فهرست مطالب

فصل ۱

مقدمه

امروزه با گسترش بیماری پارکینسون در جهان و با توجه به عدم وجود روش درمان قطعی برای این بیماری، اهمیت تشخیص به هنگام برای کنترل و جلوگیری از پیشرفت آن اهمیت یافته است. عموم روش‌های مورد استفاده به دلیل پیچیدگی‌های فرآیند، مستلزم صرف زمان و هزینه‌ی بسیار زیاد می‌باشند، به طوریکه بعضا در کشورهای جهان سوم عملا استفاده از این روش‌های تشخیص امکان پذیر نمی‌باشد، به همین دلیل اهمیت استفاده از روش‌های جایگزین نمایان گردید. امروزه با توجه پیشرفت‌های بدست آمده در زمینه‌ی یادگیری ماشین، عملا امکان تشخیص بیماری بر اساس تحلیل ویژگی‌های تکلمی افراد فراهم شده است. استفاده از این روش‌ها با توجه به پیشرفت‌های حاصل شده در بهبود دقت مدل‌های یادگیری، به صرفه بودن هزینه‌های مرتبط، سریع بودن فرآیند تشخیص را نتیجه می‌دهد. در این گزارش سعی شده است با استفاده از روش‌های موجود در زمینه‌ی یادگیری ماشین، استفاده از تحلیل‌های ریاضی و شهودی مناسب و همچنین به کارگیری روش‌های پیش‌پردازش و کاهش بعد، به صورت موثر عملکرد مدل‌ها با یکدیگر مورد بررسی و سنجش قرار بگیرند.

فصل ۲

خلاصه مقاله

۱.۲ مقدمه

در این فصل سعی در پاسخ به سوالات مطرح شده در صورت پروژه پیرو مطالعه‌ی یکی از مقالات پیشمهادی شده است. در این گزارش مقاله تحت عنوان "Novel Speech Signal Processing Algorithms for High-Accuracy Classification of Parkinson's Disease" مورد بررسی و تحلیل قرار گرفته است.

۲.۲ سوال ۱

بیماری‌های ناشی از اختلالات عصبی بر زندگی روزمره‌ی انسان‌ها در مقیاس جهانی تاثیرگذار است. یکی از اختلالات شایع در زمان حاضر، بیماری پارکینسون می‌باشد، به طوریکه تحلیل‌های آماری گسترش با سرعت این بیماری را تایید می‌نماید. از طرفی، باتوجه به سختی‌هایی که در امر تشخیص این بیماری وجود دارد، تحلیلگران بر این باور هستند که نرخ شیوع این بیماری چیزی بیش از آمارهای محاسبه شده می‌باشد. شیوع روز افزون، به همراه دشواری‌های موجود در تشخیص این بیماری دانشمندان را ترغیب به گسترش روش‌هایی مبتنی بر تحلیل داده برای ایجاد سیستم‌های پشتیبان تصمیم‌گیر [۱] کرده است. به طور عمده این بیماری با علائمی همچون لرزش در سراسر بدن، سفتی عضلات ، از دست رفتن قدرت کنترل ماهیچه‌ها و همچنین ضعف‌های شناختی همراه است. دانشمندان متوجه شده‌اند یکی از اولین و مهم‌ترین علائم در این بیماری افت قدرت تکلم است، به طوریکه علائم قدرت تکلم بعضا تا ۵ سال پیش از تشخیص بالینی بیماری نیز قابل تشخیص بوده است. در بسیاری از تحقیقات مرتبط سعی شده است با جمع آوری داده‌های مرتیط با تکلم و بررسی معیارهای مرتبط به تشخیص بیماری با استفاده از روش‌های پردازش سیگنال پرداخته بشود. برای مثال با بررسی تلفظ دنبال داره یک حرف و یا بررسی تکلم پیوسته و واضح [۲] می‌توان به دقت مناسبی در امر تشخیص دست یافت. برای اینکار به طور عمده دو معیار: ۱- توانایی در تلفظ صحیح (dysphonia) ۲- سختی در تلفظ کلمات (dysarthia) را در نظر گرفته و بر این اساس به تولید ویژگی‌های مختلف پرداخته می‌گردد. از آنجا که پردازش جملات و دیالوگ‌ها با سختی‌هایی همراه است

[1] Decision Support Systems

[2] Running Speech

مشاهده شده که تلفظ ادامه دار یک حرف صدا دار مانند "آ" می‌تواند ویژگی‌های موثری را در امر تحلیل‌های مورد نظر در اختیار قرار بدهد، سپس از این معیارها برای امر تشخیص و طبقه‌بندی به دو کلاس "سالم" و "ناسالم" می‌توان استفاده کرد. برای مثال در تحلیل‌های قدیمی مبتنی بر بررسی نسبت نویز به سیگنال (SNR)
اغتشاش در انداه سیگنال (Shimmer measures) ، اغتشاش در فرکانس سیگنال (Jitter measures) و... دقتی در حدود 0.9 را نتیجه می‌دهند. در دسته‌ای از تحقیقات گذشته نشان داده شد که با استفاده از روش های رگرسیون(بعضا مبتنی بر روش‌های یادگیری ماشین [۳]) می‌توان به تخمین میزان شدت بیماری(شدت بر مبنای مترهای بالینی و پزشکلی) بر اساس ویژگی‌های بدست آمده از تکلم افراد پرداخت. این رگرسیون‌ها می‌توانند غیرخطی و برمبنای روش‌های جدید باشند. ایده اصلی مقاله جاری آن است که با استفاده از ویژگی‌های جدید تعریف شده در مقالات گذشته، به طبقه‌بندی و تشخیص فرد بیماری و سالم پرداخته بشود.

۳.۲ سوال ۲

در این مقاله از مجموعه دادگان در پایگاه داده National Center for Voice and Speech استفاده شده است، که شامل ۲۶۳ تلفظ مصوت از ۴۳ شخص می‌باشد. از بین این افراد ۱۷ زن و ۲۶ مرد حضور داشتند و ۱۰ نفر آن‌ها سالم و ۳۳ نفر دیگر دارای بیماری پارکینسون می‌باشند. از بین ۱۰ فرد سالم تعداد ۶۱ تلفظ مصوت سالم اخذ شده و از هرکدام از ۳۳ فرد ناسالم تعداد ۶-۷ تلفظ "آ" در نظر گرفته شد و در مجموع ۲۰۲ تلفظ ناسالم در اندازه و فرکانس مناسب برای هر فرد ضبط می‌شود. تمامی این داده‌ها توسط یک میکروفن AKG C420 نصب شده بر روی سر، با فاصله ۸ سانتی‌متر از دهان با سرعت نمونه‌برداری $44.1kHz$ و با دقت ۱۶ بیت ضبط می‌شود.

۴.۲ سوال۳

برمبنای آنچه در بخش قبل ذکر شد، این داده‌ها توسط National Center for Voice and Speech تهیه شده بود، اما محاسبه و استخراج ویژگی‌ها برمبنای کار نویسندگان مقاله بوده است.

۵.۲ سوال۴

۱.۵.۲ استخراج ویژگی‌ها

همانطور که در بخش ۱ مطرح شد عمده‌ی ویژگی‌های [۴] استخراج شده برمبنای Jitter, Shimmer می‌باشند. ایده کلی به این دلیل است که ماهیچه‌های دستگاه صوتی، عموما دارای یک رفتار نوسانی پایدار هستند. بنابراین سه خانواده‌ی عمده این ویژگی‌ها به ترتیب بر مبنای: ۱-ارتعاشات و لرزش‌های بدون قائده(مانند معیارهای تعریف شده GP, PPE,...) ۲-نسبت نویز به اندازه سیگنال(مانند HNR, DFA,...)

[3] Machine Learning
[4] Feature

۳- ویژگی‌های مبتنی بر حرکات ناگهانی لب و زبان در افراد مبتلا می‌باشد (مانند MFCC).

۲.۵.۲ روش‌های انتخاب ویژگی‌ها

ابتدا با محاسبه‌ی اطلاعات متقابل [۵] سعی در مشخص نمودن میزان ارتباط ویژگی‌ها بایکدیگر داریم. سپس در طول مقاله از ۴ روش عمده برای کاهش بعد و انتخاب ویژگی‌های موثر استفاده شده است. این ۴ روش به شرح زیر می‌باشند.

۱. LASSO: این الگوریتم برای اندازه‌ی غیرصفر ویژگی‌ها نیز یک خطا در نظر می‌گیرد، بنابراین بزرگ شدن یک ویژگی تنها در صورتی منطقی است که کاهش قابل توجهی را در تابع هزینه ایجاد نماید و بدین‌ترتیب زیاد شدن خطای ناشی از بزرگی اندازه را جبران نماید. بدین‌ترتیب انتظار می‌رود در این روش تمامی ویژگی‌های نامرتبط که تاثیری در مسئله ندارند(ویا اطلاعات آنها در سایر ویژگی‌های یافت می‌شود) ضریب صفر پیدا کنند. این روش زمانی‌که ویژگی‌ها فاقد وابستگی باشند در تشخیص ویژگی‌های بهینه به خوبی عمل می‌نماید اما در غیر این صورت، این روش کاملا نسبت به نویز حساس خواهد بود.

۲. mRMR: این الگوریتم با استفاده از روش‌های ابتکاری [۶] سعی در انتخاب ویژگی‌ها به صورتی دارد که اطلاعات مرتبط بیشینه بشوند و از طرفی وابستگی اطلاعاتی بین ویژگی‌ها کمینه گردد. این روش به صورت Greedy (در هر مرحله یک ویژگی را مورد بررسی قرار می‌دهد) عمل می‌نماید و تنها وابستگی اطلاعاتی دو ویژگی نسبت به یکدیگر را در نظر می‌گیرد و ارتباط ویژگی انتخاب شده با مسئله طبقه بندی را لحاظ نمی‌نماید.

۳. RELIEF این الگوریتم سعی دارد با اتکار بر افزایش حاشیه [۷] ویژگی‌هایی را انتخاب نماید که بیشترین پراکندگی را در اطلاعات باقی‌مانده ایجاد می‌کند. این الگوریتم به طبقه‌بندهای KNN بسیار مرتبط است.

۴. LLBFS: این روش سعی در یررسی مسئله به صورت محلی دارد و سعی می‌نماید تا با برازش ابر صفحه‌های مماس به ساختار غیرخطی مسئله را تحلیل کند. سپس با استفاده از تحلیل مذکور برای هر ویژگی وزنی را در نظر می‌گیرد و در نهایت ویژگی‌های با وزن بیشتر را به عنوان ویژگی‌های بهینه انتخاب می‌نماید. این روش تعمیمی از روش RELIEF می‌باشد.

۳.۵.۲ اجرای روش‌ها و متدهای ارزیابی

در این بخش برای هر یک از ۴ بخش ذکر شده از روش Cross Validation بر روی داده‌های آموزش استفاده می‌نماییم. در واقع، در یک روش پیش‌پردازش، در هر مرحله آن، ۱۰ بار متد را اجرا می‌نماییم، سپس از بین تمامی دفعات اجرا شده ویژگی که بیشترین تعداد تکرار را داشته انتخاب می‌نماییم و آن را به مجموعه ویژگی‌های مورد انتخاب اضافه می‌نماییم و به همین ترتیب. البته دقت نمایید که الگوریتم LASSO می‌تواند در یک مرحله

[5] mutual information
[6] Heuristic
[7] Margin

ویژگی‌های انتخاب شده در مراحل قبل را حذف نماید به همین دلیل این فرآیند در الگوریتم LASSO به صورت مستقل اجرا می‌شود.

۶.۲ سوال۵

در مقاله‌ی مذکور از دو مدل Random Forest و SVM استفاده شده است.

۱. Random Forest در این مدل از ۵۰۰ درخت استفاده شده و تنها پارامتر قابل تنظیم تعداد ویژگی‌های مورد استفاده بوده که در تحلیل‌های سابق مشاهده شده است که این روش نسبت به این پارامتر مقاوم است. با این حساب سه پارامتر که معمولا به صورت $\sqrt{\text{Number of Features}}, \frac{\sqrt{\text{Number of Features}}}{2}, 2 \times \sqrt{\text{Number of Features}}$ انتخاب و مورد بررسی قرار گرفته است.

۲. SVM در این بخش از یک مدل SVM با هسته RBF استفاده شده است. دو پارامتر C, γ آن به استفاده از روش Grid Search محاسبه گردیده است.

۷.۲ سوال ۶

۱.۷.۲ بیان روش ارزیابی

در این مقاله نتایج با استفاده از روش‌های مختلفی گزارش شده است. اطلاعات متقابل ویژگی‌ها به عنوان یک تحلیل اولیه مورد استفاده قرار می‌گیرد. همچنین برای بررسی عملکرد کلی طبقه‌بندها و انتخاب ویژگی‌ها، ابتدا داده‌هابه دو بخش تست و آموزش تقسیم شده‌اند. سپس با استفاده از روش CV به آموزش مدل پرداخته شده و در نهایت دقت بر روی داده‌های تست مورد بررسی قرار گرفته. سپس این روند را ۱۰۰ بار تکرار می‌نمایند تا عملکرد کلی طبقه‌بند (میانگین و واریانس)آن مشخص گردد، و سپس نتایج بر اساس نمودارهای همراه با Barplot ها مشخص می‌گردد. و درنهایت دقت نزدیک به 0.99 بدست آمده است.

۲.۷.۲ تحلیل و بررسی روش ارزیابی

روش اجرا شده اگرچه به از جهات بسیار مقاوم و مناسب است، اما با توجه به تعداد کم مجموعه دادگان، و همچنین بالانس نبودن دادگان افراد سالم و غیرسالم باید اولا نتایج را بر روی دادگان بیشتری توضیح داد و از طرفی باید مترهای دیگر نیز مشخص بشوند. برای مثال استفاده از ماتریس Confusion می‌تواند مناسب باشد.

۸.۲ سوال ۷

در این مقاله روش‌های کلاسیک و نوین پردازش سیگنال و طبقه‌بندی بکار گرفته شد و حاصل این ترکیب مدل با قدرت موثر بوده است. به طور کلی مدل‌های پیش از مقاله حاضر، دقت حدود 0.93 را فراهم می‌نمودند، اما مدل حاضر با دقت 0.99 نشان دهنده بهبود چشمگیر در عملکرد مدل‌های مورد بحث می‌باشد. همچنین از دیگر نوآوری‌های مقاله مذکور استفاده از ۴ روش پیش‌پردازش بوده. مشاهده که LASSO به دلیل وجود وابستگی بین ویژگی‌ها و حضور نویز نتوانسته عملکرد بهینه‌ای داشته باشد. همچنین الگوریتم mRMR به این دلیل که عملا میزان شرکت ویژگی در نتیجه مسئله را کنار می‌گذاشته است، نتوانسته عملکرد مناسبی داشته باشد. اما به عکس همین دلایل روش‌های RELIEF, LLBFS به طرز موثری موفق عمل کرده‌اند.

همچنین مشاهده شده که در بخش انتخاب ویژگی، خانواده ویژگی‌های SNR, MFCC به طور پیوسته و دائم انتخاب شده‌اند. باتوجه به نویز و لرزش مورد انتظار در بدن و ماهیچه‌ها وجود SNR منتظره بوده‌است، اما حضور MFCC که به دلیل حساسیت به کنترل ناکافی ارگان‌های صدایی در تبیین تلفظ می‌باشد بوجود می‌آید، غیرمنتظره بوده و این یافته جدید می‌تواند به مطالعات آینده در مورد این بیماری جهت بدهد.

همچنین مشاهده شد که بین عملکرد SVM , RF تفاوت قابل ملاحظه‌ای وجود داشته است. دلیل این موضوع باید در بررسی‌های آینده مشخص گردد، اما به طور کلی می‌توان گفت که باید علاوه بر دقت میزان اعتماد [۸] به تصمیم را مدنظر داشت و سپس بر این اساس طبقه‌بندها را مقایسه نمود. برای مثال می‌توان خروجی‌های SVM, RF را به خروجی‌های احتمالاتی تبدیل نمود.

همجنین در مطالعات مشابه پیشین مشاهده شد که تقسیم بندی دادگان بر اساس جنسیت می‌تواند به دقت طبقه‌بند کمک بسیار نماید، اما در تحقیق جاری این موضوع محقق نگردید که این می‌تواند به دلیل حجم پایین دیتای در دست باشد.

همچنین دیتای جمع آوری شده در این آزمایش در یک محیط کاملا کنترل شده انجام شده است، باید سعی کرد شرایط جمع‌آوری دیتا به محیط واقعی نزدیک تر بشود. همچنین در بررسی‌های آینده سعی می‌گردد اطلاعات بیشتری از نمونه‌های موجود، به عنوان ویژگی استخراج گردد.

[8] Confidence

فصل ۳

معرفی روش‌ها

در این بخش به معرفی ابزارهای مورد استفاده در این گزارش می‌پردازیم. روند کلی کار مبتنی بر سه بخش پیش‌پردازش، طبقه‌بندی و ارزیابی می‌باشد، به همین دلیل در این بخش به تفصیل به بیان هر کدام از بخش‌ها به طور جداگانه خواهیم پرداخت.

۱.۳ پیش‌پردازش

پیش پردازش دیتا در جهت حذف نویز، مقاوم کردن طبقه‌بند نسبت به مقیاس ویژگی‌های مختلف و کاهش ابعاد مسئله و در نتیجه کاهش حجم محاسبات صورت می‌گیرد. برای دستیابی به موارد ذکر شده مراحل یکه سازی، بالانس دیتا و کاهش بعد به ترتیب صورت می‌گیرد که در این بخش به هرکدام به تفصیل پرداخته خواهد شد.

۱.۱.۳ یکه‌سازی دیتا

یکه‌سازی دیتا جهت هم‌مقیاس کردن داده‌ها، مقاوم سازی به نویز و ویژگی‌های نامرتبط ضروری است. در واقع در صورت عدم یکه‌سازی دیتا، مشاهده می‌گردد که بعضا تنها به دلیل تفاوت در مقیاس به یک ویژگی کمتر مهم وزن بیشتری در فرآیند طبقه‌بندی داده می‌شود. مدل‌هایی که از داده‌های یکه نشده بهره می‌برند در شرف واگرایی در فرآیند یادگیری می‌باشند. در این گزارش از دو روش یکه‌سازی دیتا استفاده شده است که در این بخش به بررسی هر کدام خواهیم پرداخت.

Min Max Scaler

در این روش سعی می‌گردد که اندازه‌ی مطلق داده‌ها تماما در بازه‌ی $[0, 1]$ قرار بگیرد. در واقع از فرمول ریاضی زیر برای انجام این عملیات استفاده می‌گردد:

$$x_n = \frac{x - \min(x)}{\max(x) - \min(x)}$$

. همانطور که مشخص است، برای استفاده از روش مذکور به یک تخمین نسبتا دقیق از کمینه و بیشینه‌ی رنج مقادیر دیتا نیازمندیم.

Standard Scaler

در این روش فرض می‌گردد که توزیع داده‌ها به صورت نرمال می‌باشد، بنابراین با استفاده از این روش توزیع دیتای نهایی به صورت یک گوسی با میانگین صفر و واریانس یکه می‌باشد. در واقع با این کار توزیع مجموعه داده از حالت یک بیضی گون، به صورت یک دایره تبدیل می‌گردد و این موضوع موجب سادگی هندسی در یادگیری می‌گردد(بیضی دارای دو پارامتر شعاع کوچک و بزرگ می‌باشد اما دایره یک شکل کاملا متقارن و با شعاع ثابت است)در واقع از فرمول ریاضی زیر برای انجام این عملیات استفاده گردد:

$$x_n = \frac{x - \mu}{\sigma}$$

که در رابطه‌ی بالا مقصود از μ همان میانگین نمونه‌ها و مقصود از σ همان انحراف معیار نمونه‌ها می‌باشد. برای استفاده از این روش باید تخمین مناسبی از میانگین و واریانس را در دست داشت.

از آنجا که تخمین واریانس و میانگین نسبت به تخمین بیشینه و کمینه مجموعه داده مقاوم‌تر می‌باشد، استفاده از روش Standard Scaler نسبت به روش Min Max Scaler ارجح است.

از این روش می‌توان برای داده‌هایی که دارای توزیع غیر نرمال می‌باشند نیز استفاده کرد، اما ممکن است که این جواب‌ها قابل اتکا و اعتماد نباشند.

۲.۱.۳ نحوه محاسبه یکه‌سازی

برای اعمال روش‌های یکه سازی ابتدا با استفاده از داده آموزش پارامترهای مهم مسئله آموزش محاسبه می‌گردد(بیشینه و کمینه داده در روش Min Max Scaler و میانگین و انحراف معیار در روش Standard Scaler)، سپس برای یکه سازی داده‌های تست نیز از پارامترهای محاسبه شده بر روی داده‌های آموزش استفاده می‌گردد.

۳.۱.۳ بالانس کردن دیتا

در برخی از دیتاست ها (از جمله دیتاستی که برای این تمرین در نظر گرفته شده است) تعداد داده‌های یکی از کلاس‌ها از دیگری بسیار بیشتر است. به عبارت دیگر، توزیع داده در کلاس‌ها یکنواخت نیست. در این حالت، معمولا طبقه‌بند با پیش‌بینی درست در رابطه با کلاس با داده‌های بیشتر به دقت بالایی می‌رسد. اما به این علت که در رابطه با کلاس دیگر داده کافی ندارد، دچار خطا در پیش‌بینی می‌شود.

برای رفع مشکل بیان شده،راه‌های متفاوتی وجود دارد که در زیر به دو مورد محبوب‌تر آن اشاره می‌شود:

۱. **Random Undersampling and Oversampling**: در این روش نمون‌های اضافه مربوط به کلاس با داده بیشتر حذف شده و/یا داده‌هایی به کلاس با داده کمتر اضافه می‌گردد.

۲. **Undersampling and Oversampling using 'imbalanced-learn'**: در این روش که با استفاده از کتابخانه‌های آماده پایتون است، داده‌ها بالانس می‌شوند. یکی از این روش‌ها oversampling by SMOTE است که در این روش داده‌هایی برای کلاس با داده کمتر ساخته می‌شود.

۴.۱.۳ کاهش بعد

همواره داده‌های جمع آوری شده دارای ویژگی‌های [۱] مختلفی هستند که معمولا این ویژگی‌ها به یکدیگر وابستگی اطلاعاتی دارند، بطوریکه با دانستن یک یا چند ویژگی دیگر می‌توان حدس‌هایی راجع به ویژگی مورد بحث در نظر داشت. در مدلسازی‌های ریاضی مسائل طبقه‌بندی، سعی می‌گردد داده‌های جمع آوری شده کمی سازی شده و سپس در یک فضای جبری نمایش داده شوند، به طوریکه برای مثال اگر $\mathbb{R}^n$ فضای جبری مدنظر باشد، آنگاه هر داده به صورت یک بردار $x \in \mathbb{R}^n$ نمایش داده می‌شود که هر بعد این فضا نمایانگر ویژگی خاصی از داده‌ها است و هر درایه‌ی بردار x بیانگر مقدار ویژگی متناظر می‌باشد. حال در این فضای جبری، وابستگی‌ها می‌توانند به صورت خطی و یا غیرخطی ظهور پیدا کنند. از طرفی با افزایش بعد در فضای جبری، مدلسازی مسئله در شرف Overfitting قرار می‌گیرد. برای جلوگیری از پدیده‌های مذکور و همچنین برای برآورد موفق مدل مسئله، حجم داده‌های مورد نیاز به صورت نمایی افزایش می‌یابد. از آنجا که فرآیند جمع آوری دیتا با صرف هزینه وقت و سایر عوامل همراه است، عملا امکان جمع آوری دیتا به اندازه کافی برای مسائل با ابعاد بالا وجود ندارد. راه حل موثر برای مقابله با این مسئله استفاده از روش‌های کاهش بعد است. بدین‌ترتیب سعی می‌گردد اطلاعات مستقل داده‌ها استخراج و اطلاعات غیرمرتبط یا کمتر مرتبط را از آنها حذف نمود. بدین‌ترتیب با مسئله‌ای با ابعاد کمتر مواجه هستیم که مدلسازی در این محیط نسبت به محیط سابق مقاوم‌تر و کاراتر خواهد بود.همچنین به دلیل کاهش بعد مسئله عملا حجم محاسبات کاهش یافته و بدین‌ترتیب می‌توان در استفاده از زمان و منابع سخت‌افزاری پیشرفته کاملا بهینه عمل کرد. به طور کلی این روش‌ها از نظر عملکردی به دو دسته تقسیم می‌شوند. برخی از روش‌ها سعی در استخراج بیشینه اطلاعات موجود در مجموعه داده بدون توجه به مسئله طبقه‌بندی را دارند. در این روش‌ها به این دلیل که عموم اطلاعات مفید قابل بازیابی هستند قدرت تعمیم مناسبی دارند، اما اشکال این دسته از روش‌ها آنست که بسیاری از اطلاعات نامرتبط با مسئله خاص طبقه‌بندی نیز همچنان در دیتا باقی است که می‌تواند خطر بروز Overfitting را بیشتر کرده و هزینه محاسباتی را افزایش بدهد. در طرف مقابل دسته دیگری از روش‌ها وجود دارند که با توجه به مسئله طبقه‌بندی و باتوجه به نوع لیبل داده‌ها سعی در کاهش بعد دارند. این روش‌ها برای مسئله خاص طبقه‌بندی بسیار بهینه می‌باشند اما از طرفی قدرت تعمیم خودرا برای سایر مسائل از دست می دهند. در این بخش نمونه‌هایی از روش‌های هر بخش ذکر می‌گردد.

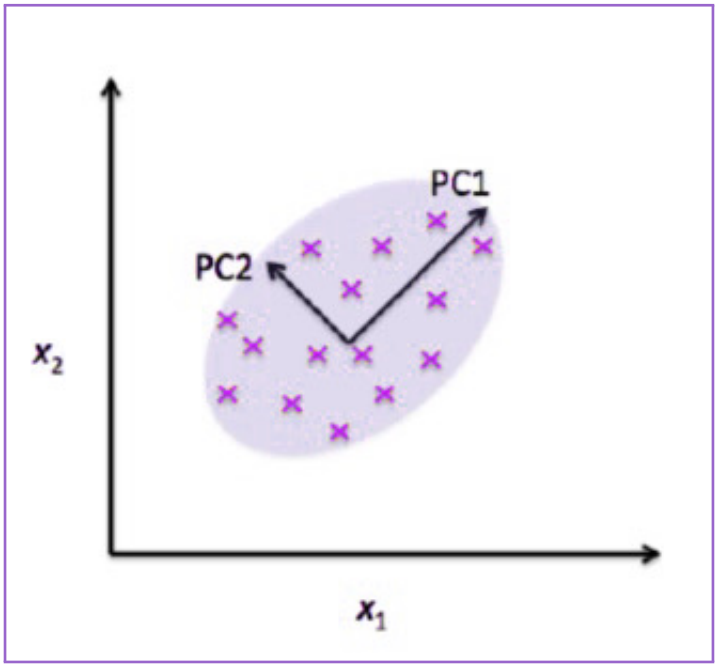
Filter Methods

این روش‌ها سعی در کاهش بعد مسئله با حفظ تمام(یا بخش اعظم) اطلاعات مجموعه داده را دارند. در واقع مشکل اصلی در مسئله‌های با بعد بالا این است که مدل دچار overfitting میشود و این اتفاق باعث کاهش قدرت تعمیم مدل به داده‌هایی به غیر از داده‌های آموزش می‌گردد. لازم به ذکر است که با افزایش ابعاد، این توانایی به صوزت نمایی کاهش میابد، به همین علت کاهش بعد داده‌ها دارای اهمیت می‌باشد. به‌علاوه با کاهش بعد میتوان به مدل‌هایی رسید که کارآمدتر هستند، چراکه سرعت یادگیری در آن‌ها بالاتر بوده و هزینه‌های محاسباتی کم‌تری را صرف این کار می‌کنند. این روش‌ها مستقل از مسئله کلاس‌بندی عمل می‌نمایند و قابلیت تعمیم دارند. از جمله این روش‌ها می‌توان به موارد زیر اشاره نمود:

[1] Feature

۱. **PCA**: این روش، روشی بدون نظارت [۲] و ناپارامتری [۳] است که در درجه اول برای کاهش بعد مورد استفاده قرار میگیرد. میتوان آن را از این منظر مورد بررسی قرار داد که یک روش تصویر کردن داده است. به این ترتیب که داده با m ستون (ویژگی [۴]) به یک زیرفضا با m و یا کمتر ستون تبدیل میشود درحالی که ماهیت داده اصلی حفظ میشود. به همین علت، میتوان از این روش برای فیلتر کردن نویز بهره برد. به این ترتیب که اولین مولفه‌ی اصلی در بردارنده بیشترین واریانس است. به همین ترتیب میزان واریانس در مولفه‌های بعدی کاهش و میزان نویز افزایش می‌یابد. پس، نمایش داده‌ها با زیرمجموعه کوچک‌تری از مولفه‌های اصلی دیتاست باعث فیلتر شدن هرچه بیشتر نویز میگردد.

به عبارت دیگر، این روش می‌تواند تغییرات موجود در دیتاست را نمایان سازد چراکه به نوعی محورها را میچرخاند. در واقع اولین مؤلفه‌ی اصلی با استفاده از رابطه ای خطی از ویژگی‌ها ساخته شده و در راستای بیشترین واریانس دیتاست است. به همین علت بیشترین تغییرات این مجموعه داده و در نتیجه بیشترین اطلاعات را شامل میشود. دومین مؤلفه‌ی اصلی شامل روابط خطی دیگری بوده، واریانس های باقی مانده را شامل شده و از مؤلفه‌ی اول نابسته است. (بر آن عمود می‌باشد.) به همین ترتیب، مؤلفه های دیگر واریانس های باقی‌مانده را در برگرفته و بر سایر مؤلفه‌ها عمودند. این موضوع را می‌توان در شکل زیر مشاهده کرد.



شکل ۱.۳: مؤلفه‌های اصلی یک دیتاست

تعداد این ترکیب‌های خطی و در واقع بیشترین تعداد مؤلفه های قابل تعریف روی یک دیتاست برابر با تعداد ابعاد آن دیتاست می‌باشد.

براساس مفاهیم جبری میدانیم که بردارها و مقادیر ویژه به عنوان معیارهایی برای جهت و مقدار تغییراتی که در هر جهت رخ میدهند، شناخته می‌شوند. در واقع بردارهای ویژه جهت و یا زاویه و مقادیر ویژه اندازه

[2] Unsupervised
[3] Non-parametric
[4] Feature

واریانس داده در جهت محور ها را نمایش می‌دهند. به همین علت استفاده از این معیار ها برای بدست آوردن مؤلفه‌های اصلی واضح و منطقی به نظر می‌رسد.

در PCA فرض های زیر وجود دارند:

(آ) کمترین تعداد داده های برابر ۱۵۰ است و در حالت ایده‌آل تعداد مشاهدات ۵ برابر تعداد ویژگی هاست.

(ب) ویژگی ها نسبت به هم همبستگی دارند. به همین علت دیتاستی که کاهش بعد یافته است، دیتاست اصلی را به خوبی نمایش میدهد.

(ج) همه‌ی مغیرها رابطه‌ی ثابت نرمال چندمتغیره را به نمایش می‌گذارند و مؤلفه های اصلی ترکیبی خطی از ویژگی‌های اصلی اند.

(د) باتوجه به این که داده‌های خارج از محدوده [۵] می‌توانند آثار نامتناسبی بر نتایج بگذارند، فرض می‌شود که داده‌های خارج از محدوده‌ی قابل توجهی در دیتاست وجود ندارد.

(ه) همان طور که بیان شد، محورها با واریانس زیاد به عنوان مؤلفه‌های اصلی و محورهای با واریانس کم به عنوان نویز طلقی می‌شوند.

با توجه به فرض‌های بالا، به موارد زیر باید توجه کرد:

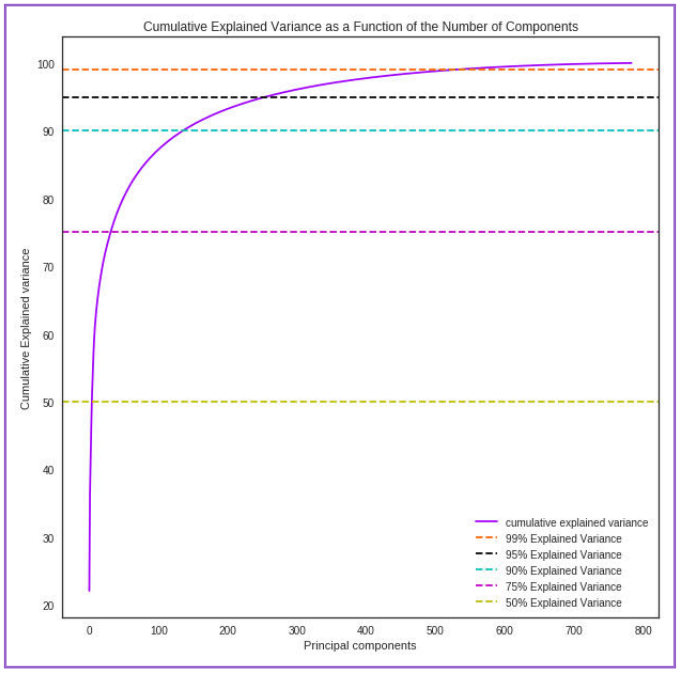
(آ) در PCA مؤلفه‌ها با بیشترین مقدار واریانس مورد توجه هستند و تاثیرگذاری هر مؤلفه بر اساس اندازه واریانس تعیین میگردد. بنابراین، بهترین کار این است که قبل از اجرای PCA داده‌ها را نرمالیزه نماییم. چراکه در غیر این صورت، داده‌های اسکیل نشده، با معیارهای اندازه‌گیری متفاوت، این قیاس نسبی واریانس ویژگی ها را از شکل درست و طبیعی خارج می‌نمایند.

(ب) یکی از راه‌های کاربردی برای بدست آوردن تمامی روابط ممکنه بین ابعاد، بدست آوردن ماتریس کواریانس است. به این ترتیب، برای بدست آوردن فراوانی تجمعی واریانسی که توسط هر مؤلفه‌ی اصلی در بر گرفته شده است، به راحتی می‌توان از این ماتریس استفاده کرد.

(ج) برای انتخاب تعداد مؤلفه‌های اصلی بهینه باید به نسبت فراوانی که توسط واریانس در برگرفته شده [۶] بدست می‌آید و تابعی از تعداد مؤلفه‌هاست بهره برد. تعداد این مؤلفه‌ها کاملا به tradeoff میان کاهش ابعاد و میزان از دست دادن اطلاعات باز می‌گردد. همان طور که در نمودار زیر مشاهده می‌شود، تقریبا ۷۵ درصد واریانس با انتخاب ۱۰۰ تا از ۷۸۴ ویژگی و ۹۵ درصد با انتخاب ۳۰۰ از ۷۸۴ ویژگی در بر گرفته می‌شود.

[5] Outliers

[6] Cumulative explained variance ration



شکل ۲.۳: میزان CUMULATIVE EXPLAINED VARIANCE بر اساس تابعی از تعداد اجزا

باتوجه به توضیحات بیان شده، برای بدست آوردن این مؤلفه‌ها، کافی است داده‌ها را بدون بایاس کنیم. (به این معنی که میانگین آن‌ها را بدست آورده و از کل دیتاست کم کنیم.) سپس، ماتریس کوواریانس دیناست حاصل را بدست می‌آوریم. در مرحله‌ی بعدی، بردارها و مقادیر ویژه این ماتریس بدست می‌آید. بردارهای ویژه را براساس مقادیر ویژه متناظر به صورت نزولی مرتب کرده و تعداد بهینه‌ی آن‌ها را به عنوان مؤلفه‌های اصلی انتخاب میکنیم.مانند سایر روش‌ها این روش نیز ابتدا بر روی داده‌های آموزش اعمال می‌شود سپس با ویژگی‌های انتخاب شده در مسئله کلاس بندی استفاده می‌گردد.

باید دقت شود که علارغم سادگی و فراگیری این روش، PCA محدودیت‌هایی نیز دارد که باید مورد توجه قرار گیرند. از جمله ای محدودیت‌ها میتوان به موارد زیر اشاره کرد:

(آ) این روش میتواند باعث کاهش عملکرد مدل بر روی دیتاست شود. چرا که وابستگی‌ها را حذف می‌کند. به علاوه ممکن است فرض خطی بودن را نتواند برآورده سازد.

(ب) وابستگی این روش به واریانس ممکن است باعث درنظر نگرفتن تفاوت‌های میان کلاس‌های متفاوت گردد. به علاوه ممکن است ویژگی‌هایی که باعث تفاوت کلاس‌ها با یکدیگر می‌شوند در مؤلفه‌هایی با واریانس کم وجود داشته باشند که در این روش مورد توجه قرار نگیرند.

(ج) همان طور که بیان شد، این روش به داده‌های خارج از محدوده حساس است.

(د) با توجه به این که هر مؤلفه ترکیب خطی از ویژگی‌هاست، ارزش هر ویژگی به تنهایی مورد توجه قرار نمی‌گیرد.

۲. ICA: ICA یکی از روش‌هایی است که به کمک آن می‌توان از یک سیگنال مخلوط شده دریافتی، سیگنال‌های فرستاده شده‌ی متقل از منابع متسقل را جدا کرد. برخلاف PCA که سعی در بیشینه کردن

واریانس داده‌ها داشت، در این روش به دنبال مؤلفه‌های مستقل هستیم. در واقع در این روش سیگنال مخلوط شده‌ای از منابع مستقل به ما داده می‌شود و انتظار می‌رود سیگنال‌های مستقل را مشخص کنیم.

برای این که این مؤلفه‌های مستقل را به دست آوریم، مراحل زیر را انجام می‌دهیم:

(آ) اگر سیگنال دریافتی مخلوط را x بنامیم، در مرحله اول، آن را سفید می‌کنیم.

(ب) دراین مرحله، یک مقدار رندوم برای ماتریس de-mixing که آن را $\mathbf{w}$ می‌نامیم، در نظر میگیریم.

(ج) سپس مقدار $\mathbf{w}$ جدید را بدست آورده و آن را نرمایزه میکنیم.

(د) بررسی میکنیم که آیا الگوریتم به پایان رسیده یا خیر. (معمولا بررسی میشود که آیا ضرب نقطه ای $\mathbf{w}$ در ترانهاده‌اش تقریبا برابر ۱ شده است سا خیر.) در صورت همگرا نشدن، مرحله قبل را تا همگرایی تکرار می‌کنیم.

(ه) در مرحله آخر، بعد از بدست آوردن مقدار مناسب $\mathbf{w}$ از مراحل قبل، ضرب نقطه ای آن در x به ما سینگنال‌های مستقل ورودی را بدست می‌دهد.

باید دقت شود که این روش را نمی‌توان روی هر سیگنال مخلوطی به‌کار برد:

(آ) سیگنال مخلوطی باید ترکیب حطی از سیگنال منابع مستقل باشد.

(ب) در این روش فرض می‌شود که سیگنال‌های ارسالی ا منابع از هم مستقل هستند. درحالی که سیگنال‌های مخلوط شده این ویژگی را ندارند و در واقع از این همبستگی برای جداسازی آن‌ها بهره میبرد.

(ج) در این روش فرض می‌شود که سیگنال‌های ارسالی گوسی نیستند. بر اساس قضیه‌ی حد مرکزی به این نتیجه می‌رسد که مجموعه این سیگنال‌ها توزیعی دارند که بیشتر از هر تک سیگنال مشابه توزیع گوسی می‌باشد.

(د) تعداد مؤلفه‌های متسقلی که توسط این روش یافت می‌شود برابر با تعداد مخلوط‌های مشاهده شده است.

بنابراین، قبل از استفاده از این روش باید به موارد بالا توجه کرد.

شایان ذکر است که PCA و ICA تفاوت‌هایی با یکدیگر دارند، که در زیر به آن‌ها اشاره می‌کنیم:

(آ) در روش PCA ، همان طور که اشاره شد، به دنبال کاهش بعد هستیم تا از overfitting جلوگیری کنیم. این درحالی است که در ICA سیگنال مخلوط به سیگنال‌های مستقل منابع تجزیه می‌شود.

(ب) در PCA ما به دنبال مؤلفه‌های اصلی میگردیم، درحالی که در ICA به دنبال مؤلفه‌هایی مستقل هستیم.

(ج) در PCA تمرکز اصلی روی بیشینه کردن واریانس است، درحالی که در ICA مسئله واریانس در میان داده‌ها مورد توجه نیست.

(د) در PCA همچنین روی تعامد مؤلفه های اصلی هم متمرکز هستیم، درحالی که در ICA مؤلفه‌ها لزوما برهم عمود نیستند.

(ه) در PCA روی استقلال مؤلفه‌ها متمرکز نیستیم، در حالی که در ICA این مسیله کاملا بنیادین است.

در شکل زیر می‌توان تفاوت‌های بیان شده را مشاهده کرد.

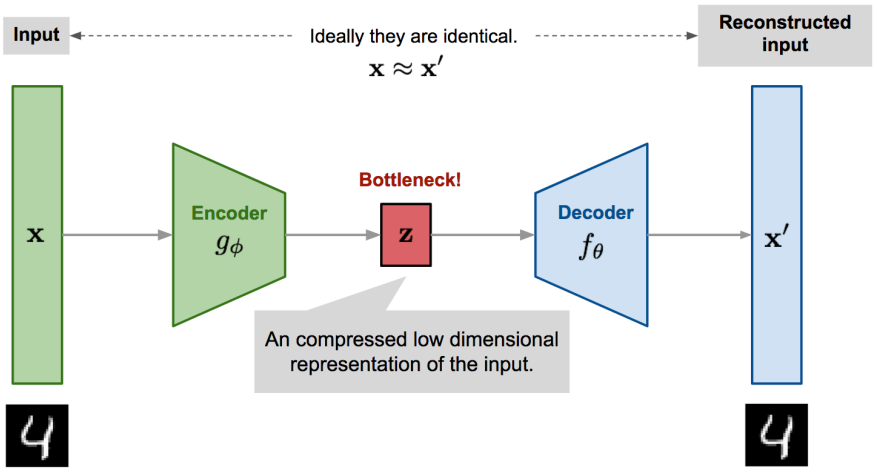
شکل ۳.۳: تفاوت روش‌های PCA و ICA

۳. Autoencoders: در این روش سعی می‌گردد از یک شبکه‌ی عصبی برای کاهش بعد و رفع وابستگی‌های خطی و غیرخطی بهره برد. شکل ؟؟ نمایی از یک Autoencoders را نشان می‌دهد. همانطور که مشاهده این ساختار از دو بخش Encoder و Decoder تشکیل شده است. Encoder سعی می‌کند داده‌ی ورودی را به فضای با بعد کمتر برده که این فضا، فضای تبدیل یافته [۷] نامیده می‌شود. سپس Decoder سعی می‌کند دیتا را از فضای تبدیل یافته مجددا به فضای اولیه ببرد، بنابراین خروجی شبکه با ورودی آن هم بعد است(تعداد نودهای ورودی و خروجی شبکه باهم یکسان می‌باشد). حال سعی می‌کنیم شبکه را طوری آموزش دهیم که خروجی‌های شبکه هم ارز و مشابه ورودی‌های شبکه باشند. برای حصول شباهت مدنظر از یک متر مناسب مانند حداقل مربعات می‌توان استفاده کرد، سپس با استفاده از روش‌های معمول آموزش شبکه به به روز رسانی وزن‌ها می‌پردازیم. در صورتی که پارامترهای ساختاری شبکه(تعداد نودهای لایه‌ی میانی، تعداد لایه‌ها و ...) به درستی انتخاب شوند در این صورت مشاهده می‌گردد که فضای تبدیل یافته حاوی تمامی اطلاعات مفید فضای اولیه است. در واقع Encoder توانسته است به خوبی وابستگی‌های خطی و غیرخطی را حذف نماید و در عین حال اطلاعات مفید را محفوظ نگاه دارد. سپس Decoder که معادل یک تابع است توانسته با استفاده از اطلاعات این فضا مجددا فضای اولیه را بازیابی نماید. در نهایت پس از آموزش Encoder را جدا کرده و از آن به عنوان تابع(ماشین) کاهش بعد دهنده استفاده می‌نماییم و خروجی آن را به عنوان ورودی طبقه‌بند در نظر می‌گیریم.

Wrapper Methods

این روش‌ها سعی در کاهش بعد مسئله باتوجه به مسئله کلاس‌بندی دارند. در واقع سعی می‌گردد علاوه بر اطلاعات تکراری، اطلاعات نامرتبط با مسئله نیز از مجموعه داده حذف گردد، به همین دلیل این روش‌ها نسبت به

[7] Latent Space



شکل ۴.۳: ساختار کلی یک AUTOENCODER

روش‌های Filter در مسئله کلاس‌بندی خاص بهینه‌تر عمل می‌نمایند، اما قدرت تعمیم چندانی نخواهند داشت. از حمله این روش‌ها می‌توان به موارد زیر اشاره نمود:

۱. LDA: در تجزیه‌ی LDA هدف تصویر کردن داده بر یک ابر صفحه با بعد کمتر است، به طوریکه داده‌های کلاس‌های مختلف به صورت کاملا تجزیه پذیر از هم قرار بگیرند، در واقع فاصله بین کلاسی بیشینه و همزمان فاصله درون کلاسی کمینه بشود. بنابراین در یک مسئله با c کلاس متفاوت، می‌توان ابرصفحه‌ای به ابعاد $c-1$ را مدنظر داشت، و دیتا را بر روی این ابر صفحه تصویر کرد. در واقع در مسئله LDA به دنبال تبدیل w هستیم، به طوریکه نسبت پراکندگی بین کلاسی در فضای جدید به پراکندگی درون کلاسی در فضای جدید، کمینه گردد:

$$arg\ min_w J(w) = \frac{w^T S_B w}{w^T S_w w}; \qquad y = w^T x$$

. همانند تمامی روش‌های دیگر در بخش پیش‌پردازش، ابتدا با استفاده از داده‌های آموزش و حل مسئله بهینه سازی مذکور به محاسبه w پرداخته می‌شود، سپس تمامی داده‌ها(از جمله داده‌های تست) با استفاده از تبدیل مورد نظر به فضای جدید کاهش بعد یافته برده می‌شوند.

۲. Sequential Backward Feature Elimination: در این روش سعی می‌گردد ابتدا تمامی d ویژگی موجود در مجموعه دادگان را در نظر بگیریم. سپس مجموعه تمامی انتخاب‌های $d-1$ ویژگی را مدنظر قرار می‌دهیم و طبقه‌بند را با استفاده از آن‌ها می‌سازیم. سپس بدترین ویژگی را از میان مجموعه ویژگی‌ها حذف می‌نماییم. این روند را تا جایی ادامه می‌دهیم که دیگر بهبودی در عملکرد طبقه‌بند مشاهده نشود و یا طبقه‌بند از منظر بهینه بودن دچار افت شود. این روش یک روش بهینه‌ی مناسب است اما در بعدهای بالا عملا کارایی خود را از دست می‌دهد. در واقع از این روش تنها می‌توان برای مسائلی با بعد حدود $30-40$ استفاده نمود و در ابعاد بالاتر عملا به دلیل افزایش حجم محاسبات به صورت نمایی این روش کارآیی خود را از دست می‌دهد.

همانند سایر روش‌ها این روش نیز ابتدا بر روی داده‌های آموزش اعمال می‌شود سپس با ویژگی‌های انتخاب شده در مسئله کلاس بندی استفاده می‌گردد.

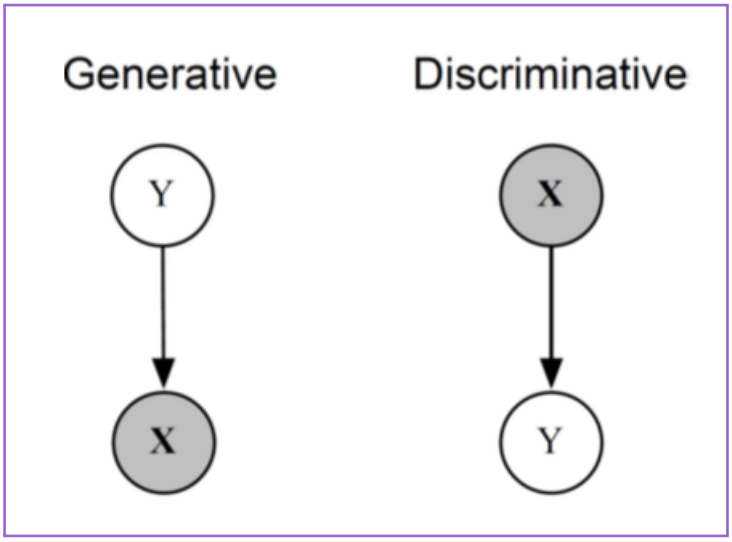
۲.۳ طبقه‌بندی

روش های Generative و Discriminative روش های بسیار معمولی در مدل سازی های machine learning هستند. به طور مثال، مدل توامی از کلاس‌ها ($Y = y$) و ویژگی‌ها ($X = \{x_1, x_2, ..., x_n\}$) که توزیع مشترک آن‌ها را میتوان به فرم $P\{Y, X\} = P\{y, x_1, x_2, ..., x_n\}$ نمایش داد، در دست داریم. هدف ما تعیین احتمال شرطی $P\{Y|X\}$ است. برای حل این مسئله از هر دو روش بیان شده می‌توان استفاده نمود.

برای بدست آوردن احتمال شرطی بیان شده، مدل های Generative احتمال بر اساس دانش اولیه [۸] را که با $P\{Y\}$ نمایش داده می‌شود و likelihood را که با $P\{X|Y\}$ نمایش داده می‌شود با استفاده از داده‌های آموزش تخمین میزنند. به عبارت دیگر به تخمین توزیع توام $P\{X, Y\}$ می‌پردازند. سپس بر اساس رابطه بیز مقدار احتمال ثانویه [۹] را بدست می‌آورند. به صورت ریاضی:

$$P\{Y|X\} = \frac{P\{X, Y\} = P\{Y\}P\{X|Y\}}{P\{X\}}$$

از سوی دیگر، مدل های Discriminative به صورا مستقیم به تخمین پارامترهای مربوط به احتمال شرط با استفاده از داده‌های آموزش می‌پردازند.



شکل ۵.۳: تفاوت روش‌های DISCRIMINATIVE و GENERATIVE

در واقع اگر به شکل بالا دقت کنیم، متوجه تفاوت این دو روش برای حل مسئله می‌شویم. در حالت Discriminative مقادیر X برای ما دانسته هستند و Y را نمی‌دانیم. بنابراین می‌توانیم با استفاده از دانسته‌های خود از X های داده شده، به $P\{Y|X\}$ برسیم. اما، در روش Generative ما باید با بهره‌جویی از $P\{Y\}$ و $P\{X|Y\}$ به احتمال شرطی هدف برسیم.

حال برای مقایسه این دو روش، به بررسی برخی از متریک‌ها میپردازیم:

[8] Prior probability
[9] Posterior probability

۱. **Accuracy**: در مسائلی که فرض استقلال شرطی قابل ارضا شدن نمی‌باشد، مدل های Generative در قیاس با روش های Discriminative دقت کم‌تری دارند.

۲. **Missing Data**: مدل های Generative توانایی کار کردن با دیتاست‌هایی شامل داده‌های از دست رفته هستند را دارند. چراکه می‌توان این داده‌ها را به عنوان حاشیه در نظر گرفته و آن‌ها را حذف نمود. این درحالی است که مدل‌های Discriminative این ویژگی را ندارند.

۳. **Performance**: در قیاس با مدل‌های Discriminative، مدل‌های Generative نیازمند داده‌های بیشتری برای یادگیری هستند. چراکه این مدل‌ها برای بدست آوردن توزیع شرطی هدف، نیازمند تخمین دقیق توزیع‌های بیان شده می‌باشند. به همین علت، میتوان گفت، این مدل‌ها هزینه محاسباتی بیشتری نیز دارند.

۴. **Application**: مدل‌های Discriminative برای جداسازی کلاس‌ها مورد استفاده قرار می‌گیرند، به همین علت از آنها تنها در مسائل کلاس‌بندی استفاده می‌شود. این در حالی است که مدل‌های Generative به جز مسائل کلاس‌بندی در مسائل دیگری مانند، نمونه‌برداری، یادگیری‌های بیزی و استنباط‌های MAP نیز قابل استفاده هستند.

۵. **Outliers**: مدل‌های Generative به داده‌های هخرج از محدوده حساس‌تر هستند. به این علت که این داده‌ها می‌توانند توزیع‌ها را تحت تاثیر قرار دهند. این درحالی است که مدل‌های Discriminative این داده‌ها را به راحتی به عنوان خطا تلقی می‌کنند.

باتوجه به موارد بیان شده بالا، به صورت کلی، مدل‌های Generative هنگامی مورد استفاده قرار می‌گیرند که ما اطلاعاتی درباره توزیع داده داریم و به دنبال یافتن پارامترهای نهفته در این توزیع هستیم. از سوی دیگر، مدل‌های Discriminative معمولا برای مسائلی که هدف جداسازی داده‌ها به کلاس‌های متفاوت است، کارایی دارند.

۱.۲.۳ Generative Approaches

Parzen Window

این روش، یکی از روش‌های بسیار پرکاربرد در تخمین ناپارامتری است. به این ترتیب که سعی می‌شود تابع چگالی در نقطه $\mathbf{x}$ با استفاده از نمونه‌های دیده شده قبلی، بدون هیچ اطلاعاتی در رابطه با توزیع داده‌ها، تخمین زده شود.

یکی از کاربردهای بسیار زیاد این روش، تخمین likelihood یا class-conditional density در مسئله شناسایی الگو [10] بانظارت [11] بر اساس داده‌های آموزش می‌باشد. در واقع در مسئله کلاس‌بندی بیز، اگر پارامترهای class-conditional density را بدانیم، طراحی این طبقه‌بند به راحتی صورت می‌گیرد.

در این روش بررسی می‌شود که چند نمونه در یک ناحیه (پنجره) مشخص R قرار می‌گیرند. بنابراین تابع چگالی در نقطه ی $\mathbf{x}$ به صورت زیر به‌دست می‌آید:

$$p(\mathbf{x}) = \frac{k}{nV}$$

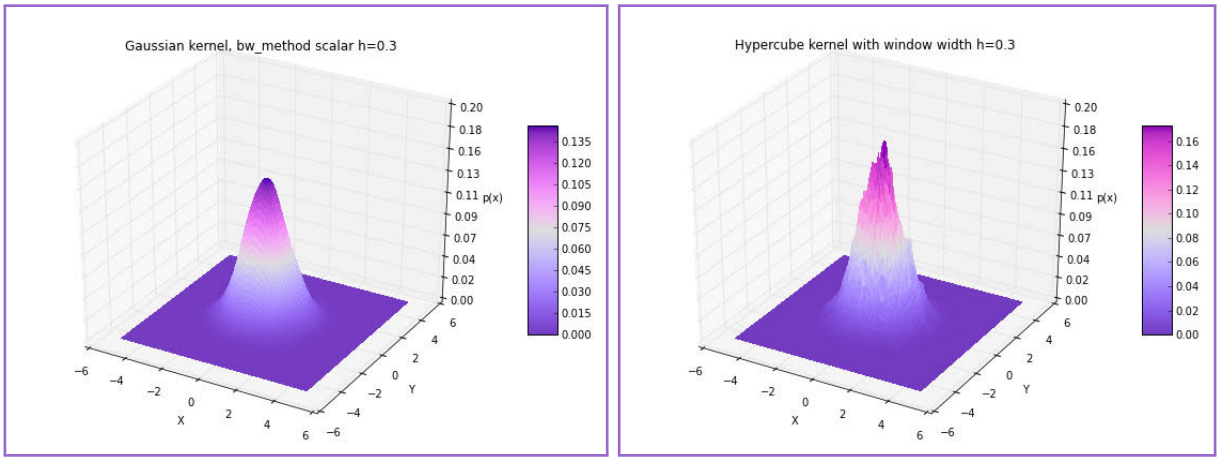
[10] Pattern recognition

[11] Supervised learning

حال اگر پنجره پارزن را در رابطه بالا اعمال نماییم داریم:

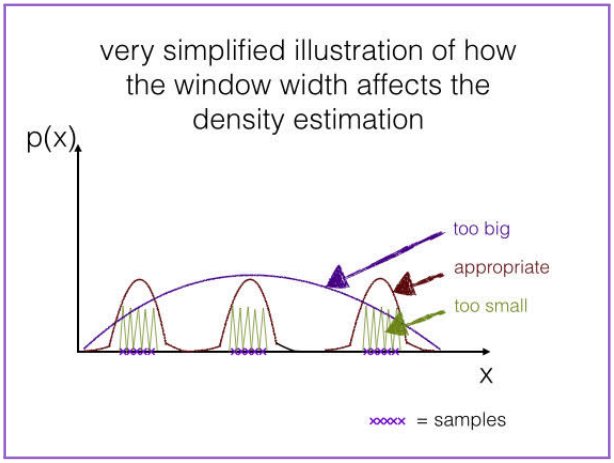
$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_n^d}\,\phi[\frac{\mathbf{x}-\mathbf{x}_i}{h_n}]$$

همان طور که در رابطه بالا دیده می‌شود، دو پارامتر اندازه پنجره و نوع کرنل مورد استفاده، قابل تنظیم هستند. کرنل های مستطیلی و گوسی معمولا بیشترین استفاده را دارند. اما در حالت کلی، نحوه انتخاب بین آن‌ها به داده‌ها وابسته است. در واقع بهترین نوع کرنل با استفاده از نتایجی که در انتها دیده می‌شود، انتخاب می‌گردد. همان طور که در اشکال زیر دیده می‌شود، انتخاب کرنل مناسب در دقت تخمین بسیار مؤثر است.



شکل ۶.۳: تفاوت کرنل گوسی و مستطیلی

در رابطه با سایز پنجره‌ها هم معمولا چندین عدد انتخاب شده و در نهایت مقداری که بهترین عملکرد را دارد انتخاب می‌گردد. در واقع باید اندازه پنجره با عکس مجذور تعداد داده‌ها نسبت عکس داشته باشد. همان طور که در شکل‌های زیر دیده می‌شود، اندازه پنجره بسیار مهم است.



شکل ۷.۳: تاثیر مقادیر مختلف پهنای‌باند بر تخمین توزیع

باتوجه به موارد بیان شده، مشکلات این روش عبارت‌اند از:

۱. یکی از بزرگ‌ترین اشکالات این روش این است که برای تخمین چگالی لازم است تمامی دیتاست را همواره در دست داشته باشیم. البته باید دقت شود که این مسئله در واقع مشکل تمامی روش‌های تخمین ناپارامتری است.

۲. باتوجه به این که در این روش تخمین بر اساس داده‌ها انجام می‌شود، برای این که به تخمینی مناسب برسیم، باید تعداد نمونه‌های کافی داشته باشیم. به صورت سرانگشتی، هرچقدر داده‌های بیشتری داشته باشیم، می‌توانیم تخمین دقیق‌تری داشته باشیم. اما همان طور که در مورد قبل اشاره شد، داده‌های زیاد هم مشکلات خود را به‌همراه دارد.

۳. بدست آوردن مقدار بهینه برای اندازه پنجره سخت است. چرا که هیچ اطلاعاتی در رابطه با توزیع داده‌ها در دست نیست.

۴. مشابه اندازه پنجره، اندازه کرنل مناسب هم مسئله سختی است.

K-Nearest Neighbors

در این روش بر خلاف روش Parzen Window که اندازه پنجره ثابت فرض شده و تعداد نقاط موجود در آن شمارش می‌شوند، اندازه پنجره آن‌قدر تغییر می‌کند تا K نقطه در آن قرار گیرد. در واقع در این روش مراجل زیر طی می‌شود:

۱. فرض می‌شود یک ابرکره داده را دربرگرفته‌است.

۲. آن‌قدر شعاع این ابرکره را افزایش می‌دهیم تا K نقطه در آن قرار گیرند.

۳. بر اساس شعاع مطلوب، حجم ابرکره به‌دست می‌آید.

۴. براساس روابط بیان شده قسمت قبلی، چگالی تخمین زده می‌شود.

در واقع به‌صورت ریاضی، اگر فرض کنیم حجم V در اطراف داده‌های X که شامل K نمونه است به طوری که k_i نمونه مربوط به کلاس w_i هست، در دست داریم. در این صورت:

$$P(w_i) = \frac{n_i}{n},\ P_n\{X|w_i\} = \frac{k_i}{n_i V} \rightarrow P_n\{X, w_i\} = \frac{k_i}{nV}$$
$$\rightarrow P_n\{w_i|X\} = \frac{P_n\{X, w_i\}}{\sum_{j=1}^{C} P_n\{X, w_j\}} = \frac{k_i}{K}$$

بنابراین، همان‌طور که دیده می‌شود، در این روش K پارامتری است که باید به‌صورت بهینه تعیین شود و باعث smooth شدن نتیجه‌ی تخمین چگالی می‌گردد.

در این روش هم چالش‌هایی که در KNN مربوط به Discrimiative Approach موجود بود، وجود دارند.

Gaussian Mixture Models

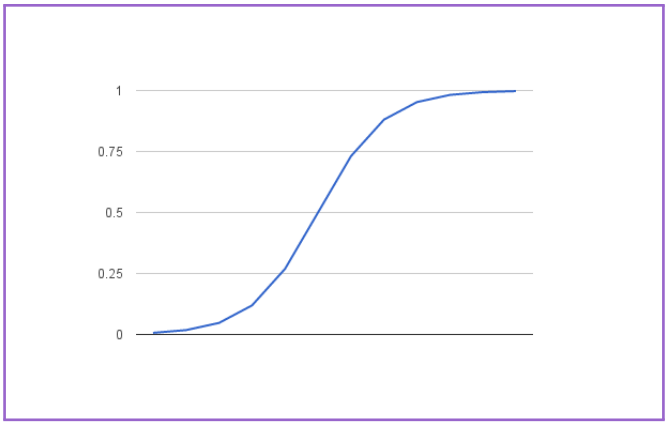
از روش GMM می‌توان برای تخمین ساختاری چگالی احتمال استفاده نمود و سپس با ترکیب چگالی تقریب زده شده و استراتژی بیز، به طبقه‌بندی دادگان بپردازیم. ایده‌ی اصلی روش GMM بر این مبناست که توابع

گوسی، توابع پایه هستند و توانایی بازسازی دقیق هر تابع و در نتیجه هر شکل هندسی مرز تصمیم را دارند. بنابراین استفاده از چند تابع گوسی به عنوان تخمین زن برای توزیع دادگان می‌توان مناسب باشد. برای محاسبه‌ی این نقاط می‌توان رویکرد عددی EM را در پیش‌گرفت. در صورتی که مسئله اصلی یک مسئله طبقه‌بندی به همراه لیبل است، می‌توان باتوجه به لیبل دادگان، به تخمین مرکز دسته هر کلاس پرداخت و از آن‌ها به عنوان نقاط اولیه‌ی الگوریتم GMM استفاده نمود. رویکرد کلی این الگوریتم بسیار مشابه رویکرد K-Means می‌باشد با این تفاوت اصلی که روش K-Means با فرم‌های هندسی به شکل ابرکره به تخمین چگالی می‌پردازد GMM از فرم‌های هندسی به شکل ابربیضی استفاده می‌نماید، که نسبت به ابرکره قابلیت انعطاف بسیار بیشتری دارد.

۲.۲.۳ Discriminative Approaches

Logistic Regression

این روش نام خود را از تابعی که ماهیت درونی آن را ایجاد کرده، وام گرفته است. تابع logistic که به آن تابع سیگموید [۱۲] هم گفته می‌شود، شکلی به صورت زیر دارد.



شکل ۸.۳: تابع سیگموید

همان طور که از شکل بالا بر می‌آید، این تابع هر مقداری را به بازه ۰ و ۱ نگاشت می‌کند.
برای Logistic Regression، مقادیر ورودی x با استفاده از وزن‌هایی با یکدیگر به صورت خطی ترکیب می‌شوند و سپس به عنوان ورودی به تابع سیگموید داده می‌شوند. مثلا برای مسئله با یک داده:

$$y = \frac{1}{1 + exp\{-(\theta_0 + \theta_1 x)\}}$$

برقرار است که در این عبارت θ_0 بایاس و θ_1 وزن را نمایش می‌دهند. در صورتی که داده‌های بیشتری داشته باشیم، با استفاده از وزن‌هایی بیشتر به ترکیب آن‌ها می‌پردازیم. شایان ذکر است که مقادیر این پارامترها با استفاده از داده‌های آموزش و تخمین maximum-likelihood بدست می‌آیند. در واقع این روش تخمین سعی می‌کند وزن‌ها و بایاس را به‌گونه ای تعیین کند که احتمال خطای پیش‌بینی را حداقل کند. پس از بدست آوردن این مقایر، با در نظر گرفتن حدی مناسب [۱۳] ، میتوان مقادیر باینری کلاس‌ها را تعیین کرد.

[12] Sigmoid
[13]

این مورد شایان ذکر است که برای استفاده از Logistic Regression فرض‌های زیر را در نظر میگیریم. (البته اگر بدون برخی از آن‌ها هم بتوان به مدلی پابدار با دقت عملکردی مناسب رسید، در نظر نگرفتن آن‌ها هم مورد قبول است.)

۱. **Binary Output Variables**: همان طور که بیان شد، از این روش برای مسائل کلاس‌بندی در حالت دو کلاسه استفاده می‌نماییم.

۲. **Remove Noise**: باتوجه به این که این روش فرض می‌کند خطا در کلاس‌بندی وجود ندارد، داده‌های آموزش باید بدون داده‌های خارج از محدوده و دارای خطا باشند.

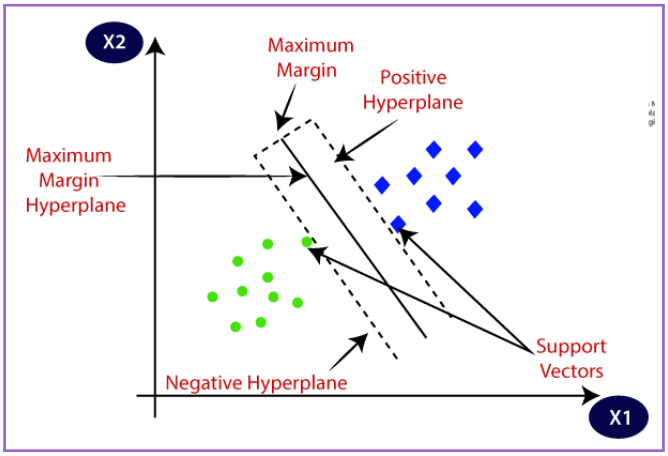
۳. **Gausssian Distribution**: این روش، الگوریتمی خطی است. در واقع Logistic Regression فرض می‌کند که رابطه‌ای خطی میان ورودی‌ها و خروجی وجود دارد. به همین علت، تبدیل‌های که این خطی بودن را به بهترین شمل نمایش دهند، بباعث عملکرد بهتری می‌شوند.

۴. **Remove Correlated Inputs**: داده‌هایی که به شدت به یکدیگر وابسته هستند، باعث overfit شدن مدل‌های بر اساس Logistic Regression می‌شوند. به علاوه این داده‌ها باعث همگرا نشدن تخمین maximum-likelihood هم می‌شوند.

Support Vactor Machines

این الگوریتم بر مبنای جداسازی دادگان کلاس‌های مختلف بر اساس بیشترین حاشیه‌ی ممکن صفحات جداکننده می‌باشد. در واقع می‌توان مسئله SVM را به صورتی مدنظر داشت که تاکید آن بر روی جداسازی دادگان در درجه اول و سپس بیشینه کردن فاصله بین این دو صفحه باشد. نمونه‌ای از مسئله مذکور که به Hard Margin SVM معروف است در شکل ۹.۳ آمده است. به طورکلی فرمولازیسیون مسئله مذکور به شرح زیر می‌باشد:

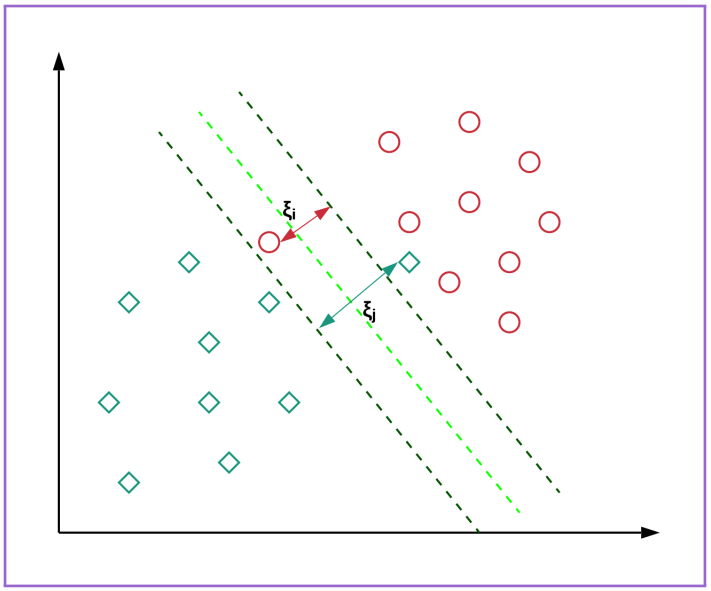
$$Minimize \frac{1}{2}||w||^2$$
$$subject\,to\, y_i(w^T x_i + b) \geq 1; \qquad \forall i \in \{1, 2, ..., C\}$$



شکل ۹.۳: مسئله SVM نامنعطف

از آنجا که اغلب دادگان همراه با نویز، خطای اندازه‌گیری، اشتباهات پیش‌آمده در امر لیبل‌گذاری و .. مواجه هستند، طبقه‌بند SVM که به صورت Hard Margin تعریف می‌شود در معرض اشتباهات مرتبط است. بدین‌ترتیب برای مقاوم نمودن طبقه‌بند نسبت به عوامل مذکور، با گسترش رویه‌ی سابق مسئله Soft Margin را تعریف می‌نماییم که در این مسئله تلاش بر ایجاد مصالحه میان اندازه‌ی حاشیه و میزان اشتباهات است. نمونه‌ای از مسئله مذکور در شکل ۱۰.۳ مشاهده می‌شود. همچنین مسئله مذکور را با تعریف یک متغیر اضافی به‌نام ζ_i می‌توان به صورت زیر مدلسازی کرد:

$$Minimize \frac{1}{2}||w||^2 + C\Sigma_{i=1}^{n}\zeta_i$$
$$subject to\, y_i(w^T x_i + b) \geq 1 - \zeta_i; \qquad \forall i \in \{1, 2, ..., C\}$$



شکل ۱۰.۳: مسئله SVM منعطف

که C یک پارامتر تنظیم کننده اهمیت درست طبقه‌بندی کردن دادگان و اندازه‌ی Margin طبقه‌بند می‌باشد. همچنین می‌توان مسئله مربوط به طبقه‌بند را به صورت دوگان و با استفاده از توابع Kernel بازنویسی نمود. به طور مثال می‌توان در نظر داشت:

$$w = \Sigma_{j=1}^{s} a_{t_j} y_{t_j} \phi(x_{t_j})$$
$$f = < w, \phi(z) > + b = \Sigma_{j=1}^{s} \alpha_{t_j} y_{t_j} K(x_{t_j}, z) + b$$

که در روابط بالا مقصود از $\phi(.)$ یک تابع Kernel است. در صورتی که این هسته (Kernel) را به صورت خطی در بگیریم، مشاهده می شود که مجددا دو مسئله اول SVM حاصل می‌شود. اما در صورت لزوم می‌توان توابعی دیگر، مانند RBF ها را مدنظر داشت، به طوریکه این توابع با انتقال مسئله به ابعاد بالاتر موجب سهولت حل مسئله در فضای جدید می‌گردند، به طوریکه هندسه‌ی اولیه‌ی داده‌های موجود در اثر انتقال به ابعاد بالاتر تفسیرپذیر و پیچیدگی کمتری پیدا خواهند کرد. اما هرکدام از این هسته‌ها مزایا و معایبی را به همراه دارند که در این بخش به بررسی آنها می‌پردازیم.

۱. Linear Kernel:

به طور کلی در مسائل(به خصوص مسائل با ابعاد بالا و دیتای کم)، ابتدا سعی می‌گردد با استفاده از توابع Kernel به فرم خطی، به حل مسئله پرداخته شود. در صورتی که دادگان به صورت خطی جداپذیر باشند این روش به خوبی عمل می‌نماید، به خصوص آنکه روش خطی به دلیل سادگی و پیچیدگی کمتر، عملا به دیتای کمتری برای آموزش احتیاج دارد. از طرفی اگر که دادگان به صورت خطی جداپذیر نباشند این روش موثر نخواهد بود.

۲. RBF Kernel:

در صورتی که دادگان به صورت خطی جداپذیر نباشند و طبقه‌بند با هسته خطی جواب مناسبی را در بر نداشته باشد، می‌توان از توابع RBF به عنوان Kernel طبقه‌بند استفاده نمود. به صورت تئوری ثابت می‌گردد که هر تابعی توسط توابع پایه‌ی RBF به طور دقیق قابل ساخت است و بنابراین هر مرز تصمیم به شکل دلخواهی را می‌توان با استفاده از توابع مذکور بازسازی کرد. اشکال این روش‌ها آنست که در حضور دیتای کم ممکن است به خوبی جواب ندهند و به سرعت Overfit پیش‌بیاید.

Decision Tree

این الگوریتم جزو خانواده الگوریتم‌های یادگیری بانظارت می‌باشد. اما برخلاف سایر اعضای این خانواده، هم برای مسئله رگرسیون و هم طبقه‌بندی کاربرد دارد. در درخت تصمیم،برای ویش‌بینی لیبل یک داده، با ریشه درخت شروع می‌کنیم. سپس، با قیاس‌های مناسب، به شاخه‌ها می‌رسیم. در واقع در این درخت‌ها، نمونه‌ها از ریشه تا برگ‌ها یا نودهای پایانی مرتب می‌شوند و به این ترتیب به طبقه‌بندی می‌رسیم. هر نود در درخت، یک روش تست برای یک ویژگی بوده و تمامی قسمت‌های بعدی جواب‌های ممکنه برای این تست هستند.
البته ساختن این درخت با مسائل بیان شده نیازمند فرض‌هایی نیز هست. این فرض‌ها عبارتند از:

۱. در ابندا، تمامی داده‌های آموزش به عنوان ریشه‌ی درخت درنظر گرفته می‌شوند.

۲. ویژگی‌ها به صورت categprical هستند. البته اگر مقادیر پیوسته داشته باشند، قبل از ساختن مدل باید گسسته شوند.

۳. ساختن درخت به صورت بازگشتی و بر اساس مقدار ویژگی‌ها در نظر گرفته می‌شوند.

۴. براساس بررسی‌های آماری می‌توان ترتیب ویژگی‌ها را که به عنوان شاخه یا ریشه درنظر گرفته شوند را مشخص کرد.

این درخت‌ها، از الگوریتم‌های مختلفی برای تصمیم‌گیری مبنی بر تقسیم نودها استفاده می‌کنند. انتخاب الگوریتم مناسب وابسته به نوع لیبل‌هایی که هدف پیش‌بینی هستند، می‌باشد.
همان طور که گفته شد، انتخاب این که کدام ویژگی را به عنوان ریشه در نظر بگیریم عملی مهم و در عین حال سخت می‌باشد. اگر به صورت رندوم یک ویژگی انتخاب شود، ممکن است نتیجه دقت بسیار پایینی داشته و با عملکرد بدی مواجه شویم. بنابراین، برخی از روش های زیر برای این کار عبارت‌اند از:

۱. Entropy: این معیار میزان رندوم بودن در اطلاعات را بیان می‌کند. هرچه میزان آن بالاتر باشد، رسیدن به نتیجه با استفاده از داده‌ی موجود سخت تر می‌گردد.

۲. Information Gain: این معیار یک ویژگی آماری است که بیان می‌کند ویژگی‌های داده شده تا چه حد داده‌های آموزش را به خوبی و بر اساس هدف طبقه‌بندی، تقسیم می‌کنند.

در اصل این معیار نشان‌دهنده کاهش آنتروپی است. چراکه میزان اختلاف آنتروپی قبل از تقسیم و میانگین آن بعد از تقسیم بر اساس یک ویژگی را محاسبه مینماید. بنابراین، در ساختن درخت، هدف کاهش مقدار آنتروپی و افزایش information gain است.

$$Information\ Gain = Entropy(before) - \sum_{j=1}^{K} Entropy(j,\ after)$$

۳. Gini Index: این معیار را به عنوان یک تابع هزینه که برای ارزیابی تقسیم‌بندی در دیتاست مورد استفاده قرار می‌گیر، میتوان در نظر گرفت.

$$Gini = 1 - \sum_{i=1}^{n} (p_i)^2$$

۴. Gain Ratio: معیار information gain به سمت ویژگی‌هایی که بیشترین مقدار ایم معیار را دارند بایاس است و آن‌ها را به عنوان نودهای ریشه در نظر می‌گیرد. اما gain ratio این مقدار بایاس را کمتر داشته و معمولا معیار بهتری است. در واقع در این روش مشکل information gain با در نظر گرفتن تعداد شاخه‌هایی که در نتیجه‌ی تقسیم ایجاد می‌شوند، رفع می‌شود.

$$Gain\ Ratio\ =\ \frac{Information\ Gain}{Split\ Information}\ = \frac{Entropy(before) - \sum_{j=1}^{K} Entropy(j,\ after)}{\sum_{j=1}^{K} w_j log_2 w_j}$$

باید دقت شود که یکی از مشکلات معمول در درخت‌های تصمیم overfitting است. حتی در برخی موارد به‌نظر می‌رسد که درخت داده‌های آموزش را به طور کامل حفظ کرده است. به عبارت دیگر، اگر هیچ محدودیتی برای درخت تصمیم قرار داده نشود، دقت آن روی داده‌های آموزش ۱۰۰ درصد می‌شود، چرا که حتی ممکن است برای هر مشاهده یک برگ در نظر گرفته شود. برای رفع این مشکل، از دو راه استفاده می‌شود:

۱. هرس کردن

۲. random forrest

از جمله مزایای درخت تصمیم می‌توان به موارد زیر اشاره کرد:

۱. به سادگی قابل فهمیدن و پیاده‌سازی هستند.

۲. هم داده‌های عددی و هم categorical را می‌توانند مورد بررسی قرار دهند.

۳. در برابر داده‌های خارج از محدوده مقاوم هستند. به همین علت نیازمند پیش‌پردازش کمی می‌باشند.

در برابر ویژگی‌ها و مزایای بیان شده بالا، این درختان معایب زیر را نیز دارا هستند:

۱. نسبت به overfitting حساس هستند و تمایل به overfit شدن روی داده‌ها را دارند.

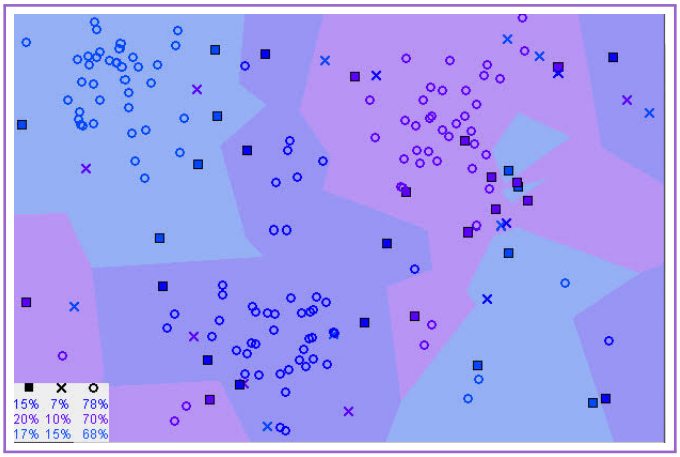
۲. نیازمند معیاری هستند که بیان کند به چه میزات عملکرد مناسبی دارند.

۳. در انتخاب پارامترها و تنظیم آن‌ها باید بسیار دقت کرد.

۴. اگر برخی از کلاس‌ها نسبت به سایر آن‌ها برتری داشته باشند، درخت دارای بایاس می‌شود.

K-Nearest Neighbors

این الگوریتم، روشی ساده، با پیاده‌سازی راحت، و همین‌طور یادگیری بانظارت [۱۴] می‌باشد که از آن هم در مسائل کلاس‌بندی و هم regression استفاده می‌گردد. در این الگوریتم فرض می‌شود داده‌هایی که مشابه هستند در نزدیکی یکدیگر قرار دارند.



شکل ۱۱.۳: استفاده از طبقه‌بند نزدیک‌ترین همسایه

همان‌طور که در شکل بالا دیده می‌شود، در اکثر مواقع، داده‌های مشابه در نزدیکی یکدیگر قرار دارند و برای قابل استفاده بودن الگوریتم KNN، این موضوع باید تا حد خوبی برقرار باشد. در واقع این الگوریتم، مفهوم مشابهت را با استفاده از ریاضیاتی که در کودکی مطالعه کردیم، مانند فاصله اقلیدسی، مورد بررسی قرار می‌دهد.
البته، همان طور که میدانیم، روش‌های مختلفی برای بدست آوردن فاصله بین نقاط وجود دارد. از جمله آن‌ها می توان به موارد زیر اشاره کرد:

۱. Minkowski:

$$d = \left(\sum_{i=1}^{k} (|x_i - y_i|)^q \right)^{\frac{1}{q}}$$

این روش معمولا در ابعاد بالا دچار مشکل می‌گردد. در واقع این متریک دچار curse of dimensionality می‌شود. به علاوه، معمولا به دست آوردن مقدار q مناسب می‌تواند از نظر هزینه‌ای به‌صرفه نباشد.

۲. Euclidean: (همان فاصله Minkowski با $q = 2$ است.)

$$d = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

فاصله اقلیدسی در واقع کوتاه‌ترین فاصله میان دو نقطه است. اما دارای معیایبی می‌باشد. از جمله آن‌ها این است که قبل از استفاده از آن توصیه می‌گردد که داده‌ها را نرمالیزه نماییم، چرا که به مقایس داده‌های مختلف حساس است. به علاوه این روش، برای داده با ابعاد زیاد دچار مشکل می‌گردد. به همین علت معمولا برای هنگامی‌که بعد داده‌ها زیاد نباشد، مورد استفاده است.

[14] Supervised learning

۳. Manhattan: (همان فاصله Minkowski با $q = 1$ است.)

$$d = \sum_{i=1}^{k} |x_i - y_i|$$

مشابه روش اقلیدسی، این روش هم برای داده‌ها با ابعاد بالا دچار مشکل می‌گردد. به‌علاوه، به این علت که برخلاف فاصله اقلیدسی کمترین فاصله را گزارش نمی‌کند، می‌تواند مقادیر بزرگ‌تری را بیان کند. به‌همین علت، معمولا در مواردی که ویژگی‌های باینری ویا گسسته داریم، این روش متریک مناسب‌تر است.

با توجه به مسئله‌ای که می‌خواهیم به حل آن بپردازیم، انتخاب نوع فاصله انجام می‌شود. با این حال فاصله اقلیدسی بسیار محبوب است.

الگوریتم KNN شامل مراحل زیر می‌شود:

۱. داده‌ها لود شده و عملیات پیش‌پردازش مناسب روی آن‌ها انجام می‌گردد.

۲. مقدار موردنظر K تعیین می‌گردد.

۳. برای هر داده فاصله سایر داده‌ها تا آن محاسبه شده و این مقدار در یک مجموعه ترتیبی نگه‌داری می‌گردد.

۴. مجموعه مذکور بر اساس فاصله کم به زیاد مرتب می‌شود.

۵. از مجموعه مرتب شده K نمونه اول انتخاب می‌گردد.

۶. لیبل‌های این نمونه‌ها استخراج می‌شوند.

۷. در مسئله regression، میانگین این K لیبل و در مسئله طبقه‌بندی، مود آن‌ها را به عنوان خروجی بیان می‌گردد.

همان طور که از توضیحات بالا برمی‌آید، تعیین مقدار K مسئله مهمی است. یکی از روش‌های انتخاب بهترین مقدار، اجرای الگوریتم به ازای مقادیر مختلف K و انتخاب مقداری که باعث کوچکترین خطا می‌گردد، است. یکی دیگر از روش‌های مناسب برای انتخاب K استفاده از روش cross-validation می‌باشد. اما باید نکات زیر را مدنظر قرار دهیم:

۱. معمولا کاهش مقدار K به یک، باعث کاهش پایداری پیش‌بینی می‌گردد.

۲. برعکس حالت قبل، افزایش مقدار K باعث افزایش پایداری پیش‌بینی و در نتیجه افزایش دقت می‌شود.

۳. در مواردی که از majority voting استفاده می‌شود، سعی می‌شود مقدار K عددی فرد اتخاذ گردد تا به این ترتیب حالت تساوی بین لیبل‌ها رخ ندهد.

از جمله مزایای این روش می‌توان به موارد زیر اشاره کرد:

۱. همان طور که اشاره شد، این الگوریتم به سادگی پیاده‌سازی شده و به راحتی قابل فهم است.

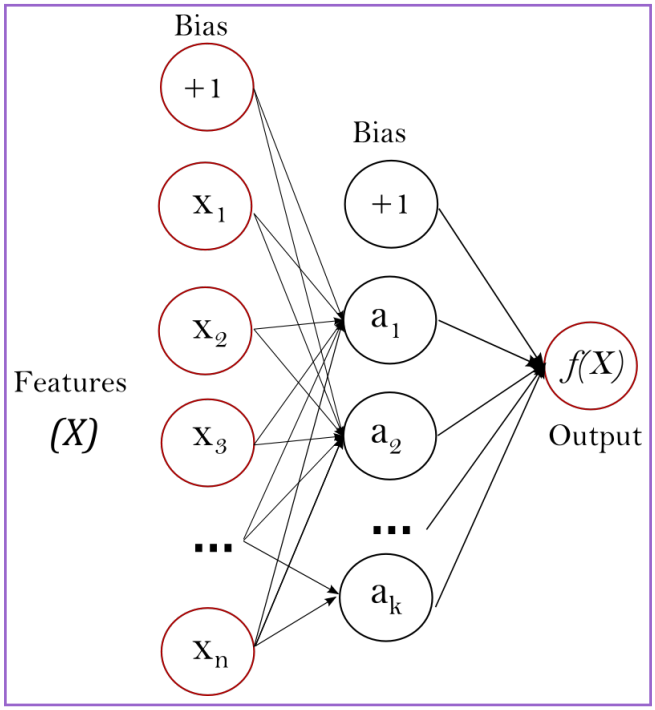
۲. در این روش نیازی به ساختن مدل، تنظیم پارامترهای مختلف و فرض‌های زیاد نیست.

۳. این الگوریتم به نوعی همه‌کاره است. چراکه همان‌طور که اشاره شد، هم در مسائل طبقه‌بندی، regression و جستجو قابل استفاده است.

اما در برابر مزایای اشاره شده، باید ذکر کرد که این روش با افزایش سایز داده‌ها به‌شدت کند می‌گردد.

Multi-Layer Perceptron

این روش، یکی از روش‌های یادگیری بانظارت در یادگیری ماشین است که یک سعی می‌کند تابع $f(.) : R^m \rightarrow R^o$ را بر اساس داده‌های آموزش، یادبگیرد. در رابطه‌ی بیان شده، m ابعاد ورودی و o ابعاد خروجی است. در واقع اگر ویژگی‌ها را به صورت $X = x_1,\ x_2,\ ..., x_m$ و هدف را y در نظر بگیریم، در این روش تخمین غیر خطی‌ای یاد گرفته می‌شود که هم برای طبقه‌بندی و هم برای رگرسیون قابل استفاده است. اما باید دقت شود که این مسئله با روش logistic regression که قبلا به توضیح آن پرداختیم متفاوت است. چراکه در MLP می‌توان بین لایه ورودی و خروجی، یک یا چند لایه غیرخطی که به آن‌ها hidden layer گفته می‌شود، قرار داد.



شکل ۱۲.۳: یک MLP با یک لایه مخفی

در شکل ۱۲.۳ ، یک شبکه‌ی MLP به همراه یک hidden layer قابل مشاهده است. در این شکل، چپ‌ترین لایه که لایه‌ی ورودی نیز نامیده می‌شود، از یک مجموعه از نورون‌های $\{x_i|x_1,\ x_2,\ ...,\ x_m\}$ که نمایش‌دهنده ویژگی‌های ورودی‌اند، تشکیل شده است. در hidden layer ، هر نورون مقادیر لایه قبلی را با یک تبدیل خطی به فرم $w_1x_1 + w_2x_2 + ... + w_mx_m$ و پس از آن یک فعال‌سازی غیرخطی به فرم $g(.) : R \rightarrow R$ به مقادیر جدیدی تبدیل کرده و به لایه بعد می‌فرستد. در لایه‌ی خروجی، مقادیر آخرین hidden layer گرفته شده و به خروجی تبدیل می‌شوند. از جمله مزایای این روش می‌توان به موارد زیر اشاره کرد:

۱. توانایی یادگیری مدل‌های غیرخطی

۲. توانایی یادگیری مدل‌ها به صورت بلادرنگ

اما در برابر مزایای اشاره شده، معیاب زیر را نیز می‌توان بیان کرد:

۱. روش MLP با hidden layers تابع هزینه‌ی غیرمحدبی دارد که می‌تواند بیشتر از یک مینیمم محلی داشته باشد. بنابراین مقداردهی اولیه متفاوت برای وزن‌ها می‌تواند باعث دقت‌های عملکردی متفاوت در مدل‌ها شود.

۲. در این روش نیازمند تنظیم پارامترهای زیادی از جمله تعداد hidden neurons و تعداد لایه‌هاست.

۳. این روش به اسکیل ویژگی‌ها بسیار حساس است.

RBF Neural-Networks

این دسته طبقه‌بندها ساختاری مشابه شبکه‌های MLP را تداعی می‌نمایند، تنها با این تفاوت که در شبکه‌های RBF از توابع فعالساز RBF استفاده می‌گردد. این توابع به دلیل آنکه تنها به فاصله بستگی دارند و از طرفی توابع پایه نیز می‌باشند(هر تابع خوش‌تغریفی با استفاده از توابع مذکور قابل بازیابی هستند)، می‌توان ادعا نمود که مانند شبکه‌های MLP می‌توان هر تابعی را با استفاده از این شبکه‌ها بازسازی نمود اما از طرف دیگر نسبت به نویز مقاوم هستند. برای استفاده از این شبکه‌ها مانند ساختارهای متداول شبکه‌های عصبی به آموزش شبکه پرداخته و در نهایت از شبکه مورد استفاده برای تشخیص دادگان دیده نشده

۳.۳ ارزیابی

در این مرحله، روش‌های مختلف طبقه‌بندی به همراه پارامترهای تنظیم شده آن‌ها مورد ارزیابی قرار می‌گیرند تا به این ترتیب راه‌حل بهینه برای جداسازی و طبقه‌بندی دیتاست داده شده یافت گردد. برای ارزیابی، روش‌های مختلفی وجود دارد که از میان آن‌ها، میزان دقت مدل، مستقیم‌ترین راه است. اما از اصلی‌ترین معایب آن می‌توان به عدم توجه به انواع خطای ایجاد شده و وابستگی آن به توزیع کلاس‌ها اشاره کرد. به همین علت، در این بخش، علاوه بر توضیحات گسترده‌تر دقت، به روش‌های ارزیابی دیگر هم پرداخته شده و در بخش پیاده‌سازی، مقادیر آن‌ها برای مقایسه گزارش می‌گردد.

۱.۳.۳ Accuracy

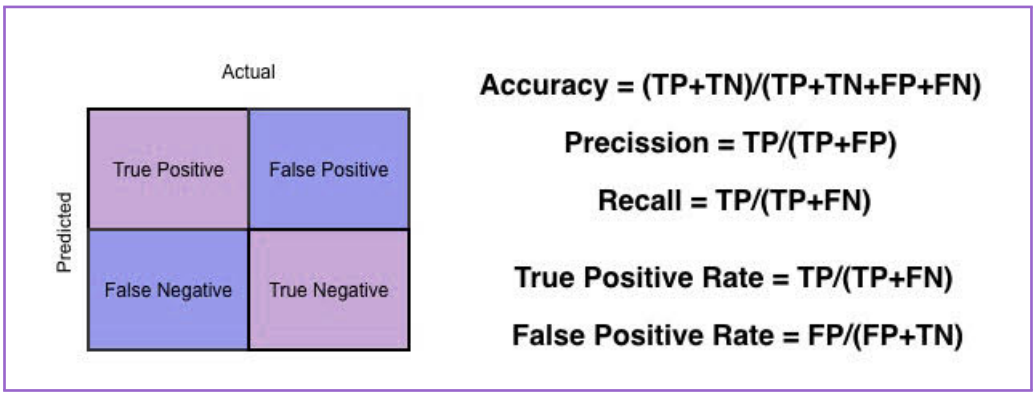
همان طور که قبل‌تر اشاره شد، این معیار بیان می‌کند که طبقه‌بند طراحی شده، هر چند وقت پیش‌بینی درستی می‌کند. در واقع:

$$Accuracy = \frac{number\ of\ correctly\ classified\ examples}{total\ number\ of\ cases}$$

معایب این روش قبل‌تر اشاره شد. از جمله آن‌ها عدم توانایی این روش در بیان نوع خطاست. این مسئله در برخی کاربردهای عملی و به طور مثال تشخیص بیماری بسیار اهمیت دارد. از این رو، برای ارزیابی مدل به این روش اکتفا نمی‌شود.

۲.۳.۳ Confusion Matrix

در مواردی نیازمند تعیین انواع خطاها هستیم، به‌دست آوردن ماتریس آشفتگی راهکار مناسب است. در این روش هر سطر بیان‌کننده‌ی یک کلاس است و اعداد موجود در هر درایه بیان کننده کلاس تخمین زده شده است. به بیان دیگر، اعداد روی قطر اصلی نمایش‌دهنده تخمین‌های درست و سایرین بیان کننده خطاهای پیش‌بینی است. به همین علت است که این روش محدود به مسئله‌های دوکلاسه نمی‌باشد و قابلیت ارزیابی سایرین را نیز دارد. در اینجا برای بیان دقیق‌تر مسائل بالا، این ماتریس را برای یک مسئله دو کلاسه بیان می‌کنیم.



شکل ۱۳.۳: ماتریس آشفتگی و خطاهای قابل محاسبه

برای مثال فرض می‌کنیم که ماتریس بالا برای پیش‌بینی یک بیماری در نظر گرفته شده است. به این ترتیب:

۱. **True Positive**: این مورد مربوط به بیمارانی می‌شود که بیمار بوده و به درستی هم بیمار پیش‌بینی شده اند.

۲. **True Negative**: این مورد مربوط به افرادی می‌شود که سالم بوده و به درستی سالم پیش‌بینی شده اند.

۳. **False Positive**: این مورد مربوط به افرادی می‌شود که سالم بوده اما به شکل نادرستی بیمار شناسایی شده اند.

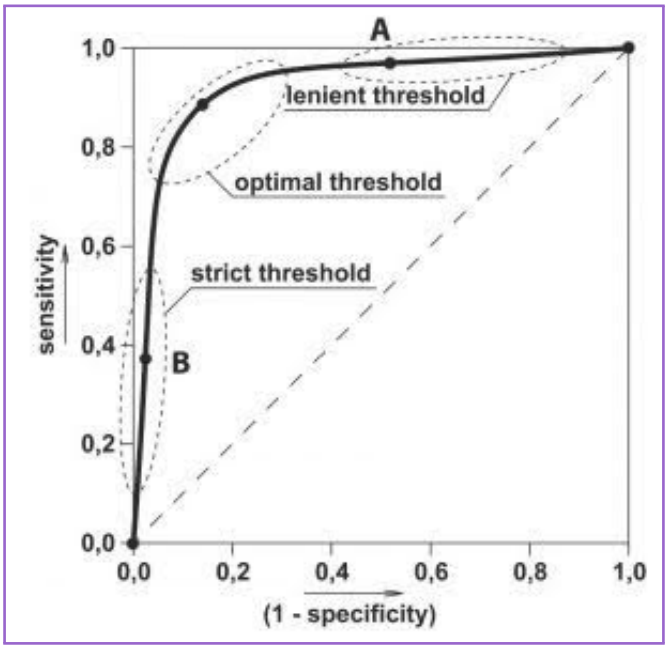
۴. **False Negative**: این مورد مربوط به افرادی می‌شود که بیمار بوده اما به شکل نادرستی سالم پیش‌بینی شده اند.

با توجه به موارد بیان شده بالا، خطاهای بیان شده در تصویر تعریف می‌گردند. بر اساس این که هدف از طبقه‌بندی چیست، می‌توان از هر یک از معیارهای بالا برای ارزیابی مدل استفاده کرد.

۳.۳.۳ ROC Curve

یکی از معیارهایی که با استفاده از آن می‌توان میزان Sesitivity یا True Positive در برابر میزان Fall-out یا False Positive مورد بررسی قرار داد، نمودار ROC است. در واقع با استفاده از این نمودار می‌توان متوجه شد

که مدل طراحی شده تا چه میزانی درست است. در واقع با استفاده از تعریف مقدار threshold مناسب میتوان به موضوع بیان شده پی برد. این threshold بیان می‌کند که لیبل‌های True و False در چه بازه‌ای قرار گیرند تا بیشترین مطابقت را با دیتاست مورد بررسی داشته باشند. در شکل ۱۴.۳ به خوبی میتوان تاثیر آن را مشاهده کرد.



شکل ۱۴.۳: نمودار ROC به همراه THRESHOLDهای مختلف

در این نمودار، محور افقی یا x نماینده False Positive Rate و محور عمودی یا y نماینده True Positive Rate است. در این نمودار نقطه (۱و۰) نمایش‌دهنده یک طبقه‌بند عالی است. چراکه این طبقه‌بند تمامی داده‌ها را به درستی کلاس‌بندی کرده است. این موضوع را بر این اساس می‌توان متوجه شد که میزان False Positive Rate برابر صفر و میزان True Positive Rate برابر یک است. در برابر آن نقطه (۰و۱) قرار دارد. باتوجه به توضیحات بیان شده، واضح است که در این حالت طبقه‌بند همواره اشتباه کلاس‌بندی می‌کند. نقطه (۰و۰)، نقطه ای است که تمام کلاس‌ها را به عنوان Negative مشخص کرده است. در برابر آن نقطه (۱و۱) قرار دارد که تمامی نقاط را به عنوان Positive در نظر می‌گیرد. بنابراین، واضح است که هرچه مساحت زیر نمودار کاهش پیدا کند، مدل کم‌تر دقیق است.

این معیار در مواردی که میخواهیم میزان عملکرد موفق مدل را مورد بررسی قرار دهیم اهمیت بسیاری پیدا می‌کند. در واقع این معیار یکی از اولین معیارهایی است که در بررسی عملکرد اکثر الگوریتم‌ها مورد بررسی قرار می‌گیرد چرا که به کمک آن می‌توان مشکلات مدل را به زودترین شکل ممکنه دریافت.

۴.۳.۳ Area Under Curve (AUC)

مساحت زیر سطح نمودار ROC بیان‌گر این معیار است. براساس تعاریف بیان شده برای نمودار ROC، واضح است که هر چقدر میزان AUC به یک نزدیک‌تر باشد، دقت طبقه‌بند بالاتر است.

در واقع این معیار را به عنوان تک عددی که بیان کننده عملکرد طبقه‌بند است به‌شمار می‌آورند. اما برتری‌ای که نسبت به Accuracy دارد در این است که در مواردی که دو کلاس نسبت به یکدیگر بالانس نیستند هم این

معیار می‌تواند ارزش‌یاب مناسبی باشد.

F1-Score ۵.۳.۳

این معیار هم یک تک عدد است که معیارهای Precision و Recall را به وسیله‌ی Harmonic Mean این دو بیان می‌کند. به صورت ریاضی:

$$F1 - Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

فصل ۴

پیاده سازی مدل‌های Discriminative

در این فصل سعی می‌گردد با استفاده از طبقه‌بندها و روش‌های پیش‌پردازش که در بخش قبل ذکر شد به طراحی مدل‌های مختلف پرداخته بشود و سپس از منظر عملکرد این مدل‌ها مورد بررسی قرار بگیرند. برای این مهم سعی می‌گردد ترکیب‌های مناسب و بهینه که اغلب با تحلیل و شهود حاصل می‌شود مورد بررسی قرار بگیرند، سپس سرعت آموزش مدل، دقت آموزش و جامعیت آن‌ها نیز به عنوان معیارهایی از عملکرد بهینه مطرح می‌گردند. همچنین باتوجه به حجم کم دیتا، پایداری مدل‌ها نسبت به فرآیند آموزش با حجم کم داده‌ها از جمله موارد مورد بحث خواهد بود. در ادامه مدل‌ها بر اساس طبقه‌بند مورد استفاده، مانند درخت تصمیم، SVM و... دسته‌بندی شده و سپس برای هر طبقه‌بند مجموعه پیش‌پردازش‌های پیشنهادی که باتوجه به ذات مسئله و طبقه‌بند می تواند موثر باشد اعمال می‌گردد و سپس نتایج آن مورد بررسی قرار خواهد گرفت.

۱.۴ پیاده سازیSVM

۱.۱.۴ پیش‌پردازش در SVM

ابتدا داده‌ها را به دو دسته‌ی تست و آموزش تقسیم می‌نماییم. در واقع ۲۰ درصد داده‌ها به تست و ۸۰ درصد دیگر به آموزش اختصاص می‌یابند. سپس مطابق مطالب بیان شده در بخش دیتاست، با استفاده از ماژول SMOTE در پایتون به بالانس کردن دیتا در بخش آموزش و تست می‌پردازیم. همانطور که ذکر شد این روش بالانس کردن دیتا می‌تواند موجب کورولیشن بین دادگان کلاسی شود که دیتای کمتری از آن در دست است. باتوجه به مطالب مطرح شده در ۲.۲، طبقه‌بند SVM می‌تواند در آستانه Overfitting نسبت به دیتاست موجود قرار بگیرد. بنابراین اهمیت کاهش ابعاد و پیچیدگی مسئله در این طبقه‌بند حائز اهمیت است.در این بخش به دلایل مطرح شده، ابتدا با استفاده از یکه‌سازی استاندارد (Standard Scaler) سعی در سفید کردن دیتا داریم و در واقع با تبدیل توزیع دیتا از حالت بیضی به دایره، سعی می‌کنیم پیچیدگی‌های پارامترهای مسئله را کاهش دهیم.
سپس با استفاده از دو روش کاهش بعد PCA و Autoencoder سعی می‌نماییم که بعد مسئله را پایین بیاوریم:

۱. PCA: ابتدا مولفه‌های اصلی [1] را محاسبه نموده، سپس با انتخاب ۲۵۴ مولفه اول، مشاهده می‌نماییم که

[1] Principal Components

واریانس فضای نهایی برابر با ۹۰ درصد واریانس اولیه است. بدین‌ترتیب فضای اولیه را که حدود ۷۵۰ بعد داشت را به تنها ۲۵۴ بعد کاهش دادیم.

۲. Autoencoder: یک شبکه‌ی Autoencoder با یک لایه‌ی میانی میسازیم به طوریکه در لایه‌ی میانی داری ۵۰ نود باشد. سپس بر مبنای تابع هزینه حداقل مربعات و با استفاده از الگوریتم SGD به آموزش شبکه برای ۲۰۰ ایپاک، با سایز دسته‌های [۲] ۱۶ می‌پردازیم(تعداد ایپاک و سایز دسته‌ها با استفاده از آزمون و خطا محاسبه گردید). بدین ترتیب مشاهده می‌نماییم که با استفاده از پیش‌پردازش انتظار داریم بعد مسئله به ۵۰ کاهش بیابد. به دلیل دیتای کم انتظار نتایج ناپایدار را از شبکه داریم.

۳. LDA: این روش برای طبقه‌بند SVM عملکرد مناسبی نداشت. در واقع به دلیل آنکه تعداد کلاس‌ها تنها برابر با ۲ بوده است، عملا LDA سعی در کاهش بعد مسئله به یک دارد که این تبدیل فضا از ۷۵۴ به ۱ با خطای زیادی همراه است.

۴. BFS: این روش بسیار زمان‌بر می‌باشد، به دلیل حجم محاسبات بسیار زیاد، حتی بعد از کاهش بعد قابل توجه PCA نمی‌توان از آن برای بررسی جامع استفاده نمود. همچنین در حین فرآیند مشاهده شد که استفاده محدود و جزئی از آن (برای مثال کنار گذاشتن ۱۰ ویژگی کمتر موثر) تاثیر چندانی بر عملکرد کلی مدل نداشته است، بنابراین از این روش استفاده نشده است.

۵. ICA:این روش نیز سعی در حذف تمامی وابستگی‌ها و از بین بردن اثر اسکیل دارد. از این روش به دلیل لزوم کاهش بعد در این بخش استفاده نمی‌شود اما در سایر بخش‌ها مورد استفاده قرار گرفته است.

۲.۱.۴ طراحی طبقه‌بند SVM

همانطور که در بخش ۲.۲ مطرح شد، در تنظیم و طراحی این طبقه‌بندها تابع هزینه به صورت زیر می‌باشد:

$$min\ \frac{1}{2}||w||^2 + C\Sigma_{i=1}^{n}\zeta_i$$

بنابراین یکی از پارامترهای مورد بحث در این طبقه‌بند تعیین پارامتر C می‌باشد. همچنین از دیگر موارد مهم در طراحی این طبقه‌بند استفاده از هسته مناسب می‌باشد. در مسئله جاری دو هسته Linear و RBF انتخاب می‌شوند. هسته خطی در صورتی که داده‌ها به صورت خطی جداپذیر باشند عملکرد مناسبی خواهد داشت و می‌تواند در حضور دیتای کم به خوبی آموزش ببیند و به جداسازی مسئله بپردازد، همچنین هسته RBF با انتقال بعد طبقه‌بند به بعد نامحدود، توانایی تخمین هر مرز تخمینی را دارد(توابع RBF به عنوان توابع پایه از نظر تئوری توانایی ساخت دقیق توابع را دارند). باتوجه به بعد بالای مسئله امکان نمایش دیتا و بررسی رفتار آن وجود ندارد به همین دلیل باید با استفاده از آزمون و خطا و بررسی خطای Cross Validation به تعیین مدل بهینه بپردازیم.

۳.۱.۴ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم.

[2] Batch

پیش‌پردازش PCA

پارامترهای مسئله در ماژول grid به صورت شکل ۱.۴ انتخاب می‌شوند.

```
grid = {'kernel': ['rbf', 'linear'],
        'gamma': [ 1e-3, 2*(1e-3), 3*(1e-3), 4*(1e-3)],
        'C': np.arange(1, 1000, 100).tolist()}
```

شکل ۱.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۲.۴ می‌باشند:

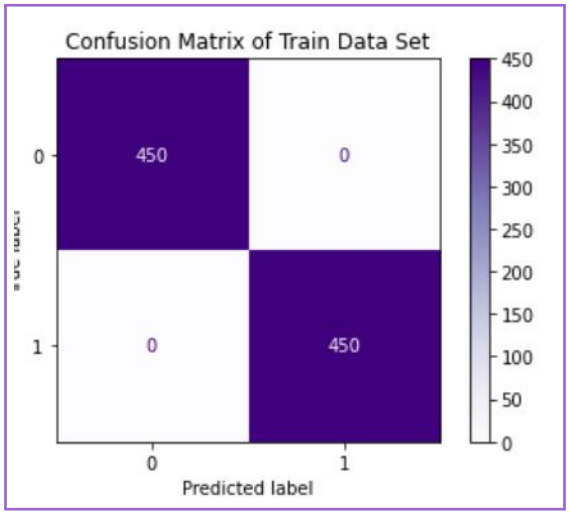
```
Tuned hpyerparameters are: {'C': 101, 'gamma': 0.002, 'kernel': 'rbf'}
```

شکل ۲.۴: پارامترهای بهینه مدل

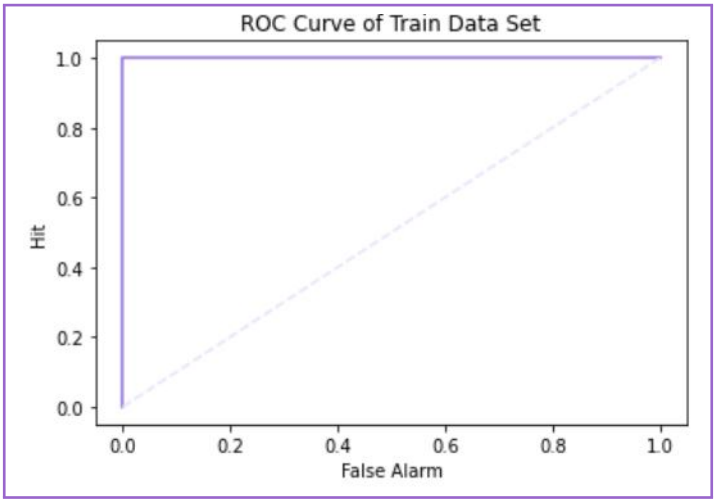
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	97.11±2.64	1	1	1

شکل ۳.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش PCA بر روی داده‌های آموزش



شکل ۴.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش PCA بر روی داده‌های آموزش

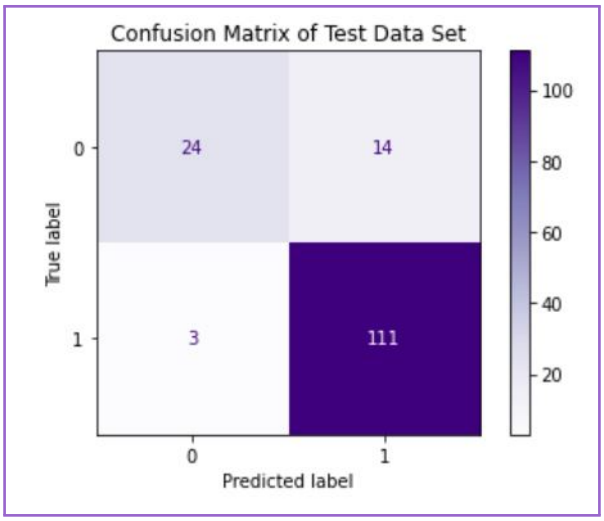


شکل ۵.۴: منحنی ROC طبقه‌بند SVM با پیش‌پردازش PCA

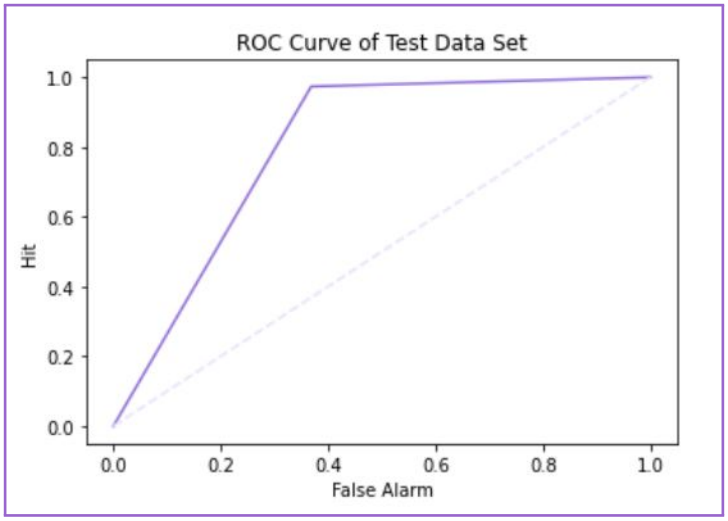
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	88.82	0.74	0.93	0.8

شکل ۶.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۷.۴: ماتریس CONFUSION طبقه‌بند SVM با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۸.۴: منحنی ROC طبقه‌بند SVM با پیش‌پردازش PCA بر روی داده‌های تست

پیش‌پردازش **Autoencoders**

حال مانند بخش قبل عمل می‌نماییم اما بجای پیش‌پردازش PCA از پیش‌پردازش Autoencoder با ۵۰ نود میانی استفاده می‌نماییم. بنابراین مشاهده می‌گردد که بعد مسئله به ۵۰ کاهش یافته است. مانند بخش قبل با استفاده از روش Gridsearch سعی در پیدا کردن مدل بهینه داریم. ابتدا ماژول grid را با پارامترهای مطرح شده در شکل ۹.۴ در نظر می‌گیریم.

```
grid = {'kernel': ['rbf', 'linear'],
        'gamma': [ 1e-3, 2*(1e-3), 3*(1e-3), 4*(1e-3)],
        'C': np.arange(1, 1000, 100).tolist()}
```

شکل ۹.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۱۰.۴ می‌باشند:

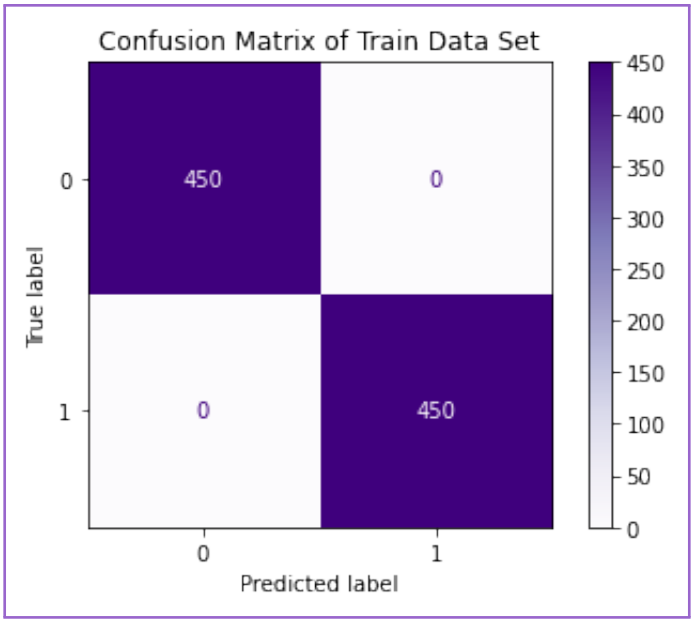
```
Tuned hpyerparameters are: {'C': 101, 'gamma': 0.004, 'kernel': 'rbf'}
```

شکل ۱۰.۴: پارامترهای بهینه مدل

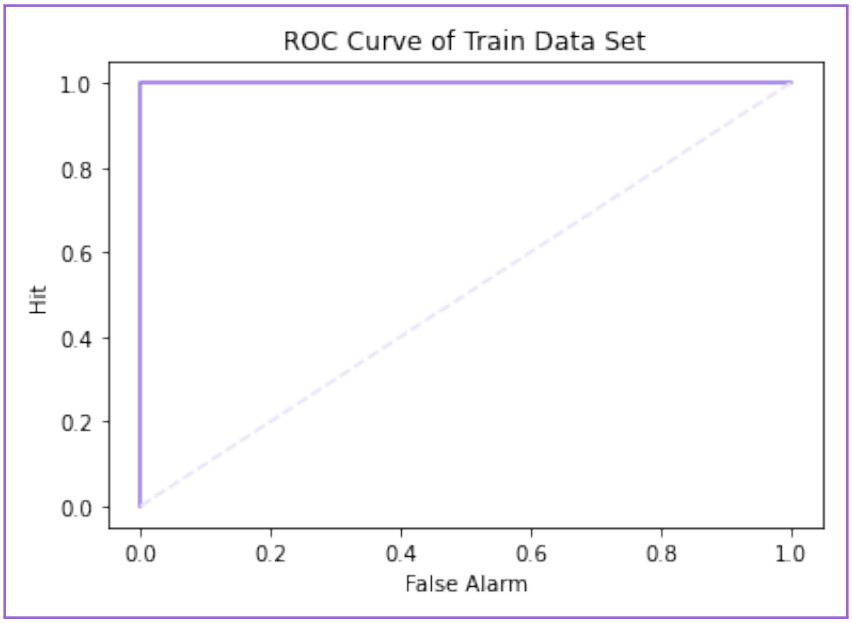
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	95.44±2.92	1	1	1

شکل ۱۱.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۱۲.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش

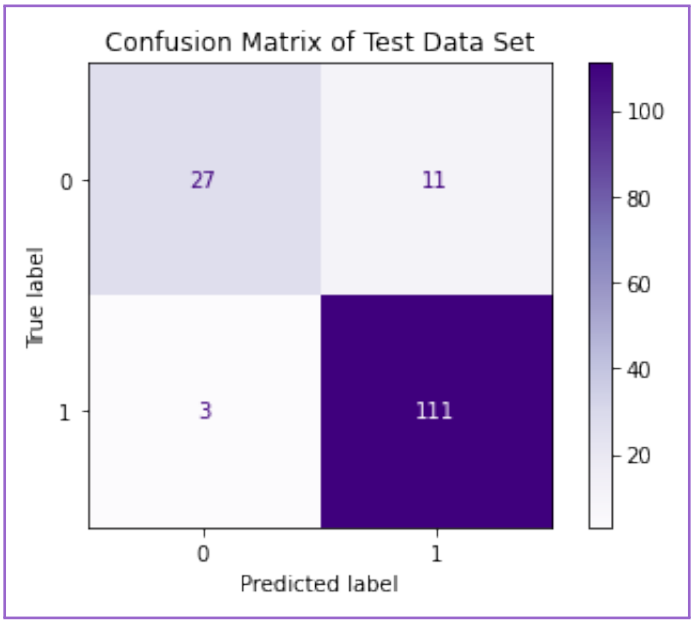


شکل ۱۳.۴: منحنی ROC طبقه‌بند SVM با پیش‌پردازش AUTOENCODER

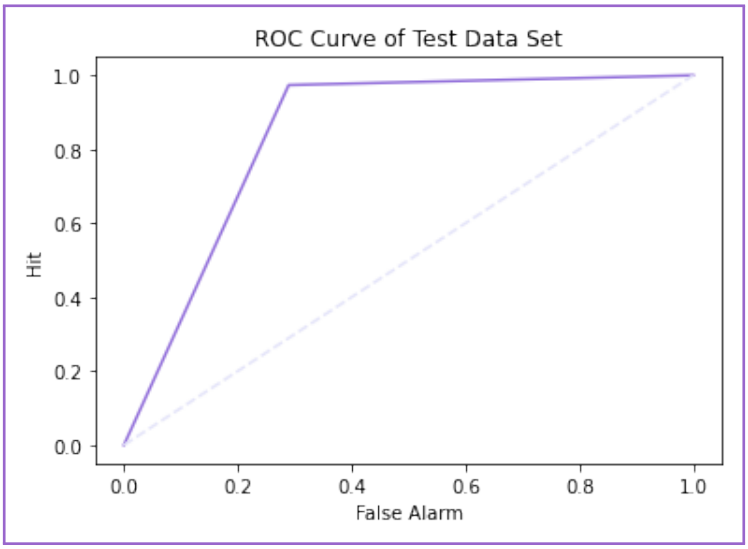
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	90.79	0.79	0.94	0.84

شکل ۱۴.۴: مترهای مختلف عملکرد طبقه‌بند SVM با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۱۵.۴: ماتریس CONFUSION طبقه‌بند SVM با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۱۶.۴: منحنی ROC طبقه‌بند SVM با پیش‌پردازش AE بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همانطور که مشاهده می‌گردد در مدل‌های ساخته شده توسط SVM دقت مدل حدود $0.9 - 0.88$ بر روی داده‌های تست بوده است. همانطور که از ماتریس‌های Confusion پیداست، مشکل عمده‌ی طبقه‌بند بر روی دسته‌بندی داده‌های مربوط به کلاس با داده اولیه کمتر بوده، که این موضوع کاملا قابل انتظار بوده است. در واقع به دلیل بالانس نبودن داده، کمبود حجم کلی آن و از طرفی بعد بالای مسئله انتظار می‌رفت که طبقه‌بند در برابر تشخیص کلاس مذکور دارای چالش بیشتری باشد. مشاهده می‌شود که روش پیش‌پردازش Autoencoder توانسته است با کاهش بعد مسئله به ۵۰، نسبت به روش PCA که فضا را به بعد ۲۵۴ تبدیل می‌نماید موفق‌تر عمل نماید. در واقع با کاهش بعد می‌توان به بهبود مشکلات ذکر شده کمک نمود(از جمله مسئله تشخیص کلاس با دیتای کمتر و جلوگیری از Overfitting به تعداد کم دیتا).

۲.۴ پیاده سازی Logistic Regression

۱.۲.۴ پیش‌پردازش در LR

همان طور که در بخش ۳.۱.۱ بیان شد، پیش‌پردازش بخش مهمی از مراحل جداسازی داده‌ها است. مطابق دلایل بیان شده در بخش مذکور، داده‌ها بعد از قسمت شدن، بالانس شده و با استفاده از روش یکه‌سازی استاندارد، آماده مرحله کاهش‌بعد می‌شوند. مطابق بخش قبل، از Autoencoder با همان ویژگی‌های بیان شده استفاده شده است. استفاده از سایر روش‌های کاهش‌بعد به‌جز طولانی بودن زمان اجرا، منجر به نتیجه‌های مطلوبی نیز نمی‌شدند.

۲.۲.۴ طراحی طبقه‌بند LR

همان طور که در بخش ۲.۲.۲ اشاره شد، در این مسائل باید وزن‌ها و بایاس با توجه به نوع مسئله و همچنین تابع هزینه تعریف شوند. در مرحله بعد باید روش مناسب برای حل مسئله بهینه‌سازی تعیین شود. همچنین پارامتر C که به عنوان پیچ تنظیم در مسئله بهینه‌سازی حضور دارد هم باید تعیین گردد.

۳.۲.۴ پیاده‌سازی و محاسبه مدل بهینه

مانند بخش ۳.۱.۳ با استفاده از روش Gridsearch سعی در پیدا کردن مدل بهینه داریم. ابتدا ماژول grid را با پارامترهای مطرح شده در شکل ۱۷.۴ در نظر می‌گیریم.

Parameters to Adjust	C	penalty	solver
Hyperparameters	[1.e-03 1.e-02 1.e-01 1.e+00 1.e+01 1.e+02 1.e+03]	['l1', 'l2']	['saga', 'liblinear']

شکل ۱۷.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۱۸.۴ می‌باشند:

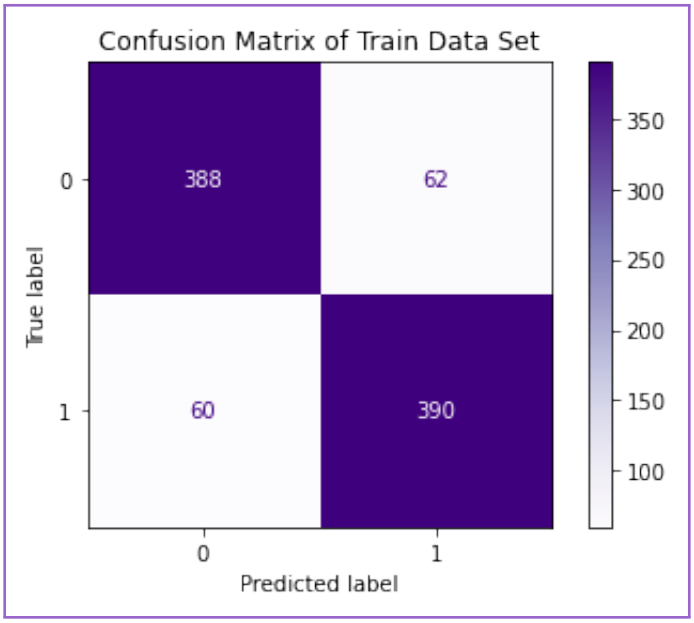
Parameters to Adjust	C	penalty	solver
Hyperparameters	1	l1	saga

شکل ۱۸.۴: پارامترهای بهینه مدل

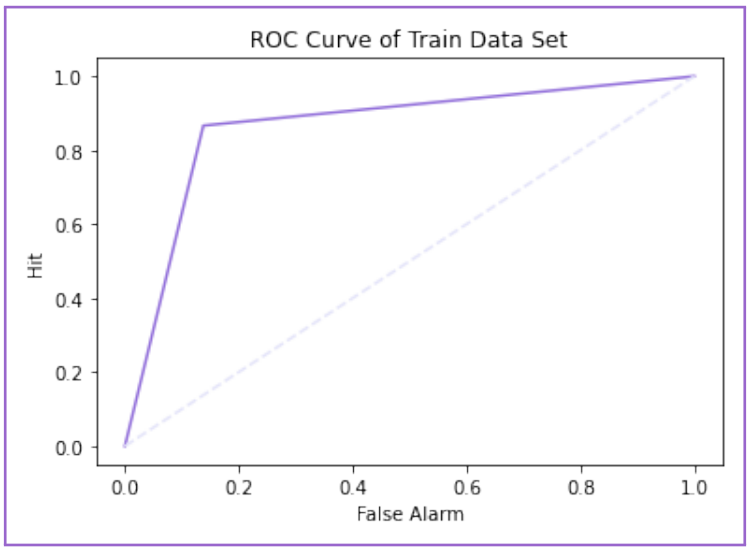
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	83.67±5.07	0.86	0.86	0.86

شکل ۱۹.۴: مترهای مختلف عملکرد طبقه‌بند LR با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۲۰.۴: ماتریس CONFUSION طبقه‌بند LR با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش

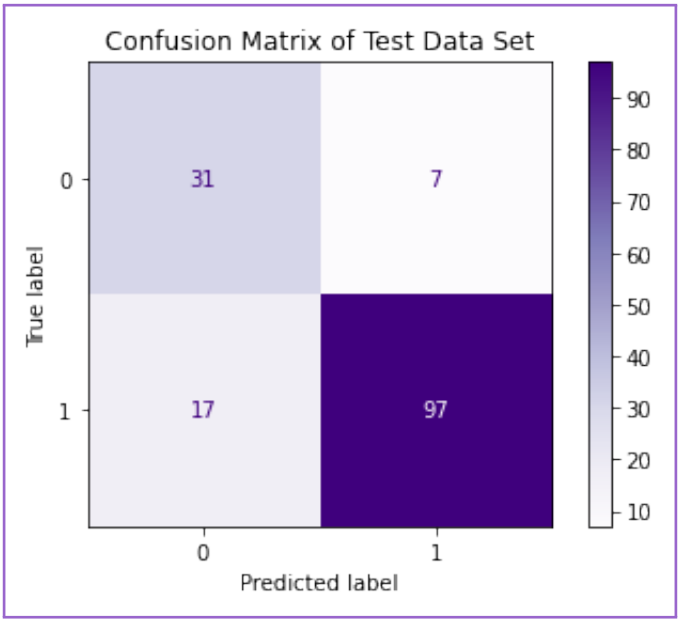


شکل ۲۱.۴: منحنی ROC طبقه‌بند LR با پیش‌پردازش ICA

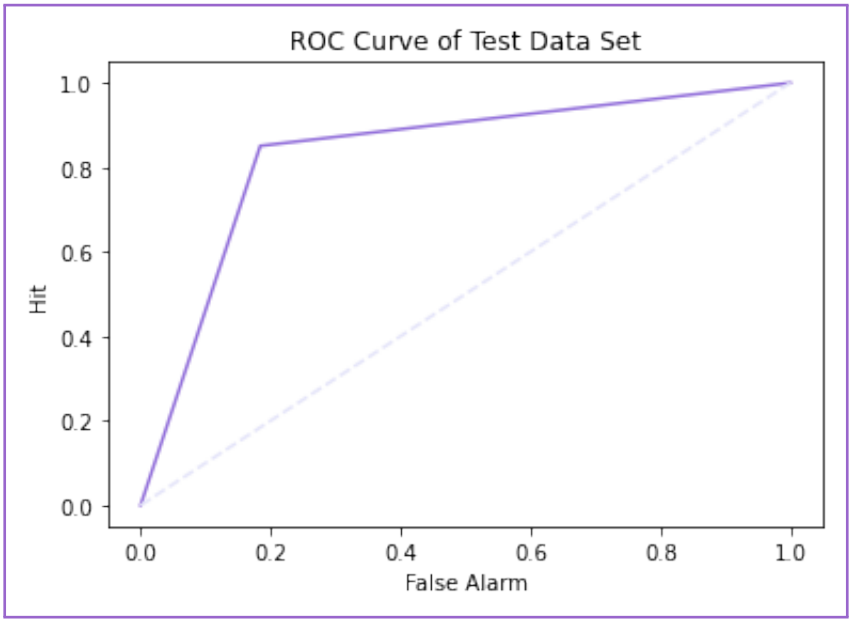
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	84.21	0.72	0.89	0.83

شکل ۲۲.۴: مترهای مختلف عملکرد طبقه‌بند LR با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۲۳.۴: ماتریس CONFUSION طبقه‌بند LR با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۲۴.۴: منحنی ROC طبقه‌بند LR با پیش‌پردازش AUTOENCODER بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که دیده می‌شود، طبقه‌بند حتی داده‌های آموزش را کاملا به درستی نتوانسته جدا کند. تقریبا می‌توان گفت توانایی طبقه‌بند برای جداسازی کلاس صفر و یک روی داده‌های آموزش یکسان است و به همین علت معیارهای متفاوت اندازه‌گیری،رفتارهای نشان داده شده را نمایش می‌دهند. اما اگر به رفتار طبقه‌بند به روی داده تست نگاه کنیم، مشاهده می‌کنیم که توانایی طبقه‌بند در جداسازی کلاس یک بهتر است. این موضوع را می‌توان به وضوح به تعداد داده‌های موجود در هر کلاس ربط داد.

۳.۴ پیاده سازی K-Nearest Neighbors

۱.۳.۴ پیش‌پردازش در KNN

همان طور که در بخش ۳.۱.۱ برای طبقه‌بند SVM و همچنین در بخش ۲.۲.۲ برای طبقه‌بند KNN اشاره شد، لازم است پیش‌پردازش‌هایی روی داده‌ها قبل از استفاده از آن‌ها در طبقه‌بند انجام شود. به همین علت، داده‌ها را نرمالیزه میکنیم. در رابطه با دیتاست داده شده و این نوع طبقه‌بند، مشاهده می‌شود که نرمالیزه کردن به روش Min Max Scaler به نتایج نهایی بهتری می‌انجامد. مشابه بخش ۳.۱.۱، پس از این مرحله به بالانس کردن داده‌ها پرداخته و داده را آماده مرحله کاهش‌بعد می‌کنیم. باتوجه به این که نتایج با استفاده از روش Min Max Scaler بهتر از Standard Scaler است، می‌توان متوجه شد روش ICA که برای داده‌های غیرگوسی روشی بسیار مناسب است، نتایج بهتری خواهد داشت. به‌علاوه، برای این نوع طبقه‌بند، با توجه به این که از بین بردن هرگونه وابستگی می‌تواند باعث بهتر شدن تخمین شود، روش ICA مناسب‌ترین روش می‌باشد.

۲.۳.۴ طراحی طبقه‌بند KNN

همان طور که در بخش ۲.۲.۲ اشاره شد، برای طراحی این طبقه‌بند تنها تنظیم پارامتر K کافی است.

۳.۳.۴ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم. پارامترهای مسئله در ماژول grid به صورت شکل ۲۵.۴ انتخاب می‌شوند.

Parameters to Adjust	n_neighbors	weights	p
Hyperparameters	[1, 4, 6, 8, 10, 12]	['uniform', 'distance']	[1, 2]

شکل ۲۵.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۲۶.۴ می‌باشند:

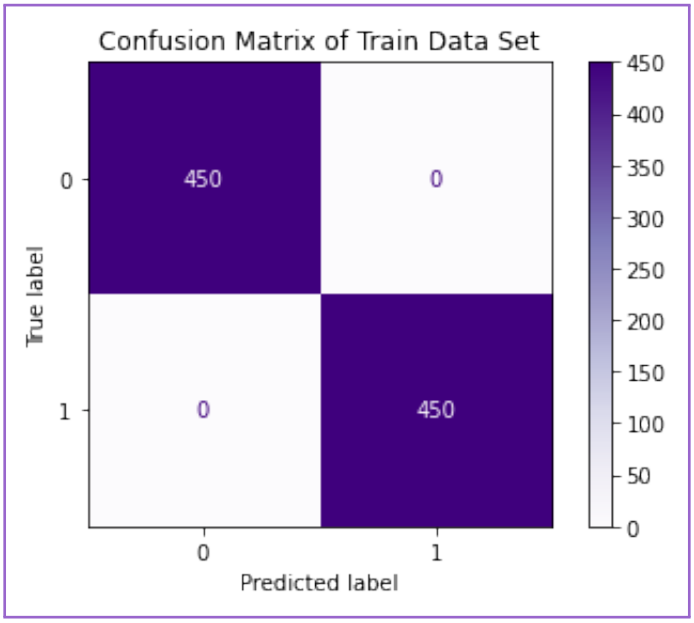
Parameters to Adjust	n_neighbors	p	weights
Hyperparameters	1	2	uniform

شکل ۲۶.۴: پارامترهای بهینه مدل

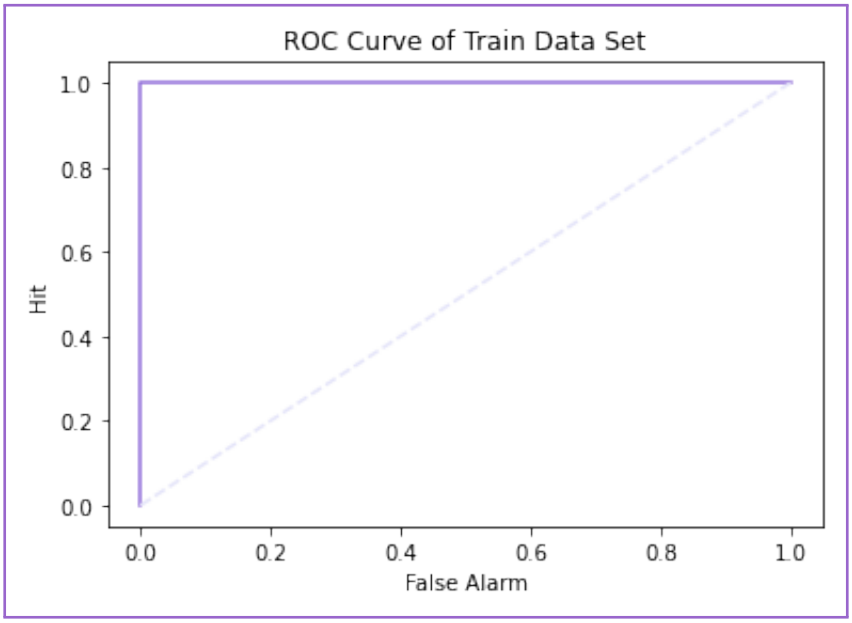
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	95.67±2.79	1	1	1

شکل ۲۷.۴: مترهای مختلف عملکرد طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های آموزش



شکل ۲۸.۴: ماتریس CONFUSION طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های آموزش

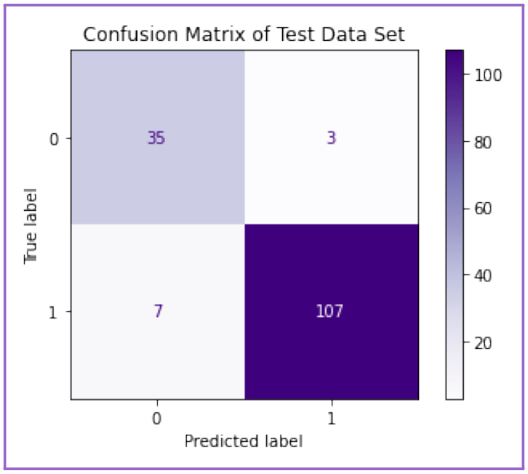


شکل ۲۹.۴: منحنی ROC طبقه‌بند KNN با پیش‌پردازش ICA

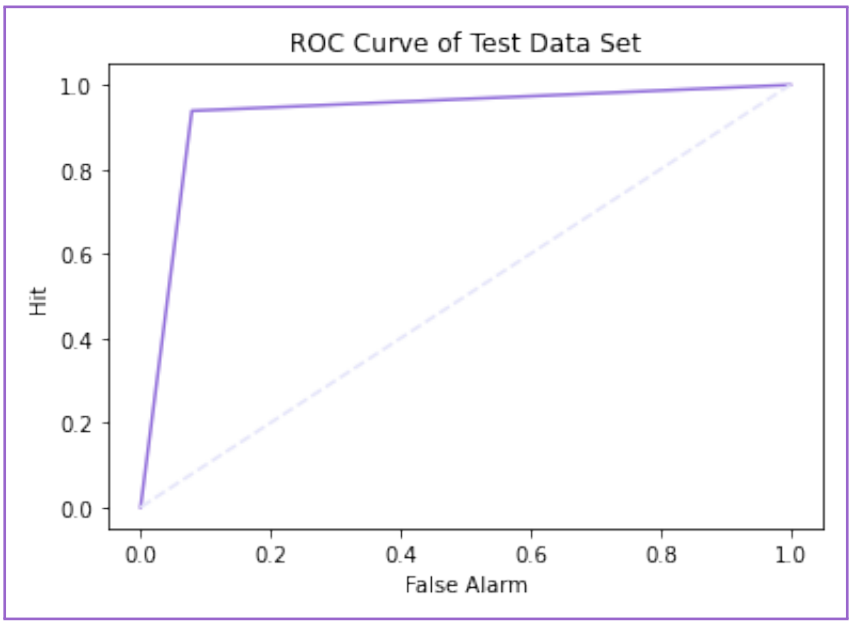
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	93.42	0.88	0.96	0.93

شکل ۳۰.۴: مترهای مختلف عملکرد طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۳۱.۴: ماتریس CONFUSION طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۳۲.۴: منحنی ROC طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که دیده می‌شود، طبقه‌بند به خوبی داده‌های آموزش را جدا کرده است. این موضوع از روی نتایج مترهای مختلف واضح است. اما بر روی داده‌های تست، همان طور که در قسمت‌های قبل مشاهده شد، طبقه‌بند دچار اشتباهاتی در طبقه‌بندی هر دو کلاس شده است. البته به علت کمبود داده‌های کلاس ۰، مشکلات موجود برای این کلاس بیشتر است.

۴.۴ پیاده سازی Decision Tree

۱.۴.۴ پیش‌پردازش در DT

در این قسمت، مشابه قسمت قبل، روش‌های پیش‌پردازش متفاوتی برای رسیدن به بهترین حالت عملکرد بر روی داده‌ها صورت گرفته است. البته باید دقت شود که مشابه قسمت‌های قبل، تقسیم داده و بالانس کردن انجام شده است. اما در رابطه با یکه‌سازی، از روش Min Max Scaler استفاده می‌کنیم. در رابطه با کاهش‌بعد، از بین روش‌های بیان شده قبلی، Autoencoder به عنوان بهترین روش، چه از منظر زمان یادگیری و چه از منظر عملکرد طبقه‌بند انتخاب شده است.

۲.۴.۴ طراحی طبقه‌بند DT

همان طور که در بخش ۲.۲.۲ اشاره شد، پارامترهای مربوط به درخت تصمیم باید با دقت تنظیم شوند تا به این ترتیب از overfit شدن بر داده‌های آموزش جلوگیری شود. به همین علت، پارامترهای criterion، بیشترین عمق درخت و کم‌ترین تعداد نمونه‌ها برای تقسیم باید تنظیم گردند.

۳.۴.۴ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم. پارامترهای مسئله در ماژول grid به صورت شکل ۳۳.۴ انتخاب می‌شوند.

Parameters to Adjust	criterion	max_depth	min_samples_split	max_features
Hyperparameters	['gini', 'entropy']	[4, 8, 12, 15, 16, 20, 24, 28]	[2, 3, 4, 5]	['auto', 'sqrt', 'log2']

شکل ۳۳.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۳۴.۴ می‌باشند:

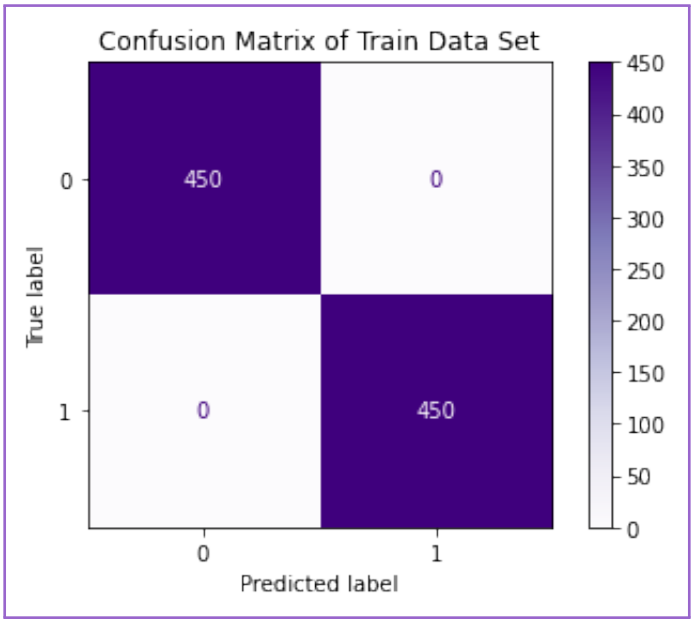
Parameters to Adjust	criterion	max_depth	max_features	min_samples_split
Hyperparameters	entropy	20	auto	2

شکل ۳۴.۴: پارامترهای بهینه مدل

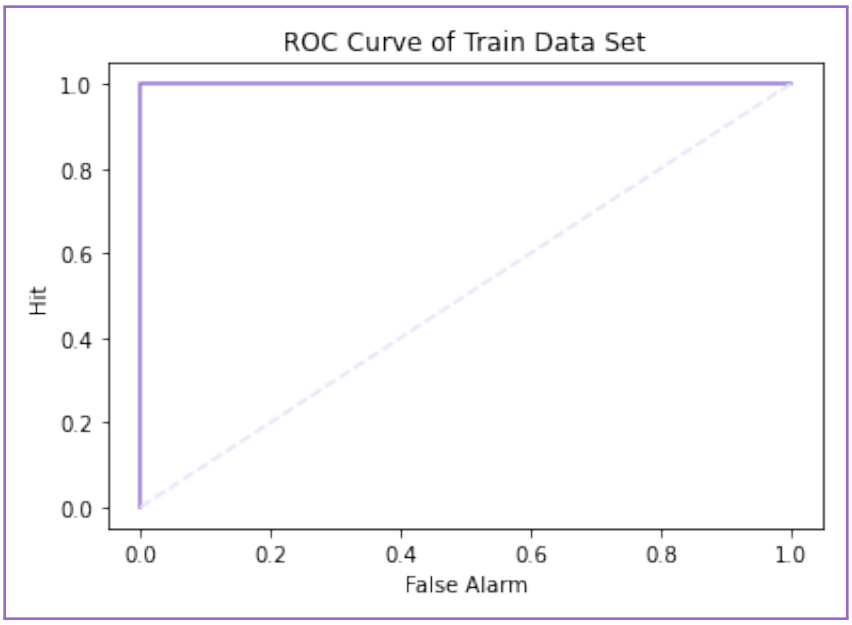
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	84.44±2.90	1	1	1

شکل ۳۵.۴: مترهای مختلف عملکرد طبقه‌بند DT با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۳۶.۴: ماتریس CONFUSION طبقه‌بند DT با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش

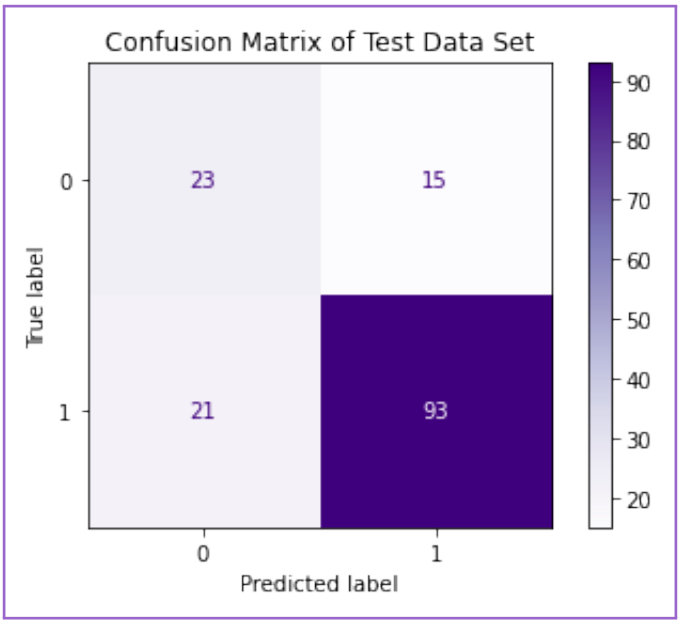


شکل ۳۷.۴: منحنی ROC طبقه‌بند DT با پیش‌پردازش AUTOENCODER

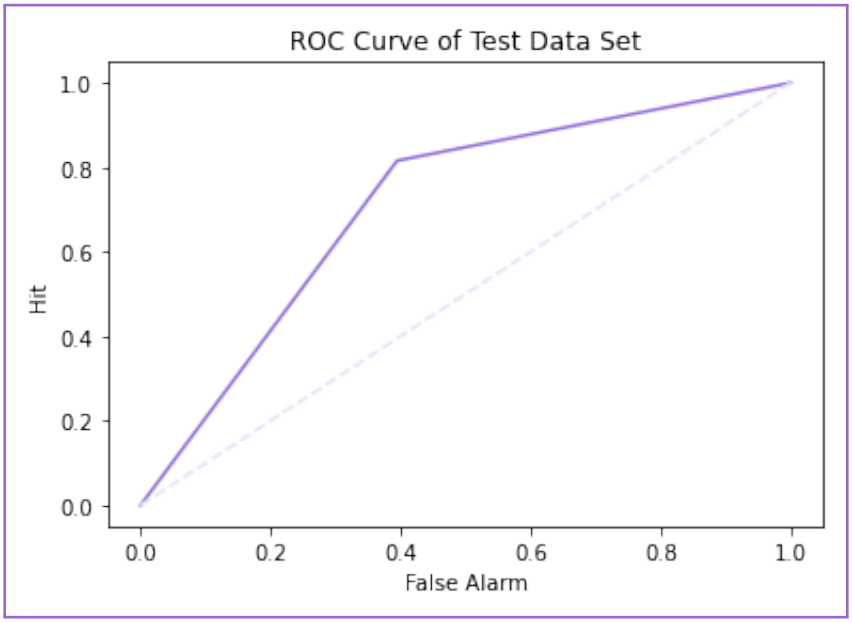
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	76.32	0.56	0.84	0.71

شکل ۳۸.۴: مترهای مختلف عملکرد طبقه‌بند DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۳۹.۴: ماتریس CONFUSION طبقه‌بند DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۴۰.۴: منحنی ROC طبقه‌بند DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که دیده می‌شود، طبقه‌بند تا حد خوبی داده‌های آموزش را جدا کرده است. این موضوع از روی نتایج مترهای مختلف واضح است. اما بر روی داده‌های تست، همان طور که در قسمت‌های قبل مشاهده شد، طبقه‌بند دچار اشتباهاتی در طبقه‌بندی هر دو کلاس شده است. البته به علت کمبود داده‌های کلاس صفر، مشکلات موجود برای این کلاس بیشتر است که تاثیر آن را مخصوصا در f1 score مشاهده می‌کنیم.

۵.۴ پیاده سازی Multi-Layer Perceptron

۱.۵.۴ پیش‌پردازش در MLP

همان طور که در بخش ۳.۱.۱ برای طبقه‌بند SVM و همچنین در بخش ۲.۲.۲ برای MLP اشاره شد، لازم است پیش‌پردازش‌هایی روی داده‌ها قبل از استفاده از آن‌ها در طبقه‌بند انجام شود. به همین علت، داده‌ها را نرمالیزه میکنیم. در رابطه با دیتاست داده شده و این نوع طبقه‌بند، مشاهده می‌شود که نرمالیزه کردن به روش Min Max Scaler به نتایج نهایی بهتری می‌انجامد. مشابه بخش ۳.۱.۱، پس از این مرحله به بالانس کردن داده‌ها پرداخته و داده را آماده مرحله کاهش‌بعد می‌کنیم. باتوجه به این که نتایج با استفاده از روش Min Max Scaler بهتر از Standard Scaler است، می‌توان متوجه شد روش ICA که برای داده‌های غیرگوسی روشی بسیار مناسب است، نتایج بهتری خواهد داشت. به‌علاوه، برای این نوع طبقه‌بند، با توجه به این که از بین بردن هرگونه وابستگی می‌تواند باعث بهتر شدن تخمین شود، روش ICA مناسب‌ترین روش می‌باشد.

۲.۵.۴ طراحی طبقه‌بند MLP

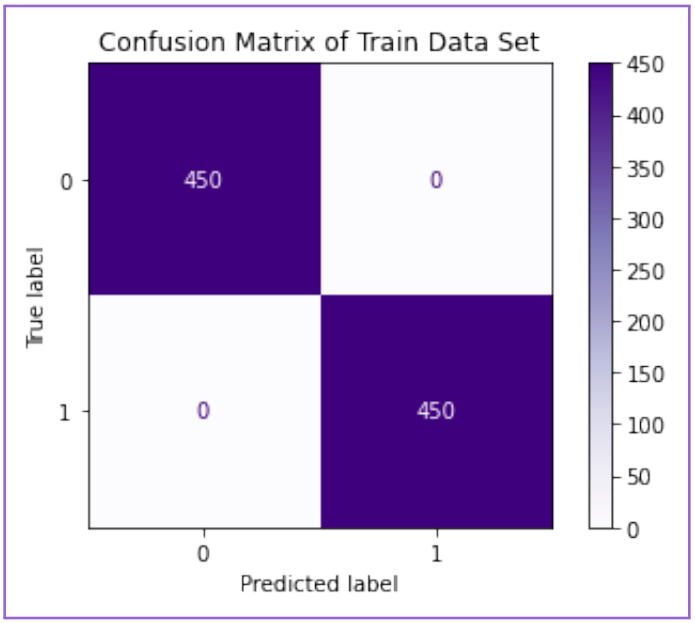
همان طور که در بخش ۲.۲.۲ اشاره شد، برای طراحی این طبقه‌بند پارامترهای زیادی را باید مورد بررسی قرار داده و به مقدار مناسبی تنظیم کرد.

۳.۵.۴ پیاده‌سازی و محاسبه مدل بهینه

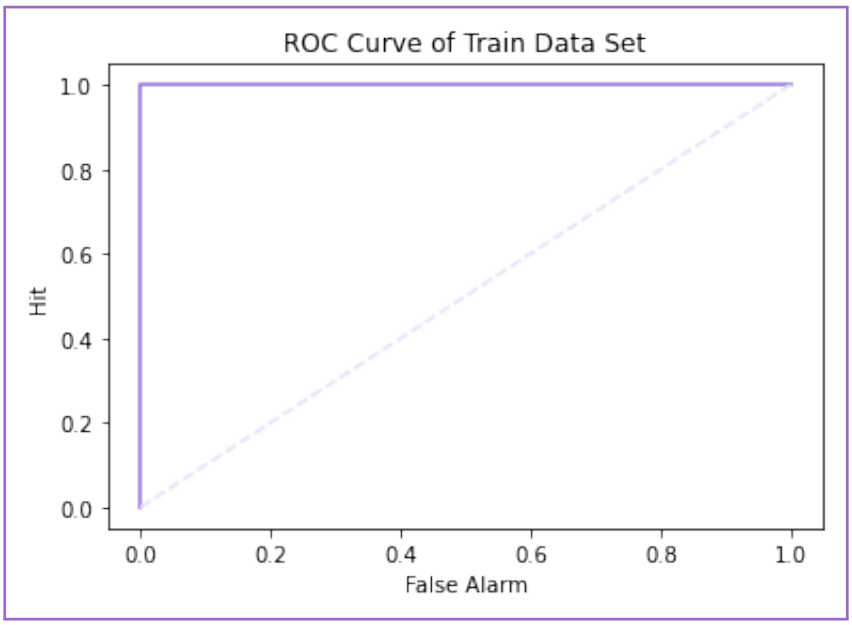
با استفاده از سعی و خطا پارامترهای بهینه مناسب را انتخاب می‌کنیم. به این ترتیب یک شبکه با دو لایه میانی داریم که تعداد نودهای آن را بر اساس ابهاد مسئله تنظیم کرده‌ایم. با توجه به این که به دنبال robust ترین روش برای حل مسئله هستیم، سایر پارامترها نیز ست می‌شوند. حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	95.67±2.92	1	1	1

شکل ۴۱.۴: مترهای مختلف عملکرد طبقه‌بند MLP با پیش‌پردازش ICA بر روی داده‌های آموزش



شکل ۴۲.۴: ماتریس CONFUSION طبقه‌بند MLP با پیش‌پردازش ICA بر روی داده‌های آموزش

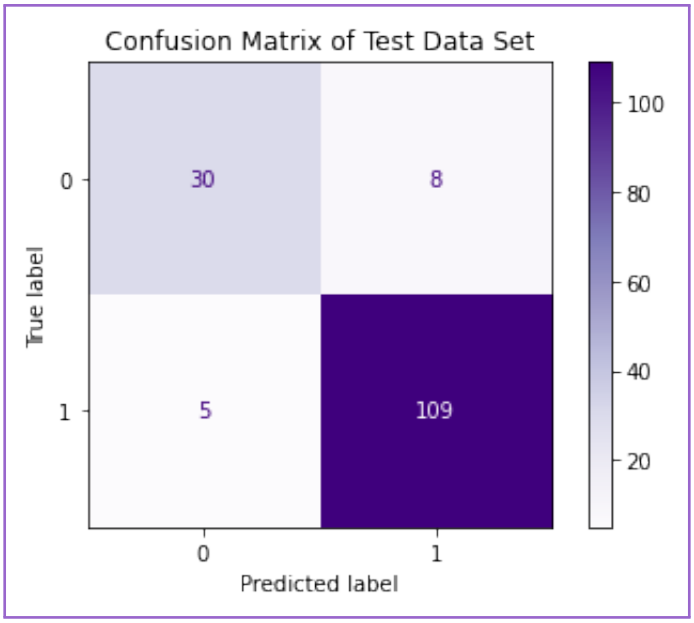


شکل ۴۳.۴: منحنی ROC طبقه‌بند MLP با پیش‌پردازش ICA

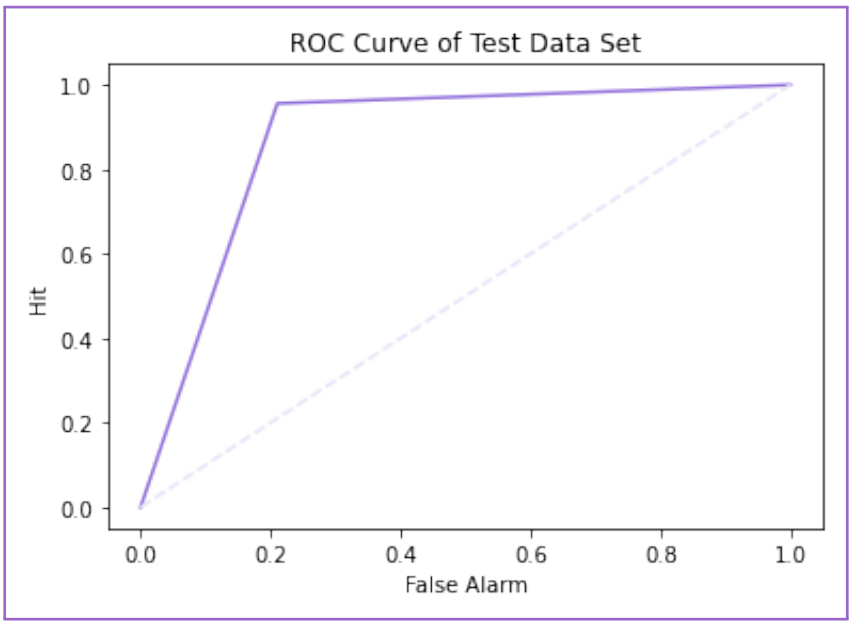
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	91.45	0.82	0.94	0.87

شکل ۴۴.۴: مترهای مختلف عملکرد طبقه‌بند MLP با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۴۵.۴: ماتریس CONFUSION طبقه‌بند MLP با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۴۶.۴: منحنی ROC طبقه‌بند MLP با پیش‌پردازش ICA بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که دیده می‌شود، طبقه‌بند به خوبی داده‌های آموزش را جدا کرده است. این موضوع از روی نتایج مترهای مختلف واضح است. به طور مثال، دیده می‌شود که مانند قسمت‌های قبل، سطح زیر نمودار ROC که آن را با AUC نمایش می‌دهیم،برابر یک شده است. این خود به معنی این است که طبقه‌بند بدون خطا داده‌های مربوط به هر کلاس را از کلاس دیگر تمیز داده‌است. اما بر روی داده‌های تست، همان طور که در قسمت‌های قبل مشاهده شد، طبقه‌بند دچار اشتباهاتی در طبقه‌بندی هر دو کلاس شده است. البته به علت کمبود داده‌های کلاس صفر، خطاهای موجود برای این کلاس بیشتر است. به عبارت دیگر، شبکه عصبی مورد استفاده برای طبقه‌بندی، کلاس یک را بهتر به خاطر آورده است. این موضوع را از روی معیار f1-score می‌توان به خوبی مشاهده کرد.

۶.۴ پیاده سازی مدل RBF

۱.۶.۴ پیش‌پردازش در RBF

در پیش پردازش این مدل کاملا مانند مدل SVM عمل می‌گردد، تنها در Autoencoder مورد استفاده بجای ۵۰ نود، از ۸۰ نود میانی استفاده می‌گردد.

۲.۶.۴ طراحی طبقه‌بند RBF

همانطور که در بخش ۲.۲ توضیح داده شد، در طبقه‌بند RBF مانند طبقه‌بندهای MLP عمل می‌گردد. در این بخش پارامترهای آموزش و بهینه سازی را به صورت SGD, categorical crossentropy در نظر میگیریم.

پیش‌پردازش PCA

پارامترهای مسئله در ماژول grid به صورت شکل ۴۷.۴ انتخاب می‌شوند.

```
grid = {'nodes': [10,25,50], 'gamma':[0.01,0.1,0.4]}
```

شکل ۴۷.۴: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۴۸.۴ می‌باشند:

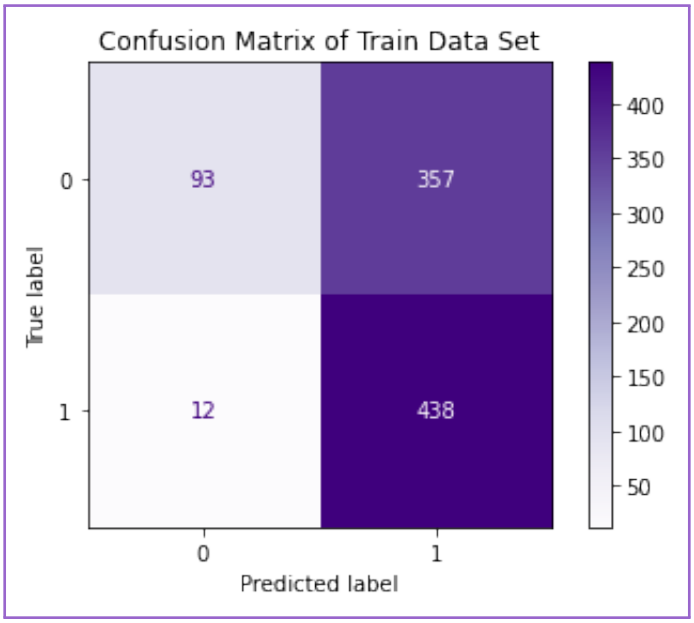
Parameters to Adjust	gamma	nodes
Hyperparameters	0.1	25

شکل ۴۸.۴: پارامترهای بهینه مدل

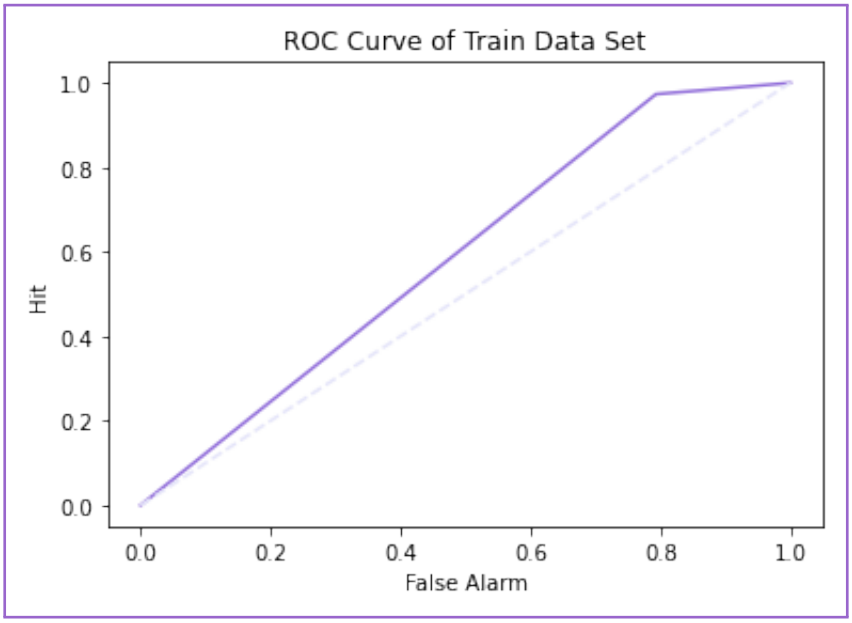
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	73.11±4.38	0.7	0.79	0.75

شکل ۴۹.۴: مترهای مختلف عملکرد طبقه‌بند RBF با پیش‌پردازش PCA بر روی داده‌های آموزش



شکل ۵۰.۴: مترهای مختلف عملکرد طبقه‌بند RBF با پیش‌پردازش PCA بر روی داده‌های آموزش

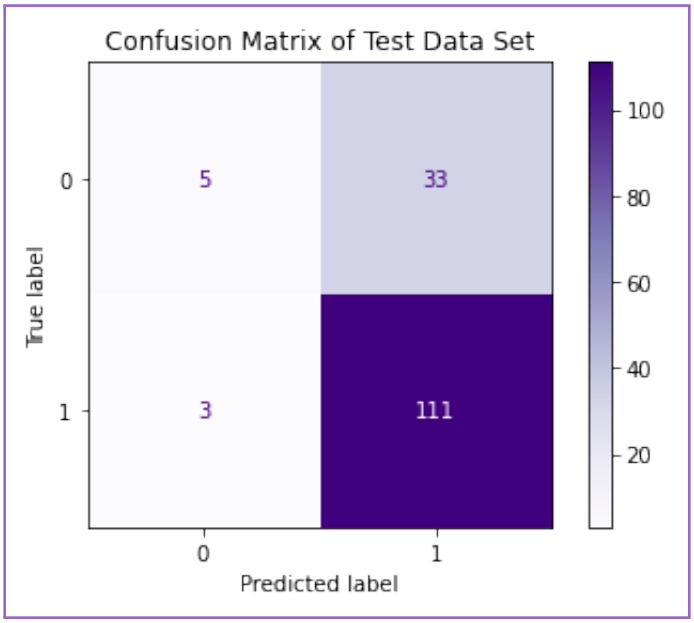


شکل ۵۱.۴: منحنی ROC طبقه‌بند RBF با پیش‌پردازش PCA

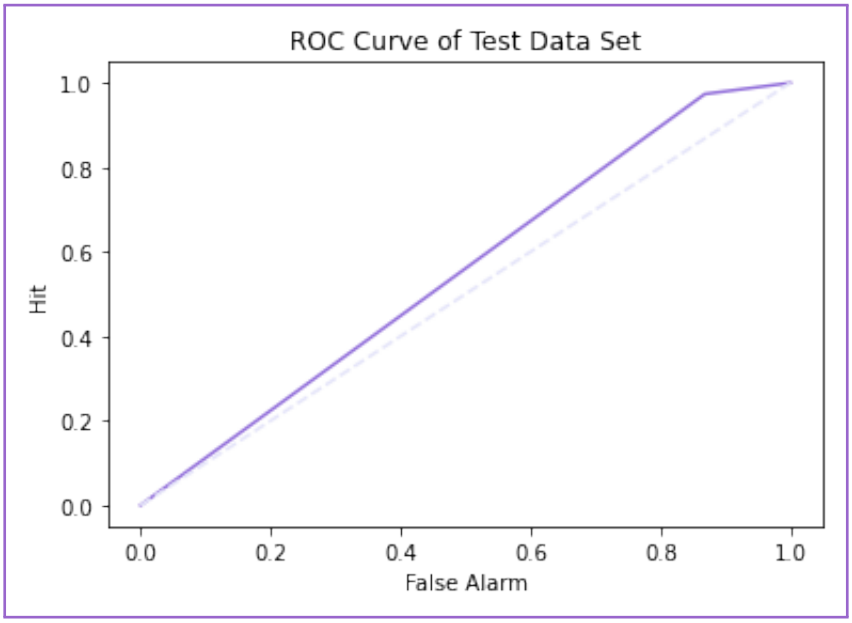
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	81.58	0.6	0.88	0.73

شکل ۵۲.۴: مترهای مختلف عملکرد طبقه‌بند RBF با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۵۳.۴: ماتریس CONFUSION طبقه‌بند RBF با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۵۴.۴: منحنی ROC طبقه‌بند RBF با پیش‌پردازش PCA بر روی داده‌های تست

فصل ۵

پیاده سازی مدل‌های Generative

در این فصل سعی می‌گردد به بررسی طبقه‌بندهای Generative، با رویکردی مشابه فصل گذشته بپردازیم. در نهایت برای هر طبقه‌بند پیش‌پردازش‌های در نظر گرفته شده با رویکرد تحلیلی بررسی خواهند شد و عملکرد هر کدام از طبقه‌بندها مورد تحلیل قرار خواهد گرفت.

۱.۵ پیاده سازی مدل GMM

۱.۱.۵ پیش‌پردازش در GMM

در پیش پردازش این مدل کاملا مانند مدل SVM عمل می‌گردد، تنها در Autoencoder مورد استفاده بجای ۵۰ نود، از ۸۰ نود میانی استفاده می‌گردد.

۲.۱.۵ طراحی طبقه‌بند GMM

همانطور که در بخش ۲.۲ توضیح داده شد، در طبقه‌بند GMM پارامترهای ساختاری فرم‌های مختلف ماتریس کواریانس هستند. بنابراین مانند بخش‌های قبل با استفاده از بررسی مدل‌های مختلف و بررسی عملکرد آن‌ها بر روی دیتا به تعیین ساختار بهینه می‌پردازیم. همچنین باتوجه به آنکه مجموعه داده‌های مورد استفاده دارای لیبل می‌باشند، می‌توان میانگین هر کلاس را مرکز دسته اولیه در GMM در نظر گرفت.

۳.۱.۵ پیاده‌سازی و محاسبه‌ی مدل بهینه

حال با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم.

۴.۱.۵ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم.

پیش‌پردازش PCA

پارامترهای مسئله در ماژول grid به صورت شکل ۱.۵ انتخاب می‌شوند.

```
grid = {'covtype': ['spherical', 'diag', 'tied', 'full']}
```

شکل ۱.۵: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۲.۵ می‌باشند:

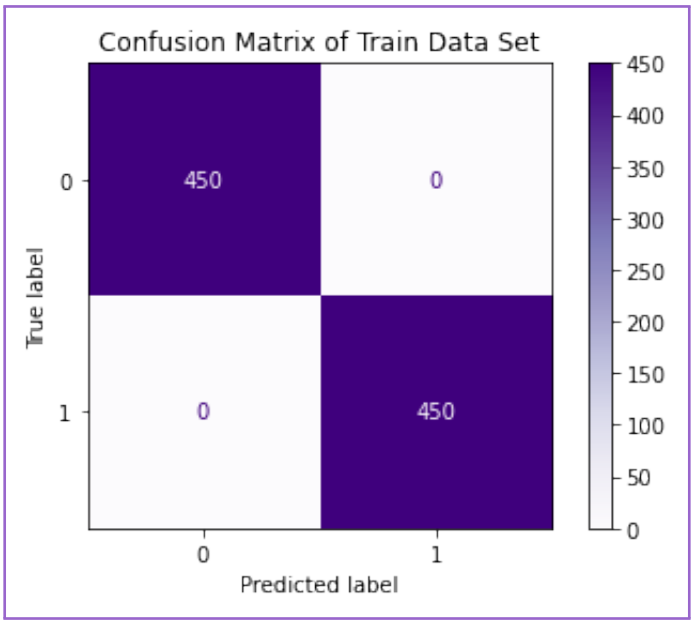
```
Tuned hpyerparameters are: {'covtype': 'tied'}
```

شکل ۲.۵: پارامترهای بهینه مدل

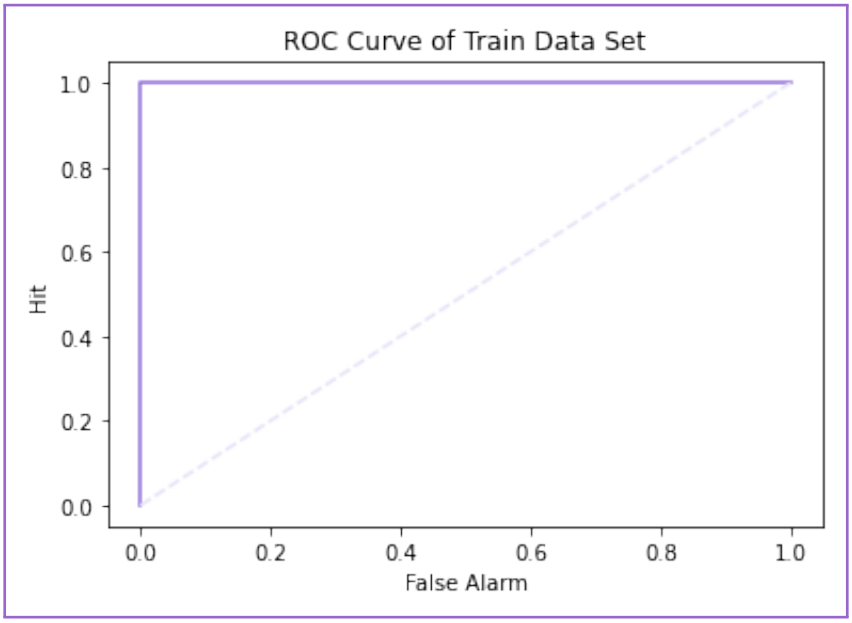
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	91.00±4.23	1	1	1

شکل ۳.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش PCA بر روی داده‌های آموزش



شکل ۴.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش PCA بر روی داده‌های آموزش

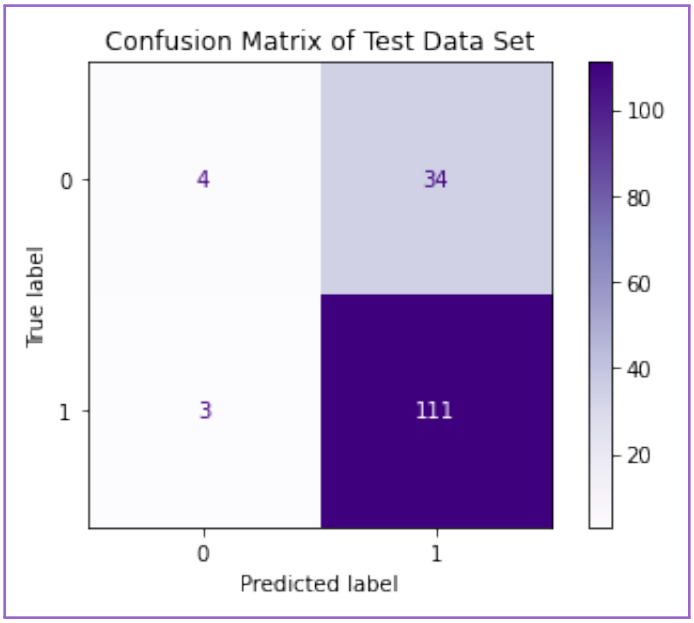


شکل ۵.۵: منحنی ROC طبقه‌بند GMM با پیش‌پردازش PCA

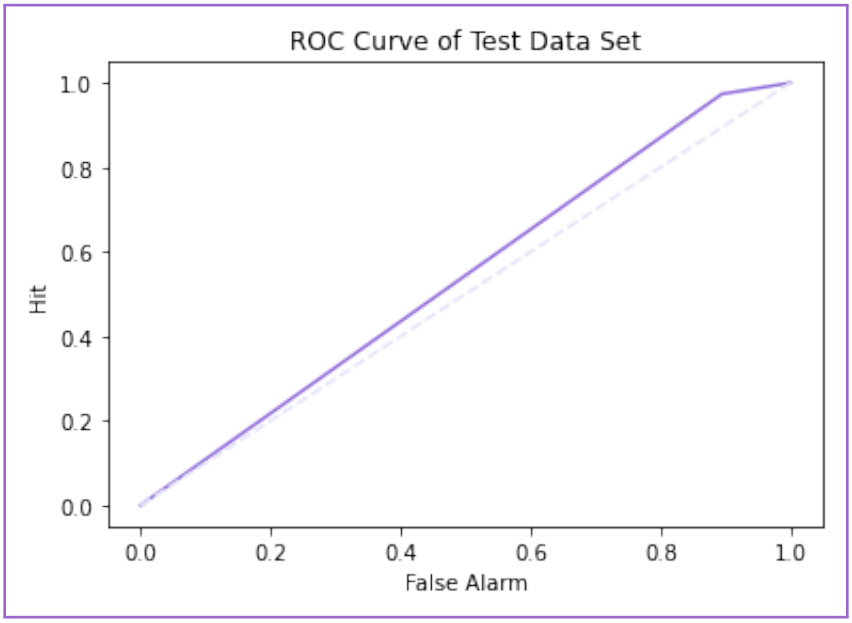
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	75.66	0.18	0.86	0.54

شکل ۶.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۷.۵: ماتریس CONFUSION طبقه‌بند GMM با پیش‌پردازش PCA بر روی داده‌های تست



شکل ۸.۵: منحنی ROC طبقه‌بند GMM با پیش‌پردازش PCA بر روی داده‌های تست

پیش‌پردازش **Autoencoders**

حال مانند بخش قبل عمل می‌نماییم اما بجای پیش‌پردازش GMM از پیش‌پردازش Autoencoder با ۸۰ نود میانی استفاده می‌نماییم. بنابراین مشاهده می‌گردد که بعد مسئله به ۸۰ کاهش یافته است. مانند بخش قبل با استفاده از روش Gridsearch سعی در پیدا کردن مدل بهینه داریم. ابتدا ماژول grid را با پارامترهای مطرح شده در شکل ۹.۵ در نظر می‌گیریم.

```
grid = {'covtype': ['spherical', 'diag', 'tied', 'full']}
```

شکل ۹.۵: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۱۰.۵ می‌باشند:

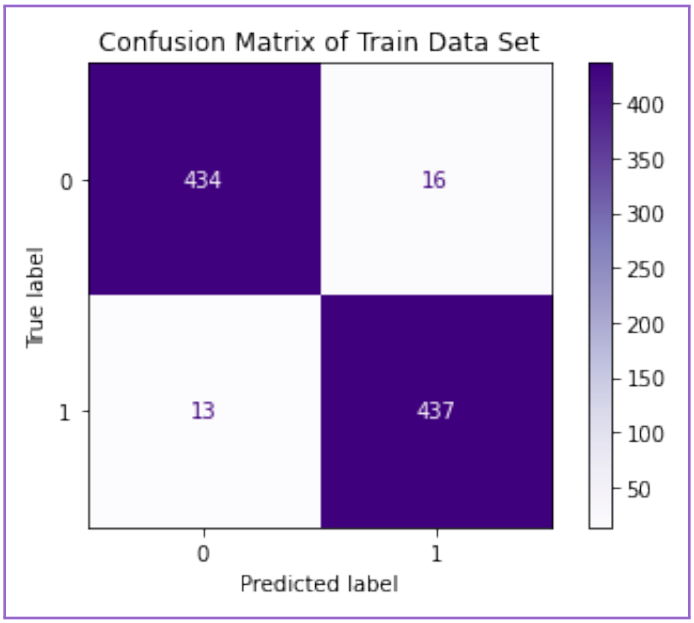
```
Tuned hpyerparameters are: {'covtype': 'tied'}
```

شکل ۱۰.۵: پارامترهای بهینه مدل

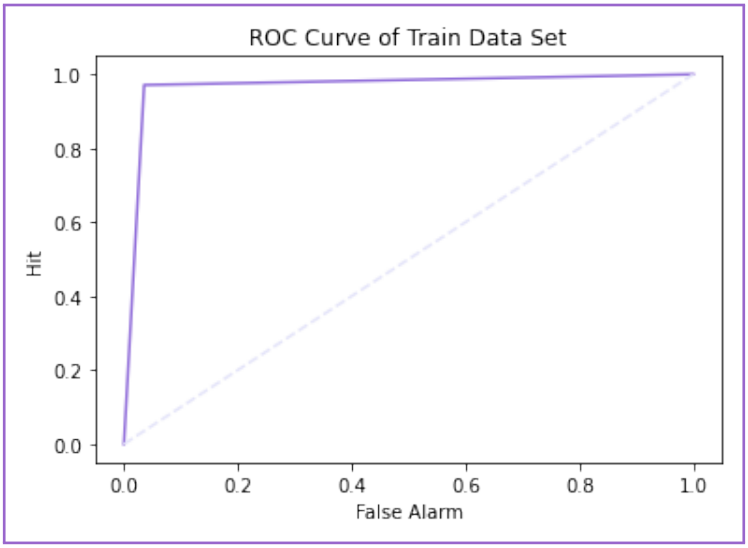
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	91.22±5.20	0.97	0.97	0.97

شکل ۱۱.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۱۲.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش

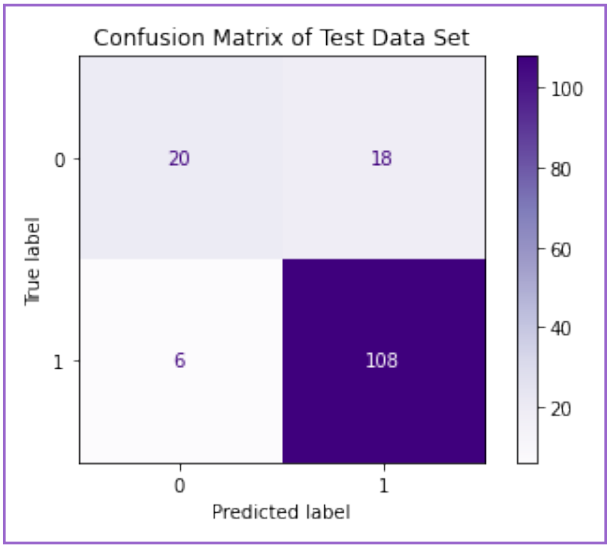


شکل ۱۳.۵: منحنی ROC طبقه‌بند GMM با پیش‌پردازش AUTOENCODER

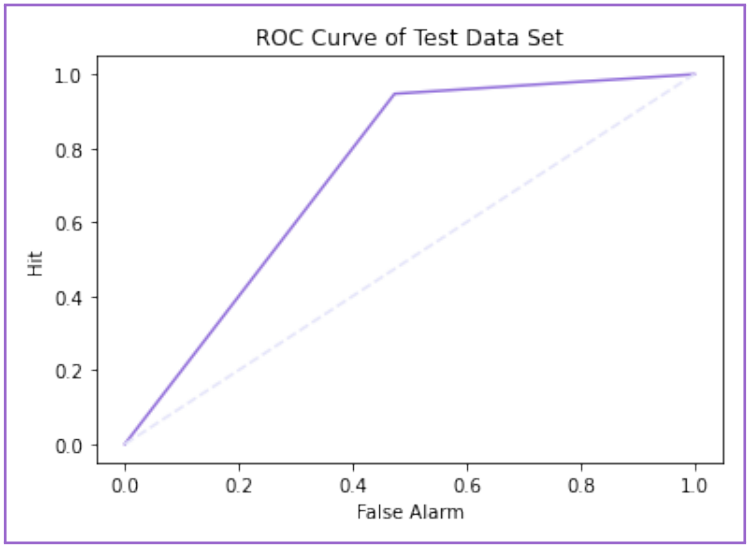
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	84.21	0.62	0.9	0.74

شکل ۱۴.۵: مترهای مختلف عملکرد طبقه‌بند GMM با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۱۵.۵: ماتریس CONFUSION طبقه‌بند GMM با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۱۶.۵: منحنی ROC طبقه‌بند GMM با پیش‌پردازش AE بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همانطور که مشاهده می‌گردد در مدل‌های ساخته شده توسط GMM دقت مدل حدود $0.75 - 0.84$ بر روی داده‌های تست بوده است. همانطور که از ماتریس‌های Confusion پیداست، مشکل عمده‌ی طبقه‌بند بر روی دسته‌بندی داده‌های مربوط به کلاس با داده اولیه کمتر بوده، که این موضوع به دلایلی کاملا مشابه دلایلی که در بخش SVM ذکر شد کاملا قابل انتظار بوده است. از طرفی باتوجه به آنکه در این بخش، به طور مستقیم در حال تخمین توزیع هستیم، توزیع‌های غیربالانس موجب خطای بیشتری نسبت به روش‌های Discriminative می‌شوند. این اثر به وضوح در این بخش و در قیاس بین دو طبقه‌بند GMM با طبقه‌بندی مانند SVM دیده می‌شود. همچنین مشاهده می‌گردد که روش کاهش بعد با استفاده از Autoencoder موجب بهبود عملکرد در قیاس با روش PCA می‌گردد، که این به این دلیل است که Autoencoder توانسته با حذف وابستگی‌های غیرخطی(علاوه بر حذف وابستگی‌های خطی)، موجب بهبود عملکرد بشود.

۲.۵ پیاده‌سازی Parzen Window

۱.۲.۵ پیش‌پردازش در Parzen Window

در این قسمت، مشابه قسمت قبل، روش‌های پیش‌پردازش متفاوتی برای رسیدن به بهترین حالت عملکرد بر روی داده‌ها صورت گرفته است. البته باید دقت شود که تقسیم داده و بالانس کردن آن مشابه قسمت‌های قبلی انجام شده، اما در رابطه با یکه‌سازی، از روش Standard Scaler استفاده می‌کنیم. در رابطه با کاهش‌بعد، از بین روش‌های بیان شده قبلی، Autoencoder به عنوان بهترین روش، چه از منظر زمان یادگیری و چه از منظر عملکرد طبقه‌بند انتخاب شده است. برای این حالت هم، ابعاد را به ۱۰۰ تغییر داده‌ایم.

۲.۲.۵ طراحی طبقه‌بند Parzen Window

همان طور که در بخش ۲.۲.۱ اشاره شد، پارامترهای قابل تنظیم برای این روش نوع کرنل و میزان پهنای‌باند (سایز پنجره‌ها) است. باتوجه به اهمیتی که این دو موضوع دارند، مقدار آن‌ها را باید به‌دقت انتخاب کرد. پس از ایتن مرحله، با استفاده از تصمیم‌گیری بیز، کلاس مربوط به هر دسته پیش‌بینی می‌شوند.

۳.۲.۵ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم. پارامترهای مسئله در ماژول grid به صورت شکل ۱۷.۵ انتخاب می‌شوند.

Parameters to Adjust	bandwidth	kernel
Hyperparameters	[1. 1.12883789 1.27427499 1.43844989 1.62377674 1.83298071 2.06913808 2.33572147 2.6366509 2.97635144 3.35981829 3.79269019 4.2813324 4.83293024 5.45559478 6.15848211 6.95192796 7.8475997 8.8586679 10.]	['gaussian', 'tophat', 'epanechnikov', 'exponential', 'linear', 'cosine']

شکل ۱۷.۵: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۱۸.۵ می‌باشند:

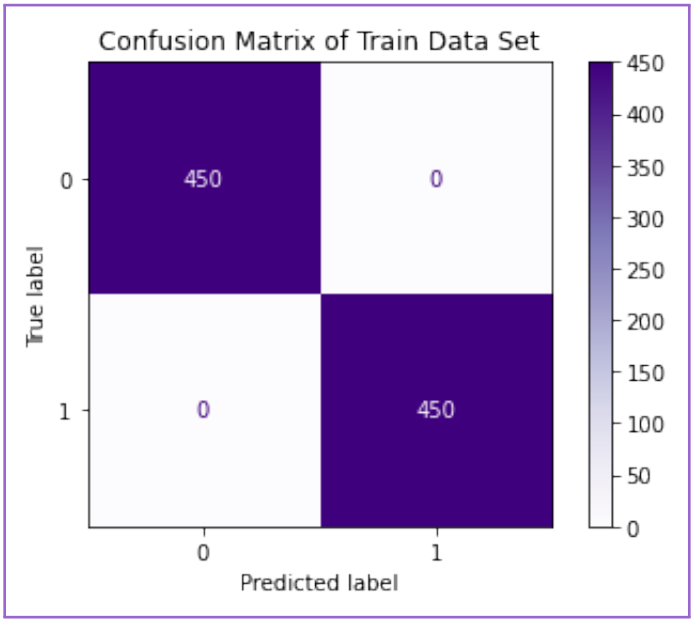
Parameters to Adjust	bandwidth	kernel
Hyperparameters	2.06914	gaussian

شکل ۱۸.۵: پارامترهای بهینه مدل

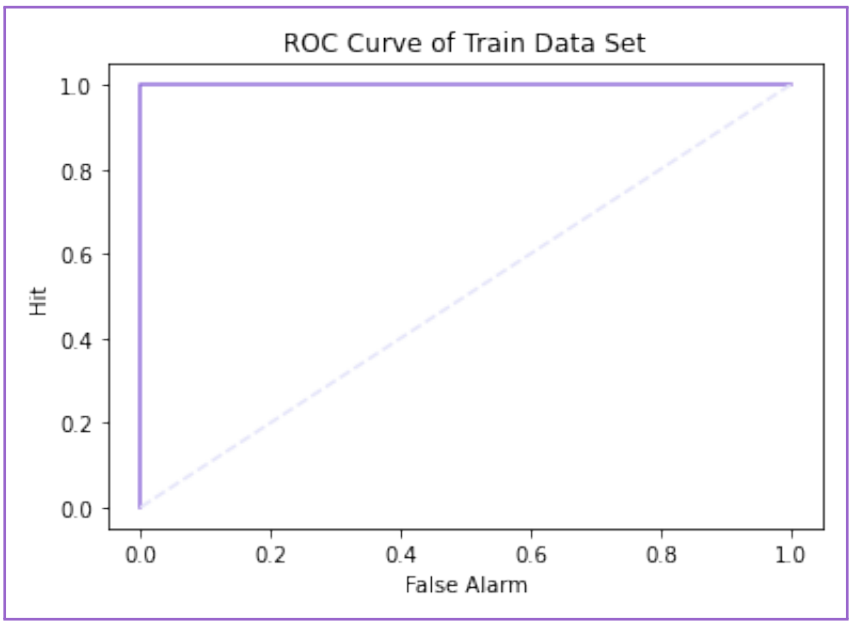
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	95.89±3.55	1	1	1

شکل ۱۹.۵: مترهای مختلف عملکرد طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۲۰.۵: ماتریس CONFUSION طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش

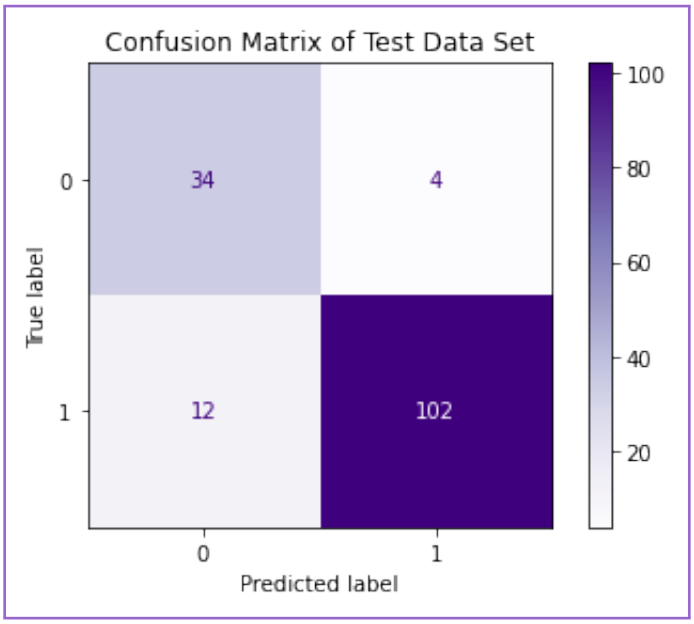


شکل ۲۱.۵: منحنی ROC طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER

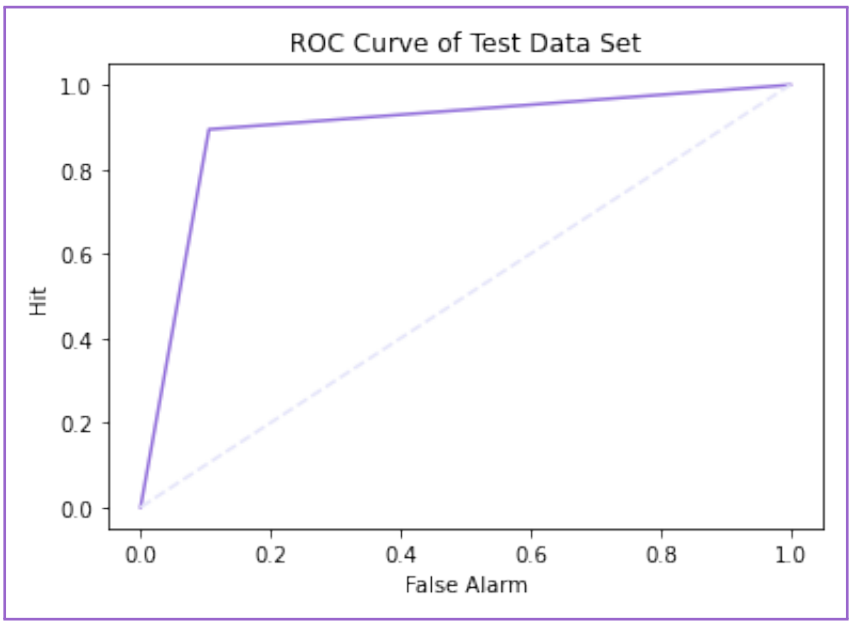
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	89.47	0.81	0.93	0.89

شکل ۲۲.۵: مترهای مختلف عملکرد طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۲۳.۵: ماتریس CONFUSION طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۲۴.۵: منحنی ROC طبقه‌بند PARZEN WINDOW با پیش‌پردازش AUTOENCODER بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که در نتایج بالا دیده می‌شود، استفاده از روش Parzen Window برای تخمین likelihood و استفاده از این مقدار در طبقه‌بند بیز، منجر به دقت خوبی روی داده‌های آموزش می‌گردد. به بیان دیگر، همان طور که از ماتریس آشفتگی می‌توان دید، طبقه‌بند طراحی شده به خوبی داده‌های کلاس صفر و یک را جدا کرده و هیچ خطایی ندارد. از طرف میزان AUC یک نیز نشان‌دهنده توانایی بالای طبقه‌بند در تمیز دادن داده‌های کلاس‌ها از یکدیگر است.

در رابطه با داده‌های تست، مشاهده می‌شود که طبقه‌بند دچار خطاهایی در دسته‌بندی هر دو کلاس شده است. اما همان طور که نتیجه ماتریس آشفتگی نشان می‌دهد، طبقه‌بند کلاس یک را بهتر از کلاس صفر به خاطر دارد. به همین علت هم میزان f1-score آن بیشتر است.این موضوع می‌تواند به علت بیشتر بودن داده‌های کلاس یک و در اصل نابالانس بودن داده‌ها باشد. با این حال می‌توان گفت میزان دقت و AUC قابل قبول هستند.

۳.۵ پیاده سازی K-Nearest Neighbors

۱.۳.۵ پیش‌پردازش در KNN

در این قسمت، مشابه قسمت قبل، روش‌های پیش‌پردازش متفاوتی برای رسیدن به بهترین حالت عملکرد بر روی داده‌ها صورت گرفته است. البته باید دقت شود که تقسیم داده و بالانس کردن آن مشابه قسمت‌های قبلی انجام شده، اما در رابطه با یکه‌سازی، از روش Min Max Scaler استفاده می‌کنیم. در رابطه با کاهش‌بعد، از بین روش‌های بیان شده قبلی، ICA به عنوان بهترین روش، چه از منظر زمان یادگیری و چه از منظر عملکرد طبقه‌بند

انتخاب شده است. در رابطه با این روش تعداد componentها برای رسیدن به بهترین نتیجه برابر ۴۰ قرار داده شده است.

۲.۳.۵ طراحی طبقه‌بند KNN

همان طور که در بخش ۲.۲.۱ اشاره شد، پارامترهای قابل تنظیم برای این روش تعداد همسایه‌های در نظر گفته شده در یک شعاع مشخص برای تصمیم‌گیری به همراه روش اندازه‌گیری فاصله است. از این رو تعیین این مقادیر باید با دقت انجام گردد.

۳.۳.۵ پیاده‌سازی و محاسبه مدل بهینه

در این بخش با استفاده از روش Grid Search به بررسی پارامترهای مختلف مسئله می‌پردازیم. پارامترهای مسئله در ماژول grid به صورت شکل ۲۵.۵ انتخاب می‌شوند.

Parameters to Adjust	n_neighbors	p
Hyperparameters	[1, 4, 6, 8, 10, 12]	[1, 2, 3]

شکل ۲۵.۵: پارامترهای ساختاری

در این حالت بهترین پارامترها به صورت شکل ۲۶.۵ می‌باشند:

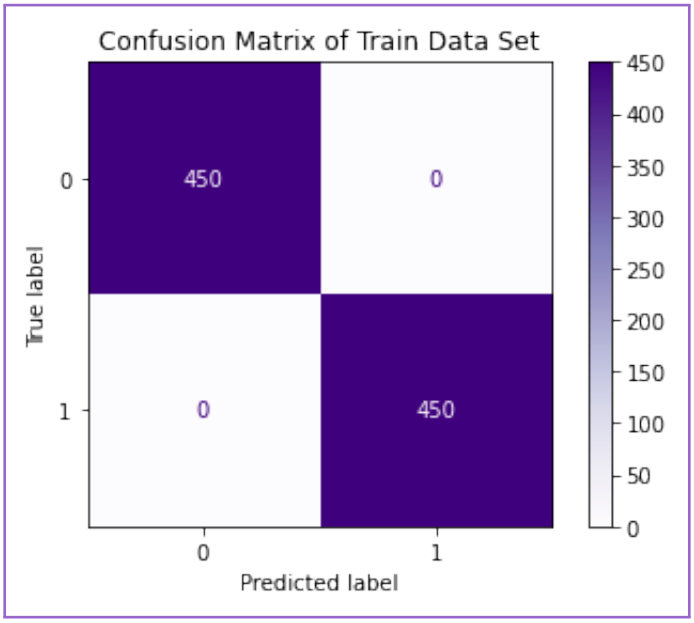
Parameters to Adjust	n_neighbors	p
Hyperparameters	1	2

شکل ۲۶.۵: پارامترهای بهینه مدل

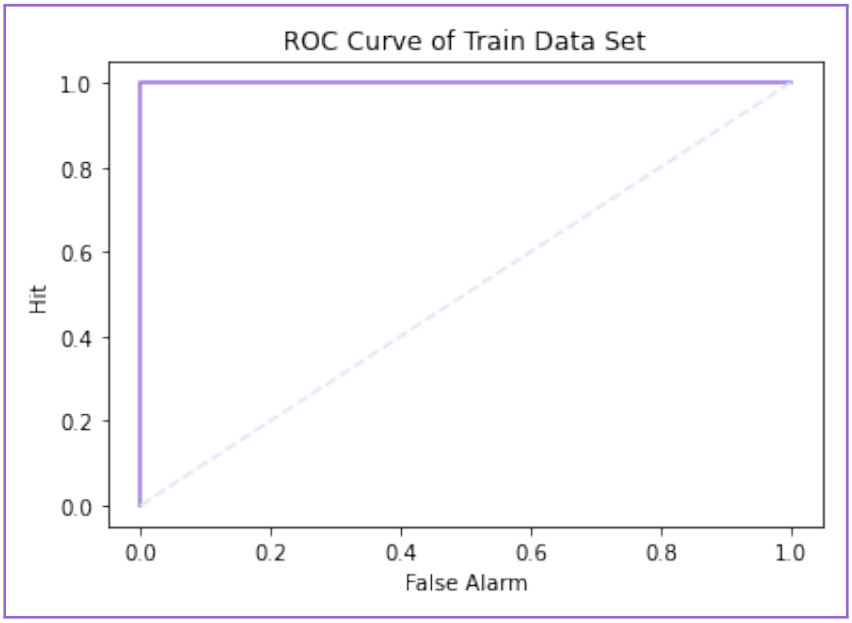
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	95.67±2.79	1	1	1

شکل ۲۷.۵: مترهای مختلف عملکرد طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های آموزش



شکل ۲۸.۵: ماتریس CONFUSION طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های آموزش

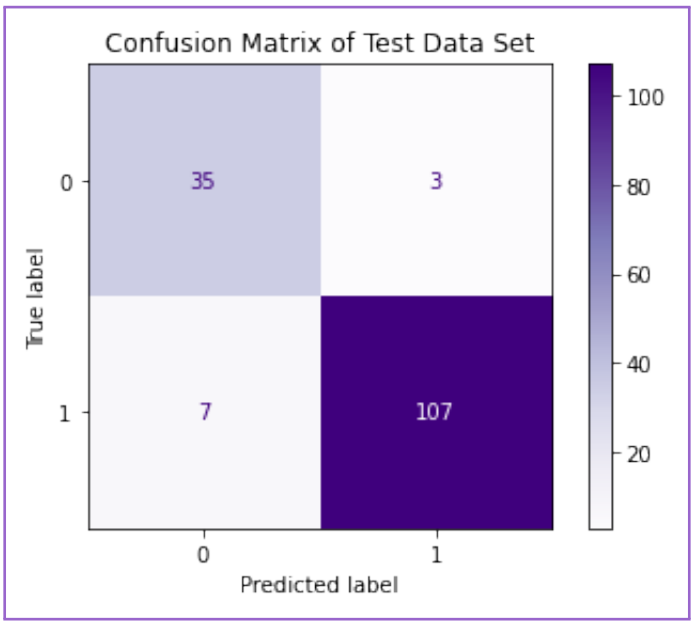


شکل ۲۹.۵: منحنی ROC طبقه‌بند KNN با پیش‌پردازش ICA

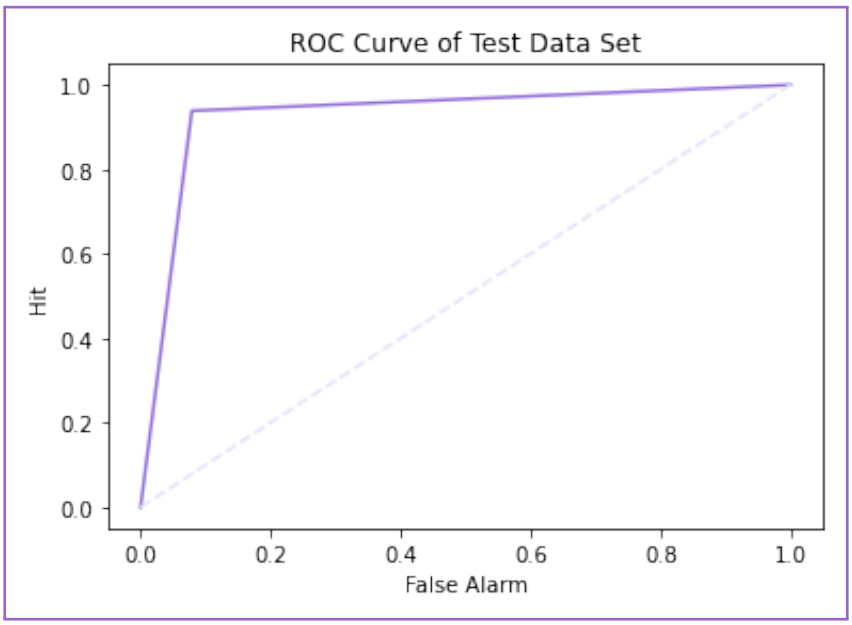
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	93.42	0.88	0.96	0.93

شکل ۳۰.۵: مترهای مختلف عملکرد طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۳۱.۵: ماتریس CONFUSION طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست



شکل ۳۲.۵: منحنی ROC طبقه‌بند KNN با پیش‌پردازش ICA بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همان طور که در نتایج بالا دیده می‌شود، استفاده از روش KNN برای تخمین likelihood و استفاده از این مقدار در طبقه‌بند بیز، منجر به دقت خوبی روی داده‌های آموزش می‌گردد. به بیان دیگر، همان طور که از ماتریس آشفتگی می‌توان دید، طبقه‌بند طراحی شده به خوبی داده‌های کلاس صفر و یک را جدا کرده و هیچ خطایی ندارد. از طرف میزان AUC یک نیز نشان‌دهنده توانایی بالای طبقه‌بند در تمیز دادن داده‌های کلاس‌ها از یکدیگر است. در رابطه با داده‌های تست، مشاهده می‌شود که طبقه‌بند دچار خطاهایی در دسته‌بندی هر دو کلاس شده است.

اما همان طور که نتیجه ماتریس آشفتگی نشان می‌دهد، طبقه‌بند کلاس یک را بهتر از کلاس صفر به خاطر دارد. به همین علت هم میزان f1-score آن به مقدار قابل توجهی بیشتر است. این موضوع می‌تواند به علت بیشتر بودن داده‌های کلاس یک و در اصل نابالانس بودن داده‌ها باشد. چراکه در این صورت تعداد همسایگی‌های نزدیک به آن بیشتر از توع کلاس یک خواهند بود. با این حال می‌توان گفت میزان دقت و AUC قابل قبول هستند.

.

فصل ۶

یادگیری تجمیعی

یادگیری تحمیعی، به نوعی از یادگیری گفته می‌شود که برمبنای ترکیب چند مدل یادگیری مختلف باشد. در واقع هدف از این ترکیب، تحلیل یک مدل بهینه‌تر بر مبنای نقاط قوت متعدد مدل‌های مختلفاست. اصولا استفاده از این نوع مدل یادگیری دو دلیل مهم را در بر دارد:

۱. بهبود عملکرد

ترکیب مدل‌های مختلف یادگیری به افزایش دقت و کارایی کمک می‌نماید و می‌تواند موجب بهبود آن شود، به طوریکه هیچ مدل یادگیری به تنهایی قابلیت دستیابی به دقت مذکور را نداشته باشد.

۲. مقاوم بودن مدل [۱] استفاده از مدل‌های مختلف می‌تواند حساسیت به نویز، اغتشاش و سایر خطاهای غیرذاتی مدل را کاهش دهد.

موارد مطرح شده را می‌توان مانند مصالحه بایاس و و واریانس [۲] در نظر گرفت.

۱.۶ Bagging

نخست توجه بفرمایید که به نمونه برداری با جایگذاری از دیتاست، Bootstrap گفته می‌شود. همانطور که در بخش‌های قبل ذکر شد، یک درخت علی‌رغم بعضا بالا، دارای واریانس زیاد است. ایده اصلی برای رفع مشکل واریانس زیاد، استفاده از یادگیری تجمیعی است. در روش Bagging که یکی از روش‌های یادگیری تجمیعی است، یک طبقه‌بند خاص(برای مثال درخت تصمیم) در نظر گرفته می‌شود، سپس بر روی Bootstrap های مختلف از دیتا، تمامی مراحل آموزش (پیش‌پردازش، انتخاب ویژگی‌هاو...)به صورت مستقل پیاده سازی می‌گردد. در نهایت فرآیند طبقه‌بندی بر مبنای تحلیل آماری بر روی تمامی مدل‌ها(برای مثال بیشترین رای) انجام می‌شود. با این روش عملا وارایانس طبقه‌بند افت می‌کند. در روش مذکور، در انتخاب مدل‌های مختلف سعی می‌گردد مدل‌هایی با دقت و واریانس بالا مورد استفاده قرار بگیرند تا در نهایت و با تحلیل آماری بر روی طبقه‌بندهای مختلفی که آموزش داده شده‌اند واریانس نیز به همراه بایاس به حد مطلوبی همگرا شود.

[1] Robustness

[2] Bias-Variance Tradeoff

۲.۶ پیاده سازی Bagging با مدل پایه‌ی DT

۱.۲.۶ پیش‌پردازش در Bag-DT

در پیش پردازش این مدل کاملا مانند مدل DT عمل می‌گردد.

۲.۲.۶ طراحی طبقه‌بند

در یادگیری Bagging سعی می‌شود ترکیب‌های متفاوت طبقه‌بندها را در نظر گرفته، و سپس با استفاده از Grid Search به جستجوی ترکیب بیهنه می‌پردازیم. در این بخش سعی می‌گردد نسبت به حالت طبقه‌بند درخت معمولی از درخت‌های عمیق‌تری استفاده گردد تا دقت افزایش یافته سپس با استفاده از Bagging واریانس آن نیز کاهش یابد.

پیش‌پردازش **Autoencoder**

پارامترهای مسئله در ماژول grid به صورت شکل ۱.۶ انتخاب می‌شوند.

```
criterion_set =   ['gini', 'entropy']
max_depth_set =  [10, 12, 16, 20, 24]
samples_spli_set = [2, 3, 4]
```

شکل ۱.۶: پارامترهای ساختاری

در این حالت و با در نظر گرفتن، ۲۰ طبقه‌بند در فرآیند Bagging بهترین پارامترها به صورت شکل ۲.۶ می‌باشند:

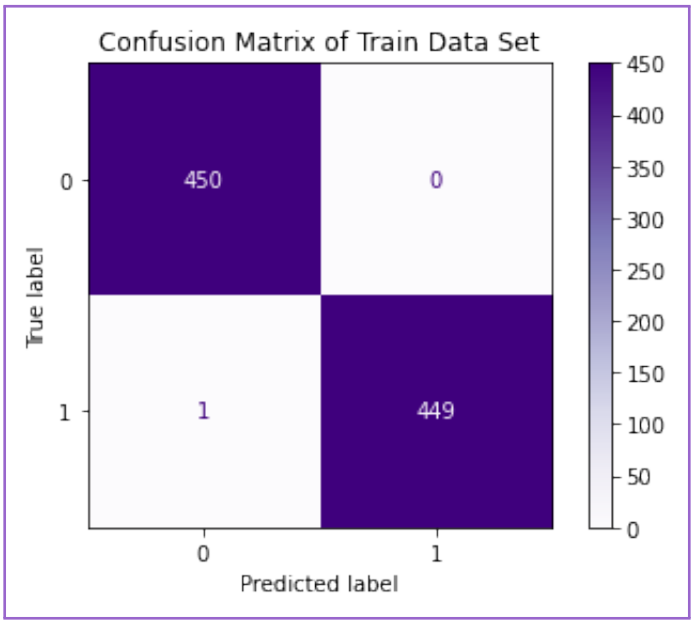
```
Tuned hpyerparameters are: {'base_estimator': DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=10, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')}
```

شکل ۲.۶: پارامترهای بهینه مدل

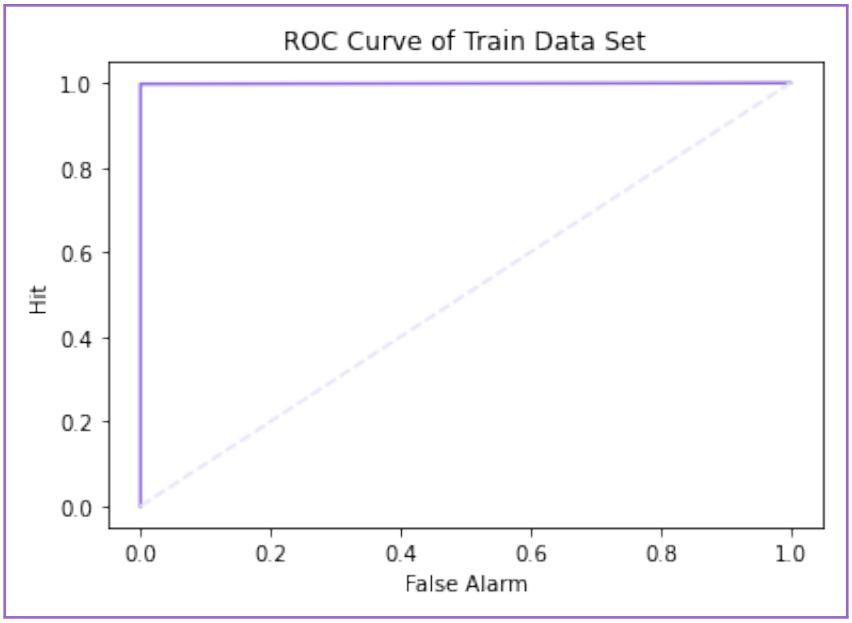
حال با استفاده از پارامترهای بیهنه به آموزش مدل می‌پردازیم مشاهده می‌گردد که معیارهای مختلف نتایج زیر خاصل می‌گردد:

Metric Type	Accuracy (with CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	90.78±1.49	1	1	1

شکل ۳.۶: مترهای مختلف عملکرد طبقه‌بند BAG-DT با پیش‌پردازش AUTOENCODER بر روی داده‌های آموزش



شکل ۴.۶: مترهای مختلف عملکرد طبقه‌بند BAG-SVM با پیش‌پردازش PCA بر روی داده‌های آموزش

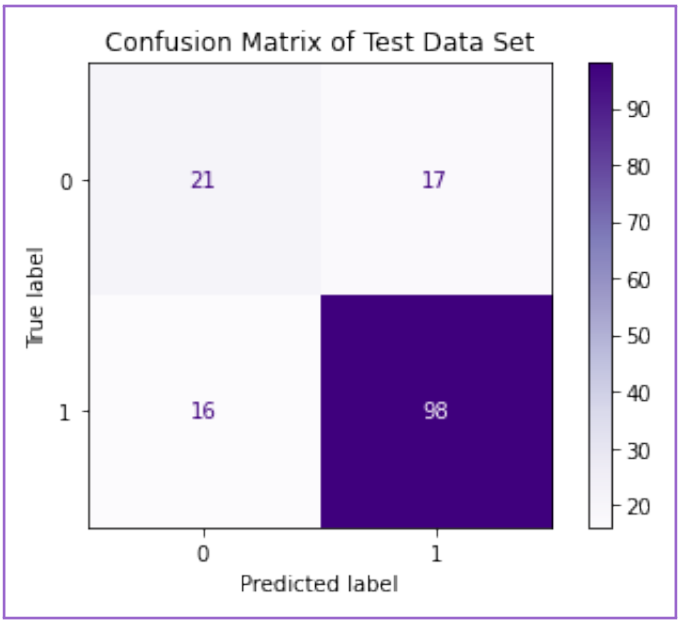


شکل ۵.۶: منحنی ROC طبقه‌بند BAG-DT با پیش‌پردازش AUTOENCODER

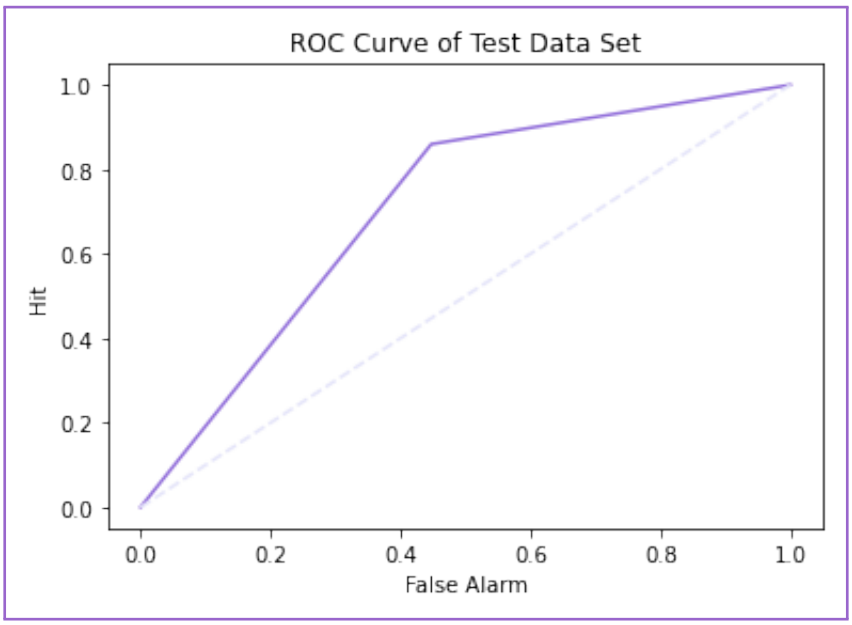
همچنین پس از آموزش مدل به بررسی عملکرد آن بر روی داده‌های تست می‌پردازیم:

Metric Type	Accuracy (without CV)(%)	f1 Score of Class 0	f1 Score of Class 1	Area Under Curve
Values	78.29	0.56	0.86	0.71

شکل ۶.۶: مترهای مختلف عملکرد طبقه‌بند BAG-DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۷.۶: ماتریس CONFUSION طبقه‌بند BAG-DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست



شکل ۸.۶: منحنی ROC طبقه‌بند BAG-DT با پیش‌پردازش AUTOENCODER بر روی داده‌های تست

تحلیل و نتیجه‌گیری

همانطور که مشاهده شد، با استفاده از تکنیک Bagging به صورت افزایش دقت و کاهش واریانس بدست آمد. در واقع استفاده از زیر مدل‌های با واریانس بالا و در نهایت تحلیل آماری آنها منجر به نتیجه‌ی مطلوب گردیده و افزایش دقت در حدود 2% بدست آمده است. از طرفی مشکل روش فوق افزایش حجم محاسبه و افزایش پیچیدگی مدل می‌باشد، که در ازای بهبود دقت بدست آمده است.

فصل ۷

مقایسه و تحلیل مدل‌ها

۱.۷ بررسی و مقایسه

در گزارش فوق به تشخیص بیماری پارکینسون با استفاده از دیتاست مربوط به تلفظ حرف "آ" پرداخته شد. در طی این بررسی انواع طبقه‌بندها و روش‌های پیش‌پردازش و انتخاب ویژگی مورد بررسی قرار گرفت. در بررسی طبقه‌بندها می‌توان شاخص‌هایی مانند دقت تشخیص، واریانس طبقه‌بندها و سرعت آموزش و اجرای آن‌ها را مدنظر قرار داد. طبقه‌بندهای SVM ، MLP ، RBF Neural Nets،Logistic Regression و Decision Tree زمان آموزش زیادی دارند اما استفاده از آنها بسیار پر سرعت است. از طرفی طبقه‌بندهای KNN علی‌رغم آنکه زمان آموزش بسیار سریعی دارند، اما استفاده از آن‌ها برای امر تشخیص بسیار زمانبر می‌باشد. همچنین طبقه‌بند Decision Tree دارای واریانس ذاتی بالایی است که می‌توان با استفاده از روش Bagging ضمن افزایش دقت درخت، واریانس آن را نیز کاهش داد. به طوریکه دقت آموزش درخت از $84.44\% \pm 2.9\%$ به $90.78\% \pm 1.49\%$ رسیده و دقت تست از 76% به 78% بهبود یافته است. اما استفاده از روش Bagging در سایر روش‌ها مانند SVM و KNN موجب بهبود نمی‌گردد، که این موضوع به این دلیل است که این طبقه‌بندها به طور ذاتی با واریانس کمی مواجه هستند و عملا با استفاده از تکنیک Bagging تفاوت معنا داری ایجا نمی‌شود.

یکی از چالش‌های ایجاد شده در طی روند بررسی بالانس نبودن دیتا بوده است. اگرچه سعی شده است با استفاده از تکنیک‌های مرتبط با بالانس کردن دیتا مانند SMOTE تا حدی مشکل مرتفع شود اما اغلب مشاهده می‌شد که تفاوت قابل توجهی میان دقت آموزش و تست وجود دارد و مطابق ماتریس Confusion مشاهد می‌شود که عمده‌ی خطای طبقه‌بندها در هنگام تشخیص اعضای کلاس با دیتای کم می‌باشد. طبقه‌بندهایی مانند KNN و MLP در تعیین مرز تصمیم‌گیری موفق‌تر عمل کردند و کمتر تحت تاثیر بالانس نبودن دیتا قرار گرفتند اما بعضا مدل‌های Generative مانند GMM بسیار متاثر مورد ذکر شده بوده‌اند.

همچنین در امر پیش‌پردازش، مشاهده می‌گردد که روش‌های LDA, SBF چندان موفق عمل نکردند. در واقع روش LDA باتوجه به دو کلاسه بودن مسئله، مسئله را به بعد یک تبدیل می‌نماید که منجر به از دست رفتن اطلاعات زیاد می‌گردد و از طرفی روش SBF باتوجه به بعد بسیار بالای مسئله حتی به صورت ترکیب با روش‌های کاهش بعد مانند PCA, Autoencoder به دلیل حجم محاسباتی بسیار زیاد عملا قابل استفاده نمی‌باشد. اما سه روش ICA, Autoencoder, PCA به طور گسترده مورد استفاده قرار گرفته‌اند. روش Autoencoder باتوجه به

آنکه توانایی حذف وابستگی‌های خطی و غیرخطی را دارد، توانایی کاهش بعد مسئله را به ابعاد پایین‌تر از فضای تبدیل شده‌ی PCA دارد. در نتیجه استفاده از Autoencoder در بسیاری از طبقه‌بندها نسبت به PCA منجر به سرعت و دقت بیشتری شده است. همچنین ICA در طبقه‌بندهایی مانند KNN که حذف تمام وابستگی‌ها بدون کاهش بعد مورد اهمیت است استفاده شده، همچنین در شبکه MLP نیز از پیش‌پرداش مذکور استفاده شده است.

در جدول صفحه‌ی آینده جزئیات بررسی مدل‌ها آورده شده است:

ردیف	پیش‌پردازش	طبقه‌بند	دقت آموزش	دقت تست	زمان آموش	زمان اجرا
۱	ICA	KNN	95.67 ± 2.79	93.42	بسیار کم	زیاد
۲	AE	LR	83.56 ± 5.35	80.92	زیاد	کم
۳	AE	SVM	95.89 ± 2.44	89.47	زیاد	بسیار کم
۴	ICA	MLP	95.67 ± 2.92	91.45	بسیار زیاد	بسیار کم
۵	AE	Parzen	95.89 ± 3.55	89.47	زیاد	کم
۶	ICA	KNN(Generative)	95.67 ± 2.79	93.42	زیاد	کم
۷	AE	GMM	92.44 ± 5.68	82.89	زیاد	کم
۸	AE	DT	84.44 ± 2.9	76.32	زیاد	کم
۹	AE	Bag-DT	90.78 ± 1.49	78.29	بسیار زیاد	کم
۱۰	AE	RBF	73.11 ± 4.38	81.58	بسیار زیاد	کم

همانطور که مشاهده می‌شود در نهایت طبقه‌بند KNN نسبت به سایر طبقه‌بندها موفق عمل کرده است. همچنین همانطور که مشاهده می‌گردد استفاده از روش Bagging موجب شده است که طبقه‌بند مذکور کمترین واریانس را میان تمامی طبقه‌بندهای دیگر داشته باشد. از آنجا که با یک مسئله Real Time مواجه نیستیم به نظر می‌رسد استفاده از طبقه‌بند KNN می‌تواند مناسبترین گزینه باشد.

۲.۷ کارهای آینده

به نظر می‌رسد در امر انتخاب ویژگی بتوان از الگوریتم‌هایی مبتنی بر ویژگی‌های آماری استفاده نمود. همچنین تقویت دیتاست یکی دیگر از گزینه‌های آینده می‌باشد. همچنین استفاده از شبکه‌های عمیق [1] به شرط جمع‌اوری دیتاست غنی‌تر می‌تواند از جمله کارهای آینده باشد.

[1] Deep Neural Nets