دانشگاه شهیدبهشتی

دانشکده مهندسی و علوم کامپیوتر

پروژه درس شبکه‌های عصبی

تشخیص دیابت بارداری به کمک شبکه عصبی RBF

نگارش :

پدرام یزدی پور

شماره دانشجویی:

۹۹۴۴۳۲۴۱

استاد درس:

دکتر آرمین سلیمی بدر

بهمن ۱۴۰۰

فهرست مطالب

فهرست اشکال و جداول

چکیده

هدف ما در این پروژه توسعه‌ی یک شبکه عصبی RBF [1]برای مسئله‌ی تشخیص بیماری دیابت بارداری است. در این پروژه، ابتدا روی داده‌های آموزشی خوشه‌بندی انجام می‌شود و سپس به کمک روش درون‌یابی، وزنهای شبکه به دست می‌آیند. از جمله چالش‌های موجود در موضوعات پزشکی می‌توان به کمبود داده اشاره کرد که ممکن است باعث افت عملکرد مدل شود. رویکرد مقاله‌ی [۱] استفاده از یک شبکه عصبی RBF تک لایه است که دقت مناسبی را هم در آزمایشات نشان داده است. در این پروژه پس از پیاده‌سازی الگوریتم ذکر شده، پنج آزمایش شامل تعیین تعداد خوشه‌های بهینه، مقایسه‌ی زمان اجرا به ازای الگوریتم‌های مختلف خوشه‌بندی، تکرارپذیری نتایج، تاثیر کاهش ابعاد بر عملکرد مدل و بررسی تفاوت هنجارسازی فاصله‌ها با دو ماتریس قطری واریانس و ماتریس کوواریانس است. در آخر برای ادامه‌ی مسیر پژوهشی این پروژه، پیشنهاداتی ارائه شده است.

کلمات کلیدی فارسی

تشخیص دیابت، شبکه عصبی با توابع پایه‌ی شعاعی، هوش مصنوعی، شبکه عصبی، بارداری

کلمات کلیدی انگلیسی

Gestational diabetes Diagnosis, Radial Basis Function Neural Network, Artificial Intelligence, Neural Network ,Pregnancy

[1] Radial Basis Function

۱ – مقدمه

۱-۱ موضوع پروژه

دیابت زمان بارداری تا ۱۸ درصد زنان را می‌تواند درگیر کند. این بیماری هر ساله هزینه‌های هنگفتی را بر سیستم درمانی کشورهای مختلف جهان تحمیل کرده و می‌تواند در موارد حاد، موجب قطع عضو یا مرگ بسیاری از مادران شود. در این پروژه بر آن هستیم تا با استفاده از شبکه عصبی RBF یک مدل هوشمند برای تشخیص بیماری دیابت توسعه دهیم تا در صورت امکان در روند تشخیص و درمان آن موثر واقع شود.

۱-۲ چالش‌ها و پیشینه

از جمله چالش‌های موجود در این پروژه و سایر کارهای پژوهشی در حوزه‌ی پزشکی، کمبود داده است. برای رفع این چالش در صورتی که دسترسی به داده‌های بیشتر ممکن نباشد، تنها راه استفاده از مدلهای دقیق است. در مقاله‌ی [۲] از یک شبکه عصبی پرسپترون چندلایه برای این مسئله و با همین مجموعه داده استفاده شده است. نتایج [۲] به نتایج مقاله‌های [۱] و [۳] که اولی مقاله‌ی مرجع ما و دومی الگوریتم درخت تصمیم هستند، بسیار نزدیک است. مزیت درخت تصمیم و همچنین شبکه عصبی RBF، تفسیرپذیر بودن این مدلهاست. یکی دیگر از چالشهای مسائل پزشکی، قانع کردن پزشکان برای استفاده از این مدلهای هوشمند است و در صورت تفسیرپذیر بودن، می‌توان آنها را به استفاده از هوش مصنوعی برای تشخیص و درمان بیماران ترغیب نمود.

۱-۳ روش پیشنهادی

در این پروژه، از یک شبکه عصبی تک لایه‌ی RBF برای تشخیص بیماری دیابت استفاده شده است. پس از خوشه‌بندی داده‌های آموزشی، وزنهای شبکه به کمک درون‌یابی روی مراکز خوشه‌ها تعیین می‌شوند. این روش، متفاوت از روش‌های یادگیری همراه با تکرار مانند پس‌انتشار خطا است و وزنها به صورت همزمان و طی یک عملیات جبرخطی حاصل می‌شوند.

۱-۴ ساختار گزارش

در فصل دوم به بررسی روش پیشنهادی مقاله‌ی [۱] خواهیم پرداخت. در فصل سوم آزمایشات مختلف را انجام خواهیم داد. در فصل چهارم نیز به جمع‌بندی و نتیجه‌گیری کار می‌پردازیم.

۲- روش ارائه‌شده

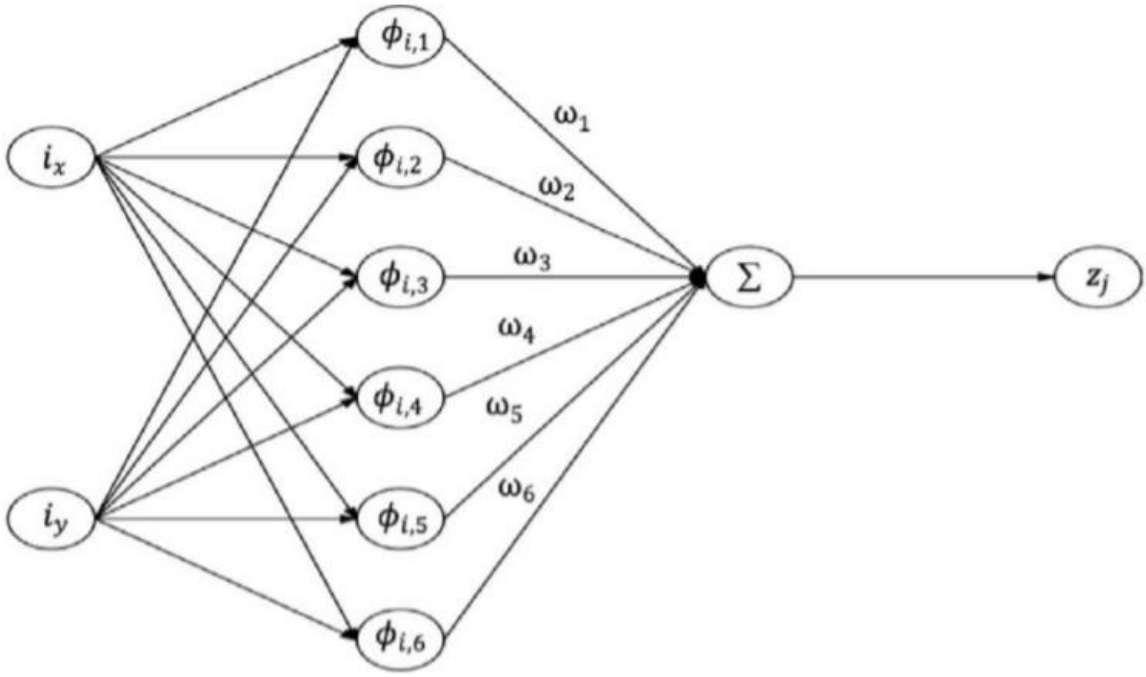
۲-۱ مقدمه:

در این قسمت بعد از ارائه‌ی توضیحات کلی راجع به رویکرد مقاله‌ی [۱]، هر قسمت با جزئیات کامل و شکل‌های مربوط پوشش داده خواهد شد تا روش پیشنهادی کاملا روشن شود؛ سپس به مقایسه‌ی اجمالی این پژوهش با تحقیقات پیشین خواهیم پرداخت.

۲-۲ توضیح کلیات:

در مقاله‌ی [۱]، برای تشخیص بیماری دیابت در زمان بارداری، از شبکه عصبی تک لایه‌ی RBF استفاده شده است. شکل ۱-۲ تصویری از ساختار کلی این شبکه را نمایش می‌دهد. در این شبکه، هر داده با مراکز خوشه‌ها به وسیله‌ی نورونها شباهت‌سنجی شده و سپس خروجی هر نورون در وزن مربوطه ضرب شده و مجموع این مقادیر از یک تابع فعالسازی عبور داده می‌شود.

شکل ۱-۲: ساختار کلی یک شبکه عصبی RBF



۲-۳ توضیح روش درون‌یابی برای یافتن وزن‌های شبکه:

در مقاله‌ی [۱]، برای خوشه‌بندی از الگوریتم K-Means بهره برده شده اما متاسفانه برخی جزئیات کار به طور دقیق گفته نشده و به همین علت، در پروژه‌ی حاضر سعی کرده‌ایم نقاط تاریک موجود را روشن نماییم. در این روش، برای یافتن مقادیر وزنهای شبکه از رویکرد درون‌یابی استفاده شده است. راه‌های دیگری از جمله الگوریتم‌های تکاملی، الگوریتم پس‌انتشار خطا و شبه‌معکوس نیز قابل استفاده‌اند. همانطور که در رابطه‌ی ۱-۲ قابل مشاهده است ابتدا خوشه‌بندی انجام شده و بعد مراکز خوشه‌ها با هم فاصله‌سنجی می‌شوند؛ این اتفاق را می‌توانیم این‌گونه تفسیر کنیم که مراکز خوشه‌ها به عنوان داده‌های آموزشی به شبکه داده می‌شود و بعد وزنهای شبکه طوری تعیین می‌شود که خروجی شبکه دقیقا برابر برچسب‌های مراکز خوشه‌ها باشند. منظور از Φ همان خروجی یک نورون یا به عبارتی میزان شباهت دو مرکز خوشه است. منظور از Z نیز خروجی واقعی هر مرکز خوشه است. به این ترتیب، وقتی داده‌ای جدید به شبکه داده می‌شود ابتدا با تک‌تک مراکز خوشه‌ها(به عنوان نماینده‌ی هر خوشه) شباهت‌سنجی می‌شود و بعد مقدار هر شباهت در وزن نظیرش ضرب شده و مجموع این مقادیر از یک تابع فعالسازی پله عبور داده می‌شود تا خروجی نهایی شبکه یا صفر باشد یا یک. هر چه یک داده به یک مرکز خوشه نزدیکتر باشد، خروجی آن(برچسب پیش‌بینی شده) به برچسب آن مرکز خوشه نزدیکتر است.

$$\overbrace{\begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \cdots & \phi_{1,n} \\ \phi_{2,1} & \phi_{2,2} & \cdots & \phi_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n,1} & \phi_{n,2} & \cdots & \phi_{n,n} \end{bmatrix}}^{\Phi} \overbrace{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_n \end{bmatrix}}^{\Omega_r} = \overbrace{\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}}^{Z} \longrightarrow \phi \cdot \Omega_r = Z \Rightarrow \Omega_r = \phi^{-1} Z$$

رابطه‌ی (۱-۲):

در واقع در این روش، یادگیری با تکرار صورت نمی‌گیرد و تمام وزن‌ها به صورت همزمان محاسبه می‌شوند. چالش اصلی این روش این است که باید برچسب مراکز خوشه‌ها را داشته باشیم؛ اما در روش خوشه‌بندی K-Means مراکز خوشه لزوما از خود داده‌ها نیستند و برچسب آنها در دسترس نیست. راه حل ما در این پروژه استفاده از الگوریتم PAM [٤][۲] است.

۲-۴ الگوریتم خوشه‌بندی PAM:

الگوریتم PAM یک ماتریس مربعی به اندازه‌ی تعداد داده‌ها به عنوان ماتریس فواصل و نیز تعداد خوشه‌های مورد نظر را به عنوان ورودی دریافت می‌کند. در ابتدا مانند K-Means، تعدادی مراکز تصادفی را انتخاب

[۲] Partitioning Around Medoids

می‌کند و سپس، هر داده را به نزدیکترین مرکز نسبت می‌دهد. حلقه‌ی اصلی الگوریتم اینجاست که تا زمانی که هزینه‌ی خوشه‌بندی کاهش می‌یابد، انجام بده: (هزینه در این الگوریتم برابر مجموع فواصل هر داده با مرکز خود است و فاصله‌ها نیز از ماتریس ورودی که ذکر شد، قابل استخراج است.)

برای هر مرکز، هر داده‌ی غیر مرکزی را با نزدیکترین مرکز تعویض کن(مرکز جدید تعیین شد) و هزینه را محاسبه کن، اگر هزینه کاهش نیافت، این تعویض مرکز را برعکس کن.

مزایای PAM:

۱- قابل درک و ساده برای پیاده‌سازی
۲- سریع بوده و همگرایی‌اش قطعی است.
۳- نسبت به داده‌های پرت، حساس نیست.

معایب PAM:

۱- از فشردگی داده‌ها به جای اتصال آنها به هم برای خوشه‌بندی استفاده می‌کند و در نتیجه خوشه‌های به شکل دلخواه(و نه کروی) را نمی‌تواند تشخیص دهد.
۲- به ازای هر بار اجرا، به دلیل انتخاب تصادفی مراکز اولیه، نتایج کاملا متفاوتی را به دست می‌دهد.

نسخه‌ی جدیدتر این الگوریتم، در مقاله‌ی [۵] آمده است. در این نسخه، نتایج کاملا با الگوریتم اصلی یکسان است اما در زمان کمتری نتیجه می‌دهد. در نسخه‌های سریعتر PAM، تعویض داده‌ها با مراکز پیشین کمی هوشمندانه‌تر انجام شده و از برخی محاسبات قبلی که ذخیره شده، مجددا استفاده می‌شود.

۳- آزمایشات

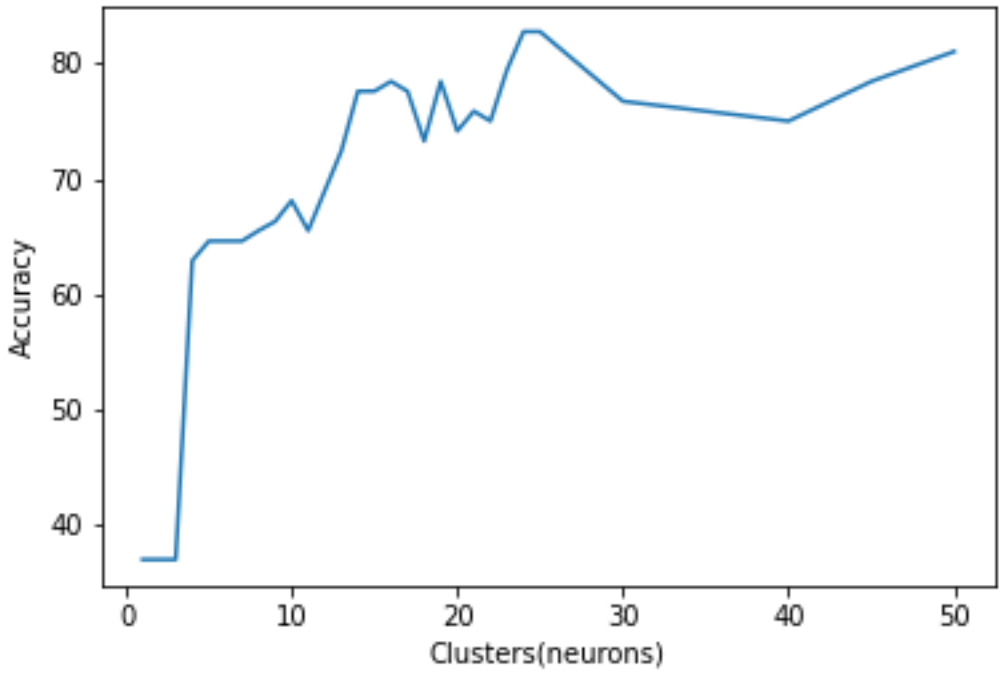
۳-۱- مقدمه

در این بخش تمام آزمایشات روی سیستم Google Colaboratory و به زبان پایتون پیاده‌سازی و اجرا شده است. برای ارزیابی عملکرد مدل ارائه‌شده در این پروژه از چهار معیار Accuracy, Precision, Recall و False Positive Rate استفاده شده است. معیار Accuracy عملکرد مدل را به صورت کلی و برای هر دو کلاس نمونه‌های مثبت و منفی می‌سنجد. معیار Precision دقت مدل را در درستی نمونه‌های مثبت اعلام شده می‌سنجد و معیار Recall نشان‌گر درصد نمونه‌های مثبتی است که درست تشخیص داده شده‌اند. معیار FPR نیز از این جهت مهم است که باید بدانیم چند درصد از نمونه‌های مثبت اعلام شده اشتباه بوده و هزینه‌ی انجام آزمایشات پزشکی ممکن است چقدر بالا رود.

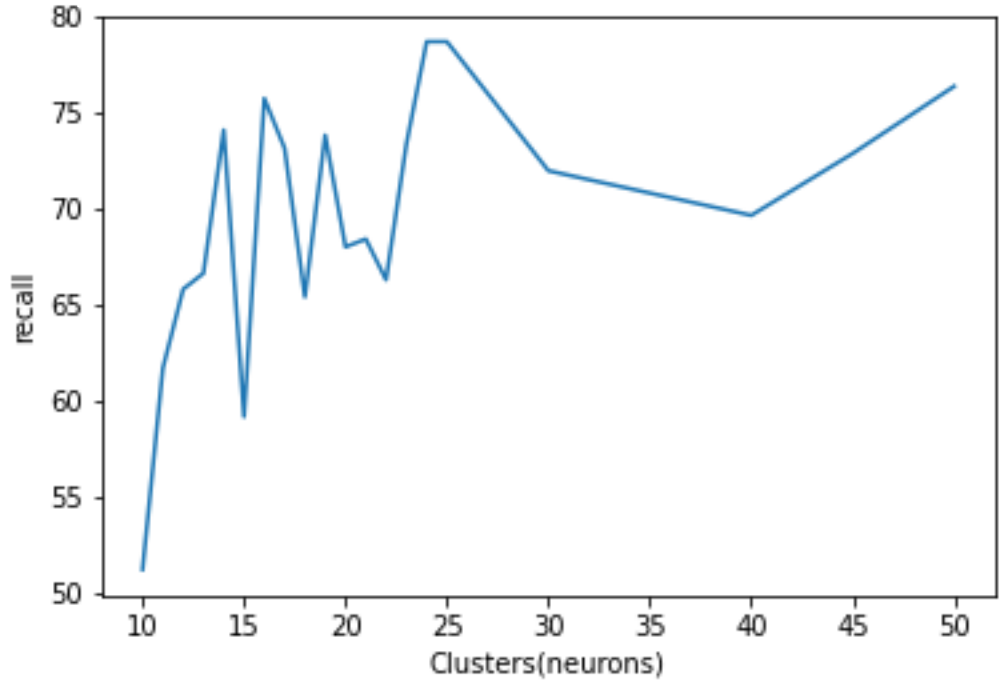
۳-۲- آزمایش اول: تعیین تعداد بهینه‌ی خوشه‌ها

اولین گام در راستای رسیدن به دقت‌های بالا در این روش، تعیین تعداد نورون یا همان خوشه‌های لازم برای پوشش‌دهی کامل فضای جستجو است. برای این هدف، سه معیار ارزیابی Accuracy، Recall(Macro) و Precision(Macro) را در نظر می‌گیریم و به ازای هر تعداد نورون، نمودارهای ارزیابی عملکرد مدل را رسم می‌کنیم تا سپس با مقایسه‌ی این نمودارها بتوانیم تعداد مورد نیاز نورون را مشخص نماییم. تعداد کل داده‌ها ۷۶۸ نمونه است که البته روی ۸۵ درصد این داده‌ها به عنوان داده‌های آموزشی معادل ۶۵۲ نمونه، خوشه‌بندی انجام می‌شود(چندین بار با تقسیم‌بندی‌های مختلف تصادفی)؛ در نتیجه، تعداد خیلی کم خوشه ممکن است نتواند کل فضا را به درستی پوشش دهد و تعداد بالای خوشه‌ها نیز داده‌ها کمتری را در یک خوشه قرار می‌دهد و مدل را پیچیده می‌کند که همین عامل ممکن است موجب بیش‌برازش شده و با کاهش قابلیت تعمیم‌پذیری مدل، خطای آنرا افزایش دهد. برای انتخاب تعداد خوشه‌ها، فعلا از معیار False Positive Rate چشم‌پوشی می‌کنیم؛ زیرا در اولویت نیست و ضمنا چون موضوع مورد بحث مربوط به پزشکی است، نرخ تشخیص مثبت نادرست اگر کمی بالاتر باشد، ضریب اطمینان را افزایش می‌دهد و البته آزمایش‌های مربوط به بیماری دیابت چندان هزینه‌بر نیستند. شکل ۳-۱ و ۳-۲ و ۳-۳ مربوط به نمودارهای معیارهای Accuracy، Recall و Precision بر حسب تعداد خوشه‌ها(نورون‌ها) است.

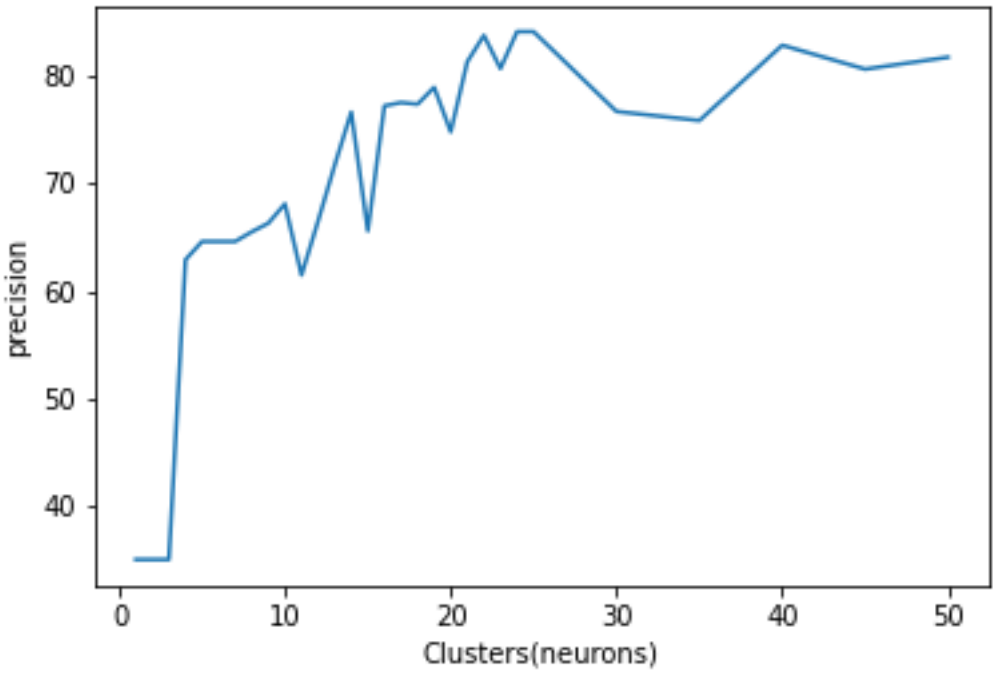
شکل ۱-۳: نمودار Accuracy بر حسب تعداد نورون‌های شبکه عصبی RBF



شکل ۲-۳: نمودار Recall بر حسب تعداد نورون‌های شبکه عصبی RBF



شکل ۳-۳: نمودار Precision بر حسب تعداد نورون‌های شبکه عصبی RBF



جدول ۱-۳: معیارهای ارزیابی برای ۲۴ نورون

Accuracy	۸۲.۷۵ ٪
Precision(Macro)	۸۴.۱۳ ٪
Recall(Macro)	۷۸.۶۵ ٪
False Positive Rate	۵.۴۷ ٪

از مشاهده‌ی نمودارهای بالا می‌توان به وضوح، تعداد ۲۴ نورون را گزینه‌ی مناسبی دانست؛ زیرا از هر سه معیار، بالاترین امتیاز را کسب کرده است. معیارهای ارزیابی مربوط به انتخاب ۲۴ نورون برای این مدل را در جدول ۱-۳ ملاحظه می‌فرمایید.

از بررسی جدول ۱-۳ نتیجه می‌شود که جز معیار فراخوانی، از سایر جهات نتایج کسب شده نسبت به مقاله‌ی انتخابی برای سمینار، بهتر است. افزایش تعداد نورون‌ها از حدود ۳۰ به بالا، باعث وقوع مشکل Singularity در ماتریس فواصل می‌شود و در نتیجه نیاز است تا به اعداد روی قطر اصلی ماتریس، یک مقدار مثبت کوچک افزوده شود تا این مشکل به کمک افزایش رتبه‌ی ماتریس مرتفع گردد؛ اما این کار، با دورتر کردن مقدار فواصل از هم موجب ایجاد خطا در عملکرد مدل شده و دقت مدل را کاهش می‌دهد. آخرین تعداد بالای نورون که

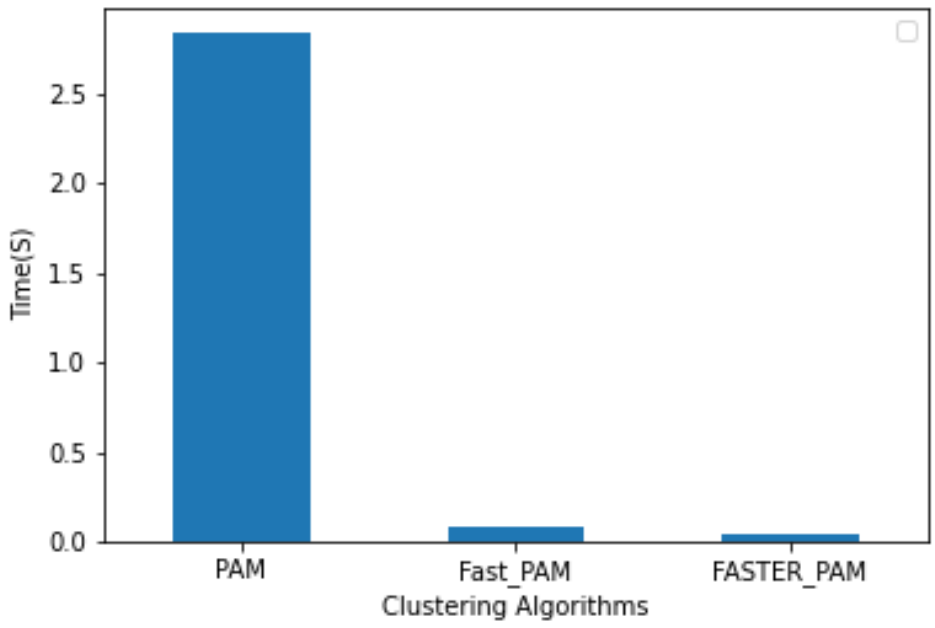
دقت‌های خوبی را ارائه می‌دهد عدد ۵۰ است. به دلیل کاهش دقت به ازای بیش از ۵۰ نورون و وجود حالات بسیار زیاد(می‌توان به تعداد داده‌ها، خوشه تعریف نمود)، نمودار تا ۵۰ نورون رسم شده است تا تمرکز روی تعداد پایینتر خوشه‌ها باشد. افزایش تعداد خوشه‌ها همچنین باعث افزایش سربار محاسباتی شده و خطر بیش‌برازش را به همراه خواهد داشت.

۳-۳- آزمایش دوم: بررسی زمان اجرا به ازای الگوریتم‌های مختلف خوشه‌بندی

در روش درون‌یابی نیاز است تا برچسب تمام مراکز خوشه‌ها را داشته باشیم؛ بنابراین مراکز خوشه‌ها باید لزوما از بین خود داده‌ها انتخاب شوند و به همین دلیل، الگوریتم K-Means قابل استفاده نیست. یکی از الگوریتم‌های مورد استفاده PAM است. این الگوریتم دو نسخه‌ی جدیدتر نیز دارد که ادعا شده سریعتر عمل می‌کنند. از آنجا که عملکرد مدل در این پروژه، به شدت به خوشه‌بندی وابسته است(هم زمان هم دقت)، کاوش بیشتر در این زمینه لازم است.

شکل ۴-۳، نمودار میله‌ای مربوط به مقایسه‌ی دو نسخه‌ی PAM با خود آن قابل مشاهده است. در واقع، زمان اجرای کد، تنها به ازای قسمت خوشه‌بندی آزمایش شده چون باقی قسمتها نظیر محاسبه‌ی وزنها یکسان است.

شکل ۴-۳: زمان اجرای سه نسخه‌ی PAM



همانطور که از نمودار شکل ۴-۳ مشخص است، نسخه‌ی سریعتر همان نسخه‌ی جدیدتر است و بسیار سریعتر است(چیزی در حدود ۹۳ برابر!). قسمت کلیدی این پروژه مربوط به خوشه‌بندی است و بخش قابل توجهی از

زمان اجرای مدل برای یافتن وزنها، به خوشه‌بندی اختصاص دارد؛ پس با انتخاب نسخه‌ی سریعتر الگوریتم خوشه‌بندی، می‌توان زمان اجرا را به شکل قابل توجهی کاهش داد.

۴-۳- آزمایش سوم: بررسی تکرارپذیر بودن نتایج

در این پروژه، وزنها به کمک یک عملیات ماتریسی حاصل می‌شوند. یکی از ماتریسها، برچسبهای مراکز خوشه است و دیگری مربوط به شباهت مراکز خوشه بر اساس یک تابع گاوسی نسبت به هم. در این آزمایش می‌خواهیم بررسی کنیم که آیا می‌توان دقتهای یکسانی را به ازای شرایط برابر انتظار داشت یا خیر.

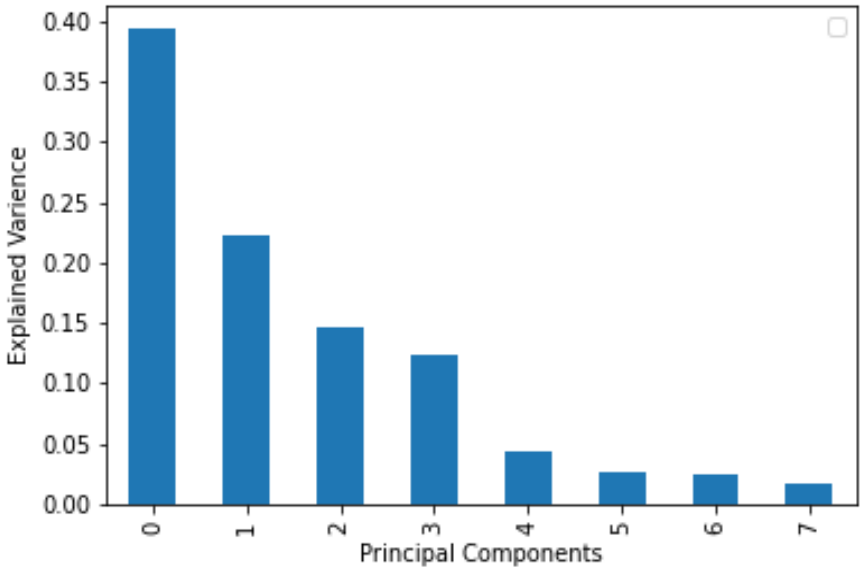
اگر کد را اجرا کنید متوجه می‌شوید دقتها به ازای هر بار اجرا بسیار نوسان دارند و پایدار نیستند. علت این موضوع، بسته به ماهیت تصادفی الگوریتم خوشه‌بندی است. به ازای ۲۴ نورونی که در آزمایش اول انتخاب شد، دقتها گاه تا ۶۰ درصد پایین می‌آیند؛ پس چطور می‌توان از این مدل انتظار عملکرد مناسبی داشت و آیا اصلا این مدل روش خوبی برای مسائل دسته‌بندی هست؟

پاسخ مثبت است. ماهیت این روش با سایر شبکه‌های عصبی که مبتنی بر الگوریتم‌های تنزل گرادیان هستند کمی متفاوت است. در این آزمایش، یک فرضیه در نظر می‌گیریم تا ببینیم قابل اثبات هست یا خیر. فرضیه‌ی ما این است که در صورتی که حتی یکبار به دقت بالایی رسیدیم، اگر نتایج خوشه‌بندی را ذخیره کنیم آیا می‌توانیم همان دقتهای بالا را انتظار داشته باشیم؟ برای این آزمایش، الگوریتم را به ازای ۲۴ خوشه آنقدر اجرا می‌کنیم تا بالاخره در یکی از اجراها، دقت مورد نظر(مثلا ۸۰ درصد) حاصل شود، سپس در اجرای دوم خوشه‌بندی نمی‌کنیم بلکه از نتایج خوشه‌بندی پیشین بهره می‌بریم. دقتها دقیقا برابر اعداد اجرای اول هستند و فرضیه تایید می‌شود. برای اثبات این موضوع، داده‌ها و اطلاعات خوشه‌بندی نظیر مراکز خوشه‌ها و تعلق هر داده به یک خوشه را در فایلهای جداگانه ذخیره کرده‌ایم و در فولدر MAX آورده‌ایم. این ادعا فقط در صورت اجرای کد قابل اثبات عملی است. در واقع پایداری الگوریتم معرفی شده در این پروژه نباید با دقتهای مختلف به ازای خوشه‌بندی‌های متفاوت ارزیابی شود؛ علت این است که وقتی خوشه‌بندی بهینه روی داده‌های آموزشی را یافتیم، می‌توانیم از آن بارها و بارها استفاده کنیم. آمار مربوط به این آزمایش می‌تواند برابری دقتها باشد که از آوردن آن خودداری می‌کنیم زیرا ارزش اطلاعاتی چندانی ندارد. این مدل برای پیش‌بینی داده‌های تست، به وزنها و مراکز خوشه‌ها و واریانس آنها نیاز دارد. وزنها هم به کمک خوشه‌بندی حاصل می‌شود، پس وقتی خوشه‌بندی ذخیره شود، حتی در صورت تغییر داده‌های تست، می‌توان نتایج مشابهی را انتظار داشت.

۵-۳- آزمایش چهارم: بررسی تاثیر کاهش ابعاد

کاهش فضای جستجو همواره به عنوان یک راه حل برای بهبود دقت و کاهش زمان اجرا مطرح بوده است. در این پروژه نیز، با استفاده از الگوریتم PCA[۳]، می‌خواهیم تاثیر کاهش ابعاد را بررسی نماییم. شکل ۵-۳، نشان می‌دهد هر مولفه‌ی اصلی چقدر از واریانس را نگه می‌دارد و ما بر اساس آن چقدر می‌توانیم به حذف برخی ویژگی‌ها امیدوار باشیم. لازم به ذکر است که پیش از اجرای PCA، داده‌ها نرمال‌سازی شده‌اند تا نتایج صحیح باشد.

جدول ۵-۳: نمودار واریانس – مولفه‌های اساسی برای تحلیل اطلاعات مولفه‌های اساسی



از بین ۸ مولفه‌ی اساسی، چهار مولفه‌ی اول می‌توانند ۸۸ درصد واریانس را حفظ کنند؛ اما نه تنها به ازای این حالات بلکه به ازای تمام ترکیبات، مدل را ارزیابی می‌کنیم و نتایج در جدول ۲-۳ آمده است. آنچه از جدول ۲-۳ مشخص است، اگر هدف رسیدن به بالاترین دقتها باشد، باید به داشتن ۷ مولفه‌ی اساسی اکتفا کرد و در واقع فضای جستجو به جای ۸ ویژگی به ۷ ویژگی تقلیل پیدا می‌کند. کاهش فضای جستجو منجر به بهبود نتایج خوشه‌بندی خواهد شد. اگر هم هدف کاهش حداکثری فضای جستجو ضمن حفظ دقتهای معقول باشد، می‌توانیم ۴ مولفه‌ی اساسی اول را در نظر بگیریم.

[۳] Principle Component Analysis

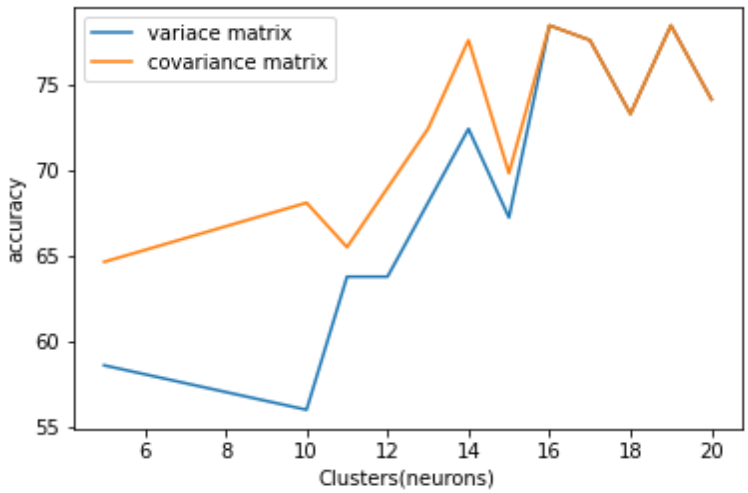
جدول ۲-۳: معیارهای ارزیابی به ازای تعداد مختلف مولفه‌های اساسی

Principle Components	Accuracy	Recall	Precision	FPR
۲	۶۸.۱۰	۶۱.۲۷	۷۱.۵۶	۵.۷۱
۳	۷۳.۲۷	۵۹.۵۷	۶۷.۶۲	۷.۳۱
۴	۷۵	۶۷.۲۴	۷۳.۴۶	۹.۰۰
۵	۷۵.۸۶	۶۶.۴۹	۷۱.۸۵	۹.۸۷
۶	۷۵.۸۶	۷۰.۸۵	۷۲.۳۶	۱۵.۷۸
۷	۸۲.۷۵	۷۸.۴۰	۸۱.۳۲	۸.۹۷
۸	۸۲.۷۵	۷۸.۴۰	۸۱.۳۲	۸.۹۷

۶-۳- آزمایش پنجم: بررسی تفاوت ماتریس هنجارسازی واریانس و کوواریانس در تابع RBF

می‌دانیم پوشش‌دهی فضای ویژگی‌ها با بیضی‌های چرخیده با استفاده از فاصله‌ی هنجارسازی شده به کمک ماتریس کوواریانس، نیاز به نورون‌های کمتری دارد و در نتیجه انتظار داریم که به ازای نورون‌های کمتر، پوشش‌دهی بهتری داشته باشیم. در این آزمایش، اثر استفاده از دو ماتریس متفاوت را بررسی می‌کنیم. نتایج معیار Accuracy در شکل ۶-۳ آمده است. از نمودار شکل ۶-۳ مشخص است که هر چه تعداد نورون‌ها کمتر باشد، پوشش‌دهی فضای ویژگی‌ها به کمک بیضی‌های چرخیده(ماتریس هنجارسازی کوواریانس) بهتر رخ داده و در نتیجه دقت بالاتر است؛ اما هر چه تعداد نورونها بیشتر می‌شود، تفاوت این دو ماتریس هنجارسازی کمتر می‌شود تا بالاخره در تعداد ۱۶ نورون، این تفاوت به صفر برسد. در واقع، چون ما با ۲۴ نورون توانستیم بیشترین دقت را به دست آوریم، تفاوتی نمی‌کند که از کدام ماتریس برای هنجارسازی فواصل استفاده کنیم مگر آنکه تعداد نورونها را کاهش دهیم و بعد برای رسیدن به عملکرد بهتر، از ماتریس کوواریانس استفاده کنیم.

شکل ۶-۳: مقایسه‌ی دقت به ازای تعداد نورونها و نوع ماتریس هنجارسازگر فواصل



جالب است که در تمام معیارهای ارزیابی به ازای ۱۶ نورون و بیشتر، هیچ تفاوتی میان دو ماتریس هنجارساز کننده وجود ندارد؛ به همین دلیل، سایر نمودارها را نیاوردیم.

۴- خلاصه و نتیجه‌گیری

در این پروژه، از یک شبکه عصبی تک لایه‌ی RBF برای تشخیص بیماری دیابت استفاده شده است. پس از خوشه‌بندی داده‌های آموزشی، وزنهای شبکه به کمک درون‌یابی روی مراکز خوشه‌ها تعیین می‌شوند. این روش، متفاوت از روش‌های یادگیری همراه با تکرار مانند پس‌انتشار خطا است و وزنها به صورت همزمان و طی یک عملیات جبرخطی حاصل می‌شوند. یکی از نقاط ضعف این روش، نیاز به اجراهای متعدد برای یافتن خوشه‌بندی بهینه و رسیدن به دقت‌های بالاست که می‌تواند زمان‌بر باشد. نقطه‌ی ضعف دیگر این است که همواره ممکن است خوشه‌بندی پیدا شود که دقت‌های بالاتری را نتیجه دهد و بنابراین شاید هرگز نتوان به بالاترین دقت‌های حاصل از n بار اجرای الگوریتم، به عنوان حداکثر ظرفیت مدل نگاه کرد؛ در واقع ضمانتی وجود ندارد که اگر به ازای تعدادی اجرا به دقت‌های بالایی رسیدیم، بتوانیم از رسیدن به ظرفیت مدل مطمئن شویم که از این جهت شبیه مدلهای دیگر شبکه عصبی یا یادگیری ماشین است چون آنها هم ممکن است نقطه‌ی بهینه‌ی سراسری را هرگز پیدا نکنند. از نقاط قوت این روش می‌توان به تفسیرپذیر بودن و سادگی پیاده‌سازی اشاره نمود؛ ضمن آنکه این روش عملکرد نسبتا خوبی را نشان داده است.

برای کارهای آتی، می‌توان از این روش به عنوان راه حلی جالب برای حل مشکل کوچک شدن گرادیان در شبکه‌های عمیق بهره برد و حتی خود شبکه عصبی RBF را چندلایه نمود و هر لایه را با درون‌یابی آموزش داد و سپس لایه‌ی بعدی را اضافه نمود.

مراجع:

[١] Moreira, Mário WL, Joel JPC Rodrigues, Neeraj Kumar, Jalal Al-Muhtadi, and Valeriy Korotaev. "Evolutionary radial basis function network for gestational diabetes data analytics." *Journal of computational science* ٢٧ (٢٠١٨): ٤١٠-٤١٧.

[٢] R.M. Rahman, F. Afroz, Comparison of various classification techniques using different data mining tools for diabetes diagnosis, J. Softw. Eng. Appl. ٦ (٣) (٢٠١٣) ٨٥–٩٧, http://dx.doi.org/١٠,٤٢٣٦/jsea.٢٠١٣,٦٣٠١٣.

[٣] S. Habibi, M. Ahmadi, S. Alizadeh, Type ٢ diabetes mellitus screening and risk factors using decision tree: results of data mining, Glob. J. Health Sci. ٧ (٥) (٢٠١٥) ٣٠٤–٣١٠, http://dx.doi.org/١٠,٥٥٣٩/gjhs.v٧n٥p٣٠٤.

[٤] Kaufman, Leonard; Rousseeuw, Peter J. (١٩٩٠-٠٣-٠٨), "Partitioning Around Medoids (Program PAM)", Wiley Series in Probability and Statistics, Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. ٦٨–١٢٥, doi:١٠,١٠٠٢/٩٧٨٠٤٧٠٣١٦٨٠١.ch٢

[٥] Schubert, Erich, and Peter J. Rousseeuw. "Fast and eager k-medoids clustering: O (k) runtime improvement of the PAM, CLARA, and CLARANS algorithms." *Information Systems* ١٠١ (٢٠٢١): ١٠١٨٠٤.