

الْخَلَقُ



دانشکده مهندسی برق و کامپیو تر

سامانه پیش‌بینی ابتلا به دیابت
(مبتنی بر الگوریتم‌های یادگیری ماشین)

نام و نام خانوادگی دانشجو: قوام الدین سلیمانی

استاد راهنما: سرکار خانم دکتر فرشته دهقانی

تقدیر و تشکر

تقدیم به همه کسانی که با پاکدلی، پرتوی دانایی و آگاهی را بر تاریکی‌های کاناپی می‌افشانند.

در این زمان که به لطف پروردگار مهربان، بار دیگر فرصت آموختن به من عطا شد، و موفق به اتمام این دوره شدم؛ بر خود لازم می‌دانم از تمام بزرگوارانی که مرا در این دوره یاری کردند، قدردانی کنم.

از استاد راهنمای محترم، سرکار خانم دکتر دهقانی که راهنمایی و بزرگواری ایشان در شرایط مختلف، همواره شامل حال من بوده و انجام این پروژه بدون همیاری ایشان ممکن نبود، بی‌نهایت سپاسگزارم و همواره برای ایشان، آرزوی موفقیت و سلامتی دارم.

از خانواده عزیزم که در این دوره، بزرگترین پشتیبان و یاری‌گر من بودند، تشکر می‌کنم و امیدوارم چون همیشه، در تندرنستی و خوشبختی، مایه دلگرمی ام باشند.

همچنین از سایر اساتید محترم گروه مهندسی کامپیوتر دانشگاه کاشان و دوستانم که توجهات و محبت ایشان در این دوره مایه پیشرفت من بوده نیز کمال تشکر را دارم و برای ایشان بهترین‌ها را از خداوند منان خواستارم.

فهرست مطالب

۱	تقدیر و تشکر
۲	فهرست تصاویر
۳	چهار
۴	چکیده
۵	۱ مقدمه
۶	۱-۱ تاریخچه
۷	۱-۱-۱ علل ابتلا
۷	۱-۱-۲ راهکارهای پیشگیری و استعدادسنجی
۸	۱-۲ انگیزه و اهداف انجام این پژوهش
۸	۱-۳ موارد انجام شده
۹	۲ ادبیات پژوهش
۱۰	۱-۲ مقدمه
۱۰	۲-۱ روش‌های داده‌کاوی
۱۰	۲-۲ طبقه‌بندی و خوشه‌بندی
۱۲	۲-۲-۱ بیش‌بازش و کم‌بازش، افزایش داده‌ها
۱۵	۲-۲-۲ معرفی الگوریتم‌های طبقه‌بندی
۱۵	۲-۳ رگرسیون لجستیک
۱۸	۲-۳-۱ درخت تصمیم
۲۲	۲-۳-۲ جنگل درختان تصادفی
۲۳	۲-۳-۳ AdaBoost
۲۴	۲-۳-۴ Naïve Bayes
۲۵	۴-۱ دادگان، پیش‌پردازش و مصورسازی داده‌ها
۲۵	۴-۲ دادگان
۲۶	۴-۳ پیش‌پردازش داده‌ها
۳۰	۴-۴ انتخاب ویژگی
۳۰	۴-۵ مصورسازی
۳۰	۴-۶ انواع نمودارها و داده‌های اجمالی
۳۲	۴-۷ اندازه‌گیری میزان خطای دقت
۳۴	۴-۸ ROC و AUC

۳۵	۵-۲ نتیجه‌گیری
۳۶	۳ کارهای پیشین
۳۶	۱-۳ مقدمه
۳۶	۲-۳ مقاله ۱
۳۸	۳-۳ مقاله ۲
۳۹	۴-۳ نتیجه‌گیری
۴۰	۴ روش‌ها و نتایج
۴۰	۱-۴ مقدمه
۴۰	۲-۴ روش پیشنهادی
۴۰	۱-۲-۴ پیاده‌سازی
۵۳	۲-۲-۴ Arduino
۵۷	۳-۲-۴ ابزار Node Red
۵۹	۴-۲-۴ Flask
۶۰	۳-۴ نتیجه‌گیری
۶۱	۵ جمع‌بندی و کارهای آتی
۶۱	۱-۵ جمع‌بندی
۶۱	۲-۵ کارهای آتی
۶۳	پیوست
۶۳	۱- کد استاندارد سازی متون فارسی آمیخته به عبارات انگلیسی
۶۴	واژه‌نامه
۶۶	مراجع

فهرست تصاویر

۱-۱	روندنمای پروژه	۹
۱-۲	مقایسه خوشبندی و طبقه‌بندی	۱۲
۱-۲	بیشبرازش و کم برازش	۱۲
۳-۲	روش K-Fold	۱۴
۴-۲	روش K-Fold تکرار شونده	۱۴
۵-۲	نمودار پراکندگی	۱۵
۶-۲	نمودار رگرسیون خطی چندگانه در فضا	۱۶
۷-۲	مقایسه رگرسیون لجستیک و خطی	۱۷
۸-۲	نمودار تابع لجستیک	۱۷
۹-۲	مدل Logist Regression	۱۸
۱۰-۲	مثال دادگان مورد استفاده در درخت تصمیم	۱۹
۱۱-۲	مثال درخت تصمیم	۱۹
۱۲-۲	انواع شاخه‌ها در درخت تصمیم	۲۰
۱۳-۲	نمونه‌ای از آنتروپی	۲۰
۱۴-۲	روش Bootstrap در جنگل تصادفی	۲۲
۱۵-۲	مدل جنگل درختان تصادفی	۲۳
۱۶-۲	الگوریتم AdaBoost: در اینجا مراحل ذکر شده به صورت متناوب تکرار می‌شود [۲۸]. [۲۸]	۲۴
۱۷-۲	نمونه‌ای از داده‌های گم شده	۲۶
۱۸-۲	تعداد داده‌های گم شده در این پروژه به ازای هر ستون	۲۶
۱۹-۲	متغیرهای رقمی	۲۷
۲۰-۲	نمونه‌ای از متغیرهای رقمی	۲۸
۲۱-۲	نمونه‌برداری مجدد	۲۹
۲۲-۲	Pairplot	۳۱
۲۳-۲	نقشه حرارتی	۳۲
۲۴-۲	ساختار ماتریس آشفتگی: پارامترهای مذکور در این بخش در این ماتریس قرار گرفته اند	۳۳
۲۵-۲	ساختار ماتریس آشفتگی برای الگوریتم جنگل درختان تصادفی	۳۴
۲۶-۲	نمای کلی نمودار ROC	۳۵
۱-۳	مدل SVM	۳۷
۲-۳	دقت‌های اندازه‌گیری در مقاله اول	۳۷

۳۸	منحنی ROC در مقاله اول	۳-۳
۳۸	دقت‌های اندازه‌گیری در مقاله دوم	۴-۳
۳۹	منحنی ROC در مقاله دوم	۵-۳
۴۱	جایگزینی مقادیر نامشخص با متoste‌های آماری	۱-۴
۴۲	نمودار مقایسه ابتلا به دیابت در تراشهای مختلف	۲-۴
۴۲	ماتریس Scatter	۳-۴
۴۳	نمودار عوامل موثر در ابتلا دیابت (۱)	۴-۴
۴۴	نمودار عوامل موثر در ابتلا دیابت (۲)	۵-۴
۴۵	نقشه حرارتی	۶-۴
۴۹	نمودار ACC	۷-۴
۴۹	نمودار AUC	۸-۴
۴۹	نمودار ROC برای الگوریتم رگرسیون لجستیک	۹-۴
۵۰	نمودار ROC برای الگوریتم AdaBoost	۱۰-۴
۵۰	نمودار ROC برای الگوریتم جنگل درختان تصادفی	۱۱-۴
۵۰	نمودار ROC برای الگوریتم Naïve Bayes	۱۲-۴
۵۱	نمودار ROC برای الگوریتم درخت تصمیم	۱۳-۴
۵۳	ساختار ماتریس آشفتگی برای الگوریتم رگرسیون لجستیک	۱۴-۴
۵۷	محیط توسعه Arduino در حال دریافت اعداد تولید شده توسط بورد	۱۵-۴
۵۸	صفحه داشبورد پیش‌بینی کاربر و بهروزرسانی دادگان	۱۶-۴
۵۹	طراحی بخش بهروزرسانی در NodeRed	۱۷-۴
۵۹	طراحی بخش پیش‌بینی در NodeRed	۱۸-۴
۶۰	بخشی از کد Flask	۱۹-۴

چکیده

دیابت یک بیماری مزمن است که فرد مبتلا، قند خون بالاتر از حد مجاز را داراست و این مسئله موجب عوارض و مشکلات جدی در سلامت وی (از جمله برخی نارسایی‌ها، سکته‌ها، آسیب و از کار افتادن اندام‌ها) می‌شود. لازمه ابتلا و بروز این بیماری، عوامل ژنتیکی و محیطی می‌باشد. لذا در صورت وجود احتمال ابتلا به این بیماری در افراد، می‌توان با تغییر سبک زندگی و کنترل‌های پزشکی، تا حدی از ابتلای به این بیماری در افراد مستعد و محتمل، جلوگیری کرد.

روش‌هایی که بتواند به ما کمک کند تا با دقت مناسبی بتوانیم ابتلای افراد مختلف به بیماری را در آینده را پیش‌بینی کنیم، بسیار حائز اهمیت هستند. یادگیری ماشین و داده‌کاوی با استفاده از داده‌های مختلف که در گذشته جمع آوری شده اند می‌توانند کمک شایانی به ما در این امر داشته باشند. پس از الگوریتم‌های مختلف یادگیری ماشین از جمله رگرسیون لجستیک، جنگل درختان تصادفی و ... استفاده کردیم تا دقت هر یک را اندازه‌گیری کنیم.

داده‌هایی که برای آموزش مدل‌هایمان استفاده کردہ‌ایم، از مجموعه دادگان بیماران آمریکایی^۱ است که در سال ۲۰۰۹ تا ۲۰۱۲ جمع آوری شده بودند. این داده‌ها را با استفاده از روش‌های داده‌کاوی گوناگون، پردازش و سپس مصورسازی کردیم و سپس با الگوریتم‌های مذکور و معیارهای ارزیابی مربوط به آن‌ها، سعی در شناسایی بهترین مدل پیش‌بینی‌کننده کردیم.

مدل جنگل درختان تصادفی حاوی بهترین نتایج نسبت به سایر مدل‌ها بود.

کلید واژه‌ها: داده کاوی، دیابت، درخت تصمیم، الگوریتم درخت تصادفی، قندخون، رگرسیون لجستیک، طبقه‌بندی، پیش‌بینی، یادگیری ماشین

^۱NHANES

فصل ۱

مقدمه

۱ - ۱ تاریخچه

دیابت چیست؟

دیابت، یک بیماری مزمن است و زمانی رخ می‌دهد که بدن انسولین کافی تولید نمی‌کند یا نمی‌تواند به طور موثر از انسولین تولید شده استفاده کند [۵]. انسولین هورمونی است که به تنظیم سطح قند خون کمک می‌کند. هنگامی که دیابت به درستی مدیریت نشود، می‌تواند منجر به عوارض جدی سلامتی مانند بیماری‌های قلبی، انواع سکته مغزی و قلبی، نارسایی کلیه، کوری و آسیب عصبی شود [۱۷]. آمار ابتلا و مرگ و میر بسیار بالایی از این بیماری در جهان وجود دارد و متاسفانه روز به روز این آمار افزایش می‌یابد.

طبق آمارها، از هر ۱۰ نفر که به دیابت مبتلا هستند، بیش از ۸ نفر آن‌ها از این مسئله آگاهی ندارند و عده زیادی از افراد هم به پیش‌دیابت مبتلا هستند [۶]. در پیش‌دیابت، سطح قند خون بالاتر از حد طبیعی است، اما به اندازه کافی برای تشخیص دیابت بالا نیست. پیش‌دیابت خطر ابتلا به دیابت، بیماری قلبی و سکته را افزایش می‌دهد [۵]. اگر پیش‌دیابت در افراد وجود داشته باشد، یک برنامه برای تغییر سبک زندگی، می‌تواند به افراد در جلوگیری از این بیماری کمک کند [۱۷].

این بیماری سه نوع دارد [۵]:

۱. دیابت نوع اول که معروف به دیابت جوانی است چون افراد با سن کمتر از ۳۰ سال معمولاً مبتلا می‌شوند. در این نوع به طور ساده می‌توانیم بگوییم میزان انسولین مورد نیاز که توسط پانکراس باقیستی ساخته شود و در خون وجود داشته باشد کافی نیست.
۲. دیابت نوع دوم که به بزرگسالی معروف است و در افراد میانسال و مسن رایج‌تر است در اثر عدم جذب انسولین موجود در خون توسط سلول‌ها می‌باشد.
۳. دیابت نوع سوم دیابت بارداری است که در خانم‌های باردار به طور موقت اتفاق می‌افتد.

۱-۱-۱ علل ابتلا

در ابتلا به این بیماری بنا به نوع آن و همچنین شرایط ژنتیکی و محیطی افراد مختلف، فاکتورهای متنوعی مطرح است [۵]:

- **فاکتورهای محیطی** مطابق تحقیقات و بررسی‌های انجام شده از سال‌ها پیش تا کنون، عوامل سبک زندگی چون رژیم غذایی نامناسب، عدم فعالیت بدنی و اضافه وزن (مخصوصاً میزان توده بدنی) می‌تواند خطر ابتلا به دیابت را افزایش دهد [۱۷]. همچنین وجود بیماری‌های زمینه‌ای مثلاً در پانکراس بین افراد می‌تواند در مبتلا شدن به این بیماری موثر باشد که بنا به تعریف دیابت نوع یک، این عامل مربوط به همین نوع می‌شود [۵].

- **عوامل ژنتیکی و غیرمحیطی** برخی از افراد استعداد ژنتیکی برای دیابت دارند، به این معنی که بدن آن‌ها بیشتر در معرض ابتلا به این بیماری است. عواملی مثل جنسیت، نژاد و شاخص‌هایی خونی مختلف که می‌تواند در اثر بیماری‌های خانوادگی و ارثی دیگری در افراد وجود داشته باشد. مثل برخی ویروس‌ها، وجود کلسترول، چربی و فشار خون و ... [۱۷] [۳] [۴].

۲-۱-۱ راهکارهای پیش‌گیری و استعدادسنجی

مطابق توصیه متخصصین اگر بتوانیم افرادی را که استعداد ابتلا به این بیماری را دارند، شناسایی کنیم و این افراد سبک زندگی و روش‌هایی خاصی را در پیش‌بگیرند، می‌توانند از ابتلا به این بیماری پیش‌گیری کنند [۳].

- **مبتلایان به پیش‌دیابت** مطابق توصیه پزشکان، در افرادی که به پیش‌دیابت مبتلا باشند یا سابقه این بیماری در خانواده آن‌ها وجود داشته باشد، به طور پیش‌فرض باید بر یک سبک زندگی سالم، اهتمام ورزند. در این راستا می‌توان به موارد ذیل اشاره کرد [۴]:

- حفظ رژیم غذایی غنی از فیبر مثل انواع میوه‌ها و سبزیجات و کاهش مصرف غذاهای شور، چرب و شیرین
- ورزش منظم
- استفاده از برخی داروها مطابق تجویز پزشک

- **سایر افراد جامعه** مطابق آمارها، سالانه بخش دیگری از افراد جامعه که از دسته قبلی سوا بوده اند، به بیماری دیابت مبتلا می‌شوند [۳]. در این جا با تحلیل برخی فاکتورهای سلامتی می‌توان پیش‌بینی کرد که آیا این افراد ممکن است با ادامه سبک زندگی کنونی، در آینده به این بیماری دچار شوند و آیا بهتر است با تغییر سبک زندگی خود از ابتلا به این بیماری جلوگیری کنند یا نه؟

در این زمینه تحقیقات آماری و بررسی‌های مختلفی انجام شده تا بتوانیم با اندازه‌گیری برخی فاکتورهای کمی و کیفی در افراد، مسئله استعداد در ابتلا به این بیماری را در آن‌ها بررسی کنیم. چالشی که در این زمینه وجود دارد این است که بسیاری از داده‌های غیرخطی و غیراستاندارد پزشکی با ارتباطات و ساختارهای پیچیده وجود دارند که این بررسی‌ها را دشوار می‌سازد [۶].

۱-۲ انجیزه و اهداف انجام این پژوهش

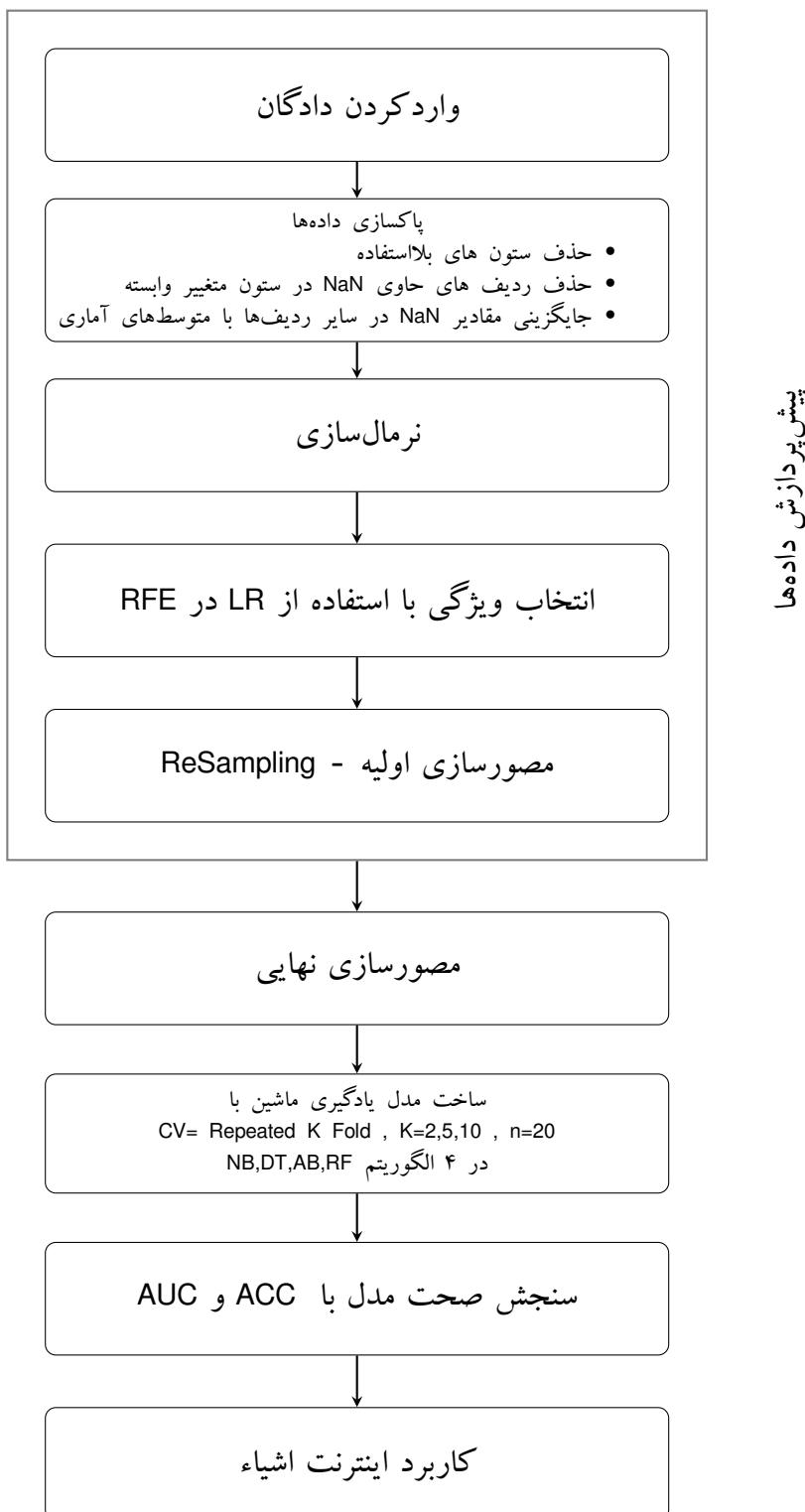
در راستای همه موارد مطرح شده در بخش‌های قبلی، بر آن شدیم تا تحقیق کنیم با توجه به امکانات و امروزی و دسترسی به مقالات و منابع گوناگون و همچنین توسعه ابزارهای مبتنی بر یادگیری ماشین و هوش مصنوعی، در صدد یافتن بهترین راهکارها برای نجات جان انسان‌های بیشتر با درنظرگیری مناسب‌ترین الگوریتم‌ها باشیم.

اگر بتوانیم افرادی را که احتمال ابتدا به دیابت در آینده برای آنان زیاد است را شناسایی کنیم می‌توانیم با ارائه برنامه‌های پزشکی مناسب از ابتلای افراد به بیماری مذکور جلوگیری کنیم.

در نهایت نیز جهت ادامه پروژه، یک سیستم پیش‌بینی آنلاین با کمک اینترنت اشیاء و دو چارچوب Flask و NodeRed طراحی و معرفی شد که افراد گوناگون بتوانند از آن بهره ببرند.

۱-۳ موارد انجام شده

در ابتدا مقالات مختلفی را مطالعه کردم و درمورد الگوریتم‌هایی که مورد بررسی قرار دادم از جمله رگرسیون لجستیک، درخت تصمیم، جنگل درختان تصادفی و بیز ساده، اطلاعات زیادی کسب کردم. پس از یافتن مجموعه دادگان نمونه که مربوط به اطلاعات بیماران آمریکایی در سال‌های ۲۰۰۹ تا ۲۰۱۲ بوده است، با به کارگیری کتابخانه‌های مختلف پایتون از جمله numpy، Pandas، scikit learn، seaborn و ... عملیات‌های گوناگونی بر روی داده‌ها انجام شد. از جمله: تمیز کردن دادگان، استانداردسازی داده‌ها، تصویر سازی و در نهایت مدل‌سازی و دستیابی به نتایج نهایی که در نهایت الگوریتم جنگل درختان تصادفی با دقت ۹۶٪ بهترین عملکرد را بین باقی الگوریتم‌ها در این پیش‌بینی، شناخته شد. در شکل ۱-۱ روند نمای پروژه نشان داده شده است.



شکل ۱-۱: روند نمای پروژه

فصل ۲

ادبیات پژوهش

۱-۲ مقدمه

روش‌های نظارت شده‌ای مانند طبقه‌بندی و تخمین تلاش می‌کنند تا رابطه‌ای میان صفات خاصه ورودی (که گاه متغیرهای مستقل نامیده می‌شوند) را با یک یا چند صفت خاصه هدف (که گاه متغیر وابسته نامیده می‌شود) کشف کنند. در نهایت این رابطه با یک ساختار به عنوان مدل نمایش داده می‌شود [۱].

به بیان دیگر، در یادگیری تحت نظارت همان‌طور که از نام آن پیداست، یک ناظر به عنوان یاددهنده در این نوع الگوریتم یادگیری ماشین حضور خواهد داشت. ما مدل خود را با داده‌های برچسب‌گذاری شده مناسب آموزش می‌دهیم. الگوریتم‌های یادگیری نظارت شده سعی می‌کنند روابط ووابستگی‌هایی بین متغیرها ایجاد کنند که به ترتیب «ویژگی‌ها» و «برچسب‌ها» نامیده می‌شوند. سپس الگوریتم‌ها از داده‌ها، (با استفاده از روابط بین ویژگی‌ها) یاد می‌گیرند و خروجی را پیش‌بینی می‌کنند [۱۴].

اما در یادگیری بدون نظارت هیچ ناظر یا یاد دهنده‌ای وجود نخواهد داشت، بنابراین هیچ آموزشی یا آموزشی به ماشین ارائه نخواهد شد. یادگیری بدون نظارت با داده‌های بدون برچسب سروکار دارد که در آن ما نمی‌توانیم روابط و وابستگی‌ها را در داده‌ها اندازه‌گیری کنیم. در این مدل، مدل‌های ما سعی می‌کنند داده‌های مرتب نشده را بر اساس الگوها و شباهت‌ها در داده‌ها به صورت خوش‌ای گروه‌بندی کنند [۱۴].

۲-۲ روش‌های داده‌کاوی

۱-۲-۲ طبقه‌بندی و خوش‌بندی

در این بخش به طور واضح‌تر درمورد تفاوت الگوریتم‌های بانظارت (طبقه‌بندی‌کننده و پیش‌بینی) و بدون نظارت (خوش‌بندی) در مدل‌های رایج توضیح می‌دهیم.

• طبقه‌بندی

در این مدل، گیرنده پیشنهاد می‌تواند پاسخ دهد یا پاسخ ندهد. متقاضی وام می‌تواند به موقع

بازپرداخت کند، دیر بازپرداخت کند یا اعلام و رشکستگی کند. تراکنش کارت اعتباری می‌تواند عادی یا تقلیبی باشد. بسته‌ای از داده‌هایی که در یک شبکه حرکت می‌کنند می‌تواند عادی یا تهدیدکننده باشد. یک اتوبوس در سیستم حمل و نقل می‌تواند در دسترس باشد یا در دسترس نباشد. قربانی یک بیماری می‌تواند بهبود یابد، یا خیر [۱۱].

یک کار رایج در داده کاوی، بررسی داده‌هایی است که در آن طبقه‌بندی ناشناخته است یا در آینده رخ خواهد داد، با هدف پیش‌بینی اینکه طبقه‌بندی چیست یا چه خواهد بود. داده‌های مشابهی که طبقه‌بندی شناخته است، برای توسعه قوانین استفاده می‌شوند که بعداً روی داده‌های دارای طبقه‌بندی ناشناخته اعمال شوند [۱۱].

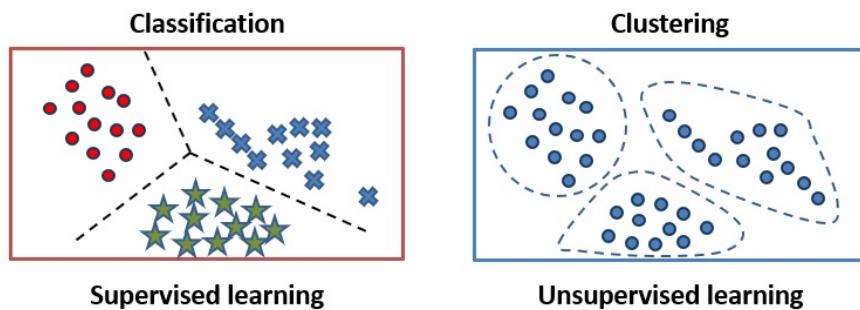
به بیان دیگر، پیش‌بینی مشابه طبقه‌بندی است، با این تفاوت که ما سعی می‌کنیم ارزش یک متغیر عددی (مثلاً مقدار خرید) را به جای یک طبقه (مثلاً خریدار یا غیرخریدار) پیش‌بینی کنیم. البته در طبقه‌بندی سعی داریم یک طبقه را پیش‌بینی کنیم، اما گاهی در ادبیات داده کاوی، از اصطلاحات تخمین و رگرسیون برای اشاره به پیش‌بینی مقدار یک متغیر پیوسته استفاده می‌شود، و پیش‌بینی ممکن است هم برای داده‌های پیوسته و هم برای داده‌های طبقه‌ای استفاده شود [۱۱].

ضمانت در فرایند ساخت مدل طبقه‌بندی، داده‌های اولیه به دو بخش تقسیم می‌شوند که در ادامه توضیح داده خواهد شد؛ داده‌های آموزشی برای مدل را ساخته و داده‌های اعتبارسنجی که زیر مجموعه‌ای از داده‌های اصلی هستند و مستقل از داده‌های آموزشی اند، عملکرد مدل را برایمان ارزیابی می‌کنند [۱۱].

• **خوشه‌بندی** تجزیه و تحلیل خوشه‌ای، برای تشکیل گروه‌ها یا خوشه‌هایی از رکوردهای مشابه بر اساس چندین اندازه‌گیری انجام شده بر روی آن‌ها انجام می‌شود. ایده کلیدی خوشه‌بندی به این صورت است که خوشه‌ها را به روش‌هایی توصیف کنیم که برای اهداف تحلیل مفید باشد. این ایده، در بسیاری از زمینه‌ها از جمله نجوم، باستان‌شناسی، پزشکی، شیمی، آموزش، روانشناسی، زبان‌شناسی و جامعه‌شناسی به کار گرفته شده است. برای مثال، زیست‌شناسان از طبقات اصلی و طبقات فرعی برای سازماندهی گونه‌ها استفاده گسترده‌ای کرده اند [۱۱]. بدیهی است که ممکن است مانند این چه تعداد دسته یا خوشه در پایان این فرایند ایجاد می‌شود.

به زبان دیگر، می‌توانیم بگوییم زمانی که پیش‌فرض خاصی درمورد کلاس‌هایمان نداریم و صرفاً می‌خواهیم بینیم داده‌هایمان در چه گروه‌هایی قرار می‌گیرند از خوشه‌بندی استفاده می‌کنیم [۱۱] (مثلاً چندین متن داریم و می‌خواهیم متن‌های مشابه در یک دسته قرار گیرند) اما اگر بخواهیم هر رکورد از داده‌ها را الزاماً در یک طبقه یا کلاس خاص قرار دهیم از طبقه‌بندی استفاده می‌شود که روش باظارت است [۱۱] (مانند همین پروژه).

از دید دیگر می‌توانیم بگوییم هرگاه حسایت ما برای تشخیص‌های گوناگون زیاد باشد، (یعنی لزوماً برچسبی بر داده‌ای بزنیم که اهمیت زیادی برای ما داشته باشد مثل تشخیص دیابت در افراد) در اینجا از مدل‌های طبقه‌بندی استفاده می‌کنیم و اگر تنها بخواهیم داده‌هایمان را در دسته‌های گوناگون قرار دهیم و نام کلاس‌های دسته‌بندی ما مشخص نباشد، می‌توانیم خوشه‌بندی را اعمال کنیم. (مثلاً یک فروشگاه اینترنتی که به گروه‌هایی از خریداران مختلف، پیشنهادهای گوناگونی در ازای خریدهای قبلی آن‌ها می‌دهد).



شکل ۱-۲: مقایسه خوشبندی و طبقه‌بندی

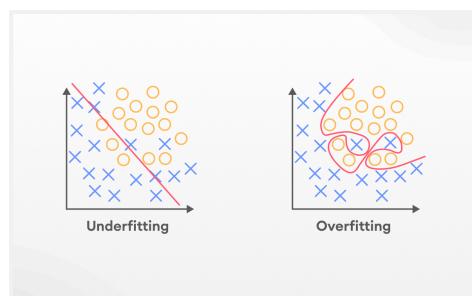
۲-۲-۲ بیش‌برازش و کم‌برازش، افزایش داده‌ها

اگر خیلی ساده بخواهیم در مورد این دو مفهوم توضیح دهیم، می‌توان گفت در یک مدل یادگیری ماشین، کم‌برازش در خروجی زمانی اتفاق می‌افتد که مدل خیلی ساده است و نمی‌تواند رابطه و الگوی بین داده‌ها را با دقت مشخص کند و از این جهت در پیش‌بینی و سنجش ناتوان خواهد بود، در حالی که بیش‌برازش زمانی اتفاق می‌افتد که مدل بیش از حد پیچیده باشد و داده‌های آموزشی را به خاطر بسپارد. بنابر این در مقابل داده‌های جدید تعمیم و انطباق پیدا نمی‌کند. [۲۰]

راه حل جلوگیری از این مشکل، تقسیم داده‌ها و اعتبارسنجی متقابل است [۲۰] که در ادامه توضیح داده می‌شود.

یک مثال خوب که می‌توان برای این مسئله بیان کرد، نمونه لیوان است. اگر فرض کنیم مدلی که ساخته‌ایم، داشتن دسته را برای یک شیء به نام "لیوان" را ضروری بداند، پس به نظر می‌رسد شرط اضافه‌ای برای تشخیص لیوان بودن یا نبودن اشیاء منظور کرده است؛ و در این حالت می‌گوییم مدل ما دچار بیش‌برازش شده است.

بر عکس در حالتی که فرض کنیم مدل ما اهمیتی برای داشتن ارتفاع نسبت به سطح را در نظر نگیرد، ممکن است یک بشقاب را به عنوان یک لیوان در نظر بگیرد و در این حالت می‌گوییم مدل ما دچار کم‌برازش شده است.



شکل ۲-۲: بیش‌برازش و کم‌برازش

بخش کردن داده‌ها یا افزایش^۱ برای ارزیابی مدل به طور مؤثر استفاده می‌شود. این تکنیک، برای ارزیابی عملکرد مدل یادگیری ماشین به این صورت عمل می‌کند شما یک مجموعه داده داده شده را می‌گیرید و آن را به دو یا سه زیر مجموعه تقسیم می‌کنید. [۲۱]

- داده‌های آموزشی یا (Train) :

مجموعه‌ای از داده‌های مورد استفاده برای ساخت و یادگیری جهت تناسب پارامترها با مدل یادگیری ماشین و ایجاد روابط در مدل مد نظرمان است.

- داده‌های اعتبارسنجی (استفاده از این مورد اختیاری است). :

مجموعه‌ای از داده‌ها برای ارائه یک ارزیابی بی‌طرفانه از یک مدل برآذش شده در مجموعه داده آموزشی در حین تنظیم روابط درون مدل استفاده می‌شود.

- داده‌های سنجش (Test) :

مجموعه‌ای از داده‌ها برای ارائه یک ارزیابی بی‌طرفانه از یک مدل نهایی برآذش شده در مجموعه داده آموزشی استفاده می‌شود. در واقع این قسمت تضمین می‌کند که مدل بر اساس داده‌های دیده نشده ارزیابی می‌شود. در واقع هدف از استفاده از داده‌های آزمایشی، تعیین میزان تعمیم مدل به داده‌های جدید و دیده نشده است.

برای دستیابی به نتایج دقیق، بسیار مهم است که مطمئن شویم داده‌ها به صورت تصادفی انتخاب شده اند و شامل طیف متنوعی از نمونه‌ها هستند. این مسئله برای به حداقل رساندن سوگیری و بهبود قابلیت‌های تعمیم مدل کمک می‌کند [۲۱].

با این حال، با تقسیم داده‌های موجود به سه مجموعه، تعداد نمونه‌هایی را که می‌توان برای یادگیری مدل استفاده کرد، به شدت کاهش می‌دهیم که راه حل آن استفاده از اعتبارسنجی متقابل است که در آن نیازی داده‌های اعتبارسنجی نیست [۲۲] و در ادامه توضیح داده می‌شود.

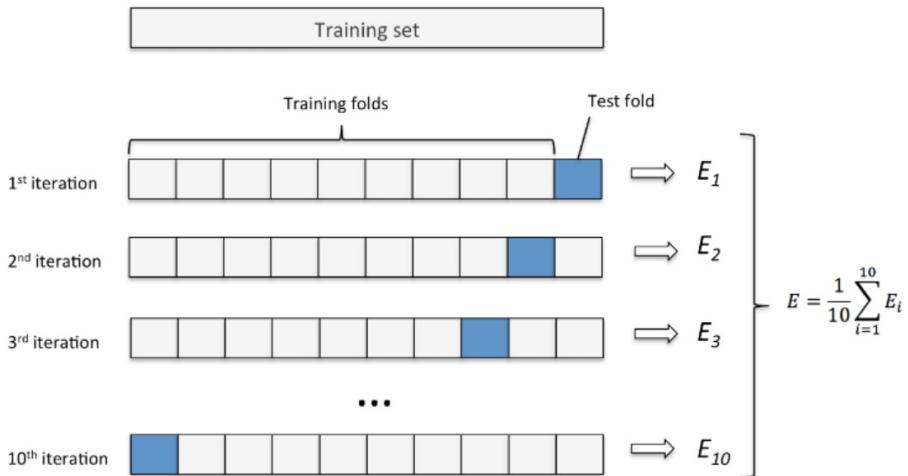
اعتبارسنجی متقابل^۲

برای جلوگیری از برآذش بیش از حد، می‌توانیم از تکنیک‌هایی استفاده کنیم که داده‌های آموزشی و سنجش ما را به قسمت‌های بیشتری تقسیم کنند و آن‌ها را به طرق مختلف در مقابل هم مورد ارزیابی قرار دهد تا مدل ما دقیق‌تر بیشتری پیدا کند.

این روش، شامل تقسیم داده‌های آموزشی به k مجموعه‌های کوچکتر و آموزش یک مدل با استفاده از $k-1$ عدد از این مجموعه‌ها است، در حالی که از مجموعه باقی مانده، به عنوان مجموعه آزمایشی، برای محاسبه میزان عملکرد مدل استفاده می‌شود. این فرآیند k بار تکرار می‌شود و هر مجموعه یک بار به عنوان مجموعه تست عمل می‌کند. سپس میانگین معیارهای عملکرد به دست آمده (که در فصل‌های آینده ذکر می‌شود) در هر تکرار، به عنوان معیار عملکرد کلی در نظر گرفته می‌شود. این روش از نظر محاسباتی گران است اما از هدر رفتن بیش از حد داده‌ها جلوگیری می‌کند [۲۳].

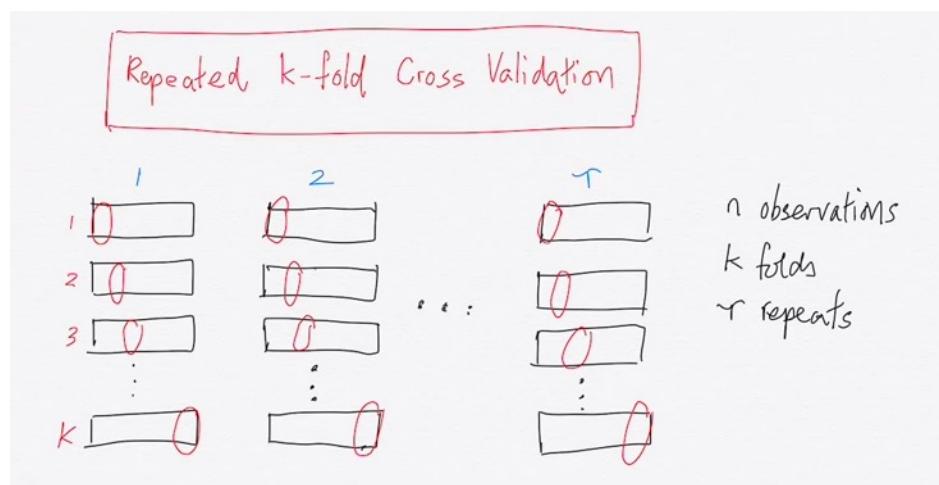
¹Data Splitting

²Cross validation



شکل ۲-۳: روش K-Fold

روش دیگری که ذکر شد، اعتبارسنجی K-Fold مکرر است که فرآیند اعتبارسنجی K-Fold را n بار با تقسیم‌های مختلف هر بار تکرار می‌کند. این روش زمانی مفید است که چندین بار لازم است فرآیند اعتبارسنجی تکرار شود [۲۳]. در واقع این روش علاوه بر تقسیم داده‌ها به K-Fold، فرآیند را چندین بار تکرار می‌کند و قبل از هر تکرار به طور مستقل داده‌ها را به هم می‌زند. این روش با میانگین گرفتن مکرر اعتماد به مدل را افزایش می‌دهد [۱۵].



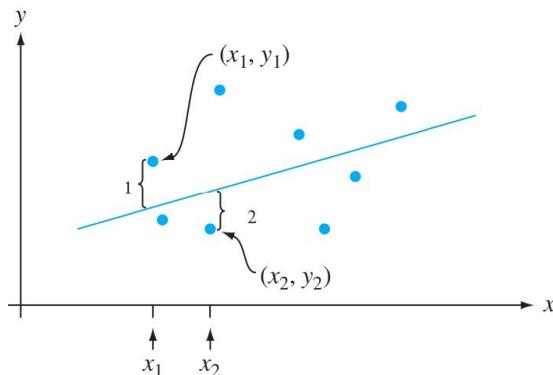
شکل ۴-۲: روش K-Fold تکرار شونده

۳-۲ معرفی الگوریتم‌های طبقه‌بندی

۱-۳-۲ رگرسیون لجستیک^۳

خوب است در ابتدا مروری بر رگرسیون خطی^۴ داشته باشیم تا در ادامه بتوانیم رگرسیون لجستیک را بهتر درک کنیم.
رگرسیون خطی:

در بسیاری از بررسی‌های آماری، لازم است یک متغیر وابسته را از روی یک یا چند متغیر مستقل پیش‌بینی کنیم که اصطلاحاً به آن رگرسیون یا برگشت می‌گوییم [۲]. برای مثلاً میزان ساعت مطالعه یک متغیر مستقل است و نمره اخذ شده در درسی متغیری وابسته است و بین این دو رابطه وجود دارد. سپس نمونه‌ای از جمعیت را در نظر گرفته و در آن مقدارهای X_1 تا X_n در متغیر مستقل خود مقابله مقادیر نظیر در متغیر وابسته از Y_1 تا Y_n قرار می‌دهیم [۲]. سپس آن‌ها را مثل یک نمودار در صفحه مختصات به یکدیگر متصل کرده که به آن نمودار پراکندگی گوییم [۲].



شکل ۲-۵: نمودار پراکندگی

حالا می‌توان خطی را در این صفحه مختصات درنظر گرفت که تا حد زیادی منطبق بر نقاط باشد که در واقع یک نمودار پیش‌بینی‌کننده Y بر مبنای X است که به آن معادله رگرسیون Y بر روی X گویند [۲]. حالا رابطه این نقاط و منحنی را با $\mu_{Y|x} = E(Y|x) = \alpha + \beta x$ مشخص می‌کنیم و α و β پارامترهایی هستند که باید مقدار دهی شوند تا خط بر نقاط منطبق باشد [۲]. مسئله‌ای که در اینجا مطرح است این است که ممکن است خط ما بر نقاط مختلف منطبق نشود. لذا اینجا باید حالت بهینه‌ای را در نظر گرفت حداقل مقدار خطأ (یا بهتر بگوییم اختلاف) در مجموع داشته باشیم که روش حداقل مربعات برای یافتن میزان بهینه α و β که دارای بیشترین انطباق و کمترین خطأ باشند برای ما کمک‌کننده است [۲]. بر اساس همین روش، با معادلات زیر به مقادیر بهینه α و β دست می‌یابیم [۲].

$$\alpha = \bar{y} - \beta \bar{x} \quad (1-2)$$

$$\beta = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2} \quad (2-2)$$

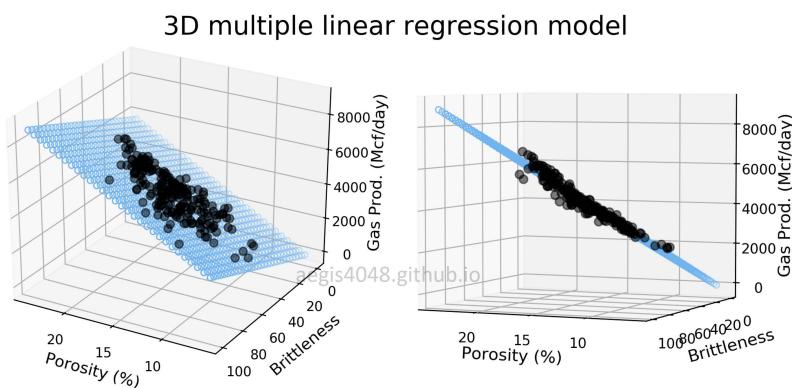
^۳Logistic Regression

^۴Linear Regression

که در روابط فوق \bar{x} و \bar{y} میانگین y و x هستند. با تعمیم این روابط و اصول بیان شده می‌توان حالتی را در نظر گرفت که چندین متغیر مستقل داریم. (مثلاً ۲ تا) که این مدل به رگرسیون خطی چندگانه معروف است و در آنچه نمودار ما حالت فضایی پیدا خواهد کرد و با رابطه زیر می‌توانیم آن را بیان کنیم.

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \epsilon \quad i = 1, \dots, n \quad (3-2)$$

که در این رابطه، p ابعاد ما می‌باشد [۳۲].



شکل ۲-۶: نمودار رگرسیون خطی چندگانه در فضا

در کل می‌توانیم بگوییم روش‌های رگرسیون زمانی مناسب است که مقادیر مستقل در مجموعه داده‌ها به کلاس‌هایمان (به بیان دیگر طبقه‌بندی‌ها) وابستگی داشته باشند. ضریب همبستگی خطی که با رابطه ۴-۲ می‌شود میزان این وابستگی را برای ما نشان می‌دهد.

$$p(X, Y) = \text{corr}(X, Y) = \frac{\text{Cov}(x, y)}{(\text{Var}(x)\text{Var}(y))^{\frac{1}{2}}} \quad (4-2)$$

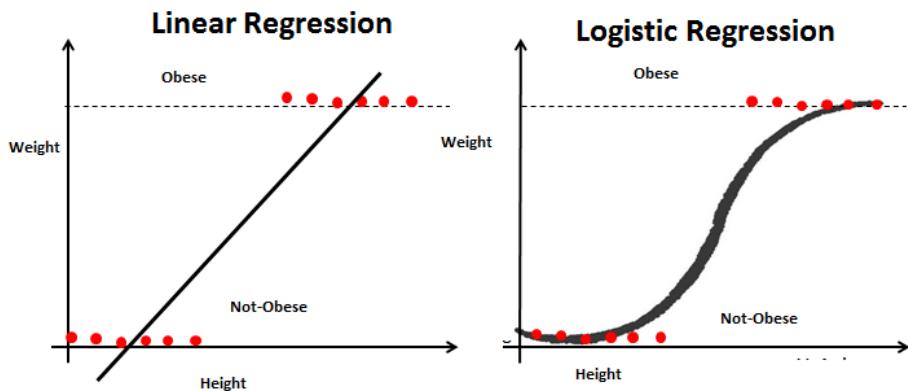
صورت کسر کواریانس X و y است و مخرج واریانس X و واریانس y می‌باشد. حاصل مقداری است از ۱ - تا ۱ که میزان وابستگی مستقیم یا معکوس را نشان می‌دهد [۱] و در صورت نبود میزان وابستگی مقدار ۰ است [۲].

رگرسیون لجستیک:

حالا که با مفاهیم ابتدایی رگرسیون آشنایی پیدا کردیم به بیان نوع دیگری از آن به نام رگرسیون لجستیک می‌پردازیم.

در این مدل به جای اینکه مقدار عددی برای متغیر وابسته تعریف شود، بر اساس احتمال متغیرهای رقیقی را برای پیشگویی در نظر داریم [۱] که در ادامه این مسئله را بیشتر توضیح می‌دهیم. مثلاً کار ما، فاکتور BMI در افراد یک متغیر مستقل است و بر ابتلا به دیابت در افراد موثر بوده؛ همچنین در اینجا متغیر وابسته ما، همان دیابت گرفتن یا نگرفتن فرد می‌باشد که با ۰ و ۱ آن را در نظر می‌گیریم. پس این مدل برای مواردی استفاده می‌شود که حالت کلاس‌بندی برای متغیرهایمان داریم [۸] [۹]. ضمناً در حالت کلاس‌بندی، نمی‌توانیم حالت ۰ یا ۱ را به عنوان اعدادی در حالت رگرسیون خطی منظور کنیم؛

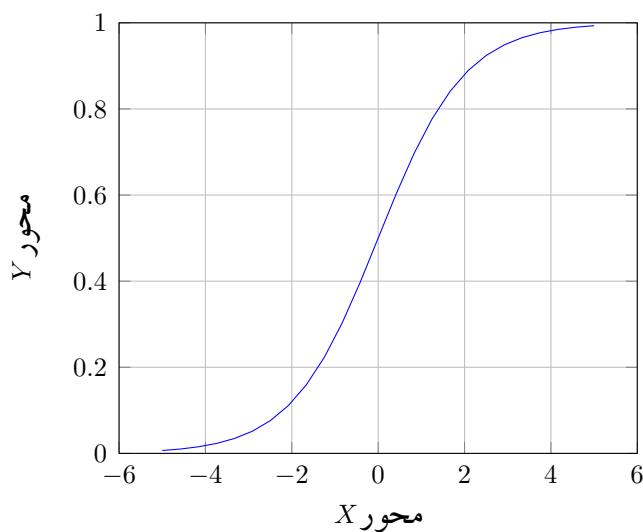
چرا که \circ و \circ را به عنوان مقادیر عدد در نظر گرفته می‌شود و امکان نمایش کلاس‌های گوناگون در در یک نمودار خطی مشخص وجود ندارد [۹]. اگر به شکل ۷-۲ دقت کنید، متوجه می‌شوید در حالت رگرسیون خطی برخی از نمونه‌ها در کلاس مربوطه قرار نگرفته‌اند.



شکل ۷-۲: مقایسه رگرسیون لجستیک و خطی

علت این امر مشخص است. زیرا زمانی که کلاس‌های گوناگون داریم، ساختار استدلالی که در الگوریتم رگرسیون خطی مطرح است نمی‌تواند طبقه‌بندی صحیحی برای ما انجام دهد. نهایتاً چاره این است که از یک نمودار منحنی شکل برای فشرده کردن نتایج بین \circ تا \circ استفاده کنیم [۹]: رابطه ۵-۲ و شکل ۸-۲

$$\text{logistic}(x) = \frac{1}{1 + \exp^{-x}} \quad (5-2)$$



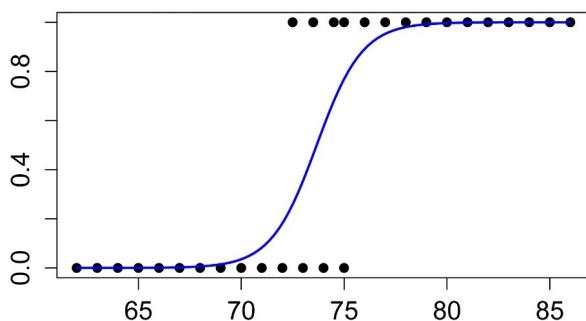
شکل ۸-۲: نمودار تابع لجستیک

سپس اگر سمت راست رابطه ۳-۲ را در جای X در معادله ۵-۲ (که آن را تابع سیگماوار^۵ می‌نامند) قرار دهیم، فقط مقادیر ° و ۱ را به ما می‌دهد.

حال در این مدل، چون قرار است هر نمونه به یک کلاس تعلق یابد، بر اساس روابط مطرح شده در این بخش و تعمیم خواص آماری (واریانس)، می‌دانیم وزن ویژگی هایمان عاملی اثر گذار تعیین مرز جداسازی در نمودار ما خواهد بود. لذا نهایتاً به رابطه زیر می‌رسیم که بر اساس احتمال قرارگیری هر نمونه در هر کلاس برای ما کلاس‌بندی را انجام می‌دهد [۸][۹]:

$$g(x) = \ln\left(\frac{p(x)}{1 - P(x)}\right) = \frac{\frac{e^{b_0 + b_1 x}}{1 + e^{b_0 + b_1 x}}}{1 - \frac{e^{b_0 + b_1 x}}{1 + e^{b_0 + b_1 x}}} = \ln(e^{b_0 + b_1 x}) = b_0 + b_1 x \quad (6-2)$$

در رابطه فوق، کاری که انجام می‌شود به این صورت است که احتمال قرارگیری یک نمونه در یک کلاس نسبت به حالت دیگر سنجیده می‌شود و در صورتی که مثلاً با احتمال بالای ۵۰ درصد در طبقه ۱ قرار می‌گیرد، این کار صورت می‌پذیرد و در غیر این صورت به کلاس ° متعلق است. (می‌دانیم که احتمال دقیقاً ۵۰ درصد روی نقطه (۰،۰) قرار دارد). [۸][۹] به بیان دیگر می‌توانیم بگوییم، Logit پاسخ ۷، ترکیب خطی پیش‌بینی‌کننده‌ها (یا همان X) است. [۶]



شکل ۹-۲: مدل Logist Regression

۲-۳-۲ درخت تصمیم

در یادگیری ماشین، درخت تصمیم یک مدل پیش‌بینی‌کننده است که مجموعه‌ای از تصمیم‌ها و پیامدهای احتمالی را در ساختاری درخت‌مانند نشان می‌دهد و درواقع تصمیم‌گیری انسانی را تقليد می‌کند، جایی که تصمیمات منجر به تصمیمات بیشتر یا نتایج نهایی می‌شود [۱۲].

یک درخت تصمیم، می‌تواند ساختاری مشابه شکل ۱۱-۲ با توجه به یک دادگان نمونه که در شکل ۲-۲ داشته باشد [۱۱]. در این نمونه، چیزی که برآورد شده این است که یک مشتری که به فروشگاه کامپیوتر مراجعه می‌کند، آیا ممکن است در نهایت یک کامپیوتر از آن فروشگاه بخرد؟ لذا در این ساختار درختی، عناصر مختلف از آن ملاحظه می‌شود مثل ریشه^۶، برگ‌ها^۷ و گره‌های میانی و همچین شاخه‌ها^۸. مثلاً در اینجا، ابتدا سن بررسی می‌شود اگر میانسال باشد خرید را انجام می‌دهد. اما اگر

^۵Sigmoid/ Logistic Function

^۶Root node

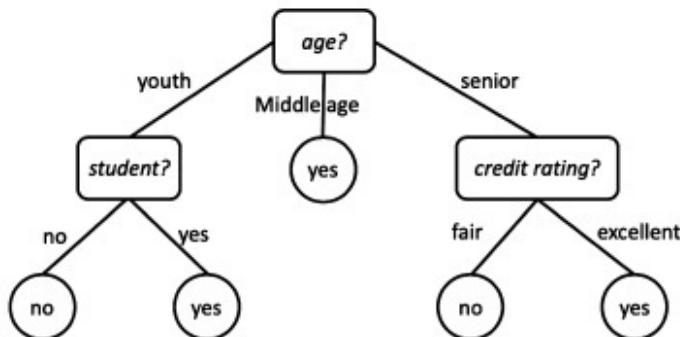
^۷Leaf node

^۸Branch

جوان باشد، تنها در صورتی که دانشجو باشد خرید را انجام می‌دهد. همچنین در صورت اینکه سن بالا داشته باشد، در صورت داشتن اعتبار مالی مناسب این خرید انجام می‌شود.

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

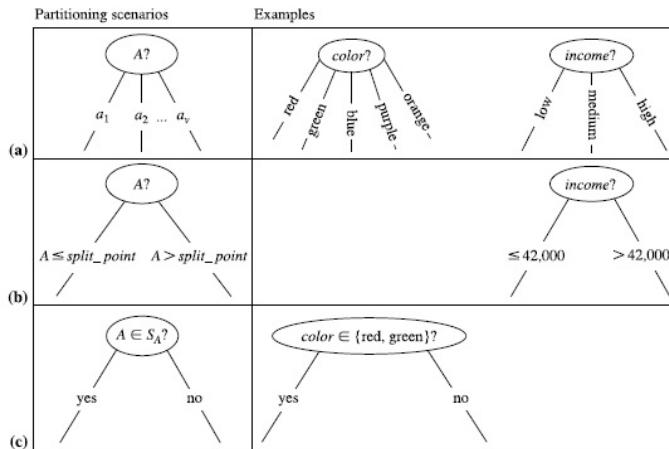
شکل ۲-۱۰: مثال دادگان مورد استفاده در درخت تصمیم



شکل ۲-۱۱: مثال درخت تصمیم

همچنین انواعی از حالت‌های شاخه‌ها را می‌توانیم داشته باشیم، که به ۳ صورت می‌توانند باشند.
(شکل ۱۲-۲)

۱. ویژگی‌های گستته چندگانه مثل رنگ یا گروه‌های خونی
۲. ویژگی‌های مقداری پیوسته مثل درآمد یا سن (به صورت ویژگی شرطی که از مقداری کمتر یا بیشتر باشند).
۳. ویژگی‌های باینری مثل داشتن فعالیت فیزیکی یا نداشتن آن



شکل ۱۲-۲: انواع شاخه‌ها در درخت تصمیم

انتخاب بهترین ویژگی یکی از مسائلی که اینجا مطرح می‌شود، این است که در گره ریشه، و گره‌های بعدی، کدام ویژگی قرار داده شود که بهترین پیش‌بینی برای ما انجام شود؟ برای این کار چندین روش وجود دارد که دو روش از آن‌ها در اینجا معرفی می‌کنیم:

- آنتروپی اطلاعات^۹ در شکل ۱۳-۲ به سادگی مفهوم آنتروپی به تصویر کشیده شده است (که در سلطی که نوع خاصی از توب‌ها به طور قابل توجهی بیشتر از انواع دیگر باشد، آنتروپی کمتر است.). [۲۷] لذا اگر یک نمونه از سلطی که دارای آنتروپی کمتری است برداریم، می‌توانیم با احتمال بالایی نوع آن توب را به درستی پیش‌بینی کنیم.



شکل ۱۳-۲: نمونه‌ای از آنتروپی

همین ایده، در انتخاب بهترین ویژگی در گره‌های درخت تصمیم می‌تواند مورد استفاده قرار گیرد. یعنی ما باید ویژگی‌هایی را برای هر گره انتخاب کنیم که کمترین آنتروپی را داشته باشد. در واقع آنتروپی معیاری از ناچالصی یا تصادفی بودن در یک مجموعه داده است. از رابطه ۷-۲، برای محاسبه آنتروپی ویژگی‌ها استفاده می‌کنیم [۱۲].

$$H(X) = - \sum_{i=1}^n p_x \log_2 p_x \quad (7-2)$$

در این رابطه :

- آنتروپی $H(X)$ آنتروپی مجموعه داده X است.

^۹Information entropy (Information gain ratio)

- p احتمال وقوع هر کلاس ممکن در مجموعه داده است.
- نهایتاً مجموع بر روی تمام مقادیر کلاس ممکن گرفته می‌شود.

• ضریب Gini

فرمول Gini نیز برای محاسبه ناخالصی یا ناهمگنی یک گره استفاده می‌شود و احتمال طبقه‌بندی نادرست یک عنصر تصادفی انتخاب شده در یک مجموعه داده را اندازه‌گیری می‌کند [۱۲].

این شاخص از ۰ تا ۱ متغیر است. مقدار ۰ نشان می‌دهد که گره خالص است، به این معنی که همه عناصر در گره متعلق به یک کلاس هستند. بر عکس، مقدار ۱ نشان دهنده ناخالصی کامل است، جایی که عناصر از کلاس‌های مختلف به طور مساوی در گره توزیع می‌شوند [۱۲].

شاخص جینی با جمع کردن مجدد احتمالات هر کلاس در گره و کم کردن آن از یک محاسبه می‌شود. می‌توان آن را با فرمول زیر نشان داد:

$$Gini = 1 - \sum_j (p_j)^2 \quad (۸-۲)$$

این روش نیز مشابه روش قبلی (که در الگوریتم CART) نیز استفاده می‌شود، p_j نشان دهنده احتمال انتخاب تصادفی یک عنصر متعلق به کلاس j است [۱۲].

الگوریتم‌های گوناگونی برای ساخت درخت تصمیم استفاده می‌شود مثل (... ID3, C4.5, CART, ...). که در کتابخانه مورد استفاده ما یعنی scikit-learn از الگوریتم CART برای ساخت درخت تصمیم استفاده می‌شود [۲۶] (این الگوریتم مانند ID3 و C4.5 حریصانه است) و در اینجا عملکرد آن را شرح می‌دهیم.

الگوریتم CART ^۱: الگوریتم درخت طبقه‌بندی و رگرسیون، یک الگوریتم درخت تصمیم محظوظ است و با بخش‌بندی بازگشتی مجموعه داده بر اساس مقادیر ویژگی‌های مختلف، برای رسیدن به یک تصمیم یا پیش‌بینی نهایی عمل می‌کند. این یک الگوریتم حریصانه است (یعنی درخت را از بالا به پایین بر اساس بهترین ویژگی در گره‌ها، می‌سازد [۱۱]). مرحله کلی این الگوریتم‌ها تقریباً مشابه همدیگر هستند و به این ترتیب است [۱۱].

۱. ابتدا همه ویژگی‌ها را در ریشه قرار بده.
۲. بهترین آن‌ها را بر اساس یک ویژگی مثل آنتروپی یا ضریب Gini در آن گره انتخاب کن.
۳. شاخه‌ها را ایجاد کن.

۴. این روند را در هر گره ادامه بده تا به نتایج (برگ‌های نهایی) برسید.

همچنین لازم به ذکر است که درختان تصمیم‌گیری تمایل دارند که داده‌های آموزشی را بیش از حد برآراش دهند، بسیار خاص می‌شوند و قابلیت تعمیم را از دست می‌دهند. هر س درخت، فرآیند کاهش اندازه درخت تصمیم برای جلوگیری از برآراش بیش از حد است که شاخه‌های غیر ضروری را برای افزایش سادگی و دقت با روش‌های مختلف کوتاه می‌کند [۱۱].

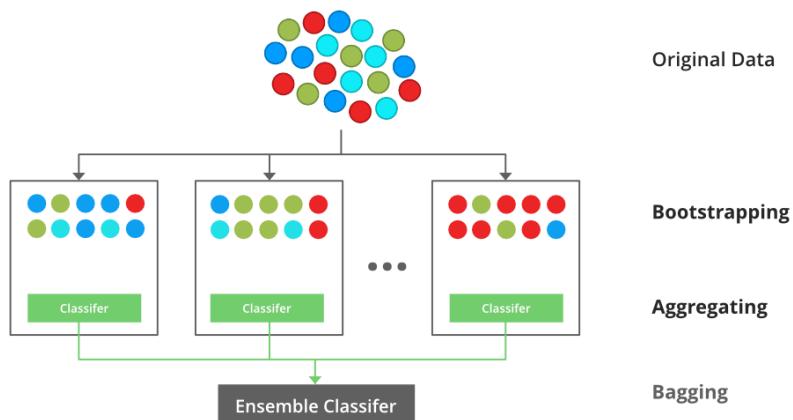
- پیش‌حرس: پیش روی ساخت گره را در عمق خاصی را محدود می‌کنیم.

^۱ Classification and Regression Trees

- پس حرس: برخی از گره‌ها با روش‌هایی انتخاب و به برگ تبدیل می‌شوند. سپس مجدداً معیارهای ارزیابی خاصی را در نظر می‌گیریم که مشخص کند این حرس دقت مدل ما را افزایش می‌دهد یا نه. مشخصاً این روش کنتر است.

۳-۳-۲ جنگل درختان تصادفی

این الگوریتم از ساختار درختان تصادفی، مجموعه‌ای از درختان تصمیم را با انتخاب تصادفی چندین زیر مجموعه، از مجموعه مرجع داده‌های آموزشی ایجاد (که این را روش Bootstrap می‌نامیم) و مجدداً انتخاب زیر مجموعه‌ای از برخی ویژگی‌های دادگان را در هر تقسیم ایجاد می‌کند. هر درخت تصمیم به طور مستقل بر روی نمونه‌های مختلف Bootstrap و زیر مجموعه‌های ویژگی آموزش داده می‌شود (انتخاب ویژگی مطابق مباحث قبلی در درخت تصمیم یعنی با ضریب جینی یا آنتروپی است [۱۲]) و با درختان دیگر متفاوت است. هنگام انجام یک پیش‌بینی، تمام درختان جنگل تصادفی به نتیجه نهایی رأی می‌دهند و طبقه‌بندی‌ای که اکثریت آرا را داشته باشد، به عنوان کلاس یا طبقه‌بندی پیش‌بینی شده نهایی، انتخاب می‌شود [۱۲].



شکل ۲-۱۴: روش Bootstrap در جنگل تصادفی

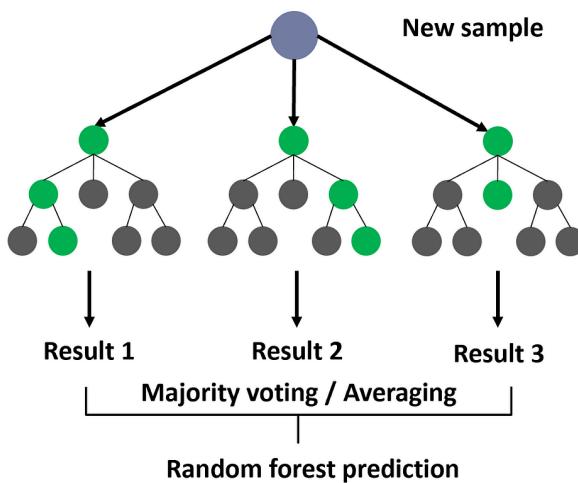
با ترکیب پیش‌بینی‌های درخت‌های تصمیم چندگانه، الگوریتم جنگل تصادفی به کاهش بیش‌برازش و بهبود عملکرد تعمیم کمک می‌کند. همچنین پایداری بیشتر در ساخت مدل در اثر به وجود آمدن مواردی چون نقاط دورافتاده و داده‌های گم شده را فراهم می‌کند [۱۲].

مثالی که می‌توان برای یک نمونه از این مدل زد، این است در یک جنگل تصادفی ۳ درخت حکم می‌کنند که فرد به دیابت مبتلا خواهد شد و ۱ درخت حکم می‌کند که این فرد به دیابت مبتلا نخواهد شد. لذا اکثریت آراء بر مبتلا شدن فرد اتفاق نظر دارند. پس نتیجه آن پیش‌بینی، مبتلا شدن فرد است.

حال فرض کنیم میخواهیم یک مدل با این روش بسازیم و ورودی‌هایی را با آن پیش‌بینی کنیم. پس این مراحل را به ترتیب انجام می‌دهیم [۶]:

۱. مجموعه داده را به دو قسمت آموزشی و آزمایشی تقسیم کنید. از مجموعه آموزشی داده شده، یک مجموعه داده جدید با استفاده از روش Bootstrap ایجاد کنید.

۲. بر اساس نتایج مرحله ۱ یک درخت تصمیم بسازید.
۳. مرحله ۱ و ۲ را تکرار کنید و درختان زیادی را تولید کنید که از یک جنگل تشکیل شده اند.
۴. از هر درخت در جنگل برای رأی دادن به ورودی (یک ردیف داده که می خواهیم نتیجه آن را پیش‌بینی کنیم) استفاده کنید.
۵. میانگین آرا برای هر کلاس را محاسبه کنید. کلاسی که بیشترین رأی را می دهد متعلق به برچسب طبقه‌بندی برای ورودی داده شده است.
۶. در نهایت دقت طبقه‌بندی کننده مبتنی بر جنگل تصادفی را محاسبه کنید.



شکل ۲-۱۵: مدل جنگل درختان تصادفی

AdaBoost ۴-۳-۲

این الگوریتم یک فرض ساده در نظر دارد و آن یادگیری گروهی است [۱۳]. فرض کنید کسانی که در مجموعه داده‌های آموزشی دچار اشتباه در تشخیص ابتلا به بیماری دیابت شدند، از مجموعه بقیه داده‌های آموزشی جدا می‌شوند و در یک طبقه‌بندی جدید مجدداً مورد ارزیابی قرار می‌گیرند؛ همان‌طور که در واقعیت ممکن است هرپزشکی معیارهای مختلفی را برای تشخیص بیماری مراجعه کننده‌اش داشته باشد و هر پزشکی نمی‌تواند همه افراد را درست تشخیص دهد پس اگر یک تیم پزشکی داشته باشیم نظرات جمیعی می‌توانند بیماران بیشتری را به درستی شناسایی کنند [۱۱].

این روند جداسازی داده‌های آموزشی و امتیاز دهی در تشخیص توسط طبقه‌بندی کننده‌های مختلف درون درختان تصمیم ضعیف، مکرراً تکرار می‌شود تا درنهایت طبقه‌بندی‌های متعددی داشته باشیم که هر کدام بر اساس میزان سرآمدی در تست‌های مختلف، امتیازات مختلفی دریافت کنند [۱۳]. بعد از ساخته شدن مدل، حالا می‌توانیم به نسبت امتیاز هر طبقه‌بندی کننده داده‌های دریافتی را به صورت شانسی بین هر کدام تقسیم کنیم تا پیش‌بینی صورت گیرد و این مسئله موجب تقویت سنجش می‌گردد. به همین دلیل به آن الگوریتم Adaptive Boosting یعنی تقویت کننده تطبیقی گویند [۱۱] [۱۳].

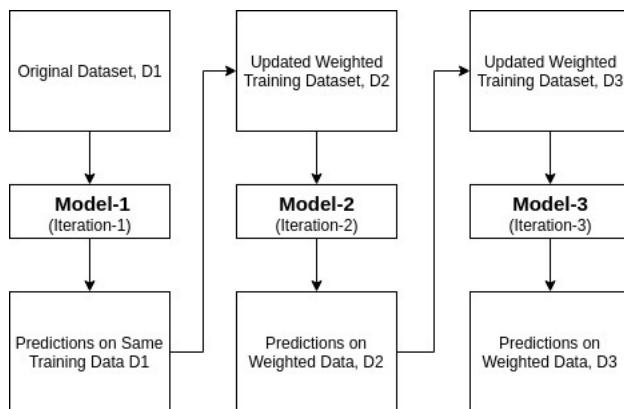
اگر از دید ریاضی الگوریتم را بررسی کنیم، برای محاسبه میزان خطای مدل، M_i وزن هر یک از

تاپل‌های D_i را که M_i اشتباه طبقه‌بندی کرد، جمع می‌کنیم.

$$\text{error}(M_i) = \sum_{j=1}^d w_i \times \text{err}(X_j) \quad (9-2)$$

حالا اگر طبقه‌بندی‌کننده X_j اشتباه کند، میزان ارور برای آن ۱ اندازه‌گیری می‌شود و در غیر این صورت ۰ است. اگر یک طبقه‌بندی‌کننده آن قدر ضعیف باشد که خطایش از ۰.۵ بیشتر شود آن را دیگر در نظر نمی‌گیریم [۱۱]. سپس یک مجموعه داده جدید به نام D_i تولید می‌کنیم و از آن یک M_i جدید استخراج می‌کنیم و دوباره این روند را ادامه می‌دهیم. هر چه میزان خطای طبقه‌بندی‌کننده کمتر باشد، دقیق‌تر است و بنابراین، وزن آن برای انتخاب شدن باید بیشتر باشد. از رابطه ۹-۲ برای محاسبه وزن هر طبقه‌بندی استفاده می‌شود [۱۱].

$$\log\left(\frac{1 - \text{error}(M_i)}{\text{error}(M_i)}\right) \quad (10-2)$$



شکل ۱۶-۲: الگوریتم AdaBoost: در اینجا مراحل ذکر شده به صورت متناوب تکرار می‌شود [۲۸].

Naïve Bayes ۵-۳-۲

طبقه‌بندی‌کننده بیز ساده بر روی مفهوم احتمال شرطی کار می‌کند و به این به این سؤال پاسخ می‌دهد که احتمال اینکه یک تاپل داده از تعلق یک مجموعه داده به یک کلاس خاص، چقدر است [۱۲].

$$P(H|x) = \frac{P(x|H)P(H)}{P(x)} \quad (11-2)$$

- احتمال وقوع رویداد H با توجه به اینکه رخداد X به وقوع پیوسته است. (احتمال پسین) (مثلاً احتمال این که فردی با ویژگی خاصی به دیابت مبتلا شود.)
- احتمال وقوع رویداد X با توجه به اینکه رویداد H به وقوع پیوسته است. (برعکس مورد قبل است، یعنی چند درصد از افراد دارای دیابت، ویژگی مورد نظر ما را مثلاً چاقی را دارا می‌باشد.)

^{۱۱}Likelihood

- $P(X)$: احتمال رویداد X (درصد افرادی که ویژگی خاصی را در دادگان ما دارند، مثلاً در کل فلان درصد از افراد چاق هستند).

- $P(H)$: احتمال رویداد H (احتمال پیشین) (مثلاً می‌دانیم احتمال مبتلا شدن افراد چاق به دیابت درصد خاصی دارد).

حال ما مثلاً برای مبتلا شدن به دیابت یا مبتلا نشدن به دیابت، این احتمال را برای افراد محاسبه و احتمال وقوع هر کدام از کلاس‌ها که بیشتر شد (رابطه ۱۲-۲)، آن داده متعلق به همان طبقه است [۱۲]. (رابطه ۱۳-۲)

$$P(C_i|x) = \frac{P(x|C_i)P(C_i)}{P(x)} \quad (12-2)$$

$$\text{argmax} P(C_i|x) = \text{argmax} P(x|C_i)P(C_i) \quad (13-2)$$

در نهایت، زمانی که این احتمال را برای تک تک ویژگی‌ها برای هر کلاس به صورت جداگانه محاسبه کردیم، نتیجه را برای هر ویژگی در همان کلاس ضرب می‌کنیم تا در کل بفهمیم برای این ویژگی‌ها، وقوع کدام کلاس متحتمل‌تر است [۱۲]: (رابطه ۱۴-۲)

نکته‌ای که قابل ذکر است این است که چون ممکن است احتمال موردی صفر شود و در عملیات ضرب نتیجه کل حاصل شده صفر شود، به هر احتمال عدد ۱ اضافه می‌شود که این روش را Laplacian correction می‌نامند [۲۵].

$$P(c|x) = P(x_1|x) \times P(x_2|x) \times P(x_3|x) \times \dots \times P(x_n|x) \times P(c) \quad (14-2)$$

یکی از مزایایی که این روش دارد این است که روش سریعی است، به این دلیل که پس از افزودن داده‌های جدید، نیاز نیست مدل را مجدداً باز سازی کنیم [۱۲].

۴-۲ دادگان، پیش‌پردازش و مصورسازی داده‌ها

۱-۴-۲ دادگان

دادگانی^{۱۲} که از آن استفاده کردیم، برگرفته از یک برنامه مطالعاتی به نام NHANES بوده که جهت بررسی سلامتی کودکان و بزرگسالان از سوی CDC^{۱۳} تدوین شده است.

برنامه NHANES در اوایل دهه ۱۹۶۰ آغاز شد و به صورت مجموعه‌ای از نظرسنجی‌ها با تمرکز بر گروه‌های مختلف جمعیتی یا موضوعات بهداشتی انجام شده است. در سال ۱۹۹۹، این نظرسنجی به

^{۱۲}Dataset

^{۱۳}US Centers for Disease Control and Prevention

یک برنامه مستمر تبدیل شد که تمرکز در حال تغییری بر روی انواع اندازه گیری‌های سلامت و تغذیه برای رفع نیازهای نوظهور دارد [۱۸].

صاحب NHANES شامل سوالات جمعیت شناختی، اجتماعی-اقتصادی، رژیم غذایی و سلامتی است. جزء معاینه شامل اندازه گیری‌های پزشکی، دندانی و فیزیولوژیکی و همچنین تست‌های آزمایشگاهی است که توسط پرسنل پزشکی بسیار آموزش دیده انجام می‌شود [۱۸].

۲-۴-۲ پیش‌پردازش داده‌ها

پیش‌پردازش داده‌ها چیست؟

داده‌هایی که جمع آوری می‌کنیم، انواع مختلفی دارند. مانند رشته‌ها، انواع اعداد، مقادیر نامشخص و ... همچنین منابع داده‌های ما نیز ممکن است طبقه‌بندی‌های مختلفی از داده‌ها را با فرمتهای گوناگون ارائه کنند که کار را برای سیستم یادگیری ماشین ما دشوار می‌سازد.

داده‌های گم‌شده چیست؟

وقتی دادگانمان را مورد بررسی قرار می‌دهیم، در برخی از سلوول‌ها، جای برخی مقادیر خالی هستند (در پروسه جمع آوری داده‌ها این اطلاعات به هر دلیلی ثبت نشده‌اند) یا در اثر عملیات‌های مربوط به شناسایی داده‌های پرت، برخی از داده‌ها را حذف می‌کنیم و جای آن‌ها خالی می‌ماند. در این موقع باید با استفاده از تدابیری، داده‌های گم‌شده را با مقادیری جایگزین کنیم یا کلا آن رکوردها را حذف کنیم تا مدل‌های دقیق‌تری داشته باشیم. (شکل ۱۷-۲)

	Gender	Age	Race1	Education	MaritalStatus	Work	Weight	Height	BMI	BPSysAve	BPDiaAve	DirectChol
0	male	34	White	High School	Married	NotWorking	87.4	164.7	32.22	113.0	85.0	1.29
1	male	34	White	High School	Married	NotWorking	87.4	164.7	32.22	113.0	85.0	1.29
2	male	34	White	High School	Married	NotWorking	87.4	164.7	32.22	113.0	85.0	1.29
3	male	4	Other	Nan	Nan	Nan	17.0	105.4	15.30	Nan	Nan	Nan
4	female	49	White	Some College	LivePartner	NotWorking	86.7	168.4	30.57	112.0	75.0	1.16

شکل ۱۷-۲: نمونه‌ای از داده‌های گم‌شده

```
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype  
--- 
 0   AgeDecade      9859 non-null   object 
 1   Race1          9859 non-null   object 
 2   Poverty         9859 non-null   float64
 3   Work            9859 non-null   object 
 4   DirectChol     9859 non-null   float64
 5   TotalChol      9859 non-null   float64
 6   TotalBili      9859 non-null   float64
 7   Depressed       9859 non-null   object 
 8   SleepTrouble   9859 non-null   object 
 9   SmokeNow        9859 non-null   object 
 10  HardDrugs       9859 non-null   object 
 11  SameSex         9859 non-null   object 
 12  SameOrientation 9859 non-null   object 
 13  PregnantNow    9859 non-null   object 
 14  Diabetes        9859 non-null   object 

dtypes: float64(3), object(12)
```

شکل ۱۸-۲: تعداد داده‌های گم‌شده در این پروژه به ازای هر ستون

راه حل‌های داده‌های گم‌شده

• حذف ردیف‌های حامل داده‌های گم‌شده

یکی از راهکارهایی که در هنگام کار با داده‌های گم شده انجام می‌شود، حذف کل رکوردهایی است که دارای این مقادیر هستند. این مسئله یک بدی دارد و بدی آن این است که داده‌های کمتری برای آموزش مدلمان در اختیار خواهیم داشت و مدل ما ضعیف‌تر خواهد بود. اما اگر به هر دلیلی نتوانیم این مقادیر را با مقادیری دارای تقریب خوب پر کنیم، چاره دیگری نداریم.

• جایگزینی با متغیرهای آماری

یکی از راهکارهای دیگر برای مدیریت داده‌های گم شده، جایگزین کردن مقادیر گم شده با جایگزینی با متغیرهای آماری است. برخی از ستون‌ها را که زیاد اهمیت نداشته باشند می‌توان با مقادیر میانگین پر کنیم. مثلاً اگر یک دادگان داشته باشیم که حاوی اطلاعات مشتریان یک بانک باشد که بخواهیم از آن برای وام دادن به آن‌ها استفاده کنیم، می‌توانیم در ستونی که مربوط به میزان حقوق ماهیانه هر فرد می‌باشد، در صورت مشاهده مقادیر NaN، آن‌ها را با میانگین حقوق مشتریان جایگزین کرد. این راهکار، صرفاً به ما امکان می‌دهد تا تحلیل را ادامه دهیم و اطلاعات موجود در این رکورد را برای متغیرهای دیگر از دست ندهیم [۱۲].

استاندارد سازی داده‌ها

برای ادامه مراحل، لازم است کارهای بیشتری را روی داده‌های خود انجام دهیم. از جمله ایجاد متغیرهای ساختگی (رقمی) و مقیاس‌بندی متغیرهای ساختگی (رقمی)

در مدل‌های مختلف یادگیری ماشین به عنوان مثال در همین رگرسیون لجستیک، لازم است تا تنها متغیرهای عددی را به عنوان ورودی جهت مدل سازی ارائه کنیم و این مدل نمی‌تواند متغیرهای رشته‌ای را تشخیص دهد. پس از یک راهکار استفاده می‌کنیم. مقادیر درون ستون‌های رشته‌ای را با یک مقدار عددی از ۱ تا ۷ جایگزین می‌کنیم. ستون دیگر تقسیم می‌کنیم (شکل ۱۹-۲) و مقادیر ستون‌های جدید را با بله/خیر (۱و۰) پر می‌کنیم [۱۲]. به عنوان مثال در ستون مربوط به وضعیت تا هل فرد، به هر کدام از وضعیت‌های گوناگون عددی اختصاص می‌دهیم. (شکل ۲۰-۲)

```
df3 = dataset_new.copy()
# These columns must be converted
df3 = pd.get_dummies(df3,columns = ['Gender','Race1','Education','MaritalStatus','Work','PhysActive','Diabetes'],
print(df3.columns)
dataset_new=df3.copy()

Python
Index(['Age', 'Weight', 'Height', 'BMI', 'BPSysAve', 'BPDiaAve', 'DirectChol',
       'TotChol', 'Gender_male', 'Race1_Hispanic', 'Race1_Mexican',
       'Race1_Other', 'Race1_White', 'Education_9 - 11th Grade',
       'Education_College Grad', 'Education_High School',
       'Education_Some College', 'MaritalStatus_LivePartner',
       'MaritalStatus_Married', 'MaritalStatus_NeverMarried',
       'MaritalStatus_Separated', 'MaritalStatus_Widowed', 'Work_NotWorking',
       'Work_Working', 'PhysActive_Yes', 'Diabetes_Yes'],
      dtype='object')
```

شکل ۱۹-۲: متغیرهای رقمی

:Decade	Race1	Poverty	Work	DirectChol	TotChol	HealthGen	Depressed	SleepTrouble	SmokeNow	HardDrugs	SameSex	SexOrientation	PregnantNow	Dial
3	4	1.36	1	1.290000	3.490000	2	2	1	0	1	0	1	0	
3	4	1.36	1	1.290000	3.490000	2	2	1	0	1	0	1	0	
3	4	1.36	1	1.290000	3.490000	2	2	1	0	1	0	1	0	
0	3	1.07	2	1.365029	4.878875	2	1	0	0	0	0	1	0	
4	4	1.91	1	1.160000	6.700000	2	2	1	1	1	1	1	0	

شکل ۲-۲: نمونه‌ای از متغیرهای رقمی

مقیاس‌بندی در نرمال‌سازی

برخی از الگوریتم‌ها نیاز دارند که داده‌ها قبل از پیاده سازی مؤثر الگوریتم نرمال‌سازی شوند. برای نرمال‌سازی یک متغیر، میانگین را از هر مقدار کم می‌کنیم و سپس بر انحراف استاندارد تقسیم می‌کنیم. این عملیات گاهی اوقات استانداردسازی نیز نامیده می‌شود.^[۱۲]

علت این امر این است که واحد اندازه‌گیری مورد استفاده می‌تواند بر تجزیه و تحلیل داده‌ها تأثیر بگذارد. به عنوان مثال، تغییر واحدهای اندازه‌گیری از متر به اینچ برای قد، یا از کیلوگرم به پوند برای وزن، ممکن است به نتایج بسیار متفاوتی منجر شود.^[۱۲] به طور کلی، درنظرگرفتن یک ویژگی در واحدهای کوچک‌تر، به محدوده بزرگتری برای آن منجر می‌شود و بر وزن آن ویژگی اثرگذار است. برای جلوگیری از به وجود آمدن این مشکل، داده‌ها باید نرمال یا استاندارد شوند. یعنی داده‌ها در محدوده‌های کوچک‌تر یا نرمال‌تر قرار بگیرند. این مسئله باعث می‌شود تا به همه ویژگی‌ها وزن یکسانی داده شود. ^[۱۲] لذا قبل از انجام این عملیات، بررسی داده‌ها و ساختارشان دارای اهمیت است. ^[۲۴]

برای روش‌های مبتنی بر فاصله، نرمال‌سازی به جلوگیری از برتری ویژگی‌هایی با دامنه‌های اولیه بزرگ (مانند درآمد) کمک می‌کند تا ویژگی‌هایی با دامنه‌های اولیه کوچک‌تر (مانند ویژگی‌های باينری) سبقت بگیرد. ^[۱۲]

روش مورد استفاده در این پروژه از بین روش‌های مختلف StandardScaler است.

از رابطه زیر مقدار جدید ویژگی مقیاس‌بندی شده به دست می‌آید ^[۲۴]:

$$x_{scaled} = (x - mean(x)) \div StandardDeviation(x) \quad (15-2)$$

در واقع استانداردسازی فرآیندی است که متغیر را روی ۰ (میانگین صفر) قرار می‌دهد و واریانس را به ۱ (واریانس واحد) استاندارد می‌کند. پس از مقیاس‌بندی استاندارد، انحراف معیار نیز ۱ (انحراف معیار استاندارد) خواهد بود. در واقع برای استاندارد کردن ویژگی‌ها، میانگین را از هر مشاهده کم کردیم و سپس نتیجه را بر انحراف استاندارد تقسیم کردیم. خاصیت دیگر این کار، سرعت بخشیدن به روند ساخته شدن مدل‌های یادگیری ماست.^[۲۴]

نتیجه این تبدیل، z-score نامیده می‌شود که نشان می‌دهد یک مشاهده معین از میانگین چند انحراف استاندارد انحراف دارد. از این رو، استانداردسازی، نرمال‌سازی امتیاز z نیز نامیده می‌شود.^[۲۴]

در ایجاد بسیاری از مدل‌های یادگیری ماشین انجام این فرایند ضروری است. ^[۱۲]

نمونه‌برداری مجدد

نمونهبرداری مجدد^{۱۴} به تکنیک‌هایی اشاره دارد که توزیع مجموعه داده را برای مشکل نامتعادل بودن کلاس خاصی، انجام می‌دهد. دو روش رایج نمونه‌گیری مجدد عبارتند از نمونه‌گیری کم و نمونهبرداری بیش از حد [۳۱] [۱۲].

نمونه‌گیری کم^{۱۵} شامل کاهش اندازه طبقه اکثریت با حذف برخی از نمونه‌ها است. این کار می‌تواند به متعادل شدن توزیع آماری نمونه‌ها کمک کند، اما ممکن است منجر به از دست رفتن اطلاعات مهم شود. [۳۱].

از طرف دیگر، نمونهبرداری بیش از حد^{۱۶} شامل روش‌های افزایش اندازه طبقه اقلیت با افزودن نمونه‌های مصنوعی یا تکراری است. هدف این تکنیک، ارائه نمونه‌های آموزشی بیشتر برای طبقه اقلیت و بهبود توانایی مدل در یادگیری الگوهای داده‌ها است [۳۱].

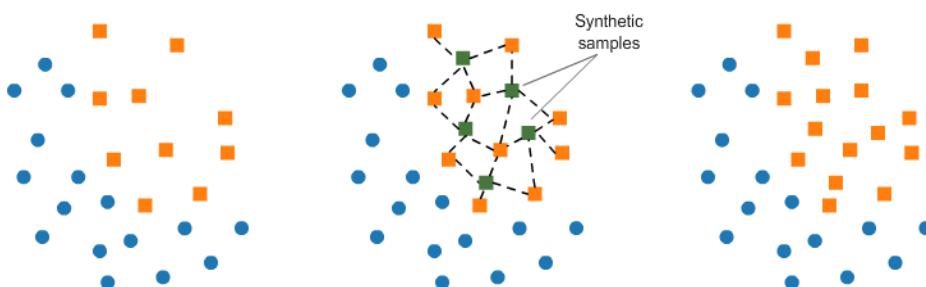
SMOTE^{۱۷} یک روش محبوب برای نمونهبرداری بیش از حد از کلاس اقلیت است که برای تولید نمونه‌های مصنوعی مابین نمونه‌های کلاس اقلیت موجود طراحی شده است [۳۱].

نحوه عملکرد SMOTE [۳۱]:

۱. برای هر نمونه کلاس اقلیت، k همسایه نزدیک خودش (معمولاً $k=5$) بر اساس متريک فاصله انتخاب می‌شود.

۲. یک نمونه مصنوعی جدید با انتخاب تصادفی یکی از k نزدیکترین همسایه‌هایش تولید می‌شود. تولید این نمونه، در امتداد خطی که نمونه اصلی و همسایه انتخابی را به هم متصل می‌کند، انجام می‌شود.

۳. این فرآيند برای تولید تعداد مورد نظر نمونه مصنوعی برای کلاس اقلیت تکرار می‌شود.



شکل ۲۱-۲: نمونهبرداری مجدد

SMOTE با تولید نمونه‌های مصنوعی، به طور موثر فضای ویژگی‌های کلاس اقلیت را گسترش می‌دهد. این کار به مدل کمک می‌کند تا الگوهای مبنا را بیاموزد و عملکرد خود را در طبقه اقلیت بهبود بخشد [۳۱].

^{۱۴}Resampling

^{۱۵}Underfitting

^{۱۶}Overfitting

^{۱۷}Synthetic Minority Oversampling Technique

۳-۴-۲ انتخاب ویژگی

[۱۹]

۴-۴-۲ مصورسازی

تعریف: به طور ساده می‌توانیم بگوییم زمانی که داده‌هایمان را به صورت انواع نمودارها، نقشه‌ها و شکل‌های مختلف بصری دریاباوریم تا نتیجه‌گیری و تحلیل آن‌ها توسط مغز جهت شناسایی الگوها و نقاط پرت در داده آسان‌تر شود، این کار انجام می‌گیرد [۲۲].

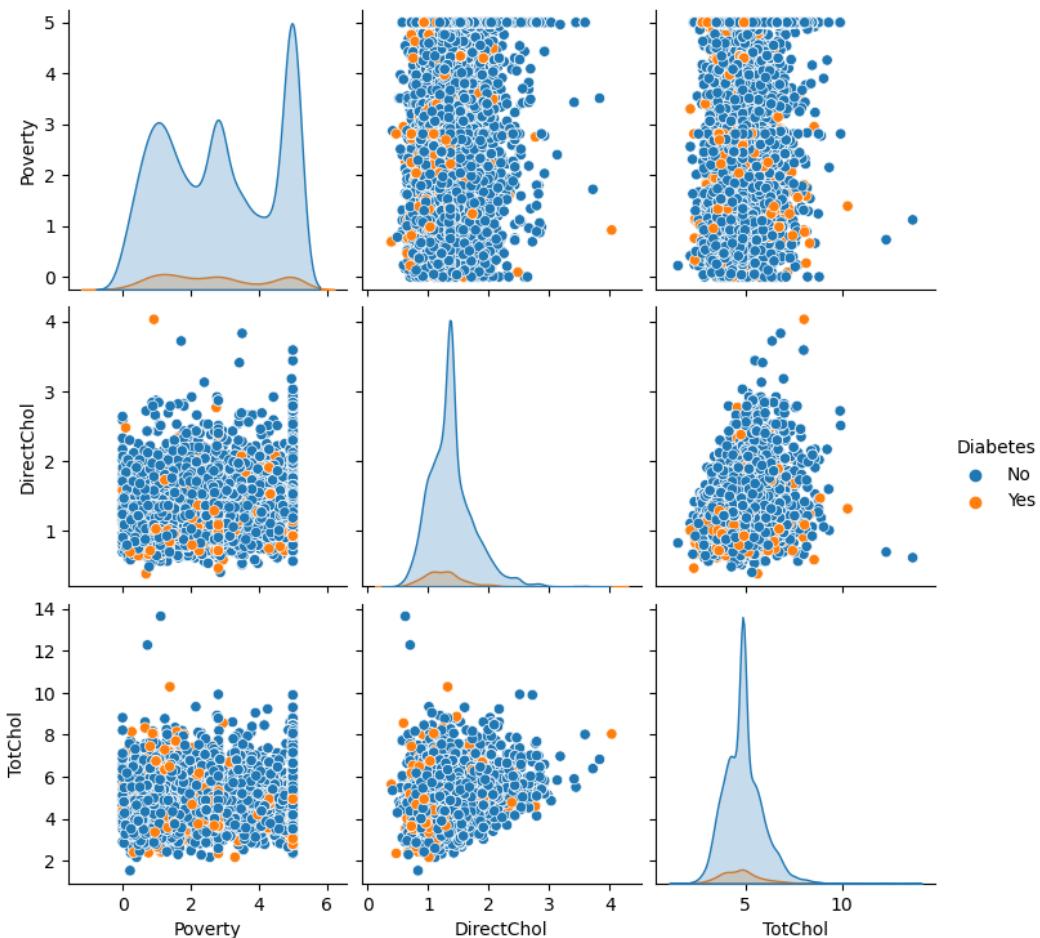
اهمیت: یک تصویر هزاران برابر بیشتر از کلمات ارزش دارد.

جمله فوق، به نوعی اهمیت مصورسازی را برای ما نمایان می‌کند. درواقع ما با انجام این‌کار، درک بهتر و سریع‌تری نسبت به داده‌های عظیم خود خواهیم داشت. قابل ذکر است که Dataset مورد استفاده ما در اینجا، حدود ۱۰۰۰۰ رکورد را در خود جای داده است.

ضمونا در کنار این موارد، با مصورسازی داده‌ها دیگر نیازی به توضیحات اضافه نخواهیم داشت و بسیاری از افراد عادی قادر به درک موضوع مطرح شده خواهد بود [۱۰].

۵-۴-۲ انواع نمودارها و داده‌های اجمالی

نمودار Scatter یا **Pairplot**



شکل ۲۲-۲

در شکل ۲۲-۲ نمودار Pairplot را مشاهده می‌کنیم. این نمودار برای تجسم رابطه بین دو متغیر استفاده می‌شود. هر نقطه داده با یک نقطه در نمودار نشان داده می‌شود که محور x و y متریک‌های مختلف را نشان می‌دهد. نمودارهای پراکندگی، به شناسایی خوش‌ها، الگوها یا همبستگی‌های بین متغیرها کمک می‌کند و محققان را قادر می‌سازد تا در انتخاب شاخصه‌های کلیدی، تصمیمات آگاهانه بگیرند [۱۰].

در این نمودار، نقاط نارنجی رنگ نشان دهنده افراد مبتلا به دیابت است و نقاط آبی رنگ نشان دهنده افرادی بدون ابتلا به این بیماری است و این نمودار از دادگان همین پروژه استخراج شده است. به راحتی می‌توان تشخیص داد که افرادی که به دیابت مبتلا هستند در هر ویژگی فیزیولوژی در چه سمتی تمرکز دارند.

نقشه‌های حرارتی

نقشه‌های حرارتی در تجسم مقادیر زیادی از داده‌ها در قالب ماتریس مانند موثر هستند. آنها از گرادیان رنگ برای نشان دادن مقدار مقادیر در دو متغیر استفاده می‌کنند. نقشه‌های حرارتی به ویژه در شناسایی همبستگی‌ها و کشف الگوهای پنهان در مجموعه داده‌های چند بعدی مفید هستند، و به انتخاب ویژگی و تشخیص نقاط پرت کمک می‌کنند. سایه‌های تیره‌تر با همبستگی قوی‌تر (مثبت یا منفی) مطابقت دارد [۱۲].

نمونه‌ای از این نمودار که در پروژه خودمان استفاده کرده‌ایم در شکل ۲۳-۲ نمایش داده شده است.

	Poverty	DirectChol	TotChol
Poverty	1.000000	0.114370	0.078125
DirectChol	0.114370	1.000000	0.221467
TotChol	0.078125	0.221467	1.000000

شکل ۲۳-۲: نقشه حرارتی

همان‌طور که ملاحظه می‌شود، در نقاط گرم‌تر، ارتباط بین دو متغیر قوی‌تر است.

نمودارهای هیستوگرام (میله‌ای) برای نمایش داده‌های دسته‌بندی یا عددی در قالب گرافیکی استفاده می‌شود. نمودارهای میله‌ای داده‌ها را از طریق میله‌های عمودی نشان می‌دهند، در حالی که هیستوگرام‌ها توزیع متغیرهای پیوسته را نشان می‌دهند. این تجسم‌ها برای درک توزیع هر ویژگی و شناسایی عدم تعادل داده‌ها یا نقاط پرت مفید هستند [۱۲]. نمونه‌ای از آن‌ها در شکل ۲-۴ و ۴-۴ در بخش پیاده‌سازی نمایش داده شده‌اند.

۶-۴-۲ اندازه‌گیری میزان خطای دقت

به دلیل پدید آمدن مشکلاتی از قبیل پیش‌بازش و کم‌بازش داده‌ها که در بخش‌های قبلی به آن اشاره شد مدل‌های ما دچار اشتباهاتی در پیش‌بینی‌ها می‌شود. در این بخش به بیان انواع خطاهای و روابطی که برای سنجش آن‌ها به کار گرفته‌ایم می‌پردازیم [۱۱].

ابتدا به معرفی ۴ تاپل مختلف می‌پردازیم که بعدا از آن‌ها در محاسبه میزان خطای استفاده خواهیم کرد [۱۱]:

- **TP^{۱۸}:** این تاپل موارد مثبت واقعی را علامت‌گذاری می‌کند. مثلا در این پروژه مواردی که احتمال ابتلا به دیابت در آن‌ها مثبت پیش‌بینی شده و در دادگان هم مثبت بوده در این دسته قرار می‌گیرد.
- **TN^{۱۹}:** این تاپل موارد منفی واقعی را علامت‌گذاری می‌کند. مثلا در این پروژه مواردی که احتمال ابتلا به دیابت در آن‌ها منفی پیش‌بینی شده و در دادگان هم منفی بوده در این دسته قرار می‌گیرد.
- **FP^{۲۰}:** این تاپل موارد مثبت کاذب را علامت‌گذاری می‌کند. مثلا در این پروژه مواردی که احتمال ابتلا به دیابت در آن‌ها مثبت پیش‌بینی شده اما در دادگان هم منفی بوده در این دسته قرار می‌گیرد.
- **FN^{۲۱}:** این تاپل موارد منفی کاذب را علامت‌گذاری می‌کند. مثلا در این پروژه مواردی که احتمال ابتلا به دیابت در آن‌ها منفی پیش‌بینی شده اما در دادگان هم مثبت بوده در این دسته قرار می‌گیرد.

^{۱۸}True positives

^{۱۹}True negatives

^{۲۰}False positives

^{۲۱}False negatives

:**(1 - Specificity)** یا **FPR** •

رابطه ۱۶-۲ نشان دهنده نسبت افراد سالم شناسایی شده واقعی به کل افراد سالم (افراد اشتباهها بیمار تشخیص داده شده به علاوه افراد سالم شناسایی شده واقعی) است.

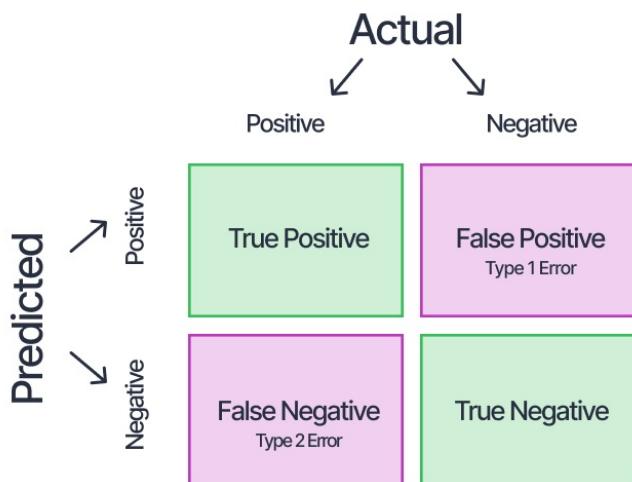
$$1 - Specificity = FPR = \frac{FP}{FP + TN} \quad (16-2)$$

:**Sensitivity** یا **TPR** •

رابطه ۱۷-۲ نشان دهنده نسبت بیماران شناسایی شده واقعی به کل بیماران (افراد اشتباهها سالم تشخیص داده شده به علاوه افراد بیمار شناسایی شده واقعی) است.

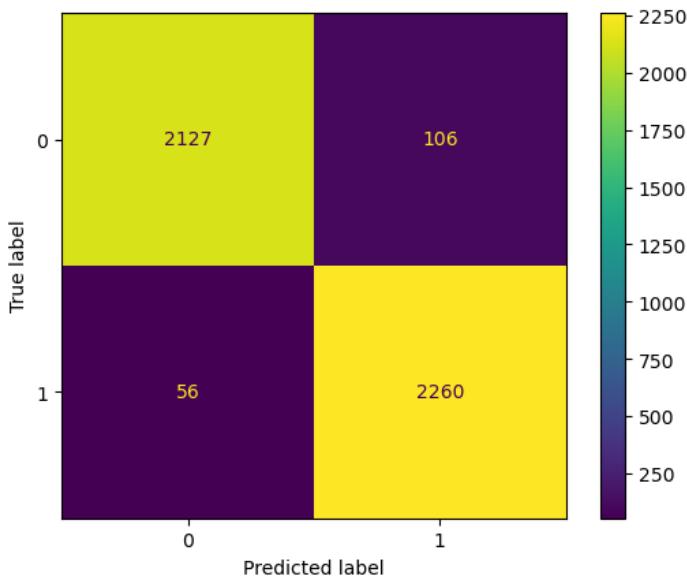
$$Sensitivity(Recall) = TPR = \frac{TP}{TP + FN} \quad (17-2)$$

تعدادی از این مقادیر را در ماتریس آشفتگی^{۲۲} نشان می‌دهند [۳۰] و ساختار آن در شکل ۲۴-۲ است. همچنین این ماتریس را برای مدل جنگل درختان تصادفی رسم کردیم (شکل ۲۵-۲) :



شکل ۲: ساختار ماتریس آشفتگی: پارامترهای مذکور در این بخش در این ماتریس قرار گرفته اند.

^{۲۲}Confusion



شکل ۲-۲۵: ساختار ماتریس آشفتگی برای الگوریتم جنگل درختان تصادفی

به طور کلی این ماتریس به ما خلاصه‌ای از درستی نتایج پیش‌بینی هایمان را نشان می‌دهد [۳۰]. حال میزان میزان دقیق را با رابطه ۱۸-۲ محاسبه می‌کنیم.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (18-2)$$

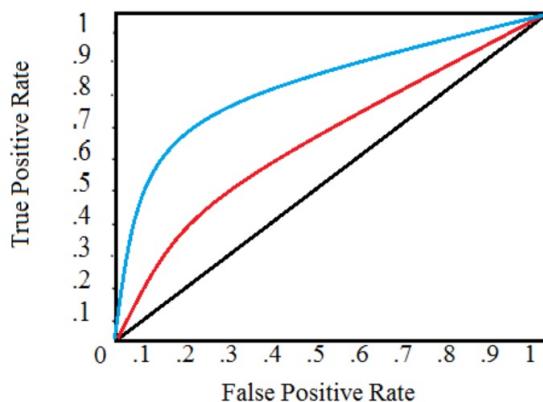
ROC و AUC ۷-۴-۲

برای اینکه مقادیر پیش‌بینی‌های صحیح و غلط را به صورت یک نمودار نمایش دهیم، از دو شاخصه TPR و FPR که در بخش قبلی آنها را معرفی کردیم استفاده می‌کنیم [۱۱]. ما در اینجا مقادیر مختلف برای هر مرز (اشاره به تابع Sigmoid که در بخش‌های قبلی مطرح شد) پیش‌بینی را از ۰ تا ۱ اندازه‌گیری و در نمودار علامت می‌زنیم [۱۱]. اگر مقدار TPR برای یک مرز برابر با ۱ باشد یعنی در آن مرز مدل خیلی خوب عمل کرده و اگر برابر ۵٪ باشد یعنی تصادفی عمل شده است [۱۱]. مثلاً در یک مرز در نظر می‌گیریم اگر مقدار تابع Sigmoid از ۷٪ بیشتر بود، آن مورد را یک مورد مثبت شناسایی کن. در نمودار ROC ^{۱۳} مولفه طول‌ها برابر با مقدار FPR است و مولفه عرض‌ها برابر TPR در آستانه‌های گوناگون است [۱۱].

سطح زیر این نمودار را با AUC ^{۱۴} نمایش می‌دهیم که بالا بودن این مساحت بیانگر این است که پیش‌بینی‌های بیشتری در این مدل درست بوده است [۱۱].

^{۱۳}Receiving Operating Characteristic

^{۱۴}Area Under the Curve



شکل ۲۶-۲: نمای کلی نمودار ROC

۵-۲ نتیجه‌گیری

در این فصل سعی شد تا ادبیات و مفاهیم مهمی که در انجام این پژوهه با آنها درگیر هستیم بیان شود از جمله بیان مفاهیم ابتدایی روش‌های داده کاوی شامل خوشه‌بندی و طبقه‌بندی (یادگیری بی‌نظرات و بانظرات)، معرفی و بیان روش عملکرد الگوریتم‌های مورد استفاده از جمله رگرسیون لجستیک، درخت تصمیم، جنگل درختان تصادفی، AdaBoost و بیز ساده.

همچنین درمورد داده‌ها و عملیات آنها از جمله روش‌های پیش‌پردازش داده و انواع مصور سازی‌ها و درنهایت روش‌هایی برای سنجش و مقایسه کارایی الگوریتم‌هاییمان نیز مفاهیمی مطرح شد.

یکی از نکاتی که از بخش روش‌های سنجش کارایی الگوریتم‌ها می‌توان برداشت کرد، این است که AUC، اندازه‌گیری عملکرد کلی یک الگوریتم در تمام آستانه‌های طبقه‌بندی است و به نوعی می‌توان گفت AUC بالاتر، نشان دهنده عملکرد بهتر در تشخیص نمونه‌های مثبت و منفی است.

از سوی دیگر، ACC، اندازه‌گیری نسبت نمونه‌های طبقه‌بندی صحیح (یعنی TN و TP) از تعداد کل نمونه‌ها است. این معیار، نشان دهنده صحت کلی پیش‌بینی‌های طبقه‌بندی‌کننده است.

در فصل آینده در مورد پیش زمینه‌های انجام این تحقیق مطالبی بیان می‌شود.

فصل ۳

کارهای پیشین

۱-۳ مقدمه

پیرو مباحث قبلی، در انجام پژوهش‌های گوناگون بررسی و مقایسه روش‌های مختلف در روند توسعه و تحقیق مجدد، نکته بسیار مهمی است و می‌تواند اشکالات کارهای پیشین را برطرف نمود و در زمینه موارد به روز آن‌ها را به کار گرفت. همچنین ایده‌ها و نکاتی در هر مقاله ذکر شده است که می‌تواند رهنماوهای مفید برای کارهای آتی طلقی شوند. از جمله الگوریتم‌های مختلف و بیان روش‌های داده کاوی مربوط به آن.

۲-۳ مقاله ۱

مقاله [۷] در سال ۲۰۱۸ با استفاده از داده‌های بیماران هندی^۱ روند بررسی را بر روی سه الگوریتم درخت تصمیم، SVM^۲ و بیز ساده انجام داده و نتایج بیانگر این بوده که الگوریتم بیز ساده ۷۶٪ به عنوان موثر ترین الگوریتم در این بررسی درنظر گرفته شده است.

نکته قابل ملاحظه در این مقاله این است که از الگوریتم SVM هم برای پیش‌بینی استفاده شده است که معمولاً برای دادگان با مقادیر زیاد روش مناسبی است [۲۹].

به طور خلاصه می‌توان گفت الگوریتم SVM یا ماشین بردار پشتیبان، با تقسیم داده‌ها به دسته‌های مختلف بر اساس داده‌های آموزشی داده شده کار می‌کند که سعی می‌کند خطی را ترسیم کند که به عنوان ابرصفحه (یک فضای n بعدی زیرمجموعه ابعاد $n-1$ یا معادل آن که ۱ بعده‌مبند در V می‌باشد). شناخته می‌شود که به بهترین وجه دسته‌های مختلف داده‌ها را از هم جدا می‌کند. این کار را با یافتن ابرصفحه بهینه که فاصله بین نزدیکترین نقاط داده هر دسته را به حداقل می‌رساند، انجام می‌دهد [۲۹].

به عبارت ساده‌تر، داده‌ها را به صورت نقطه‌هایی روی یک تکه کاغذ در نظر بگیرید و هر نقطه به دسته خاصی تعلق دارد. الگوریتم SVM سعی می‌کند خطی را بر روی کاغذ بکشد که نقاط را به بهترین

^۱PIDD

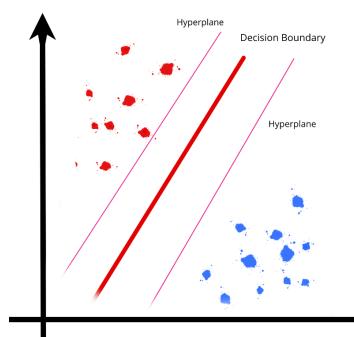
^۲Support Vector Machine

نحو از هم جدا کند، به طوری که نقاط متعلق به یک دسته در یک طرف خط و نقاط متعلق به دسته دیگر در سمت دیگر قرار گیرند.

با این حال، گاهی اوقات نقاط داده را نمی‌توان به طور کامل با یک خط از هم جدا کرد. در چنین مواردی، الگوریتم از تکنیکی به نام "ترفند هسته" برای تبدیل داده‌ها به فضایی با ابعاد بالاتر استفاده می‌کند، جایی که یافتن یک خط جداکننده آسان‌تر می‌شود [۲۹].

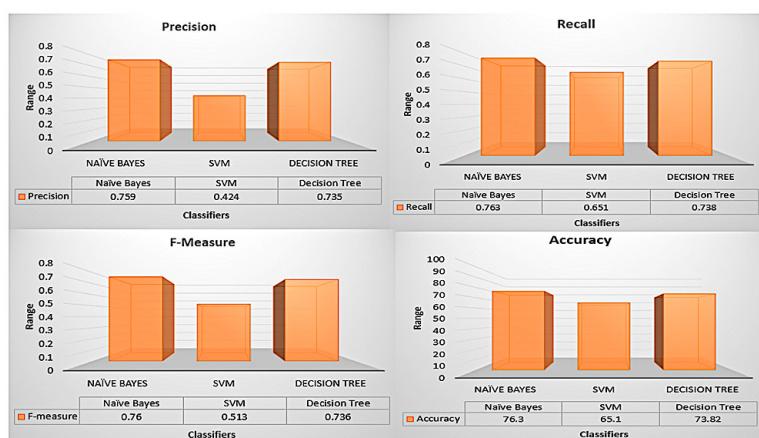
هنگامی که خط رسم شد، نقاط داده جدید و نادیده را می‌توان با تعیین اینکه در کدام سمت خط قرار می‌گیرند، به یک دسته اختصاص داد.

هدف SVM یافتن بهترین ابرصفحه است که حاشیه بین نقاط داده دسته‌های مختلف را به حداقل می‌رساند، که به پیش‌بینی دقیق داده‌های جدید کمک می‌کند [۲۹].

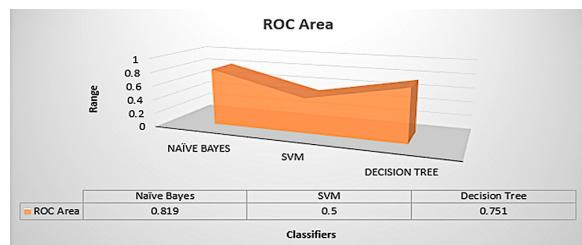


شکل ۱-۳: مدل SVM

خلاصه معیارهای دقت طبقه‌بندی در این مقاله به شرح زیر است:



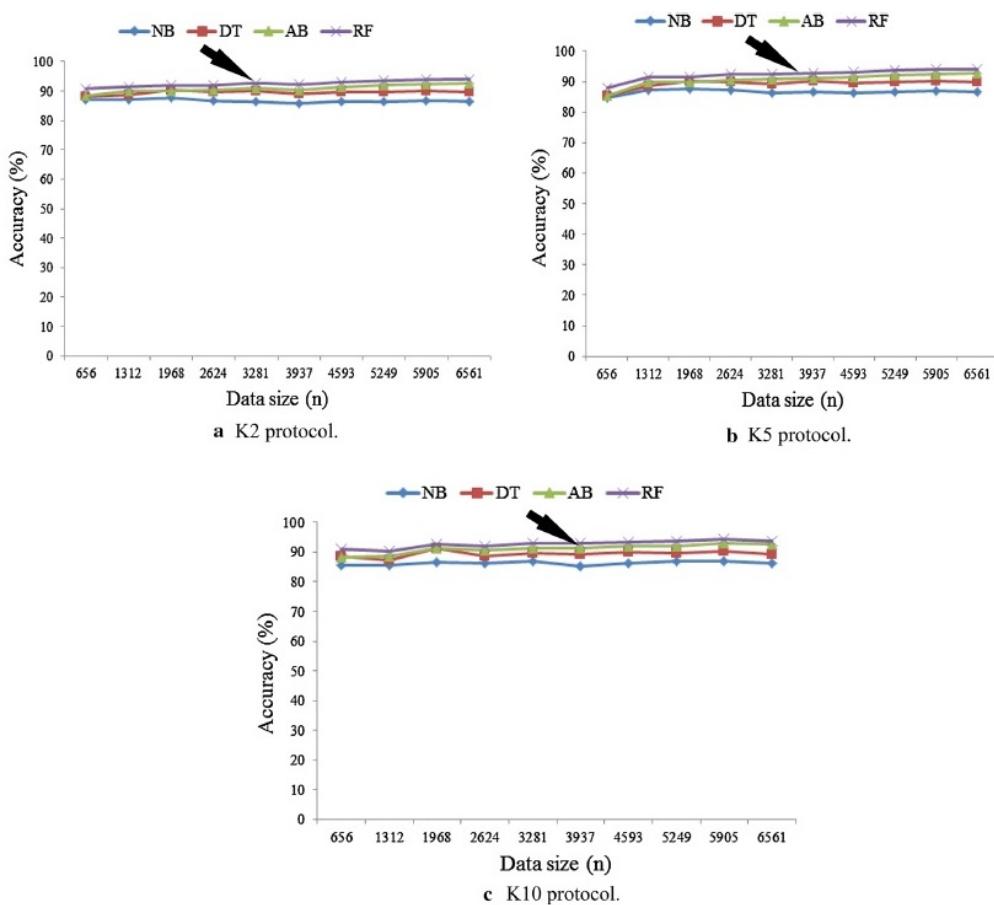
شکل ۲-۳: دقت‌های اندازه‌گیری در مقاله اول



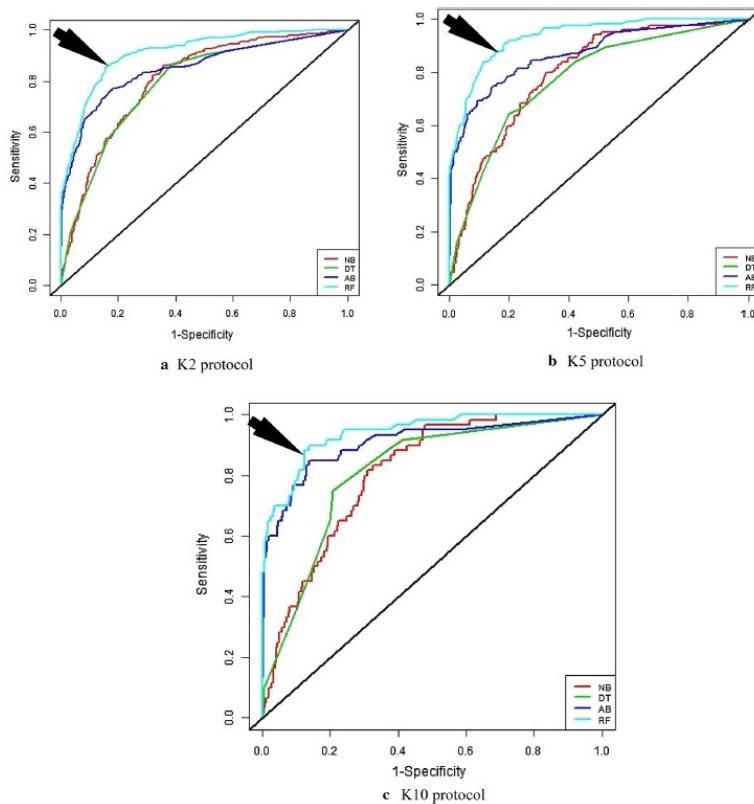
شکل ۳-۳: منحنی ROC در مقاله اول

۳-۳ مقاله ۲

در مقاله اشاره شده [۶] در سال ۲۰۲۰ روش‌های مختلفی برای پیش‌بینی ابتلا به دیابت انجام شده و جنگل درختان تصادفی در اعتباری سنجی متقابل K-Fold "مکرر" با مقدار $K=10$ بهترین نتیجه طبقه‌بندی را با دقت ۹۴٪ حاصل کرده است. در این مقاله روش‌های بیز ساده و AdaBoost هم مورد استفاده قرار گرفتند که نتایج آنها داری دقت مناسبی نبوده است.



شکل ۳-۴: دقت‌های اندازه‌گیری در مقاله دوم



شکل ۳-۵: منحنی ROC در مقاله دوم

ضمیرنا در این مقاله، از همان مجموعه دادگانی استفاده شده که در همین پایان نامه مورد استفاده قرار گرفته است.

۴-۳ نتیجه‌گیری

در این بخش، کوشیدیم تا برخی از مقالاتی که این موضوع را مورد بحث قرار دادند را بررسی کنیم و متوجه تفاوت‌ها در زمینه یافتن بهترین الگوریتم برای تشخیص ابتلا به دیابت، شویم.

در نتیجه، حاصل مطالعه مقالات اساسی که در گذشته تالیف شده بودند، این شد که الگوریتم‌های مختلفی در روند توسعه استفاده پروژه‌مان مورد بررسی واقع شوند که از جمله می‌توان به الگوریتم‌های بیز ساده، جنگل درختان تصادفی و الگوریتم درخت تصمیم و رگرسیون لجستیک اشاره کرد.

فصل ۴

روش‌ها و نتایج

۱-۴ مقدمه

در این بخش بررسی می‌کنیم که الگوریتم‌های گوناگونی که برای ساخت مدل‌های گوناگون استفاده کردیم تا چه حد می‌تواند قابل اعتماد واقع شود و میزان خطا در هر کدام چقدر است. چرا که ما در این پژوهه، کار پیش‌بینی انجام دادیم و داده‌های آزمون به ما نشان داده که برخی از ردیف‌های به صورت اشتباه پیش‌بینی شده‌اند. برای انتخاب بهترین مدل روشی که انجام می‌شود این است که میزان خطاهای را با روش‌های گوناگون به دست آوریم و سپس الگوریتمی که بیشترین صحت را ارائه داده به عنوان بهترین مدل برگزینیم [۱۱].

همچنین برای بخش دیگری از این سنجش می‌توانیم نمودارهای گوناگونی را رسم کنیم از جمله ROC که می‌تواند این میزان دقت را به صورت بصری به ما نمایش دهد.

و در نهایت به بررسی روش طراحی یک مدل ساده برای سامانه آنلاین پیش‌بینی دیابت با ابزارهای اینترنت اشیاء و دو چارچوب NodeRed و Flask پرداختیم.

۲-۴ روش پیشنهادی

۱-۲-۴ پیاده‌سازی

پس از وارد کردن کتابخانه‌های مورد نیاز، به عملیات وارد کردن داده‌ها و پیش‌پردازش آن‌ها پرداختیم:

```
dataset_new = dataset_new.dropna(subset=['Diabetes'])
```

در کد بالا، عملیات حذف داده‌های گم شده در ستون متغیر وابسته انجام شد. درواقع ردیف‌هایی که ستون آخرشان (یعنی Diabetes) دارای مقادیر نامشخص بودند را حذف کردیم. زیرا تمام مدل سازی‌های ما وابسته به آخرین ستون است و اگر این ستون مقادیر نادرستی داشته باشد، مدل‌هایی که می‌سازیم دقت پایینی خواهند داشت.

پس از شمارش تعداد ردیف‌هایی با مقادیر گم شده، عملیات مربوط به جایگزینی آن‌ها را با روش زیر

انجام دادیم. یعنی مقادیر ستون‌هایی که مقدار پیوسته داشتند را با میانگین ستون جایگزین کردیم و در مقادیر گسته، از مُد (نمونه‌ای با بیشترین تکرار) استفاده کردیم.

```
dataset_new["Weight"].fillna(dataset_new["Weight"].mean(), inplace = True)
dataset_new["Height"].fillna(dataset_new["Height"].mean(), inplace = True)
dataset_new["BMI"].fillna(dataset_new["BMI"].mean(), inplace = True)
dataset_new["BPSysAve"].fillna(dataset_new["BPSysAve"].mean(), inplace = True)
dataset_new["BPDiaAve"].fillna(dataset_new["BPDiaAve"].mean(), inplace = True)
dataset_new["DirectChol"].fillna(dataset_new["DirectChol"].mean(), inplace = True)
dataset_new["TotChol"].fillna(dataset_new["TotChol"].mean(), inplace = True)

dataset_new.isnull().sum()
dataset_new["PhysActive"].fillna(dataset_new["PhysActive"].mode()[0], inplace = True)
dataset_new["Education"].fillna(dataset_new["Education"].mode()[0], inplace = True)
dataset_new["MaritalStatus"].fillna(dataset_new["MaritalStatus"].mode()[0], inplace = True)
dataset_new["Work"].fillna(dataset_new["Work"].mode()[0], inplace = True)

dataset_new.isnull().sum()
```

شکل ۴-۴: جایگزینی مقادیر نامشخص با متoste‌های آماری

یکی از کارهای دیگری که انجام شد، جایگزینی مقادیر \circ با NaN در ستون‌هایی با مقادیر پیوسته بود مثل سن. زیرا به خوبی مشخص است این موارد از جمله داده‌های پرت محاسبه می‌شوند که دقت مدل‌های مارا کاهش می‌دهند.

```
dataset_new[['Education', 'MaritalStatus', 'Work', 'Weight', 'Height', 'BMI',
'BPSysAve', 'BPDiaAve', 'DirectChol', 'TotChol', 'PhysActive']] = dataset_new
[[['Education', 'MaritalStatus', 'Work', 'Weight', 'Height', 'BMI', 'BPSysAve',
'BPDiaAve', 'DirectChol', 'TotChol', 'PhysActive']]].replace(0, np.NaN)
```

سپس عملیات مربوط ایجاد مصورسازی داده‌ها صورت گرفت:

```
YesDia = dataset_new['Diabetes'].values == 'Yes'
NoDia = dataset_new['Diabetes'].values == 'No'
YesDia=dataset_new[YesDia]
NoDia=dataset_new[NoDia]
```

```
Race1 = YesDia['Race1'].tolist()
Race0 = NoDia['Race1'].tolist()
```

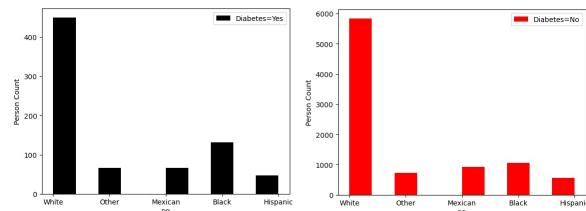
```
plt.hist([Race1], color=[
'Black'], label=['Diabetes=Yes'])
plt.xlabel('BP')
plt.ylabel('Person Count')
plt.legend()
plt.show()
```

```
plt.hist([Race0], color=[

'Red'], label=['Diabetes=No'])
plt.xlabel('BP')
```

```
plt.ylabel('Person Count')
plt.legend()
plt.show()
```

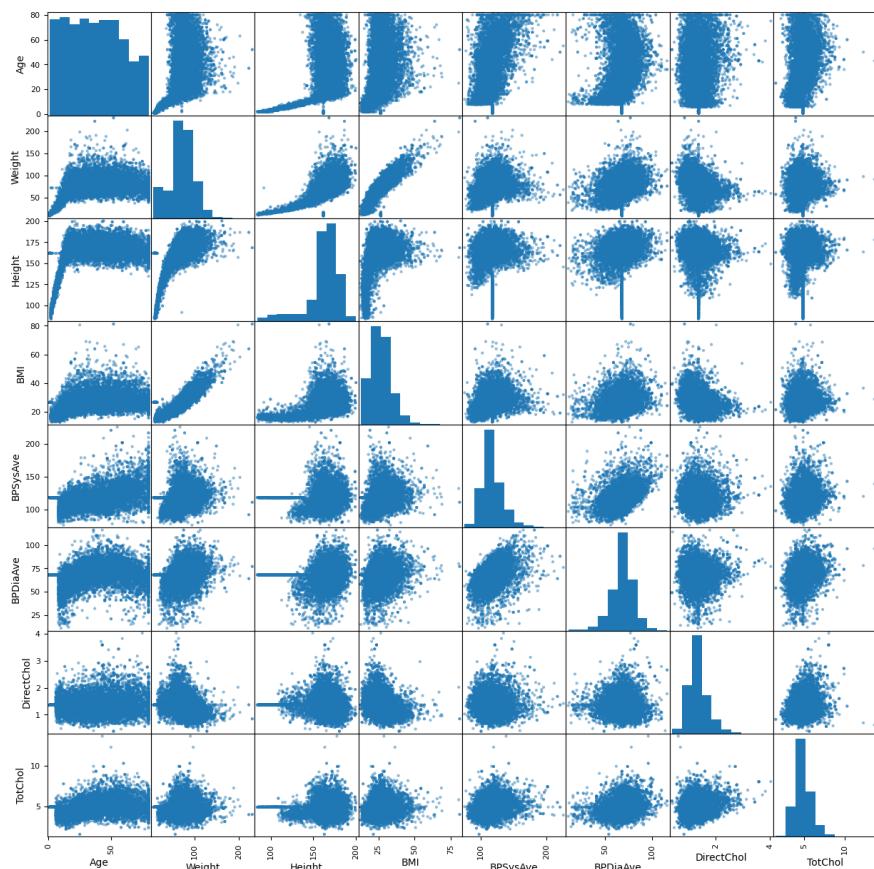
نتیجهٔ عملیات فوق، نمودارهای زیر می‌باشد (در این نمودار، تاثیر نژاد در ابتلا به دیابت مشخص است):



شکل ۲-۴: نمودار مقایسهٔ ابتلا به دیابت در نژادهای مختلف

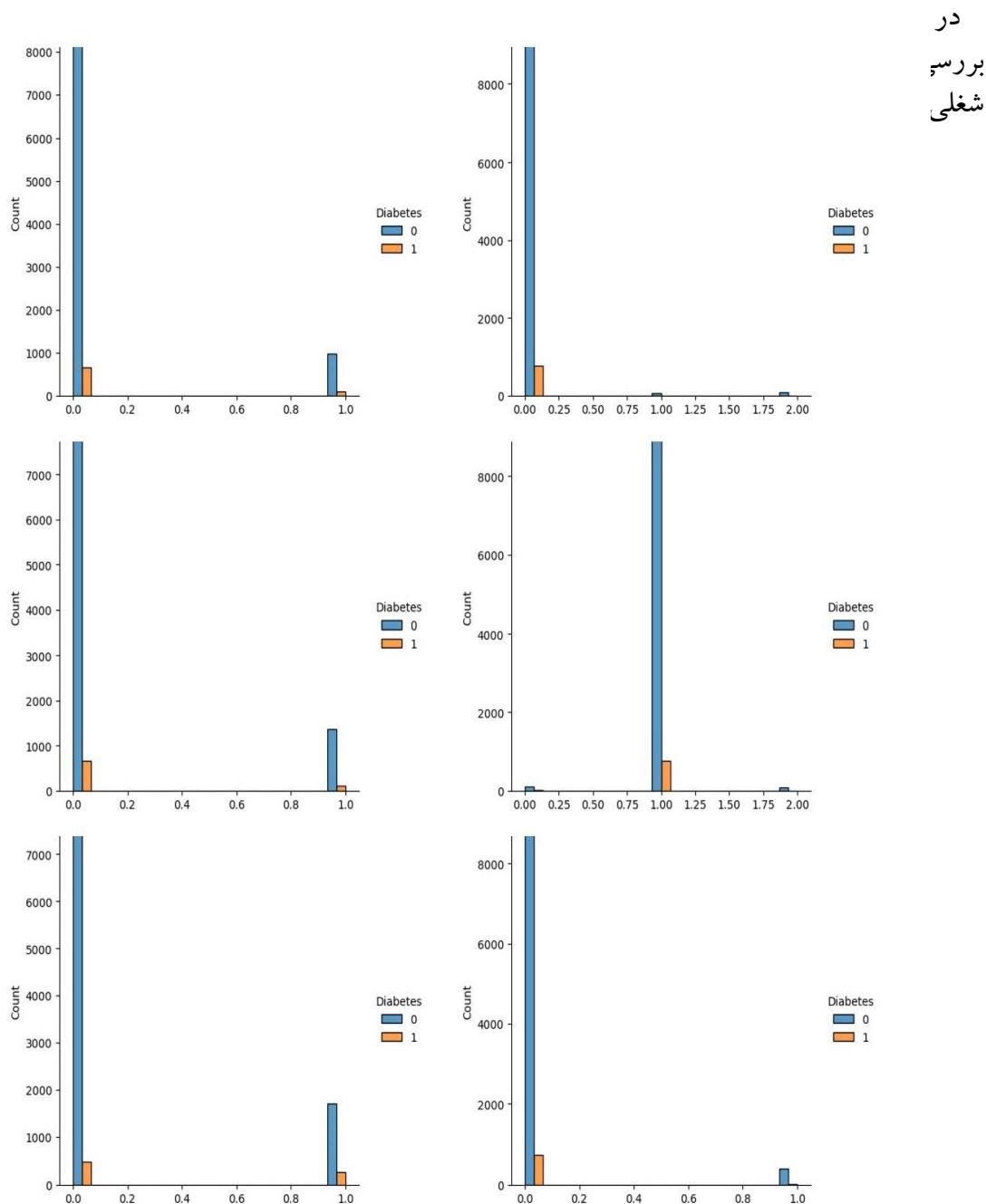
سپس با کد زیر، ماتریس scatter برای نمایش همبستگی میان داده‌ها رسم می‌کنیم:

```
scatter_matrix(dataset , figsize=(15, 15))
plt.show()
```

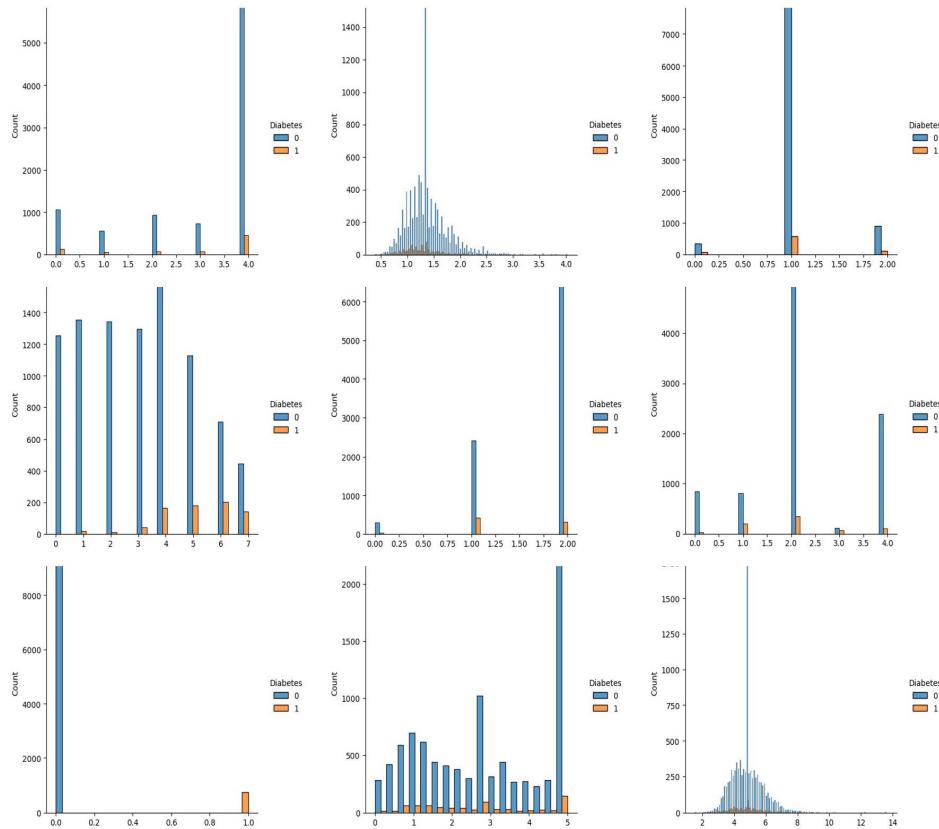


شکل ۳-۴: ماتریس Scatter

در شکل ۳-۴، برخی عوامل مرتبط با هم مشخص اند. (مثل رابطهٔ کلسترول و وزن)



شکل ۴-۴: نمودار عوامل موثر در ابتلا دیابت (۱)



شکل ۴-۵: نمودار عوامل موثر در ابتلا دیابت (۲)

همچنین از کد زیر برای رسم نمودارهای فوق استفاده شد.

```

sns.countplot(x = 'PhysActive', data = YesDia)
sns.countplot(x = 'PhysActive', data = NoDia)
sns.lineplot(x="Age", y="Diabetes", data=dataset_new)

for i in dataset_new.columns:
    sns.displot(dataset, x=i, multiple="dodge", hue="Diabetes")

```

نمودار نقشه حرارتی را نیز با این کد رسم کردیم:

```

df = dataset_new
corr = df.corr()
#Displaying dataframe of correlation values
corr.style.background_gradient(cmap = 'coolwarm')

```

پارامترهایی که با هم ارتباط دارند، با رنگ‌های گرم مشخص اند.

	Poverty	DirectChol	TotChol
Poverty	1.000000	0.114370	0.078125
DirectChol	0.114370	1.000000	0.221467
TotChol	0.078125	0.221467	1.000000

شکل ۴-۶: نقشه حرارتی

حال پس از بخش مصور سازی به سراغ آماده سازی داده‌ها برای ایجاد مدل‌ها پرداختیم. پس متغیرهای رقمنی را با کد زیر ایجاد کردیم:

```
df3 = dataset_new.copy()
df3 = pd.get_dummies(df3, columns = ['Gender', 'Race1', 'Education',
'MaritalStatus', 'Work', 'PhysActive', 'Diabetes'], drop_first = True)
print(df3.columns)
```

سپس ستون Y و X هایمان مشخص کردیم تا داده‌هایمان را به بخش‌های آموزشی و سنجش تقسیم کنیم. چون در ساخت مدل‌ها، باید داده‌ها را به چهار دسته‌های X آموزشی، های X سنجش، های Y آموزشی و های Y سنجش، تقسیم کنیم [۱۲]. مقیاسی که برای داده‌های سنجش در نظر گرفته شده ۲۵٪ است.

```
X = dataset_new.iloc[:, :-1].values
Y = dataset_new.iloc[:, -1].values
```

```
XTrain, XTest, YTrain, YTest = train_test_split(X, Y, test_size = 0.25,
random_state = 0)
```

با توجه به اینکه برای هر مدل باید الگوریتم K-Fold تکرار شونده برای $K=2,5,10$ به تعداد ۲۰ مرتبه انجام شود و مقادیر AUC و ACC به دست آیند، دوتابع ایجاد می‌کنیم:
ساخت لیست مربوط به مقادیر ACC :

```
#Importing required libraries
from sklearn.model_selection import RepeatedKFold
from sklearn.metrics import accuracy_score
from numpy import mean
from numpy import std
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

def Kfold_modulation(input_model):
    #Implementing cross validation
```

```

k_list = [2,5,10]
acc_list=[]
for k in k_list:
    kf = KFold(n_splits=k,shuffle=False, random_state=None)
    model = input_model
    acc_score = []
    scores = cross_val_score(model, X, Y, scoring='accuracy', cv=kf)
    avg_acc_score = mean(scores)
    # print('Avg acc: avg_acc_score')
    acc_list.append(avg_acc_score)
return acc_list

```

ساخت لیست مربوط به مقادیر : AUC

```

from sklearn.model_selection import cross_val_score
def Kfold_modulation2(input_model):
    #Implementing cross validation
    k_list = [2,5,10]
    auc_list=[]
    for k in k_list:
        mean_score = cross_val_score
            (input_model, X, Y, scoring="roc_auc", cv = k).mean()
        auc_list.append(mean_score)
    return auc_list

```

سپس هر مدل را ساخته و میزان AUC و ACC را برای هر کدام در یک لیست می‌ریزیم:

```

from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg_acc=Kfold_modulation(logreg)
logreg_auc=Kfold_modulation2(logreg)
logreg.fit(XTrain, YTrain)
logreg_pred=logreg.predict(XTest)
print(logreg_acc)

```

```

from sklearn.ensemble import RandomForestClassifier
ranfor = RandomForestClassifier()
ranfor_acc=Kfold_modulation(ranfor)
ranfor_auc=Kfold_modulation2(ranfor)
ranfor.fit(XTrain, YTrain)
ranfor_pred=ranfor.predict(XTest)

```

```

from sklearn.tree import DecisionTreeClassifier
DecTree = DecisionTreeClassifier()
DecTree_acc=Kfold_modulation(DecTree)
DecTree_auc=Kfold_modulation2(DecTree)
DecTree.fit(XTrain, YTrain)

from sklearn.ensemble import AdaBoostClassifier
AdaBoost = AdaBoostClassifier()
AdaBoost_acc=Kfold_modulation(AdaBoost)
AdaBoost_auc=Kfold_modulation2(AdaBoost)
AdaBoost.fit(XTrain, YTrain)
AdaBoost_pred=AdaBoost.predict(XTest)

from sklearn.naive_bayes import GaussianNB
nb=GaussianNB()
nb_acc=Kfold_modulation(nb)
nb_auc=Kfold_modulation2(nb)
nb.fit(XTrain, YTrain)
nb_pred=nb.predict(XTest)

```

سپس مقادیر AUC و ACC را که برای هر الگوریتم را که در لیست‌های جداگانه قرار دارند، را باهم ترکیب می‌کنیم و یک دیتابست حاوی تمامی مقادیر AUC و ACC ایجاد می‌کنیم تا با استفاده از آن، نمودارهای ROC ACC و AUC را رسم می‌کنیم.

```

acc_list0=[logreg_acc,ranfor_acc,DecTree_acc,AdaBoost_acc,nb_acc]
acc_list=[]
for i in acc_list0 :
    my_formatted_list = [ '%.4f' % elem for elem in i ]
    list=[]
    acc_list.append(my_formatted_list)
auc_list0=[logreg_auc,ranfor_auc,DecTree_auc,AdaBoost_auc,nb_auc]
auc_list=[]
for i in auc_list0 :
    my_formatted_list = [ '%.4f' % elem for elem in i ]
    list=[]
    auc_list.append(my_formatted_list)
print(auc_list)

```

```

bar=pd.DataFrame([acc_list[0],acc_list[1],acc_list[2],acc_list[3],acc_list[4]])
bar['algo'] = ['LR','RF','DT','AB','NB']
bar.columns=['K2' , 'K5' , 'K10','Algorithm']

```

```

bar['K2']=bar['K2'].astype('float64')*100
bar['K5']=bar['K5'].astype('float64')*100
bar['K10']=bar['K10'].astype('float64')*100

print('ACC','\n',bar,'\\n')
barauc=pd.DataFrame([auc_list[0],auc_list[1],auc_list[2],auc_list[3],auc_list[4]])
barauc['algo'] = ['LR', 'RF', 'DT', 'AB', 'NB']
#barauc.insert()=['logreg_auc', 'ranfor_auc', 'DecTree_auc', 'AdaBoost_auc', 'nb_auc']
barauc.columns=['K2' , 'K5' , 'K10','Algorithm']
barauc['K2']=barauc['K2'].astype('float64')*100
barauc['K5']=barauc['K5'].astype('float64')*100
barauc['K10']=barauc['K10'].astype('float64')*100
print('\\n','AUC','\\n',barauc)

```

سپس با کد زیر، نمودار مقایسه ACC و AUC را برای الگوریتم‌های مختلف مان در Fold‌های ۵، ۲ و ۱ رسم می‌کنیم:

```

bar['K2'] = bar['K2'].astype('float64')
bar['K5'] = bar['K5'].astype('float64')
bar['K10'] = bar['K10'].astype('float64')

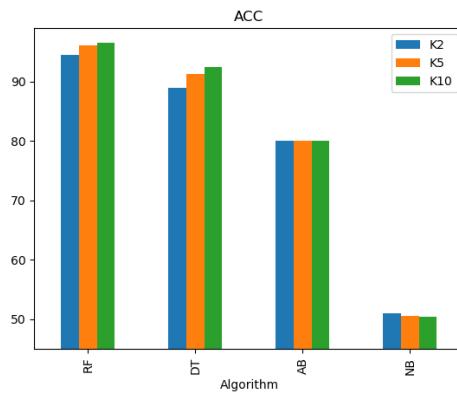
# plot grouped bar chart
bar.plot(x='Algorithm',
          kind='bar',
          stacked=False,
          ylim=[82, 97],
          title='ACC')

barauc['K2'] = barauc['K2'].astype('float64')
barauc['K5'] = barauc['K5'].astype('float64')
barauc['K10'] = barauc['K10'].astype('float64')

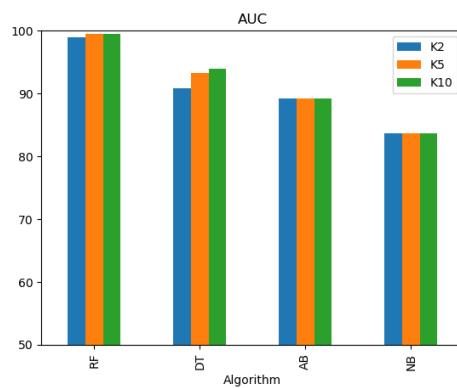
# plot grouped bar chart
barauc.plot(x='Algorithm',
            kind='bar',
            stacked=False,
            ylim=[60, 98],
            title='AUC')

```

نمودار ها:

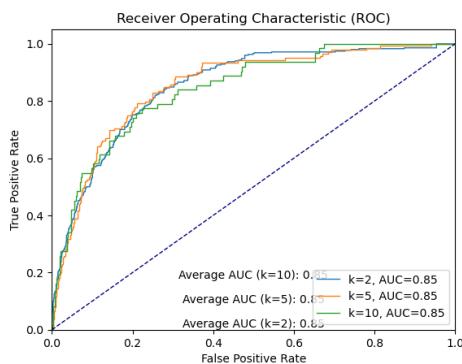


شکل ۴-۷: نمودار ACC

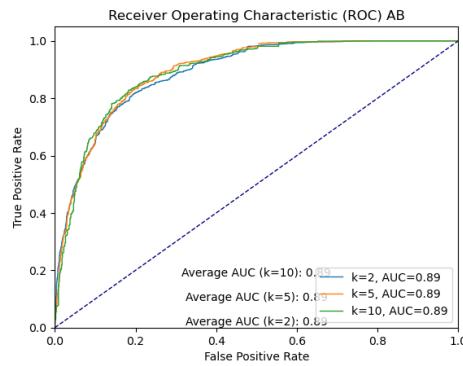


شکل ۴-۸: نمودار AUC

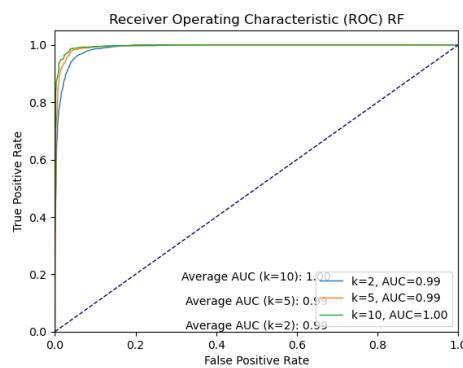
حال برای الگوریتم‌های مختلف، نمودار ROC Fold را در ROC های مختلف رسم می‌کنیم و میانگین آن نیز مشاهده می‌شود:



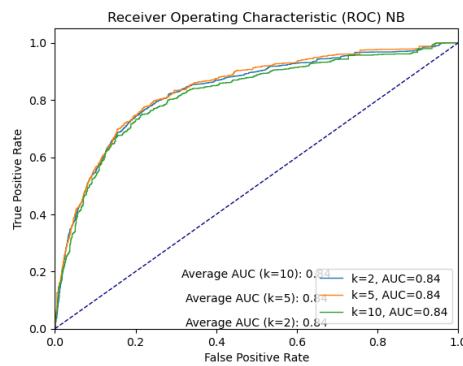
شکل ۴-۹: نمودار ROC برای الگوریتم رگرسیون لجستیک



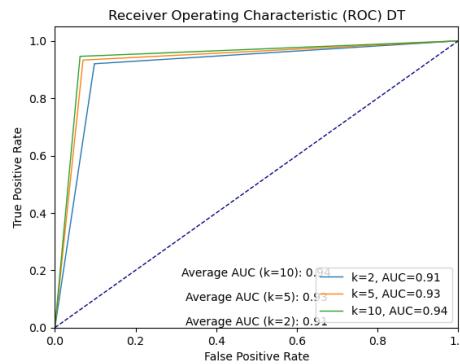
شکل ۴-۱۰: نمودار ROC برای الگوریتم AdaBoost



شکل ۱۱-۴: نمودار ROC برای الگوریتم جنگل درختان تصادفی



شکل ۱۲-۴: نمودار ROC برای الگوریتم Naïve Bayes



شکل ۱۳-۴: نمودار ROC برای الگوریتم درخت تصمیم

از کد زیر برای رسم نمودارهای فوق استفاده کردیم:

```

import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import StratifiedKFold, RepeatedKFold

# Define the repeated k-fold cross-validation parameters
n_folds = [2, 5, 10]
n_repeats = 20

# Define the classifier
clf = logreg

# Initialize the mean AUC list for each k
mean_auc_list = []

# Perform repeated k-fold cross-validation for each k
for n_fold in n_folds:
    auc_list = []
    rskf = RepeatedKFold(n_splits=n_fold, n_repeats=n_repeats)

    # Perform cross-validation
    for train_index, test_index in rskf.split(X, Y):
        X_train, X_test = X[train_index], X[test_index]
        Y_train, Y_test = Y[train_index], Y[test_index]
        clf.fit(X_train, Y_train)
        Y_prob = clf.predict_proba(X_test)[:, 1]
        fpr, tpr, _ = roc_curve(Y_test, Y_prob)
        auc_val = auc(fpr, tpr)
        auc_list.append(auc_val)

    mean_auc_list.append(np.mean(auc_list))

```

```

# Calculate the mean AUC and update mean AUC list
mean_auc = np.mean(auc_list)
mean_auc_list.append(mean_auc)

# Plot ROC curve
plt.plot(fpr, tpr, lw=1, label=f'k={n_fold}, AUC={mean_auc:.2f}')

# Set plot settings
plt.plot([0, 1], [0, 1], color='navy', lw=1, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc="lower right")

# Display average AUC values
plt.text(0.5, 0.02, f'Average AUC (k=2): {mean_auc_list[0]:.2f}', transform=plt.gca().transAxes,
ha='center', va='center')
plt.text(0.5, 0.1, f'Average AUC (k=5): {mean_auc_list[1]:.2f}', transform=plt.gca().transAxes,
ha='center', va='center')
plt.text(0.5, 0.18, f'Average AUC (k=10): {mean_auc_list[2]:.2f}', transform=plt.gca().transAxes,
ha='center', va='center')
plt.show()

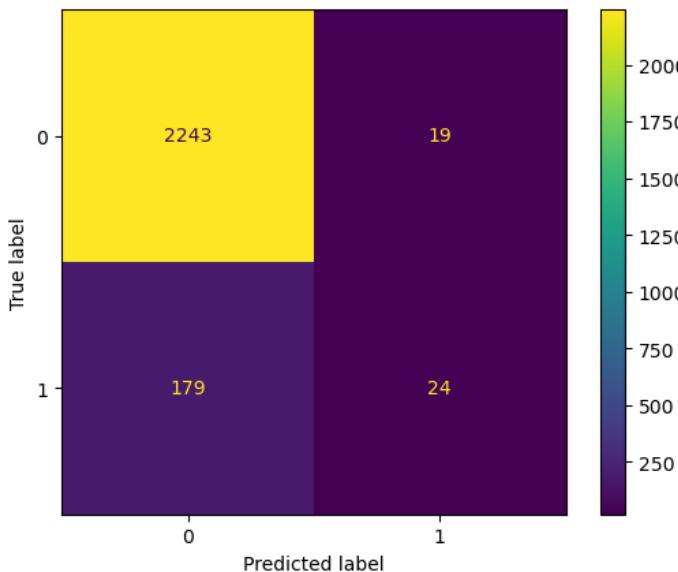
```

و در نهایت ماتریس آشتنگی را با استفاده از کد زیر برای الگوریتم‌های مختلف رسم کردیم:

```

from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
#In the models, we put test data to make predictions for us.
YP_ranfor = ranfor.predict(XTest)
cm = confusion_matrix(YTest, YP_ranfor, labels=None)
print(cm)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()

```



شکل ۱۴-۴: ساختار ماتریس آشفتگی برای الگوریتم رگرسیون لجستیک

Arduino ۲-۲-۴

آردوینو یک بستر منع باز است که برای ساخت پروژه‌های الکترونیکی استفاده می‌شود و شامل یک برد مدار فیزیکی قابل برنامه ریزی و یک نرم افزار به نام محیط توسعه یکپارچه (IDE) است که روی کامپیوتر اجرا می‌شود و برای نوشتن و آپلود کدهای کامپیوتري روی برد استفاده می‌شود. پلتفرم آردوینو به این دلیل محبوب است که برای بارگذاری کد جدید روی برد نیازی به برنامه نویس جداگانه ندارد و از نسخه ساده شده C++ برای برنامه نویسی استفاده می‌کند. این پلتفرم یک فرم فاکتور استاندارد را فراهم می‌کند که استفاده از عملکردهای میکروکنترلر را آسان‌تر می‌کند [۳۵].

بوردهای آردوینو در اتصالات برق، پین‌ها برای اتصال سیم‌ها، دکمه تنظیم مجدد، انواع نشانگرهای LED برای ارتباطات گوناگون از جمله سریال، IC اصلی و تنظیم‌کننده ولتاژ در مدل‌های مختلف تفاوت دارند [۳۵]. آردوینو می‌تواند با قطعات و سنسورهای مختلف تعامل داشته باشد و از طریق پروتکل MQTT می‌تواند با دستگاه‌های مختلف مبتنی بر اینترنت اشیاء تعامل داشته باشد [۳۶].

در مرور راه اندازی Arduino IDE دو نکته ضروری است:

۱. حتما باید پکیج مربوط به بورد مد نظر را بروی آن نصب کنید. (برای مدلی که استفاده می‌کردم file > performances > additional board manager url (WeMos D1 R1) آن را با لینک ^۱ در وارد کردم.)

۲. برای استفاده از MQTT باید بسته espMQTTClient را از طریق Arduino IDE نصب کرد.

MQTT

این پروتکل، یک پروتکل پیام رسانی است که برای ارتباط دستگاه‌ها در اینترنت اشیاء (IoT) استفاده می‌شود. سبک، کارآمد و مقیاس‌پذیر است و برای انتقال داده‌ها از طریق شبکه‌هایی با محدودیت منابع،

^۱http://arduino.esp8266.com/stable/package_esp8266com_index.json

ایده‌آل است. MQTT از پیام رسانی بین دستگاه‌ها و ابر^۲ پشتیبانی می‌کند و قابلیت اطمینان و ویژگی‌های امنیتی را ارائه می‌دهد [۳۳].

به جای ارتباط مستقیم بین کاربران و سرورها، MQTT از یک واسطه پیام برای مدیریت ارتباط بین فرستنده‌گان و دریافت کننده‌گان استفاده می‌کند. سرور، پیام‌های فرستنده‌گان را فیلتر کرده و بین گیرنده‌گان توزیع می‌کند. این کار فرستنده و گیرنده پیام‌ها را از نظر مکان و زمان جدا می‌کند [۳۳].

اجزای MQTT شامل کلاینت‌های MQTT است که دستگاه‌هایی هستند که با استفاده از MQTT ارتباط برقرار می‌کنند و یک سرور MQTT که پیام‌ها را بین کاربران روبدل می‌کند. کاربران می‌توانند پیام‌هایی را با موضوعات و داده‌ها منتشر کنند و همچنین می‌توانند برای دریافت پیام‌ها مشترک موضوعات خاصی شوند [۳۳].

همچنین (MQTT over WebSockets (WSS) یک پیاده‌سازی از MQTT است که اجازه می‌دهد داده‌ها مستقیماً در یک مرورگر وب دریافت شوند. ارتباط MQTT ایمن است و می‌توان با استفاده از پروتکل‌های SSL، گواهی‌ها و رمزهای عبور محافظت کرد [۳۳].

برای راه اندازی یک سرور MQTT به صورت محلی، می‌توان از ابزاری به نام mosquitto^۳ که در سیستم عامل‌های مختلف در دسترس است، استفاده کرد.

در مورد راه اندازی mosquitto نکته‌ای که قابل توجه است این است که بایستی پورت 1883 در تنظیمات این برنامه باز باشد و امکان اتصال به صورت ناشناس نیز ممکن باشد تا بتوان به راحتی و بدون دردسر به آن متصل شد برای این منظور مراحل زیر بایستی طی شوند.

۱. ابتدا آن را متوقف می‌کنیم:

در لینوکس: sudo systemctl stop mosquitto

در ویندوز: sc stop mosquitto

۲. سپس به فایل mosquitto.conf دسترسی پیدا می‌کنیم و آن را با یک ویرایشگر باز می‌کنیم. این فایل در سیستم عامل‌های مختلف مسیر متفاوت دارد:

در لینوکس: /etc/mosquitto

در ویندوز: C:\Program Files\mosquitto\

۳. پس از بازگشایی این فایل، دو خط زیر را به آن می‌افزاییم:

```
allow_anonymous true
listener 1883
```

۴. دستور mosquito.c را در خط فرمان وارد می‌کنیم.

۵. سپس آن را مجدداً راه اندازی می‌کنیم:

در لینوکس: sudo systemctl start mosquitto

در ویندوز: sc start mosquitto

²Cloud

³mosquitto.org

در کدی که برای بورد آردوینو نوشتم، ابتدا تنظیمات اتصال به WiFi را انجام و تنظیمات مربوط به اتصال MQTT را با وارد کردن IP سرور و تعیین نام و گذر واژه برای آن انجام داده و پس از مشخص کردن پین‌های مربوط به LED، تعیین می‌کنیم در هر ۱۰ ثانیه یک عدد دو رقمی تولید و از طریق پروتکل موردن بحث به سرور ارسال شود که توسط NodeRed دریافت شده و زوج و فرد بودن آن به اصطلاح مشخص می‌کند که مثلاً فرد موردنظر ما دیابت دارد یا ندارد. این یک مثال است که مدل ساده‌ای را ساخته‌ایم که با فرض اتصال ابزارهای سنجش‌گر قدمخون به صورت آنلاین که مبتنی بر اینترنت اشیاء باشند، می‌توان آخرین اطلاعات فرد بیمار را دریافت کرد.

کد بورد آردوینو:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Network credentials
const char* ssid = "WiFi6";
const char* password = "*****";
const char* mqtt_username = "mqtt";
const char* mqtt_password = "mqtt";

// MQTT broker address
const char* mqtt_server = "192.168.191.3";

// Initializing the WiFi and MQTT clients
WiFiClient DiabetesPredictor20231;
PubSubClient client(DiabetesPredictor20231);

void setup() {
    // Serial communication for debugging purposes
    pinMode(LED_BUILTIN, OUTPUT);
    digitalWrite(LED_BUILTIN, HIGH);
    Serial.begin(9600);

    // Connecting to Wi-Fi
    Serial.println();
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
}
```

```

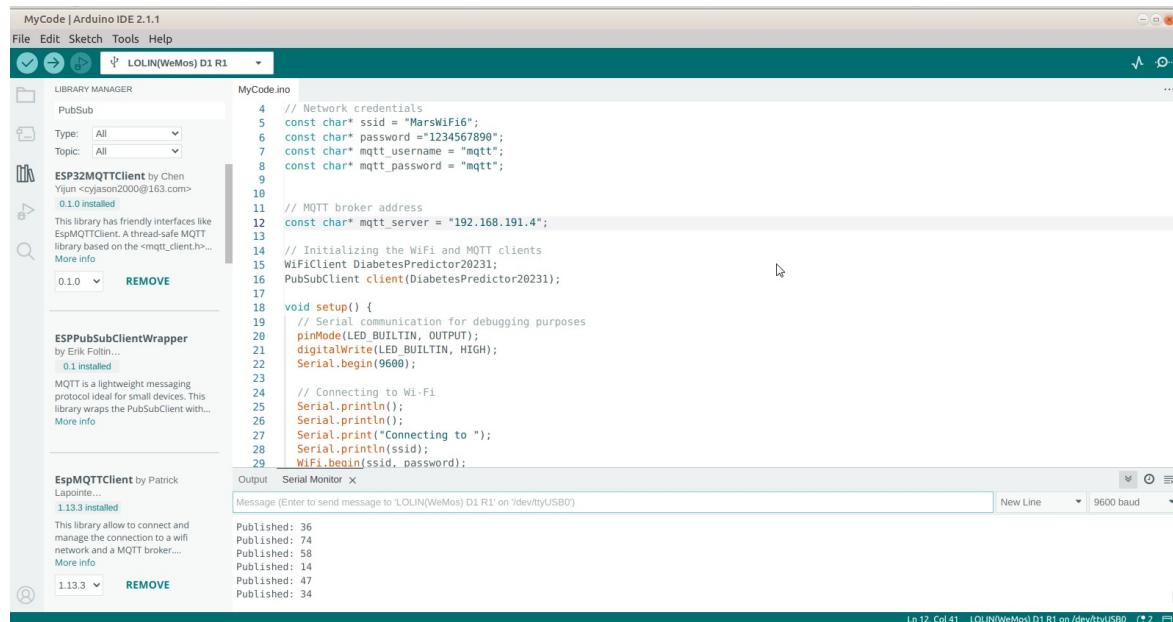
Serial.println("");
Serial.println("WiFi connected");
Serial.println(WiFi.localIP());
Serial.println('\n');

// Connecting to MQTT broker
client.setServer(mqtt_server, 1883);
while (!client.connected()) {
    Serial.print("Connecting to MQTT broker...");
    if (client.connect("DiabetesPredictor2023", mqtt_username, mqtt_password)) {
        Serial.println("connected");
        //client.subscribe("GetData2023", 2);
    } else {
        Serial.print("failed with state ");
        Serial.print(client.state());
        Serial.print("\n");
        delay(2000);
    }
}
void loop() {
    // Generating a two-digit random number and convert it to a string
    int random_num = random(10, 100);
    String payload = String(random_num);

    // Publishing the payload to the MQTT topic
    client.publish("GetData2023", payload.c_str());
    digitalWrite(LED_BUILTIN, LOW);
    Serial.print("Published: ");
    Serial.println(payload);
    delay(400);
    digitalWrite(LED_BUILTIN, HIGH);

    // Printing the payload and wait for 10 seconds before publishing again
    delay(10000);}
```

ادامه توسعه این پروژه در بخش پیانی مطرح می‌شود.



شکل ۱۵-۴: محیط توسعه Arduino در حال دریافت اعداد تولید شده توسط بورد

۳-۲-۴ ابزار Node Red

ابزار Node-RED یک چارچوب نرم افزاری مبتنی بر جریان^۴ همه کاره و کاربرپسند^۵ است که توسط IBM توسعه یافته است که توسعه اینترنت اشیاء را ساده می‌کند. این چارچوب مبتنی بر Node.js است و امکان ادغام یکپارچه با دستگاه‌های مختلف، API‌ها و انواع خدمات آنلاین را فراهم می‌کند. با رابط بصری جذاب آن، کاربران می‌توانند به سادگی با کشیدن و رها کردن، گره‌هایی را روی یک صفحه ایجاد و به هم متصل کنند تا رفتار مورد نظر برنامه خود را تعریف کنند [۳۴]. این ابزار با ادغام یکپارچه با پلتفرم‌هایی مانند AWS، Arduino، Raspberry Pi، ضمن سادگی و تطبیق پذیری عالی، آن را به گرینه‌ای محبوب برای پروژه‌های اینترنت اشیاء و برنامه‌های اتوماسیون خانگی تبدیل کرده است [۳۴].

همچنین نصب داشبورد در این بستر، با وارد کردن دستور `npm install node-red-dashboard` در خط فرمان انجام می‌شود.

در این ابزار، یک داشبورد (شکل ۱۶-۴) طراحی کردیم که دو بخش دارد.

^۴Flow-Based

^۵User friendly

^۶Node

Prediction of diabetes

Prediction		Update	
AgeDecade	10-19	AgeDecade	Select option
Race	White	Race	Select option
Poverty	2	Poverty	
Work	NotWorking	Work	Select option
DirectChol	23	DirectChol	
TotChol	36	TotChol	
HealthGen	Fair	HealthGen	Select option
Depressed	Several	Depressed	Select option
SleepTrouble	No	SleepTrouble	Select option
SmokeNow	No	SmokeNow	Select option
HardDrugs	No	HardDrugs	Select option
SameSex	No	SameSex	Select option
SexOrientation	Heterosexual	SexOrientation	Select option
PregnantNow	No	PregnantNow	Select option
SEND		Diabetes	Select option
Result	You may develop diabetes!	GET THE LATEST PATIENT STATUS	
		SEND	

شکل ۱۶-۴: صفحه داشبورد پیش‌بینی کاربر و بهروزرسانی دادگان

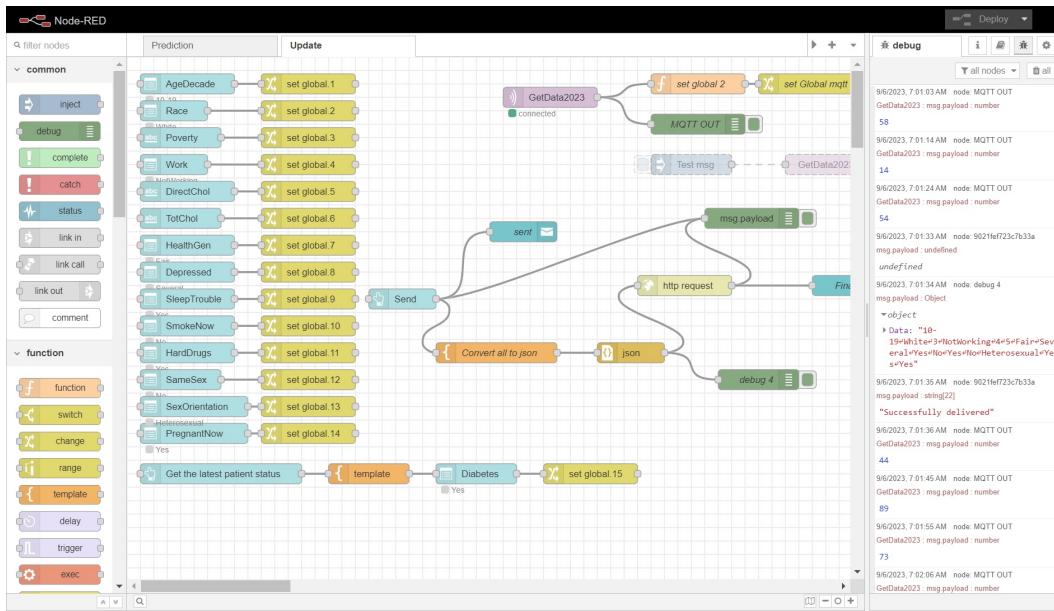
• بهروزرسانی دادگان:

در این بخش داده‌های جدید از یک سامانه دیگر دریافت شده و از طریق چارچوب Flask در فایل csv دادگان اضافه می‌شوند. بدیهیست که گزینه GET THE LAST PATIENT STATUS آخرين وضعیت بیمار را درمورد داشتن یا نداشتن دیابت را از طریق اینترنت اشیاء دریافت می‌کند. بدیهیست داشتن دادگان به روز، به صحت مدل و افزایش دقت آن، کمک شایانی می‌کند.

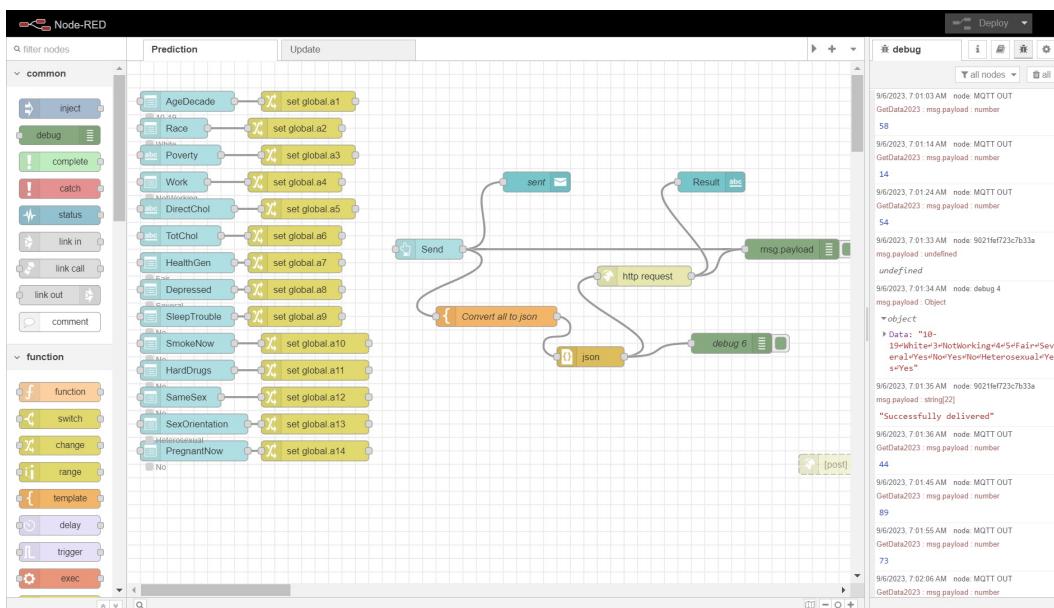
• پیش‌بینی: در این بخش، افراد با وارد کردن ویژگی‌های سلامتی خود، می‌توانند بینند که آیا در آینده ممکن است به دیابت مبتلا شوند یا خیر.

در هر کدام از بخش‌های مطرح شده، گره‌های مربوط به دریافت مشخصات را ایجاد و مقادیر از طریق مرورگر دریافت و در آن ثبت می‌شود. سپس در ساختار داده json ثبت شده و در نهایت از طریق یک request برای Flask ارسال می‌شود.

بعد از دریافت صحیح اطلاعات، یک اعلان درمورد دریافت صحیح داده‌ها نیز دریافت می‌شود.



شکل ۱۷-۴: طراحی بخش بهروزرسانی در NodeRed



شکل ۱۸-۴: طراحی بخش پیش‌بینی در NodeRed

Flask ۴-۲-۴

می‌توان گفت Flask یک چارچوب توسعه وب سبک و کاربرپسند است که به زبان پایتون نوشته شده است. این سادگی، انعطاف‌پذیری و ساختار مدولار را برای ساخت برنامه‌های کاربردی مقیاس پذیر ارائه می‌دهد. مستندات گسترده و جامعه پر جنب و جوش این چارچوب، یادگیری و استفاده از آن را آسان می‌کند. علاوه بر این، Flask به خوبی با کتابخانه‌ها و ابزارهای دیگر ادغام می‌شود و از پایگاه داده‌های مختلف و فناوری‌های وب پشتیبانی و پردازش انواع درخواست‌ها را آسان می‌کند. این یک

انتخاب ارجح برای توسعه دهنده‌گان با تمام سطوح تجربه است و به آنها اجازه می‌دهد تا به طور کارآمد برنامه‌های کاربردی وب قوی بسازند [۳۷].

لذا در این پروژه از این چارچوب برای استفاده از توابع و کتابخانه‌های پایتون جهت پردازش درخواست‌های کاربران استفاده کردیم. وقتی از طریق داشبورد در NodeRed یک درخواست مبتنی بر به روز رسانی داده‌های دادگان یا درخواست از طرف یک فرد درمورد پیش‌بینی احتمال ابتلای آن فرد با مشخصاتش در چارچوب Flask دریافت شود، در تابع مربوطه این درخواست مورد پردازش قرار می‌گیرد و نتیجه به سادگی برای کاربر ارسال می‌شود.

```

88     # Selecting X & Y
89     X = dataset_new.iloc[:, :-1].values
90     Y = dataset_new.iloc[:, -1].values
91
92     # Splitting the dataset into the Training set and Test set
93     from sklearn.model_selection import train_test_split
94     XTrain, XTest, YTrain, YTest = train_test_split(X,Y, test_size = 0.25, random_state = 0)
95
96     #print(dataset_new.columns)
97
98     from sklearn.linear_model import LogisticRegression
99     logreg = LogisticRegression()
100    #print(XTest[0])
101    logreg.fit(XTrain, YTrain)
102    logreg_pred=logreg.predict(XTest)
103
104    YP_ranfor = logreg.predict(XTest)
105
106    TempX=x
107    TempXnpArr=np.array(TempX)
108    TempXnpArr=TempXnpArr.reshape(1,-1)
109    Answer=logreg.predict(TempXnpArr)
110
111    if Answer[0]==1:
112        print("You may develop diabetes!")
113        return 'You may develop diabetes!'
114    else:
115        print('The chances of you developing diabetes are very low')
116        return 'The chances of you developing diabetes are very low'
117
118    if __name__ == '__main__':
119        app.run()

```

Ln 54, Col 50 Spaces: 4 UTF-8 CRLF Python 3.9.13 64-bit

شکل ۱۹-۴: بخشی از کد Flask

۳-۴ نتیجه‌گیری

در این بخش، کوشیدیم تا ضمن بررسی روش انتخاب بهترین الگوریتم پیش‌بینی دیابت با پایتون و سنجش میزان صحت و دقت روش‌ها، یک روش پیاده‌سازی مختصر نیز برای استفاده همگان از این سیستم معرفی شود.

این سنجش دقت از طریق مقایسه ROC، AUC و نمودارهای آن‌ها انجام شد که در فصل قبلی نیز مقالاتی که ارائه شده بودند نیز بر این اساس مقایسه شدند که نتیجه کلی آن روش جنگل درختان تصادفی بود که به عنوان مدل برتر برگزیده شد.

بدترین عملکرد مربوط به الگوریتم‌های بیز ساده و درخت تصمیم بود که مطابق نمودارهای مشاهده شده عملکرد پایین‌تری نسبت به سایرین داشتند.

و در نهایت به بررسی روش طراحی یک مدل ساده برای سامانه آنلاین پیش‌بینی دیابت با ابزارهای اینترنت اشیاء و دو چارچوب Flask و NodeRed پرداختیم.

فصل ۵

جمع‌بندی و کارهای آتی

۱-۵ جمع‌بندی

بر هیچ کس پوشیده نیست که پیشگیری بهتر از درمان است. لذا با استفاده از راهکارهای علمی، توانستیم مدلی را طراحی کنیم تا با دقت خوبی برای افراد جامعه، بتواند پیش‌بینی کند که آیا در آینده به دیابت مبتلا می‌شوند یا خیر تا در صورت لزوم، اقدامات پیش‌گیرانه را سریع‌تر آغاز کنند.

در نتیجه انجام فرایندهای بسیار، مطابق نمودارهای ایجاد شده در فصل قبلی، الگوریتم جنگل درختان تصادفی در اعتبارسنجی متقابل با $K=10$ Fold، $K=10$ برای ما بهترین نتیجه را با میزان حدودی $ACC=96\%$ در این پیش‌بینی به ارمغان آورد که نشان می‌دهد دفعات بیشتری که الگوریتم‌های اعتبارسنجی متقابل مدل ما را مورد آزمایش قرار دهند، دقت ارزیابی بالاتری برایمان به دنبال خواهد داشت.

لازم به ذکر است از طریق بررسی عملکرد با شاخصه AUC پس از الگوریتم جنگل درختان تصادفی، نمودارهای رگرسیون لجستیک و AdaBoost تقریباً در یک عرض قرار گرفتند.

البته به خوبی مشخص است که هرچه دفعات انجام الگوریتم‌های اعتبارسنجی متقابل بیشتر باشد، زمان و هزینه کار نیز بیشتر خواهد بود.

۲-۵ کارهای آتی

در مورد روش‌های توسعه این سیستم می‌توانیم به مواردی چون سیستم‌های یادگیری ماشین آنلاین اشاره کنیم که دادگان همواره با اطلاعات جدید به روزرسانی می‌شوند و در بازه‌های مختلف توسط ناظر سیستم، بهترین الگوریتم‌ها بر رویشان اعمال می‌گردد تا بهترین نتایج ارائه شوند. ضمن اینکه می‌توانیم از اینترنت اشیاء و امکاناتی از این قبیل استفاده کنیم. به تازگی گجت‌ها و کیت‌های مخصوصی برای گوشی‌های هوشمند طراحی شده اند که برای سنجش پارامترهای گوناگون سلامتی مورد استفاده قرار می‌گیرند و با استفاده از گسترش شبکه‌های پرسرعت اینترنت نظیر 5G به سرعت می‌توانیم حجم عظیمی از داده‌های جدید را دریافت کنیم.

به عنوان یک نمونه ساده و سمبولیک می‌توان از برد الکترونیکی آردوبینو که قابلیت نصب گجت‌های

مختلف و سنسورهای گوناگون را فراهم می‌آورد، استفاده کرد و یک سیستم یادگیری ماشین آنلاین را طراحی کرد که با سرور ما (مثلا برای آردوینو NodeRed می‌باشد) تبادل دارد.

همچنین می‌توانیم سامانه‌های موازی با این سیستم را نیز راه اندازی کرد مثل یک سایت پیش‌بینی‌کننده احتمال ابتلا به بیماری دیابت در افراد که هر شخصی با وارد کردن پارامترهای خودش می‌تواند نسبت به وضعیت سلامتی خود در آینده برآورده تقریبی داشته باشد.

پیوست

۱- کد استانداردسازی متون فارسی آمیخته به عبارات انگلیسی

با توجه به اینکه این پایان نامه در LATEX نوشته شده شد، یکی از معضلات مربوط به تایپ فارسی و انگلیسی در این سیستم، این است که حتماً عبارات انگلیسی بایستی درون تگ lr نوشته شوند تا فونت تعیین شده بر روی آن اعمال شود. پس یک برنامه ساده به زبان پایتون نوشتیم تا عبارات انگلیسی درون متن های فارسی که را برايمان در این تگ قرار دهد.

در این کد پایتونی از مبحث RegEx که در درس کامپایلر با آن آشنا شدیم، استفاده کردم. فایل ورودی متن عادی است که پس از انجام عملیات، در فایل خروجی شاهد قرار گرفتن عبارات انگلیسی درون تگ lr خواهیم بود.

```
import re
filename='import.txt'
lines=[]
with open(filename) as file:
    lines = [str(line.rstrip()) for line in file]

def myfun(a):
    e=a.group(0)
    e=' \lr{'+e+'}'
    return e

w=[]
for i in lines:
    x =re.findall(r"[a-zA-Z0-9\s]+\b(?=\\s[^a-zA-Z0-9]*)", i)
    tempi=re.sub(r"[a-zA-Z0-9\s]+\b(?=\\s[^a-zA-Z0-9]*)", myfun, i)
    w.append(tempi)
    print(tempi)

with open(r'export_text.txt', 'w+') as fp:
    for item in w:
        fp.write("%s\n" % item)
print('Done')
```

واژه‌نامه

ACC(Accuracy)	دقت		الف
Dummy	ساختگی	Specificity	اختصاصی
Sigmoid	سیگماوار	Validation	اعتبارسنجی
		Split	افراز
		Confusion	آشتفتگی
BMI	شاخص توده بدنی	Naïve Bayes	بیز ساده
Classification	طبقه‌بندی	Overfitting	بیش‌برازش
		PIDD	بیماری‌های زمینه‌ای
FrameWork	چارچوب	Preprocessing	پیش‌پردازش
PhysActive	فعالیت فیزیکی		پ
TotChol	کلسترول کل		
DirectChol	کلسترول مستقیم	Random forest	ج
Underfitting	کم‌برازش		جنگل تصادفی
Scatter matrix	ماتریس پراکندگی	Sensitivity	حساسیت
FP	ثبت کاذب		ح
TP	ثبت واقعی		
IDE	محیط توسعه	Dataset	دادگان
Mode	مد (آمار)	NaN	داده نامشخص
Visualization	تصویرسازی	Training data	داده‌های آموزشی
	منحنی مشخصه	Testing data	داده‌های سنجش
		Missing data	داده‌های گم شده

TPR	نرخ مثبت واقعی	ROC	عملکرد سیستم
Normalization	نرمال سازی	FN	منفی کاذب
Race	نژاد	TN	منفی واقعی
Heatmap	نقشه حرارتی	Mean	میانگین
Lineplot	نمودار میله‌ای	BPDiaAve	میانگین فشار خون
	و		دیاستولیک
MaritalStatus	وضعیت تاہل	BPSysAve	میانگین فشار خون
Feature	ویژگی		سیستولیک
			ن
		AUC	ناحیه زیر منحنی
		FPR	نرخ مثبت کاذب

مراجع

- [۱] اسماعیلی . مهدی، مفاهیم و تکنیک‌های داده کاوی
- [۲] نعمت الهی . نادر، آمار و احتمالات مهندسی

- [3] What is diabetes?, Aoife M Egan, Sean F Dinneen
- [4] Epidemiology of diabetes,Nita Gandhi Forouh,Nicholas J Wareha
- [5] Diabetes Cookbook For Dummies, by Alan L. Rubin, MD with Cait James, MS
- [6] Classification and prediction of diabetes disease using machine learning paradigm,Md.Maniruzzaman, Md. Jahanur Rahman, Benojir Ahammed and Md. Menhazul Abedin
- [7] Prediction of Diabetes using Classification Algorithms, Deepti Sisodia ,Dilip Singh
- [8] Logistic Regression,Lynne Connelly
- [9] Interpretable Machine Learning,Christoph Molnar
- [10] An Overview of Big Data Visualization Techniques in Data Mining ,Samuel Soma Ajibade, Anthonia Adediran
- [11] Data Mining Concepts and Techniques ,Jiawei Han,Micheline Kamber,Jian Pei
- [12] Data Mining for Business Analytics - Concepts, Techniques and Applications in Python , Galit Shmueli, Peter C. Bruce, Peter Gedeck, Nitin R. Patel
- [13] Top 10 algorithms in data mining,Xindong Wu,Jiannong Cao
- [14] Machine Learning for Predictive Analysis, Amit Joshi, Mahdi Khosravy, Neeraj Gupta
- [15] Cross-validation,Daniel Berrar
- [16] The Working Principle Of An Arduino, Yusuf Abdullahi Badamasi
- [17] <https://www.cdc.gov/diabetes/basics/diabetes.html>
- [18] <https://www.kaggle.com/datasets/cdc/national-health-and-nutrition-examination-survey>
- [19] <https://medium.com/@rithpansanga/logistic-regression-for-feature-selection-selecting-the-right-features-for-your-m>
- [20] <https://towardsdatascience.com/overfitting-vs-underfitting-a-complete-example-d05dd7e19765>

- [21] <https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c>
- [22] <https://www.techtarget.com/searchbusinessanalytics/definition/data-visualizatio>
- [23] https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation
- [24] <https://www.blog.trainindata.com/feature-scaling-in-machine-learning/>
- [25] <https://towardsdatascience.com/laplace-smoothing-in-na%C3%A9AFve-bayes-algorithm-9c237a8bdece>
- [26] <https://scikit-learn.org/stable/modules/tree.html>
- [27] <https://medium.com/udacity/shannon-entropy-information-gain-and-picking-balls-from-buckets-5810d35d54b4>
- [28] <https://www.datacamp.com/tutorial/adaboost-classifier-python>
- [29] <https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/>
- [30] <https://machinelearningmastery.com/confusion-matrix-machine-learning/>
- [31] <https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>
- [32] <https://www.investopedia.com/terms/m/mlr.asp>
- [33] <https://aws.amazon.com/what-is/mqtt/>
- [34] <https://nodered.org/docs/>
- [35] <https://learn.sparkfun.com/tutorials/what-is-an-arduino/all>
- [36] <https://docs.arduino.cc/>
- [37] <https://pythonbasics.org/what-is-Flask-python/>