برابر است با احتمال وقوع رخداد B با توجه به این که A قبلاً رخ داده است ضرب در احتمال وقوع A و تقسیم بر احتمال وقوع B » که به صورت زیر نشان داده می‌شود.

$$P(A|B)=P(A)\times\frac{P(B|A)}{P(B)} \qquad (1)$$

در رابطه فوق، $P(A|B)$ را احتمال پسین یا مؤخر، $P(B|A)$ را احتمال درست‌نمایی و $P(A)$ را احتمال پیشین یا مقدم می‌نامند. این الگوریتم احتمال تعلق هر نمونه به کلاس‌های مسئله را بررسی می‌کند و کلاسی را انتخاب می‌کند که احتمال تعلق نمونه به آن کلاس بیشتر باشد.

۲-۵. ماشین بردار پشتیبان

الگوریتم ماشین بردار پشتیبان مدلی را ایجاد می‌کند که حداکثر فضای بین نزدیک‌ترین نمونه‌های آموزشی از دو کلاس را با مرزهایی مشخص کند. این الگوریتم به دنبال یک ابرصفحه جدا کننده بهینه بین دو کلاس است به‌طوری که فضای بین نزدیک‌ترین نقاط از کلاس‌های متفاوت به حداکثر برسد و منطبق بر آن نقاط که بردارهای پشتیبان نامیده می‌شوند مرزهایی رسم شود[10]. سپس به وسیله این مرزها مشخص می‌کند که هریک از نمونه‌ها به کدام کلاس تعلق دارند. یعنی الگوریتم به نوعی طبقه‌بند خطی باینری تبدیل می‌شود. نمونه‌های آموزشی در این الگوریتم به نقاطی در فضا تبدیل می‌شوند به‌طوری که فضای بین دو دسته به حداکثر برسد. سپس نمونه‌های جدید نیز در همان فضا ترسیم می‌شوند و الگوریتم تصمیم می‌گیرد که این نمونه‌ها متعلق به کدام سمت این فاصله هستند. اگر مرزهای الگوریتم از نوع سخت تعریف شده باشند در فضای بین دو گروه، نباید هیچ نمونه‌ای قرار بگیرد. اما اگر مرزها از نوع نرم تعریف شده باشند، تعداد کمی از نمونه‌ها می‌توانند در این فضا قرار بگیرند.

۲-۶. K همسایه نزدیک

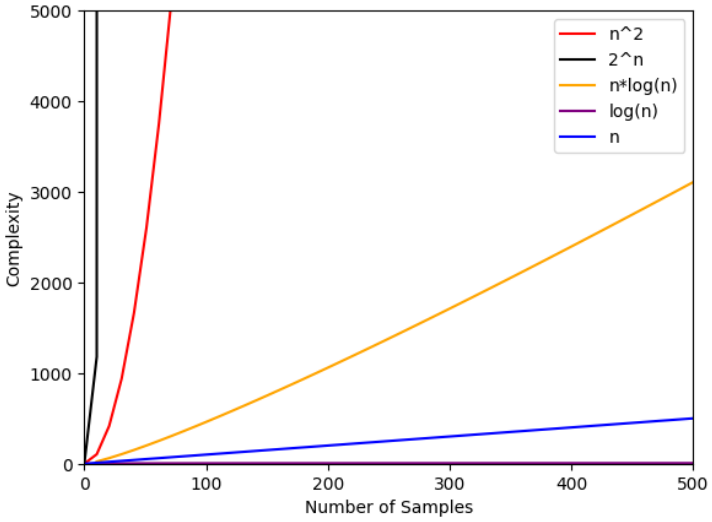
الگوریتم K همسایه نزدیک در ساده‌ترین تعریف، با توجه به نمونه جدید، به دنبال K نمونه که در نزدیک‌ترین حالت به آن قرار دارند می‌گردد. سپس کلاس‌ها یا برچسب‌های آن‌ها را بررسی می‌کند و هر برچسبی که تعداد بیشتری از K نمونه نزدیک دارای آن باشند را به عنوان برچسب و کلاس داده جدید انتخاب می‌کند. اگر مسئله از نوع رگرسیون باشد، میانگین مقادیر داده‌های مشخص شده برای داده جدید پیش‌بینی می‌شود. در فرایند طبقه‌بندی ابتدا تعداد همسایه‌ها تعیین می‌شود و سپس پیش‌بینی در مورد نمونه جدید را می‌توان با توجه به توزیع کلاس‌ها در بین این همسایه‌ها انجام داد[11].

فرض اصلی این الگوریتم نزدیک بودن نمونه‌های مشابه به یکدیگر است. در مجموع الگوریتم K همسایه نزدیک برای داده‌های جدید از تخمین زدن جهت طبقه‌بندی و یا تعیین مقادیر در مسائل رگرسیون استفاده می‌کند. اگرچه می‌توان از این الگوریتم هم در مسائل طبقه‌بندی و هم در مسائل رگرسیون استفاده کرد، به طور معمول به عنوان یک الگوریتم طبقه‌بندی از آن یاد می‌شود.

۳. تحلیل پیچیدگی محاسباتی

منابع محاسباتی در تمام کاربردهای عملی مرتبط با الگوریتم‌ها بسیار مهم‌اند. پیچیدگی محاسباتی یک الگوریتم، اندازه‌گیری منابع مورد نیاز آن الگوریتم برای اجرا است. قابل ذکر است که پیچیدگی محاسباتی الگوریتم با پیچیدگی ذاتی مسائل نمونه متفاوت است. نظریه پیچیدگی محاسباتی همواره در حوزه تحقیقات الگوریتم مورد استفاده و دارای اهمیت است؛ چرا که این محاسبات موجب یافتن الگوریتم‌هایی با پیچیدگی زمانی کمتر می‌شود که با وجود فراابتکاری بودن، پیچیدگی محاسباتی آن‌ها از مرتبه چند جمله‌ای است[12].

پیچیدگی محاسباتی یک الگوریتم تعیین‌کننده عملکرد آن الگوریتم است. اگر پیچیدگی محاسباتی یک الگوریتم از مرتبه نمایی باشد، می‌توان گفت که آن الگوریتم عملکرد بسیار بدی دارد و با افزایش داده‌ها عملاً مثل این است که هیچ الگوریتمی برای حل مسئله ارائه نشده است. شکل ۱ مقایسه‌ای از تغییرات پیچیدگی محاسباتی نسبت به افزایش داده‌ها برای مراتب مختلف است که نشان می‌دهد برای مجموعه داده بزرگ پیچیدگی محاسباتی نمایی عملکرد بسیار بدی دارد درحالی که پیچیدگی محاسباتی از مرتبه چندجمله‌ای و یا لگاریتمی بهتر عمل می‌کند.



شکل ۱ – تغییرات پیچیدگی محاسباتی الگوریتم‌ها با افزایش نمونه‌ها

ارزیابی پیچیدگی محاسباتی یک الگوریتم یادگیری ماشین بسیار پیچیده و اغلب اوقات به الگوریتم‌های دیگر متکی است. به طور معمول در مبحث الگوریتم‌ها پس از طراحی یک الگوریتم، تحلیل از زمان لازم برای اجرای آن در حالت مجانبی صورت می‌گیرد. این مطلب در خصوص الگوریتم‌های یادگیری ماشین نیز از اهمیت ویژه‌ای برخوردار است؛ لذا در ادامه تحلیلی از مرتبه پیچیدگی روش‌های یادگیری ماشین مورد نظر ارائه می‌دهیم. قابل ذکر است که منظور از پیچیدگی محاسباتی در این مقاله، مرتبه زمانی الگوریتم است.

۳-۱. درخت تصمیم

می‌توان گفت درخت تصمیم یک مدل محاسباتی است که از دنباله‌ای از پرسمان‌ها تشکیل شده است. برای مثال سؤالات بله و خیر متوالی یا دسته‌بندی داده‌ها در کلاس‌های متفاوت از این قبیل پرسمان‌ها است. بنابراین در این الگوریتم نتیجه هر مرحله از آزمون‌ها و پرسمان‌ها می‌توانند بر سطح بعدی آزمون‌ها تاثیر بگذارند. در نتیجه پیچیدگی زمانی یک الگوریتم درخت تصمیم با عمق آن، مطابقت دارد.

درخت تصمیم در هر گره غیر برگ، با توجه به ویژگی ذکر شده در آن گره، داده‌ها را به بخش‌های متفاوت تقسیم می‌کند. این کار تا زمانی که سطوح بعدی وجود داشته باشد یا به بیانی عمق کاملا طی نشده باشد، ادامه پیدا می‌کند. در بهترین حالت، اگر درخت تصمیم متعادل شده باشد، عمق آن در $O(log\ n)$ خواهد بود. اما درخت‌های تصمیم بدون توجه به متعادل کردن، تقسیم‌بندی را انجام می‌دهند. در نتیجه عمق در بدترین حالت $O(n)$ خواهد بود. همچنین یک مرتب‌سازی مقایسه‌ای نیز در زمان $O(log\ n)$ قابل انجام است. در هر مرحله تقسیم داده‌ها به 1 و $n\text{-}1$ نمونه تقسیم می‌شود. که در آن n تعداد نمونه‌ها در گره فعلی است. در نتیجه پیچیدگی زمانی الگوریتم درخت تصمیم در مرحله آموزش از مرتبه $O(np\ log\ n)$ است که در آن p تعداد ویژگی‌ها و نشان‌دهنده عمق است.

از آنجایی که درخت تصمیم برای نمونه جدید در مرحله آزمون، به تعداد ویژگی‌ها احتمال تعلق نمونه به کلاس را می‌سنجد و به همین میزان عمق درخت را طی می‌کند، پیچیدگی محاسباتی الگوریتم درخت تصمیم در مرحله آزمون از مرتبه $O(p)$ است.

۳-۲. جنگل تصادفی

اولین و بدیهی‌ترین جنبه پیچیدگی محاسباتی در درخت‌های تصمیم‌گیری و جنگل‌های تصادفی، تعداد عملیات مورد نیاز برای ساخت مدل از داده یا همان پیچیدگی زمانی برای مدل‌های یادگیری است[13]. در بخش ۲-۲ دیدیم که یک جنگل تصادفی از چندین درخت تصمیم تشکیل شده است. همچنین در بخش ۳-۱ نشان دادیم که پیچیدگی زمانی الگوریتم درخت تصمیم در مرحله آموزش و در بدترین حالت از مرتبه $O(np\ \log n)$ است. بدیهی است که اگر m تعداد درخت‌های تصمیم موجود در یک جنگل تصادفی باشد، پیچیدگی محاسباتی این الگوریتم در مرحله آموزش از مرتبه $O(mnp\ log\ n)$ خواهد بود.

پیچیدگی محاسباتی الگوریتم جنگل تصادفی برای مرحله آزمون نیز با توجه به توضیحات فوق، از مرتبه $O(mp)$ است. که m تعداد درخت‌های موجود در جنگل است.

۳-۳. رگرسیون خطی

مهم‌ترین و چالش‌برانگیزترین بخش در محاسبات پیچیدگی الگوریتم رگرسیون خطی برای مرحله آموزش ارزیابی معادله زیر است.

$$\beta = (X'X)^{-1}X'Y \qquad (2)$$

پیچیده‌ترین قسمت، محاسبه عبارت $X'X$ است. ماتریس X برای هر داده، نشان‌دهنده ویژگی‌های آن داده است. اگر p نشان‌دهنده تعداد ویژگی‌ها باشد، ابعاد این ماتریس $1\times p$ است. یعنی در حقیقت ماتریس X، یک بردار p بعدی است. بردار X' نیز ترانهاده بردار X است و ابعاد آن به صورت $1\times p$ است. در نتیجه عبارت $X'X$ یک ماتریس با ابعاد $p\times p$ است. اگر محاسبه این عبارت به تعداد داده‌ها یعنی n بار انجام شود، پیچیدگی محاسبه این عبارت در p^2n عملیات انجام می‌شود. همچنین پیچیدگی محاسبه وارون یک ماتریس با ابعاد ذکر شده از مرتبه $O(p^3)$ است. اگرچه در بیشتر پیاده‌سازی‌ها استفاده از گرادیان کاهشی ترجیح داده می‌شود، هزینه محاسبه تغییر چندانی ندارد. در نتیجه پیچیدگی محاسباتی الگوریتم رگرسیون خطی در مرحله آموزش با ویژگی‌های متعدد از مرتبه $O\ (p^2n+p^3)$ است.

الگوریتم رگرسیون خطی در بخش آزمون، فقط ماتریس Y که نشان‌دهنده پیش‌بینی مقادیر ویژگی‌ها است را محاسبه می‌کند. بدیهی است که ابعاد این ماتریس $1\times p$ است. در نتیجه پیچیدگی محاسباتی این الگوریتم برای مرحله آزمون $O(p)$ است.

۳-۴. بیز ساده

اگر n تعداد نمونه‌ها، p تعداد ویژگی‌ها و c تعداد کلاس‌های مسئله باشد، پیچیدگی محاسباتی الگوریتم بیز ساده در طول مرحله آموزش از مرتبه $O(npc)$ است. این الگوریتم احتمال تعلق هر نمونه از داده آزمون به تمامی کلاس‌ها را به ازای همه ویژگی‌ها محاسبه می‌کند. با کمک بهینه‌سازی می‌توان پیچیدگی محاسباتی فوق را به $O(np)$ کاهش داد. همچنین اگر تعداد ویژگی‌ها کم باشد، می‌توان این الگوریتم را از مرتبه زمانی $O(n)$خواند.

در مرحله آزمون الگوریتم بیز ساده به ازای هر ویژگی احتمال تعلق نمونه به کلاس‌ها را بررسی می‌کند. بدیهی است که پیچیدگی محاسباتی الگوریتم بیز ساده در مرحله آزمون از مرتبه $O(pc)$ است.

۳-۵. ماشین بردار پشتیبان

در بخش آموزش، الگوریتم ماشین بردار پشتیبان نیاز به ارزیابی ماتریسی دارد که در آن k هسته مشخص شده است. پیچیدگی محاسبه این ماتریس از مرتبه $O(p)$ است. این پیچیدگی برای هسته‌های رایج مانند گوسی و سیگموئیدی صادق است. همچنین اگر n تعداد نمونه‌ها باشد، محاسبه معکوس ماتریس، از مرتبه $O(n^3)$ است. در نتیجه پیچیدگی محاسباتی مرحله آموزش برای الگوریتم ماشین بردار پشتیبان از مرتبه $O(n^3p)$ است. که در آن n تعداد نمونه‌ها و p تعداد ویژگی‌ها است.

پیچیدگی محاسباتی مرحله آزمون برای الگوریتم ماشین بردار پشتیبان از مرتبه $O(vp)$ است که در آن v تعداد بردارهای پشتیبان به دست آمده در مرحله آموزش است. این الگوریتم برای داده آزمون و به ازای هر ویژگی بررسی می‌کند که این نمونه به کدام سمت از مرزها تعلق دارد.

۳-۶. K همسایه نزدیک

پیچیدگی محاسباتی آموزش داده‌ها در الگوریتم K همسایه نزدیک برابر با $O(knp)$ است که در این عبارت k تعداد همسایه‌ها، n تعداد داده‌ها و p تعداد ویژگی‌ها است. در روش کار این الگوریتم در هر دور آموزش، فاصله هر داده آموزش تا k داده همسایه را حساب می‌کند. این کار برای هر یک از داده‌ها به ازای هر ویژگی یعنی به تعداد np بار تکرار می‌شود. بدیهی است که پیچیدگی این الگوریتم در مرحله آموزش از مرتبه $O(knp)$ خواهد بود.

پیچیدگی محاسباتی الگوریتم K همسایه نزدیک برای داده‌های آزمون از مرتبه $O(kp)$ است. این الگوریتم برای نمونه آزمون مشاهده شده به تعداد ویژگی‌ها k همسایه نزدیک را می‌یابد و میانگین مقادیر آن‌ها یا پرتکرارترین کلاس نسبت داده شده را به دست می‌آورد.

۴. ارزیابی نتایج

در این قسمت نتایج به دست آمده در مرحله قبلی با یکدیگر مقاله مقایسه و مورد تجزیه و تحلیل قرار می‌گیرند. هر الگوریتم بسته به نوع عملکرد، یک یا چند پارامتر منحصر به فرد دارد. اما هر مسئله باید تعداد ویژگی‌هایی با تعداد مشخص و مجموعه آموزشی با اندازه معین داشته باشد؛ بنابراین می‌توان گفت مهم‌ترین معیار جهت انتخاب الگوریتمی با پیچیدگی محاسباتی مناسب، تعداد ویژگی‌ها و اندازه مجموعه داده است.

در بخش ۳ پیچیدگی محاسباتی الگوریتم‌های یادگیری ماشین برحسب تعداد ویژگی‌ها، تعداد نمونه‌های مجموعه داده و همچنین برخی پارامترهای منحصر به فرد هر الگوریتم، ذکر شد. در این بخش پیچیدگی محاسباتی الگوریتم‌ها بر اساس تعداد ویژگی‌ها و داده‌ها با یکدیگر مقایسه می‌شوند.

در جدول ۱ می‌بینیم که پیچیدگی محاسباتی همه الگوریتم‌ها در مراحل آموزش و آزمون به چه صورت است و هر یک بیشتر از چه مواردی تاثیر می‌پذیرد.

جدول ۱- پیچیدگی محاسباتی به دست آمده برای الگوریتم‌ها

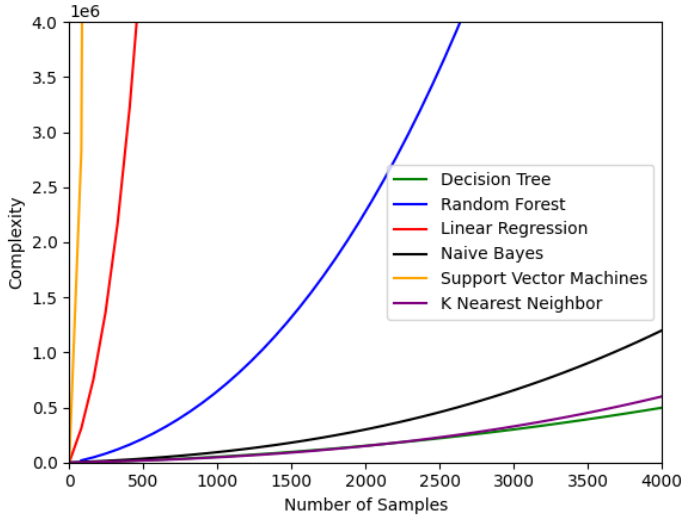
	مرحله آموزش	مرحله آزمون
درخت تصمیم	$O(np\ log\ n)$	$O(p)$
جنگل تصادفی	$O(mnp\ log\ n)$	$O(mp)$
رگرسیون خطی	$O\ (p^2n+p^3)$	$O(p)$
بیز ساده	$O(npc)$	$O(pc)$
ماشین بردار پشتیبان	$O(n^3p)$	$O(vp)$
K همسایه نزدیک	$O(knp)$	$O(kp)$

همان طور که در جدول ۱ مشخص شد، مؤثرترین عوامل بر پیچیدگی محاسباتی مرحله آموزش برای تمام الگوریتم‌ها تعداد نمونه‌ها بود. از این رو تغییرات پیچیدگی محاسباتی با افزایش تعداد نمونه‌ها برای مرحله آموزش در شکل ۲ نشان داده شده است.

برای مقایسه پیچیدگی محاسباتی الگوریتم‌های یادگیری ماشین در شکل ۲، میانگین تعداد ویژگی‌ها بین ۱۰ تا ۲۰ عدد در نظر گرفته شده است. طبق قانون۱۰ که یک قاعده سرانگشتی است، تعداد نمونه‌ها نباید کمتر از ۱۰ برابر تعداد ویژگی‌ها باشد[14]. جهت نمایش بهتر وضعیت پیچیدگی محاسباتی با افزایش نمونه‌ها، تعداد آن‌ها تا ۴۰۰۰ عدد انتخاب شده است.

نشان داده شده است که افزایش تعداد درختان یک جنگل تصادفی، بیش از یک حد آستانه مشخص در حدود ۱۲۸ درخت، عملکرد قابل توجهی را به همراه ندارد و فقط هزینه محاسباتی را افزایش می‌دهد[15]. لیکن در مقاله حاضر تعداد درخت‌های موجود در جنگل تصادفی جهت نمایش بهتر پیچیدگی‌ها بین ۱۰ تا۲۰ عدد در نظر گرفته شده است.

تعداد کلاس‌ها در حدود ۱۰ الی ۲۰ عدد در نظر گرفته شده است. همچنین تعداد بردارها ۲ الی ۶ و تعداد همسایه‌ها بین ۵ تا ۱۰ عدد انتخاب شده است.

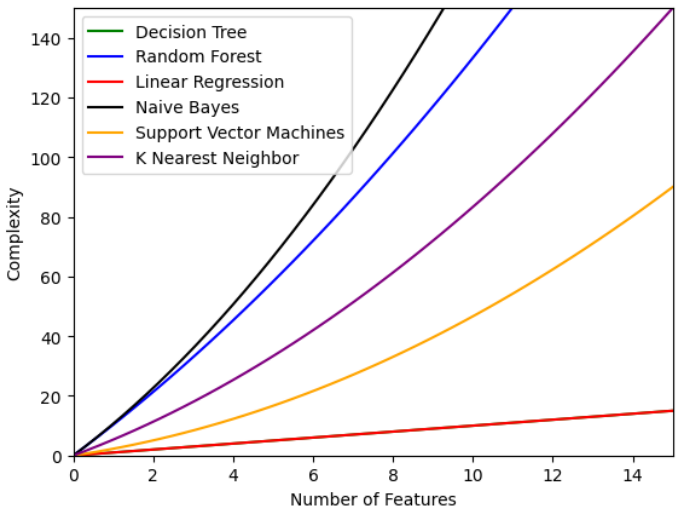


شکل ۲ – تغییرات پیچیدگی محاسباتی مرحله آموزش برای الگوریتم‌ها با افزایش تعداد داده‌ها

همان‌طور که انتظار می‌رفت، پیچیدگی محاسباتی الگوریتم‌های رگرسیون خطی و ماشین بردار پشتیبان با افزایش تعداد داده‌ها به شکل چشمگیری افزایش یافته است.

شایان ذکر است که عوامل دیگر نیز در افزایش پیچیدگی محاسباتی الگوریتم‌ها دخیل بوده‌اند. برای مثال تغییرات سریع پیچیدگی محاسباتی الگوریتم رگرسیون خطی که از مرتبه $O\ (p^2n+p^3)$ است، بیش از این که متأثر از افزایش تعداد داده‌ها باشد، متأثر از افزایش تعداد ویژگی‌ها است. همچنین افزایش پیچیدگی محاسباتی جنگل تصادفی نسبت به درخت تصمیم به دلیل تعداد درخت‌ها اتفاق افتاده است. در مورد الگوریتم بیز ساده نیز تعداد کلاس‌هایی که باید احتمال تعلق نمونه به آن‌ها محاسبه شود باعث افزایش پیچیدگی محاسباتی است.

بر اساس اطلاعات مندرج در جدول ۱، موثرترین عامل بر پیچیدگی محاسباتی مرحله آزمون برای تمام الگوریتم‌ها تعداد ویژگی‌ها است. از این رو تغییرات پیچیدگی محاسباتی با افزایش تعداد ویژگی‌ها برای مرحله آزمون در شکل ۳ نشان داده شده است. قابل ذکر است که ضرایب m، c، v و k جهت نمایش بهتر پیچیدگی‌ها در بازه‌های مختلفی انتخاب شده‌اند.



شکل ۳ – تغییرات پیچیدگی محاسباتی مرحله آزمون برای الگوریتم‌ها با افزایش تعداد ویژگی‌ها

همان‌طور که در شکل ۳ مشخص است نمودار تغییرات الگوریتم‌های رگرسیون خطی و درخت تصمیم که در مرحله آزمون پیچیدگی محاسباتی از مرتبه $O(p)$ دارند بر یکدیگر منطبق شده است. باقی الگوریتم‌ها نیز بر اساس تعداد ویژگی‌ها به نوعی از همین مرحله به حساب می‌آیند و تنها تفاوت هرکدام ضریبی است که بر اساس نوع عملکرد الگوریتم در مرحله آزمون بر پیچیدگی زمانی تاثیر گذاشته است.

به طور کلی شکل ۳ نشان می‌دهد که برای انتخاب الگوریتمی با پیچیدگی محاسباتی مناسب در مرحله آزمون، باید بسته به نوع مسئله انتخاب صورت گیرد. تأثیر تعداد ویژگی‌ها در همه الگوریتم‌ها در این مرحله یکسان است و باید دید که مسئله مورد نظر در کدام نوع از الگوریتم‌های یادگیری ماشین عملکرد بهتری را نشان خواهد داد. این بررسی بر اساس تعداد کلاس‌ها، تعداد بردارهای پشتیبان، تعداد همسایه‌های نزدیک و تعداد درخت‌های موجود در جنگل تصادفی انجام می‌شود.

البته تفاوت چشمگیر پیچیدگی محاسباتی الگوریتم‌ها در مرحله آموزش باعث می‌شود که انتخاب الگوریتم بیشتر بر این اساس صورت گیرد.

۵. نتیجه‌گیری

حوزه یادگیری ماشین دارای الگوریتم‌هایی است که می‌توانند بدون برنامه‌نویسی کاملاً مشخص از طریق قدرت یادگیری، نظم خاص موجود در داده‌ها را به کمک مدل‌ها بیابند و مسائل پیچیده و دشوار را حل کنند. این الگوریتم‌ها انواع مختلفی دارند که می‌توان به عنوان شناخته‌شده‌ترین آن‌ها از الگوریتم‌های درخت تصمیم، جنگل تصادفی، رگرسیون خطی، بیز ساده، ماشین بردار پشتیبان و K همسایه نزدیک نام برد. عملکرد الگوریتم‌های ذکر شده در این مقاله به طور مختصر مورد بررسی قرار گرفت و پیچیدگی محاسباتی آن‌ها در مراحل آموزش و آزمون به دست آمد.

نتایج حاصل نشان دادند که پیچیدگی محاسباتی تمام الگوریتم‌ها در مرحله آموزش با افزایش تعداد نمونه‌ها افزایش می‌یابد. علاوه بر آن عوامل مؤثر بر افزایش پیچیدگی محاسباتی الگوریتم درخت تصمیم افزایش تعداد نمونه‌ها و تعداد ویژگی‌ها است. در الگوریتم جنگل تصادفی علاوه بر موارد تأثیرگذار بر پیچیدگی محاسباتی درخت تصمیم، تعداد درخت‌های موجود در جنگل نیز باعث افزایش پیچیدگی محاسباتی می‌شود. در الگوریتم رگرسیون خطی اگرچه افزایش تعداد نمونه‌ها باعث افزایش پیچیدگی محاسباتی می‌شود، اما افزایش تعداد ویژگی‌ها آن را به طرز چشمگیری افزایش می‌دهد. در الگوریتم بیز ساده، افزایش تعداد ویژگی‌ها، تعداد نمونه‌ها و تعداد کلاس‌ها به یک میزان بر افزایش پیچیدگی محاسباتی تأثیر دارند. پیچیدگی محاسباتی الگوریتم ماشین بردار پشتیبان با افزایش تعداد نمونه‌ها نسبت به دیگر الگوریتم‌ها افزایش سریع‌تری دارد. پیچیدگی محاسباتی الگوریتم K همسایه نزدیک با افزایش تعداد نمونه‌ها، تعداد ویژگی‌ها و تعداد همسایه‌ها به یک میزان افزایش می‌یابد.

افزایش پیچیدگی محاسباتی به دست آمده برای تمام الگوریتم‌ها در مرحله آزمون با افزایش تعداد ویژگی‌ها اتفاق می‌افتد. علاوه بر این عوامل مؤثر دیگری نیز وجود دارند که به برای الگوریتم‌های جنگل تصادفی، بیز ساده، ماشین بردار پشتیبان و K همسایه نزدیک به ترتیب تعداد درخت‌های موجود در جنگل، تعداد کلاس‌ها، تعداد بردارها و تعداد همسایه‌ها است.

به عنوان جمع‌بندی می‌توان پیشنهاد داد که برای مسائلی با تعداد نمونه‌های بسیار زیاد استفاده از الگوریتم ماشین بردار پشتیبان باعث پیچیدگی محاسباتی بسیار بالایی می‌شود. برای تعداد ویژگی‌های بسیار زیاد، استفاده از رگرسیون خطی توصیه نمی‌شود اگرچه ممکن است در مسائل پیش‌بینی با تعداد ویژگی‌های کمتر، از دیگر الگوریتم‌ها عملکرد بهتری داشته باشد. استفاده از الگوریتم بیز ساده برای مسائل طبقه‌بندی با تعداد کلاس‌های کم پیچیدگی محاسباتی کمتری را به همراه خواهد داشت. استفاده از الگوریتم K همسایه برای مسائل طبقه‌بندی، در صورتی که این الگوریتم با تعداد همسایه‌های کم به دقت مناسبی دست یافت، توصیه می‌شود. استفاده از الگوریتم درخت تصمیم در بسیاری از مسائل طبقه‌بندی پیشنهاد می‌شود چرا که پیچیدگی محاسباتی این الگوریتم حتی با افزایش تعداد نمونه‌ها نیز افزایش سریعی ندارد. از این رو توصیه می‌شود اگر تعداد درخت‌های موجود در جنگل تصادفی کم بودند از این الگوریتم نیز استفاده شود.

پیچیدگی محاسباتی الگوریتم‌ها نشان‌دهنده تعداد دستورالعمل و زمان مورد نیاز آن‌ها برای اجرا است. توجه به این امر و انتخاب الگوریتم‌ها با در نظر داشتن پیچیدگی محاسباتی آن‌ها می‌تواند عملکرد این الگوریتم‌ها را در حل مسائل مورد نظر تا حد قابل توجهی بهبود دهد.

۱۲. مراجع

[1] K. P. Murphy, “Probabilistic Machine Learning: Advanced topics.” MIT Press, 2023.

[2] D. Sharma and N. Kumar, "A review on machine learning algorithms, tasks and applications," *International Journal of Advanced Research in Computer Engineering & Technology* (*IJARCET*), vol. 6, no. 10, pp. 2278-1323, 2017.

[3] A. Majeedd, " Improving time complexity and accuracy of the machine learning algorithms through selection of highly weighted top k features from complex datasets," *Annals of Data Science*, vol. 6, no. 4, pp. 599-621, 2019.

[4] G. Bonaccorso, " Machine Learning Algorithms : reference guide for popular algorithms for data science and machine learning." Packt Publishing, 2017.

[5] B. Mahesh, " Machine learning algorithms-a review," *International Journal of Science and Research (IJSR)*, vol. 9, no. 1, pp. 381-386, 2020.

[6] H. M. Sani, C. Lei, and D. Neagu, “Computational complexity analysis of decision tree algorithms,” in *Lecture Notes in Computer Science*, Cham: Springer International Publishing, 2018, pp. 191–197.

[7] L. Breiman, " Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.

[8] D. Maulud and A. M. Abdulazeez, “A review on linear regression comprehensive in machine learning,” *Journal of Applied Science and Technology Trends*, vol. 1, no. 4, pp. 140–147, 2020.

[9] I. Wickramasinghe and H. Kalutarage, “Naive Bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation,” *Soft Computing*, vol. 25, no. 3, pp. 2277–2293, 2021.

[10] D. Meyer and F. T. Wien, “Support Vector Machines,” *R News*, vol. 1, no. 3, pp. 23–26, 2001.

[11] B. Li, S. Yu, and Q. Lu, “An improved k-Nearest Neighbor algorithm for text categorization,” *arXiv [cs.CL]*, 2003.

[12] M. Roos and J. Rothe, “Introduction to Computational Complexity,” *Technical report, Institut fur Informatik, Dusseldorf, Germany*, 2010.

[13] G. Louppe, “Understanding Random Forests: From Theory to Practice,” *arXiv.org*, 2014.

[14] J. C. Westland, " Lower bounds on sample size in structural equation modeling," *Electronic commerce research and applications*, no. 6, pp. 476-487, 2010.

[15] T. M. Oshiro and P. S. Perez, " How many trees in a random forest?," in *Machine Learning and Data Mining in Pattern Recognition: 8th International Conference, MLDM, Berlin, Germany,* 2012, pp. 154-168.