گزارش تمرین یادگیری ماشین سری سوم

محمد زیاری

سوال 1

کرنل ها به ما در تفکیک کردن داده ها بسیار کمک میکنند. در اکثر مواقع ، داده هایی که داریم، به صورت خطی از هم جدا نمی شوند و برای جدا کردن آنها نیاز است که تابع های غیر خطی داشته باشیم. در این شرایط بعد فضای ویژگی ها را توسط کرنل های غیر خطی افزایش میدهیم تا تفکیک پذیری داده ها خیلی بهتر انجام شود.

این کرنل ها معمولا در همه موارد استفاده میشوند و شرایط خاصی ندارند، اما برخی از آنها در موارد به خصوصی بهتر عمل میکنند. برخی از این کرنل ها و نحوه عملکرد آنها را باهم میبینیم.

ساده ترین نوع کرنل ها ، linear kernels هستند. این کرنل ها در مواقعی به کار میروند که داده های ما به صورت خطی جدایی پذیر هستند و به صورت تابع خطی هستند.

کرنل های چند جمله‌ای، polynomial kernels به طور خیلی خاص در پردازش تصویر ها بسیار کاربرد دارند.

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d$$

کرنل های گاوسی، gaussian kernels از توزیع گاوسی تبعیت میکنند و در اکثر موارد مورد استفاده قرار میگیرند.

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$

Gaussian kernel radial basis function یا RBF بسیار مشابه کرنل های گاوسی است با این تفاوت که یک شعاع خاصی را در نظر میگیرد و بر اساس آن، قسمت بندی میکند.

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2) \text{ و } \gamma > 0 \text{ و } \gamma = 1/2\sigma^2$$

کرنل سیگموید ، sigmoid kernels معمولا در شبکه های عصبی به کار میرود و به عنوان تابع activation از آن استفاده میکنیم. $k(x, y) = \tanh(\alpha x^T y + c)$

آخرین کرنلی که باهم میبینیم، Linear splines kernel in one-dimension هست. این کرنل بیشتر مواقعی به کار میرود که با داده هایی با پراکندگی زیاد مواجه هستیم. برای جدا کردن متن ها بیشتر مورد استفاده قرار میگیرد و در رگرسیون نیز خوب عمل میکند.

$$k(x, y) = 1 + xy + xy\ min(x, y) - \frac{x+y}{2}\ min(x, y)^2 + \frac{1}{3}\min(x, y)^3$$

سوال 2

اول از آنکه راجب سوال توضیح دهم ، دیتاست این سوال مقدار نال یا کتگوریکال نداشت و پاکسازی دیتای خاصی انجام ندادم و پس از جدا کردن تارگت و اسکیل کردن و جدا کردن داده های تست و آموزشی به سراغ انجام مدل ها رفتیم.
این سوال مربوط به پیاده سازی روش svm است که با پکیج sklearn پیاده سازی شده است. در این سوال پس از بدست آوردن دقت روی داده های تست ، confusion matrix را نمایش داده ایم و به سوال بعدی می رویم تا پارامتر های مختلف svm را آزمایش کنیم.

سوال 3

در این سوال ابتدا بر اساس نوع کرنل ها بررسی انجام داده ایم. دیفالت بر روی rbf است و ابتدا با تغییر بر Linear میبینیم مدل ما دقت بالاتری داشت و به 97 درصد رسید.
برای بررسی های بعدی Poly بررسی شده است. poly متدی به نام degree دارد که در واقع درجه آن را میتوان تعیین کرد که به صورت دیفالت 3 تعریف میشود که در آن حالت دقت ما کاهش پیدا کرده و به 78 درصد رسیده است. با تغییر درجه به 4 میبینم دقت اینبار به میزان قابل توجهی

بیشتر کاهش داشته و به 39 رسیده است. به نظر میرسد داده ها با درجه چهارم دچار overfitting میشوند و مدل بیش از حد تمرکز را بر روی داده آموزشی معطوف کرده است.
در حالت بعدی با تغییر درجه به 1 می بینیم تا میزان خوبی درصد ما بهتر شده است و تا حدی حدس overfitting ما تایید میشود و دقت به 96 درصد رسید.
اما در بخش بعد نوع کرنل به sigmoid تغییر داده شده که دقت آن 92 درصد بود. پارامتری که در آن بررسی کردیم ، coeff0 بود که ترمی در تابع کرنل ما است و میزان دیفالت آن 0 بود که با تغییر آن ، خیلی مدل خوبی پیدا نشد اما با تغییرات کم حول 0 و با مقدار 0.15 توانستیم دقت 93.2 درصدی پیدا کنیم که روی داده های تست بهتر عمل کرد.
در آخر نتیجه ای که من از این تغییرات پیدا کردم آن بود که بهترین مدل مان در حالت linear و poly با درجه یک بود پس مدل خطی بهتر روی دیتاست کار میکند و بیش از آن باعث overfitting میشود.

سوال 5

الف) در این بخش برای binning من از میانه ها استفاده کردم. برای سه حالت اعداد 5 و 4 و 6 را برای تعداد bin ها در نظر گرفتم. پس از آنکه بازه ها مشخص شد میانه هر بازه را به جای عدد درون آن بازه در ستون مجزایی به دیتاست اضافه کردم.
در اولین بخش با بازه هایی به طول تقریبا مساوی دقتمان کمی از svm عادی بیشتر شد اما با کم کردن تعداد bin ها به 4 در حالت بعدی دقت ما کاهش یافت.
در بخش آخر این سوال خواستم با تعداد bin شش مقدار دقت را بالاتر ببریم. برای نامساوی کردن طول بازه ها هم ابتدا بازه های ابتدایی و انتهایی را بزرگتر کردم با این استدلال که از داده های پرت دورتر شویم اما دقت 87 درصدی داشتیم اما با کوچک کردن بازه های ابتدایی و انتهایی دقت ما تقریبا به 90 درصد رسید که پیشرفتی نسبت به svm بدون استفاده از کرنل محسوب میشود.

ب) در این دیتاست ما داده ی categorical نداشتیم که روی آن one hot coding انجام دهیم اما در سری های پیشین و به طور کل در قبل با نحوه کار one hot coding آشنایی داریم که این روش برای عددی کردن داده هاست و به نظرم برتری آن نسبت به روش label encoding آن است که با جدا سازی همه متغیر های یک ستون به صورت ستون مجزا باعث میشود اگر مقداری در ستون مهم بود ، مدل ما راحت تر آن را کشف کند و در واقع در لیبل انکدینگ مشکلی که وجود دارد

آن است که انگار رابطه ای برای عناصر جدول در نظر گرفته میشود (مثلا به اولی یک میدهیم به بعدی 2 به بعدیش 3 در حالی که این مقادیر هیچ ارتباطی به هم ندارند مقدار نزدیکی به یکدیگر دارند.)
البته در ستون هایی که تعداد متغیر ها یا عناصر زیاد باشد استفاده از one hot منطقی به نظر نمی رسد و اضافه کردن بیشمار ستون به دیتاست در مواقعی شاید کار درستی نباشد.

ج) از تبدیل log transform استفاده کردیم. بر روی فیچرهای دیتاست این تبدیل را اعمال میکنیم. ممکن است اعداد منفی هم داشته باشیم و چون می خواهیم با لگاریتم گرفتن مشکلی پیش نیاید، رنج داده ها را تغییر میدهیم. ابتدا آنها را از مینیمم شان کم می کنیم و سپس با 1 جمع میکنیم. حال پس از Log گرفتن تمامی مقادیر از 0 بیشتر خواهند بود.
Log transform بیشتر مواقعی به کار میرود که داده ها گوناگونی زیادی دارند و از توزیع نرمال دور هستند و توزیع آنها چولگی داشته باشد. پس از انجام این تبدیل داده ها به توزیع نرمال نزدیک میشوند .همچنین میزان تاثیر داده های پرت را نیز با اینکار کم میکند و مدل را بهتر میکند.
با توجه به نتیجه SVM بر روی این تبدیل، مشاهده میکنیم که میزان دقت آن پایین آمده و در سوال ششم هم از این روش بهره نبردیم و فقط الف و د را اعمال کردیم.

د) در این سوال به راحتی با ضرب width در height به مساحت می رسیم و به جای این دو ستون مساحت را در جدول قرار میدهیم. حال با دیتاست جدید svm را ران کردیم که دقتی نزدیک به 88 درصد داشتیم که کمی بهتر از قبل بود.

سوال 6

برای این سوال ابتدا آنکه نتایج svm هر مرحله بالا را در هر مورد توضیح داده ایم. اما برای حالتی که همه آنها با هم صورت بگیرد که حالت 1 و 4 است (دلیل عدم استفاده از 3 ذکر شده است) که نتیجه بسزایی نداشت و 87 درصد بود و در واقع تغییر خاصی در آن صورت نگرفت و هریک از مراحل بالا به تنهایی دقت بالاتری داشتند

سوال 7

الگوریتم های مختلفی برای ساخت درخت وجود دارد، این الگوریتم ها از معیار های متفاوتی استفاده میکنند، در برخی از آنها زا عمل هرس کردن استفاده می شود اما در برخی خیر. همچنین روش های تقسیم بندی کردن درخت در آنها متفاوت است.

الگوریتم اول ID3 است یا نام کامل آن Iterative Dichotomiser 3 که این الگوریتم از information gain برای ساخت درخت استفاده میکند. این معیار تعیین میکند که کدام ویژگی ها اطلاعات بیشتری دارند و برای ساخت درخت لازم هستند. آنهایی که اطلاعات بیشتری دارند در راس درخت قرار می گیرند و بدین ترتیب زیر درخت های دیگر نیز ساخته میشوند. این الگوریتم داده های گسسته را نمیتواند جدا کند.

ورژن پیشرفته تر از الگوریتم ID3 الگوریتم C4.5 است که علاوه بر معیار information gain از gain ratio نیز استفاده میکند. مانند ID3 عمل میکند اما بر روی داده های گسسته نیز میتواند تفکیک را انجام دهد و همچنین این الگوریتم بر خلاف قبلی، بر روی درخت عمل هرس کردن را نیز انجام میدهد

الگوریتم دیگری که ساخت اکثر درخت ها به این روش انجام میشود CART است که مخفف Classification and Regression Tree است. این روش علاوه بر ساخت درخت classification میتواند درخت هایی بر مبنای regression نیز بسازد.برای آنکه بتواند این کار را انجام دهد از معیار جینی (gini index) استفاده میکند. هرچه این معیار کمتر باشد یعنی آن ویژگی اطلاعاتی بهتری در اختیار ما قرار میدهد، پس مهم تر است.

الگوریتم دیگر به نام CHAID وجود دارد. برای ساخت درخت، داده ها را متناوبا به زیر مجموعه های مشابه تقسیم میکند تا آنجا که هر زیرمجموعه دارای تعداد مشخصی نمونه شود.ممکن است که درختی تولید کند که در برخی موارد به صورت غیر باینری عمل کند و به چند بخش تقسیم میکند.این الگوریتم از آزمون Chi squared برای تصمیم گیری در هر تقسیم برای مشخص کردن زیر درخت ها استفاده می کند هرس کردن درخت نیز از طریق یافت تفاوت های مشابه انجام میشود.

سوال 8

با استفاده از پکیج sklearn ، این سوال را هم پاسخ می دهیم و درخت تصمیمان را پیاده سازی میکنیم . دقت بدست آمده بیش از 84 درصد بود و نیازی به توضیح اضافه نیست تا در بخش بعدی پارامترها را تغییر دهیم.

سوال 9

قاعدتا تغییر دادن پارامترها تاثیر دارد اما به چه میزان و حد؟ در واقع چه پارامترهایی را از حالت دیفالت عوض کنیم تغییر منفی و کدامشان تغییر مثبت دارند.
پس از سوال 8 ، حالا max_depth و internal node را تغییر میدهیم تا گره و عمق درخت را تعیین کنیم. با تغییر دادن max_depth به عدد پایین به شکل واضح accuracy پایین میاید و تنها وقتی به حالت نرمال برمیگردد که ما مقدار 15 را برای آن در نظر بگیریم که فکر میکنم دلیلش آن است که درخت ما در حالت اولیه عمق 15 تایی یا شبیه به آن داشته و با تغییر max_depth به آن درختمان را محدود نکرده ایم. حالت های با عمق 5 و 15 در کد موجود است.

اما در ارتباط با گره ها من 3 بخش را تغییر دادم و نتایج را گزارش می دهم. در min_samples_leaf دیفالت که 1 بود اما بهترین دقت در وقتی به دست آمد که برابر با 7 باشد و اگر آن را بیش از 7 میکردیم دقت پایین می آمد.مدل در کد با میزان 7 موجود است.
اما متد min_samples_spilit چیزمفهمومی از تغییراتش نیافتم و با تغییر همیشه نویز داشت ولی بهترین حالت در همان دیفالت نودها است که 2 است.
برای متد آخر که max_leaf_nodes است ، قطعا با محدود کردن نودهای برگ نمیتوانیم به جواب خوبی برسیم (البته شاید در مسائل خاص برسیم) که همین طور هم شد و وقتی این مقدار را به میزان کمی همگرا می کنیم ، مثلا 4 دقت به طور شدیدی پایین می آید و فکر میکنم استفاده از میزان دیفالت که none است بهتر باشد.

سوال 10

عمل هرس کردن به معنای حذف کردن برخی از زیر شاخه های درخت است به گونه‌ای که شاخه هایی که باعث ایجاد داده های پرت و نویز می شوند از بین برود. در واقع با این کار،

از overfitting جلوگیری میشود. هرس کردن باعث میشود تا درخت ما کوچکتر شود و پیچیدگی در آن کمتر شود، بدین ترتیب داده هایی که باعث ناهنجاری میشوند از بین می روند و عملکرد مدل ما بهتر خواهد بود.

دو رویکرد رایج برای هرس درخت به شرح ذیل وجود دارد:

پیش هرس (Pre pruning): در این رویکرد یک درخت به وسیله توقف های مکرر در مراحل اولیه ساخت درخت، هرس میشود.

هرس پسین (Post pruning): به این صورت است که زیر درخت ها از یک درخت رشد یافته کامل را حذف می‌کند. یک زیر درخت در یک گره به وسیله حذف کردن شاخه‌ها و جایگزینی آن‌ها با یک برگ، هرس می‌شود.

سوال 12

با استفاده از پکیج مدلی از random forrest ساختیم و بدون تغییر هیچ پارامتری دقتی بالاتر از درخت تصمیم داشتیم. من به طور کل پارامتری در این سوال را تغییر نمیدهم زیرا جواب مشخصا آن است که random forrest بهتر است. دلیلش هم آن است که random forest به جای آنکه از یک درخت استفاده کند، از چندین درخت بهره میگیرد و نتیجه نهایی آن از تجمیع این درخت ها پدید می‌آید. همچنین میتوان تعداد درخت های موجود را تعیین کرد و می توان تعداد ماکزیمم ویژگی هایی که در هر درخت باشد را انتخاب کرد. هرچه تعداد این درخت ها بیشتر باشد، میزان دقت در مدل افزایش خواهد یافت. اما پس از یه تعدادی که افزایش می دهیم دیگر میزان دقت افزایش نمی یابد و ثابت میماند.

سوال 13

یکی از دلایلی که باعث محبوب بودن الگوریتم درخت تصمیم شده است، تفسیر پذیری آن است. در این الگوریتم به راحتی میتوانیم نحوه دسته بندی داده ها و عملکرد تقسیم شدن آن ها را مشاهده کنیم و تفسیر آنها بسیار ساده است اما در شبکه ها عصبی و شبکه های عمیق اینگونه نیست و عملکرد آنها پیچیده تر است. همچنین درخت تصمیم الگوریتم سریع تری

نسبت به شبکه عصبی است و روی داده هایی با حجم کمتر، بهتر عمل میکند اما اگر داده ها زیاد شوند، میزان عملکرد درخت تصمیم پایین می‌آید و مدل به خوبی نمی تواند تصمیم گیری کند، مگر از روش های baggin و یا boosting استفاده کنیم.

سوال 14

روش RIPPER یکی از معروف ترین روش هایی است که در استخراج قوانین استفاده می شود، این روش از روش های مبتنی بر الگوریتم های grow and prune است که با استفاده از رویکرد تقسیم و غلبه عمل استخراج قوانین را انجام میدهد. این روش IREP را اجرا میکند تا با استفاده از آن، یک قانون اولیه برای کلاس مورد نظر ایجاد کند. سپس در یک مرحله بهینه سازی اضافه، هر قانون را در نظر میگیرد و با توجه به این قانون اولیه، به ترتیب دو قانون دیگر از آنها ایجاد می کند : قانون جایگزینی(replacement rule) و قانون تجدید نظر (revision rule) و سپس مدل بر اساس معیار طول حداقل، تصمیم میگیرد که بین قانون اصلی، قانون جایگزین و یا قانون تجدید نظر انتخاب کند.

روش دیگر، روش PART است یا Projective Adaptive Resonance Theory .
این روش یک الگوریتم درخت تصمیم گیری است. یک مجموعه از قوانین را توسط رویکرد تقسیم و غلبه ایجاد میکند سپس تمامی نمونه هایی از مجموعه داده های آموزشی که توسط این قوانین، پوشانده شده‌اند را از بین میبرد و به این روند را به طور بازگشتی تکرار میکند تا هیچ نمونه‌ای باقی نماند. الگوریتم PART برای آنکه یک قانون ایجاد کند، یک درخت C5.4 برای نمونه هایی که دارد میسازد و سپس برگی را که بیشتری پوشش از قانون جدید دارد را انتخاب میکند. در نهایت نمونه هایی که توسط قانون جدید ایجاد شد را از داده های آموزشی از بین میبرد تا از generalization جلوگیری کند. این روند ادامه پیدا میکند تا نمونه ها با توجه به قوانین استخراج شده پوشانده شوند.