به نام خدا

گزارش شماره ۲ (۱۰ اردیبهشت سال ۱۳۹۹)

عنوان پروژه: بررسی مسئله درهم‌تنیدگی کوانتومی برای سیستم‌های دوذره‌ای توسط روش‌های یادگیری ماشین

اعضا: سیده فاطمه دهقان، مهکامه سلیمی، سحر تغیر، محمدمهدی ماستری فراهانی

فهرست:

۰ ایجاد داده ۴

۱ آموزش و ارزیابی مدل ۵

۱.۱ متر مناسب برای مسئله: ۵

۲.۱ مدل k Nearest Neighbors: ۵

۳.۱ مدل درخت تصمیم (Decision Tree): ۶

٤.۱ مدل جنگل تصادفی (Random forest): ۸

۵.۱ مدل SVM _ linear: ۹

٦.۱ مدل چندجمله ای مرتبه ۹ (SVM_POLY): ۱۱

۷.۱ مدل SVM_RBF: ۱۲

۸.۱ مدل Naive_bayes: ۱۳

۹.۱ مدل AdaBoost: ۱۴

۱۰.۱ مدل quadratic discriminant analysis: ۱۶

۲ تنظیم دقیق مدل‌ها ۱۷

۱.۲ رسم منحنی اعتبارسنجی (Validation curve): ۱۷

۱.۱.۲ مدل k Nearest Neighbors: ۱۷

۲.۱.۲ مدل درخت تصمیم (Decision Tree): ۱۹

۳.۱.۲ مدل جنگل تصادفی (Random Forest): ۲۱

٤.۱.۲ مدل SVM_LINEAR: ۲۳

۵.۱.۲ مدل SVM_poly (چند جمله ای مرتبه ۹): ۲۴

٦.۱.۲ مدل SVM_RBF: ۲۴

۷.۱.۲ مدل naïve bayes: ۲۶

۱.۲.۲ مدل k Nearest Neighbors: ۲۷

۲.۲.۲ مدل درخت تصمیم (Decision Tree): ۲۷
۳.۲.۲ مدل جنگل تصادفی (Random Forest) ۲۸
٤.۲.۲ مدل SVM_Linear: ۲۹
٥.۲.۲ مدل SVM_poly (مرتبه ۹): ۳۰
٦.۲.۲ مدل SVM_RBF: ۳۱
۷.۲.۲ مدل naive bayes: ۳۳
۸.۲.۲ مدل AdaBoost: ۳۴
۹.۲.۲ مدل QDA: ۳۵
۳.۲ رسم منحنی یادگیری (Learning curve) و بررسی مدل و کافی بودن میزان داده ۳۶
۱.۳.۲ مدل k Nearest Neighbors: ۳۶
۲.۳.۲ مدل درخت تصمیم (Decision Tree): ۳۷
۳.۳.۲ مدل جنگل تصادفی (Random Forest): ۳۸
٤.۳.۲ مدل SVM_LINEAR: ۳۹
٥.۳.۲ مدل SVM_poly (مرتبه ۹): ۴۰
٦.۳.۲ مدل SVM_RBF: ۴۰
۷.۳.۲ مدل naive bayes: ۴۱
۸.۳.۲ مدل AdaBoost: ۴۲
۹.۳.۲ مدل QDA: ۴۳
۳ نتیجه گیری کلی و تهیه جدول برای مدل‌ها ۴۴
نقش اعضای گروه ۴۵
مراجع ۴۶

• ایجاد داده

داده های ساخته شده قبلی در بعضی موارد دارای برچسب ۲ به معنای مشخص نبودن دقیق حالت درهم‌تنیده یا جداپذیر بوده‌اند و برای برچسب‌گذاری دقیق‌تر این ماتریس‌های چگالی تصادفی ساخته شده از روش بهینه سازی به نام linear programming استفاده شده است. مساله‌ی ما این است که از آن‌جایی که ماتریس‌های چگالی خالص جداپذیر، نقاط اکستریم فضای ماتریس‌های چگالی جداپذیر هستند، می‌توانیم با آن‌ها یک کانوکس هال بسازیم و برای تشخیص ماتریس‌های جداپذیر تصادفی، از چک کردن این که در این کانوکس هال هستند یا نه استفاده کنیم. در ابتدا برای حل این مسئله تلاش برای پیدا کردن روشی برای تشخیص مستقیم نقاط قرار گرفته داخل convex hull بود، اما بعد از آزمون چند روش تصمیم بر آن شد که از روش بهینه سازی ترکیب های محدب (convex combination) به دست آمده از بردارهای ویژگی (ضرایب گلمان) ماتریس‌های تصادفی استفاده کنیم.[1] بدین منظور بالغ بر ۶۴۰۰ حالت جداپذیر که محدب هستند را به تابع linear programing به عنوان مقادیر اولیه داده شد تا یک ناحیه محدب مناسب برای بهینه سازی ساخته شود.[1]

در ابتدا از تابع linear programing موجود در کتابخانه Scipy پایتون استفاده شد و برای پوشش دادن کامل فضا تمامی نقاط مرزی با استفاده از ماتریس های یکانی ۱۰ بار دوران داده شدند. مدت زمان تحلیل داده‌ها با توجه به توضیحات ذکر شده در پایتون بسیار طولانی می‌شد و برای تحلیل داده‌های بیشتر استفاده از پایتون ممکن نبود به همین جهت با استفاده از نرم افزار MATLAB و استفاده از تابع linear programming متلب با سرعت بسیار بالاتر و با **خطای ۰ درصد** داده ها برچسب گذاری شدند.

با توجه به این مهم که تعداد حالت‌های جداپذیری که PPT تشخیص داده می‌شوند با افزایش بعد فضا به صورت نمایی افت پیدا می‌کند[2] تصمیم بر اضافه کردن داده‌های جداپذیر به صورت دستی به کل داده‌ها شد تا جمعیت داده‌ها به نحوی باشد که بتوان از روش‌های classification به صورت مطلوب استفاده کرد.

۱ آموزش و ارزیابی مدل

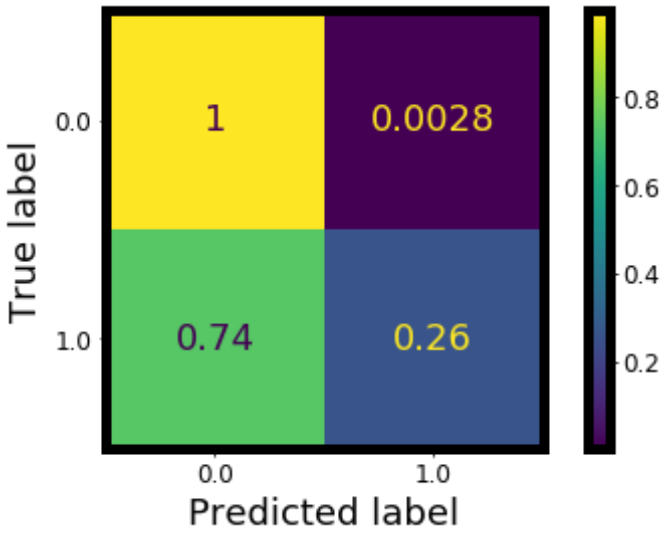
۱.۱ متر مناسب برای مسئله:

با توجه به داده‌های آماده شده، مشخص است که جمعیت حالت‌های جداپذیر و درهم تنیده در یک مرتبه عددی هستند و بنابراین مترهای precision و recall به تنهایی مترهای مناسبی محسوب نمی‌شوند، به این علت که اگر هرکدام به تنهایی درنظر گرفته شود، اطلاعات مناسبی از میزان فاصله نتایج مدل و داده‌های اصلی به ما نمی‌دهند. به نوعی باید هر دو در نظر گرفته شوند که می‌توان مترهای f1-score یا accuracy را برای اینکار انتخاب کرد. به نظر می‌رسد که هر دو متر مناسب باشند و در بخش‌های بعدی از هر دو متر برای اعتبار سنجی مدل‌ها استفاده خواهیم کرد.

۲.۱ مدل k Nearest Neighbors:

این مدل دارای پارامترهایی هست که شاید مهم ترین آن‌ها *تعداد همسایه‌ها* باشد. مدل را در ابتدا برای ۶ همسایه آموزش دادیم و نتایج زیر به ازای تعداد ۱۱۳۱ داده آزمون در خروجی مدل بدست آمد.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

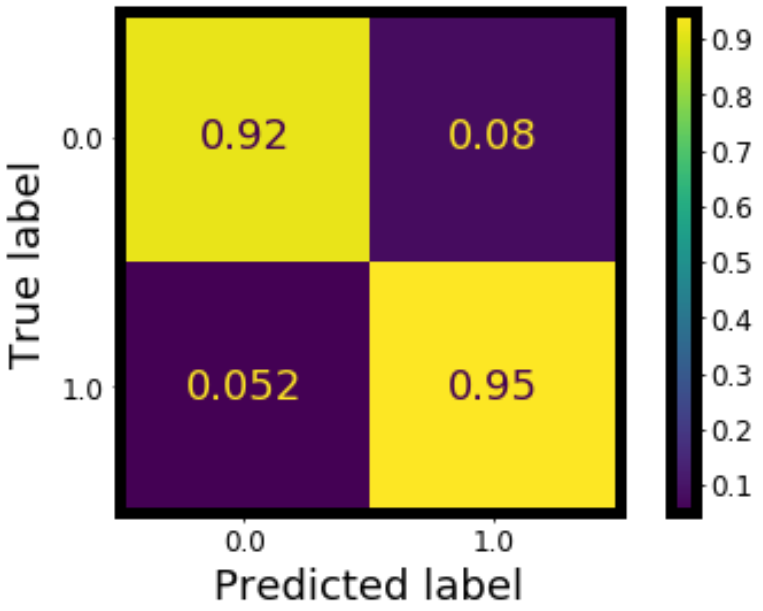
<table>
<tr><th colspan="6">k Nearest Neighbors (k=6)</th></tr>
<tr><td>accuracy</td><td colspan="5">0.518</td></tr>
<tr><td>Confusion matrix</td><td colspan="5">Report</td></tr>
<tr><td rowspan="3">$\begin{pmatrix} 2110 & 6 \\ 2607 & 930 \end{pmatrix}$</td><td>label</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr>
<tr><td>0</td><td>0.43</td><td>1.00</td><td>0.60</td><td>405</td></tr>
<tr><td>1</td><td>1.00</td><td>0.25</td><td>0.40</td><td>726</td></tr>
</table>

از این گزارش معلوم است که این مدل برای داده‌های ما مناسب نبوده است . همانطور که گفته شد مترهای presicion و recall به تنهایی معیار مناسبی نیستند. ولی از نگاه به هر دو متر accuracy و f1-score متوجه می‌شویم که در مجموع این مدل مناسب نبوده است. اما اینکه آیا می‌توان این نامناسب بودن را با تعداد بیشتر داده جبران کرد و یا با تغییر پارامتر مدل بحثی است که در بخش بعدی گزارش مورد بررسی قرار خواهد گرفت.

۳.۱ مدل درخت تصمیم (Decision Tree):

این مدل دارای پارامترهایی هست که شاید مهم ترین آن‌ها عمق درخت یا همان تعداد سوال‌ها باشد. مدل را در ابتدا برای عمق درخت ۵ آموزش دادیم و نتایج زیر به ازای تعداد ۱۱۳۱ داده آزمون در خروجی مدل بدست آمد.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

Decision Tree (max_depth=5)					
accuracy	0.912				
Confusion matrix	**Report**				
$\begin{pmatrix} 1946 & 170 \\ 184 & 3353 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.86	0.90	0.88	405
	1	0.94	0.92	0.93	726

از این گزارش می‌توان دریافت که این مدل برای داده‌های ما نتایج نسبتاً مناسبی داشته است. هر دو متر accuracy و f1-score نتایج خوبی را برای داده‌های ورودی به دست آورده‌اند. همچنین از ماتریس درهم

آمیختگی میفهمیم که این مدل نسبتا پیش‌بینی‌های مناسبی دارد. هم اینکه عناصر غیرقطری ماتریس‌ها جمعیت کمی دارند و این خود نشان از پرجمعیت بودن True-negative و True-positive دارد. بنابراین در مجموع مدل مناسبی برای این مسئله به نظر می‌رسد. البته که باید مدل بیشتر مورد بررسی قرار بگیرد و پارامترهای آن بهینه شوند و همچنین کافی بودن میزان داده نیز مورد بررسی قرار بگیرد که در بخش بعدی پیگیری خواهد شد.

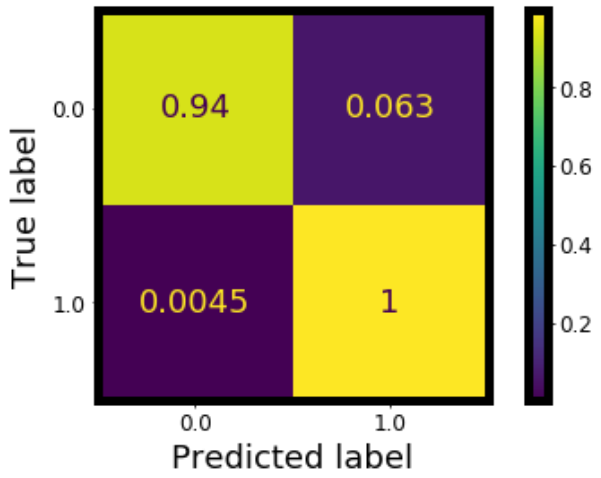
٤.١ مدل جنگل تصادفی (Random forest):

در حقیقت این مدل از مدل قبلی نشئت می‌گیرد به این صورت که با در نظر گرفتن چندین درخت (و نه فقط یک درخت)، یک جنگل درست می‌کند. سپس با تقسیم بندی داده به تعداد درخت‌ها، هرکدام از درخت‌ها را آموزش می‌دهد. در نهایت هنگام آزمودن مدل توسط یک داده جدید، هر درخت پیش‌بینی دارد که توسط رای گیری یکی از این پیش‌بینی‌ها به عنوان خروجی مدل نتیجه می‌شود.

از همین الگوریتم به راحتی می‌توان دید که دو پارامتر برای این مدل حیاتی هستند. یکی *تعداد درخت‌ها* است و دیگری حداکثر *عمق درخت‌ها*. البته که پارامترهای دیگری نیز وجود دارند ولی ما به بررسی همین دو پارامتر بسنده کردیم.

با تعداد درخت ۱۰ و حداکثر عمق درخت ۵، این مدل برای تعداد داده ۱۱۳۱ آموزش دادیم و نتایج زیر بدست آمد.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

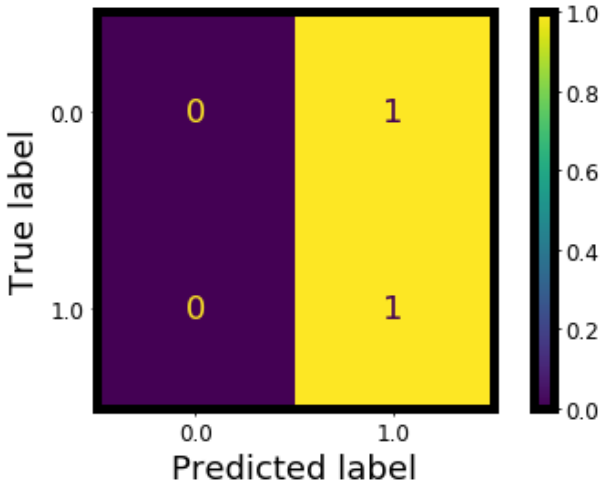
Random Forest (max_depth=5, n_estimators=10)					
accuracy	0.972				
Confusion matrix	**Report**				
$\begin{pmatrix} 1982 & 134 \\ 16 & 3521 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.98	0.94	0.96	413
	1	0.96	0.99	0.98	718

از گزارش می‌توان دریافت که البته یک جنگل از درخت‌ها بسیار مناسب‌تر از یک درخت کارایی دارند. با نگاه به ماتریس درهم ریختگی متوجه می‌شویم که جمعیت عناصر غیر قطری به نسبت مدل درخت تصمیم کمتر است و برای داده‌های درهم‌تنیده (label = 1) بسیار خوب تشخیص داده شده است. همچنین با نگاه به مترهای accuracy و f1-score می‌توان دریافت که در مجموع مدل به خوبی برای داده‌های ما کار می‌کند. بررسی بیشتر این مدل و بهینه کردن پارامترهای آن کاری است که در ادامه انجام خواهیم داد.

۵.۱ مدل SVM _ linear:

این مدل دارای پارامتر هایی از جمله c (regularization parameter) و گاما که فاصله تاثیر آموزش تکی در مدل را نمایش می‌دهد. این مدل را ابتدا با c =0.025 آموزش داده‌ایم و نتایج زیر دست به دست آمده است.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

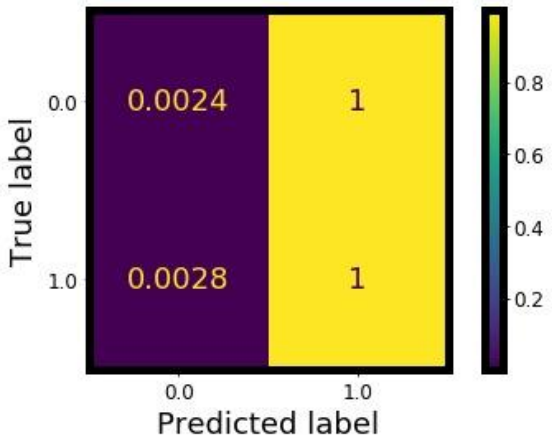
SVM_LINEAR (C = 0.025)					
accuracy	0.635				
Confusion matrix	**Report**				
$\begin{pmatrix} 0 & 2116 \\ 0 & 3537 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.0	0.0	0.0	412
	1	0.64	0.1	0.78	718

با توجه با گزارش بالا و کم بودن مقادیر accuracy , f1_score میتوان فهمید این مدل نیاز به بهینه سازی دارد. برای داده ها با برچسب ۱ نسبت به داده ها با برچسب ۲ نتایج بهتری حاصل شده است اما در حالت کلی با توجه به معیار های مورد نظر ما مدل مناسبی برای مسئله ما نیست.

۶.۱ مدل چندجمله ای مرتبه ۹ (SVM_POLY):

در این مدل از چند جمله ای مرتبه ۹ استفاده شده است. از جمله پارامترهای موثر برای این مدل میتوان به c و گاما که در بخش قبل ذکر شده بود، اشاره کرد. این مدل را برای c = 0.025 آموزش دادیم.

ماتریس درهم ریختگی (Confusion matrix):



گزارش کلی مدل:

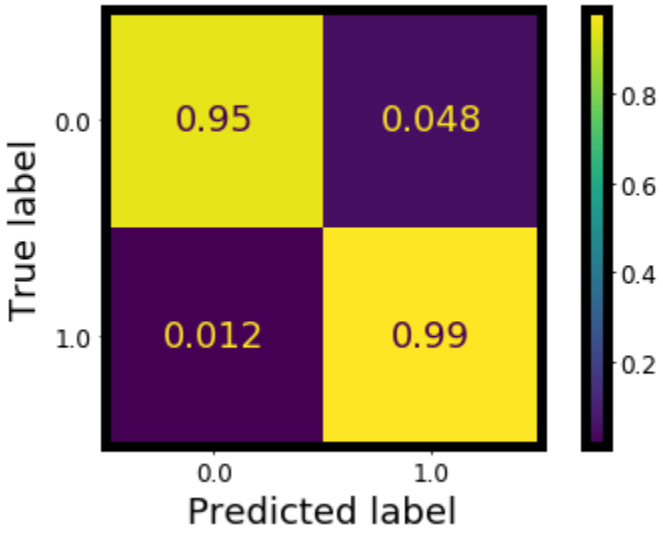
SVM_poly (degree = 9, C = 0.025)					
accuracy	0.634				
Confusion matrix	**Report**				
$\begin{pmatrix} 7 & 2109 \\ 2 & 3535 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.33	0.0	0.0	412
	1	0.64	0.1	0.78	718

با توجه با گزارش بالا و کم بودن مقادیر accuracy , f1_score میتوان فهمید این مدل نیاز به بهینه سازی دارد. این مدل نسبت به مدل SVM_LINEAR اندکی داده های بیشتری را با برچسب ۰ تشخیص داده است اما کماکان با توجه به معیارهای مورد نظر ما مدل مناسبی برای مسئله ما نیست.

۷.۱ مدل SVM_RBF:

این مدل مانند سایر SVMهاست، و تنها فرق آن کرنل RBF است. این مدل نیز دارای پارامترهای C و gamma است. نخست این مدل را برای پارامترهای C=0.5, gamma =10 آموزش داده، نتایج زیر حاصل شده است.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

SVM_rbf (C=0.5 , gamma =10)	
accuracy	0.974

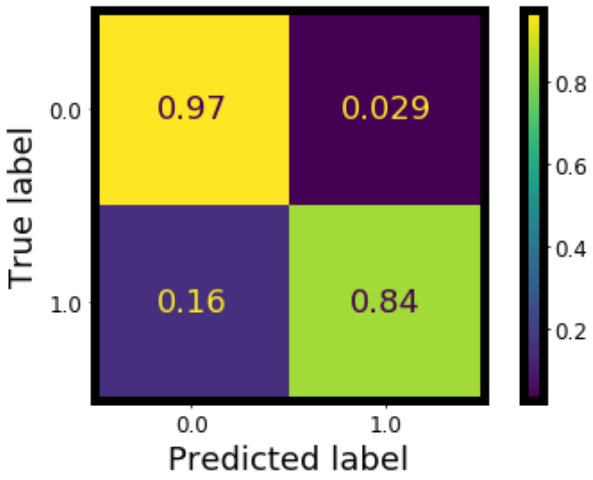
Confusion matrix	Report				
$\begin{pmatrix} 2007 & 109 \\ 22 & 3515 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.98	0.95	0.97	442
	1	0.97	0.99	0.98	689

همچنین از ماتریس درهم آمیختگی میفهمیم که این مدل نسبتا پیش‌بینی‌های مناسبی دارد. هم اینکه عناصر غیرقطری ماتریس‌ها جمعیت کمی دارند و این خود نشان از پرجمعیت بودن True-negative و True-positive دارد. بنابراین در مجموع مدل مناسبی برای این مسئله به نظر می‌رسد.

۸.۱ مدل Naive_bayes:

این مدل یک مدل احتمالاتی است، به این معنی که برای تشخیص کلاس، دو عامل احتمال بودن در آن کلاس طبق نتایج کنونی و ضمنا همسایه‌های عضو را بررسی می‌کند. به نظر تنها پارامتر مهم در این مدل واریانس یک تابع توزیع احتمال گاوسی است که var_smoothing نام دارد. مدل را آموزش دادیم و نتایج زیر به دست آمد:

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

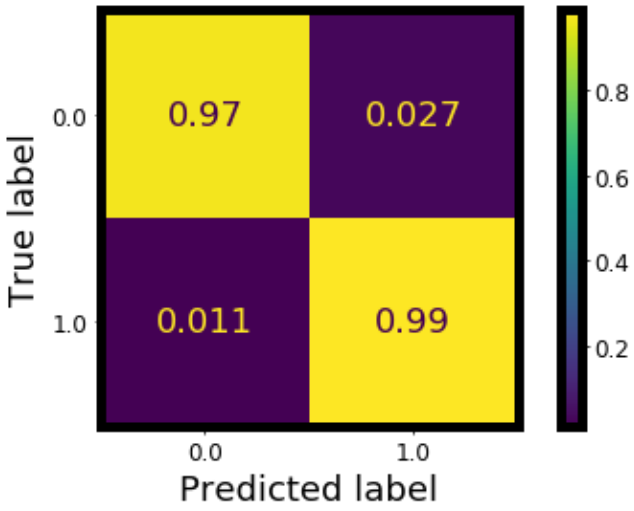
<table>
<tr><th colspan="6">naive_bayes</th></tr>
<tr><td>accuracy</td><td colspan="5">0.887</td></tr>
<tr><td>Confusion matrix</td><td colspan="5">Report</td></tr>
<tr><td rowspan="3">$\begin{pmatrix} 2038 & 678 \\ 498 & 3039 \end{pmatrix}$</td><td>label</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr>
<tr><td>0</td><td>0.78</td><td>0.97</td><td>0.86</td><td>412</td></tr>
<tr><td>1</td><td>0.98</td><td>0.84</td><td>0.90</td><td>719</td></tr>
</table>

این مدل در زمینه‌ی accuracy و f1-score نسبتا خوب عمل کرده اما نیاز به بهینه‌سازی دارد و می‌توان نتیجه‌ی بهتری از آن گرفت.

۹.۱ مدل AdaBoost:

این مدل در واقع یک meta-estimator است که با فیت کردن یک classifier بر دیتا و سپس فیت کردن کپی‌های دیگری از این classifier بر همان دیتا و وزن مختلف دادن به آن‌ها یک مدل قوی از چند مدل ضعیف می‌سازد. در واقع این کار برای بالا بردن accuracy است. پارامترهای قابل تغییر در این مدل base_estimator، learning_rate و n_estimators هستند. base_estimator در حالت پیش‌فرض Decision Tree است.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

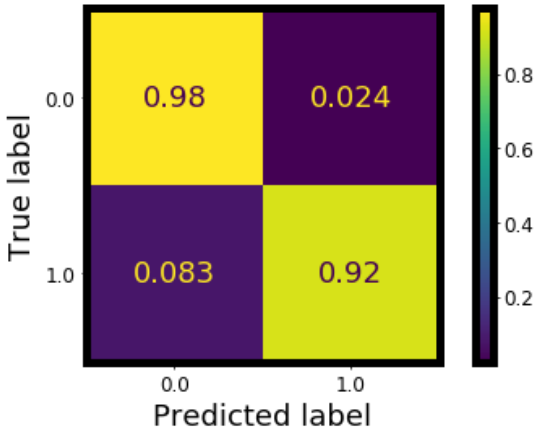
AdaBoost					
accuracy	0.983				
Confusion matrix	**Report**				
$\begin{pmatrix} 2073 & 43 \\ 32 & 3505 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.98	0.97	0.98	412
	1	0.98	0.99	0.99	719

این مدل accuracy بسیار بالایی داشت و بقیه‌ی متریک‌ها هم برای هر دو کلاس خوب هستند.

۱۰.۱ مدل quadratic discriminant analysis:

مرز تصمیم‌گیری این مدل یک تابع درجه ۲ است از این رو به آن QDA می‌گویند. پارامتر قابل تغییر در این مدل reg_param است.

ماتریس درهم ریختگی (Confusion Matrix):



گزارش کلی مدل:

QDA					
accuracy	0.938				
Confusion matrix	**Report**				
$\begin{pmatrix} 2058 & 58 \\ 244 & 3293 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.87	0.98	0.92	412
	1	0.99	0.92	0.95	719

می‌بینیم که مدل خوب عمل کرده و f1-score و recall آن بالا هستند. و accuracy بالایی نیز دارد.

۲ تنظیم دقیق مدل‌ها

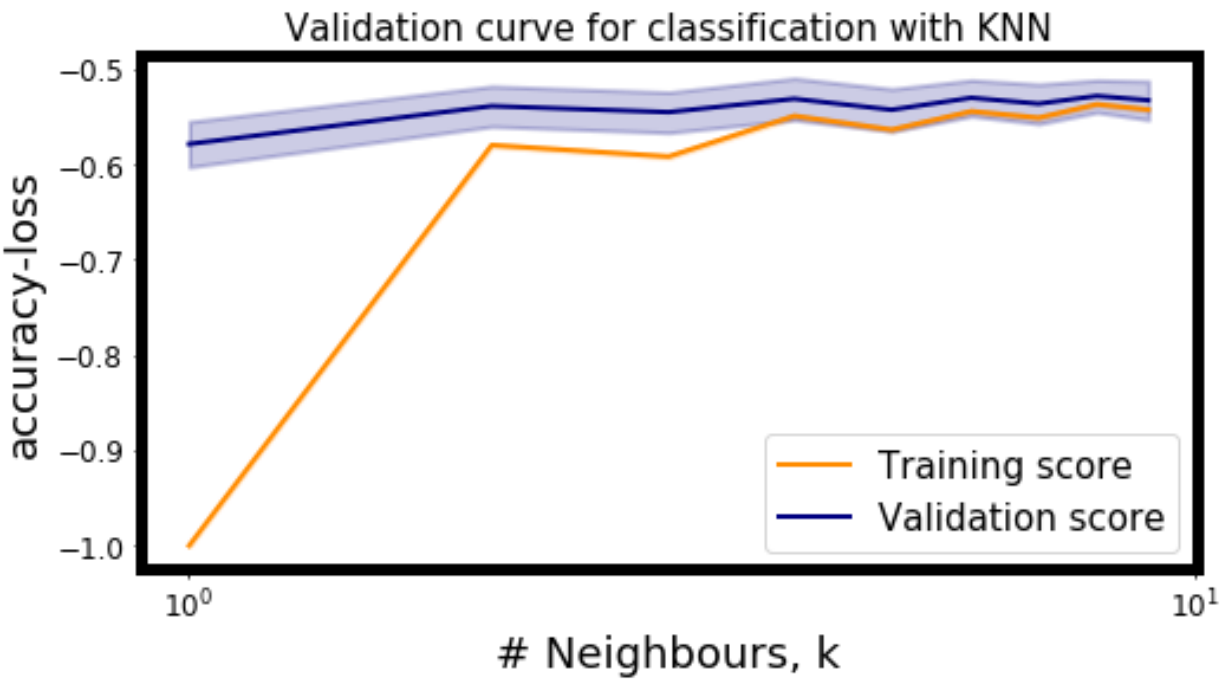
۱.۲ رسم منحنی اعتبارسنجی (Validation curve):

در این بخش برای هر مدل منحنی اعتبارسنجی را رسم می‌کنیم تا مقدار بهینه پارامترهای مدل را ارزیابی و پیدا کنیم. منحنی اعتبارسنجی، اتلاف (loss) مدل را بر حسب پیچیدگی مدل برای داده‌های ورودی و داده‌های آزمون رسم می‌کند و با مقایسه رفتار این دو منحنی می‌توان پارامتر بهینه مدل را یافت.

۱.۱.۲ مدل k Nearest Neighbors:

از آنجایی که این مدل با تعداد همسایه ۶ نتایج خوبی را برای داده‌های ما بدست نیاورد، اولین انتظاری که می‌رود اینست که شاید با پیچیده کردن مدل بتوان پیش‌بینی‌های بهتری یافت. بنابراین منحنی اعتبارسنجی را برای تعداد همسایه پایین رسم کرده‌ایم. (می‌دانیم که در این مدل هرچه تعداد همسایه‌ها کمتر باشد مدل پیچیده‌تر خواهد شد).

منحنی اعتبارسنجی:



دقت شود که علامت منفی در محور عمودی در حقیقت از منفی کردن score مدل بدست آمده و هرچه این مقدار به سمت 1- می‌رویم اتلاف کمتری داریم.

منحنی برای اتلافی که توسط متر accuracy تعریف می‌شود، رسم شده است. از این منحنی می‌توان دریافت که هرچه مدل پیچیده‌تر می‌شود، میزان اتلاف در داده‌های ورودی کم می‌شود ولی از آن طرف این اتلاف اختلاف زیادی با اتلاف ناشی از داده‌های آزمون دارد. از نمودار پیداست که نقطه بهینه چیزی حدود تعداد همسایه برابر ۳ است که هم اتلاف داده‌های ورودی و داده‌های آزمون در یک مرتبه هستند و هم کمترین میزان اتلاف را در مجموع دارد. ولی همچنان در این نقطه نیز اتلاف زیاد است. مدل را برای تعداد همسایه ۳ و ۴ آموزش دادیم و نتایج را در جداول زیر آوردیم:

<table>
<tr><th colspan="6">k Nearest Neighbors (k=3)</th></tr>
<tr><td>accuracy</td><td colspan="5">0.555</td></tr>
<tr><td>Confusion matrix</td><td colspan="5">Report</td></tr>
<tr><td rowspan="3">$\begin{pmatrix} 2110 & 6 \\ 2369 & 1168 \end{pmatrix}$</td><td>label</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr>
<tr><td>0</td><td>0.46</td><td>0.99</td><td>0.63</td><td>436</td></tr>
<tr><td>1</td><td>0.98</td><td>0.28</td><td>0.44</td><td>695</td></tr>
</table>

k Nearest Neighbors (k=4)	
accuracy	0.532

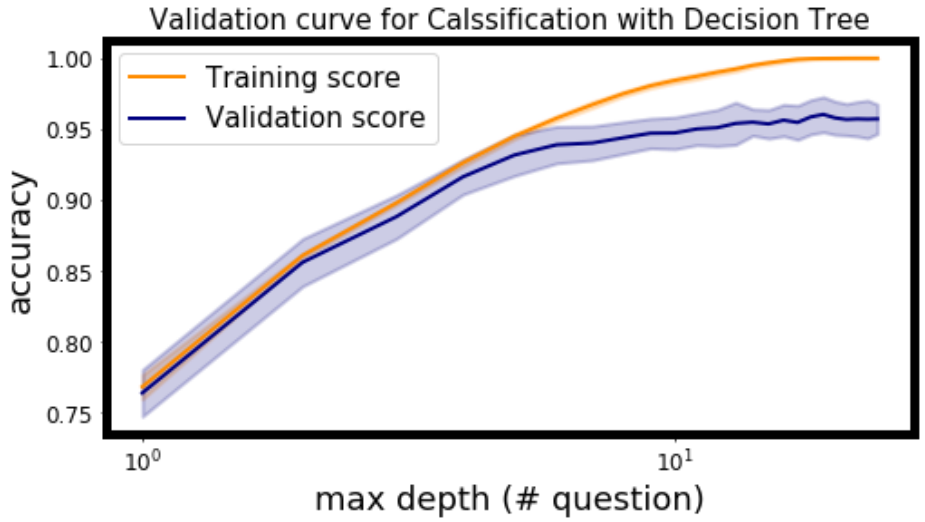
Confusion matrix	Report				
$\begin{pmatrix} 2116 & 0 \\ 2597 & 940 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.45	1.00	0.62	436
	1	1.00	0.24	0.39	695

از این نتایج پیداست که حتی بعد از پیدا کردن نقطه بهینه، همچنان مدل دارای accuracy و f1-score پایینی است. از ماتریس درهم آمیختگی می‌توان فهمید که مدل برای داده‌های جداپذیر خوب عمل کرده است ولی برای داده‌های درهم تنیده میزان زیادی را اشتباه تشخیص داده است. همه این دلایل ما را به این واقعیت می‌رسانند که در مجموع مدل k nearest neighbors مدل خوبی برای مسئله ما نیست.

۲.۱.۲ مدل درخت تصمیم (Decision Tree):

در نگاه اولی که به این مدل داشتیم متوجه شدیم که این مدل برای داده‌های ما دارای accuracy و f1-score بالایی است و امید است که مدل خوبی برای مسئله ما باشد. اکنون قصد داریم که دقیق‌تر به این مدل نگاه کنیم. ابتدا با رسم منحنی اعتبارسنجی پارامتر بهینه مدل را می‌یابیم. سپس با این پارامتر بهینه، مدل را آموزش می‌دهیم و نتایج را بررسی می‌کنیم.

منحنی اعتبارسنجی:



دقت شود در این مدل برخلاف مدل k nearest neighbor، هرچه به سمت جلو می‌رویم مدل پیچیده‌تر می‌شود. یعنی هرچه عمق درخت‌ها زیاد می‌شود مدل هم پیچیده‌تر می‌شود.

از این نمودار می‌توان دریافت که درست در نقطه‌ای که دو نمودار داده‌های ورودی و آزمون از هم دور می‌شوند، نقطه بهینه است. علت هم واضح است چراکه از این نقطه به بعد مدل over fit کرده و از این نقطه به قبل مدل under fit کرده است. می‌توان دریافت که این نقطه تقریباً برای عمق درخت به اندازه ۷ و یا ۸ است. اکنون مدل را بار دیگر برای این دو عمق رسم می‌کنیم و نتایج را در زیر می‌آوریم:

Decision Tree (max_depth=7)					
accuracy	0.942				
Confusion matrix	**Report**				
$\begin{pmatrix} 1981 & 135 \\ 60 & 3477 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.95	0.90	0.92	434
	1	0.94	0.97	0.95	697

Decision Tree (max_depth=8)					
accuracy	0.938				
Confusion matrix	**Report**				
$\begin{pmatrix} 1981 & 135 \\ 45 & 3492 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.94	0.89	0.92	434
	1	0.93	0.97	0.95	697

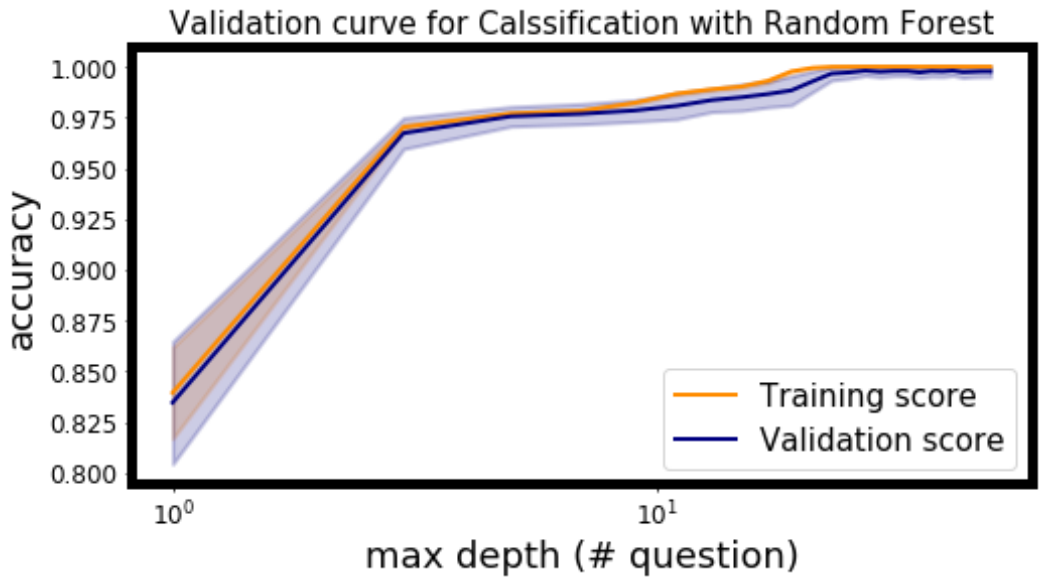
به نسبت آموزش اولیه نتایج بهتر شده‌اند ولی همچنان تعداد false-positive و false-negative نسبتاً زیاد است و تعدادی داده همچنان توسط مدل درست پیش‌بینی نشده‌اند. انتظار داریم که مدل جنگل تصادفی که از تعدادی درخت به جای یک درخت استفاده می‌کند، نتایج حتی بهتر از این هم بگیریم. با نگاه به مترهای accuracy و f1-score متوجه می‌شویم که بهترین انتخاب برای حداکثر عمق درخت، ۷ است و همچنین در این سطح از پیچیدگی مدل دچار over fit و یا under fit نمی‌شود.

۳.۱.۲ مدل جنگل تصادفی (Random Forest):

این مدل بر خلاف مدل درخت تصمیم دارای ۲ پارامتر مهم است که یکی تعداد درختان است و دیگری حداکثر عمق هر درخت. بنابراین رسم نمودار اعتبار سنجی برای این مدل سرراست نخواهد بود و مناسب‌تر است که پارامترهای این مدل با الگوریتم‌هایی نظیر جستجوی شبکه‌ای و یا جستجوی تصادفی بهینه شوند. البته در اینجا منحنی‌های اعتبار سنجی را برای هر کدام از پارامترها رسم می‌کنیم تا نتایج را ببینیم ولی باید این نکته را در نظر گرفت که برای بهینه کردن پارامترها باید هر دو را با هم در نظر بگیریم و این ویژگی در رسم منحنی‌های اعتبار سنجی وجود ندارد.

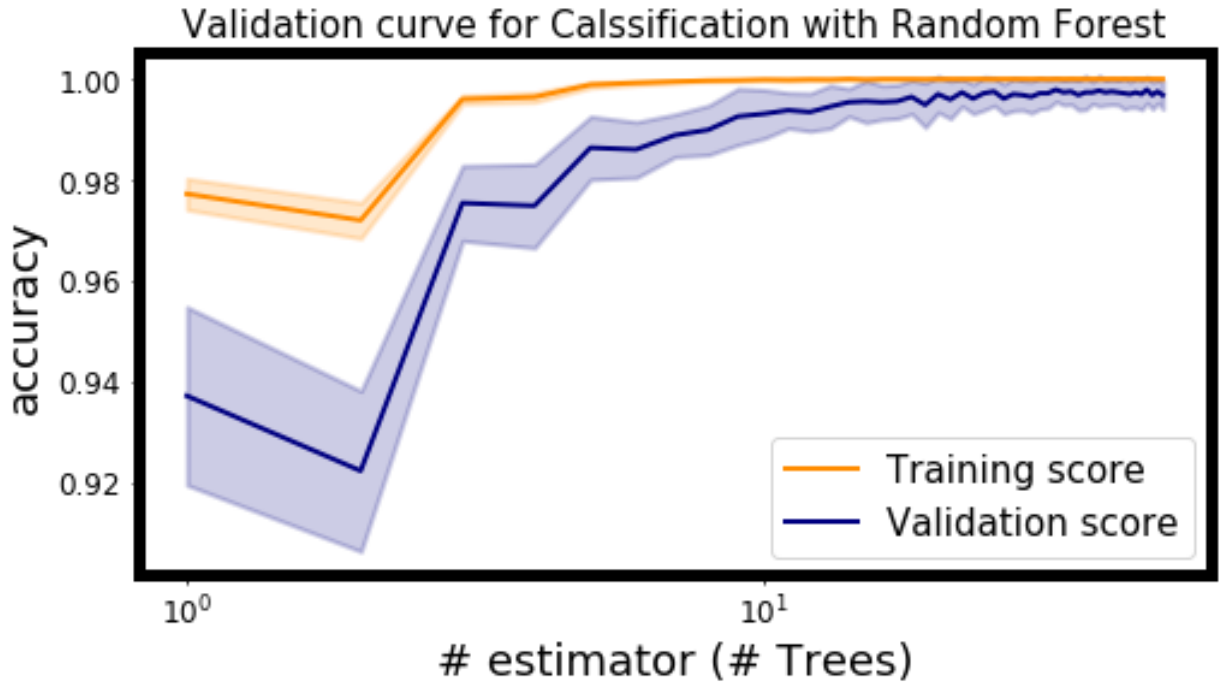
منحنی اعتبارسنجی:

در اینجا ما ابتدا منحنی اعتبار سنجی را برای پارامتر عمق درخت‌ها رسم کردیم که به صورت زیر درآمد:



این نمودار برای تعداد درخت ۱۰ رسم شده است. آنچه از نمودار می‌فهمیم اینست که برای عمق‌های مختلف score هر دو منحنی یکسان است و حتی با پیچیده کردن هم تفاوتی حاصل نمی‌شود. ما نتیجه گرفتیم که این رویداد ناشی از اینست که پارامتر عمق درختان نقش اساسی در تعیین score مدل به نسبت پارامتر تعداد درختان بازی نمی‌کند. یعنی در بررسی از روی منحنی اعتبارسنجی، تعیین نقطه بهینه برای تعداد درختان کافی است. البته که هر دو پارامتر در ادامه توسط الگوریتم‌های جستجو بهینه می‌شوند ولی در این بررسی ما فقط می‌توانیم نقطه بهینه را برای تعداد درختان تعیین کنیم.

منحنی اعتبارسنجی پارامتر تعداد درختان برای حداکثر عمق درخت ۵ به صورت زیر می‌باشد:



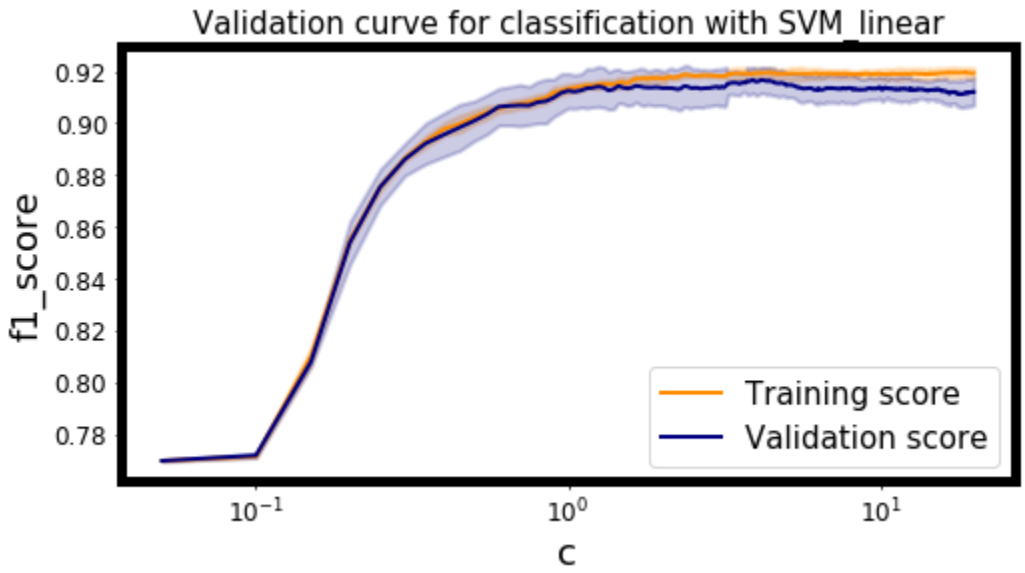
منحنی برای بازه ۰ تا ۵۰ عدد درخت رسم شده است. این نمودار نیز کمی با حالت کلی منحنی اعتبارسنجی تفاوت دارد. معمولا انتظار می‌رود که با افزایش پیچیدگی، score دو منحنی از هم فاصله گرفته و نقطه‌ای که دو منحنی شروع به فاصله گرفتن می‌کنند همان نقطه بهینه می‌شود. ولی این مدل نشان می‌دهد که ظاهراً با افزایش پیچیدگی هر دو منحنی به سمت میل پیدا می‌کنند و شاید برخی این نتیجه را بگیرند که هرچه مدل پیچیده‌تر شود، بهینه‌تر است. البته که چنین نیست و پارامتر مهمی که در این نمودار نمایش داده نمی‌شود، bias مدل است که البته در هنگام رسم منحنی یادگیری با آن مواجه خواهیم شد. بنابراین این نتیجه‌گیری اشتباه است.

اما یک نتیجه مهم می‌توان از این نمودار گرفت و آن اینست که تعداد درختان باید از تقریباُ ۲۰ عدد بیشتر باشد تا مدل under fit نکند. تعیین دقیق‌تر نقطه بهینه کاری است که در بخش بعدی انجام می‌دهیم. بنابراین در اینجا مدل را آموزش نخواهیم داد و منتظر می‌مانیم تا پارامترهای بهینه توسط الگوریتم‌های جستجو تعیین شوند.

٤.١.٢ مدل SVM_LINEAR:

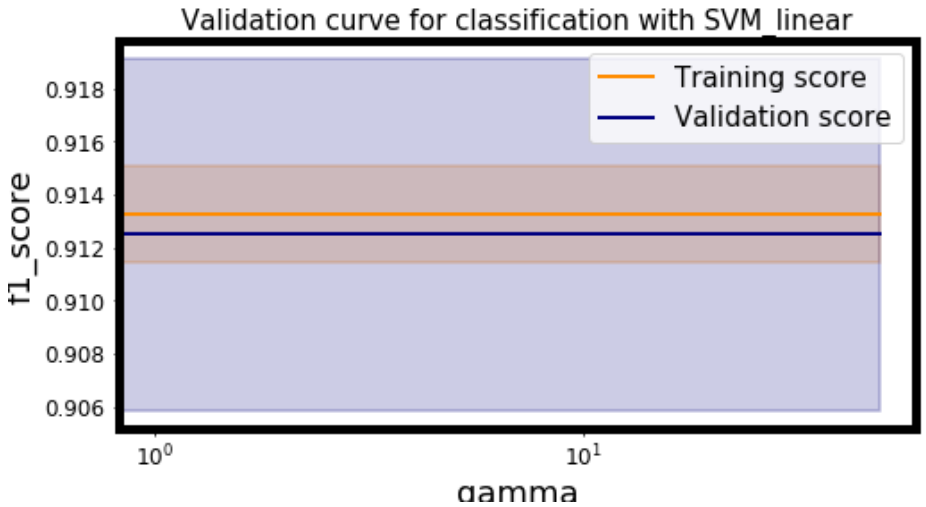
در این مدل دو پارامتر موثر وجود دارد و بار سم جداگاه این پارامتر ها به بهینه سازی این دو پارامتر میپردازیم.

منحنی اعتبار سنجی برای پارامتر C:



این منحنی برای بازه ۰ تا ۱۰۰ رسم شده است و با توجه به شکل نمایان میشود که با افزایش C میزان score تفاوت زیادی نمیکند به همین علت میتوان نتیجه گرفت این منحنی نقش بیار بسزایی در مدل ما ندارد اما در ادامه با استفاده از روش gridserch مقدار بهینه این پارامتر رار نیز محاسبه خواهیم کرد ولی با توجه به نمودار به مقدار دقیقی برای بهینه سازی دست پیدا کردیم.

منحنی اعتبار سنجی برای پارامتر گاما :



این منحنی برای بازه ۰ تا ۲۰ رسم شده ایت و با توجه شکل منحنی میتوان دریافت که پارامتر گاما را نمیتوان به عنوان پارامتر موثر برای این مدل معرفی کرد ولی کماکان مقدار دقیق بهینه شده این پارامتر را اندازه گیری خواهیم کرد.

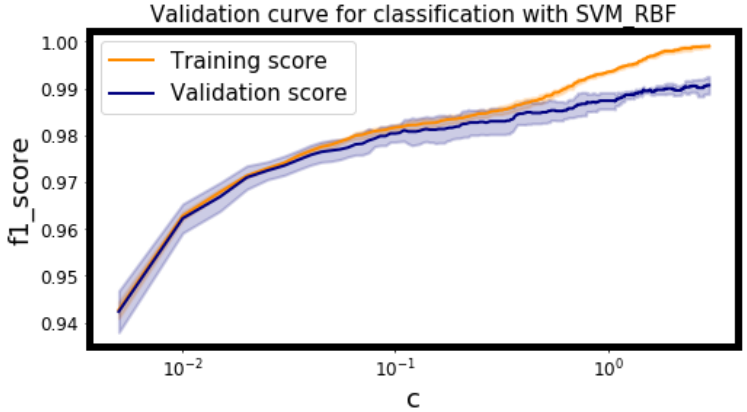
۵.۱.۲ مدل SVM_poly (چند جمله ای مرتبه ۹):

در این مدل دو پارامتر موثر وجود دارد اما امکان رسم منحنی اعتبار سنجی در این مدل وجود ندارد به همین علت برای بهینه کردن پارامتر ها از روش های قسمت بعد استفاده خواهیم کرد.

۶.۱.۲ مدل SVM_RBF:

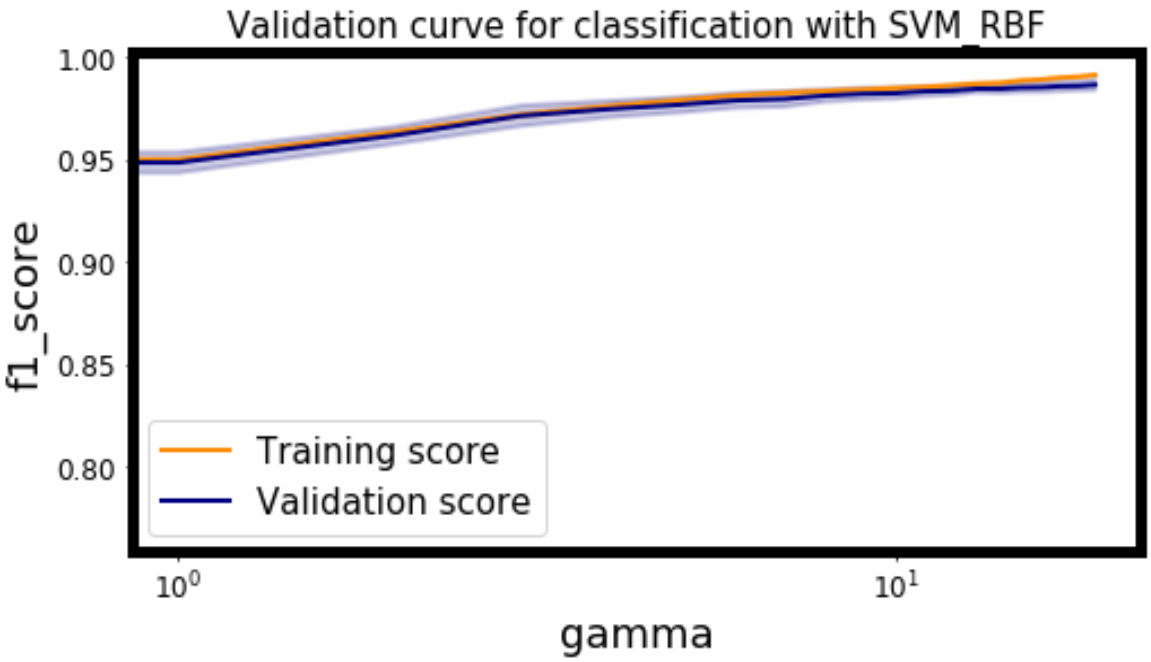
در این مدل دو پارامتر موثر وجود دارد و بار سم جداگانه این پارامترها به بهینه سازی این دو پارامتر می‌پردازیم.

منحنی اعتبار سنجی برای پارامتر C:



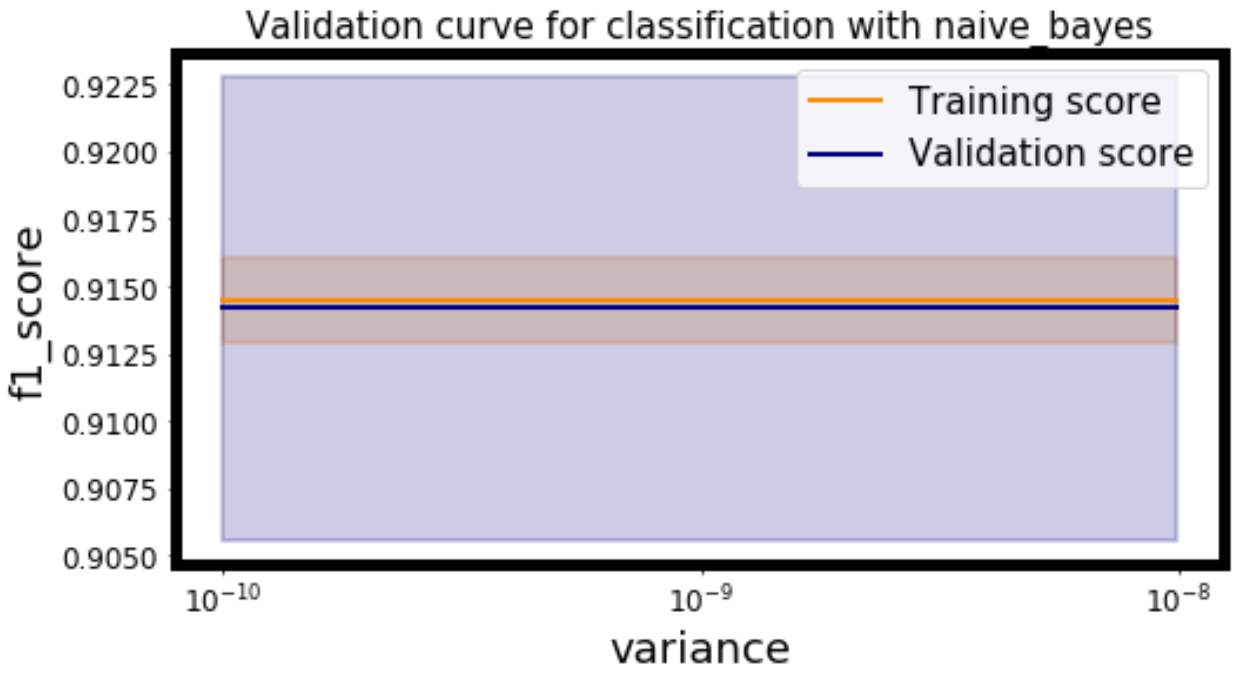
این منحنی برای بازه ۰ تا ۵ رسم شده است و هرچه به سمت جلو می‌رویم مدل پیچیده‌تر می‌شود. یعنی هرچه C بیشتر می‌شود میزان پیچیدگی نیز افزایش میابد به همین علت مقدار بهینه C را میتوان تعیین کرد ولی مقدار دقیق تر آن را در قسمت بعد با استفاده از gridsearch به دست خواهیم آورد.

منحنی اعتبار سنجی برای پارامتر گاما :



این منحنی برای بازه ۰ تا ۱۰۰ رسم شده است و با توجه به شکل نمایان میشود که با افزایش C میزان score تفاوت زیادی نمیکند به همین علت میتوان نتیجه گرفت این منحنی نقش بیار بسزایی در مدل ما ندارد اما در ادامه با استفاده از روش gridserch مقدار بهینه این پارامتر رار نیز محاسبه خواهیم کرد ولی با توجه به نمودار به مقدار دقیقی برای بهینه سازی دست پیدا کردیم.

۷.۱.۲ مدل naïve bayes:



تنها پارامتر این مدل var smoothing بود که با رسم منحنی اعتبارسنجی می‌بینیم تغییر آن تاثیر چندانی روی f1 score نمی‌گذارد.

برای بقیه مدل های منتخب امکان رسم منحنی اعتبار سنجی وجود ندارد.

۲.۲ یافتن پارامتر بهینه توسط الگوریتم‌های جستجوی شبکه (Grid search) و جستجوی تصادفی (Random search)

در این بخش یافتن پارامترهای بهینه را به این دو الگوریتم می‌سپاریم تا فرآیند بهینه کردن را دقیق‌تر انجام داده باشیم. همچنین مزیت استفاده از این الگوریتم‌ها در اینست که می‌تواند همزمان پارامترهای مدل را بهینه کند، حال آنکه برای رسم نمودار اعتبارسنجی برای مدل‌های چند پارامتری این مشکل را داشتیم. در نهایت نتایج ناشی از این الگوریتم‌ها را با نتایج حاصل از رسم منحنی اعتبار سنجی مقایسه می‌کنیم.

۱.۲.۲ مدل k Nearest Neighbors:

از منحنی اعتبار سنجی این مدل فهمیدیم که در تعداد همسایه ۳ و یا ۴ مدل ما بهینه خواهد بود. حال یک محدوده از این تعداد همسایه‌ها را به الگوریتم GridSearchCV کتابخانه sklearn پایتون می‌دهیم تا نتیجه حاصل از این الگوریتم را با بخش قبل مقایسه کنیم. اینکار را برای محدوده تعداد همسایه از ۱ تا ۱۰ انجام دادیم و نتیجه حاصل برابر شد با:

k Nearest Neighbors				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
n_neighbors	[1,10]	accuracy	48 s	{'n_neighbors': 1}

از جدول بالا می‌فهمیم که الگوریتم جستجوی شبکه‌ای تعداد همسایه ۱ را به عنوان پارامتر بهینه مدل پیشنهاد می‌کند. این حالت پیچیده‌ترین حالت این مدل است و همین نشان می‌دهد که مدل برای داده‌های ما و مسئله ما مناسب نیست. از آن گذشته می‌دانیم که در این حالت مدل ما over fit می‌کند و این مطلب را از رسم منحنی اعتبارسنجی مدل در بخش قبل متوجه شدیم.

به عنوان جمع بندی، پارامتر بهینه مدل را با توجه به هر دو روش بهینه‌سازی همان تعداد همسایه ۳ در نظر می‌گیریم.

۲.۲.۲ مدل درخت تصمیم (Decision Tree):

برای این مدل هم با همان تک پارامتر عمق درخت کار الگوریتم را آغاز می‌کنیم. از رسم نمودار اعتبارسنجی این مدل فهمیدیم که پارامتر بهینه مدل برای عمق درخت ۷ و یا ۸ است، بنابراین محدوده پارمتر ورودی الگوریتم جستجوی شبکه‌ای را در نزدیکی این پارامترها انتخاب می‌کنیم. نتایج الگوریتم به شرح زیر است:

Decision Tree				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
max_depth	[1,19]	accuracy	2min 14s	{'max_depth': 19}

نتیجه‌ای الگوریتم به ما به عنوان پارامتر بهینه مدل داده است عمق درخت ۱۹ است و ما از نمودار اعتبار سنجی می‌دانیم که در این حالت مدل over fit می‌کند.

در نهایت با توجه به نتیجه الگوریتم جستجوی شبکه‌ای و نتایج مربوط به نمودار اعتبارسنجی، پارامتر بهینه مدل را عمق درخت ۸ در نظر می‌گیریم.

۳.۲.۲ مدل جنگل تصادفی (Random Forest)

برای این مدل ما منحنی‌های اعتبار سنجی برای هر یک از پارامترها رسم کردیم و تنها نتیجه‌ای که گرفتیم این بود که برای تعداد درخت بالای ۲۰ مدل بهینه خواهد بود چراکه برای قبل آن مدل under fit می‌کند. همچنین پارامتر عمق هر درخت هم که در آنالیز نموداری نقشی بازی نکرد. حال قصد داریم هر دو پارامتر را با هم توسط الگوریتم‌های جستجو، بهینه کنیم:

Random Forest				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
max_depth	[1,40]	accuracy	1min 33s	{'max_depth': 30}
n_estimators	[1,40]			{'n_estimators': 27}

در این مدل به علت داشتن طیف وسیعی از پارامترها، ما عملیات بهینه کردن را توسط الگوریتم جستجوی تصادفی انجام دادیم و نه جستجوی شبکه‌ای. علت هم صرف زمان بسیار زیادی است که در الگوریتم جستجوی شبکه‌ای وجود دارد.

هر دو پارامتر بهینه شده‌اند و نتایج بدست آمده با منحنی‌های اعتبارسنجی منافاتی ندارند. حال مدل را با این پارامترها آموزش می‌دهیم و نتایج را بررسی می‌کنیم. نتایج بدست آمده در جدول زیر آورده شده است.

<table>
<tr><th colspan="6">Random Forest (max_depth=30, n_estimators=27)</th></tr>
<tr><td>accuracy</td><td colspan="5">0.994</td></tr>
<tr><td>Confusion matrix</td><td colspan="5">Report</td></tr>
<tr><td rowspan="3">$\begin{pmatrix} 2109 & 7 \\ 1 & 3536 \end{pmatrix}$</td><td>label</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr>
<tr><td>0</td><td>1.00</td><td>0.99</td><td>0.99</td><td>434</td></tr>
<tr><td>1</td><td>0.99</td><td>1.00</td><td>0.99</td><td>697</td></tr>
</table>

از جدول بالا پیداست که مدل به خوبی برای مسئله ما و داده‌های ما کار کرده است. از ماتریس درهم آمیختگی بدست آمده پیداست که مدل به خوبی توانایی پیش‌بینی دارد و عناصر غیر قطری ماتریس، یعنی false-positive و false-negative دارای جمعیت بسیار پایینی هستند و همین باعث شده که مترهای precision و recall برای هر دو داده درهم‌تنیده و جداپذیر مقدار خوبی بگیرند. از طرفی همین ویژگی باعث بالا رفتن مقدار f1-score و به تبع آن accuracy شده است. در مجموع این مدل تا به اینجای کار برای مسئله ما بسیار مناسب نشان داده شده است. نظر نهایی بعد از رسم منحنی یادگیری مشخص می‌شود.

٤.٢.٢ مدل SVM_Linear:

با بهینه کردن دو پارامتر c ,gamma که در قسمت های قبل مورد بررسی قرار گرفته بود حال با استفاده از gridsearch مقادیر بهینه این پارامتر ها را اندازه گیری کرده و مدل را مجددا با این پارامتر های بهینه آموزش خواهیم داد.

SVM_linear				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
c	[0.01,0.04]	F1_score=0.76	4.8min	{'c': 0.015}
gamma	[0,20]			{'gamma': 11}

با توجه به پارامترها و آموزش دوباره مدل داریم :

SVM_linear(c = 0.015, gamma =11)					
accuracy	0.636				
Confusion matrix	**Report**				
$\begin{pmatrix} 0 & 2116 \\ 0 & 3537 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.0	0.0	0.0	412
	1	0.64	1.00	0.78	719

در این مدل حتی بعد از بهینه شدن تغییر زیادی در نتایج حاصل ایجاد نشد و این نشان دهنده آن است که این مدل برای مسئله ما مدل مناسبی نیست.

۵.۲.۲ مدل SVM_poly (مرتبه ۹):

پارامتر های بهینه شده C, gamma را در جدول زیر میتوان مشاهده کرد:

SVM_poly(degree = 9)				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
c	[0.01,0.04]	F1_score=0.76	2.1 min	{'c': 0.015}
gamma	[0,20]			{'gamma': 11}

حال با توجه به مقداری بهینه شده دوباره مدل را آموزش دادیم:

SVM_poly(c = 0.015, gamma =11,degree =9)					
accuracy	0.635				
Confusion matrix	**Report**				
$\begin{pmatrix} 0 & 2116 \\ 0 & 3537 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.0	0.0	0.0	412
	1	0.64	1.00	0.78	719

در این مدل نیز حتی بعد از بهینه سازی نتایج مطلوب کسب نشد به همین علت این مدل مناسب نیست.

۶.۲.۲ مدل SVM_RBF:

پارامتر های بهینه شده C, gamma را در جدول زیر میتوان مشاهده کرد:

SVM_RBF()				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
c	[0.01,0.04]	F1_score=0.76	2.1 min	{'c': 1}
gamma	[0,100]			{'gamma': 14}

با استفاده از این پارامتر های جدید مدل را آموزش میدهیم :

SVM_RBF(c= 1, gamma =14)					
accuracy	0.984				
Confusion matrix	**Report**				
$\begin{pmatrix} 2035 & 81 \\ 11 & 3526 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.99	0.97	0.98	412
	1	0.98	0.98	0.99	719

از جدول بالا پیداست که مدل به خوبی برای مسئله ما و داده‌های ما کار کرده است. از ماتریس درهم آمیختگی بدست آمده پیداست که مدل به خوبی توانایی پیش‌بینی دارد و عناصر غیر قطری ماتریس، یعنی false-positive و false-negative دارای جمعیت بسیار پایینی هستند و همین باعث شده که مترهای precision و recall برای هر دو داده درهم‌تنیده و جداپذیر مقدار خوبی بگیرند. از طرفی همین ویژگی

باعث بالا رفتن مقدار f1-score و به تبع آن accuracy شده است. در مجموع این مدل تا به اینجای کار برای مسئله ما بسیار مناسب نشان داده شده است. نظر نهایی بعد از رسم منحنی یادگیری مشخص می‌شود.

۷.۲.۲ مدل naive bayes:

پارامتر های این مدل در ابتدا توسط grtparam مقداردهی شده بودند و نیازی به grid search این مدل نبود و با قرار دادن پارامتر های داده شده مدل را دوباره آموزش دادیم و به مقادیر زیر رسیدیم:

گزارش کلی مدل:

<table>
<tr><th colspan="6">naive_bayes</th></tr>
<tr><td>accuracy</td><td colspan="5">0.887</td></tr>
<tr><td>Confusion matrix</td><td colspan="5">Report</td></tr>
<tr><td rowspan="3">$\begin{pmatrix} 2038 & 78 \\ 498 & 3039 \end{pmatrix}$</td><td>label</td><td>precision</td><td>recall</td><td>f1-score</td><td>support</td></tr>
<tr><td>0</td><td>0.78</td><td>0.97</td><td>0.86</td><td>412</td></tr>
<tr><td>1</td><td>0.98</td><td>0.84</td><td>0.90</td><td>719</td></tr>
</table>

از جدول بالا پیداست که مدل به خوبی برای مسئله ما و داده‌های ما کار کرده است. از ماتریس درهم آمیختگی بدست آمده پیداست که مدل به خوبی توانایی پیش‌بینی دارد و عناصر غیر قطری ماتریس، یعنی false-positive و false-negative دارای جمعیت بسیار پایینی هستند و همین باعث شده که مترهای precision و recall برای هر دو داده درهم‌تنیده و جداپذیر مقدار خوبی بگیرند. از طرفی همین ویژگی باعث بالا رفتن مقدار f1-score و به تبع آن accuracy شده است. در مجموع این مدل تا به اینجای کار برای مسئله ما بسیار مناسب نشان داده شده است. نظر نهایی بعد از رسم منحنی یادگیری مشخص می‌شود.

۸.۲.۲ مدل AdaBoost:

AdaBoost				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
Base_estimator	-	F1_score	6min 50s	DecisionTreeClassifier (max_depth=2)
learning_rate	[-1,1]			1
n_estimators	[50,100]			100

بعد از بهینه سازی مدل را آموزش مجدد دادیم و به نتایج زیر رسیدیم:

AdaBoost					
accuracy	0.994				
Confusion matrix	Report				
$\begin{pmatrix} 2116 & 0 \\ 6 & 3531 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.99	1.00	0.99	412
	1	1.00	0.99	1.00	719

از جدول بالا پیداست که مدل به خوبی برای مسئله ما و داده‌های ما کار کرده است. از ماتریس درهم آمیختگی

بدست آمده پیداست که مدل به خوبی توانایی پیش‌بینی دارد و عناصر غیر قطری ماتریس، یعنی false-positive و false-negative دارای جمعیت بسیار پایینی هستند و همین باعث شده که مترهای precision و recall برای هر دو داده درهم‌تنیده و جداپذیر مقدار خوبی بگیرند. از طرفی همین ویژگی باعث بالا رفتن مقدار f1-score و به تبع آن accuracy شده است. در مجموع این مدل تا به اینجای کار برای مسئله ما بسیار مناسب نشان داده شده است. نظر نهایی بعد از رسم منحنی یادگیری مشخص می‌شود.

۹.۲.۲ مدل QDA:

QDA				
Algorithm	GridSearchCV			
پارامتر	**محدوده پارامتر**	**متر**	**زمان صرف شده**	**نتیجه**
reg_param	[0.1, 0.5]	F1_score	3.56 s	0.1

بعد از آموزش مجدد مدل با استفاده از پارامتر بهینه شده بالا به مقادیر زیر دست پیدا کردیم.

QDA					
accuracy	**0.635**				
Confusion matrix	Report				
$\begin{pmatrix} 0 & 2116 \\ 0 & 3537 \end{pmatrix}$	label	precision	recall	f1-score	support
	0	0.0	0.0	0.0	412
	1	0.64	1	0.78	719

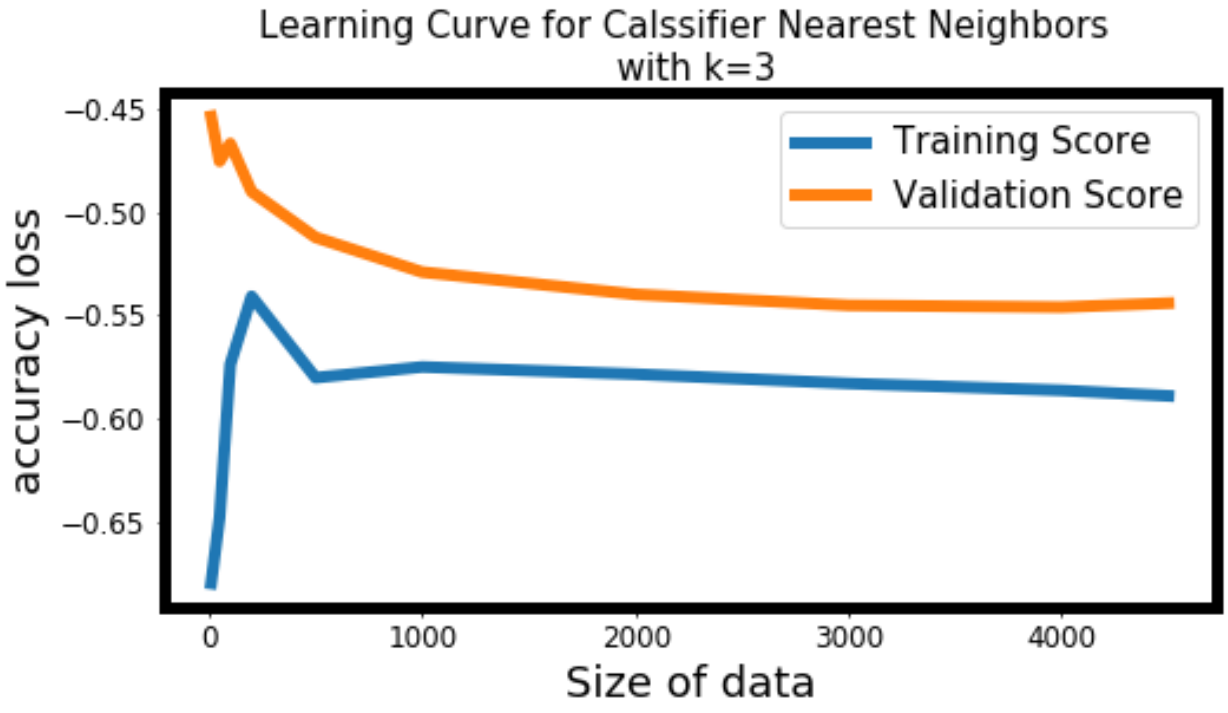
با توجه به مقادیر به دست آمده بعد از اعمال پارامتر های بهینه و چک کردن با مقادیر به دست آمده قبل از بهینه سازی میتوان دریافت که تغییری حاصل نشده و هیچ پیشرفتی وجود نداشته است به همین علت با توجه به مقدار بسیار کم accuracy میتوان به مناسب نبودن این مدل برای این مسئله پی برد .

۳.۲ رسم منحنی یادگیری (Learning curve) و بررسی مدل و کافی بودن میزان داده

در این بخش از اصول یادگیری آماری (Statistical learning) برای بررسی مدل‌ها و میزان کافی بودن داده‌ها می‌پردازیم. در این میان از کمیت‌های bias و variance برای این بررسی‌ها استفاده خواهیم کرد.

۱.۳.۲ مدل k Nearest Neighbors:

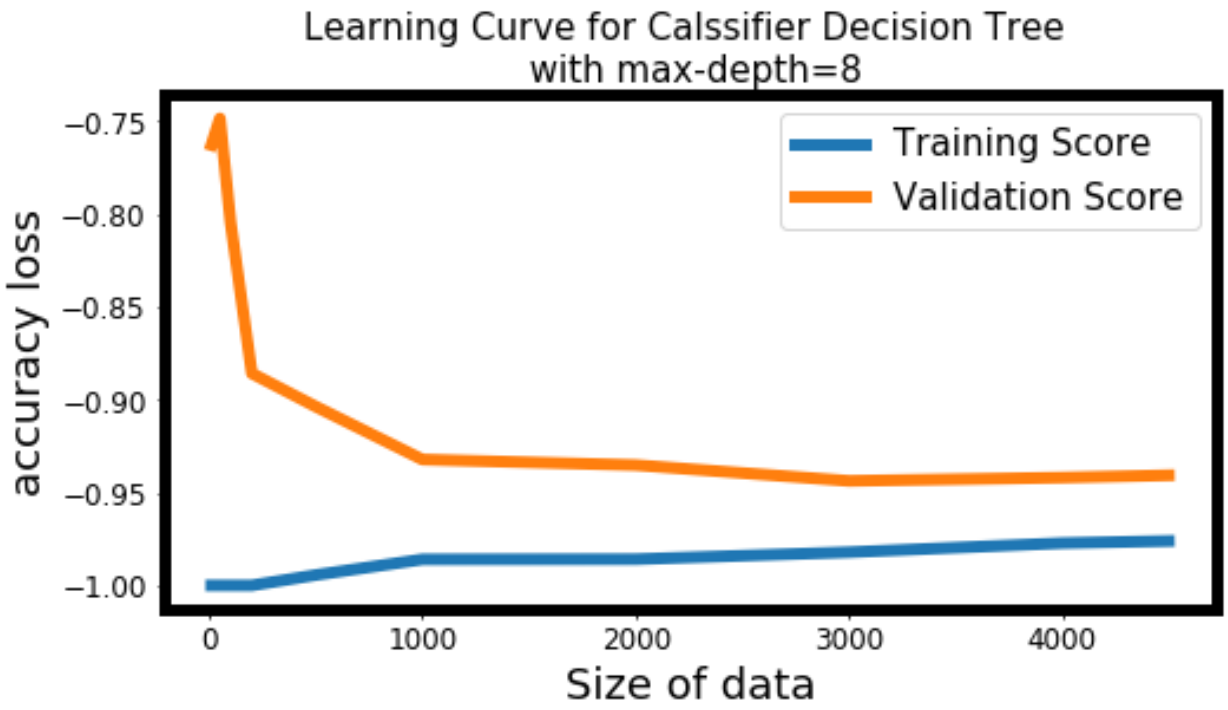
برای همان پارامتر بهینه‌ای که در بخش قبل بدست آوردیم (k=3) منحنی یادگیری را رسم کرده‌ایم:



از این نمودار می‌فهمیم که مقدار bias مدل ما نسبتاً بالاست و این ناشی از پیچیدگی مدل است. همچنین variance مدل ما برای کل داده ورودی هم مقداری غیر صفر دارد و این ناشی از این است که میزان داده ما برای این مدل کافی نبوده است. البته حتی اگر میزان داده را آنقدر زیاد کنیم که variance کم شود بر bias مدل نمی‌توان غلبه کرد، چراکه اگر بخواهیم پیچیدگی مدل را کم کنیم، score مدل پایین خواهد آمد. در مجموع این مدل برای داده‌های ما خوب کار نکرد.

۲.۳.۲ مدل درخت تصمیم (Decision Tree):

در بخش قبل تعیین کردیم که پارامتر بهینه مدل درخت تصمیم، عمق درخت برابر با ۸ است. حال برای این مدل که نسبتاُ accuracy خوبی دارد، منحنی یادگیری را رسم می‌کنیم تا بازهم این مدل و همچنین میزان داده را برای این مدل بررسی کنیم. نتیجه رسم منحنی یادگیری برای این مدل به صورت زیر درآمد:

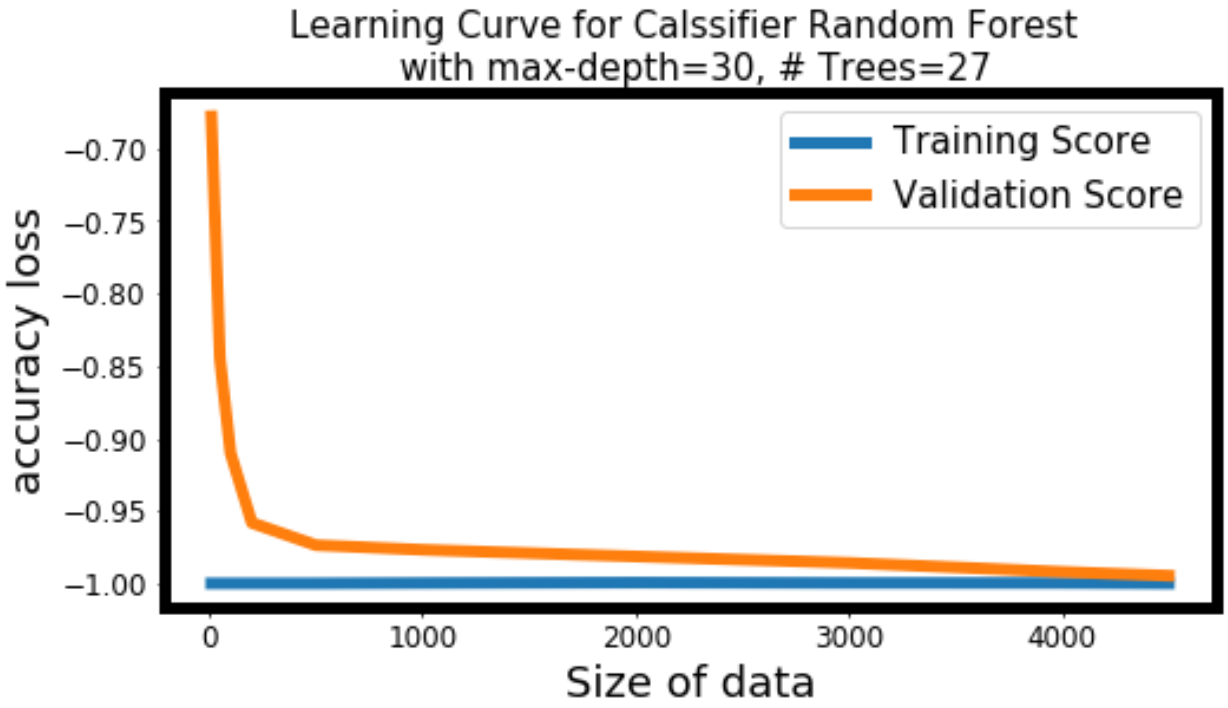


از این نمودار واضح است که مقدار bias بسیار کم است و این ناشی از بهینه بودن پارامترهای مدل و over fit نکردن مدل است. اما این نمودار نشان می‌دهد که variance مقداری غیر صفر و قابل توجه است. این

ناشی از ناکافی بودن میزان داده است. بنابراین در این مدل برای بهتر شدن وضعیت به داده‌های بیشتری نیاز داریم.

۳.۳.۲ مدل جنگل تصادفی (Random Forest):

پارامترهای بهینه این مدل در بخش قبل به وسیله منحنی اعتبارسنجی و همچنین الگوریتم random search تعیین شدند. این پارامترها عبارتند از تعداد درخت ۲۷ و حداکثر عمق هر درخت ۳۰. گزارش کلی مدل نشان داد که این مدل برای داده‌های ما خوب کار کرده است. حال یکبار دیگر این مدل را توسط منحنی یادگیری مورد آزمایش قرار می‌دهیم تا ابعاد دیگر این مدل را مورد بررسی قرار دهیم. منحنی یادگیری این مدل به صورت زیر درآمد:

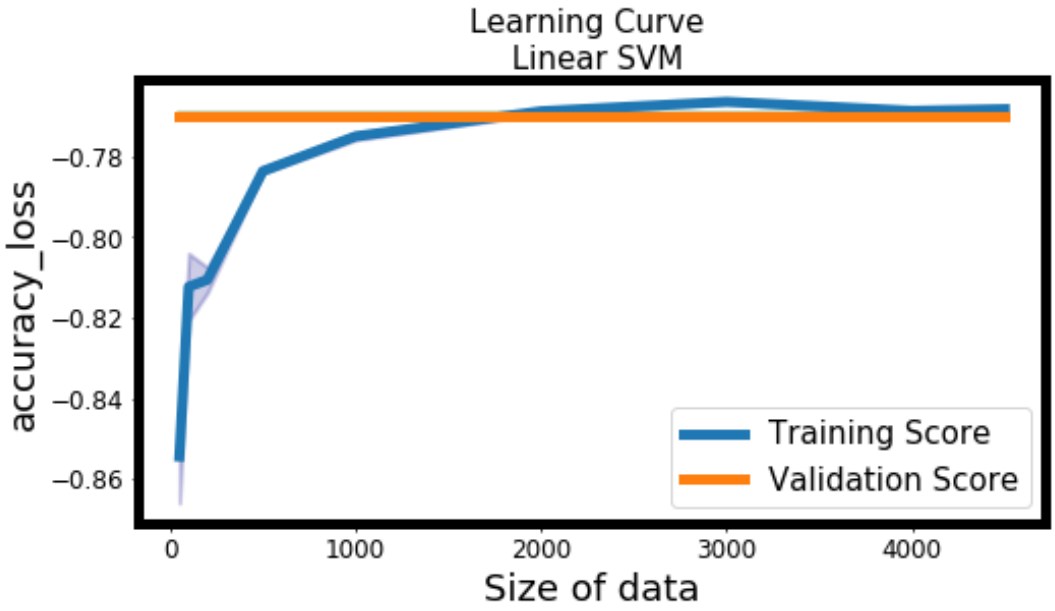


نمودار به خوبی نشان می‌دهد که bias مدل به شدت کم است و این ناشی از بهینه بودن پارامترها و دوری از over fitiing است. همچنین برای همین تعداد داده مقدار variance مدل نیز تقریباً صفر است و هر دو نمودار به هم میل کرده‌اند.

بنابراین درنهایت و پس از بررسی این مدل از سه طریق (منحنی اعتبارسنجی و منحنی یادگیری و بهینه کردن از طریق جستجوی تصادفی)، نتیجه می‌گیریم که این مدل برای مسئله ما و داده‌های ما بسیار مناسب است و نتایج خوبی داده است.

٤.٣.٢ مدل SVM_LINEAR:

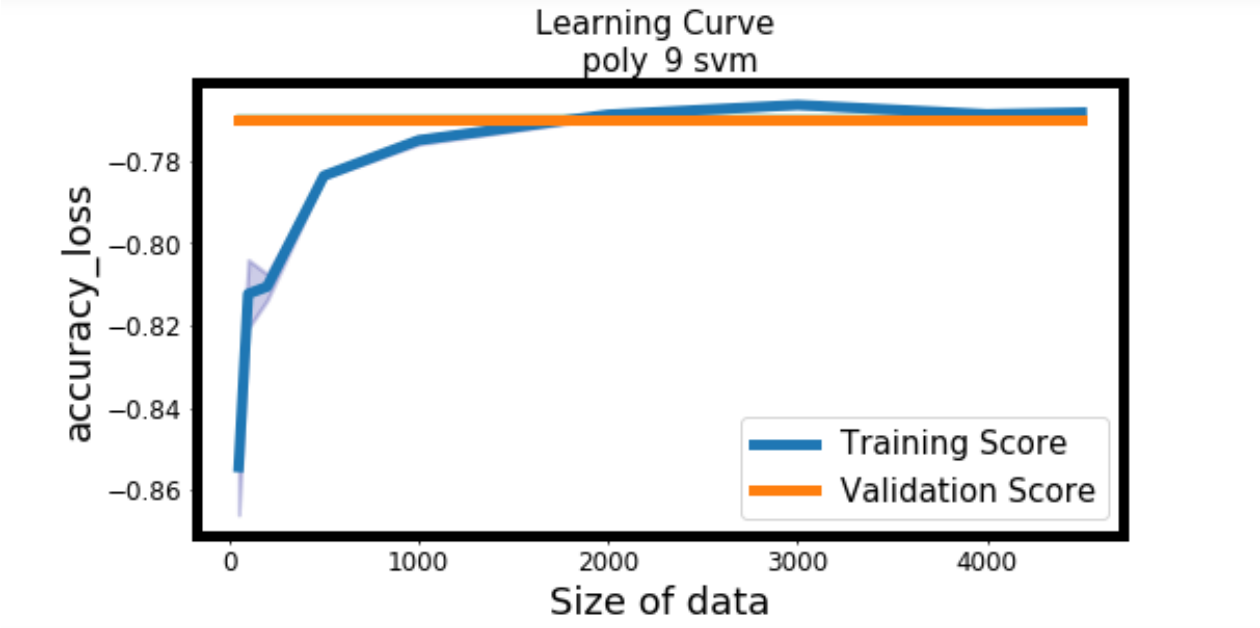
با توجه به پارمتر های بهینه شده توسط gridsearch برای C=0.015 و گاما برابر ۱۱ منحنی آموزش زیر را رسم کرده ایم:



در این منحنی مقادیر واریانس نیز نمایش داده شده است و با توجه به این منحنی میتوان دریافت که مقادیر داده های داده شده به مدل کافی بوده است و مقدار بایاس برای این مدل برابر ۰.۷۶ – میباشد و این ناشی از پیچیدگی مدل است.

بنابراین درنهایت و پس از بررسی این مدل از سه طریق (منحنی اعتبارسنجی و منحنی یادگیری)، نتیجه می‌گیریم که این مدل برای مسئله ما و داده‌های ما مناسب نیست.

۵.۳.۲ مدل SVM_poly (مرتبه ۹):



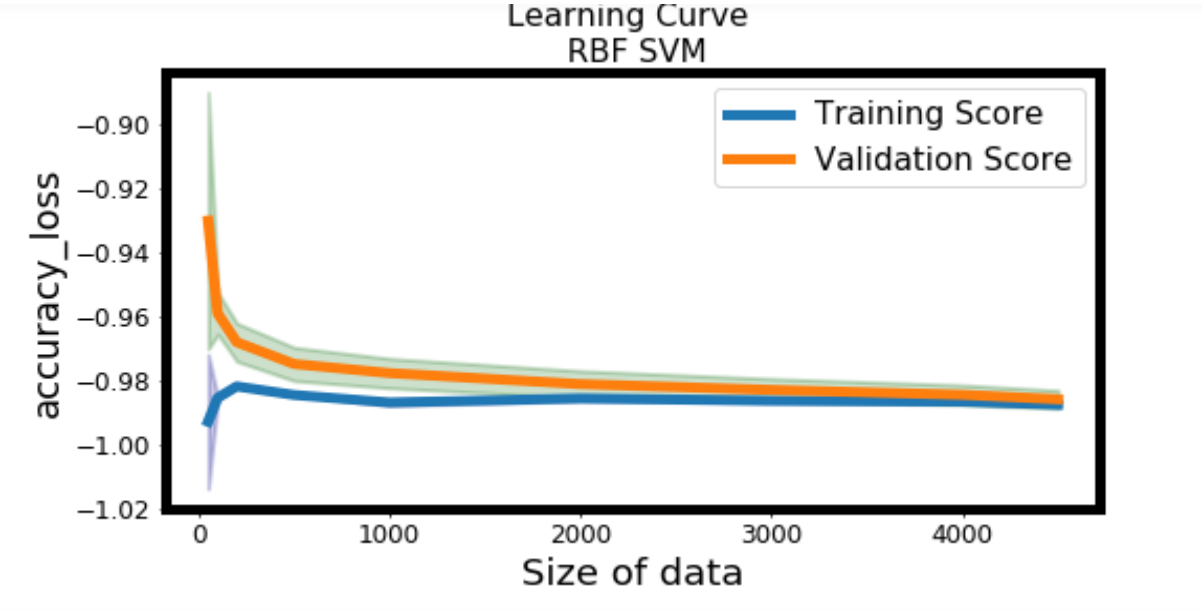
بعد از آموزش مجدد با پامتر های بهینه منحنی آموزش بالا رسم شده است در این منحنی مقادیر واریانس نیز نمایش داده شده است که به صفر میل میکند و با توجه به این منحنی میتوان دریافت که مقادیر داده های داده شده به مدل کافی بوده است و مقدار بایاس برای این مدل برابر ۰.۷۶ – میباشد و این ناشی از پیچیدگی مدل است.

بنابراین درنهایت و پس از بررسی این مدل از طریق (منحنی اعتبارسنجی و منحنی یادگیری)، نتیجه می‌گیریم که این مدل برای مسئله ما و داده‌های ما مناسب نیست.

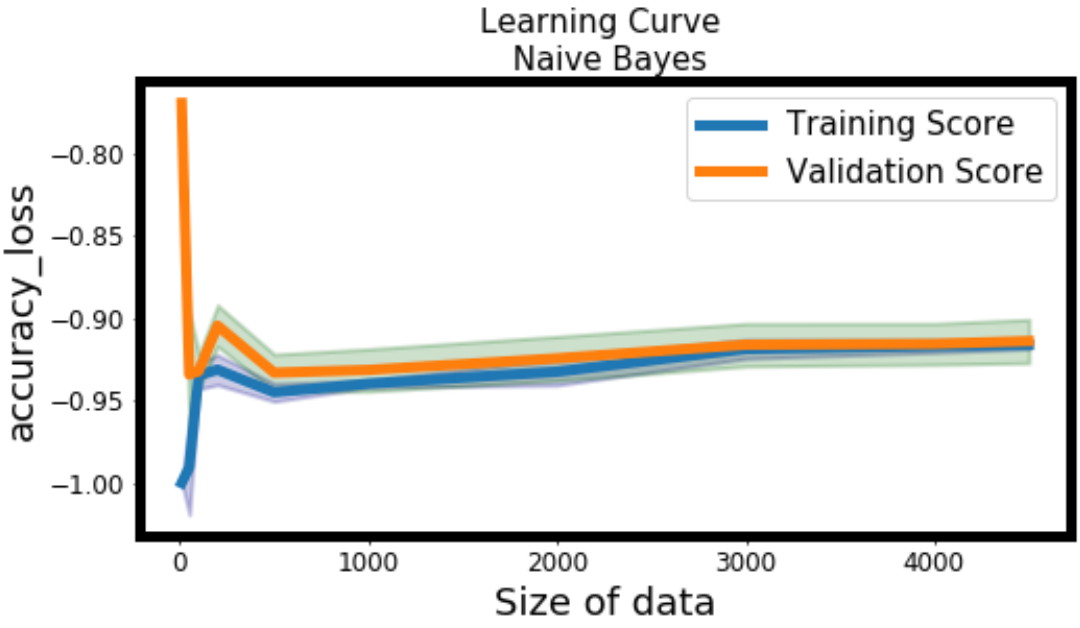
٦.۳.۲ مدل SVM_RBF:

بعد از آموزش مجدد با پامتر های بهینه منحنی آموزش پایین رسم شده است در این منحنی مقادیر واریانس نیز نمایش داده شده است که به صفر میل میکند و با توجه به این منحنی میتوان دریافت که مقادیر داده های داده شده به مدل کافی بوده است و مقدار بایاس برای این مدل برابر ۰.۹۸ – میباشد نمودار به خوبی نشان می‌دهد که bias مدل به شدت کم است و این ناشی از بهینه بودن پارامترها و دوری از over fitiing است.

بنابراین درنهایت و پس از بررسی این مدل از طریق (منحنی اعتبارسنجی و منحنی یادگیری)، نتیجه می‌گیریم که این مدل برای مسئله ما و داده‌های ما مناسب نیست.



۷.۳.۲ مدل naive bayes:

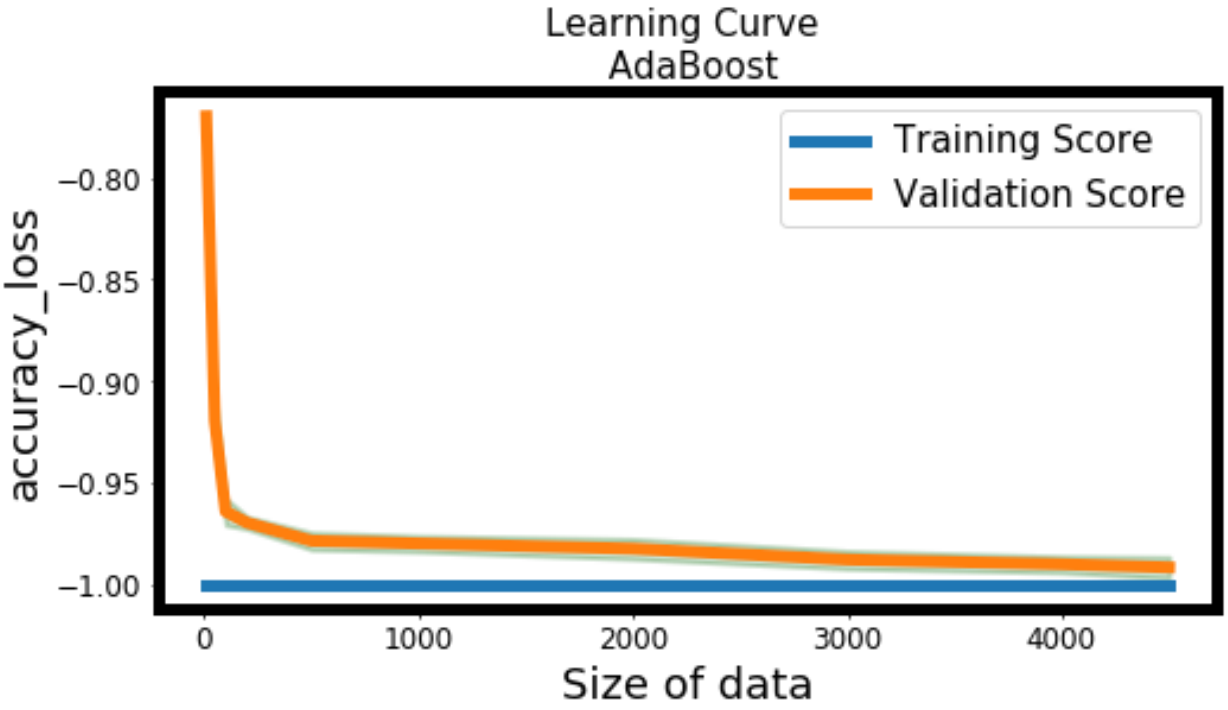


نمودار به خوبی نشان می‌دهد که bias مدل به کم است و این ناشی از بهینه بودن پارامترها و دوری از over fitiing است. همچنین برای همین تعداد داده مقدار variance مدل نیز تقریباً صفر است (روی شکل نمایش داده شده است) و هر دو نمودار به هم میل کرده‌اند که این نشن از کافی بودن مقادیر داده های داده شده به مدل دارد.

با توجه به بررسی ها در قسمت قبل (بعد از بهینه سازی) و با توجه به منحنی آموزش میتوان نتیجه گرفت این مدل برای استفاده در مسئله ما مناسب است.

۸.۳.۲ مدل AdaBoost:

در این مدل بایاس پایین می‌باشد و دو نمودار به یک خط میل کرده‌اند و در محدوده خطا یکدیگر قرار دارند. اما هنوز واریانس کاملا صفر نشده. با توجه به عملکرد خوب این مدل با کمی افزایش داده می‌توان نتیجه بهتری گرفت.



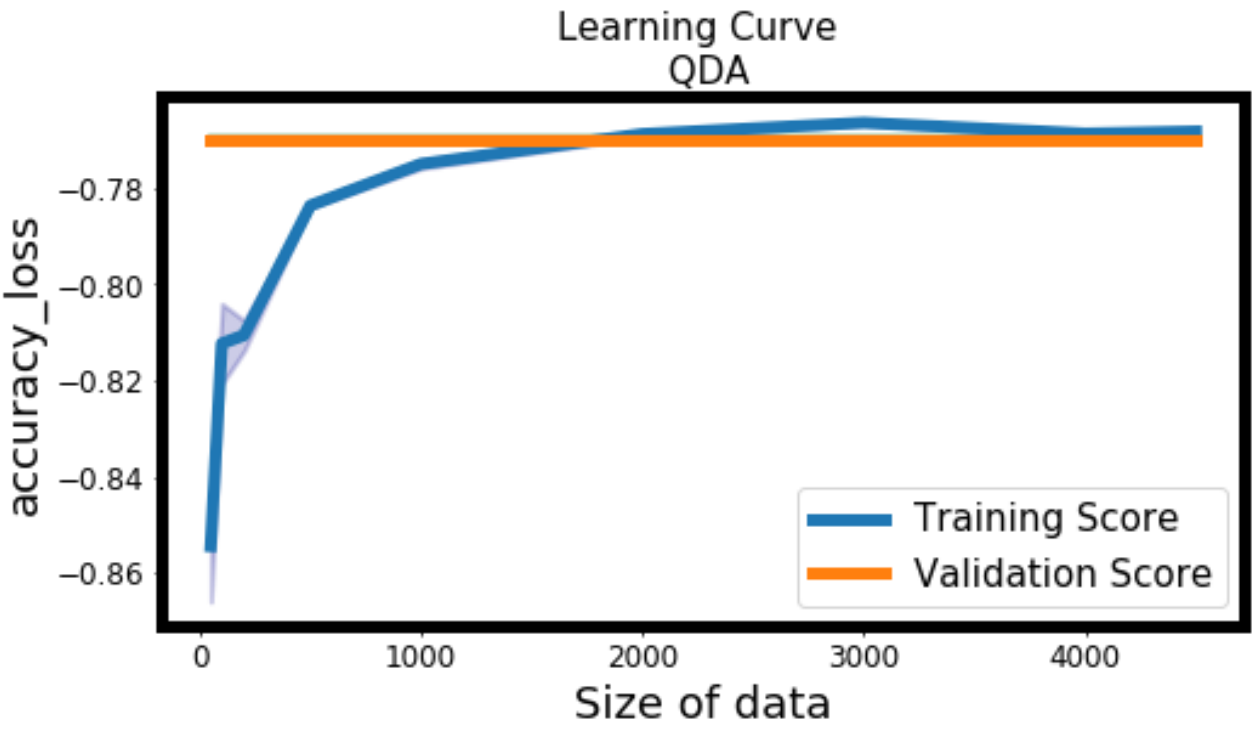
این مدل از ابتدا هم بسیار خوب عمل می‌کرد اما پس از بهینه سازی متریک‌ها نزدیک به ۱ را نشان می‌دهد.

در نتیجه این مدل برای استفاده در مسئله ما مناسب میباشد.

۹.۳.۲ مدل QDA:

در این منحنی مقادیر واریانس نیز نمایش داده شده استکه به سمت صفر میل میکند و با توجه به اینکه این در منحنی به یکدیگر میل میکنند میتوان دریافت که مقادیر داده های داده شده به مدل کافی بوده است و مقدار بایاس برای این مدل برابر ۰.۷۶ – میباشد و این ناشی از پیچیدگی مدل است.

بنابراین درنهایت و پس از بررسی این مدل از طریق (منحنی اعتبارسنجی و منحنی یادگیری)، نتیجه می‌گیریم که این مدل برای مسئله ما و داده‌های ما مناسب نیست.



نکته مهم: از آنجایی که در تمامی مدل رفتارهای hursh گونه در نمودارها و منحنی‌ها مشاهده نشد، عملیات regularization را انجام ندادیم.

۳ نتیجه گیری کلی و تهیه جدول برای مدل‌ها

در این بخش تمام نتایج بخش‌های قبلی را در قالب یک جدول عرضه می‌کنیم. به غیر از نتایج، زمان یادگیری مدل‌ها و پیش‌بینی مدل‌ها را به عنوان یک پارامتر اضافی، درج خواهیم کرد. جدول از قرار زیر است:

گزارش کلی			
مدل	زمان یادگیری	زمان پیش‌بینی	accuracy
Nearest Neighbors	0.11 s	3.99 s	0.555
Decision Tree	0.23 s	3 ms	0.938
Random Forest	0.54 s	31 ms	0.994
SVM_linear	7.42 s	1.15 s	0.635
SVM_RBF	1.39 s	0.13 s	0.984
SVM_POLY(9)	7.86 s	0.89 s	0.635
Ada Boost	21.4 s	0.20 s	0.994
Naïve Bayes	28 ms	6.0 ms	0.886
QDA	0.10 s	0.011 s	0.635

نقش اعضای گروه

- مهکامه سلیمی:

کار روی برچسب گذاری داده‌ها، کار روی مدل‌های Linear SVM و SVM-Poly، آماده سازی و تایپ گزارش، آماده سازی کدها

- سیده فاطمه دهقان:

کار روی برچسب گذاری داده‌ها، کار روی مدل‌های SVM-RBF و QDA، آماده سازی و تایپ گزارش

- سحر تغیر:

کار روی برچسب گذاری داده‌ها، کار روی مدل‌های Navie Bayes و Ada Boost ، آماده سازی و تایپ گزارش

- محمدمهدی ماستری فراهانی:

کار روی برچسب گذاری داده‌ها، کار روی مدل‌های Nearest neighbors و Dicision Tree و Random forest، آماده سازی و تایپ گزارش

مراجع

[1] A Separability-Entanglement Classifier via Machine Learning, Sirui Lu, Shilin Huang, Keren Li, Jun Li, Jianxin Chen, Dawei Lu, Zhengfeng Ji, Yi Shen, Duanlu Zhou, Bei Zeng

[2] Volume of the set of separable states, Karol Z yczkowski, Paweł Horodecki, Anna Sanpera and Maciej Lewenstein