



دانشگاه شهرستان

دانشکده برق و کامپیوتر

پایان نامه کارشناسی مهندسی کامپیوتر

ساخت مدل یادگیری ماشین برای دسته بندی موارد سرطان

سینه به دو دسته ملایم و بد خیم

نگارش:

امیرحسین باقریان کلات

استاد راهنما:

دکتر دری گیو

تابستان ۱۴۰۲

الله اکبر جناب

چکیده

در این پایان نامه سعی شده با بهره‌گیری از مدل‌های تقویت گرادیان مفرط و ماشین افزایش گرادیان سبک و با تمرکز بر تنظیم پارامترهای آنها جهت بهبود عملکرد، طبقه‌بندی بیماران مبتلا به سرطان سینه را در دو دسته ملایم و بدخیم با دقت بالایی انجام دهیم.

در این پایان‌نامه، پژوهشگر با بهره‌گیری از منابع مختلف و آشنایی با پارامترهای مهم این دو مدل اقدام به ایجاد و هرس مجموعه فضای جستجو جهت غلبه بر چالش زمان یادگیری طولانی داشته است.

با تلاش‌های صورت گرفته، دقت مدل‌های استفاده شده در این پایان‌نامه از حالت پیش‌فرض به میزان قابل توجهی بهبود یافته‌است و با بهره‌گیری از روش اعتبارسنجی متقابله چند دسته‌ای این بهبود عملکرد را به نمایش گذاشته است.

علاوه بر روش اعتبارسنجی متقابله، با به دست آوردن ماتریس در هم‌ریختگی، امتیاز صحت، مقادیر دقت، حساسیت، اختصاصیت و صحت نیز برای هر مدل بهبود یافته محاسبه شده‌است. همچنین نمودارهای منحنی ویژگی عملکرد گیرنده (راک) و منحنی دقت-حساسیت برای مدل‌های نهایی بهبود یافته ترسیم شده است.

برای یافتن تاثیر هر ویژگی در پیش‌بینی نهایی، از نمودارهای مقادیر توضیحات افزونی شپلی استفاده شده است که در هر نمودار اطلاعاتی در مورد وضعیت عملکرد مدل مربوطه قابل مشاهده است. در فصل پایانی با استفاده از فریم‌ورک آپتوна، تلاش برای بهبود عملکرد مدل ماشین افزایش گرادیان سبک به روشهای خودکار و سریع انجام گرفته است و اطلاعات مربوط به ارزیابی مدل آموزش دیده به وسیله پارامترهای به دست آمده توسط این فریم‌ورک آورده شده است.

کلیدواژگان: مدل تقویت گرادیان مفرط؛ مدل ماشین افزایش گرادیان سبک؛ اعتبارسنجی متقابله؛ منحنی ویژگی عملکرد گیرنده (راک)؛ منحنی دقت-حساسیت؛ نمودار مقادیر توضیحات افزونی شپلی؛ آپتونا

فهرست مطالب

۱۰	۱	مقدمه
۱۱	۱-۱	هدف پژوهه
۱۱	۲-۱	کاربرد پژوهه
۱۱	۳-۱	مشخصات کلی سیستم طراحی شده
۱۲	۲	مبانی تئوری
۱۳	۱-۲	هوش مصنوعی
۱۳	۲-۲	یادگیری ماشین
۱۴	۳-۲	یادگیری ناظارت شده
۱۴	۴-۲	درخت تصمیم
۱۵	۵-۲	یادگیری گروهی
۱۵	۶-۲	تقویت گرادیان (شیب)
۱۶	۷-۲	تقویت گرادیان (شیب) مفرط
۱۷	۸-۲	اعتبارسنجی متقابل
۱۸	۹-۲	مقادیر توضیحات افزونی شپلی
۱۹	۱۰-۲	ماتریس درهم ریختگی
۱۹	۱۱-۲	پارامترهای ارزیابی نتایج مدل
۱۹	دقت	
۱۹	حساسیت	
۱۹	اختصاصیت	
۱۹	صحت	
۲۰	$F\beta$	امتیاز
۲۰	$F1$	امتیاز
۲۰	۱۲-۲	نمودارهای ارزیابی نتایج مدل
۲۰	منحنی ویژگی عملکرد (راک)	
۲۱	منحنی دقتشناسیت	
۲۱	۱۳-۲	فریمورک آپتونا
۲۲	۳	پیادهسازی مدل تقویت گرادیان مفرط (XGBoost) با پارامترهای پیشفرض
۲۳	۱-۳	اضافه کردن کتابخانه ها و مجموعه داده مربوطه

۲۳	استخراج متغیرهای مستقل و وابسته	۲ - ۳
۲۳	ارزیابی تاثیر ستون‌های مختلف در پیش‌بینی نهایی	۳ - ۳
۲۴	پیاده‌سازی ضریب همبستگی پیرسون روی مجموعه داده	۴ - ۳
۳۰	حذف ستون‌هایی با تاثیر ناچیز بر نتیجه نهایی از مجموعه متغیرهای مستقل	۵ - ۳
۳۰	رمزگذاری متغیر وابسته	۶ - ۳
۳۱	تقسیم مجموعه داده به مجموعه آموزشی و مجموعه آزمایشی	۷ - ۳
۳۱	آموزش مدل تقویت گرادیان مفرط با داده‌های مجموعه آموزش	۸ - ۳
۳۲	ارزیابی مدل	۹ - ۳
۳۲	۱ ارزیابی با ماتریس درهم ریختگی و محاسبه امتیاز صحت	۳ - ۹
۳۲	۲ ارزیابی با اعتبارسنجی متقابل چند دسته‌ای	۳ - ۹
۳۴	بررسی پارامترهای مدل برای تنظیم آنها	۴
۳۵	۱ - انواع پارامترها	۴ - ۴
۳۵	۲ - نرخ یادگیری	۴ - ۴
۳۵	۳ - حداقل عمق	۴ - ۴
۳۵	۴ - تعداد برآوردها	۴ - ۴
۳۶	۵ - گاما	۴ - ۴
۳۶	۶ - زیرنمونه	۴ - ۴
۳۶	۷ - نمونه ستون برای هر درخت	۴ - ۴
۳۶	۸ - آلفا	۴ - ۴
۳۷	۹ - لاندا	۴ - ۴
۳۷	۱۰ - حالت تصادفی	۴ - ۴
۳۸	پیاده‌سازی مدل تقویت گرادیان مفرط (XGBoost) با پارامترهای تنظیم شده	۵
۳۹	۱ - آزمایش پارامترهای مختلف برای بهینه‌سازی مدل	۵ - ۵
۴۰	۲ - تخمین زمان اجرای تکه کد ذکر شده	۵ - ۵
۴۰	۳ - محاسبه مقدار بهینه برای تعداد برآوردها و نرخ یادگیری	۵ - ۵
۴۱	۴ - آموزش مجدد مدل با پارامترهای بهینه	۵ - ۵
۴۲	۵ - ارزیابی مدل	۵ - ۵
۴۲	۱ اعمال اعتبارسنجی متقابل چند دسته‌ای	۵ - ۵
۴۲	۲ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت	۵ - ۵

- ۴۲ ۵ - ۳ نمایش ماتریس در هم ریختگی به همراه برچسب
۴۳ ۵ - ۴ گزارش دسته‌بندی
- ۴۵ ۵ - ۵ محاسبه معیارهای اختصاصیت و حساسیت
- ۴۵ ۶ - ۵ بهبود عملکرد مدل با تنظیم پارامترهای گاما، زیرنمونه، نمونه ستون برای هر درخت و حداقل عمق
- ۴۶ ۷ - ۵ ارزیابی مدل
- ۴۶ ۱ - ۷ اعمال اعتبار سنجی مقابله چند دسته‌ای
- ۴۶ ۲ - ۷ محاسبه ماتریس در هم ریختگی و امتیاز صحت
- ۴۷ ۳ - ۷ نمایش ماتریس در هم ریختگی به همراه برچسب
- ۴۸ ۴ - ۷ گزارش دسته‌بندی
- ۴۹ ۵ - ۷ محاسبه معیارهای اختصاصیت و حساسیت
- ۵۰ ۶ - ۷ رسم منحنی ویژگی عملکرد گیرنده (راک)
- ۵۲ ۷ - ۷ رسم نمودار منحنی دقت-حساسیت
- ۵۳ ۸ - ۵ رسم نمودار مقادیر توضیحات افزونی شپلی
- ۵۴ ۱ - ۸ رسم نمودار آبشاری
- ۵۵ ۲ - ۸ رسم نمودار نیرو
- ۵۶ ۳ - ۸ رسم نمودار نیرو به صورت پشتنه
- ۵۷ ۴ - ۸ نمودار میله‌ای
- ۵۸ ۵ - ۸ نمودار ازدحام زنبورها
- ۶۱ ۶ پیاده‌سازی مدل ماشین افزایش گرادیان سبک (LGBM) با پارامترهای پیش‌فرض
- ۶۲ ۱ - ۶ آموزش مدل ماشین افزایش گرادیان سبک با داده‌های مجموعه آموزش
- ۶۲ ۲ - ۶ ارزیابی مدل
- ۶۲ ۱ - ۶ ۱ - ۲ ارزیابی مدل با استفاده از ماتریس در هم ریختگی و امتیاز صحت
- ۶۲ ۲ - ۶ اعمال اعتبار سنجی مقابله چند دسته‌ای
- ۶۴ ۷ پیاده‌سازی مدل ماشین افزایش گرادیان سبک (LGBM) با پارامترهای تنظیم‌شده
- ۶۵ ۱ - ۷ آزمایش پارامترهای مختلف در آموزش مدل
- ۶۵ ۲ - ۷ تخمین زمان اجرای تکه کد ذکر شده
- ۶۶ ۳ - ۷ محاسبه مقدار بهینه برای تعداد دور و نرخ یادگیری
- ۶۷ ۴ - ۷ آموزش مجدد مدل با پارامترهای بهینه
- ۶۷ ۵ - ۷ ارزیابی مدل
- ۶۷ ۱ - ۵ ارزیابی مدل با اعتبار سنجی مقابله چند دسته‌ای
- ۶۷ ۲ - ۵ ارزیابی مدل با ماتریس در هم ریختگی و امتیاز صحت
- ۶۸ ۳ - ۵ نمایش ماتریس در هم ریختگی به همراه برچسب

۶۹	۴ - ۵ - ۷گزارش دسته‌بندی
۷۰	۵ - ۵محاسبه معیارهای اختصاصیت و حساسیت
۷۱	۶ - ۶بهبود عملکرد مدل با تنظیم پارامترهای زیرنمونه، نمونه ستون برای هر درخت و حداقل داده در برگ ارزیابی مدل
۷۲	۷ - ۷ارزیابی مدل با اعتبار سنجی مقابله چند دسته‌ای
۷۳	۷ - ۷۱ - ۷ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت
۷۴	۷ - ۷۲ - ۷ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت
۷۵	۷ - ۷۳ - ۷نمایش ماتریس درهم ریختگی به همراه برچسب
۷۶	۷ - ۷۴ - ۷گزارش دسته‌بندی
۷۷	۷ - ۷۵ - ۷محاسبه معیارهای اختصاصیت و حساسیت
۷۸	۷ - ۷۶ - ۷رسم منحنی ویژگی عملکرد گیرنده (راک)
۷۹	۷ - ۷۷ - ۷رسم نمودار منحنی دقت-حساسیت
۸۰	۷ - ۷رسم نمودار منحنی دقت-حساسیت
۸۱	۷ - ۷رسم نمودار مقادیر توضیحات افزونی شپلی
۸۲	۸ - ۷۱ - ۸رسم نمودار آبشاری
۸۳	۸ - ۷۲ - ۸رسم نمودار نیرو
۸۴	۸ - ۷۳ - ۸رسم نمودار نیرو به صورت پشته
۸۵	۸ - ۷۴ - ۸نمودار میله‌ای
۸۶	۸ - ۷۵ - ۸نمودار ازدحام زنبورها
۸۷	۸تنظیم مدل ماشین افزایش گرادیان سبک (LGBM) با کمک فریم‌ورک آپتوна
۸۸	۱ - ۸اضافه کردن کتابخانه‌ها و مجموعه داده مربوطه
۸۹	۲ - ۸تعریف تابع هدف
۹۰	۳ - ۸ارزیابی مدل
۹۱	۴ - ۳۱ - ۸ارزیابی با اعتبار سنجی مقابله
۹۲	۴ - ۳۲ - ۸ارزیابی با ماتریس درهم ریختگی و امتیاز صحت
۹۳	۴ - ۳۳ - ۸نمایش ماتریس درهم ریختگی به همراه برچسب
۹۴	۴ - ۳۴ - ۸گزارش دسته‌بندی
۹۵	۴ - ۳۵ - ۸محاسبه معیارهای اختصاصیت و حساسیت
۹۶	۴ - ۳۶ - ۸رسم منحنی ویژگی عملکرد گیرنده (راک)
۹۷	۴ - ۳۷ - ۸رسم نمودار منحنی دقت-حساسیت
۹۸	۴ - ۸رسم نمودار مقادیر توضیحات افزونی شپلی
۹۹	۴ - ۸۱ - ۴رسم نمودار آبشاری
۱۰۰	۴ - ۸۲ - ۴رسم نمودار نیرو
۱۰۱	۴ - ۸۳ - ۴رسم نمودار نیرو به صورت پشته
۱۰۲	۴ - ۸۴ - ۴نمودار میله‌ای

۹۶	۵ - ۴ نمودار ازدحام زنیبورها
۹۸	۹ جمع‌بندی و پیشنهادها
۹۹	۱ - ۹ جمع‌بندی
۹۹	۲ - ۹ نتیجه‌گیری
۹۹	۳ - ۹ پیشنهادها
۱۰۱	منابع و مراجع

فصل اول

۱ مقدمه

در این فصل به توضیح مختصری درباره موضوع پایان نامه به نام ساخت مدل یادگیری ماشین برای دسته بندی موارد سرطان سینه به دو دسته ملایم و بد خیم می پردازیم و هدف کلی پروژه را به اختصار بیان می کنیم.

در این فصل سعی شده طرح موضوع به گونه ای ساده و روان بیان گردد.

۱-۱ هدف پروژه

بیماری سرطان امروزه یکی از بیماری‌های شایع در جوامع امروز است. در یک نوع از این بیماری که مرتبط با بافت‌های سینه است، عوامل بسیاری همچون نقاط مقعر^۱، محیط پیرامونی^۲، ابعاد فرکتال^۳ و موارد دیگر تاثیر بسزایی دارد. در این پروژه با استفاده از تکنولوژی هوش مصنوعی یکی از زیر مجموعه‌های اصلی آن که یادگیری ماشین است؛ سعی شده مدلی آموزش داده شود که با ورودی گرفتن داده‌های پزشکی بیمار، بیمار را در یکی از دو دسته ملایم و یا بدخیم طبقه‌بندی کند.

۱-۲ کاربرد پروژه

امروزه با افزایش جمعیت جهانی و تعدد بیماران مختلف امکان معاينه تمامی افراد برای تشخیص وضعیت بیماری آنها به علت محدودیت منابع پزشکی در عمل وجود ندارد. در دنیای امروز ناگزیر هستیم برای ارائه خدمات بهتر چه در زمینه پزشکی و چه در زمینه‌های دیگر از تکنولوژی هوش مصنوعی استفاده کنیم. تکنولوژی هوش مصنوعی این امکان را به ما می‌دهد که صرفاً با وارد کردن اطلاعات بیمار و در مدت زمان بسیار کوتاهی بتوانیم بیماری شخص را با دقت بالایی تشخیص داده بیمار را در طبقه‌بندی مناسب از نظر پزشکی قرار دهیم.

۱-۳ مشخصات کلی سیستم طراحی شده

- دقت بسیار بالا
- استفاده از فاکتورهای متعدد پزشکی در آموزش مدل ماشین برای تشخیص بیماری
- استفاده از روش اعتبارسنجی نتیجه در هنگام استفاده از مدل ماشین
- سادگی استفاده از سیستم برای افراد مبتدی

¹ concave points

² perimeter

³ fractal dimension

فصل دوم

۲ مبانی تئوری

در این فصل به مرور مختصری بر مبانی تئوری مرتبط با موضوع پایان نامه پرداخته شده، سعی می‌شود مقدمات علمی بحث برای افرادی که آشنایی چندانی با این قبیل موضوعات ندارند مطرح گردد. در ابتدا در مورد هوش مصنوعی و یادگیری ماشینی به اختصار توضیح داده شده است و پس از آن به توضیحات پیرامون الگوریتم‌های مرتبط با موضوع پروژه پرداخته شده است. لینک مطالب اشاره شده در اکثر توضیحات در قسمت منابع و مراجع قرار داده شده است و علاقمندان می‌توانند برای مطالعه بیشتر به آنها مراجعه کنند.

۲-۱ هوش مصنوعی

هوش مصنوعی شبیه‌سازی فرآیندهای هوش انسانی توسط ماشین‌ها به ویژه سیستم‌های کامپیوتری است. کاربردهای خاص هوش مصنوعی شامل سیستم‌های خبره، پردازش زبان طبیعی، تشخیص گفتار و بینایی ماشین است.

به طور کلی، سیستم‌های هوش مصنوعی با دریافت مقادیر زیادی از داده‌های آموزشی برچسب‌گذاری شده، داده‌ها را برای یافتن همبستگی‌ها و الگوها تجزیه و تحلیل کرده و از این الگوها برای پیش‌بینی وضعیت‌های آینده استفاده می‌کنند. به این ترتیب، یک چتبات که با نمونه‌هایی از متن تغذیه می‌شود، می‌تواند یاد بگیرد که تبادلات واقعی با افراد ایجاد کند، یا یک ابزار تشخیص تصویر می‌تواند با مرور میلیون‌ها تصویر، شناسایی و توصیف اشیاء در تصاویر را بیاموزد. تکنیک‌های جدید هوش مصنوعی که به سرعت در حال بهبود هستند می‌توانند متن، تصاویر، موسیقی و سایر رسانه‌های واقعی را خلق کنند [1].

۲-۲ یادگیری ماشین

یادگیری ماشینی زیرمجموعه‌ای از هوش مصنوعی است که بر توسعه الگوریتم‌ها و مدل‌های آماری تمرکز دارد و رایانه‌ها را قادر می‌سازد تا بدون برنامه‌ریزی صریح عملکرد خود را در یک کار خاص یاد بگیرند و بهبود بخشنند. ایده اساسی در پشت یادگیری ماشینی این است که به ماشین‌ها اجازه می‌دهیم از داده‌ها و تجربیات بیاموزند و برای انها این شرایط را فراهم کنیم که پیش‌بینی کنند، الگوها را تشخیص دهند و بر اساس الگوهای کشف شده در داده‌ها تصمیم بگیرند. این فرآیند شبیه به نحوه یادگیری انسان از تجربه و تطبیق رفتار خود بر اساس آن است.

این تکنولوژی بر پایه‌ی توانایی شناسایی الگوها و روابط در مجموعه داده‌های بزرگ است. فرآیند یادگیری معمولاً شامل تغذیه الگوریتم با داده‌های آموزشی است که شامل جفت‌های ورودی-خروجی است. سپس الگوریتم به طور مکرر پارامترهای داخلی خود را تنظیم می‌کند تا خطای اختلاف بین پیش‌بینی‌های خود و خروجی‌های واقعی در داده‌های آموزشی را به حداقل برساند. این فرآیند مداوم که به عنوان آموزش یا یادگیری از آن یاد می‌شود، به الگوریتم اجازه می‌دهد تا عملکرد خود را در طول زمان بهبود بخشد و به داده‌های دیگری که در آینده بعنوان ورودی دریافت می‌کند تعمیم دهد.

همچنین از یادگیری ماشینی در حوزه‌های مختلفی از جمله پردازش زبان طبیعی، بینایی کامپیوتر، سیستم‌های توصیه، تشخیص پزشکی و پیش‌بینی مالی استفاده می‌شود. با افزایش حجم داده‌ها و بهبود قدرت محاسباتی، یادگیری ماشین همچنان نقشی محوری در خودکارسازی وظایف^۱، پیش‌بینی‌ها و ایجاد پیشرفت در صنایع مختلف ایفا می‌کند.

^۱ automating tasks

۲-۳ یادگیری نظارت شده^۱

یادگیری نظارت شده که به عنوان یادگیری ماشین نظارت شده نیز شناخته می‌شود، زیر مجموعه‌ای از یادگیری ماشین و هوش مصنوعی است. یادگیری نظارت شده از یک مجموعه آموزشی برای آموزش مدل‌ها برای به دست آوردن خروجی مطلوب استفاده می‌کند. این مجموعه داده آموزشی شامل ورودی‌ها و خروجی‌های صحیح است که به مدل اجازه می‌دهد در طول زمان یاد بگیرد. این الگوریتم دقیق‌تر را از طریق تابع ضرر^۲ اندازه‌گیری می‌کند و تا زمانی که خطا به اندازه کافی به حداقل برسد، وزن‌ها را تنظیم می‌کند، که این کار به عنوان بخشی از فرآیند اعتبارسنجی متقابل^۳ انجام می‌شود. یادگیری تحت نظارت به سازمانها کمک می‌کند تا انواع مشکلات دنیای واقعی را در مقیاس بزرگ حل کنند، مانند طبقه‌بندی هرزنامه‌ها در یک پوشه جدایانه از صندوق ورودی ایمیل‌ها. یادگیری نظارت شده به دو دسته کلی طبقه‌بندی^۴ و رگرسون^۵ تقسیم می‌شود [2].

۲-۴ درخت تصمیم^۶

درخت تصمیم نوعی یادگیری ماشینی نظارت شده است که برای دسته‌بندی یا پیش‌بینی بر اساس نحوه پاسخ به مجموعه سوالات قبلی استفاده می‌شود. این مدل نوعی یادگیری نظارت شده است، به این معنی که مدل بر روی مجموعه‌ای از داده‌ها که شامل طبقه‌بندی مورد نظر است، آموزش داده و آزمایش می‌شود. درخت تصمیم شبیه درخت است. پایه درخت گره ریشه^۷ است. از گره ریشه مجموعه ای از گره‌های درخت گیری منشعب می‌شود که تصمیماتی را که باید گرفته شود را به تصویر می‌کشد. گره‌های انتهایی درخت گره‌های برگ^۸ هستند که نتایج نهایی را نشان می‌دهند. هر گره تصمیم نشان‌دهنده یک سوال یا نقطه تقسیم است و گره‌های برگ که از یک گره تصمیم نشات می‌گیرند پاسخ‌های ممکن را نشان می‌دهند. درخت تصمیم بر دو نوع کلی درخت دسته بندی^۹ و درخت تخمین مقدار پیوسته^{۱۰} است [3].

¹ Supervised learning

² Loss function

³ cross validation process

⁴ classification

⁵ regression

⁶ decision tree

⁷ root node

⁸ leaf node

⁹ Categorical tree

¹⁰ Continuous tree

۲-۵ یادگیری گروهی^۱

یادگیری گروهی یک تکنیک یادگیری ماشینی است که با ادغام پیش‌بینی‌های چند مدل، دقت و انعطاف‌پذیری را در پیش‌بینی افزایش می‌دهد. هدف آن کاهش خطاهای یا سوگیری‌هایی است که ممکن است در مدل‌های تک وجود داشته باشد. تکنیک یادگیری گروهی این کار را با استفاده از هوش جمعی گروه انجام می‌دهد.

مفهوم یادگیری گروهی بر اساس ترکیب خروجی‌های مدل‌های مختلف با دقت کم برای ایجاد یک پیش‌بینی دقیق‌تر است. با در نظر گرفتن دیدگاه‌های متعدد و استفاده از نقاط قوت مدل‌های مختلف، یادگیری گروهی عملکرد کلی سیستم یادگیری را بهبود می‌بخشد. این رویکرد نه تنها دقت را افزایش می‌دهد، بلکه انعطاف‌پذیری در برابر عدم قطعیت^۲ در داده‌ها را نیز فراهم می‌کند. با ادغام موثر پیش‌بینی‌ها از مدل‌های متعدد، یادگیری گروهی ثابت کرده است که ابزار قدرتمندی در حوزه‌های مختلف است که پیش‌بینی‌های قوی‌تر و قابل اعتمادتری ارائه می‌دهد[۴].

۲-۶ تقویت گرادیان (شیب)^۳

تقویت گرادیان یک الگوریتم تقویت^۴ محبوب در یادگیری ماشینی است که برای کارهای طبقه‌بندی و رگرسیون استفاده می‌شود. تقویت یکی از انواع روش‌های یادگیری گروهی است که مدل را به صورت متوالی آموزش می‌دهد و هر مدل جدید سعی می‌کند مدل قبلی را اصلاح کند. این روش چندین یادگیرنده ضعیف را به یادگیرنده قوی ترکیب می‌کند. این کار به این ترتیب انجام می‌شود که هر مدل جدید برای به حداقل رساندن تابع تلفات مانند میانگین مربعات خطأ^۵ یا آنتروپی متقابل^۶ مدل قبلی با استفاده از کاهش گرادیان (شیب) آموزش داده می‌شود. در هر تکرار، الگوریتم گرادیان تابع ضرر را با توجه به پیش‌بینی‌های مجموعه فعلی محاسبه می‌کند و سپس یک مدل ضعیف جدید را برای به حداقل رساندن این گرادیان آموزش می‌دهد. سپس پیش‌بینی‌های مدل جدید به مجموعه اضافه می‌شود و این فرآیند تا زمانی که به شرایط معیار توقف برسیم، تکرار می‌شود. تقویت انطباقی^۷ و تقویت گرادیان از مدل‌های شناخته شده‌ی الگوریتم‌های تقویتی هستند [۵].

¹ ensemble learning

² bias

³ uncertainty

⁴ Gradient Boosting

⁵ Boosting algorithm

⁶ mean squared error

⁷ cross-entropy

⁸ Adaptive Boosting

۲-۷ تقویت گرادیان (شیب) مفرط^۱

تقویت گرادیان مفرط یک الگوریتم یادگیری ماشینی قدرتمند و محبوب است که در دسته چارچوب‌های تقویت گرادیان قرار می‌گیرد. این توسط تیانجی چن^۲ در سال ۲۰۱۴ معرفی شد و از آن زمان به دلیل عملکرد و کارایی استثنایی آن به طور گسترده در مسابقات مختلف علوم داده و برنامه‌های کاربردی در دنیا واقعی مورد استفاده قرار گرفت.

الگوریتم تقویت گرادیان مفرط یک روش یادگیری گروهی است، به این معنی که پیش‌بینی‌های^۳ چند یادگیرنده ضعیف (معمولًاً درخت‌های تصمیم‌گیری) را برای ایجاد یک پیش‌بینی نهایی قوی و دقیق ترکیب می‌کند. این الگوریتم با اضافه کردن مکرر درختان تصمیم به گروه کار می‌کند، بطوری که هر درخت اشتباهات درختان قبلی را اصلاح می‌کند. ایده کلیدی پشت موفقیت تقویت گرادیان مفرط هدف بهینه سازی آن است که شامل یکتابع هدف منظم است که به جلوگیری از بیش برازش^۴ کمک می‌کند.

در طول هر تکرار، الگوریتم گرادیان‌ها و مشتقات مرتبه دوم تابع ضرر را با توجه به پیش‌بینی‌ها محاسبه می‌کند. سپس یک درخت تصمیم جدید می‌سازد تا تابع هدف را با استفاده از این گرادیان‌ها و مشتقات مرتبه دوم به عنوان راهنمای حداقل بررساند. علاوه بر این، تقویت گرادیان مفرط از تکنیکی به نام نرخ یادگیری^۵ استفاده می‌کند که تأثیر هر درخت بر پیش‌بینی نهایی را کاهش می‌دهد، و بیشتر از بیش برازش جلوگیری می‌کند و عملکرد کلی الگوریتم را بهبود می‌بخشد.

برای اطمینان از کارایی محاسباتی، تقویت گرادیان مفرط از تکنیکی به نام "الگوریتم تقریبی حریص" برای پیدا کردن بهترین نقاط شکست در داده‌ها بهره می‌برد. علاوه بر این، این الگوریتم از پردازش موازی پشتیبانی می‌کند و آن را مقیاس پذیر^۶ و مناسب برای مجموعه داده‌های بزرگ می‌کند.

تقویت گرادیان مفرط دارای ویژگی‌هایی از قبیل:

- منظم‌سازی^۷ با استفاده از روش‌های L1 و L2
- مدیریت داده‌های ناموجود
- دارای الگوریتم طرح کمی^۸ برای مدیریت موثر داده‌های وزنی
- ذخیره‌سازی داده‌ها در ساختار داده بلوکی برای پردازش موازی^۹

دلایل استفاده از این الگوریتم در پیاده‌سازی‌ها بطور کلی شامل دقت بالا، مقیاس پذیری، کارایی، انعطاف پذیری، منظم‌سازی، تفسیرپذیری و متن‌باز بودن آن است [6].

¹ Boosting algorithm

² Tianqi Chen

³ prediction

⁴ overfitting

⁵ Learning rate (Eta)

⁶ scalable

⁷ Regularization

⁸ quantile sketch algorithm

⁹ Block structure storage for parallel learning

امروزه برای حل مسائلی که دارای داده‌های ساختاریافته و جدولی با اندازه کوچک تا متوسط هستند، الگوریتم‌های مبتنی بر درخت تصمیم جزو بهترین‌ها در نظر گرفته می‌شوند. این درحالی است که برای داده‌های بدون ساختار مانند عکس‌ها و متون، شبکه‌های عصبی مصنوعی^۱ گزینه‌های مناسب‌تری هستند [7].

۲-۸ اعتبارسنجی متقابل

اعتبارسنجی متقابل تکنیکی برای ارزیابی مدل‌های یادگیری ماشینی با آموزش چندین مدل یادگیری ماشینی بر روی بخشی از زیرمجموعه‌های داده‌های ورودی و ارزیابی آنها بر روی زیرمجموعه مکمل همان داده‌های ورودی است. از اعتبارسنجی متقابل برای تشخیص بیش‌برازش^۲ استفاده می‌شود.

در اعتبارسنجی متقابل چند دسته‌ای^۳، داده‌های ورودی را به k زیرمجموعه داده (همچنین به عنوان دسته^۴ شناخته می‌شود) تقسیم می‌کنیم. سپس یک مدل یادگیری ماشینی را روی همه زیرمجموعه‌ها منهای یک زیرمجموعه (k-1) آموزش می‌دهیم و سپس مدل را روی زیرمجموعه‌ای که برای آموزش استفاده نشده است، آزمایش می‌کنیم. این فرآیند k بار تکرار می‌شود و هر بار یک زیرمجموعه متفاوت برای ارزیابی از مجموعه آموزش حذف شده و با مجموعه آزمایش قبلی جایگزین می‌شود.

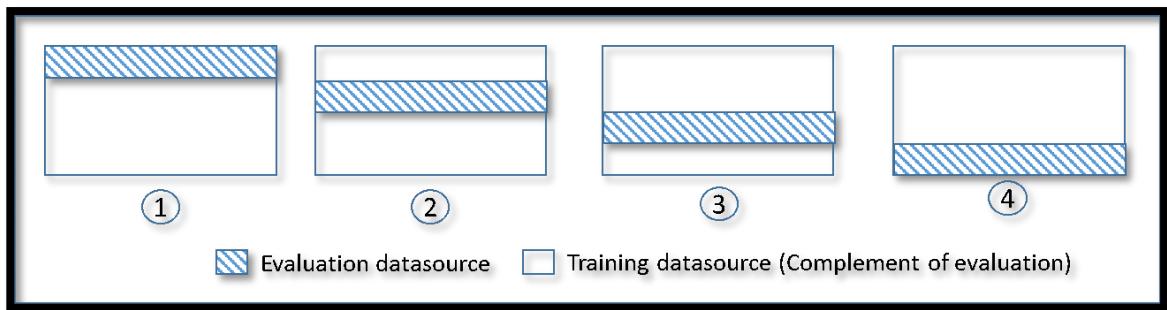
نمودار زیر، نمونه‌ای از زیرمجموعه‌های آموزشی و زیرمجموعه‌های آزمایش تولید شده برای مدلی را نشان می‌دهد که طی یک در اعتبارسنجی متقابل ۴-دسته‌ای ایجاد و آموزش داده شده‌اند. حالت یک از ۲۵ درصد اول داده‌ها برای آزمایش و از ۷۵ درصد باقی‌مانده برای آموزش استفاده می‌کند. حالت دو از زیرمجموعه دوم ۲۵ درصد (۲۵ درصد تا ۵۰ درصد) برای آزمایش و از سه زیرمجموعه باقی‌مانده از داده‌ها برای آموزش استفاده می‌کند. حالت سوم از داده‌های موجود در بازه ۵۰ تا ۷۵ درصد برای آزمایش و از باقی برای آموزش استفاده می‌کند. در نهایت حالت چهارم از ۲۵ درصد پایانی داده‌ها برای آزمایش و از ۷۵ درصد اول برای آموزش استفاده می‌کند [8] (شکل ۱-۲).

¹ artificial neural network (ANN)

² overfit

³ K-fold cross-validation

⁴ fold



شکل ۲-۱ اعتبارسنجی متقابل ۴ دسته‌ای

دلیل استفاده از اعتبارسنجی متقابل، به دست آوردن دقت واقعی مدل است. در حالتی که از اعتبارسنجی متقابل استفاده نکنیم و تنها متکی به مجموعه داده آزمایش باشیم؛ ممکن است دقت به دست آمده از دقت واقعی مدل بیشتر باشد. برای اینکه میزان خطا را کاهش دهیم به جای اتکا به مجموعه داده آزمایش، با استفاده از اعتبارسنجی متقابل و به کارگیری دسته‌های مختلف برای ارزیابی مدل سعی می‌کنیم دقت واقعی مدل را به دست آوریم.

۲-۹ مقادیر توضیحات افزونی شپلی^۱

مقادیر توضیحات افزونی شپلی راهی برای توضیح خروجی هر مدل یادگیری ماشینی است. این روش از یک رویکرد نظری بازی استفاده می‌کند که سهم هر پارامتر را در نتیجه نهایی اندازه‌گیری می‌کند. در یادگیری ماشینی، به هر ویژگی یک میزان اهمیت نسبت داده می‌شود که نشان‌دهنده سهم آن در خروجی مدل است.

مقادیر توضیحات افزونی شپلی نشان می‌دهد که چگونه هر ویژگی بر هر پیش‌بینی نهایی تأثیر می‌گذارد. همچنین این مقدار اهمیت هر ویژگی در مقایسه با سایر ویژگی‌ها و اتکای مدل به تعامل بین ویژگی‌ها را نشان می‌دهد.

ویژگی‌هایی که مقادیر توضیحات افزونی شپلی آنها مثبت است، تأثیر مثبتی در پیش‌بینی نهایی مدل دارند. در حالی که ویژگی‌هایی که مقادیر توضیحات افزونی شپلی آنها منفی است، تأثیر منفی در پیش‌بینی نهایی مدل دارند. قدر مطلق این مقدار نیز بیانگر بزرگی تأثیر این ویژگی در نتیجه نهایی است.

مقادیر توضیحات افزونی شپلی وابسته به مدل خاصی نیست و می‌توان از این روش برای مدل‌های رگرسیون خطی، درخت تصمیم، افزایش گرادیان، جنگل تصادفی و شبکه عصبی استفاده کرد [9].

¹ Shapley additive explanations

۲- ۱۰ ماتریس درهم ریختگی

برای ارزیابی نتایج مدل یادگیری ماشین که از نوع طبقه‌بندی است؛ می‌توان از ماتریس درهم ریختگی استفاده کرد. این ماتریس چهار مقدار مثبت درست، مثبت غلط، منفی درست و منفی غلط را در قالب یک ماتریس دو دوره نمایش می‌دهد. از مقادیر نام برده شده می‌توان برای محاسبه معیارهای ارزیابی دیگر بهره برد.

۲- ۱۱ پارامترهای ارزیابی نتایج مدل

دقت^۱ دقت، نسبت بین مثبت درست و تمامی مثبت‌هاست [10].

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)}$$

حساسیت^۲

حساسیت، معیاری است که مشخص می‌کند مدل ما تا چه میزان رکوردهای دسته‌ی مثبت را به خوبی ارزیابی می‌کند [10].

$$Sensitivity = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

اختصاصیت^۳

اختصاصیت مشخص می‌کند چه کسری از دسته منفی به درستی کلاس‌بندی شده‌اند [11].

$$Specificity = \frac{True\ Positive\ (TN)}{True\ Positive\ (TN) + False\ Positive\ (FP)}$$

صحت^۴

صحت، میزان دسته‌بندی‌های درست نسبت به کل دسته‌بندی‌هاست [10].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

¹ precision

² recall (sensitivity)

³ specificity

⁴ accuracy

^۱ امتیاز F_β

در مدل‌های یادگیری ماشینی که از نوع دسته‌بندی هستند؛ همواره یک بده بستان بین پارامترهای ارزیابی دقت و حساسیت برقرار است. گاهی لازم است دقت را بر حساسیت برتری داده و گاهی لازم است حساسیت را بر دقت برتری دهیم.

امتیاز F_β این امکان را به ما می‌دهد که با تنظیم پارامتر β در آن، سنجه‌ای برای ارزیابی ندل خود در هر کدام از شرایط ذکر شده در بالا به دست آوریم.

$$F_\beta = (1 + \beta^2) \times \frac{Precision \times Recall}{(\beta^2 \times Precision) + Recall}$$

امتیاز F_1

در حالت خاصی از امتیاز F_β که در آن $1 = \beta$ است؛ وزن و اهمیت پارامترهای دقت و حساسیت در فرمول برابر بوده و اصطلاحاً میانگین هارمونیک دو پارامتر دقت و حساسیت را به ما می‌دهد [12].

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

۱۲ - نمودارهای ارزیابی نتایج مدل

منحنی ویژگی عملکرد (راک)^۲

منحنی ویژگی عملکرد گیرنده (راک) یک معیار ارزیابی برای مسائل دسته‌بندی باینری است. این منحنی، یک منحنی احتمال است که نرخ مثبت‌های درست^۳ را نسبت به نرخ مثبت‌های غلط^۴ در مقادیر آستانه مختلف ترسیم می‌کند.

منحنی ویژگی عملکرد گیرنده مانند یک گراف است که نشان می‌دهد یک مدل دسته‌بندی چقدر خوب عمل می‌کند. این به ما کمک می‌کند تا بینیم مدل چگونه در سطوح مختلف قطعیت تصمیم می‌گیرد. منحنی دارای دو خط است: یک خط برای اینکه مدل چند مورد از موارد مثبت را به درستی شناسایی می‌کند (مثبت‌های درست) و دیگری برای اینکه چند مورد از موارد منفی را به اشتباه به عنوان مثبت شناسایی می‌کند (مثبت‌های غلط).

¹ F_β score

² Receiver Operating Characteristic Curve (ROC Curve)

³ True Positive Rate (TPR)

⁴ False Positive Rate (FPR)

با نگاه کردن به این گراف، می‌توان فهمید که مدل چقدر خوب عمل می‌کند و میتوانیم آستانه‌ای را انتخاب می‌کنیم که تعادل مناسبی بین پیش‌بینی‌های درست و نادرست به ما می‌دهد. هر چه مساحت زیر نمودار بیشتر باشد، عملکرد مدل بهتر است [11].

منحنی دقت-حساسیت^۱

این منحنی نیز مشابه منحنی راک، برای ارزیابی عملکرد الگوریتم‌های طبقه‌بندی باینری استفاده می‌شود. منحنی دقت-حساسیت اغلب در موقعیت‌هایی که کلاس‌ها به شدت نامتعادل هستند، استفاده می‌شود. همچنین مانند منحنی‌های راک، منحنی دقت-حساسیت بهجای یک مقدار واحد، نمایش گرافیکی عملکرد دسته‌بندی‌کننده را در مقادیر مختلف آستانه ارائه می‌کنند. بطور کلی هر چه مساحت زیر نمودار بیشتر باشد، عملکرد مدل بهتر است [13].

۲-۱۳ فریم‌ورک آپتوна

آپتونا یک چارچوب نرم‌افزاری بهینه‌سازی فرآپارامتر خودکار است که بطور خاص برای یادگیری ماشین طراحی شده است. این فریم‌ورک دارای یک واسط برنامه‌نویسی نرم‌افزاری^۲ با سبک تعریف شده توسط اجرا^۳ است. به لطف واسط برنامه‌نویسی نرم‌افزاری تعریف شده توسط آپتونا، کد نوشته شده با آپتونا از پویمانگی^۴ بالایی برخوردار است و کاربر آپتونا می‌تواند به صورت پویا فضاهای جستجو را برای فرآپارامترها بسازد [14]. روش کار این فریم‌ورک به این صورت است که با استفاده از یک تابع هدف که بر اساس پارامتر ارزیابی مشخصی نوشته شده است؛ در تعداد دور مشخص اقدام به بهینه‌سازی پارامترهای مدل از مجموعه مقادیر تعریف شده در تابع هدف به سمت جهت تعیین شده (بیشترین مقدار یا کمترین مقدار) را بصورت خودکار انجام می‌دهد.

¹ Precision-Recall Curve (PRC)

² API

³ define-by-run style

⁴ modularity

فصل سوم

۳ پیاده‌سازی مدل تقویت گرادیان مفرط (XGBoost) با پارامترهای پیش‌فرض

در این فصل با توضیح گام به گام کدهای نوشته شده و ارزیابی‌های مورد نیاز برای مدل تقویت گرادیان مفرط، دقت پیش‌فرض مدل را با پارامترهای پیش‌فرض آن محاسبه می‌کنیم. در این فصل به تنظیم پارامترهای مدل پرداخته نشده و این کار را به فصل‌های بعد موکول کرده‌ایم.

۳- ۱ اضافه کردن کتابخانه ها و مجموعه داده مربوطه

برای استفاده از داده های جدولی از کتابخانه pandas استفاده می کنیم. پس از اضافه کردن کتابخانه مدد نظر به تابع `read_csv` دسترسی پیدا می کنیم و مسیر فایل داده های خود را به آن ورودی می دهیم. برای این پروژه از داده های جدولی که از سایت معروف مجموعه داده Kaggle دریافت شده است استفاده می کنیم [15].

۳- ۲ استخراج متغیرهای مستقل و وابسته

اکنون با استفاده از متند `iloc` بر روی فریم داده دریافتی می توانیم مقادیر موجود در قسمتی از فریم داده ای را استخراج کنیم (شکل ۳-۱) تقسیم داده ها به دسته های آموزش و آزمایش [16].

```
[ ] my_ds = pd.read_csv('breast-cancer.csv')
x = my_ds.iloc[:, 2:]
y = my_ds.iloc[:, 1].values
```

شکل ۳-۱ تقسیم داده ها به دسته های آموزش و آزمایش

خط دوم در تکه کد بالا داده های موجود در تمام ردیف های حاضر در ستون های شماره دو به بعد را استخراج کرده و در یک متغیر ذخیره می کند.

خط سوم در تکه کد بالا داده های موجود در تمام ردیف های حاضر در ستون شماره یک را استخراج کرده و در یک متغیر ذخیره می کند. در اینجا با استفاده از علامت نقطه به مقادیر موجود در فریم داده دسترسی پیدا کرده ایم. در واقع نوع داده دریافتی در این خط از نوع آرایه چند بعدی^۱ است در حالی که داده دریافتی در خط قبل از نوع فریم داده^۲ بوده است. در تکه کدهای بالا از داده های درون ستون با اندیس صفر استفاده نشده است؛ زیرا این ستون دارای شناسه مربوط به هر بیمار بوده و در پیش بینی های نهایی تاثیری ندارد.

۳- ۳ ارزیابی تاثیر ستون های مختلف در پیش بینی نهایی

برای یافتن تاثیر مقدار هر ستون در نتیجه پیش بینی نهایی باید از فرمول های همبستگی استفاده نمود. در این پروژه از فرمول ضریب همبستگی پیرسون استفاده شده است.

¹ numpy.ndarray

² pandas.core.frame.DataFrame

ضریب همبستگی پیرسون اندازه گیری قدرت ارتباط خطی میان دو متغیر است. این ضریب مقداری در بازه [۰, ۱] را دارد و هرچه به دو سر این بازه نزدیک‌تر باشد، میزان قدرت ارتباط خطی میان دو متغیر یاد شده بیشتر است و هرچه به مقدار صفر نزدیک‌تر باشد، میزان قدرت ارتباط خطی میان دو متغیر یاد شده کمتر است [۱۷].

$$r = \frac{\sum(X - \bar{X})(Y - \bar{Y})}{\sqrt{\sum(X - \bar{X})^2} \sqrt{\sum(Y - \bar{Y})^2}} \quad ۱-۳$$

که در آن \bar{X} میانگین مقادیر متغیر مستقل و \bar{Y} میانگین مقادیر متغیر وابسته هستند.

۴-۳ پیاده‌سازی ضریب همبستگی پیرسون روی مجموعه داده

پس از اضافه کردن مجموعه داده به برنامه، ابتدا تعداد بیماران موجود در هر دسته را محاسبه می‌کنیم (شکل ۲-۳).



```
my_ds["diagnosis"].value_counts()
B    357
M    212
Name: diagnosis, dtype: int64
```

شکل ۲-۳ نمایش تعداد رکورد در هر دسته

تابع زیر با باینری سازی دسته‌های مختلف اقدام بالا را مجدداً انجام می‌دهد. (شکل ۳-۳)



```
Encoding Dependent variable (Diagnosis)

def categorical_to_numeric_diagnosis(x):
    if x=='M':
        return 1
    if x=='B':
        return 0

my_ds['diagnosis']= my_ds['diagnosis'].apply(categorical_to_numeric_diagnosis)
my_ds["diagnosis"].value_counts()

0    357
1    212
Name: diagnosis, dtype: int64
```

شکل ۳-۳ نمایش تعداد رکورد در هر دسته پس از رمزگذاری

تکه کد زیر همبستگی میان هر ستون با ستون دیگر را محاسبه کرده و به صورت جدولی نمایش می‌دهد (شکل ۳-۴). سپس مقدار آن را روی صفحه نمایش می‌دهیم (شکل ۳-۵).



شکل ۳-۴ محاسبه همبستگی بین ستون‌ها

	id diagnosis	radial_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave_points_mean	... radius_worst	texture_worst	perimeter_worst	area_worst	smoothness_worst	compactness_worst	concavity_worst	concave_points_worst	symmetry_worst	fractal_dimension_worst
id	1.00000	0.01979	0.01462	0.009770	0.07159	0.00980	-0.01296	0.00096	0.00000	0.04415	0.08470	0.07986	0.107187	0.010208	-0.02998	0.02203	0.03174	-0.04422	-0.02986
diagnosis	0.03976	1.00000	0.016185	0.04208	0.74208	0.70964	0.359560	0.594334	0.696300	0.776514	0.774534	0.693903	0.782359	0.421465	0.360998	0.59910	0.739566	0.476294	0.323872
radius_mean	0.074025	0.73029	1.00000	0.323782	0.99705	0.987357	0.170581	0.306134	0.676154	0.822359	0.849559	0.297088	0.951537	0.941082	0.119616	0.413463	0.529911	0.744214	0.163953
texture_mean	0.09970	0.611485	0.323782	1.00000	0.329333	0.321086	-0.023899	0.236762	0.293864	0.332573	0.612045	0.355040	0.343568	0.277830	0.301025	0.295116	0.105008	0.116205	
perimeter_mean	0.05000	0.611485	0.323782	0.329333	1.00000	0.329333	0.200000	0.236729	0.303048	0.332573	0.612045	0.355040	0.343568	0.277830	0.301025	0.295116	0.105008	0.116205	
area_mean	0.088893	0.73029	0.87357	0.321086	0.986537	0.986537	0.177508	0.499302	0.681043	0.822388	0.874889	0.299120	0.991120	0.980113	0.123523	0.394410	0.713656	0.722017	0.143375
smoothness_mean	0.027958	0.330260	0.202389	0.177508	0.994334	0.981234	0.595123	0.571584	0.559805	0.231202	0.309772	0.238053	0.209718	0.805254	0.472466	0.434839	0.302055	0.349390	0.499218
compactness_mean	0.00000	0.984334	0.594124	0.236762	0.595818	0.681902	0.499302	0.677059	0.681043	0.811135	0.805121	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721
concavity_mean	0.02000	0.985390	0.879764	0.302418	0.718157	0.683983	0.521984	0.887121	1.00000	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721
concave_points_mean	0.046158	0.774614	0.823259	0.269484	0.85977	0.823269	0.535885	0.811135	0.802191	1.00000	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721	0.803721
symmetry_mean	-0.022114	0.330449	0.147741	0.071401	0.180327	0.151293	0.505775	0.505775	0.505775	0.020341	0.305967	0.402407	0.185728	0.098051	0.219199	0.177719	0.42675	0.472020	0.433721
fractal_dimension_mean	0.025211	0.013838	-0.311447	0.074047	-0.261477	-0.281110	0.364792	0.563098	0.336783	0.166117	-0.235981	0.013949	0.205191	-0.211854	0.304894	0.346798	0.346234	0.173324	0.324019
radius_se	0.74504	0.567134	0.879690	0.235969	0.891743	0.752962	0.301457	0.497473	0.617025	0.498050	0.715085	0.719944	0.751548	0.143919	0.287103	0.380595	0.371962	0.369452	0.049259
texture_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
perimeter_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
area_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
smoothness_se	0.017731	0.255441	0.214742	0.074073	0.074073	0.729402	0.249495	0.680591	0.710001	0.681043	0.497201	0.707013	0.100203	0.214742	0.014742	0.254497	0.100203	0.214742	0.014742
compactness_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
concavity_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
concave_points_se	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
symmetry_se	0.007658	0.840542	0.376169	0.165081	0.407217	0.373230	0.380878	0.642282	0.683203	0.619584	0.398741	0.342271	0.432271	0.432271	0.432271	0.432271	0.432271	0.432271	0.432271
fractal_dimension_se	0.033724	0.00000	0.00000	-0.104077	0.009127	-0.081629	-0.072467	0.200774	0.229977	0.178027	0.093551	-0.137321	-0.077473	-0.1077473	-0.110843	-0.072626	0.060225	0.057719	0.078042
shannon_entropy	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
correlation	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
Correlation Matrix	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

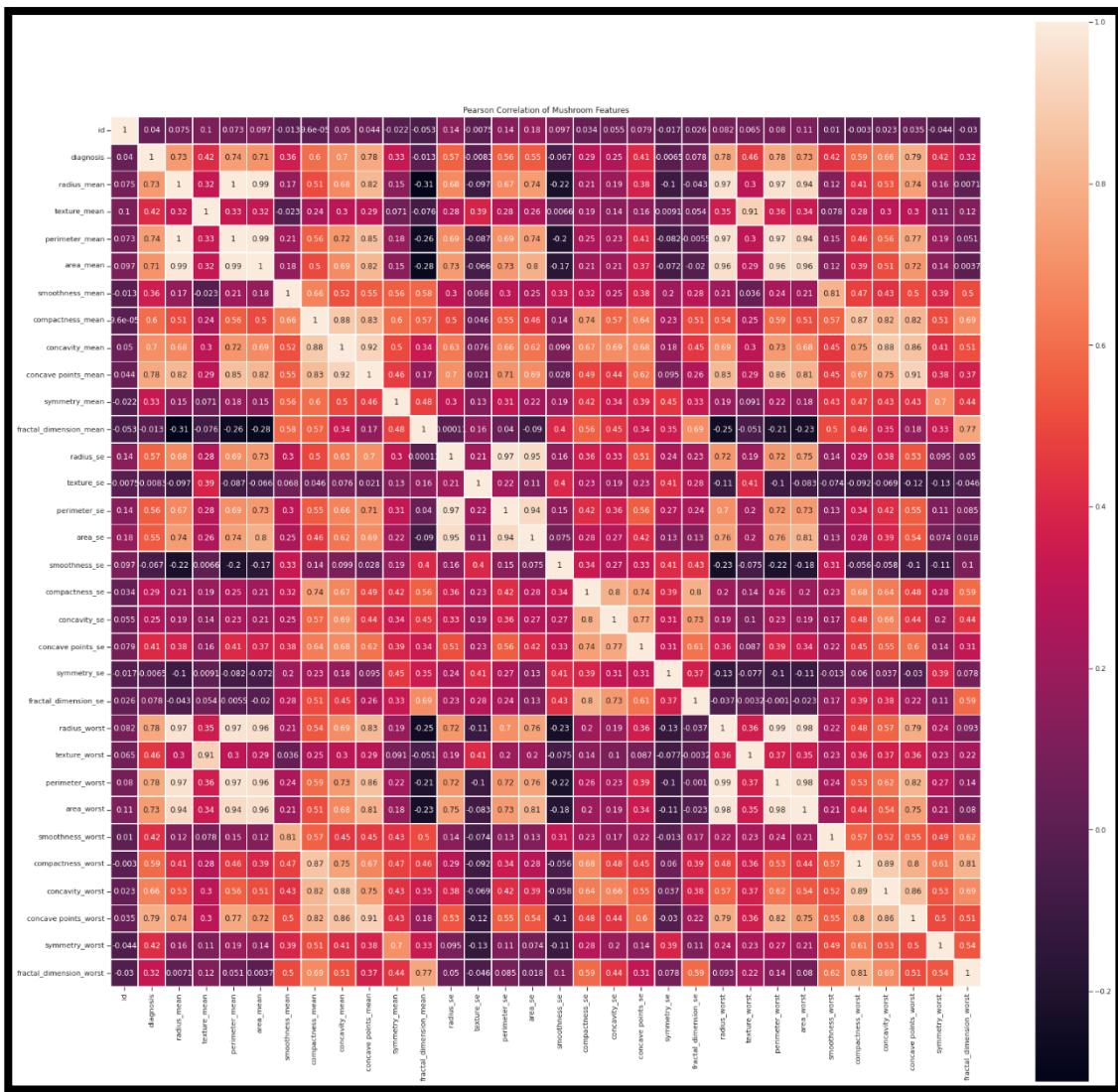
شکل ۳-۵ جدول همبستگی بین ستون‌ها

همچنین می‌توان این نمودار را به وسیله قطعه کد زیر با استفاده از رنگ‌ها به صورتی قابل درک‌تر به نمایش گذشت که به آن ماتریس همبستگی^۱ می‌گویند (شکل ۳-۶) (شکل ۳-۷).



شکل ۳-۶ رسم ماتریس همبستگی مقادیر جدول همبستگی

¹ Correlation matrix



شکل ۳-۷ ماتریس همبستگی مقادیر جدول همبستگی

از میان ضرایب همبستگی بالا، قسمتی که مرتبیط به همبستگی هر ستون با ستون متغیر وابسته است مد نظر ماست. تکه کد زیر همبستگی میان تمام ستون‌ها با ستون متغیر وابسته را محاسبه می‌کند (شکل ۳-۸).

▼ Corr coef with diagnosis col



▼ Default title text

```
# @title Default title text
tst = my_ds.corr()['diagnosis']
print(tst)
```

شکل ۱-۳ محاسبه ضریب همبستگی ستونها با ستون نتیجه

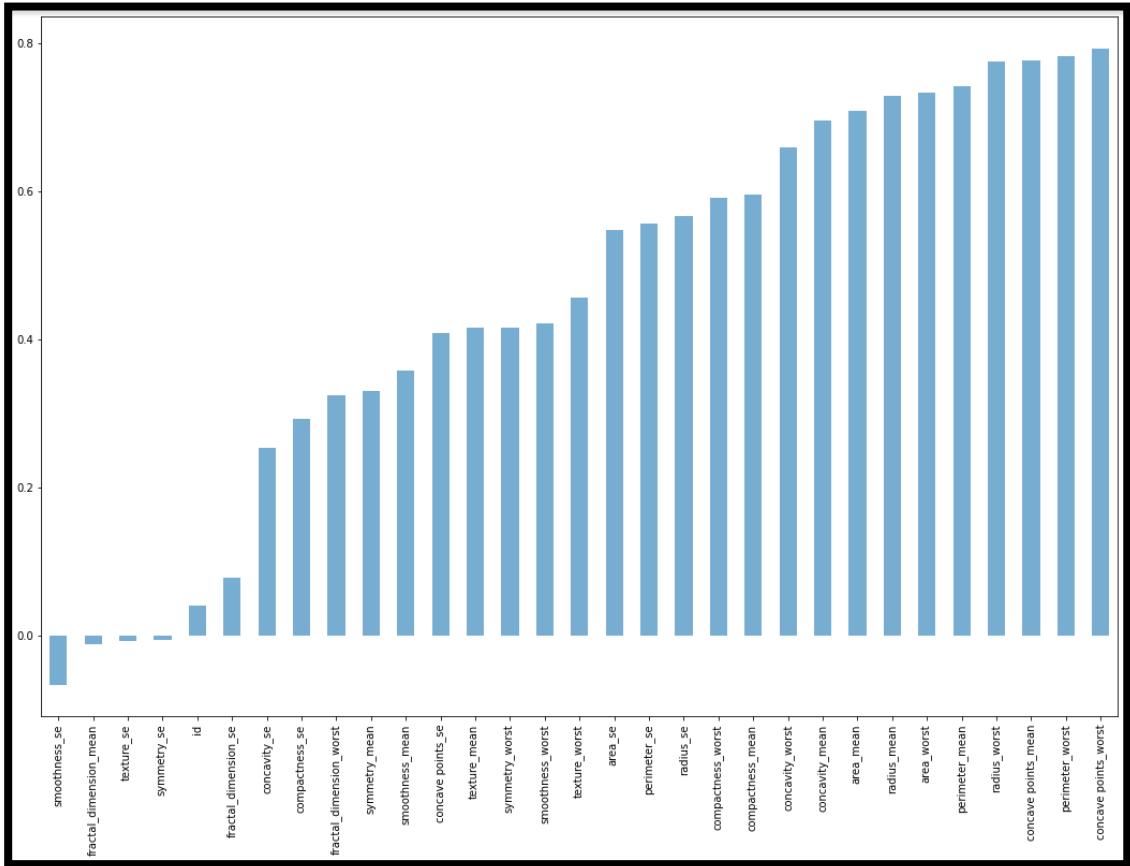
id	0.039769
diagnosis	1.000000
radius_mean	0.730029
texture_mean	0.415185
perimeter_mean	0.742636
area_mean	0.708984
smoothness_mean	0.358560
compactness_mean	0.596534
concavity_mean	0.696360
concave points_mean	0.776614
symmetry_mean	0.330499
fractal_dimension_mean	-0.012838
radius_se	0.567134
texture_se	-0.008303
perimeter_se	0.556141
area_se	0.548236
smoothness_se	-0.067016
compactness_se	0.292999
concavity_se	0.253730
concave points_se	0.408042
symmetry_se	-0.006522
fractal_dimension_se	0.077972
radius_worst	0.776454
texture_worst	0.456903
perimeter_worst	0.782914
area_worst	0.733825
smoothness_worst	0.421465
compactness_worst	0.590998
concavity_worst	0.659610
concave points_worst	0.793566
symmetry_worst	0.416294
fractal_dimension_worst	0.323872
Unnamed: 32	NaN
Name: diagnosis, dtype: float64	

شکل ۳-۹ جدول مقادیر ضریب همبستگی ستونها با ستون نتیجه

می‌توان این مقدار را با استفاده از تکه کد زیر به صورت نمودار میله‌ای نمایش داد (شکل ۳-۱۰) (شکل ۳-۹).
.



شکل ۳-۱۰ رسم نمودار میله‌ای برای نتایج شکل ۳-۹.



شکل ۳-۱۱ نمودار میله‌ای برای نتایج شکل ۳-۹.

۳-۵ حذف ستون‌هایی با تاثیر ناچیز بر نتیجه نهایی از مجموعه متغیرهای مستقل با توجه به جدول شکل ۳-۹ متغیرهای مستقلی که تاثیر آنها بر متغیر وابسته کمتر از ۱۰٪ است، از مجموعه محاسبات ما کنار گذاشته می‌شوند. این متغیرها آنهایی هستند که ضریب همبستگی‌شان با متغیر وابسته کمتر از ۰.۱ است. سپس با دسترسی به ویژگی ستون از متغیر ذخیره‌کننده فریم داده‌ی متغیرهای مستقل، نام متغیرهای مستقل را بدست آورده و آنها را در متغیری ذخیره می‌کنیم. بعدها از این متغیر برای رسم نمودار توضیحات افروزی شپلی کمک می‌گیریم. در انتها مقادیر موجود در فریم داده‌ی متغیرهای مستقل را به دست آوریم (شکل ۱۲-۳).

▼ Droping cols with tiny impact on dependent variable

```
[ ] x = x.drop(['smoothness_se', 'texture_se', 'symmetry_se', 'fractal_dimension_mean', 'fractal_dimension_se'], axis=1)
[ ] feature_names = list(x.columns)
[ ] x = x.values
```

شکل ۱۲-۳ حذف ستون‌ها با تاثیر ناچیز

۳-۶ رمزگذاری متغیر وابسته

اکنون با استفاده از کلاس LabelEncoder در تکه کد زیر متغیر وابسته را به مقادیر باینری رمز می‌کنیم (شکل ۱۳-۳) (شکل ۱۴-۳).

Encoding Dependent Variable

```
[ ] from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

شکل ۱۳-۳ رمزگذاری متغیر وابسته

۳-۷ تقسیم مجموعه داده به مجموعه آموزشی و مجموعه آزمایشی
 اکنون با استفاده ازتابع train_test_split در تکه کد زیر متغیرهای مستقل و وابسته در مجموعه داده خود را به دو دسته آموزشی^۱ و آزمایشی^۲ تقسیم می‌کنیم (شکل ۳-۱۴).

Splitting the dataset into the training set and the test set

```
[ ] from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

شکل ۳-۱۴ تقسیم مجموعه داده به سنته‌های آموزشی و آزمایشی

۳-۸ آموزش مدل تقویت گرادیان مفرط با داده‌های مجموعه آموزش
 ابتدا مدل را از کتابخانه مربوطه به برنامه اضافه کرده و سپس از آن یک طبقه‌بند^۳ تعریف می‌کنیم.
 سپس داده‌های مربوط به مجموعه آموزش را به این طبقه بند ورودی داده و مدت کوتاهی منتظر می‌مانیم تا مدل آموزش ببیند (شکل ۳-۱۵).

Training XGBoost on training set

```
[ ] from xgboost import XGBClassifier as xgbc
classifier = xgbc()
classifier.fit(x_train, y_train)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              n_estimators=100, n_jobs=None, num_parallel_tree=None,
              predictor=None, random_state=None, ...)
```

شکل ۳-۱۵ آموزش مدل با پارامترهای پیش‌فرض

¹train

²test

³classifier

۳-۹ ارزیابی مدل

۳-۹-۱ ارزیابی با ماتریس درهم‌ریختگی و محاسبه امتیاز صحت

مطابق با تکه کد پایین، با فراخوانیتابع پیش‌بینی، مقدار متغیر وابسته‌ی پیش‌بینی را می‌یابیم. سپس با ورودی دادن مقدار متغیر وابسته‌ی آزمایش و متغیر وابسته‌ی پیش‌بینی به دوتابع محاسبه ماتریس درهم‌ریختگی و محاسبه امتیاز صحت، مقادیر را یافته و نمایش می‌دهیم (شکل ۳-۱۶).

```
from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = classifier.predict(x_test)
my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)
[[66  1]
 [ 1 46]]
0.9824561403508771
```

شکل ۳-۱۶ ارزیابی مدل با ماتریس درهم‌ریختگی و امتیاز صحت

۳-۹-۲ ارزیابی با اعتبارسنجی متقابل چند دسته‌ای

مطابق با تکه کد پایین، برای انجام اعتبارسنجی متقابل چند دسته‌ای، ابتدا تابع مربوطه را به برنامه اضافه کرده و سپس با ورودی دادن طبقه‌بند، مجموعه متغیرهای مستقل آموزش و متغیرهای وابسته‌ی آموزش و تعیین تعداد دسته برابر با ده عدد، لیستی از ۱۰ مقدار برای دقت در این ۱۰ دسته به دست می‌آید. سپس با فراخوانی دوتابع محاسبه مقدار میانگین و محاسبه انحراف از معیار روی این لیست، مقادیر مورد نظر را نمایش می‌دهیم (شکل ۳-۱۷).

```
Applying k-fold cross validation
[ ] from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator=classifier, X=x_train, y=y_train, cv=10)

[ ] print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

Accuracy: 97.37 %
Standard Deviation: 1.90 %
```

شکل ۳-۱۷ ارزیابی مدل با اعتبارسنجی متقابل

با در اختیار داشتن نتیجه به دست آمده از ماتریس درهم ریختگی و امتیاز صحت و اعتبار سنجی متقابل چند دسته‌ای برای مدل آموزش دیده با متغیرهای پیش‌فرض، اکنون زمان آن رسیده که با تنظیم پارامترهای مختلف به دقتی بالاتر از دقت پیش‌فرض دست یابیم که این هدف را در فصل بعد دنبال می‌کنیم.

فصل چهارم

۴ بررسی پارامترهای مدل برای تنظیم آنها

در این فصل قصد داریم با برخی از مهم‌ترین پارامترهای موجود برای تنظیم و بهبود عملکرد الگوریتم تقویت گرادیان مفرط آشنا شویم. تعداد پارامترهای موجود بسیار است و در این بخش سعی شده بر روی پارامترهایی که تاثیر بیشتری روی نتیجه نهایی دارند، تمرکز گردد. مقدار پیش‌فرض و مقدار بهینه هر کدام از پارامترها همراه با توضیح مربوط به آن پارامتر در پاراگراف مربوطه آورده شده است.

۴-۱ انواع پارامترها

پارامترهای مدل تقویت گرادیان مفرط توسط سازندگان آن به سه دسته کلی تقسیم شده است. این دسته‌های عبارت است از پارامترهای عمومی^۱، پارامترهای تقویت‌کننده^۲ و پارامترهای یادگیری^۳. در این بخش با توجه به هدف پژوهه به تنظیم پارامترهای تقویت‌کننده خواهیم پرداخت[18].

۴-۲ نرخ یادگیری

نرخ یادگیری اندازه گام را در حال بهینه‌شدن است، در هر تکرار تعیین می‌کند. نرخ یادگیری پایین، محاسبات را کنترل می‌کند و برای دستیابی به کاهش خطای باقیمانده^۴ مانند مدلی با نرخ یادگیری بالا، مدل به دورهای بیشتری نیاز دارد. اما شанс رسیدن به جواب بهینه را افزایش می‌دهد. مقدار این پارامتر باید در بازه $[0, 1]$ باشد. مقدار پیش‌فرض آن ۰.۳ است. مقدار بهینه آن معمولاً در بازه $[0.01, 0.2]$ خواهد بود[19].

۴-۳ حداکثر عمق^۵

این پارامتر حداکثر عمق را برای هر درخت بیان می‌کند. این پارامتر برای کنترل بیش‌برازش استفاده می‌شود؛ زیرا عمق بیشتر به مدل اجازه می‌دهد تا روابط مرتبط به یک نمونه خاص را یاد بگیرد و دچار بیش‌برازش گردد. بطور کلی درختی با عمق بیشتر شанс بیشتری برای عملکرد بهتر و همچنین برای پیچیدگی و بیش‌برازش بیشتر خواهد داشت.

مقدار این پارامتر باید در بازه $[0, \infty)$ باشد. مقدار پیش‌فرض آن ۶ است. مقدار بهینه آن معمولاً در بازه $[3, 10]$ خواهد بود.

۴-۴ تعداد برآوردها^۶

این مقدار همان تعداد درختان گروه و معادل تعداد دورهای تقویتی^۷ است. مقدار این پارامتر باید در بازه $[0, \infty]$ باشد. مقدار پیش‌فرض آن ۱۰۰ است.

¹ General Parameters

² Booster Parameters

³ Learning Task Parameters

⁴ reduction in residual error

⁵ max_depth

⁶ n_estimators

⁷ boosting rounds

۴-۵ گاما

یک گره تنها زمانی تقسیم می‌شود که تقسیم نتیجه کاهش مثبتی در تابع ضرر ایجاد کند. گاما حداقل کاهش ضرر مورد نیاز برای تقسیم را معلوم می‌کند. این پارامتر باعث محافظه کارانه عمل کردن الگوریتم می‌شود؛ این بدان معنی است که ضرایب موجود در مدل به ندرت تغییر می‌کند. هر چه گاما بزرگتر باشد، الگوریتم محافظه کارانه‌تر عمل خواهد کرد [20].

مقدار این پارامتر باید در بازه $[0, \infty)$ باشد. مقدار پیش‌فرض آن ۰ است. مقدار بهینه آن معمولاً در بازه $[0, 1]$ خواهد بود.

۴-۶ زیرنمونه^۱

این مقدار برابر کسری از داده‌های آموزش است که هر درخت مناسب با آن آموزش داده می‌شود. مقادیر کم آن می‌توان باعث جلوگیری از بیش‌برازش شود. در صورتی که مقدار آن خیلی کم باشد ممکن است منجر به زیربرازش^۲ شود.

مقدار این پارامتر باید در بازه $[0, 1]$ باشد. مقدار پیش‌فرض آن ۱ است. مقدار بهینه آن معمولاً در بازه $[0.5, 1]$ خواهد بود [20].

۴-۷ نمونه ستون برای هر درخت^۳

این پارامتر شبیه به پارامتر زیرنمونه است، با این تفاوت که در این پارامتر، کسری از ستون‌های داده آموزش برای هر درخت مشخص می‌شود. این پارامتر ممکن است باعث بیش‌برازش شود.

مقدار این پارامتر باید در بازه $[0, 1]$ باشد. مقدار پیش‌فرض آن ۱ است. مقدار بهینه آن معمولاً در بازه $[0.5, 1]$ خواهد بود [19].

۴-۸ آلفا^۴

به معنی منظم‌سازی L1 روی وزن‌ها است (مشابه با رگرسیون لاسو). در حالتی که تعداد ابعاد (تعداد ستون‌های مجموعه داده) زیاد باشد، می‌توان از آن استفاده کرد تا الگوریتم هنگام پیاده‌سازی سریعتر اجرا شود. این پارامتر می‌تواند برابر با هر عدد صحیحی باشد. مقدار پیش‌فرض آن ۰ است. افزایش این مقدار باعث محافظه کار شدن مدل می‌شود [20].

¹ subsample

² underfitting

³ colsample_bytree

⁴ alpha

۴- ۹ لاند^۱

به معنی منظم‌سازی L2 روی وزن‌ها است (مشابه با رگرسیون ریج). در بخش منظم‌سازی الگوریتم تقویت گرادیان مفرط از این روش استفاده می‌شود. این پارامتر می‌تواند برابر با هر عدد صحیحی باشد. مقدار پیش‌فرض آن ۱ است. افزایش این مقدار باعث محافظه‌کار شدن مدل می‌شود [19] [20].

۴- ۱۰ حالت تصادفی^۲

حالت تصادفی برای تنظیم دانه^۳ برای مولد تصادفی استفاده می‌شود. کاربرد این پارامتر برای بازتولید نتایج یکسان در اجراهای مختلف است [21].

¹ lambda

² random state

³ seed

فصل پنجم

۵ پیاده‌سازی مدل تقویت گرادیان مفرط (XGBoost) با پارامترهای تنظیم شده

پس از معرفی پارامترهای مدل تقویت گرادیان مفرط، زمان آن رسیده که با آزمایش مقادیر مختلف برای این پارامترها، نتیجه پیش‌بینی نهایی مدل را بهبود بخشیم. در این فصل کدهای مربوطه همراه با توضیحات متناسب برای تشریح چگونگی بهبود عملکرد مدل آورده شده است.

در انتهای با رسم نمودارهای راک و دقت-حساسیت به همراه نمودارهای شپ، اطلاعات بیشتری در مورد مدل ارائه شده است.

۵- ۱ آزمایش پارامترهای مختلف برای بهینه‌سازی مدل

برای انجام این کار از کلاس GridSearchCV استفاده می‌کنیم. پیشتر در فصل‌های گذشته داده‌های جدول بیماران سلطانی را به دو بخش متغیرهای مستقل^۱ و متغیرهای وابسته^۲ تقسیم کردیم. پس از آن این دو بخش را به چهار بخش متغیرهای مستقل آموزش^۳، متغیرهای مستقل آزمایش^۴، متغیرهای وابسته‌ی آموزش^۵ و متغیرهای وابسته‌ی آزمایش^۶ تقسیم کردیم. اکنون با استفاده از مجموعه داده‌های متغیرهای مستقل آموزش و متغیرهای وابسته‌ی آموزش و کلاس اشاره شده در بالا، کار را ادامه می‌دهیم. (شکل ۵-۱)



```
from xgboost import XGBClassifier as xgb
from sklearn.model_selection import GridSearchCV
from numpy import arange

classifier = xgb()
parameters = [ {
    'n_estimators': list(range(50, 151, 1)),
    'learning_rate': arange(0.01, 0.5, 0.01),
    'gamma': arange(0, 1, 0.05),
    'subsample': [0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'colsample_bytree': [0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'max_depth': [3, 4, 5, 6, 7, 8, 9, 10],
    'reg_alpha': [0, 0.25, 0.5, 0.75, 1],
    'reg_lambda': [0, 0.25, 0.5, 0.75, 1],
    'random_state' : [1]
} ]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(x_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.2f} %".format(best_accuracy*100))
print("Best Parameters:", best_parameters)
```

شکل ۵-۱ مجموعه فضای جستجو پارامترهای مدل تقویت گرادیان مفرط

در تکه کد بالا لیستی از مجموعه مقادیری که احتمال می‌دهیم بهترین نتیجه را تولید کنند، آورده شده‌اند. در این لیست از مجموعه مقادیر محتملی که در فصل قبل بیان شد نیز استفاده شده است. سپس این پارامترها همراه با مدل تقویت گرادیان مفرط به عنوان طبقه‌بندی کلاس GridSearchCV ورودی داده شدند. امتیازدهی بر اساس دقت ارزیابی شده و اعتبارسنجی متقابل که توضیح آن در فصل دوم گفته شد، در هر دور به تعداد ۱۰ بار انجام می‌شود. آخرین پارامتر کلاس GridSearchCV بیان می‌کند که مدل‌ها برای پردازش شدن می‌توانند از تمام هسته‌های پردازشگر استفاده کنند (شکل ۵-۱).

در خطهای بعدی، شی حاصل از کلاس GridSearchCV را با داده‌های مستقل آموزش و داده‌های وابسته آموزش، آموزش داده‌ایم و پس از آن نتایج به دست آمده را در خطوط بعدی چاپ کردیم.

¹ Independent variables(features)

² Dependent variable(result)

³ x_train

⁴ x_test

⁵ y_train

⁶ y_test

۵-۲ تخمین زمان اجرای تکه کد ذکر شده

کلاس GridSearchCV برای پیدا کردن مقدار بهینه باید تمام ترکیب‌های ممکن از مقادیر مشخص شده برای پارامترها را امتحان کرده و از بین آنها ترکیبی که بهترین دقت را دارد، انتخاب نماید. برای تخمین زمان این محاسبات می‌توان از فرمول زیر استفاده کرد [22].

$$\text{زمان هر دور} \times (\text{تعداد اعتبارسنجی‌های متقابل}) \times (\text{تعداد مقادیر مختلف برای هر پارامتر}) \approx (\text{زمان تخمینی پردازش}) / (\text{تعداد هسته‌های پردازشگر})$$

به طور میانگین هر دور آموزشی چیزی بین ۱ تا ۲ ثانیه زمان می‌برد که برای در نظر گرفتن حد بالای محاسبات، فرض را بر دو ثانیه می‌گذاریم. با اعداد داده شده نتیجه محاسبات ۲,۸۸۰,۰۰۰ ثانیه به دست می‌آید که این عدد بیان می‌دارد این محاسبات با سیستم پنج هسته‌ای پژوهشگر، چیزی بیشتر از ۹۱ سال طول می‌کشد که این عدد احتمالاً از عمر پژوهشگر و خوانندگان این متن بیشتر است!

در نتیجه برای انجام این محاسبات ناچاریم بازه‌های کوچکتری را برای پارامترهای گفته شده لحاظ کنیم. در ادامه این پایان‌نامه پژوهشگر تلاش کرده کار آزمایش پارامترهای مختلف در آموزش مدل را به صورت هدفمندانه‌تر و به دور از روش جستجوی فراگیر^۱ انجام دهد تا زمان مورد نیاز برای محاسبات را تا حد چند ساعت پایین بیاورد. قطعاً نتیجه بهینه نهایی ممکن است با نتیجه بهینه در واقعیت متفاوت باشد؛ چرا که تمام ترکیبات ممکن مقادیر مشخص شده برای پاراکترها، آزمایش نشده است؛ ولی تا زمانی که سیستم‌های پردازشگر به قدرت مورد نیاز برای پردازش این حجم عظیم از محاسبات نرسیده‌اند، ناگزیر به استفاده از این روش هستیم.

۵-۳ محاسبه مقدار بهینه برای تعداد برآوردها و نرخ یادگیری

دوتا از مهم‌ترین پارامترهای مدل تقویت گرادیان مفرط، تعداد برآوردها و نرخ یادگیری هستند. در اولین تلاش برای بهبود عملکرد مدل، پژوهشگر بر روی این دو پارامتر تمرکز کرده تا مقدار بهینه آنها را به دست آورد. با در نظر نگرفتن مقدار برای پارامترهای دیگر، مقدار پیش‌فرض آنها در نظر گرفته خواهد شد. در این تلاش، مقدار امتحانی برای تعداد برآوردها، بازه اعداد [50, 201] با گام یک واحد و برای نرخ یادگیری، این بازه برابر با [0.01, 0.5] با گام یک صدم واحد است (شکل ۵-۲).

^۱ brute force

```

from xgboost import XGBClassifier as xgb
from sklearn.model_selection import GridSearchCV
from numpy import arange

classifier = xgb()
parameters = [ {
    'n_estimators': list(range(50, 201, 1)),
    'learning_rate': arange(0.01, 0.5, 0.01),
    'random_state' : [0]
} ]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(x_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.2f}%".format(best_accuracy*100))
print("Best Parameters:", best_parameters)

```

Best Accuracy: 97.81 %
Best Parameters: {'learning_rate': 0.27, 'n_estimators': 157, 'random_state': 0}

شکل ۲-۵ پیدا کردن مقدار بهینه برای پارامترهای تعداد برآوردگر و نرخ یادگیری

همانطور که در تصویر مشاهده می‌شود، مقدار بهینه برای پارامترهای تعداد برآوردگرها و نرخ یادگیری به ترتیب برابر با $157/0.27$ است. از این مقادیر در دورهای بعدی برای بهبود عملکرد با اضافه کردن امتحان کردن پارامترهای دیگر استفاده می‌کنیم.

۴-۵ آموزش مجدد مدل با پارامترهای بهینه

برای این کار مطابق با شکل زیر، پارامترهای بهینه به دست آمده که در ساختار داده لغتنامه^۱ ذخیره شده است را به کلاس ورودی داده و مدل جدیدی را مطابق با پارامترهای بهینه می‌سازیم. سپس مدل را با مجموعه داده‌های مستقل آموزش ووابسته آموزش مجدد آموزش می‌دهیم (شکل ۳-۵).

```

new_classifier = xgb(**best_parameters)
new_classifier.fit(x_train, y_train)

```

```

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.27, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing='nan', monotone_constraints=None,
              n_estimators=157, n_jobs=None, num_parallel_tree=None,
              predictor=None, random_state=0, ...)

```

شکل ۳-۵ آموزش مجدد مدل با مقادیر بهینه

¹ dictionary

۵-۵ ارزیابی مدل

۱-۵ اعمال اعتبار سنجی متقابل چند دسته‌ای

مجدداً اعتبار سنجی متقابل چند دسته‌ای را بر روی مدل به دست آمده در این اجرا اعمال می‌کنیم. در مقایسه نتایج با نتایج دور قبلی که برای مدل با پارامترهای پیش‌فرض انجام شده بود، مشاهده می‌شود که علی‌رغم افزایش اندک انحراف از معیار از $1/90$ به $1/96$ ، میزان دقت حدود $0/5$ درصد و از $97/37$ به $97/81$ افزایش یافته است (شکل ۵-۴).



```
[30] from sklearn.model_selection import cross_val_score
    accuracies = cross_val_score(estimator=new_classifier, X=x_train, y=y_train, cv=10)

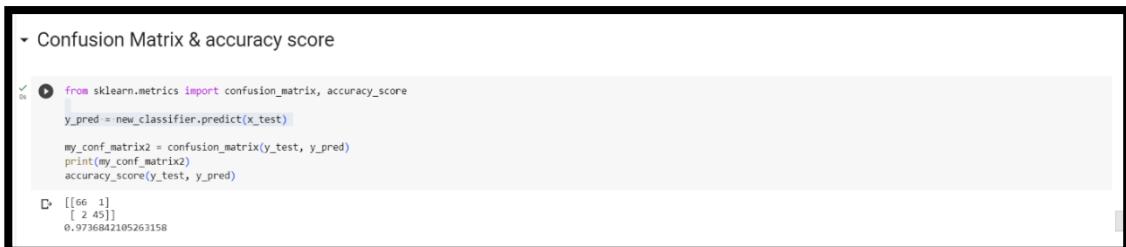
[31] print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
    print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

Accuracy: 97.81 %
Standard Deviation: 1.96 %
```

شکل ۵-۴ ارزیابی مدل با اعتبار سنجی متقابل

۲-۵ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

مجدداً ماتریس درهم ریختگی و امتیاز صحت را محاسبه می‌کنیم. مشاهده می‌شود که هرچند دقت مدل در اعتبار سنجی متقابل چند دسته‌ای افزایش داشته است، دقت مدل جدید بر روی داده‌های آزمایش کاهش داشته است (شکل ۵-۵).



```
[1] from sklearn.metrics import confusion_matrix, accuracy_score

y_pred = new_classifier.predict(x_test)

my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)

[[66  1]
 [ 2 45]]
0.9736842105263158
```

شکل ۵-۵ ارزیابی مدل با ماتریس درهم ریختی و امتیاز صحت

۳-۵ نمایش ماتریس درهم ریختگی به همراه برچسب

برای نمایش ماتریس درهم ریختگی به همراه برچسب از تابع heatmap از کتابخانه seaborn استفاده می‌کنیم. ماتریس درهم ریختگی به دست آمده در قسمت قبل را به این تابع ورودی داده و سپس برچسب‌های مورد نظر را بر روی آن قرار می‌دهیم (شکل ۵-۶).

```

▼ Confusion Martix with labels

import seaborn as sns
import matplotlib.pyplot as plt

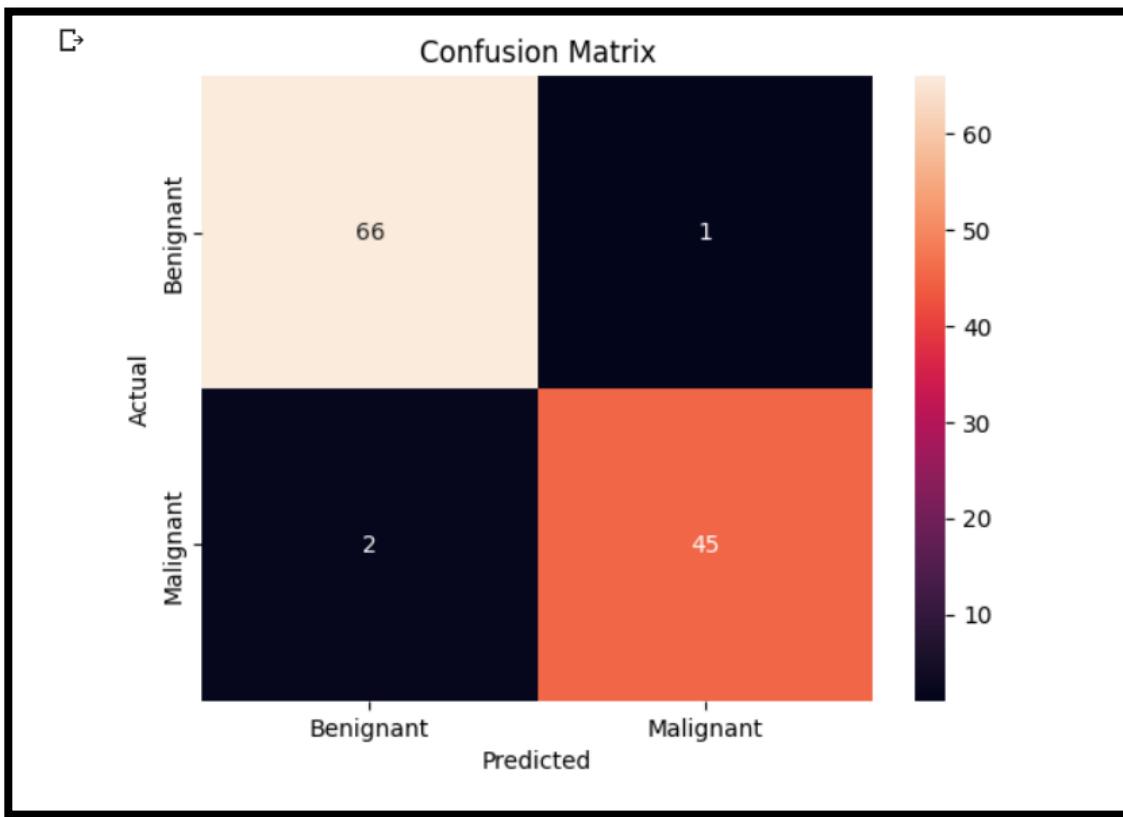
ax=plt.subplot()
sns.heatmap(my_conf_matrix2, annot=True, fmt='g', ax=ax); #annot=True to annotate cells, fmt='g' to disable scientific notation

# labels, title and ticks
ax.set_xlabel('Predicted');ax.set_ylabel('Actual');
ax.set_title('Confusion Matrix');
ax.xaxis.set_ticklabels(['Benignant', 'Malignant']); ax.yaxis.set_ticklabels(['Benignant', 'Malignant']);

```

شکل ۷-۵ نمایش ماتریس درهم ریختگی با برچسب

نتیجه به دست آمده ماتریسی مطابق با شکل پایین است (شکل ۷-۵).



شکل ۷-۵ ماتریس درهم ریختگی با برچسب

۴ - ۵ گزارش دسته‌بندی
می‌توانیم با استفاده از تکه کد زیر گزارش دسته‌بندی را برای کلاس‌های مختلف مجموعه داده خود پیدا کنیم (شکل ۸-۵).

```

from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))

```

	precision	recall	f1-score	support
0	0.97	0.99	0.98	67
1	0.98	0.96	0.97	47
accuracy	0.97	0.97	0.97	114
macro avg	0.97	0.97	0.97	114
weighted avg	0.97	0.97	0.97	114

شکل ۸-۵ گزارش دسته‌بندی

طبق گزارش آورده شده در جدول بالا، ۹۷ درصد داده‌های مربوط به کلاس با اندیس صفر (دسته بیماران سرطانی خوش‌خیم) به درستی دسته‌بندی شده است. این آمار برای داده‌های مربوط به کلاس با اندیس یک (دسته بیماران سرطانی بدخیم) برابر با ۹۸ درصد است.

معیار بعدی میزان یادآوری^۱ است. همانطور که از جدول مشاهده می‌شود، مدل ما توانسته ۹۹ درصد داده‌های مربوط به کلاس با اندیس صفر (دسته بیماران سرطانی خوش‌خیم) را از بین مجموعه داده‌های این دسته پیدا کند. این مقدار برای داده‌های مربوط به کلاس با اندیس یک (دسته بیماران سرطانی بدخیم) برابر با ۹۶ درصد است.

معیار بعدی امتیاز f1 است که همانطور که در فصل دوم توضیح داده شد، بیانگر میانگین هارمونیک مقدار دقت و حساسیت است.

همانطور که از جدول مشاهده می‌شود این مقدار برای داده‌های مربوط به کلاس با اندیس صفر (دسته بیماران سرطانی خوش‌خیم) برابر با ۹۸ درصد و برای داده‌های مربوط به کلاس با اندیس یک (دسته بیماران سرطانی بدخیم) برابر با ۹۷ درصد است. این آمار نشان می‌دهد به طور میانگین مدل ما در ارزیابی بیماران سرطانی خوش‌خیم کمی موفق‌تر است.

معیار آخر میزان حمایت^۲ است که بیان می‌کند تعداد داده‌های مربوط به کلاس با اندیس صفر (دسته بیماران سرطانی خوش‌خیم) در داده‌های آزمایش برابر با ۶۷ عدد تعداد داده‌های مربوط به کلاس با اندیس یک (دسته بیماران سرطانی بدخیم) برابر با ۴۷ عدد است.

همانطور که از این آمار و ارقام مشخص است، تعداد داده‌های موجود در دسته اول و دوم با هم تفاوت زیادی نداشته و این مجموعه داده اصطلاحاً متعادل^۳ است.

صحت دسته‌بندی‌ها^۴ نیز مطابق با جدول برابر با ۹۷ است [23].

¹ recall

² support

³ balanced

⁴ accuracy

۵-۵ محاسبه معیارهای اختصاصیت و حساسیت بطور مشابه با ورودی دادن مقدار متغیر وابسته‌ی آزمایش و متغیر وابسته‌ی پیش-بینی به تابع زیر، مقادیر موجود حاصل می‌گردد.

همانطور که از مقادیر داخل شکل زیر مشاهده می‌شود، میزان اختصاصیت برای داده‌های مربوط به کلاس با اندیس صفر (دسته بیماران سرطانی خوش‌خیم) برابر با ۹۵ درصد و برای داده‌های مربوط به کلاس با اندیس یک (دسته بیماران سرطانی بدخیم) برابر با ۹۸ درصد است. این مقدار نسبت منفی‌های درست برای کلاس‌های ذکر شده را بیان می‌دارد (شکل ۹-۵).

```

[35]: from imblearn.metrics import sensitivity_specificity_support
pd.DataFrame(sensitivity_specificity_support(y_test, y_pred, average=None), index=['Sensitivity', 'Specificity', 'Support']).T

```

	Sensitivity	Specificity	Support
0	0.985075	0.957447	67.0
1	0.957447	0.985075	47.0

شکل ۹-۵ مقادیر دقت و حساسیت

۶-۵ بهبود عملکرد مدل با تنظیم پارامترهای گاما، زیرنمونه، نمونه ستون برای هر درخت و حداکثر عمق

در تلاش دوم پژوهشگر به سراغ پارامترهای گاما، زیرنمونه، نمونه ستون برای هر درخت و حداکثر عمق رفته و با تنظیم مقدار آنها به دقتی بالاتر از دقت مدل با پارامترهای پیش‌فرض دست یافت (شکل ۱۰-۵).

```

[ ] classifier = xgbc()
parameters = [
    'n_estimators': [157], #list(range(10, 201, 1)),
    'learning_rate': [0.27], #range(0.01, 0.5, 0.01),
    'gamma': range(0, 1, 0.02),
    'subsample': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'colsample_bytree': [0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'random_state': [0]
]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(x_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.10f} %".format(best_accuracy*100))
print("Best Parameters:", best_parameters)

Best Accuracy: 98.4685990338 %
Best Parameters: {'colsample_bytree': 0.6, 'gamma': 0.16, 'learning_rate': 0.27, 'max_depth': 2, 'n_estimators': 157, 'random_state': 0, 'subsample': 0.5}

```

شکل ۱۰-۵ پیدا کردن مقدار بهینه برای پارامترهای گاما، زیرنمونه، نمونه ستون برای هر درخت و حداکثر عمق

همانطور که در شکل مشاهده می‌شود با استفاده از پارامترهای به دست آمده از اجرای قبلی، در این اجرا مجدداً مدل را برای چند ساعت با مقادیر مختلف برای پارامترهای مذکور آزمایش کردیم و مقادیر بهینه به دست آمده برای پارامترها در تصویر قابل مشاهده است. مدل به دست آمده پس از این اجرا از مدل پیش فرض موجود در فصل سوم به میزان حدود ۰/۲۲ درصد عملکرد بهتری دارد.

۵-۷ ارزیابی مدل

۵-۷-۱ اعمال اعتبارسنجی متقابل چند دسته‌ای می‌توان مجدداً اعتبارسنجی متقابل چند دسته‌ای را روی مدل حاصل اعمال کرد. نتایج به دست آمده به صورت زیر است (شکل ۱۱-۵).

```
[ ] Applying k-fold cross validation

[ ] from sklearn.model_selection import cross_val_score
    accuracies = cross_val_score(estimator=new_classifier, X=x_train, y=y_train, cv=10)

[ ] print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
    print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

Accuracy: 98.47 %
Standard Deviation: 1.40 %
```

شکل ۱۱-۵ ارزیابی مدل با اعتبارسنجی متقابل

در مقایسه با مدل اجرای قبل، مشاهده می‌شود مقدار دقت مدل از ۹۷/۸۱ واحد به مقدار ۹۸/۴۷ افزایش یافته و انحراف از معیار نیز از مقدار ۱/۹۶ به مقدار ۱/۴۰ کاهش یافته است. در مقایسه با مدل اولیه با پارامترهای پیش‌فرض، مشاهده می‌شود مقدار دقت مدل از ۹۷/۳۷ واحد به مقدار ۹۸/۴۷ افزایش یافته و انحراف از معیار نیز از مقدار ۱/۹۰ به مقدار ۱/۴۰ کاهش یافته است. همانطور که مشخص است، مدل فعلی در مقایسه با هر دو مدل قبلی عملکرد بهتری دارد.

۵-۷-۲ محاسبه ماتریس درهم‌ریختگی و امتیاز صحت

ماتریس درهم ریختگی و امتیاز صحت را مجدداً محاسبه می‌کنیم (شکل ۱۲-۵).

▼ Confusion Matrix & accuracy score

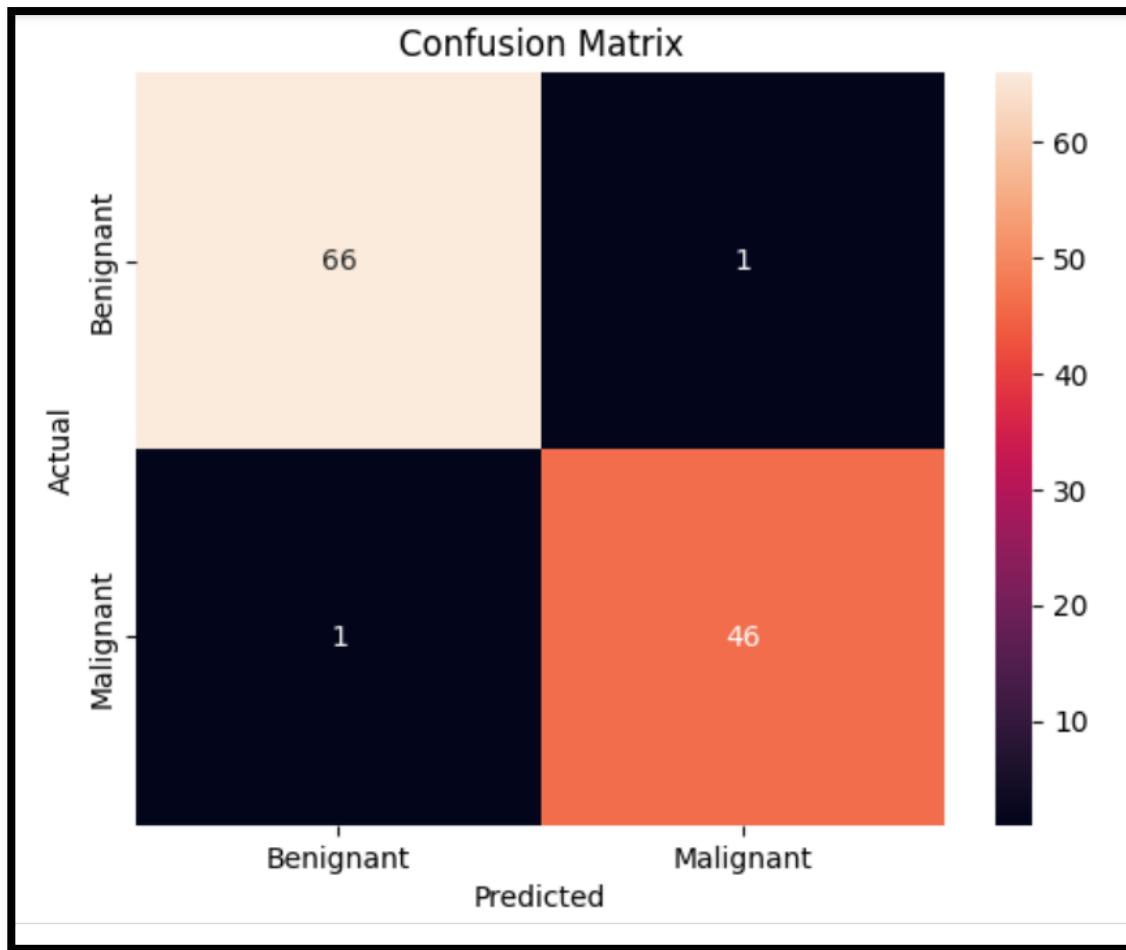
```
[ ] from sklearn.metrics import confusion_matrix, accuracy_score  
  
y_pred = new_classifier.predict(x_test)  
  
my_conf_matrix2 = confusion_matrix(y_test, y_pred)  
print(my_conf_matrix2)  
accuracy_score(y_test, y_pred)  
  
[[66 1]  
 [ 1 46]]  
0.9824561403508771
```

شکل ۱۲-۵ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

در مقایسه با مدل اجرای قبل، مشاهده می‌شود که نتایج بهبود داشته است. از تعداد منفی غلط در ماتریس درهم ریختگی یک واحد کم شده است. همچنین امتیاز صحت از ۰/۹۷۳۶ در اجرای قبل به مقدار ۰/۹۸۲۴ رسیده است.

در مقایسه با مدل اولیه با پارامترهای پیش‌فرض، مشاهده می‌شود مقدار دقت مدل بر روی داده‌های آزمایش تغییری نداشته و ماتریس‌های درهم ریختگی نیز کاملاً مشابه هستند.

۵-۷ - ۳ نمایش ماتریس درهم ریختگی به همراه برچسب با استفاده از تکه کد مشابه قسمت متناظر در اجرای قبل، ماتریس درهم ریختگی به همراه برچسب بصورت زیر بدست می‌آید (شکل ۱۳-۵).



شکل ۱۳-۵ ماتریس درهم ریختگی با برچسب

۷-۴ - گزارش دسته‌بندی

گزارش دسته‌بندی در مدل جدید به شرح زیر است (شکل ۱۴-۵).



شکل ۱۴-۵ گزارش دسته‌بندی

طبق گزارش آورده شده در جدول بالا، ۹۹ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم به درستی دسته‌بندی شده است. این آمار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است.

معیار بعدی میزان یادآوری^۱ است. همانطور که از جدول مشاهده می‌شود، مدل ما توانسته ۹۹ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم را از بین مجموعه داده‌های این دسته پیدا کند. این مقدار برای داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم برابر با ۹۸ درصد است.

همانطور که از جدول مشاهده می‌شود امتیاز ۴۱^۲ برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۹ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است. این آمار نشان می‌دهد به طور میانگین مدل ما در ارزیابی بیماران سرطانی خوش‌خیم کمی موفق‌تر است. معیار میزان حمایت^۳ نیز مربوط به مجموعه داده بوده و برای تمامی مدل‌های این مجموعه داده یکسان است؛ مگر اینکه شیوه تقسیم داده‌ها به دسته‌های آزمایش و آموزش را با عوض کردن دانه حالت رندوم^۴، تغییر دهیم.

صحت دسته‌بندی‌ها نیز مطابق با جدول برابر با ۹۸ است.

همانطور که مشخص است جدول گزارش دسته‌بندی نشان می‌دهد به طور کلی دقت مدل جدید نسبت به مدل قبل افزایش داشته است.

۵-۷-۵ محاسبه معیارهای اختصاصیت و حساسیت

همانطور که از مقادیر داخل شکل زیر مشاهده می‌شود؛ میزان اختصاصیت برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۷ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است. این مقدار نسبت منفی‌های درست برای کلاس‌های ذکر شده را بیان می‌دارد (شکل ۵-۵).

```
from imblearn.metrics import sensitivity_specificity_support
pd.DataFrame(sensitivity_specificity_support(y_test, y_pred, average=None), index=['Sensitivity', 'Specificity', 'Support']).T
```

	Sensitivity	Specificity	Support
0	0.985075	0.978723	67.0
1	0.978723	0.985075	47.0

شکل ۵-۵ محاسبه معیارهای اختصاصیت و حساسیت

¹ recall

² support

³ Seed for random_state

۶-۷ - رسم منحنی ویژگی عملکرد گیرنده (راک)

برای رسم این منحنی و منحنی دقت-حساسیت نیاز است مشخص کنیم هر رکورد از مجموعه داده با چه احتمالی به کدام کلاس تعلق دارد برای این کار از تکه کد زیر استفاده می کنیم (شکل ۵-۱۶).

```
[ ] from sklearn.metrics import roc_curve, roc_auc_score  
y_pred_proba = new_classifier.predict_proba(x_test)  
  
[ ] pd.DataFrame(y_pred_proba)  
  
      0      1  
0  0.000395  0.999605  
1  0.977011  0.022989  
2  0.999969  0.000031  
3  0.998685  0.001315  
4  0.999807  0.000193  
...  ...  ...  
109 0.000313  0.999687  
110 0.999693  0.000307  
111 0.000161  0.999839  
112 0.000319  0.999681  
113 0.933531  0.066470  
114 rows x 2 columns
```

شکل ۵-۱۶ مقادیر احتمال تعلق هر رکورد به هر کلاس

همانطور که از شکل مشخص است، با ورودی دادن مجموعه داده‌های وابسته آزمایش به تابع مدنظر نمایش خروجی آن مشاهده می کنیم که به ازای هر رکورد از رکوردهای درون مجموعه داده مذکور، دو عدد که هر کدام بیانگر احتمال تعلق آن رکورد به هر کدام از کلاس‌هاست، نمایش داده شده است. بدیهی است مجموع این دو عدد برای هر رکورد برابر با یک خواهد شد.

اکنون با استفاده از تکه کد زیر، نمودار منحنی ویژگی عملکرد گیرنده را رسم می کنیم. در این تکه کد با ورودی دادن مقادیر به دست آمده در بالا به همراه داده‌های وابسته آزمایش به تابع، مقادیر نرخ مثبت غلط و نرخ مثبت درست و میزان آستانه‌ها را به دست می آوریم. سپس برای رسم خط اریب با شبیب واحد، لیستی از مقادیر صفر را به همراه داده‌های وابسته آزمایش به تابع مذکور ورودی می دهیم. پس از مشخص کردن استایل نمودار، منحنی راک را به همراه خط اریب شبیب واحد ترسیم کرده و برای نمودار برچسب تنظیم می کنیم. با استفاده از خط آخر، نمودار را روی صفحه ترسیم می کنیم (شکل ۵-۱۷).

```
[ ] fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba[:, 1])

# roc curve for tpr = fpr
random_probs = [0 for i in range(len(y_test))]
p_fpr, p_tpr, _ = roc_curve(y_test, random_probs, pos_label=1)

plt.style.use('seaborn')

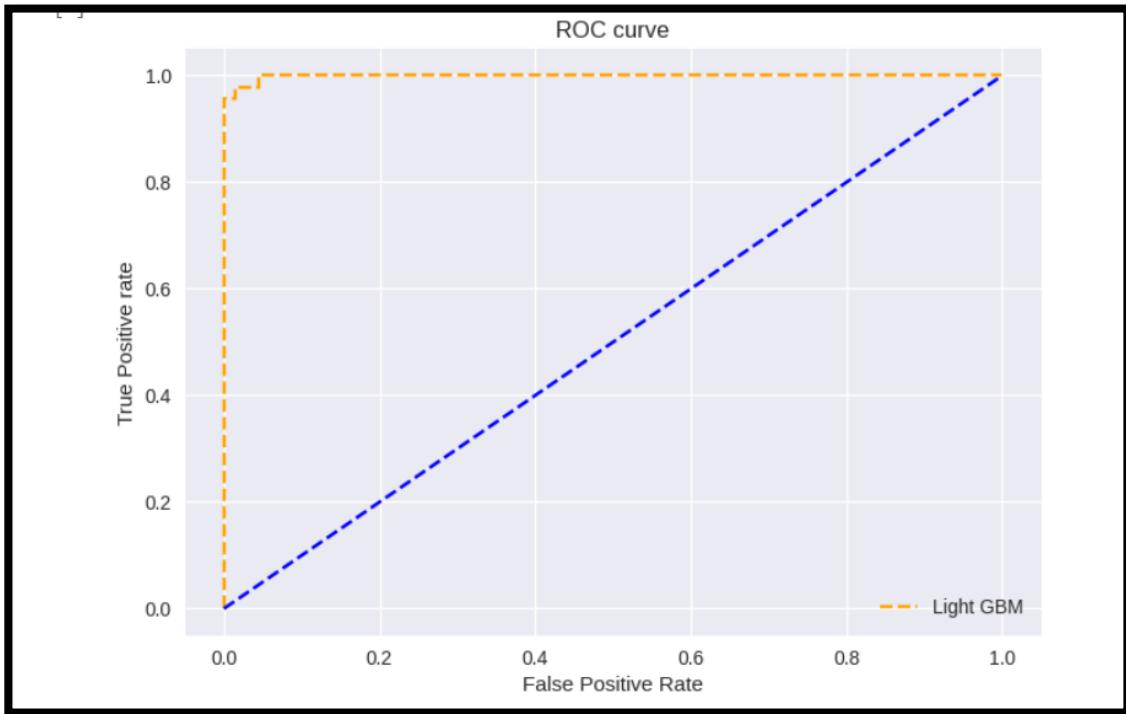
# plot roc curves
plt.plot(fpr, tpr, linestyle='--', color='orange', label='Light GBM')
plt.plot(p_fpr, p_tpr, linestyle='--', color='blue')

# title
plt.title("ROC curve")
# x label
plt.xlabel("False Positive Rate")
# y label
plt.ylabel("True Positive rate")

plt.legend(loc='best')
plt.savefig('ROC', dpi=300)
plt.show();
```

شکل ۱۷-۵ رسم نمودار راک

منحنی ویژگی عملکرد گیرنده ترسیم شده به صورت زیر است (شکل ۱۸-۵).



شکل ۱۸-۵ نمودار راک

برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۱۹-۵).

```
[ ] print("ROC AUC Score: ", roc_auc_score(y_test, y_pred_proba[:,1]))

ROC AUC Score: 0.9987297554779295
```

شکل ۱۹-۵ مساحت زیر نمودار راک

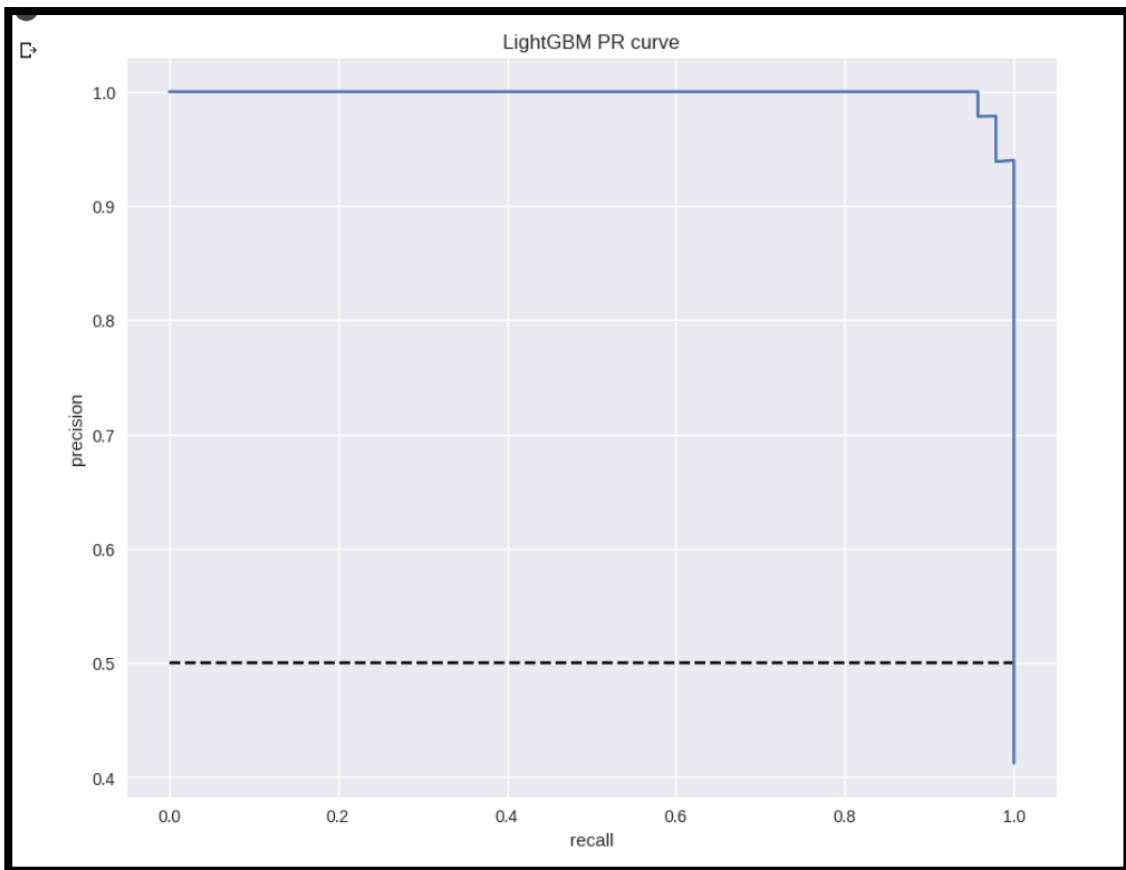
همانطور که از شکل مشخص است؛ میزان مساحت زیر نمودار برابر با $۰/۹۹۸۷$ است که نزدیکی آن به مقدار واحد بیانگر دقت بالای مدل است.

۷-۷ رسم نمودار منحنی دقت-حساسیت برای رسم نمودار منحنی دقت-حساسیت لازم است لیست احتمالات به دست آمده در قسمت قبل را به همراه مجموعه داده‌های وابسته آزمایش به تابع ورودی دهیم. سپس با استفاده از میزان دقت و یادآوری به دست آمده، نمودار را ساخته و بعد از اضافه کردن برچسب‌های مورد نیاز با استفاده از خط آخر تکه کد آن را روی صفحه نمایش می‌دهیم (شکل ۲۰-۵).



شکل ۲۰-۵ رسم نمودار منحنی دقت-حساسیت

منحنی دقت-حساسیت رسم شده به صورت زیر است (شکل ۲۱-۵).



شکل ۲۱-۵ نمودار دقت-حساسیت

برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۲۲-۵).



شکل ۲۲-۵ مساحت زیر نمودار دقت-حساسیت

همانطور که از شکل مشخص است، میزان مساحت زیر نمودار برابر با 0.9982 است که نزدیکی آن به مقدار واحد بیانگر دقیق بالای مدل است.

۸-۸ رسم نمودار مقادیر توضیحات افزونی شپلی
اکنون زمان آن رسیده که با رسم نمودار شپ، میزان تاثیر هر کدام از ویژگی‌ها را در نتیجه نهایی پیش‌بینی بررسی کنیم [24].

برای این کار لازم است ابتدا یک توضیح‌دهنده ایجاد کنیم. سپس مجموعه داده‌های مستقل (ویژگی‌ها) را به این توضیح‌دهنده ورودی داده مقادیر شپ را به دست می‌آوریم. در ادامه از این مقادیر برای رسم نمودارهای شپ متفاوت استفاده می‌کنیم. همانطور که مشاهده می‌شود، متغیری را که در فصل سوم برای نگهداری نام ویژگی‌ها استفاده کردیم؛ در اینجا به توضیح‌دهنده ورودی داده شده تا نام هر ویژگی در کنار مقدار شپ آن که بیانگر میزان تاثیر است؛ نوشته شود [25].

در صورتی که مجموعه داده دارای رکوردهای زیادی است، می‌توان تعداد محدودی از آنها را به عنوان ورودی به توضیح‌دهنده داد. در مجموعه داده فعلی به علت تعداد کم رکوردها، محدودیتی در این زمینه در نظر نمی‌گیریم (شکل ۵-۲۳).



```
[ ] # Fits the explainer
explainer = shap.Explainer(new_classifier, feature_names=feature_names)
# Calculates the SHAP values - It takes some time
shap_values = explainer(x)
```

شکل ۵-۲۳ تعریف توضیح‌دهنده و یافتن مقادیر شپ

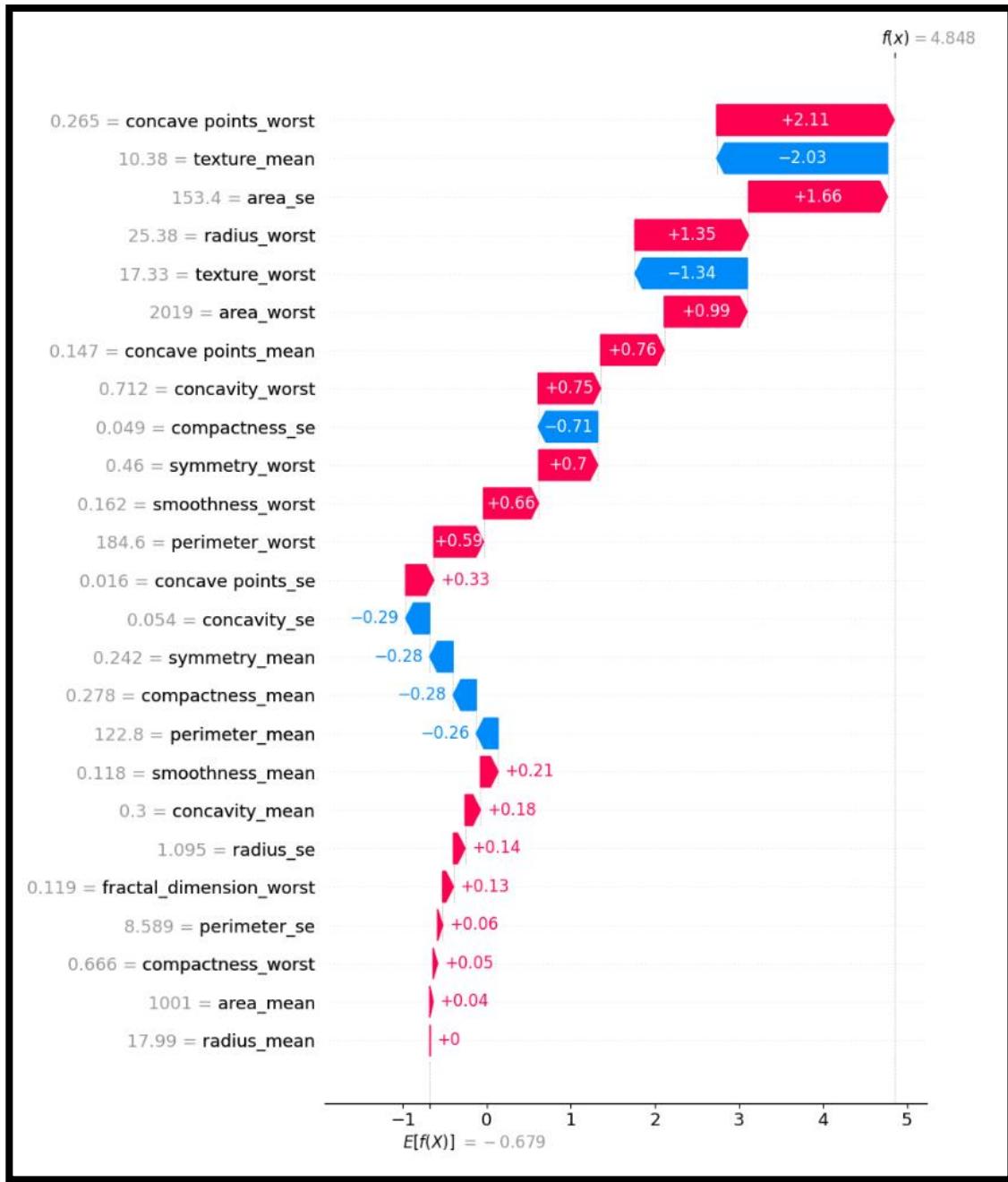
۱-۸ رسم نمودار آبشاری^۱

این نمودار ویژگی‌های موثر در پیش‌بینی نتیجه برای یک رکورد را نشان می‌دهد. در شکل زیر این نمودار برای اولین رکورد (رکورد با اندیس صفر) رسم شده است (شکل ۵-۲۴) (شکل ۵-۲۵).



شکل ۵-۲۴ رسم نمودار آبشاری

^۱ Waterfall plot



شکل ۲۵-۵ نمودار آپشاری برای رکورد با اندیس صفر

۲-۸- رسم نمودار نیرو^۱

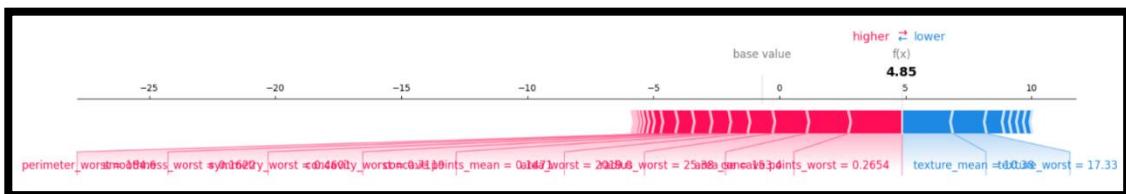
مشابه نمودار قبلی، این نمودار نیز ویژگی‌های موثر در پیش‌بینی نتیجه برای یک رکورد را نشان می‌دهد. در شکل زیر این نمودار برای اولین رکورد (رکورد با اندیس صفر) رسم شده است (شکل ۲۶-۵) (شکل ۲۷-۵).

^۱ Force plot

• Force Plot

```
[ ] shap.plots.force(shap_values[0], matplotlib=True)
```

شکل ۲۶-۵ رسم نمودار نیرو



شکل ۲۷-۵ نمودار نیرو برای رکورد با اندیس صفر

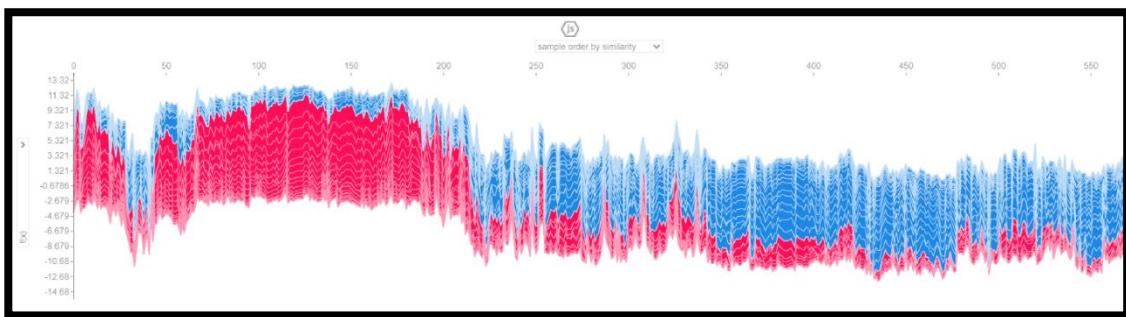
۳-۸- رسم نمودار نیرو به صورت پشته^۱

این نمودار حالت پشته شده نمودار قبلی برای تمام رکوردها است. خط اول کد آورده شده در عکس پایین صرفا برای رسم نمودار به کمک زبان برنامه‌نویسی جاوا اسکریپت در مرورگر است (شکل ۲۸-۵) (شکل ۲۹-۵).

• Stacked force plot

```
shap.initjs()  
shap.plots.force(shap_values)
```

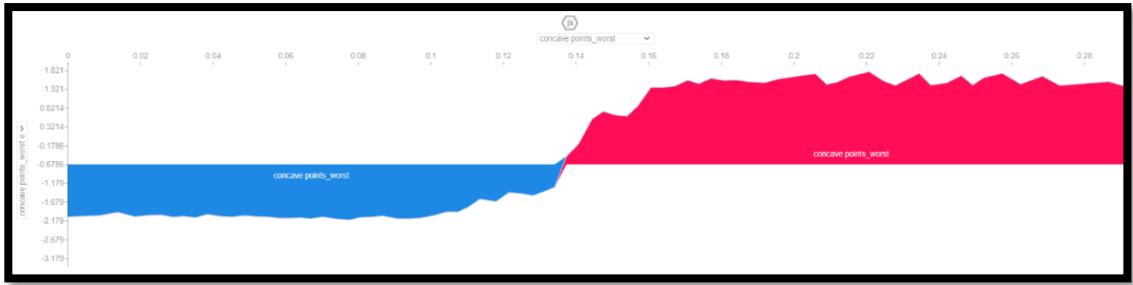
شکل ۲۸-۵ رسم نمودار نیرو به صورت پشته



شکل ۲۹-۵ نمودار نیرو به صورت پشته

این نمودار این قابلیت را دارد که متناسب با یک ویژگی خاص برای تمام رکوردها ترسیم شود. در شکل زیر، این نمودار برای پارامتری که طبق نمودار پیرسون، بیشترین اهمیت را دارد؛ ترسیم شده است (شکل ۳۰-۵).

¹ Stacked force plot



شکل ۳۰-۵ نمودار نیرو به صورت پشته برای ویژگی بینرین حالت نقاط مقعر

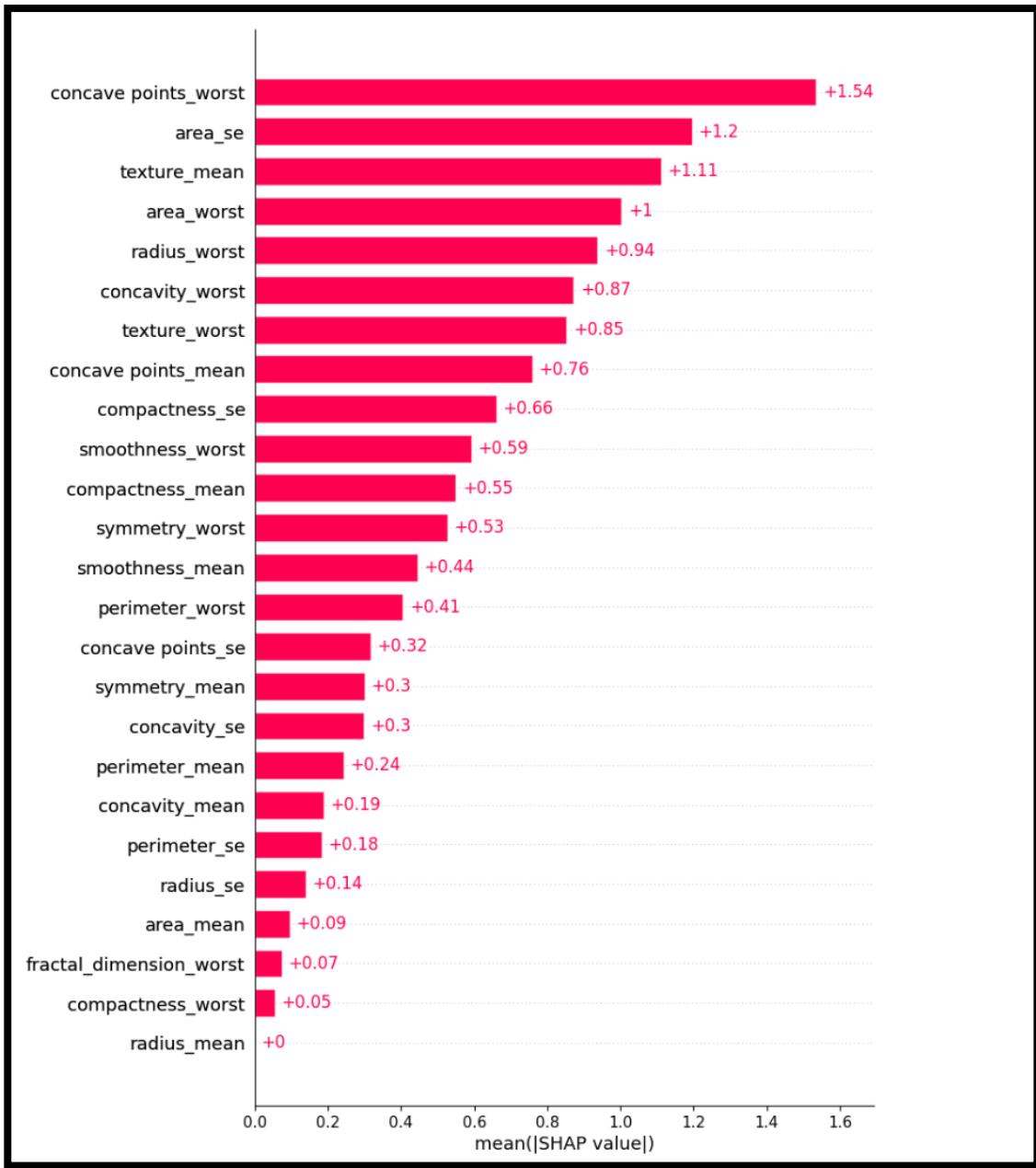
^۱۴-۸ نمودار میله‌ای^۱

این نمودار نیز ویژگی‌های موثر در پیش‌بینی نتیجه برای تمام رکوردها را بصورت عمومی نشان می‌دهد. مقادیر نمایش داده شده در این نمودار به این صورت به دست می‌آیند که ابتدا قدر مطلق تمام مقادیر شپ برای تمام رکوردها در هر ویژگی محاسبه شده و سپس میانگین این مقادیر برای هر ویژگی به دست می‌آید. بزرگی این عدد نمایانگر آن است که ویژگی مربوطه بطور کلی تاثیر مهمی در پیش‌بینی نهایی برای تمامی رکوردها داشته است (شکل ۳۱-۵) (شکل ۳۲-۵).



شکل ۳۱-۵ رسم نمودار میله‌ای

¹ Bar plot



شکل ۳۲-۵ نمودار میله‌ای^۱

۸-۵ - نمودار ازدحام زنبورها^۱

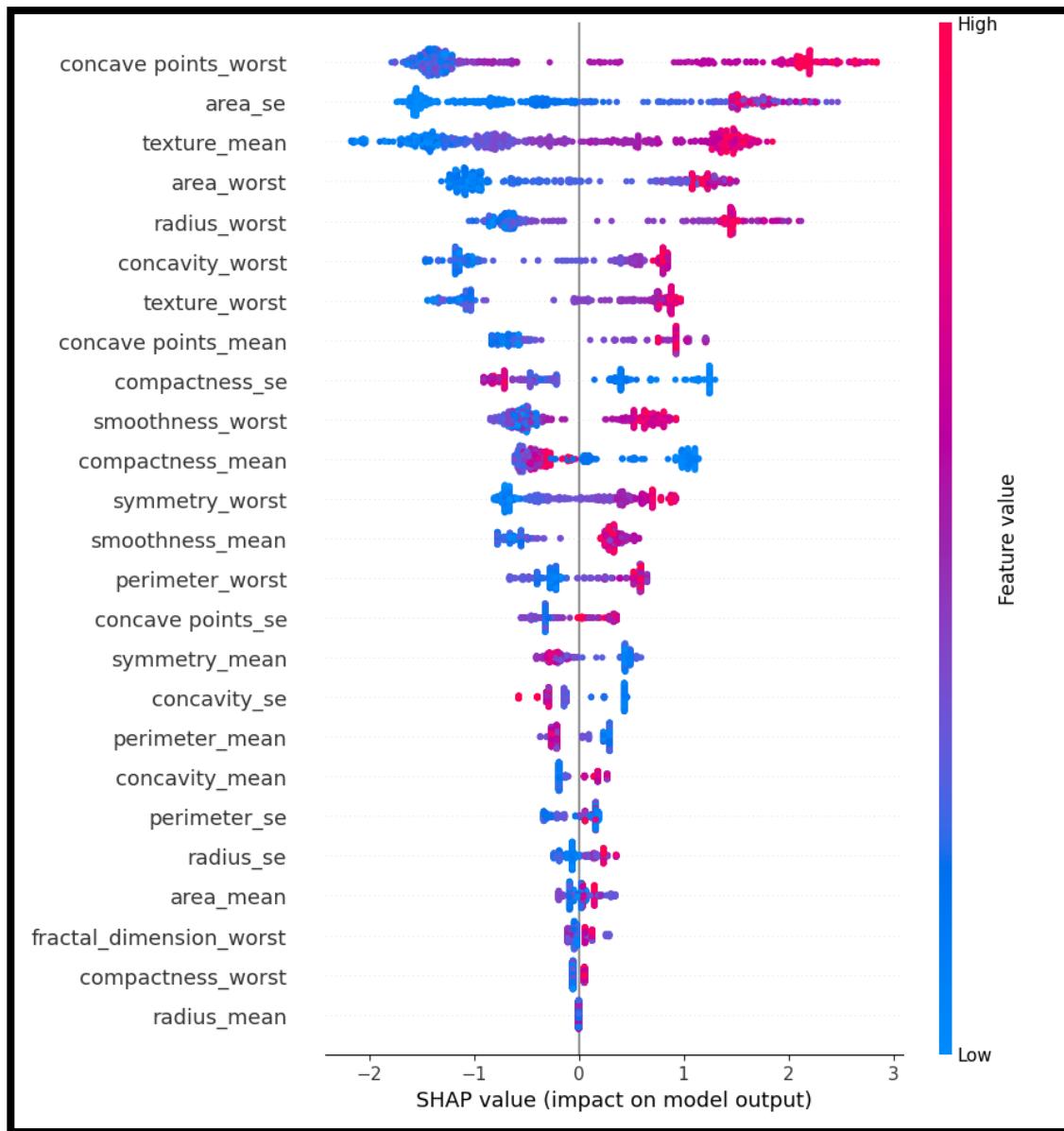
این نمودار نیز ویژگی‌های موثر در پیش‌بینی نتیجه برای تمام رکوردها را بصورت عمومی نشان می‌دهد. تفسیر این نمودار به این صورت است که برای هر ویژگی نقاط قرمز به معنی مقدار بالای آن ویژگی است که در صورتی که در سمت راست خط عمود باشد، بیانگر تاثیر مثبت مقادیر بالای آن ویژگی روی خروجی

¹ Bee swarm

مدل است. بالعکس، در صورتی که نقاط قرمز در سمت چپ خط عمود باشد، نشان‌دهنده تاثیر منفی مقادیر بالای آن ویژگی روی خروجی مدل است. به طور مشابه، نقاط آبی بیانگر مقدار پایین برای آن ویژگی است که در صورتی که در سمت راست خط عمود باشد، بیانگر تاثیر مثبت مقادیر پایین آن ویژگی و در صورتی که در سمت چپ خط عمود باشد، بیانگر تاثیر منفی مقادیر پایین آن ویژگی روی خروجی مدل است. بزرگی این تاثیر از فاصله نقطه تا خط عمود فهمیده می‌شود. هر نقطه بیانگر یک رکورد از مجموعه داده است (شکل ۳۴-۵). (شکل ۳۴-۵)



شکل ۳۴-۵ رسم نمودار از دحام زنبورها



شکل ۵-۴ نمودار ازدحام زنیورها

همانطور که از دو نمودار آخر این قسمت مشخص است، ویژگی بدترین حالت نقاط مقعر^۱ مهم‌ترین ویژگی در پیش‌بینی نتیجه نهایی مدل است. این نتیجه با نتیجه حاصل از نمودار پیرسون همخوانی دارد.

^۱ Concave points worse

فصل ششم

۶ پیاده‌سازی مدل ماشین افزایش گرادیان سبک (LGBM) با پارامترهای پیش‌فرض

در فصل‌های قبل با استفاده از مدل افزایش گرادیان مفرط، تلاش کردیم پیش‌بینی‌هایی برای ارزیابی وضعیت و خامت بیماران مبتلا به سرطان سینه، انجام دهیم.

در فصل‌های آتی با بهره‌گیری از مدل ماشین افزایش گرادیان سبک، به دنبال آن هستیم تا این مدل را در انجام پیش‌بینی‌های جدید محک بزنیم.

در این فصل از مدل ماشین افزایش گرادیان سبک با پارامترهای پیش‌فرض برای پیش‌بینی وضعیت بیماران استفاده شده است؛ در فصل‌های آینده تلاش می‌کنیم تا با تنظیم پارامترهای مدل، دقت پیش‌بینی‌های مدل مدل ماشین افزایش گرادیان سبک را افزایش دهیم.

۶- ۱ آموزش مدل ماشین افزایش گرادیان سبک با داده‌های مجموعه آموزش

مشابه فصل سوم، مدل را از کتابخانه مربوطه به برنامه اضافه کرده و سپس از آن یک طبقه‌بند تعریف می‌کنیم. سپس داده‌های مربوط به مجموعه آموزش را به این طبقه‌بند ورودی داده و مدت کوتاهی منتظر می‌مانیم تا مدل آموزش ببیند (شکل ۶-۱).



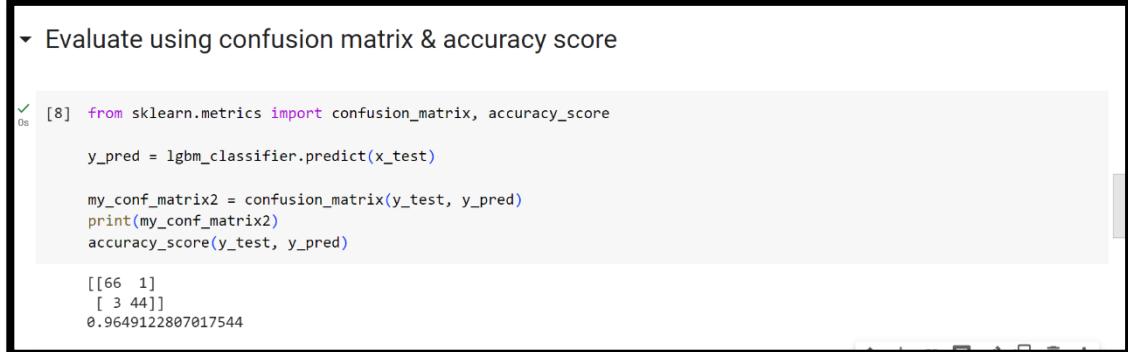
```
[ ] import lightgbm as ltb
lgbm_classifier = ltb.LGBMClassifier()
lgbm_classifier.fit(x_train, y_train)

- LGBMClassifier
LGBMClassifier()
```

شکل ۶-۱ آموزش مدل ماشین افزایش گرادیان سبک با پارامترهای پیش‌فرض

۶- ۲ ارزیابی مدل

۶- ۲ - ۱ ارزیابی مدل با استفاده از ماتریس درهم ریختگی و امتیاز صحت مشابه قسمت متناظر در فصل سوم، با فراخوانیتابع پیش‌بینی، مقدار متغیر وابسته‌ی پیش‌بینی را می‌یابیم. سپس مقادیر ماتریس درهم ریختگی و امتیاز صحت را یافته و نمایش می‌دهیم (شکل ۶-۲).



```
✓ [8] from sklearn.metrics import confusion_matrix, accuracy_score

y_pred = lgbm_classifier.predict(x_test)

my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)

[[66  1]
 [ 3 44]]
0.9649122807017544
```

شکل ۶-۲ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

۶- ۲ - ۲ اعمال اعتبار سنجی متقابل چند دسته‌ای

مشابه قسمت متناظر در فصل سوم، مقادیر اعتبار سنجی متقابل چند دسته‌ای را یافته و نمایش می‌دهیم (شکل ۶-۳).

```
[9] from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator=lgbm_classifier, X=x_train, y=y_train, cv=10)

[10] print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))

Accuracy: 96.93 %
Standard Deviation: 2.45 %
```

شکل ۶-۳ ارزیابی مدل با اعتبارسنجی متقابل

با در اختیار داشتن نتیجه به دست آمده است ماتریس درهم ریختگی و امتیاز صحت و اعتبارسنجی متقابل چند دسته‌ای برای مدل آموزش دیده با متغیرهای پیش‌فرض، اکنون زمان آن رسیده که با تنظیم پارامترهای مختلف به دقتی بالاتر از دقت پیش‌فرض دست یابیم که این هدف را در فصل بعد دنبال می‌کنیم.

فصل هفتم

۷ پیاده‌سازی مدل ماشین افزايش گرادیان سبک (LGBM) با پارامترهای تنظیم شده

در این فصل سعی شده با تنظیم پارامترهای مهم و تاثیرگذار در مدل ماشین افزايش گرادیان سبک، دقیقی بالاتر از دقت مدل با پارامترهای پیش‌فرض معرفی شده در فصل قبل به دست آوریم. برای تحقق این منظور، ابتدا به سراغ پارامترهای تعداد دور و نرخ یادگیری رفته و پس از به دست آوردن مقدار بهینه برای پارامترها، از این مقادیر در به دست آوردن مقدار بهینه برای پارامترهای زیرنمونه، نمونه ستون برای هر درخت و حداقل داده در برگ استفاده کردہ‌ایم.

همچنین نمودارهای راک و دقت-حساسیت و نمودارهای شب برای ارائه اطلاعات بیشتر در مورد حالت بهینه نهایی ترسیم شده است.

۷-۱ آزمایش پارامترهای مختلف در آموزش مدل

ابتدا لیستی از پارامترهای مهم مدل ماشین افزایش گرادیان سبک را تهیه کرده و مناسب با مقادیر پیشنهادی در منابع مقداردهی می‌کنیم [26] [27] [28] [29].

پارامتر اول تعداد دور^۱ یا همان تعداد درختها (معادل تعداد برآوردها در مدل تقویت گرادیان مفرط) است. این مدل دو پارامتر جدید به نام‌های تعداد برگ^۲ و حداقل داده در برگ^۳ را دارد. پارامتر تعداد برگ باید مناسب با حداکثر عمق تنظیم گردد. پارامتر دیگری که مرتبط با پارامتر زیرنمونه است به نام بسامد کیسه‌گیری^۴ وجود دارد که تعداد اعمال زیرنمونه را پس از هر چند دور مشخص می‌کند.

همچنین تعدادی از پارامترها به صورت کلی باید مقداردهی شوند که این پارامترها را در هنگام تعریف طبقه‌بند مقداردهی کردیم. این پارامترها عبارتند از هدف، معیار ارزیابی، متعادل بودن یا نبودن مجموعه داده، و روش تقویت است که مقادیر آنها به ترتیب طبقه‌بندی دوتایی، مساحت زیر نمودار (رایج است از مساحت زیر نمودار راک استفاده شود)، متعادل بودن مجموعه داده و روش درخت تصمیم تقویت گرادیان هستند (شکل ۱-۷).

```
from sklearn.model_selection import GridSearchCV
from numpy import arange

classifier = lgb.LGBMClassifier(objective='binary',
                                 metric='auc',
                                 is_unbalance=False,
                                 boosting='gbdt')

parameters = [
    {
        'num_iterations': list(range(50, 151, 1)),
        'learning_rate': arange(0.01, 0.2, 0.01),
        'num_leaves': list(range(10, 100)),
        'subsample': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1],
        'bagging_freq': [1],
        'colsample_bytree': [0.5, 0.6, 0.7, 0.8, 0.9, 1],
        'min_data_in_leaf': list(range(10, 100)),
        'max_depth': list(range(1, 20))
    },
    {
        'random_state': [0]
    }
]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(x_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.10f} %".format(best_accuracy*100))
print("Best Parameters:", best_parameters)
```

شکل ۱-۷ ۱ مجموعه فضای جستجو پارامترهای مدل ماشین افزایش گرادیان سبک

۷-۲ تخمین زمان اجرای تکه کد ذکر شده

به علتی که در فصل پنجم بیان شد، واضح است که نمی‌توان تمام مقادیر ذکر شده را در یک اجرای واحد مورد آزمون قرار داد. اگر تصمیم به انجام چنین کاری داشته باشیم (با توجه به اینکه این مدل سبک است و هر اجرای آن حدود ۰.۳ ثانیه طول می‌کشد)، مطابق با فرمولی که در فصل پنجم بیان شد زمان تخمینی

¹ Num_iterations

² Num_leaves

³ Min_data_in_leaf

⁴ Bagging frequency

چیزی حدود ۱۱۶۶۴۰۰۰۰۰ یا ۳۷۰ سال طول می‌کشد که مجدداً این مقدار از عمر پژوهشگر و احتمالاً عمر خوانندگان این پایان‌نامه بیشتر است. بنابراین دوباره ناچار هستیم کار آزمایش پارامترهای مختلف در آموزش مدل را به صورت هدفمندانه‌تر و به دور از روش جستجوی فراگیر انجام دهیم تا زمان مورد نیاز برای محاسبات تا حد چند ساعت پایین بیاید.

مجدداً تاکید می‌شود که احتمالاً نتیجه بهینه با نتیجه بهینه در واقعیت متفاوت است؛ چرا که تمام ترکیبات ممکن مقادیر مشخص شده برای پارامترها، آزمایش نشده است؛ ولی تا زمانی که سیستم‌های پردازشگر به قدرت مورد نیاز برای پردازش این حجم عظیم از محاسبات نرسیده‌اند، ناگزیر به استفاده از این روش هستیم.

۷-۳ محاسبه مقدار بهینه برای تعداد دور و نرخ یادگیری

دوتا از مهم‌ترین پارامترهای مدل ماشین افزایش گرادیان سبک، تعداد دور و نرخ یادگیری هستند. در اولین تلاش برای بهبود عملکرد مدل، پژوهشگر بر روی این دو پارامتر تمرکز کرده تا مقدار بهینه آنها را به دست آورد. با در نظر نگرفتن مقدار برای پارامترهای دیگر، مقدار پیش‌فرض آنها در نظر گرفته خواهد شد. در این تلاش، مقدار امتحانی برای تعداد برآورده‌گرها، بازه اعداد [۵۰, ۱۵۱] با گام یک واحد و برای نرخ یادگیری، این بازه برابر با [۰.۰۱, ۰.۲] با گام یک صدم واحد است (شکل ۷-۲).

```

Finding Best Hyperparameters

[ ] from sklearn.model_selection import GridSearchCV
from numpy import arange

classifier = ltb.IGBCClassifier(objective='binary',
                                 metric='auc',
                                 is_unbalance=False,
                                 boosting='gbdt')

parameters = [
    'num_iterations': list(range(50, 151, 1)),
    'learning_rate': arange(0.01, 0.2, 0.01),
    'random_state': [0]
]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(X_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.10f} %".format(best_accuracy*100))
print("Best Parameters:", best_parameters)

[LightGBM] [Warning] boosting is set-gbdt, boosting_type=gbdt will be ignored. Current value: boosting=gbdt
Best Accuracy: 0.815904200 %
Best Parameters: {'learning_rate': 0.10, 'num_iterations': 149, 'random_state': 0}
Found 'num_iterations' in params. Will use it instead of argument

```

شکل ۷-۲ پیدا کردن مقدار بهینه برای پارامترهای تعداد دور و نرخ یادگیری

همانطور که در تصویر مشاهده می‌شود، مقدار بهینه برای پارامترهای تعداد دور و نرخ یادگیری به ترتیب برابر با ۱۴۹ و ۰/۱۹ است. از این مقادیر در دورهای بعدی برای بهبود عملکرد با اضافه کردن امتحان کردن پارامترهای دیگر استفاده می‌کنیم.

۷-۴ آموزش مجدد مدل با پارامترهای بهینه

مشابه روشی که برای مدل تقویت گرادیان مفرط انجام دادیم، پارامترهای بهینه به دست آمده که در ساختار داده لغتنامه ذخیره شده است را به کلاس ورودی داده و مدل جدیدی را مطابق با پارامترهای بهینه می‌سازیم. سپس مدل را با مجموعه داده‌های مستقل آموزش ووابسته آموزش مجددآموزش می‌دهیم (شکل ۳-۷).



```
[ ] lgbm_classifier = ltb.LGBMClassifier(objective='binary',
                                         metric='auc',
                                         is_unbalance=False,
                                         boosting='gbdt',
                                         **best_parameters)
lgbm_classifier.fit(x_train, y_train)

Found `num_iterations` in params. Will use it instead of argument
*
LGBMClassifier
LGBMClassifier(boosting='gbdt', is_unbalance=False, learning_rate=0.19,
               metric='auc', num_iterations=149, objective='binary',
               random_state=0)
```

شکل ۳-۷ آموزش مجدد مدل با مقاییر بهینه

۷-۵ ارزیابی مدل

۷-۵-۱ ارزیابی مدل با اعتبارسنجی متقابل چند دسته‌ای

مجددآعتبارسنجی متقابل چند دسته‌ای را بر روی مدل به دست آمده در این اجرا اعمال می‌کنیم. در مقایسه نتایج با نتایج دور قبلی که برای مدل با پارامترهای پیش‌فرض انجام شده بود، مشاهده می‌شود که معیارهای دقت و انحراف از معیار به میزان قابل توجهی بهبود داشته‌اند. میزان دقت از ۹۶/۹۳ به ۹۷/۸۱ افزایش و میزان انحراف از معیار از ۲/۴۵ به ۲/۱۸ کاهش داشته است (شکل ۴-۷).



```
[ ] from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator=lgbm_classifier, X=x_train, y=y_train, cv=10)

[ ] print("Accuracy: {:.10f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.10f} %".format(accuracies.std()*100))

Accuracy: 97.8115942029 %
Standard Deviation: 2.1837725959 %
```

شکل ۷-۴ ارزیابی مدل با اعتبارسنجی متقابل

۷-۵-۲ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

با محاسبه ماتریس درهم ریختگی و امتیاز صحت برای مدل جدید، مشاهده می‌شود نتیجه پیش‌بینی‌های مدل جدید بر روی داده‌های آزمایش تغییری نسبت به مدل با پارامترهای پیش‌فرض نداشته است. همچنین

تعداد پیش‌بینی‌های در دسته منفی غلط برابر با ۳ و پیش‌بینی‌های در دسته مثبت غلط برابر با ۱ است (شکل ۵-۷).

```
✓ Evaluate using confusion matrix & accuracy score

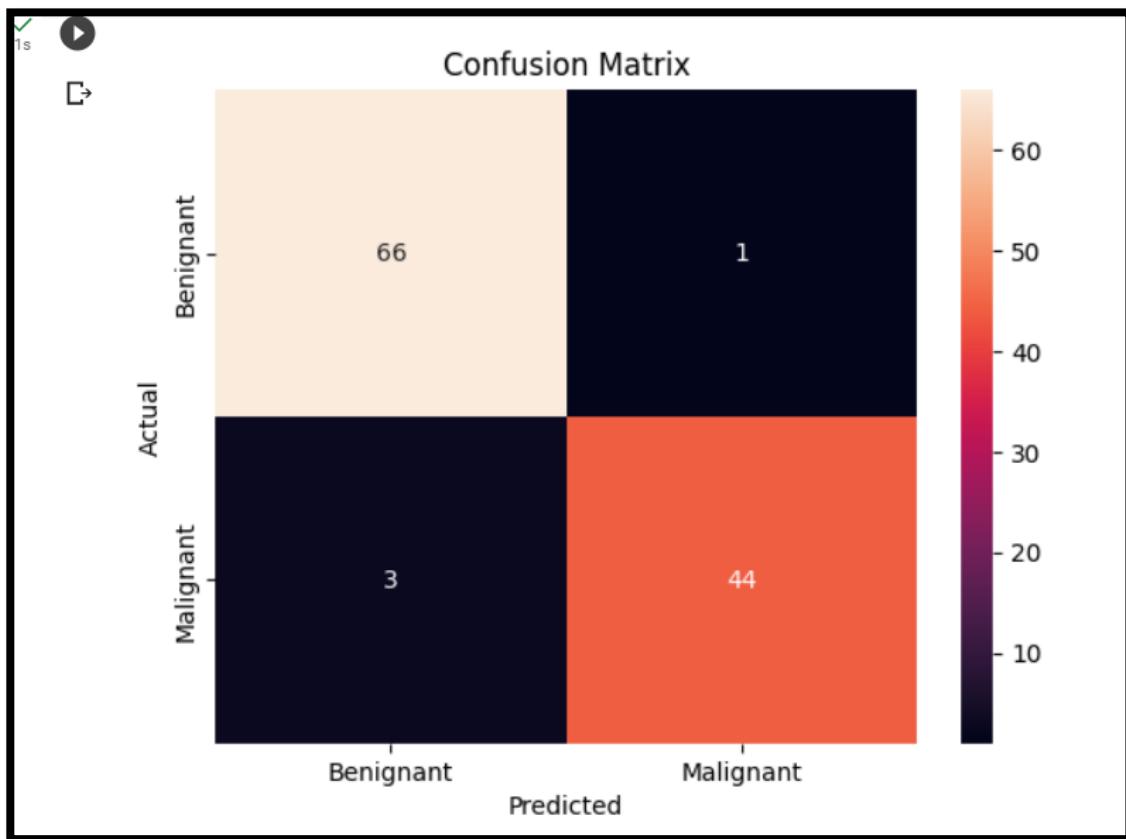
▶ from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = lgbm_classifier.predict(x_test)

my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)

[[66  1]
 [ 3 44]]
0.9649122807017544
```

شکل ۵-۷ ارزیابی مدل با ماتریس درهم‌ریختگی و امتیاز صحت

۵-۵ نمایش ماتریس درهم ریختگی به همراه برچسب با استفاده از کد مشابه در قسمت متناظر این بخش در فصل پنجم، ماتریس درهم ریختگی به همراه برچسب را ترسیم می‌کنیم (شکل ۶-۷).



شکل ۶-۷ ماتریس درهم‌ریختگی با برچسب

۴ - ۵ گزارش دسته‌بندی

مشابه قسمت متناظر در فصل پنجم، گزارش دسته‌بندی به صورت زیر بdst می‌آید (شکل ۷-۷).

```
%matplotlib inline
import numpy as np
from sklearn import datasets, linear_model, metrics
from sklearn.model_selection import train_test_split

# Import scikit-learn datasets
from sklearn import datasets

# Import linear regression model
from sklearn import linear_model

# Import metrics
from sklearn.metrics import r2_score

# Load data
diabetes = datasets.load_diabetes()

# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]

# Split the data into training = 114 samples and test = 47 samples
X_train, X_test, y_train, y_test = train_test_split(diabetes_X, diabetes.target, test_size=0.2, random_state=42)

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(X_train, y_train)

# Make predictions using the testing set
y_pred = regr.predict(X_test)

# The coefficient
print('Coefficients: \n', regr.coef_)

# The mean square error
print("Mean squared error: %.2f" % metrics.mean_squared_error(y_test, y_pred))

# Explained variance score
print('Explained variance score: %.2f' % r2_score(y_test, y_pred))

# Evaluating using classification report
```

	precision	recall	f1-score	support
0	0.96	0.99	0.97	67
1	0.98	0.94	0.96	47
accuracy			0.96	114
macro avg	0.97	0.96	0.96	114
weighted avg	0.97	0.96	0.96	114

شکل ۷-۷ گزارش دسته‌بندی

طبق گزارش آورده شده در جدول بالا، ۹۶ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم به درستی دسته‌بندی شده است. این آمار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است.

معیار بعدی میزان یادآوری است. همانطور که از جدول مشاهده می‌شود، مدل ما توانسته ۹۹ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم را از بین مجموعه داده‌های این دسته پیدا کند. این مقدار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۴ درصد است. همانطور که از جدول مشاهده می‌شود امتیاز ۱۱ برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۷ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۶ درصد است. این آمار نشان می‌دهد به طور میانگین مدل ما در ارزیابی بیماران سرطانی خوش‌خیم کمی موفق‌تر است. صحت دسته‌بندی‌ها نیز مطابق با جدول برابر با ۹۷ است.

همانطور که مشخص است جدول گزارش دسته‌بندی نشان می‌دهد به طور کلی دقت مدل جدید نسبت به مدل قبل افزایش داشته است.

۷ - ۵ محاسبه معیارهای اختصاصیت و حساسیت

همانطور که از مقادیر داخل شکل زیر مشاهده می‌شود؛ میزان اختصاصیت برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۳ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است. این مقدار نسبت منفی‌های درست برای کلاس‌های ذکر شده را بیان می‌دارد (شکل ۷-۸).

```

▼ Specificity & Sensitivity Score

[14]: from imblearn.metrics import sensitivity_specificity_support
pd.DataFrame(sensitivity_specificity_support(y_test, y_pred, average=None), index=['Sensitivity', 'Specificity', 'Support'])

Sensitivity Specificity Support
0    0.985075    0.936170    67.0
1    0.936170    0.985075    47.0

```

شکل ۱-۷ محاسبه معیارهای اختصاصیت و حساسیت

۷-۶ بهبود عملکرد مدل با تنظیم پارامترهای زیرنمونه، نمونه ستون برای هر درخت و حداقل داده در برگ

در تلاش دوم پژوهشگر به سراغ پارامترهای حداقل داده در برگ، زیرنمونه و نمونه ستون برای هر درخت رفته و با تنظیم مقدار آنها به دقتی بالاتر است دقت مدل با پارامترهای پیش‌فرض دست یافت (شکل ۹-۷).

```

from sklearn.model_selection import GridSearchCV
from numpy import arange

classifier = ltb.LGBMClassifier(objective='binary',
                                 metric='auc',
                                 is_unbalance=False,
                                 boosting='gbdt')

parameters = [
    'num_iterations': [149],
    'learning_rate': [0.19],
    'subsample': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'bagging_freq': [1],
    'colsample_bytree': [0.5, 0.6, 0.7, 0.8, 0.9, 1],
    'min_data_in_leaf': list(range(10, 100)),
    'random_state': [0]
]

grid_search = GridSearchCV(
    estimator = classifier,
    param_grid = parameters,
    scoring = 'accuracy',
    cv = 10,
    n_jobs = -1
)

grid_search.fit(x_train, y_train)
best_accuracy = grid_search.best_score_
best_parameters = grid_search.best_params_
print("Best Accuracy: {:.10f} %".format(best_accuracy*100))
print("Best Parameters: ", best_parameters)

D: Found 'num_iterations' in params. Will use it instead of argument.
[LightGBM] [Warning] boosting is set=gbdt, boosting_type=gbdt will be ignored. Current value: boosting-gbdt
[LightGBM] [Warning] bagging_freq is set=1, subsample_freq=0 will be ignored. Current value: bagging_freq=1
[LightGBM] [Warning] min_data_in_leaf is set=20, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=20
Best Accuracy: 98.2512077295 %
Best Parameters: {'bagging_freq': 1, 'colsample_bytree': 0.5, 'learning_rate': 0.19, 'min_data_in_leaf': 20, 'num_iterations': 149, 'random_state': 0, 'subsample': 1}

```

شکل ۹-۷ پیدا کردن مقدار بهینه برای پارامترهای زیرنمونه، نمونه ستون برای هر درخت و حداقل داده در برگ

همانطور که در شکل مشاهده می‌شود، با استفاده از پارامترهای به دست آمده از اجرای قبلی، در این اجرا مجدداً مدل را برای چند ساعت با مقادیر مختلف برای پارامترهای مذکور آزمایش کردیم و مقادیر بهینه به دست آمده برای پارامترها در تصویر قابل مشاهده است. مدل به دست آمده پس از این اجرا از مدل قبلی به میزان حدود ۴۴/۰ درصد عملکرد بهتری دارد.

اکنون کافی است مدل را مجدداً با پارامترهای بهینه به دست آمده آموزش دهیم (شکل ۱۰-۷).

▼ Re-training model with optimal parameters

```
[ ] lgbm_classifier = ltb.LGBMClassifier(objective='binary',
                                         metric='auc',
                                         is_unbalance=False,
                                         boosting='gbdt',
                                         **best_parameters)
lgbm_classifier.fit(x_train, y_train)

Found `num_iterations` in params. Will use it instead of argument
[ ] LGBMClassifier
LGBMClassifier(bagging_freq=1, boosting='gbdt', colsample_bytree=0.5,
                is_unbalance=False, learning_rate=0.19, metric='auc',
                min_data_in_leaf=20, num_iterations=149, objective='binary',
                random_state=0, subsample=1)
```

شکل ۷-۱۰ آموزش مجدد مدل با مقادیر بهینه

۷-۷ ارزیابی مدل

۷-۷-۱ ارزیابی مدل با اعتبار سنجی متقابل چند دسته‌ای

می‌توان مجدداً اعتبار سنجی متقابل چند دسته‌ای را روی مدل حاصل اعمال کرد. نتایج به دست آمده به صورت زیر است (شکل ۷-۱۱).

▼ Applying k-fold cross validation

```
[ ] from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator=lgbm_classifier, X=x_train, y=y_train, cv=10)

[ ] print("Accuracy: {:.10f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.10f} %".format(accuracies.std()*100))

Accuracy: 98.2512077295 %
Standard Deviation: 1.8974871331 %
```

شکل ۷-۱۱ ارزیابی مدل با اعتبار سنجی متقابل

در مقایسه با مدل اجرای قبل، مشاهده می‌شود مقدار دقت مدل از ۹۷/۸۱ واحد به مقدار ۹۸/۲۵ افزایش یافته و انحراف از معیار نیز از مقدار ۲/۱۸ به مقدار ۱/۸۹ کاهش یافته است.

در مقایسه با مدل اولیه با پارامترهای پیش‌فرض، مشاهده می‌شود مقدار دقت مدل از ۹۶/۹۳ واحد به مقدار ۹۸/۲۵ افزایش یافته و انحراف از معیار نیز از مقدار ۲/۴۵ به مقدار ۱/۸۹ کاهش یافته است. همانطور که مشخص است، مدل فعلی در مقایسه با هر دو مدل قبلی عملکرد بهتری دارد.

۷-۷-۲ ارزیابی مدل با ماتریس درهم‌ریختگی و امتیاز صحت

ماتریس درهم‌ریختگی و امتیاز صحت را مجدداً محاسبه می‌کنیم (شکل ۷-۱۲).

▼ Evaluate using confusion matrix & accuracy score

```
[23] from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = lgbm_classifier.predict(x_test)

my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)

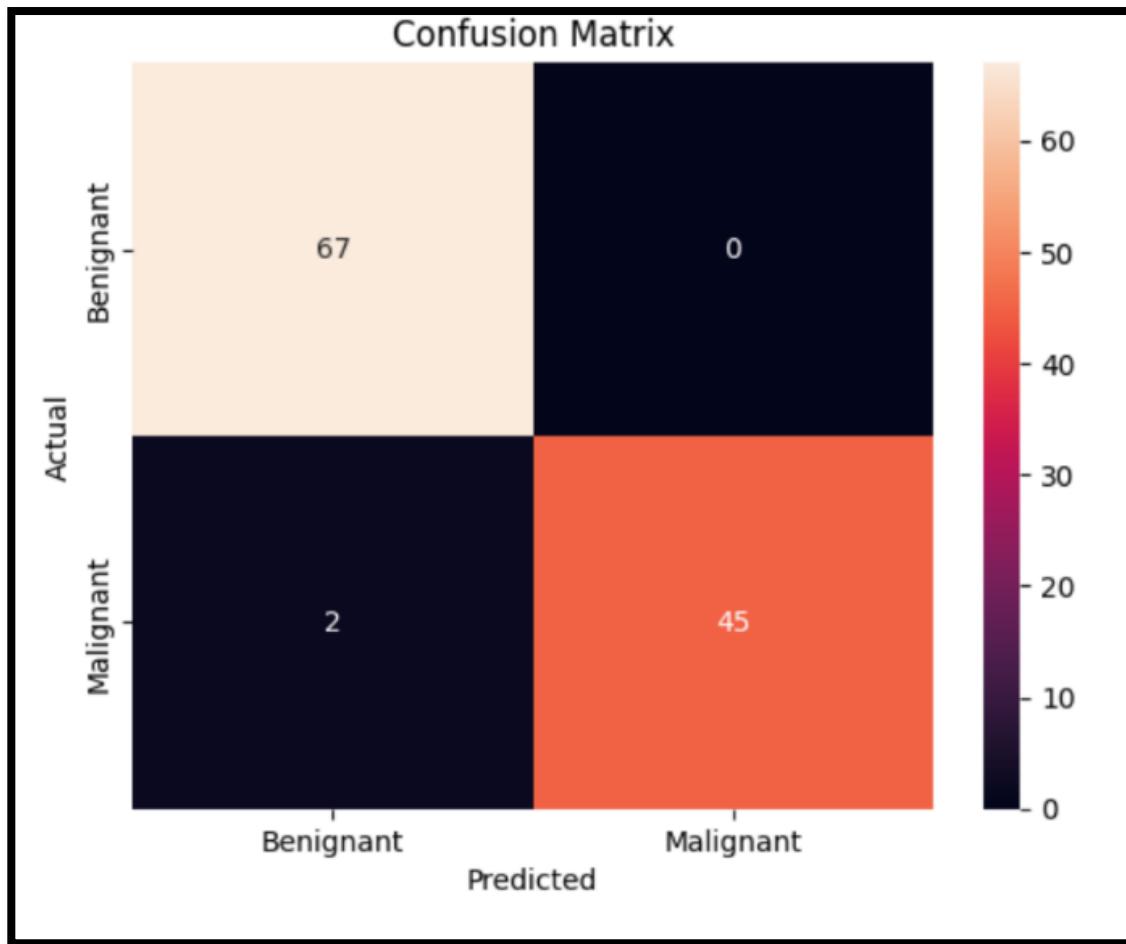
[[67  0]
 [ 2 45]]
0.9824561403508771
```

شکل ۱۲-۷ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

در مقایسه با مدل اجرای قبل، مشاهده می شود که نتایج بهبود داشته است. از تعداد منفی غلط و مثبت غلط در ماتریس درهم ریختگی هر کدام یک واحد کم شده است. همچنین امتیاز صحت از ۰/۹۶۴۹ در اجرای قبل به مقدار ۰/۹۸۲۴ رسیده است.

در مقایسه با مدل اولیه با پارامترهای پیش فرض نیز بطور مشابه، مشاهده می شود از تعداد منفی غلط و مثبت غلط در ماتریس درهم ریختگی هر کدام یک واحد کم شده است. همچنین امتیاز صحت از ۰/۹۶۴۹ در اجرای قبل به مقدار ۰/۹۸۲۴ رسیده است.

۷-۳ - نمایش ماتریس درهم ریختگی به همراه برچسب با استفاده از تکه کد مشابه قسمت متناظر در اجرای قبل، ماتریس درهم ریختگی به همراه برچسب بصورت زیر بدست می آید (شکل ۱۳-۷).



شکل ۷-۱۳ - ماتریس در هم ریختگی با برچسب

۷-۷ - ۴ گزارش دسته بندی

گزارش دسته بندی در مدل جدید به شرح زیر است (شکل ۷-۱۴).

• Evaluating using classification report

```
[25]: print(metrics.classification_report(y_test, y_pred))

      precision    recall  f1-score   support

          0       0.97     1.00     0.99      67
          1       1.00     0.96     0.98      47

   accuracy                           0.98      114
  macro avg       0.99     0.98     0.98      114
weighted avg       0.98     0.98     0.98      114
```

شکل ۷-۱۴ - گزارش دسته بندی

طبق گزارش آورده شده در جدول بالا، ۹۷ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم به درستی دسته‌بندی شده است. این آمار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۱۰۰ درصد است.

معیار بعدی میزان یادآوری است. همانطور که از جدول مشاهده می‌شود، مدل ما توانسته ۱۰۰ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم را از بین مجموعه داده‌های این دسته پیدا کند. این مقدار برای داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم برابر با ۹۶ درصد است.

همانطور که از جدول مشاهده می‌شود امتیاز ۴۱ برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۹ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است. این آمار نشان می‌دهد به طور میانگین مدل ما در ارزیابی بیماران سرطانی خوش‌خیم کمی موفق‌تر است. صحت دسته‌بندی‌ها نیز مطابق با جدول برابر با ۹۸ است.

همانطور که مشخص است جدول گزارش دسته‌بندی نشان می‌دهد به طور کلی دقت مدل جدید نسبت به مدل قبل ۲٪ افزایش داشته است.

۷-۵ محاسبه معیارهای اختصاصیت و حساسیت

همانطور که از مقادیر داخل شکل زیر مشاهده می‌شود؛ میزان اختصاصیت برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۵ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۱۰۰ درصد است. این مقدار نسبت منفی‌های درست برای کلاس‌های ذکر شده را بیان می‌دارد (شکل ۷-۱۵).

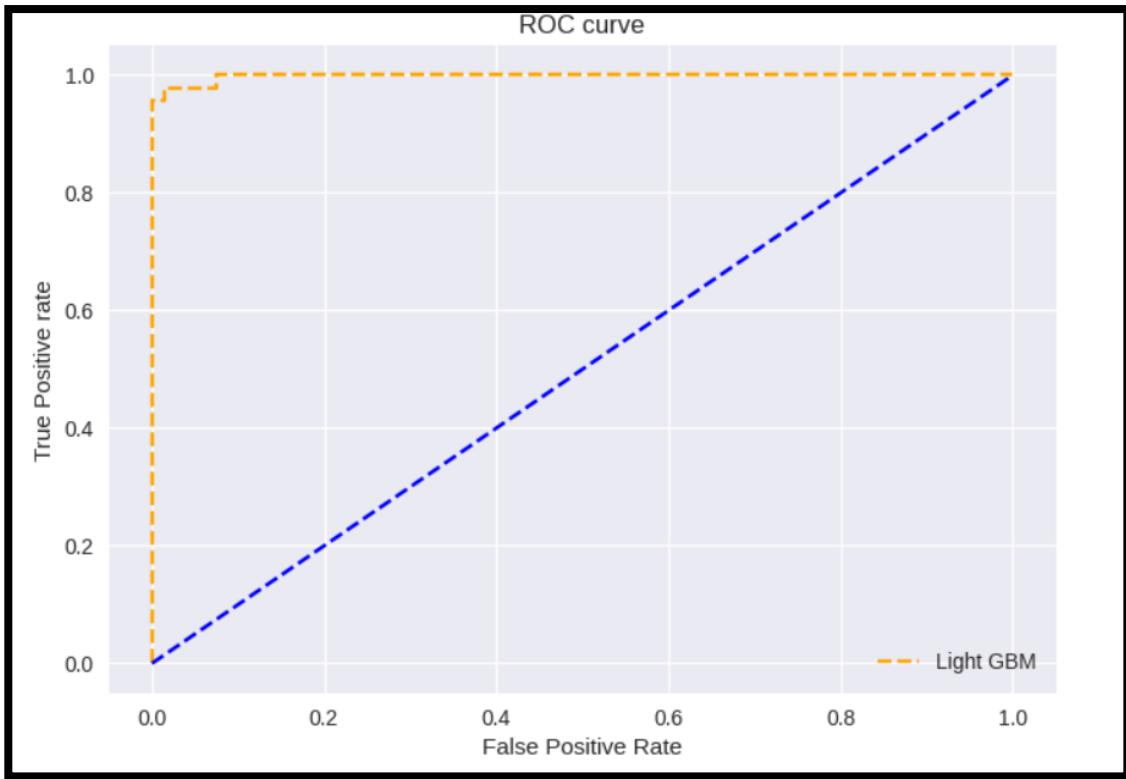
```
[26]: from imblearn.metrics import sensitivity_specificity_support
pd.DataFrame(sensitivity_specificity_support(y_test, y_pred, average=None), index=['Sensitivity', 'Specificity', 'Support']).T
```

	Sensitivity	Specificity	Support
0	1.000000	0.957447	67.0
1	0.957447	1.000000	47.0

شکل ۷-۱۵ محاسبه معیارهای دقت و حساسیت

۷-۶ رسم منحنی ویژگی عملکرد گیرنده (راک)

منحنی ویژگی عملکرد گیرنده ترسیم شده به صورت زیر است (شکل ۷-۱۶).



شکل ۱۶-۷ نمودار راک

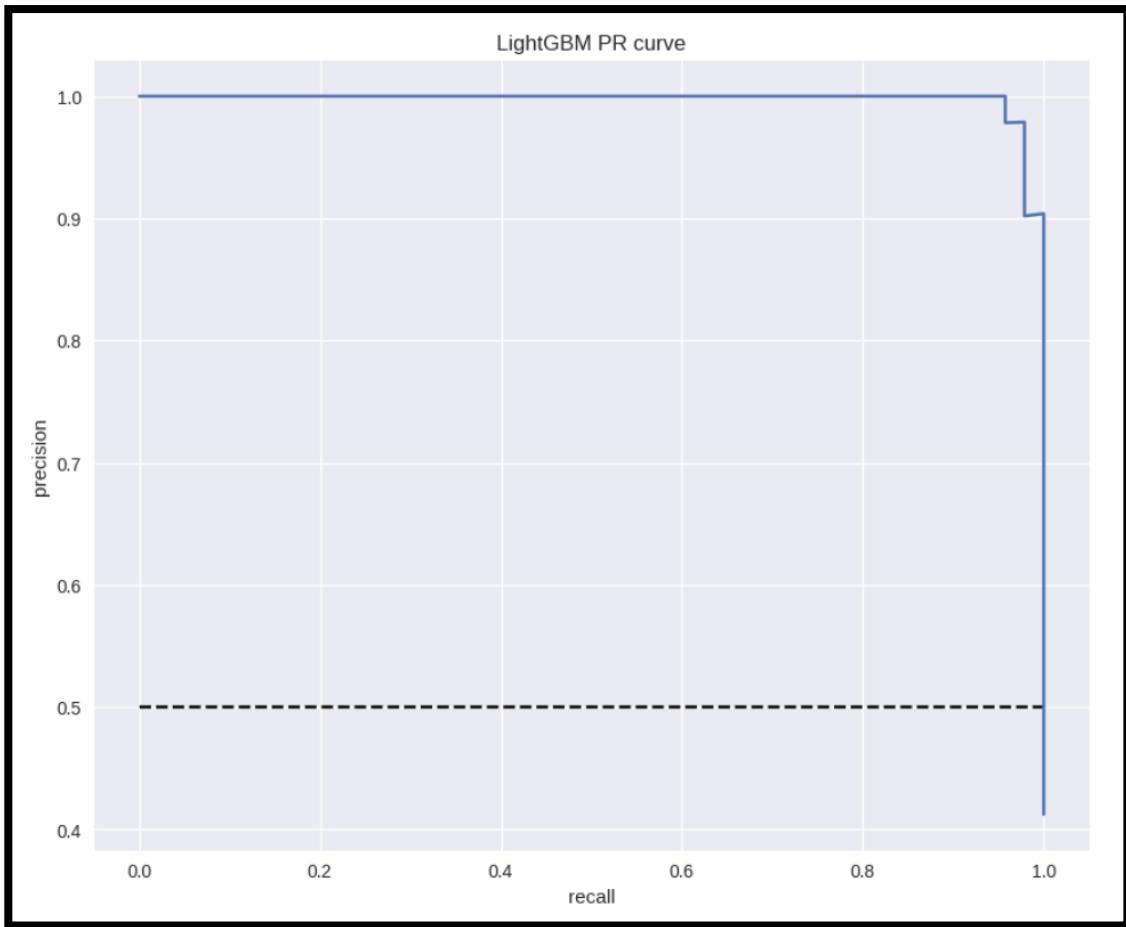
برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۱۷-۷).

```
[30] print("ROC AUC Score: ", roc_auc_score(y_test, y_pred_proba[:,1]))
```

شکل ۱۷-۷ مساحت زیر نمودار راک

همانطور که از شکل مشخص است، میزان مساحت زیر نمودار برابر با 0.9980 است که نزدیکی آن به مقدار واحد بیانگر دقت بالای مدل است.

۷-۷ - ۷ رسم نمودار منحنی دقت-حساسیت منحنی دقت-حساسیت رسم شده به صورت زیر است (شکل ۱۸-۷).



شکل ۱۸-۷ نمودار دقت-حساسیت

برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۱۹-۷).

```

  1 print("PRC AUC Score: ", auc(recall, precision))
  2 PRC AUC Score:  0.997476502255111

```

شکل ۱۹-۷ مساحت زیر نمودار دقت-حساسیت

همانطور که از شکل مشخص است، میزان مساحت زیر نمودار برابر با 0.9974 است که نزدیکی آن به مقدار واحد بیانگر دقت بالای مدل است.

۷-۸ رسم نمودار مقادیر توضیحات افزونی شپلی

حال نمودارهای شپ را ترسیم می‌کنیم.

مشابه قسمت قبل، از توضیح‌دهنده یک شی ساخته و مدل طبقه‌بند را به عنوان پارامتر به کلاس ورودی می‌دهیم. سپس داده‌های مستقل را به شی ساخته شده از توضیح‌دهنده ساخته شده ورودی داده مقادیر شپ را به دست می‌آوریم (شکل ۲۰-۷).

▼ Expainer

```
[11] # Fits the explainer  
explainer = shap.Explainer(lgbm_classifier, feature_names=feature_names)  
# Calculates the SHAP values - It takes some time  
shap_values = explainer(x)
```

شکل ۲۰-۷ تعریف توضیح‌دهنده و یافتن مقادیر شپ

در این قسمت به علت تفاوت در ساختار مدل، مقادیر شپ دو عدد قرینه را شامل می‌شود. برای رسم نمودارها تنها به یکی از این مقادیر نیازمندیم. کافیست با اندیس‌گذاری به این مقدار دست پیدا کنیم (شکل ۲۱-۷).

```
[12] print(shap_values.shape)  
(569, 25, 2)  
  
[13] print(shap_values[:, :, 1].shape)  
(569, 25)  
  
[14] print(shap_values[0].shape)  
(25, 2)  
  
[15] print(shap_values[0][:, 1].shape)  
(25,)
```

شکل ۲۱-۷ ساختار متفاوت مقادیر شپ در مدل ماشین افزایش گردیان سیک

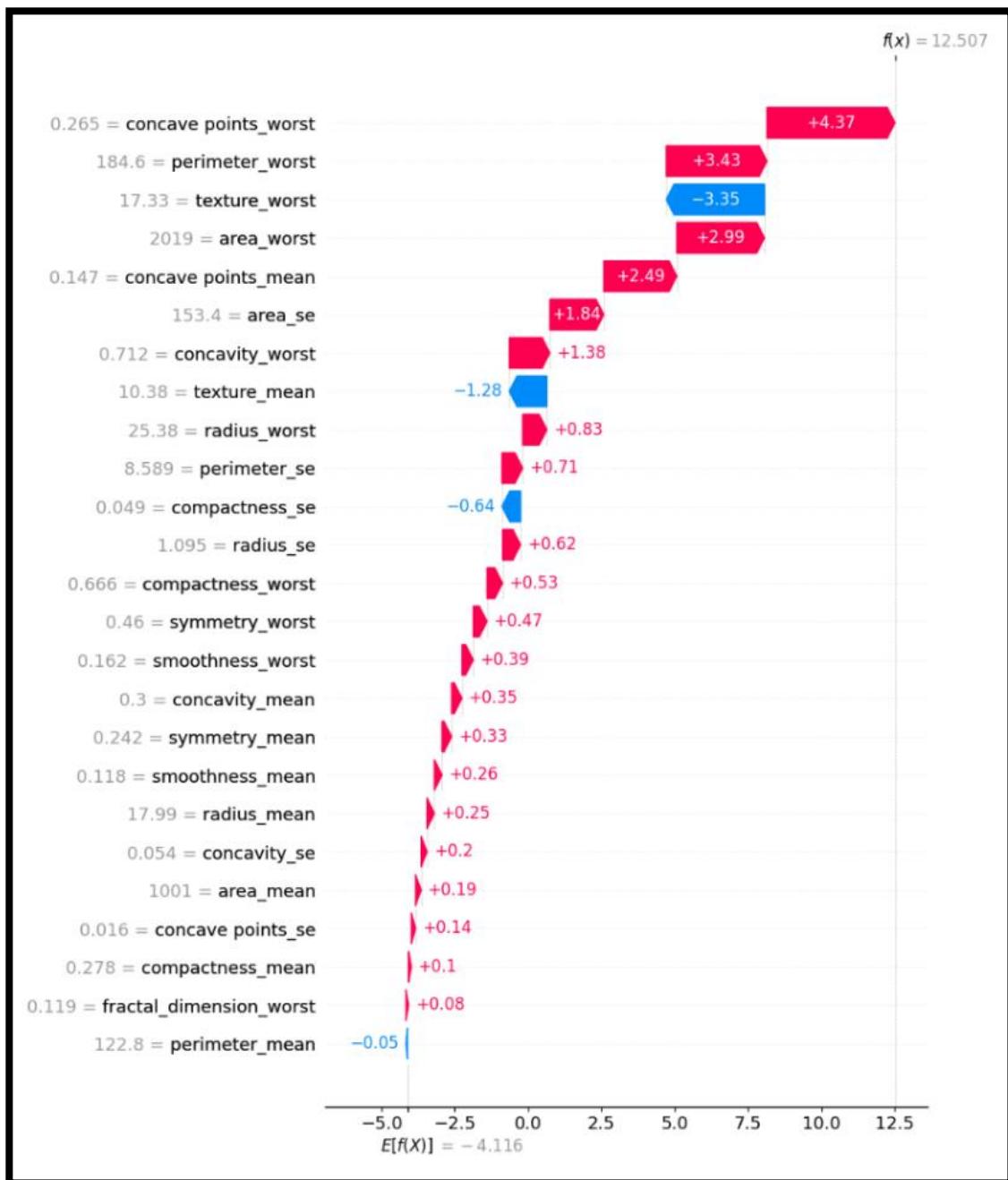
۷-۹ رسم نمودار آبشاری

مشابه قسمت قبل، با استفاده از کد زیر نمودار را رسم می‌کنیم (شکل ۲۲-۷) (شکل ۲۳-۷).

▼ Waterfall plot

```
[16] shap.plots.waterfall(shap_values[0][:, 1], max_display=25)
```

شکل ۲۲-۷ رسم نمودار آبشاری



شکل ۷-۲۳ نمودار آبشاری

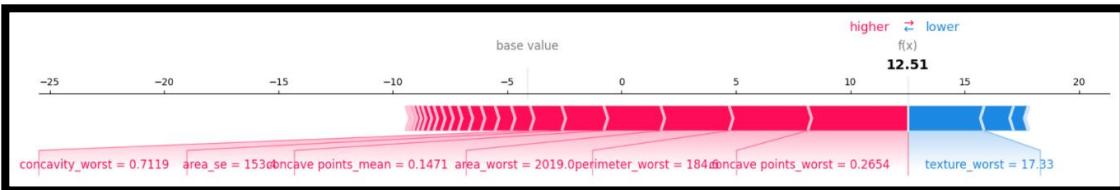
۷-۸-۲ رسم نمودار نیرو

مشابه قسمت قبل، با استفاده از کد زیر نمودار را رسم می‌کنیم (شکل ۷-۲۴) (شکل ۷-۲۵).

▪ Force Plot

```
[17] shap.plots.force(shap_values[0][:, 1], matplotlib=True)
```

شکل ۲۴-۷ رسم نمودار نیرو



شکل ۲۵-۷ نمودار نیرو

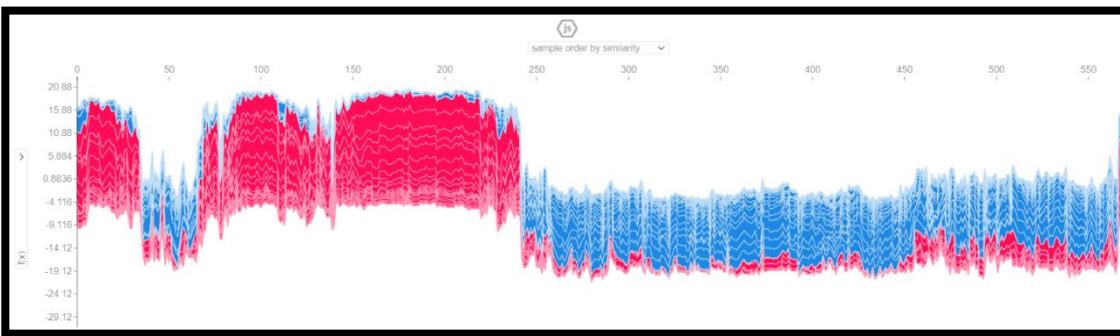
۳-۸-۳ رسم نمودار نیرو به صورت پشته

مشابه قسمت قبل، با استفاده از کد زیر نمودار را رسم می‌کنیم (شکل ۷-۲۶) (شکل ۷-۲۷).

▪ Stacked force plot

```
[18] shap.initjs()  
shap.plots.force(shap_values[:, :, 1])
```

شکل ۲۶-۷ رسم نمودار نیرو به صورت پشته



شکل ۲۷-۷ نمودار نیرو به صورت پشته

مشابه قسمت قبل، این نمودار برای پارامتری که طبق نمودار پیرسون، بیشترین اهمیت را دارد نیز ترسیم شده است (شکل ۷-۲۸).



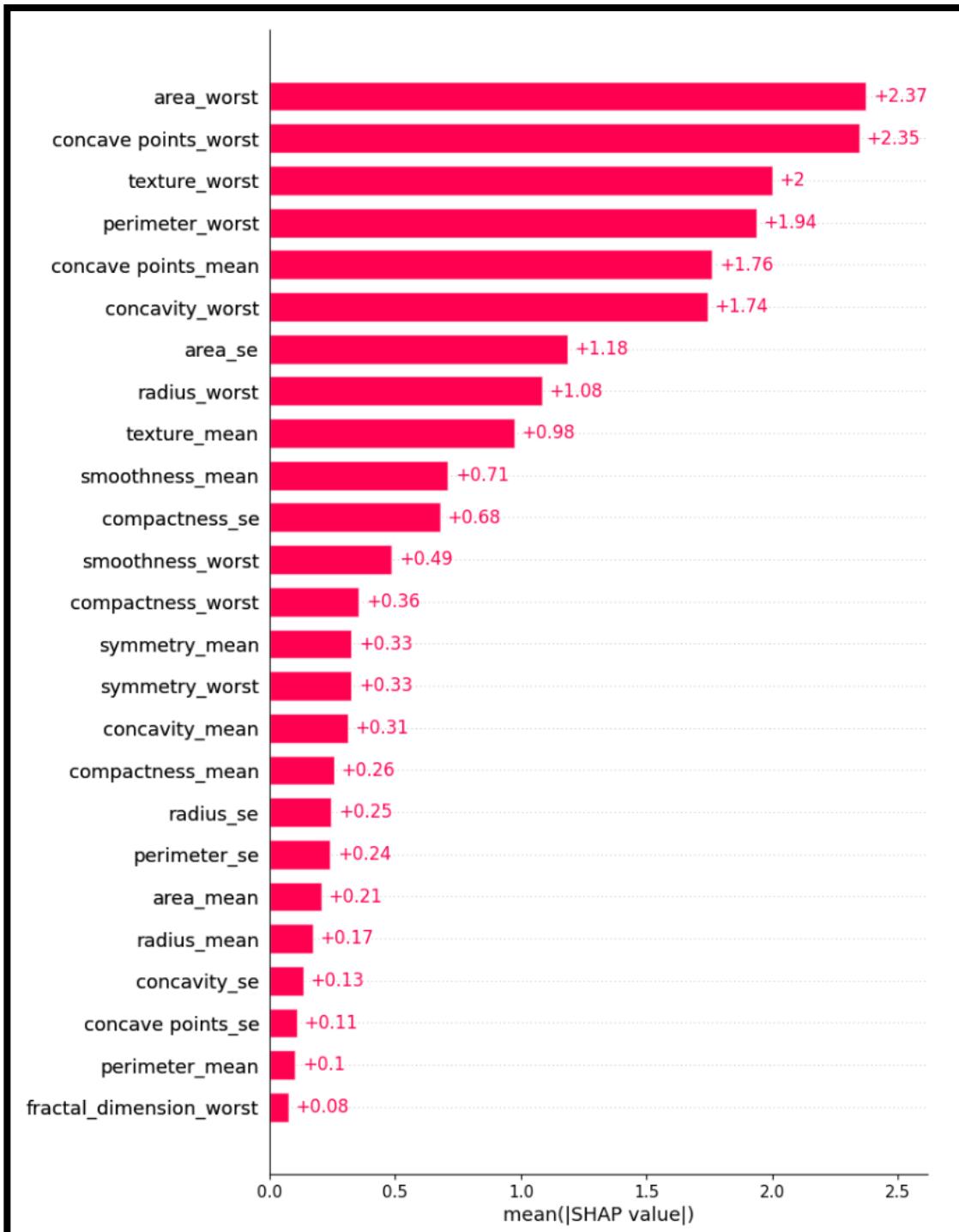
شکل ۲۸-۷ نمودار نیرو به صورت پشتی برای ویژگی بینرین حالت نقاط مقعر

۴-۸-۷ نمودار میله‌ای

مشابه قسمت قبل، با استفاده از کد زیر نمودار را رسم می‌کنیم (شکل ۲۹-۷) (شکل ۳۰-۷).



شکل ۲۹-۷ رسم نمودار میله‌ای



شکل ۷-۳۰ - نمودار میله‌ای

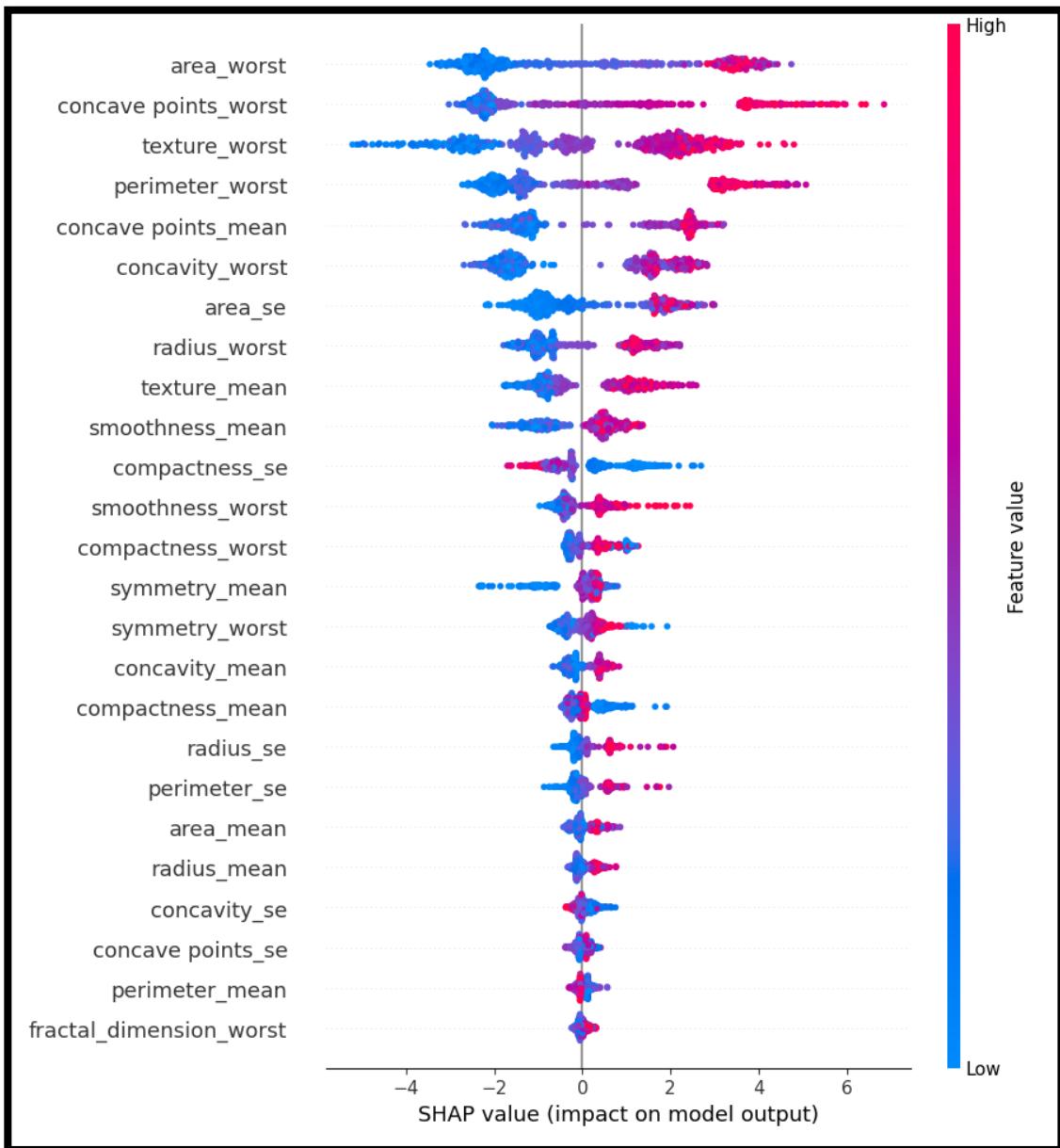
۷-۸ - نمودار ازدحام زنیورها

مشابه قسمت قبل، با استفاده از کد زیر نمودار را رسم می‌کنیم (شکل ۷-۳۱) (شکل ۷-۳۲).

▼ Beeswarm plot

```
[20]: shap.plots.beeswarm(shap_values[:, :, 1], max_display=25)  
No data for colormapping provided via 'c'. Parameters 'vmin', 'vmax' will be ignored
```

شکل ۳۱-۷ رسم نمودار از دحام زنیورها



شکل ۳۲-۷ نمودار از دحام زنیورها

برخلاف نمودار شب برای مدل تقویت گرادیان مفرد و نمودار پیرسون که در فصل های گذشته بررسی شد، در این دو نمودار آخر مشاهده می کنیم که ویژگی بدترین حالت مساحت^۱ با اختلاف اندکی از ویژگی بدترین حالت نقاط مقعر، در مدل فعلی ما به عنوان مهم ترین ویژگی شناخته شده است.

¹ area_worst

فصل هشتم

۸ تنظیم مدل ماشین افزایش گرادیان سبک (LGBM) با کمک فریم‌ورک آپتوна

در این فصل با بهره‌گیری از فریم‌ورک آپتونا و با استفاده از مدل ماشین تقویت گرادیان سبک، با جستجوی مجموعه مقادیر مختلف به صورت اتوماتیک، مقادیر بهینه برای پارامترهای مدل را در مدت زمان کوتاهی به دست آورده شده است. سپس مدل با مقادیر بهینه به دست آمده مجددآموزش داده شده است. در نهایت با رسم نمودارهای راک و دقت-حساسیت و همچنین نمودارهای شپ، اطلاعات بیشتری در مورد مدل آموزش داده شده ارائه شده است.

۱-۸ اضافه کردن کتابخانه‌ها و مجموعه داده مربوطه مشابه قسمت‌های قبل، ابتدا کتابخانه‌ها و مجموعه داده را اضافه می‌کنیم (شکل ۱-۸).

```

- Importing libraries
[ ] !pip install shap
[ ] !pip install optuna

- Importing library
[ ] from sklearn import metrics
import pandas as pd
import matplotlib.pyplot as plt
import shap

Using `tqdm.autonotebook.tqdm` in notebook mode. Use `tqdm.tqdm` instead to force console mode (e.g. in jupyter console)

- Importing the dataset
[ ] my_ds = pd.read_csv('breast-cancer.csv')
x = my_ds.iloc[:, 2:]
y = my_ds.iloc[:, 1].values

```

شکل ۱-۸ اضافه کردن کتابخانه‌ها و مجموعه داده

۲-۸ تعریفتابع هدف

همانطور که در قسمت توضیحات در فصل دوم گفته شد، فریمورک آپتوна برای انجام بهینه‌سازی احتیاج به تابع هدف دارد. در تصویر پایین تابع هدف تعریف شده قابل مشاهده است (شکل ۲-۸).

```

import optuna
import lightgbm as lgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score

def objective(trial):
    # Sample hyperparameters from the defined ranges
    num_iterations = trial.suggest_int('num_iterations', 50, 151, step=1)
    learning_rate = trial.suggest_float('learning_rate', 0.01, 0.2, step=0.01)
    subsample = trial.suggest_float('subsample', 0.05, 1, step=0.05)
    colsample_bytree = trial.suggest_float('colsample_bytree', 0.5, 1, step=0.05)
    min_data_in_leaf = trial.suggest_int('min_data_in_leaf', 10, 50, step=1)
    max_depth = trial.suggest_int('max_depth', 1, 100, step=1)
    num_leaves = trial.suggest_int('num_leaves', 2, 256, step=1)

    # Prepare the data (Replace X_train, X_test, y_train, y_test with your data)
    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)

    # Define the model with the sampled hyperparameters
    model = lgb.LGBMClassifier(
        objective='binary',
        metric='auc',
        boosting='gbdt',
        num_iterations=num_iterations,
        learning_rate=learning_rate,
        subsample=subsample,
        bagging_freq=1,
        colsample_bytree=colsample_bytree,
        min_data_in_leaf=min_data_in_leaf,
        max_depth=max_depth,
        num_leaves=num_leaves,
        random_state=0
    )

    # Fit the model to the training data
    model.fit(X_train, y_train)

    # Predict probabilities for the test set
    y_pred_proba = model.predict_proba(X_test)[:, 1]

    # Calculate ROC-AUC score as the objective to optimize
    roc_auc = roc_auc_score(y_test, y_pred_proba)

    # Return the objective value (ROC-AUC score) to Optuna for optimization
    return roc_auc

```

شکل ۲-۸ تعریف تابع هدف

روش کار تابع هدف به این صورت است که ابتدا یک شیء از کلاس آزمایش را به عنوان ورودی به آن می‌دهیم و سپس پارامترها را که قصد تنظیم آنها را داریم با استفاده از این شیء پیشنهاد می‌کنیم. سپس داده‌ها را به مجموعه آزمایش و آموزش شکسته، مدل را با پارامترهای پیشنهاد شده تعریف می‌کنیم و با داده‌های آموزش، آموزش می‌دهیم.

تابع به دنبال افزایش یا کاهش مقدار معیاری است که در هنگام فراخوانی تابع بهینه‌سازی به آن ورودی می‌دهیم. در نتیجه لازم است در داخل تابع آن مقدار را محاسبه کرده و برگردانیم. در اینجا ما از معیار مساحت زیر نمودار راک استفاده کرده‌ایم که با ورودی گرفتن مجموعه داده، آزمایش وابسته و مجموعه داده پیش‌بینی شده توسط مدل تعریف شده در تابع، نمودار این معیار را محاسبه کرده و بر می‌گرداند.

به کمک تکه کد زیر، یک شیء آموزش می‌سازیم تا کار بهینه‌سازی را انجام دهد. در این خط باید جهت افزایش یا کاهش مدنظرمان را برای شیء آموزش تعیین کنیم تا بر اساس آن، مقادیر در جهت مدنظر بهینه‌سازی شوند. در ادامه، متدهای این فراخوانی می‌کنیم تا کار بهینه‌سازی شروع شود. این متدهای هدف و تعداد آزمایش را به عنوان ورودی دریافت می‌کند. تابع هدف شامل فضای جستجو، ساخت مدل و آموزش آن و انجام پیش‌بینی به وسیله آن و در نهایت برگرداندن مقدار معیار مقایسه‌ای ماست. تعداد آزمایش همان تعداد دور است.

هرچه تعداد دور بیشتر باشد، شیء آموزش ترکیبات بیشتری از مجموعه چیده شده را انتخاب و بررسی کرده و سعی در بهینه‌سازی مقدار نهایی می‌کند. تعداد دور بیشتر نیازمند محاسبات بیشتر و زمان بیشتر است (شکل ۳-۸).

```
[ ] # Create an Optuna study
study = optuna.create_study(direction='maximize')

# Optimize the study (interrupt it if needed)
study.optimize(objective, n_trials=100)
```

شکل ۳-۸ تعریف شی آموزش و انجام بهینه‌سازی

نکته مثبت و قابل تأمل در رابطه با آپتوна این است که این فریمورک به شکل غیرقابل باوری سریع و دقیق عمل کرده و به ما در بهینه‌سازی پارامترهای مدل کمک شایانی می‌کند. نتایج به دست آمده در عکس پایین، در مدت زمانی کمتر از یک دقیقه محاسبه شده است (شکل ۴-۸).

▼ Optuna results

```
# Access the best hyperparameters found so far
best_params = study.best_params
best_score = study.best_value

# Access the trials dataframe to see the results of all trials
trials_dataframe = study.trials_dataframe()

print("Best Parameters:", best_params)
print("Best Score (ROC-AUC):", best_score)

# Print the dataframe with the results of all trials
print(trials_dataframe)

[+] Best Parameters: {'num_iterations': 151, 'learning_rate': 0.06000000000000005, 'subsample': 0.8, 'colsample_bytree': 0.9, 'min_data_in_leaf': 2
Best Score (ROC-AUC): 1.0
```

شکل ۸-۴ نمایش مقادیر بهینه بدست آمده توسط آپنرنا

همانطور که مشاهده می‌شود، پارامترهای به دست آمده مناسب با بیشترین مقدار برای مساحت زیر نمودار راک محاسبه شده‌اند. در گام بعدی مدل را مجدداً با بهترین پارامترهای به دست آمده آموزش می‌دهیم.
(شکل ۸-۵).

▼ Splitting the dataset into the training set and the test set

```
[ ] from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

▼ Re-training model with optimal parameters

```
[ ] lgbm_classifier = lgb.LGBMClassifier(**best_params)
lgbm_classifier.fit(x_train, y_train)

[LightGBM] [Warning] min_data_in_leaf is set=22, min_child_samples=20 will be ignored. Current value: min_data_in_leaf=22
Found `num_iterations` in params. Will use it instead of argument
* LGBMClassifier
LGBMClassifier(colsample_bytree=0.9, learning_rate=0.06000000000000005,
               max_depth=48, min_data_in_leaf=22, num_iterations=151,
               num_leaves=171, subsample=0.8)
```

شکل ۸-۵ آموزش مجدد مدل با مقادیر بهینه

۳-۸ ارزیابی مدل

۱-۳ ارزیابی با اعتبارسنجی متقابل
نتیجه ارزیابی با اعتبارسنجی متقابل به صورت زیر است (شکل ۸-۶).

▼ Applying k-fold cross validation

```
[ ] from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator=lgbm_classifier, X=x_train, y=y_train, cv=10)

[ ] print("Accuracy: {:.10f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.10f} %".format(accuracies.std()*100))

Accuracy: 97.3671497585 %
Standard Deviation: 2.3730823814 %
```

▼ Evaluate using confusion matrix & accuracy score

```
[ ] from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = lgbm_classifier.predict(x_test)
```

شکل ۶-۷ ارزیابی مدل با اعتبارسنجی متقابل

۳-۲-۸ ارزیابی با ماتریس درهم ریختگی و امتیاز صحت

نتیجه ارزیابی با ماتریس درهم ریختگی و امتیاز صحت به صورت زیر است (شکل ۷-۸).

▼ Evaluate using confusion matrix & accuracy score

```
[ ] from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = lgbm_classifier.predict(x_test)

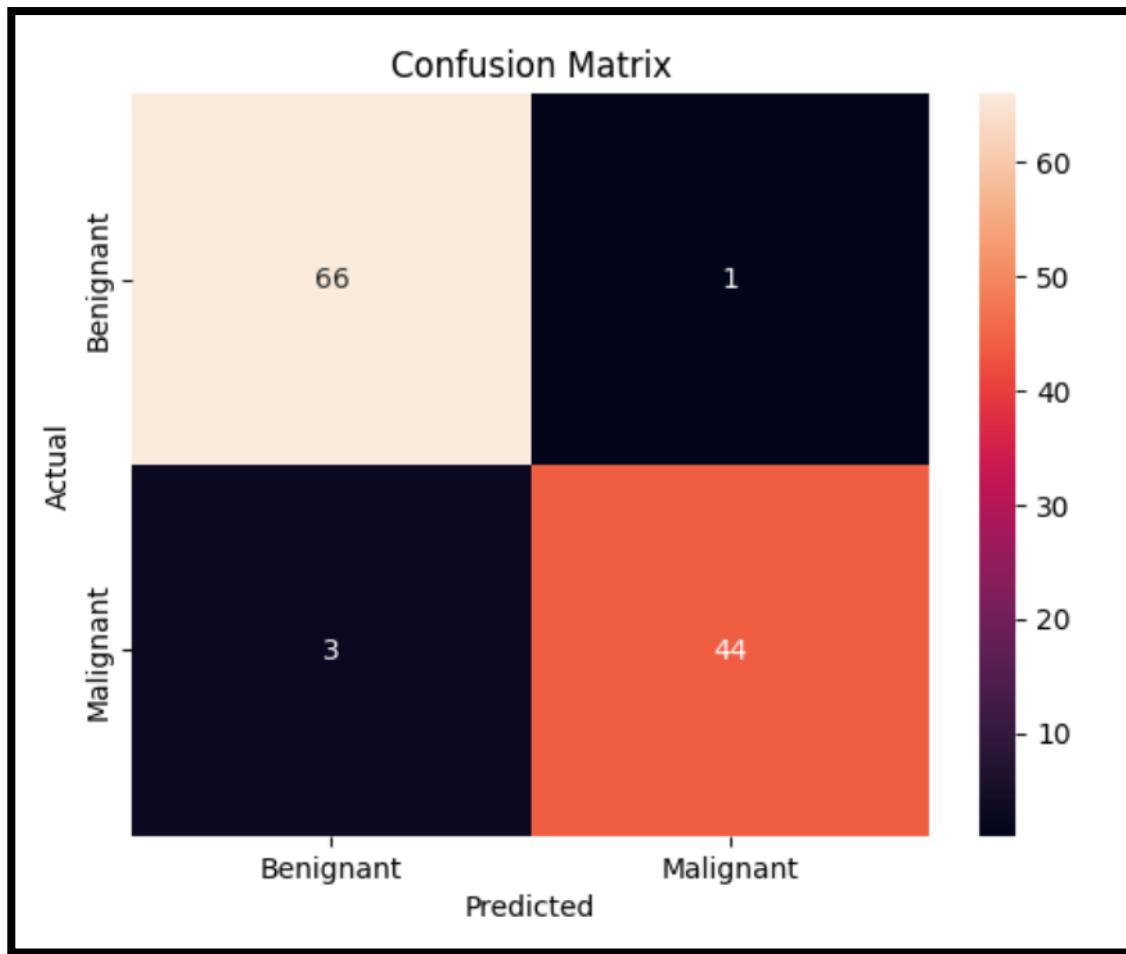
my_conf_matrix2 = confusion_matrix(y_test, y_pred)
print(my_conf_matrix2)
accuracy_score(y_test, y_pred)

[[66  1]
 [ 3 44]]
0.9649122807017544
```

شکل ۷-۸ ارزیابی مدل با ماتریس درهم ریختگی و امتیاز صحت

۳-۳-۸ نمایش ماتریس درهم ریختگی به همراه برچسب

ماتریس درهم ریختگی به همراه برچسب به صورت زیر است (شکل ۸-۸).



شکل ۱-۱۱ ماتریس درهمه‌یختگی با برچسب

همان طور که مشاهده می‌شود علی‌رغم اینکه مساحت زیر نمودار راک برابر یک بود، پارامترهای محاسبه شده نتیجه ضعیفتری را نسبت به آخرین مدل در فصل قبل در پیش‌بینی نتیجه برای داده‌های آزمایش و اعتبارسنجی متقابل چند دسته‌ای نشان می‌دهند.

۳-۴ - گزارش دسته‌بندی

گزارش دسته‌بندی در مدل جدید به شرح زیر است (شکل ۹-۸).

▼ Evaluating using classification report

```
print(metrics.classification_report(y_test, y_pred))

precision    recall  f1-score   support

          0       0.96      0.99      0.97       67
          1       0.98      0.94      0.96       47

   accuracy                           0.96      114
  macro avg       0.97      0.96      0.96      114
weighted avg       0.97      0.96      0.96      114
```

شکل ۹-۸ گزارش دسته‌بندی

طبق گزارش آورده شده در جدول بالا، ۹۶ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم به درستی دسته‌بندی شده است. این آمار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است.

معیار بعدی میزان یادآوری است. همانطور که از جدول مشاهده می‌شود، مدل ما توانسته ۹۹ درصد داده‌های مربوط به کلاس بیماران سرطانی خوش‌خیم را از بین مجموعه داده‌های این دسته پیدا کند. این مقدار برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۴ درصد است.

همانطور که از جدول مشاهده می‌شود امتیاز **f1** برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۷ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۶ درصد است. این آمار نشان می‌دهد به طور میانگین مدل ما در ارزیابی بیماران سرطانی خوش‌خیم کمی موفق‌تر است. صحت دسته‌بندی‌ها نیز مطابق با جدول برابر با ۹۶ است.

همانطور که مشخص است جدول گزارش دسته‌بندی نشان می‌دهد به طور کلی دقت مدل جدید نسبت به مدل قبل ۲٪ کاهش داشته است.

۳-۸-۵ محاسبه معیارهای اختصاصیت و حساسیت

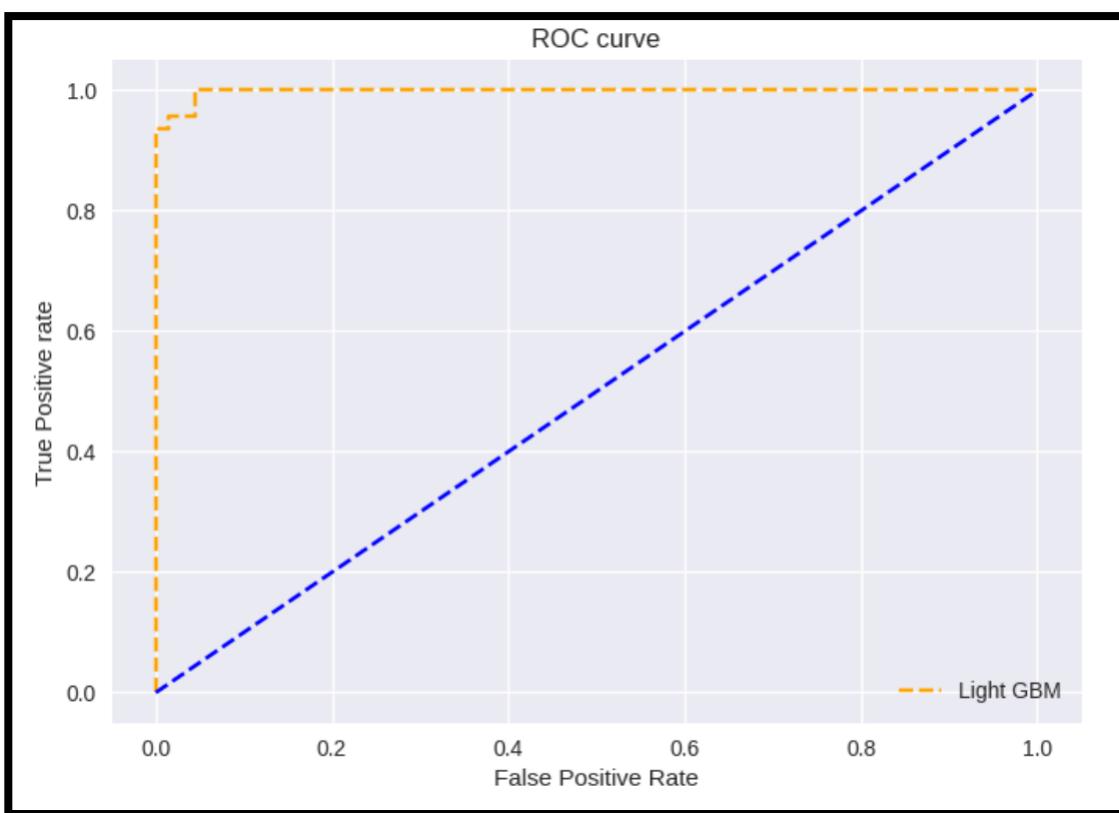
همانطور که از مقادیر داخل شکل زیر مشاهده می‌شود؛ میزان اختصاصیت برای داده‌های مربوط به بیماران سرطانی خوش‌خیم برابر با ۹۳ درصد و برای داده‌های مربوط به کلاس بیماران سرطانی بدخیم برابر با ۹۸ درصد است. این مقدار نسبت منفی‌های درست برای کلاس‌های ذکر شده را بیان می‌دارد (شکل ۸-۱۰).

▼ Specificity & Sensitivity Score

```
[ ] from imblearn.metrics import sensitivity_specificity_support  
pd.DataFrame(sensitivity_specificity_support(y_test, y_pred, average=None), index=['Sensitivity', 'Specificity', 'Support']).T  
  
Sensitivity Specificity Support  
0 0.985075 0.936170 67.0  
1 0.936170 0.985075 47.0
```

شکل ۱۰-۸ محاسبه معیارهای دقت و حساسیت

۳-۶ رسم منحنی ویژگی عملکرد گیرنده (راک) منحنی ویژگی عملکرد گیرنده ترسیم شده به صورت زیر است (شکل ۱۱-۸).



شکل ۱۱-۸ نمودار راک

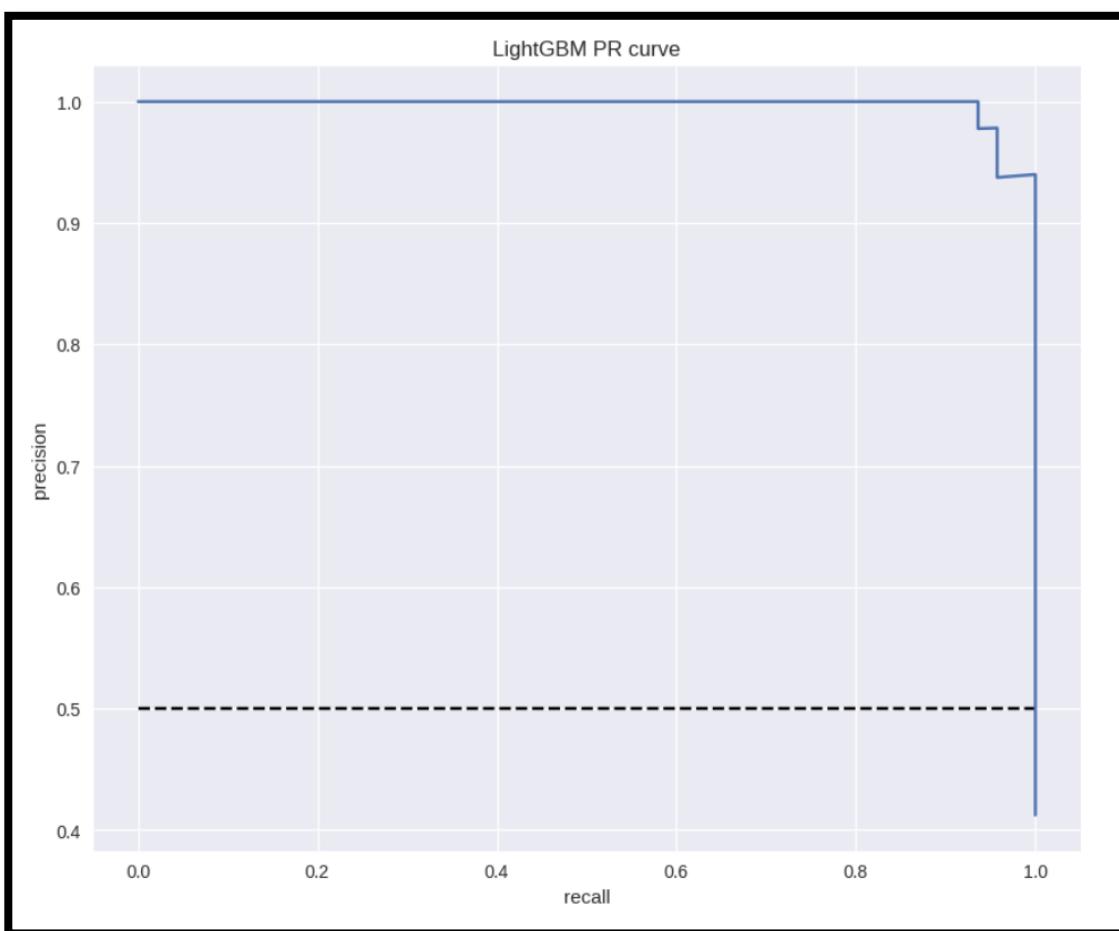
برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۱۲-۸).

```
[ ] print("ROC AUC Score: ", roc_auc_score(y_test, y_pred_proba[:,1]))  
ROC AUC Score:  0.997770720863766
```

شکل ۱۲-۱ مساحت زیر نمودار راک

همانطور که از شکل مشخص است، میزان مساحت زیر نمودار برابر با 99.77% است که نزدیکی آن به مقدار واحد بیانگر دقت بالای مدل است.

۳-۷ رسم نمودار منحنی دقت-حساسیت
منحنی دقت-حساسیت رسم شده به صورت زیر است (شکل ۱۳-۸).



شکل ۱۳-۸ نمودار دقت-حساسیت

برای ارزیابی عملکرد مدل، نیاز است مساحت زیر منحنی را تعیین کنیم. برای این کار از تیکه کد زیر استفاده می‌کنیم (شکل ۱۴-۸).

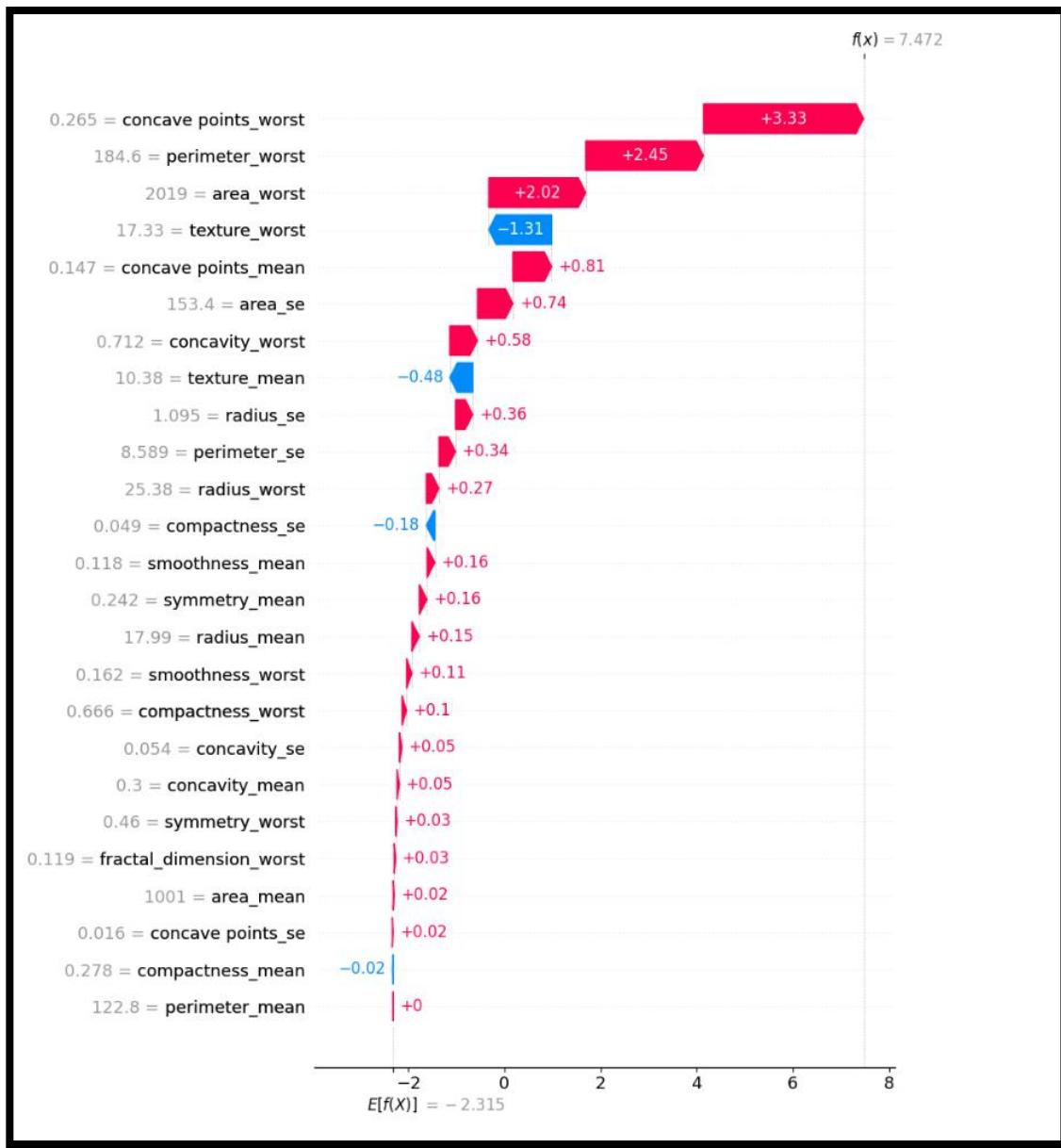
```
[ ] print("PRC AUC Score: ", auc(recall, precision))  
PRC AUC Score: 0.99692648582714
```

شکل ۱۴-۸ مساحت زیر نمودار دقت-حساسیت

همانطور که از شکل مشخص است، میزان مساحت زیر نمودار برابر با 0.9969 است که نزدیکی آن به مقدار واحد بیانگر دقت بالای مدل است.

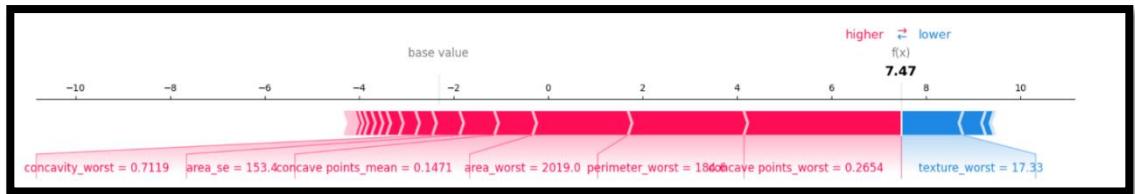
۴-۸ رسم نمودار مقادیر توضیحات افزونی شپلی

۴-۸-۱ رسم نمودار آبشاری
نمودار آبشاری به صورت زیر است (شکل ۱۵-۸).



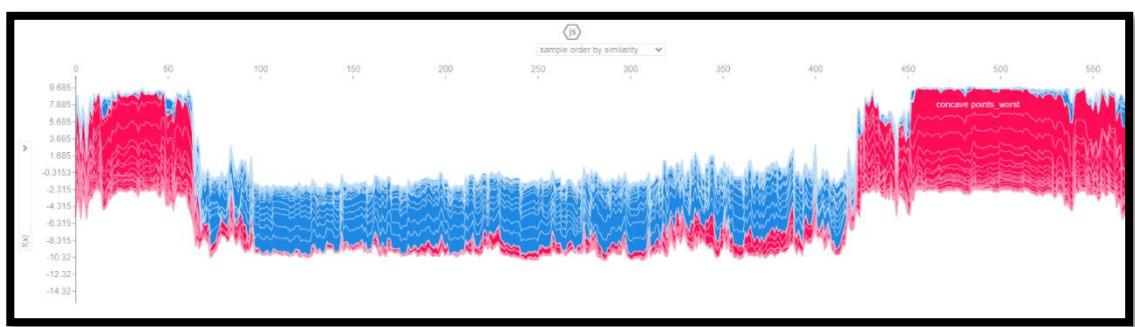
شکل ۱۵-۸ نمودار آبشاری

رسم نمودار نیرو
نمودار نیرو بصورت زیر است (شکل ۱۶-۸).



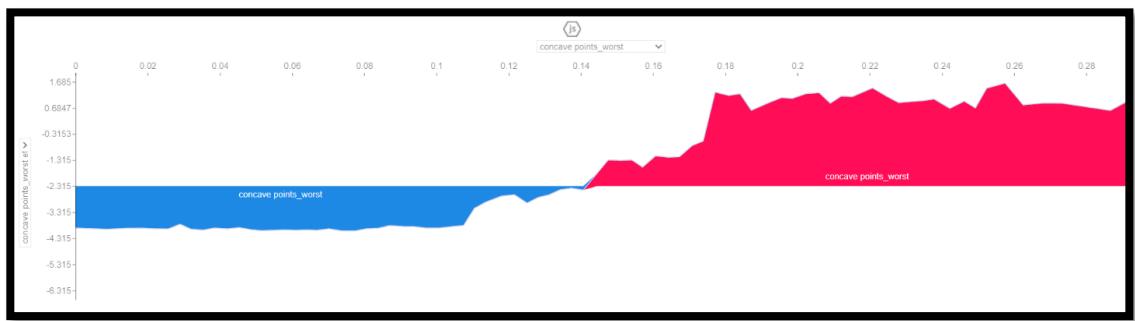
شکل ۱۶-۸ نمودار نیرو

۴-۳ رسم نمودار نیرو به صورت پشته
نمودار نیرو بصورت پشته در زیر آمده است (شکل ۱۷-۸).



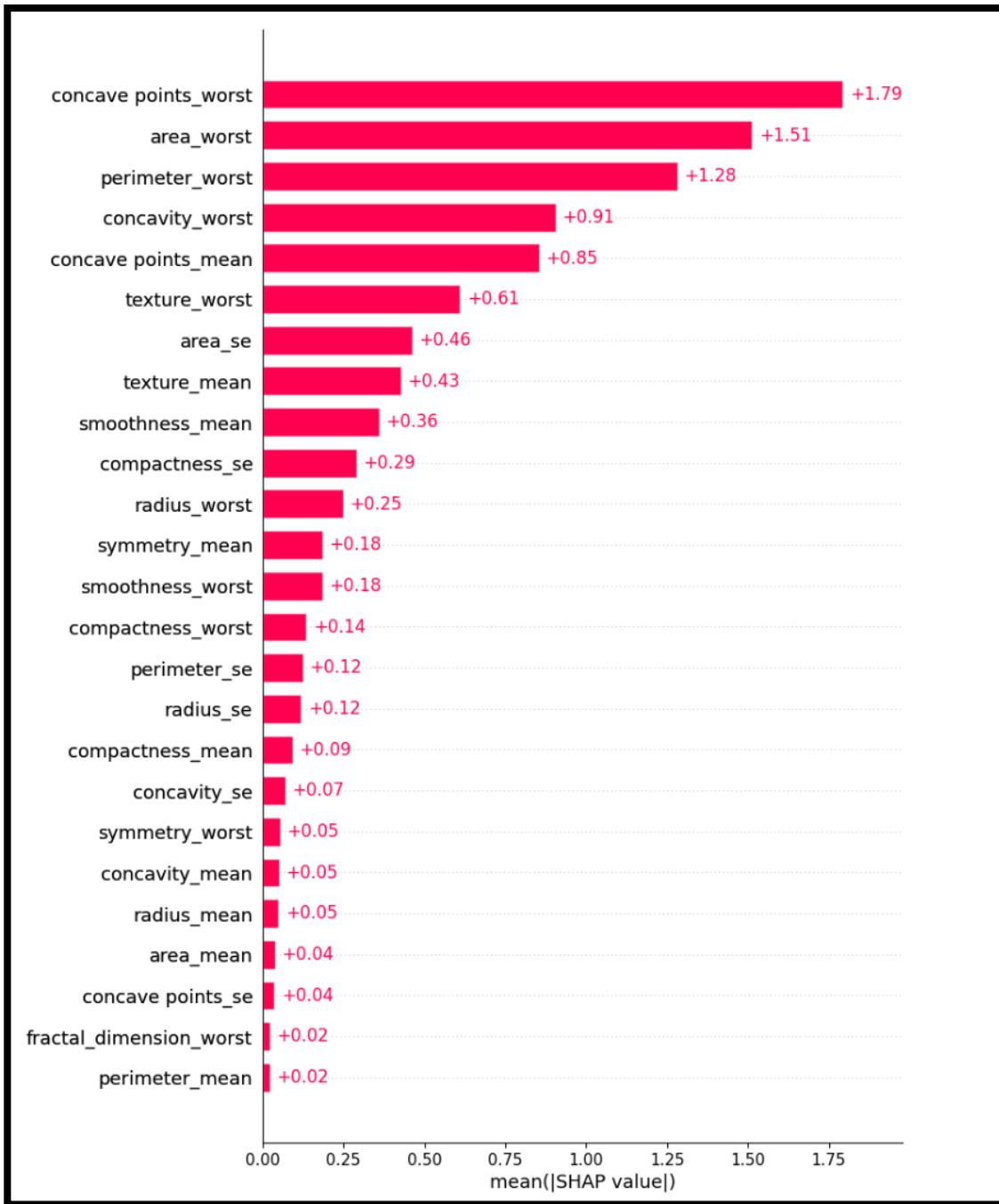
شکل ۱۷-۸ نمودار نیرو به صورت پشته

مشابه قسمت قبل، این نمودار برای پارامتری که طبق نمودار پیرسون، بیشترین اهمیت را دارد نیز ترسیم شده است (شکل ۱۸-۸).



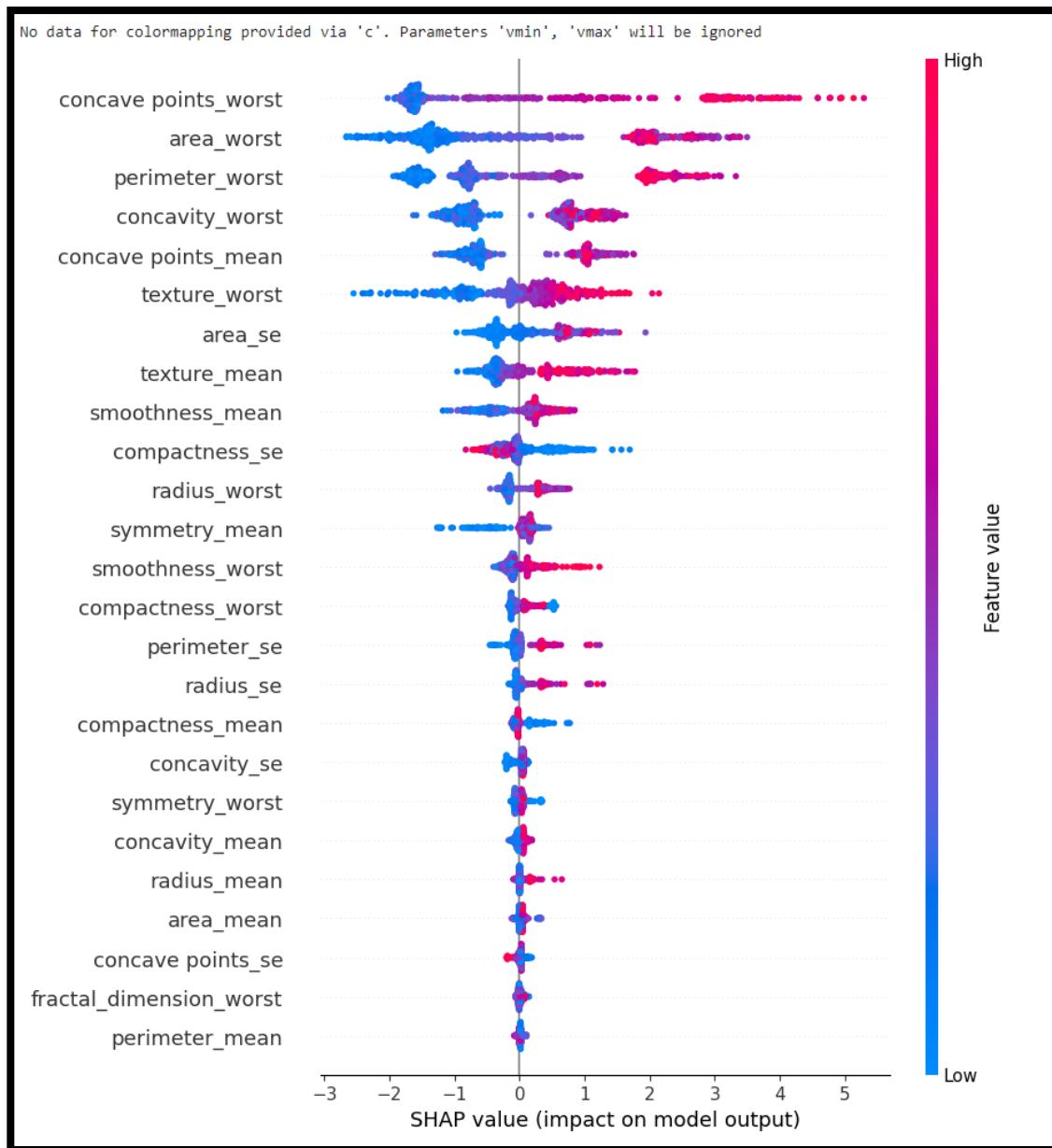
شکل ۱۸-۸ نمودار نیرو به صورت پشته برای ویژگی بیشترین حالت نقاط مقعر

۴-۴ نمودار میله‌ای
نمودار میله‌ای بصورت زیر است (شکل ۱۹-۸).



شکل ۱۹-۱ نمودار میله‌ای

۴-۵ - نمودار ازدحام زنپورها
نمودار ازدحام زنپورها بصورت زیر است (شکل ۲۰-۸).



شکل ۲۰-۸ نمودار از دحام زنیورها

همانطور که از دو نمودار آخر این قسمت مشخص است، ویژگی بدترین حالت نقاط مقعر مهمترین ویژگی در پیش‌بینی نتیجه نهایی مدل است. این نتیجه با نتیجه حاصل از نمودار پیرسون همخوانی دارد.

فصل نهم

۹ جمع‌بندی و پیشنهادها

در این فصل، جمع‌بندی و نتایج نهایی به همراه پیشنهادها برای ادامه پژوهه توسط پژوهشگران دیگر آورده شده است.

۹-۱ جمع‌بندی

در این پایان‌نامه تلاش پژوهشگر بر این بود که با استفاده از مدل‌های یادگیری ماشینی، روشی برای پیش‌بینی وضعیت سلطان سینه در افراد مبتلا به سلطان پیدا کند. با استفاده از مدل‌های تقویت گرادیان مفرط و ماشین افزایش گرادیان سبک و تنظیم پارامترهای ورودی این مدل‌ها، تلاش شد تا به این مهم دست یابیم. در بخش‌های مختلف این پایان‌نامه در فصول مختلف با آموزش و آزمایش مدل‌های مختلف، به دقت‌های نسبتاً بالایی رسیده و می‌توان گفت به هدف اصلی این پژوهه که بهینه‌سازی مدل‌های مذکور برای پیش‌بینی دقیق‌تر داده‌های ورودی بود دست یافتیم.

همانطور که در مدل‌های چهارم و هفتم بیان شد، پروسه تنظیم پارامتر برای مدل کار بسیار زمانبری است و به منابع پردازشی قوی نیازمند است. با توجه به اینکه پژوهشگر از منابع پردازشی قوی برخوردار نبود؛ تلاش گردید تا با هرس مجموعه فضای جستجو و استفاده از انتخاب پارامترهای تاثیرگذارتر در بازه‌های کوچکتر برای پیدا کردن مقدار بهینه، این زمان را تا حد امکان کاهش دهد.

۹-۲ نتیجه‌گیری

بر اساس نتایج به دست آمده در قسمت‌های مختلف و با مقایسه مدل‌های مختلف، این نتیجه‌گیری قابل برداشت است که مدل تقویت گرادیان مفرط به طور کلی از مدل ماشین افزایش گرادیان سبک دارای دقت بیشتری است و حتی بدون تنظیم پارامتر هم دقت مطلوبی دارد.

برخلاف آن، مدل ماشین افزایش گرادیان سبک، در حالت پیش‌فرض دقت کمتری داشته و نیاز به تنظیم پارامتر برای آن احساس می‌شود. با تنظیم پارامتر، دقت مدل ماشین افزایش گرادیان سبک به شکل چشمگیری افزایش می‌یابد.

نکته دیگری که می‌توان از این پایان‌نامه نتیجه گرفت این است که برای ارزیابی عملکرد مدل‌ها بهتر است از روش اعتبارسنجی متقابل چند دسته‌ای استفاده شود. استفاده صرف از مجموعه داده آزمایش برای بررسی عملکرد مدل‌ها کافی نیست و اتکا به نتایج آن ممکن است غلط انداز باشد. با استفاده از روش اعتبارسنجی متقابل چند دسته‌ای می‌توان تصور دقیق‌تری نسبت به عملکرد مدل داشت.

۹-۳ پیشنهادها

با توجه به اینکه در فصل آخر به طور مختصری به استفاده از فریم‌ورک آپتنا پرداخته شد و بنا بر ضيق وقت، تلاش زیادی روی بهبود عملکرد آن صورت نگرفت؛ پیشنهاد می‌شود پژوهشگر بعدی با تنظیم پارامترهای مختلف در بازه‌های مختلف، تلاش بر بهتر کردن نتایج حاصل از مدل در فصل آخر این پایان‌نامه داشته باشد.

همچنین پیشنهاد می‌شود از این فریم‌ورک برای بهبود عملکرد مدل افزایش گرادیان مفرط نیز استفاده شده و تلاش شود نتایج این مدل نیز بهبود داده شود.

منابع و مراجع

- [1] E. Burns, "AI," [Online]. Available: <https://www.techtarget.com/searchenterpriseai/definition/AI-Artificial-Intelligence>.
- [2] "supervised learning," [Online]. Available: <https://www.ibm.com/topics/supervised-learning>.
- [3] "decision tree," [Online]. Available: <https://www.mastersindatascience.org/learning/machine-learning-algorithms/decision-tree/>.
- [4] A. Singh, "A Comprehensive Guide to Ensemble Learning (with Python codes)," [Online]. Available: <https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/>.
- [5] "Gradient Boosting," [Online]. Available: <https://www.geeksforgeeks.org/ml-gradient-boosting/>.
- [6] "Introduction to XGBoost Algorithm in Machine Learning," [Online]. Available: <https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/>.
- [7] V. Morde, "XGBoost Algorithm: Long May She Reign!," [Online]. Available: <https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d>.
- [8] "Cross-Validation," [Online]. Available: <https://docs.aws.amazon.com/machine-learning/latest/dg/cross-validation.html>.
- [9] A. A. Awan, "An Introduction to SHAP Values and Machine Learning Interpretability," [Online]. Available: <https://www.datacamp.com/tutorial/introduction-to-shap-values-machine-learning-interpretability>.
- [10] P. Huilgol, "Precision and Recall | Essential Metrics for Machine Learning (2023 Update)," [Online]. Available: <https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/>.
- [11] A. Bhandari, "Guide to AUC ROC Curve in Machine Learning : What Is Specificity?," [Online]. Available: <https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/>.
- [12] "F-score," [Online]. Available: <https://en.wikipedia.org/wiki/F-score>.

- [13] D. Steen, "Precision-Recall Curves," [Online]. Available: <https://medium.com/@douglaspsteen/precision-recall-curves-d32e5b290248>.
- [14] "Optuna: A hyperparameter optimization framework," [Online]. Available: <https://optuna.readthedocs.io/en/stable/>.
- [15] "Breast Cancer Dataset," [Online]. Available: <https://www.kaggle.com/datasets/yasserh/breast-cancer-dataset>.
- [16] S. Mulani, "Python iloc() function – All you need to know!," [Online]. Available: <https://www.askpython.com/python/built-in-methods/python-iloc-function>.
- [17] J. Magiya, "Pearson Coefficient of Correlation Explained," [Online]. Available: <https://towardsdatascience.com/pearson-coefficient-of-correlation-explained-369991d93404>.
- [18] A. Jain, "Mastering XGBoost Parameter Tuning: A Complete Guide with Python Codes," [Online]. Available: <https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/>.
- [19] D. Martins, "XGBoost: A Complete Guide to Fine-Tune and Optimize your Model," [Online]. Available: <https://towardsdatascience.com/xgboost-fine-tune-and-optimize-your-model-23d996fab663>.
- [20] P. Banerjee, "A Guide on XGBoost hyperparameters tuning," [Online]. Available: <https://www.kaggle.com/code/prashant111/a-guide-on-xgboost-hyperparameters-tuning>.
- [21] E. Vidrascu, "random_state in Machine Learning," [Online]. Available: <https://www.kaggle.com/discussions/questions-and-answers/49890>.
- [22] "How to estimate GridSearchCV computing time?," [Online]. Available: <https://datascience.stackexchange.com/questions/29495/how-to-estimate-gridsearchcv-computing-time#:~:text=By%20default%20GridSearch%20runs%20parallel,180%2F4%3D45%20times..>
- [23] "How to interpret classification report of scikit-learn," [Online]. Available: <https://datascience.stackexchange.com/questions/64441/how-to-interpret-classification-report-of-scikit-learn>.
- [24] "SHAP with Python (Code and Explanations)," [Online]. Available: https://youtu.be/L8_sVRhBDLU.
- [25] "shap Explainer," [Online]. Available: <https://shap.readthedocs.io/en/latest/generated/shap.Explainer.html>.
- [26] M. Bahmani, "Understanding LightGBM Parameters (and How to Tune Them)," [Online]. Available: <https://neptune.ai/blog/lightgbm-parameters-guide>.

- [27] "LightGBM Hyperparameter Tuning with GridSearch," [Online]. Available: <https://www.datasnips.com/288/lightgbm-hyperparameter-tuning-with-gridsearch/>.
- [28] "Tune a LightGBM model," [Online]. Available: <https://docs.aws.amazon.com/sagemaker/latest/dg/lightgbm-tuning.html>.
- [29] [Online]. Available: <https://lightgbm.readthedocs.io/en/latest/Parameters.html>.
- [30] A. Saini, "Master the AdaBoost Algorithm: Guide to Implementing & Understanding AdaBoost," [Online]. Available: <https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/>.

Abstract

In this thesis, we classified breast cancer patients into two benign and malignant categories, with high accuracy by using light gradient boosting machine (LGBM) and extreme gradient boosting (XGBoost) models. We focused on adjusting hyper parameters of these models in order to improve performance.

The researcher has created and pruned the search space by using different sources and getting to know the important parameters of these two models to overcome the challenge of long learning time.

With the efforts made, the accuracy of the models used in this thesis has been significantly improved from the default state, and by using the k-fold cross-validation method, this performance improvement has been demonstrated.

In addition to the cross-validation method, accuracy score, precision, sensitivity, specificity and accuracy values have also been calculated for each improved model using the confusion matrix. Also, Receiver Operating Characteristic (ROC) and Precision-Recall curves are drawn for the improved final models.

In order to find the effect of each feature on the final prediction, SHapley Additive exPlanations (SHAP) value charts have been used, which present information about the performance status of the corresponding model.

In the last chapter, using the Optuna framework, an attempt has been made to improve the performance of the light gradient boosting machine model in an automatic and fast way, and the information related to the evaluation of the trained model using the parameters obtained by this framework has been given.

Keywords: extreme gradient boosting model; light gradient boosting machine model; cross validation; Receiver Operating Characteristic (ROC); Precision-Recall curve; shapley additive explanations; Optuna



Semnan University
Faculty of Electrical & Computer Engineering

B.Sc. Thesis in Electrical Engineering

**Implementation of a Machine Learning
Model for Classification of Breast Cancer
Cases into Benign and Malignant Classes**

By:

Amir Hossein Bagherian Kalat

Supervisor:

Dr. Morteza Dorrigiv

August 2023