گزارش پروژه سوم یادگیری ماشین

محمدرضا صیدگر_97222055

1)
ما در svm از کرنل استفاده میکنیم چون کرنل مجموعه ای از توابع ریاضی مورد استفاده در svm ، پنجره ای را برای دستکاری داده ها به کار می برد. بنابراین ، کرنل به طور کلی مجموعه آموزش داده ها را به گونه ای تغییر می دهد که سطح تصمیم گیری غیر خطی قادر به تبدیل شدن به یک معادله خطی در تعداد بیشتری از فضاهای بعد باشد.
درباره توابع اصلی و مهم کرنل به Gaussian Kernel میتوان اشاره کرد که هنگامی که هیچ ایده ای در مورد داده ها وجود ندارد ، برای transformation استفاده می شود.کرنل بعدی rbf است که همان شبیه کرنل فوق است با افزودن متد پایه شعاعی برای بهبود عملکرد آن.کرنل بعدی sigmoid است که این تابع معادل یک مدل پرسپترون دو لایه از شبکه عصبی است که به عنوان تابع فعال سازی برای نورونهای مصنوعی استفاده می شود.
2) برای این سوال قبل از هرچیزی اول داده ها را درون برنامه خوانده شد. سپس بررسی شد تا داده های null در دیتا ها وجود نداشته باشند که داده null اصلا وجود نداشت. در ادامه داده های پرت یا outliers رو حذف کردیم و همچنین داده ها را normalize کردیم یعنی همه را در بازه 0 و 1 آوردیم. در اخر y , x را جدا کردیم تا به مدل هایمان بدهیم تا نتایج را بدست بیاوریم.
با استفاده از پکیج های sklearn مدل ساده svm ساختیم تا روی داده ها اموزش ببیند و سپس روی داده های تست ارزیابی شود که مدل ما حدودا 84 درصد درستی داشت.

147	18	0	0
10	121	14	0
0	23	111	14
0	0	18	121

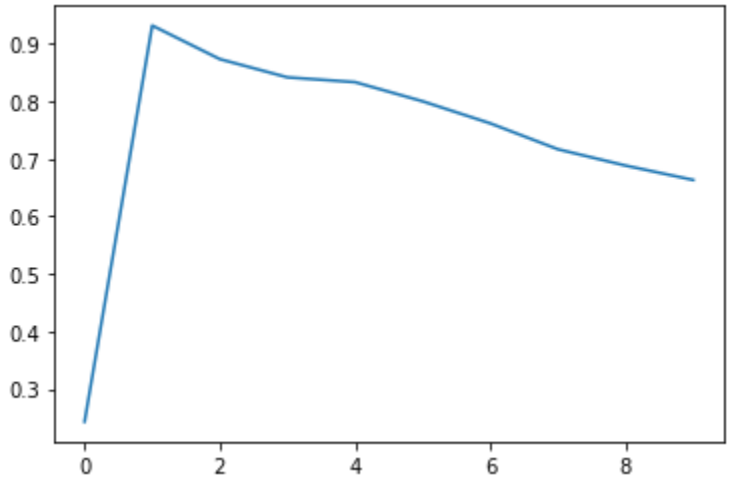
این هم confusion matrix این مدل ما بود که اعداد روی قطر نسبت به اعداد دیگر بزرگتر است یعنی درستی خوبی داشتیم.

3)

برای این بخش پارامتر ها را روی مدل سوال قبلی اجرا کردیم تا ببینیم چه تاثیری در خروجی یعنی درصد درستی ما خواهد داشت

در مدل جدید اول ما از کرنل خطی استفاده کردیم در واقع از کرنل خطی زمانی استفاده می شود که داده ها بصورت خطی قابل تفکیک باشند ، یعنی می توان آنها را با استفاده از یک خط جدا کرد. آموزش svm با کرنل خطی سریعتر از کرنل های دیگر است. در این مدل درستی ما حدود 93 درصد بود.

در مدل دوم سعی کردیم 10 مدل مختلف بگنجانیم ، از کرنل poly استفاده کردیم از درجه 0 تا 10 که درستی به شکل زیر شده است:



که x درجه poly را و y میزان درستی را نشان میدهد.
در مدل سوم از تابه تصمیمگیری one versus one استفاده کردیم تا ببینم درستی چقدر خواهیم داشت که درستی 84 درصد داشتیم.
در مدل چهارم متغیر gamma = auto قرار دادیم که این متغیر میزان تاثیر داده اموزشی را مشخص میکند که در این حالت درستی 88 درصد داشتیم که درستی بهتری شد.

(4

5) در بخش اول این سوال داده ها را در bin های مختلف ریختیم. در بازه های زیر بر حسب ستون battery power
(500,875.25),(875.25,1249.5),(1249.5,1623.75),(1623.75,1999)
و در مرحله بعدی به هرکدوم از این بازه ها برچسب های bad و medium و good و nice دادیم و این ها را به عنوان یک ستون جدید به داده ها اضافه کردیم.
در بخش دوم این سوال چون طبق بخش قبلی داده هایی با 4 برچسب مختلف به داده ها اضافه شد پس باید این داده ها را one hot encoding کنیم که از تابع get dummies در پکیج pandas استفاده کردیم و دلیل استفاده از one hot encoding این است که اجازه می دهد تا نمایش داده های categorical رساتر باشد. بسیاری از الگوریتم های یادگیری ماشین نمی توانند مستقیماً با داده های دسته ای کار کنند. این مورد هم برای متغیرهای ورودی و هم برای متغیرهای خروجی که categorical هستند لازم است.
در بخش سوم این سوال از گفته شد از log transform یا exp transform استفاده کنیم که من از تابع transform پکیج pandas استفاده کردم و داده ها را به log و exp بردم که اصلا نتایج خوبی را به دنبال نداشت چرا که طبق تصویر زیر که می بینید بعضی از داده ها برچسب -inf خورده اند به این معنی

که داده های 0 که داشتیم در لگاریتم طبیعتا حالا به منفی بینهایت تبدیل شده اند و این اجازه نمیدهد که بتوانیم مدل های یادگیری ماشین مثل svm را روی آن ها پیاده سازی کنیم.

	battery_power	blue	clock_speed	dual_sim	fc	four_g
0	6.735780	-inf	0.788457	-inf	0.000000	-inf
1	6.928538	0.0	-0.693147	0.0	-inf	0.0
2	6.333280	0.0	-0.693147	0.0	0.693147	0.0
3	6.421622	0.0	0.916291	-inf	-inf	-inf
4	7.507141	0.0	0.182322	-inf	2.564949	0.0

اما در بخش چهارم این سوال خواسته شد که ویژگی جدیدی به نام مساحت به داده ها اضافه کنیم که من ستون های طول و عرض را در هم ضرب کردم و با عنوان area یک ستون به داده ها اضافه کردم.

6) در این بخش پیاده سازی svm روی موارد بالا را بایستی ببینیم که روی مورد اول و دوم بهم وابسته اند از طرفی مدل سوم هم قابل اجرا نبود بخاطر داده های null و inf پس فقط به صورت جدا روی بخش 4 مدل svm زدیم که درستی 85 درصد داشتیم و به طور کل روی بخش 1 و 2 و 4 مدل svm درستی 82 درصد داشته است.

7)
الگوریتم درخت تصمیم به گونه ای عمل میکند که سعی دارد گوناگونی و یا تنوع را در گره ها به حداقل ممکن برساند.این عدم یکنواختی در گره ها با استفاده از معیار های measure impurity قابل اندازه گیری است که مهمترین و پرکاربردترین ان شاخص جینی میباشد.
اغلب تفاوت انواع درخت های تصمیم در همین معیار اندازه گیری measure impurity، شیوه شاخه بندی یا splitting و هرس کردن گره های درخت میباشد.

الگوریتم اول ID3 است که یکی از الگوریتم‌های بسیار ساده درخت تصمیم است که در سال 1986 توسط Quinlan مطرح شده است. اطلاعات به دست آمده به عنوان معیار تفکیک به کار می‌رود. این الگوریتم هیچ فرایند هرس کردن را به کار نمی‌برد و مقادیر اسمی و مفقوده را مورد توجه قرار نمی‌دهد.
الگوریتم دوم CART است که متغیر های ورودی را برای یافتن بهترین تجزیه می آزماید تا شاخص ناخالصی حاصل از تجزیه کمترین مقدار باشد. در تجزیه دو زیرگروه تعیین میشود و هر کدام در مرحله بعد به دو زیگروه دیگر تقسیم خواهند شد و این روند ادامه می یابد تا زمانی که یکی از معیار های توقف برآورده شود .درخت CART بازگشتی دو دویی است که گره های والدین را دقیقا به دو گروه فرزند منشعب میکند و به طور بازگشتی منشعب کردن را تا زمانی که انشعاب دیگری نتواند ساخته شود ادامه میدهد.
الگوریتم بعدی C4.5 است .این الگوریتم درخت تصمیم، تکامل یافته ID3 است

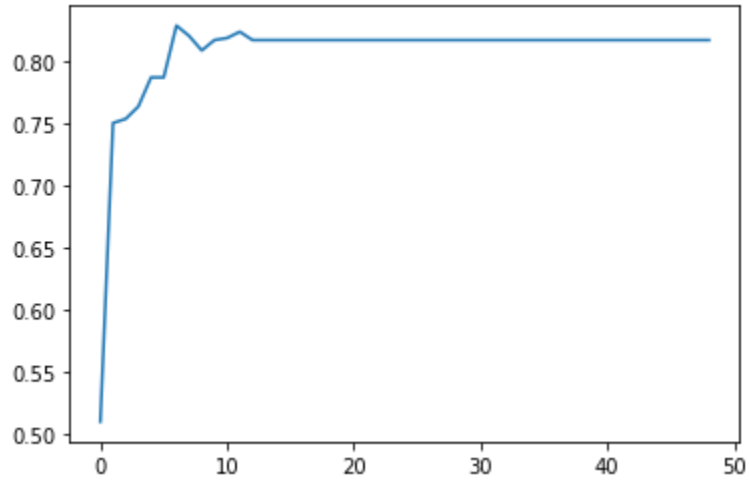
Gain Ratio به عنوان معیار تفکیک در نظر گرفته می‌شود. عمل تفکیک زمانی که تمامی نمونه‌ها پایین آستانه مشخصی واقع می‌شوند، متوقف می‌شود. پس از فاز رشد درخت عمل هرس کردن بر اساس خطا اعمال می‌شود. این الگوریتم مشخصه‌های اسمی را نیز در نظر می‌گیرد.

(8

برای این سوال از پکیج sklearn استفاده کردم از بخش decision tree classifier که یک مدل درخت تصمیم ساختیم که درستی 82 درصد داشته است روی داده های تست.

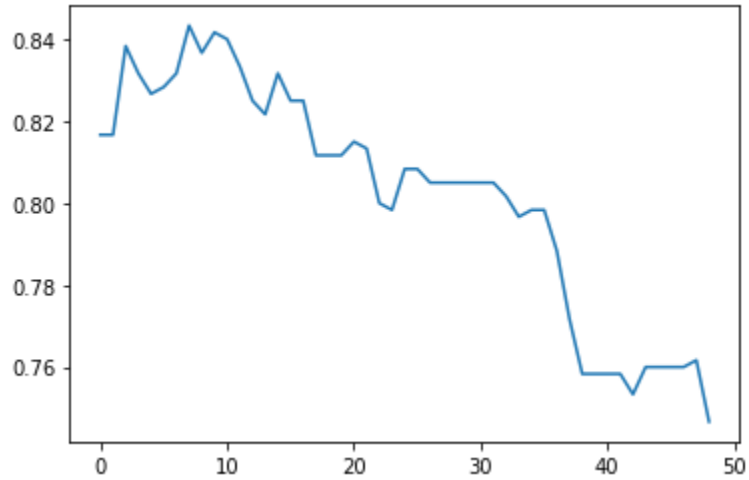
(9

در این سوال در بخش اول ماکزیمم عمق درخت رو تغییر دادیم از 1 تا 50 که نمودار تغییرات به شکل زیر شد



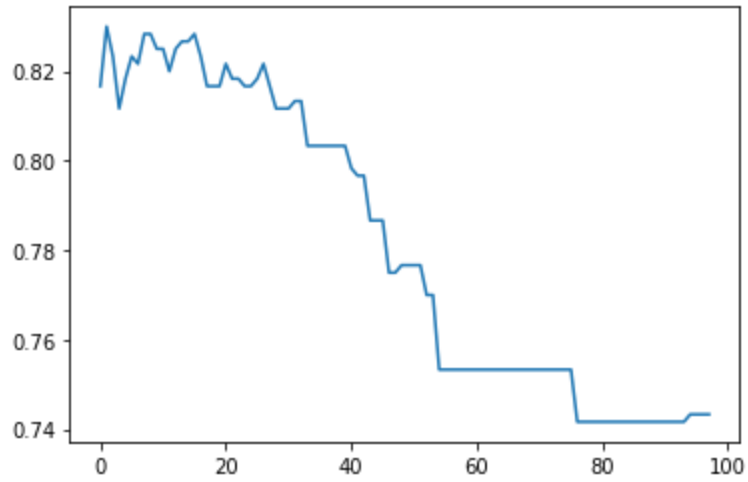
که کمترین مقدار 51 درصد در عمق 1 بوده و بیشترین 83 درصد در عمق 7 بوده است و از جایی به بعد درستی ثابت شده است.

در بخش بعدی بررسی کردیم تغییرات تعداد نمونه های موجود در گره های برگ را که شکل زیر شده است:



که کمترین مقدار 75 درصد در تعداد نمونه های 49 بوده و بیشترین 84 درصد در تعداد نمونه های 8 بوده است که به طور کلی از جایی به بعد نزولی شده است.

در بخش بعدی بررسی کردیم تغییرات تعداد نمونه های موجود در گره های داخلی را که شکل زیر شده است:



که کمترین مقدار 74 درصد در تعداد نمونه های 77 بوده و بیشترین 83 درصد در تعداد نمونه های 2 بوده است که به طور کلی از جایی به بعد نزولی شده است.

(10

هرس روش حذف شاخه های بلا استفاده از درخت تصمیم است. برخی از شاخه های درخت تصمیم ممکن است نمایانگر داده های نویز یا outlier باشد.

هرس درخت روشی است که شاخه های ناخواسته درخت را کاهش می دهد. این امر از پیچیدگی درخت می کاهد و به تحلیل پیش بینی موثر کمک می کند. به دلیل از بین بردن شاخه های غیر مهم درختان ، از overfitting آن می کاهد.
دو رویکرد رایج برای هرس درخت وجود دارد:

A)Pre pruning:

در این رویکرد یک درخت به وسیله توقف‌های مکرر در مراحل اولیه ساخت درخت، هرس می‌شود. به محض ایجاد یک توقف گره به برگ تبدیل می‌شود.

B)Post pruning:

رویکرد هرس پسین درخت تصمیم رایج‌تر است به این صورت که زیر درخت‌ها 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.

(12

برای این سوال از پکیج sklearn استفاده کردم از بخش random forest classifier که یک مدل رندم فارست ساختیم که درستی 87 درصد داشته است روی داده های تست که نشان میدهد عملکرد رندم فارست از درخت تصمیم گیری بهتر بوده است چون رندم فارست یک روش مدل سازی قوی و بسیار مقاوم تر از یک درخت تصمیم تنها هست. آنها درختان تصمیم گیری بسیاری را جمع می کنند تا برازش زیاد و همچنین خطای ناشی از overfitting را کم کنند و بنابراین نتایج مفیدی به دست می آورند.

(13

تفاوت های زیادی بین الگوریتم های درخت تصمیم و شبکه های عصبی وجود دارد ، اما از نظر عملی ، سه چیز اصلی را باید در نظر گرفت: سرعت ، تفسیرپذیری و دقت.

درباره درخت های تصمیم:

درخت های تصمیم پس از آموزش سریعتر هستند . این به این دلیل است که یک درخت تصمیم ذاتاً ویژگی های ورودی را "که دور از دسترس نیست" "دور می اندازد" ، در حالی که یک شبکه عصبی از همه آنها استفاده می کند مگر اینکه انتخاب ویژگی را به عنوان پیش پردازش انجام دهید.

درختان تصمیم گیرنده جریان طبیعی را به راحتی دنبال می کنند. با برنامه های IF، THEN، ELSE نیز برای سیستم های کامپیوتری برنامه ریزی آسان است.

می توانیم ببینیم که گره بالای درخت تأثیرگذارترین داده است که بر متغیر پاسخ در مدل تاثیر می گذارد. از آنجا که درک این درختان بسیار آسان است ، به

عنوان تکنیک های مدل سازی بسیار مفید هستند و نمایش های بصری داده ها را ارائه می دهند.
درباره شبکه عصبی:
شبکه عصبی کندتر است (هم برای آموزش و هم برای طبقه بندی)
درک شبکه عصبی از طریق نمایش بصری چندان آسان نیست. ایجاد سیستم های رایانه ای از آنها بسیار دشوار است و ایجاد توضیحات از مدل تقریباً غیرممکن است. شبکه های عصبی می توانند داده های باینری را بهتر از درختان تصمیم بگیرند اما مقادیر دسته بندی را نمی توانند مدیریت کنند.
نتیجه گیری:
با توجه به توضیحات بالا و توضیح چگونگی اداره هر مدل با داده ها ، می توان تصور کرد که درختان تصمیم مدل بهتری هستند. اما در کل این یک فرض نادرست است زیرا مناسب بودن مدل بازتاب دهنده مجموعه داده است. در حقیقت ، وجود دارد شبکه عصبی به 99٪ دقت در یک مجموعه داده رسیده است در حالی که مدل درخت تصمیم فقط 86٪ دقت را در همان مجموعه داده بدست آورده است. بهترین مدل نصب شده مدلی است که با دقت بیشتری متناسب با داده ها باشد اما در هرصورت بخاطر سرعت و تفسیر پذیری بهتر در درختان تصمیم این الگوریتم ها همچنان محبوب هستند.