دانشگاه شهید بهشتی
دانشکده مهندسی و علوم کامپیوتر

پیش‌بینی خطا با استفاده از یادگیری ماشین

پروژه کارشناسی مهندسی کامپیوتر

کوثر دست‌باز

دکتر حسن حقیقی

تابستان ۱۴۰۲

چکیده

پیشبینی خطا در واقع تخمینی است که بر اساس داده‌ها و اطلاعات موجود، تلاش می‌کند رویدادها یا وقوع خطاها در آینده را پیش‌بینی کند. در مواردی که داده‌های قبلی یا الگوهای مشابهی وجود داشته باشند، پیش‌بینی خطا می‌تواند به عنوان یک ابزار مفید در تحلیل ریسک و مدیریت خطاها استفاده شود. پیش‌بینی خطا در نرم‌افزار یک رویکرد مهم برای افزایش کیفیت و کاهش هزینه‌ها محسوب می‌شود. با استفاده از روش‌های مختلف تحلیل داده و الگوریتم‌های یادگیری ماشین، می‌توان خطاها را پیش از وقوع تشخیص داده و به موقع واکنش نشان داد. بررسی مقالات مختلف نشان می‌دهد که روش‌هایی همچون الگوریتم‌های خوشه‌بندی، شبکه‌های عصبی، مدل‌های طبقه‌بندی و تحلیل داده برای این منظور مفید واقع شده‌اند. الگوریتم‌های مشابهی همچون XGBoost، Supervised، Regression، Decision tree، Random Forest و K-means برای پیش‌بینی خطا مورد استفاده قرار می‌گیرند. الگوریتم‌های خوشه‌بندی به‌خصوص K-Means به‌عنوان یک روش یادگیری غیرنظارتی می‌تواند برای پیش‌بینی خطای ماژول‌های نرم‌افزاری مفید باشد. در این روش با اندازه‌گیری ویژگی‌های ماژول‌ها و تقسیم‌بندی آن‌ها به خوشه‌های مختلف، می‌توان الگوهای خطاپذیری را شناسایی کرد. با این حال الگوریتم K-Means به دلیل انتخاب تصادفی مراکز خوشه‌ها، گاهی عملکرد خوبی ندارد. پروژه حاضر روشی برای بهبود عملکرد K-Means پیاده‌سازی می‌کند. در این روش الگوریتم مرکز نقطه برای انتخاب اولیه مراکز خوشه‌ها بکار گرفته می‌شود تا این مشکل حل شود. سپس از این الگوریتم بهبود یافته برای پیش‌بینی خطاهای ماژول‌های نرم‌افزاری استفاده می‌شود. نتایج آزمایش روی مجموعه‌های داده نشان می‌دهد که روش پیاده‌سازی شده خطاها را بهتر پیش‌بینی می‌کند. در مجموع این پروژه راهکاری مناسب برای بهبود عملکرد الگوریتم خوشه‌بندی و پیش‌بینی دقیق‌تر خطاها در نرم‌افزار ارائه می‌دهد که می‌تواند به افزایش کیفیت و کاهش هزینه‌ها کمک کند.

فهرست مطالب

فصل اول: کلیّات

۱-۱ مقدمه

پیش‌بینی خطا یکی از موضوعات مهم در علوم کامپیوتر و مهندسی نرم‌افزار است که به کمک تحلیل داده‌ها و الگوریتم‌های مختلف، به ما امکان می‌دهد خطاها و نقص‌های موجود در سیستم‌ها و نرم‌افزارها را پیش‌بینی کرده و اقدامات مناسبی در جهت جلوگیری از آن‌ها انجام دهیم. این فرایند می‌تواند بهبود قابل توجهی در کیفیت و عملکرد نرم‌افزارها، کاهش هزینه‌ها و افزایش رضایت کاربران منجر شود. هر نرم‌افزاری با خطاها و نقص‌هایی روبرو است که ممکن است در زمان اجرا و در محیط‌های مختلف باعث به هم ریختگی و عملکرد ناپایدار شود. پیش‌بینی خطا به ما این امکان را می‌دهد تا با تحلیل داده‌های مربوطه و استفاده از الگوریتم‌های مناسب، خطرات مربوط به خطاها را قبل از وقوع آن‌ها شناسایی کنیم و اقدامات لازم را برای جلوگیری از آن‌ها انجام دهیم. روش‌های متنوعی برای پیش‌بینی خطا وجود دارد که شامل استفاده از الگوریتم‌های یادگیری ماشین، شبکه‌های عصبی، مدل‌های آماری و الگوریتم‌های داده کاوی می‌شود. این روش‌ها با تحلیل داده‌های موجود در محیط توسعه نرم‌افزار، نمونه‌های خطا، اطلاعات لاگ‌ها و سایر منابع، قادر به شناسایی الگوها و روندهای مرتبط با خطاها و نقص‌ها می‌باشند. سپس با استفاده از مدل‌های ساخته شده، می‌توان خطر خطاها را پیش‌بینی کرده و اقدامات مناسبی را برای رفع آن‌ها انجام داد. پیش‌بینی خطا در حوزه‌های مختلفی از جمله توسعه نرم‌افزار، تست و نگهداری سیستم‌ها، بهره‌برداری و مدیریت منابع می‌تواند مورد استفاده قرار گیرد. با استفاده از این تکنیک‌ها می‌توان زمان و هزینه‌های مربوط به تعمیر خطاها را کاهش داده، بهبود کیفیت نرم‌افزارها را بهبود بخشید و از نظرات و نیازهای کاربران به خوبی پاسخ‌دهی کرد. به طور خلاصه، پیش‌بینی خطا یک رویکرد مهم در حوزه علوم کامپیوتر و مهندسی نرم‌افزار است که با استفاده از تحلیل داده‌ها و الگوریتم‌های مختلف، به ما کمک می‌کند خطاها و نقص‌های موجود در سیستم‌ها و نرم‌افزارها را پیش‌بینی و مدیریت کنیم. با این رویکرد، می‌توانیم کیفیت و عملکرد نرم‌افزارها را بهبود بخشیم، هزینه‌ها را کاهش دهیم و رضایت کاربران را افزایش دهیم.

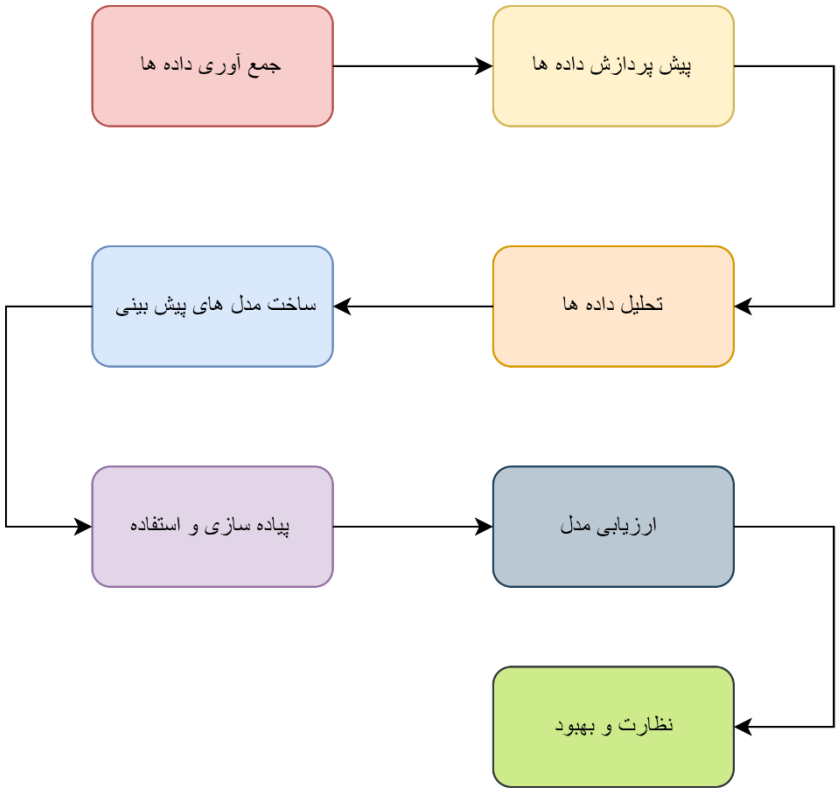
۱-۲ بیان مسئله

در حال حاضر، نرم‌افزارها و سیستم‌های کامپیوتری در حوزه‌های مختلفی از جمله صنعت، ارتباطات، بهداشت، مالی و غیره، با تعقیب روزافزون تکنولوژی و پیچیده شدن نیازها، دارای ساختارها و قابلیت‌های بسیار پیچیده‌ای هستند. با افزایش پیچیدگی و حجم برنامه‌ها، خطاها و نقص‌های نرم‌افزاری نیز از طریقه‌های مختلفی وارد سیستم‌ها می‌شوند. این خطاها ممکن است منجر به کاهش عملکرد، به هم ریختگی و حتی عدم قابلیت استفاده از سیستم شود. با توجه به اهمیت بالای نرم‌افزارها در زندگی ما، پیش‌بینی و شناسایی خطاها و نقص‌ها قبل از وقوع آن‌ها امری بسیار حیاتی است. در صورتی که بتوانیم خطر خطاها را پیش‌بینی کرده و اقدامات مناسبی را برای جلوگیری از آن‌ها انجام دهیم، می‌توانیم برنامه‌ها را پایدارتر و امن‌تر کنیم و از عواقب ناگواری همچون از دست رفتن داده‌ها، توقف سیستم و خسارت‌های مالی جلوگیری کنیم. به طور کلی، مسأله پیش‌بینی خطا در نرم‌افزارها به دنبال داشتن یک روش مؤثر و کارآمد برای تحلیل داده‌های موجود در محیط توسعه و عملکرد نرم‌افزار، شناسایی الگوها و روندهای مرتبط با خطاها، و پیش‌بینی خطاهای آینده است. این مسأله نیازمند بهره‌گیری از الگوریتم‌های یادگیری ماشین، شبکه‌های عصبی، مدل‌های آماری و الگوریتم‌های داده کاوی است تا بتوانیم خطاها را در مراحل مختلف توسعه و استفاده از نرم‌افزارها پیش‌بینی کرده و اقدامات مناسبی را برای جلوگیری و رفع آن‌ها انجام دهیم. در پیش‌بینی خطا، یادگیری ماشین نقش بسیار مهمی را ایفا می‌کند. یادگیری ماشین به ما امکان می‌دهد تا بر اساس داده‌های جمع‌آوری شده، الگوها و روابط پنهان در داده‌ها را شناسایی کنیم و مدل‌هایی بسازیم که بتوانند خطاها و اشتباهات آتی را پیش‌بینی کنند. الگوریتم پیاده‌سازی شده با رفع مشکل انتخاب تصادفی مرکزهای خوشه در k-

means، عملکرد این الگوریتم را بهبود می‌بخشد. سپس الگوریتم مرکز نقطه برای پیش‌بینی خطاهای ماژول‌های نرم‌افزاری استفاده شده است.

۳-۱ کلیات روش پیاده‌سازی شده

روش پیش‌بینی خطا یک فرایند سیستماتیک است که برای شناسایی و پیش‌بینی خطاها و نقص‌های محتمل در نرم‌افزارها استفاده می‌شود. این روش، بر پایه تحلیل داده‌های مرتبط با توسعه و استفاده از نرم‌افزار، ایجاد الگوها و مدل‌های پیش‌بینی استوار است. هدف اصلی این روش، کاهش خطر خطاها و بهبود کیفیت و پایداری نرم‌افزار است. روش پیش‌بینی خطا به طور کلی شامل مراحل جمع‌آوری داده‌ها، پیش پردازش داده‌ها، تحلیل داده‌ها، ساخت مدل‌های پیش‌بینی، ارزیابی مدل، پیاده‌سازی و استفاده، نظارت و بهبود می‌باشد. در این مراحل ابتدا داده‌های مرتبط با توسعه و استفاده از نرم‌افزار جمع‌آوری می‌شوند سپس داده‌ها پیش پردازش می‌شوند تا به صورتی قابل استفاده برای تحلیل و مدل‌سازی شوند و بعد داده‌ها تحلیل می‌شوند تا الگوها و روندهای مرتبط با خطاها و نقص‌ها شناسایی شوند، براساس ویژگی‌های استخراج شده و با استفاده از الگوریتم‌های یادگیری ماشین، شبکه‌های عصبی، مدل‌های آماری و سایر روش‌های موجود، مدل‌های پیش‌بینی خطا ساخته می‌شوند و مدل‌های ساخته شده ارزیابی می‌شوند.



شکل ۱

۴-۱ ساختار روش پیاده‌سازی شده

در فصل اول این پروژه، به بررسی و توضیح پیش‌بینی خطا و روش‌های یادگیری ماشین می‌پردازیم. پیش‌بینی خطا، فرآیندی است که در آن سعی می‌شود با استفاده از الگوریتم‌ها و مدل‌های یادگیری ماشین، خطاهای آتی در سیستم‌ها و نرم‌افزارها پیش‌بینی شوند. اینکه چگونه این پیش‌بینی خطا انجام می‌شود، مورد بحث و بررسی قرار می‌گیرد و الگوریتم‌ها و مدل‌های مختلفی برای این منظور استفاده می‌شوند. در این فصل، نیز به توضیحاتی درباره نحوه انجام پروژه و مسأله مورد بحث می‌پردازیم. مسأله ما در این پروژه، پیش‌بینی خطا در سیستم‌ها و نرم‌افزارها است. با استفاده از داده‌های موجود، سعی می‌کنیم الگوریتم‌های یادگیری ماشین را برای پیش‌بینی خطا آموزش دهیم و در نهایت، قادر باشیم خطاهای آتی را پیش‌بینی کنیم. اهمیت پیش‌بینی خطا نیز در این فصل مورد بحث و بررسی قرار می‌گیرد. با داشتن پیش‌بینی دقیق درباره خطاهای آتی، می‌توان اقدامات مناسبی را برای جلوگیری از وقوع آن‌ها انجام داد و عملکرد سیستم‌ها را بهبود بخشید. بنابراین، پیش‌بینی خطا از اهمیت بالایی برخوردار است. در فصل دوم، به بررسی مقالات مرتبط با این حوزه می‌پردازیم. در این مقالات، مزایا و معایب روش‌های مختلف پیش‌بینی خطا بررسی و آن‌ها را مشخص می‌کنیم. این بررسی نشان می‌دهد که هیچ روشی بی‌عیب و کامل نیست و هر یک دارای نقاط قوت و ضعف خاص خود هستند. اما با ترکیب و تلفیق این روش‌ها، می‌توان بهبود و دقت بیشتری در پیش‌بینی خطا داشت. در فصل سوم، به روش انتخابی برای انجام این پروژه و نحوه پیاده‌سازی آن می‌پردازیم و جزئیات آن را شرح می‌دهیم. بر اساس بررسی مقالات و تحقیقات ما، یک روش مناسب برای پیش‌بینی خطا انتخاب شده است که برای پیاده‌سازی در این پروژه مناسب به نظر می‌رسد. در این فصل، جزئیات روش انتخابی و نحوه اجرای آن توضیح داده می‌شود.

فصل دوم: مفاهیم پایه و کارهای مرتبط

۲-۱ مقدمه

برای انجام بهتر پروژه چندین مقاله مورد بررسی قرار گرفته است تا مفاهیم و روش‌های پیشبینی خطا بهتر درک شود. حضور باگ‌ها در نسخه‌های نرم‌افزاری به ندرت قابل اجتناب است و خساراتی که به شرکت‌ها به دلیل حضور باگ در یک نسخه نرم‌افزار وارد می‌شود، بسیار عظیم است. روش‌های مدرن تست و رفع عیب‌ها تمرکز خود را از "تشخیص" به "پیش‌بینی" باگ‌ها در کد منتقل کرده‌اند. مدل‌های موجود پیش‌بینی باگ تاکنون بهینه‌سازی شده برای استفاده تجاری نشده‌اند. علاوه بر این، مقیاس‌پذیری این مدل‌ها به طور کافی بررسی نشده است. دریکی از مقالات مطالعه شده، یک مدل پیش‌بینی باگ ارائه شده است که می‌تواند در یک پلتفرم ابری برای استفاده توسعه‌دهندگان نرم‌افزار قرار گیرد. هدف این مدل پیش‌بینی حضور یا عدم حضور باگ در کد با استفاده از مدل‌های طبقه‌بندی یادگیری ماشین است. با استفاده از پلتفرم یادگیری ماشین مایکروسافت آژور، این مدل به عنوان یک سرویس وب به صورت گسترده در سراسر جهان ارائه می‌شود، به این ترتیب BPaaS (پیش‌بینی باگ به عنوان سرویس) فراهم می‌کند. در مقاله‌ای دیگر به بررسی مدل‌های پیش‌بینی باگ نرم‌افزار با استفاده از الگوریتم‌های یادگیری ماشینی می‌پردازد. این مدل‌ها برای پیش‌بینی خطاهای نرم‌افزاری در مراحل مختلف توسعه نرم‌افزار استفاده می‌شوند و از معیارهای مختلفی مانند دقت و اندازه‌گیری AUC برای ارزیابی عملکرد مدل‌ها استفاده می‌شود. در مقاله‌ی بعدی، بهبود دقت پیش‌بینی عیب نرم‌افزار را با استفاده از یک مدل پیش‌بینی مبتنی بر تحلیل تأثیر خاکستری و الگوریتم Navie Bayes هدف دارد. این مدل از Navie Bayes به عنوان کلاسیفایر اصلی مدل پیش‌بینی عیب نرم‌افزار استفاده می‌کند. تحلیل تأثیر خاکستری برای تحلیل رابطه بین ماژول‌های نرم‌افزار و ماژول‌های‌ایده‌آل استفاده می‌شود. سپس درجه همبستگی خاکستری به عنوان یک ویژگی در مدل طبقه‌بندی Navie Bayes به‌عنوان ویژگی ویژگی تعبیه می‌شود. براساس مقایسه و تحلیل مجموعه داده عمومی NASA، مدل پیش‌بینی در این مقاله دقت پیش‌بینی را بهبود می‌بخشد. بطور کلی روش‌های مختلف دیگری نیز مورد بررسی قرار گرفتند و در ادامه به نقاط ضعف و قوت این روش‌ها می‌پردازیم.

۲-۲ تحلیل نقاط قوت و ضعف منابع غیر پژوهشی مشابه

برخی الگوریتم‌های محبوب پیش‌بینی باگ و نقاط قوت و ضعف آنها عبارتند از:

- Navie Bayes classifier:

الگوریتم Naive Bayes classifier یکی از روش‌های مهم در حوزه پیش‌بینی خطا است. این الگوریتم مبتنی بر قاعده بیز است و بر اساس احتمالات وقوع رویدادها برای پیش‌بینی خطا استفاده می‌شود. الگوریتم Naive Bayes به صورت ساده و سریع عمل کرده و در مواردی که فرضیات آن رعایت شود، نتایج قابل قبولی ارائه می‌دهد. برای استفاده از الگوریتم Naive Bayes classifier در پیش‌بینی خطا، ابتدا باید داده‌ها را به دو دسته خطا دار و خطا ندار تقسیم کنیم. سپس با استفاده از داده‌های آموزشی، مدل را آموزش می‌دهیم. برای آموزش این مدل، ابتدا باید احتمالات پیشین [۱]را محاسبه کنیم که نشان دهنده احتمال وقوع هر کلاس (خطا دار یا خطا ندار) است. سپس با استفاده از ویژگی‌های موجود در داده‌ها، احتمالات شرطی [۲]را محاسبه می‌کنیم که نشان دهنده احتمال وقوع هر ویژگی به شرط داشتن یک کلاس است. در اینجا فرض ساده «نسبت استقلال شرطی» می‌شود، به این معنی که ویژگی‌ها به طور مستقل از یکدیگر در نظر گرفته می‌شوند. با محاسبه احتمالات پیشین و احتمالات شرطی، می‌توان با استفاده از قاعده بیز، احتمال وقوع هر کلاس به شرط داشتن یک مجموعه ویژگی را محاسبه کرد. سپس با مقایسه احتمالات وقوع هر کلاس، می‌توانیم تصمیم نهایی را درباره کلاس مربوط به داده جدید بگیریم. به عبارت دیگر، الگوریتم Naive Bayes classifier با محاسبه

[۱] prior probabilities

[۲] conditional probabilities

احتمالات، داده جدید را به کلاسی اختصاص می‌دهد. از مزایای الگوریتم Naive Bayes classifier می‌توان به سرعت اجرا، سادگی پیاده‌سازی و کارایی در دسته‌بندی داده‌های بزرگ اشاره کرد. با این حال، این الگوریتم دارای فرضیات خاصی است که ممکن است در برخی موارد و با توجه به خصوصیات داده‌ها، نتایج دقیقی ندهد. همچنین، وقوع پدیده‌های نادر در داده‌ها می‌تواند باعث بهبود نتایج این الگوریتم شود.

نقاط قوت

- سادگی بسیار بالا: این الگوریتم از ساده‌ترین الگوریتم‌هاست و پیاده‌سازی آن آسان است.
- میزان داده کم : می‌تواند با داده‌های کم و ورودی‌های کم مقدار کار کند.
- سرعت بالا: به دلیل سادگی، سرعت بالایی دارد و می‌تواند بر روی داده‌های بزرگ هم اجرا شود.

نقاط ضعف

- فرض استقلال ویژگی‌ها: در عمل اغلب ویژگی‌ها به هم وابسته هستند که این الگوریتم آن را در نظر نمی‌گیرد.
- عملکرد ضعیف در موارد پیچیده: برای مسائل پیچیده‌ای که وابستگی‌ها مهم است عملکرد خوبی ندارد.

- Logistic Regression:

الگوریتم Logistic Regression یکی از روش‌های تحلیل پیش‌بینی است که برای مسائل دسته‌بندی استفاده می‌شود، از جمله پیش‌بینی خطا. این الگوریتم، بر اساس logistic function که یک تابع غیرخطی است، مدلی را برای تخمین احتمال وقوع یک کلاس در مقابل کلاس دیگر ساخته و استفاده می‌کند. برای استفاده از الگوریتم Logistic Regression در پیش‌بینی خطا، ابتدا باید داده‌ها را به دو دسته خطا دار و خطا ندار تقسیم کنیم. سپس با استفاده از داده‌های آموزشی، مدل Logistic Regression را آموزش می‌دهیم. هدف این مدل، یافتن یک تابع خطی که وابستگی میان ویژگی‌ها و احتمال وقوع خطا را توصیف کند.

نقاط قوت

- قادر به مدل‌سازی ارتباط بین ویژگی‌هاست.
- عملکرد خوبی برای مسائل خطی دارد و برای داده‌های بزرگ مناسب است.
- پارامترهایش قابل تفسیر هستند.
- پیاده‌سازی ساده‌ای دارد.

نقاط ضعف

- فقط قادر به مدل‌سازی روابط خطی است.
- ممکن است برای مسائل غیرخطی عملکرد ضعیفی داشته باشد.
- نیاز به پیش‌فرض‌های بیشتری نسبت به طبقه‌بندی Navie Bayes classifier دارد.

- Decision tree

الگوریتم درخت تصمیم [۳]یک روش تحلیل پیش‌بینی است که برای مسائل دسته‌بندی و رگرسیون استفاده می‌شود، از جمله پیش‌بینی خطا. این الگوریتم بر اساس ساختار گرافیکی یک درخت تصمیم، جهت پیش‌بینی خروجی برای داده‌های ورودی استفاده می‌کند. در الگوریتم درخت تصمیم، هدف اصلی ساختن یک درخت تصمیم است که بتواند به صورت مرحله به مرحله تصمیم‌هایی را برای داده‌های ورودی بگیرد و خروجی مورد نظر را پیش‌بینی کند. درخت تصمیم از تعدادی گره تشکیل شده است، که هر گره نماینده

[۳] Decision Tree

یک تصمیم یا یک شرط است. هر گره دارای شاخه‌هایی است که به گره‌های دیگری اشاره می‌کنند و با توجه به شرایط مختلف، مسیر مناسب را در درخت طی می‌کنند.

نقاط قوت

- قادر به مدل‌سازی روابط غیرخطی و پیچیده بین ویژگی‌هاست.
- نتایج به صورت ساختار شجری هستند که قابل تفسیر و مطالعه است.
- پیاده‌سازی آسانی دارد.

نقاط ضعف

- درصورت وجود اختلال در داده‌ها یا ویژگی‌های بی‌ربط عملکردش پایین می‌آید.
- عملکردش برای داده‌های بزرگ ضعیف است.

- Random forest

الگوریتم جنگل تصادفی [۴]یک روش ماشین برداری است که برای مسائل دسته‌بندی و رگرسیون استفاده می‌شود، از جمله پیش‌بینی خطا. این الگوریتم بر پایه ترکیب تعدادی درخت تصمیم که به صورت تصادفی ساخته می‌شوند، عمل می‌کند. در الگوریتم جنگل تصادفی، برای ساخت هر درخت تصمیم در جنگل، از یک زیرمجموعه تصادفی از داده‌ها و ویژگی‌ها استفاده می‌شود. به این ترتیب، هر درخت تصمیم با دیدگاهی متفاوت از داده‌ها ساخته می‌شود. فرآیند ساخت هر درخت تصمیم، شامل انتخاب تصادفی داده‌ها (با جایگزینی) و تصادفی از میان ویژگی‌ها است. با ساخت جنگل تصادفی که شامل تعدادی درخت تصمیم است، هر درخت به صورت جداگانه و به صورت مستقل از سایر درختان کلاس‌بندی را انجام می‌دهد. در هنگام پیش‌بینی، خروجی نهایی تعیین می‌شود توسط رأی‌گیری اکثریت بین درختان تصمیم. به عبارت دیگر، خروجی نهایی برابر با برچسبی است که بیشترین تعداد آن را در درختان تصمیم دارد.

نقاط قوت

- دقت بالایی دارد و یکی از دقیق‌ترین الگوریتم‌هاست.
- می‌تواند روابط غیرخطی پیچیده را مدل کند.
- حساسیت کمتری به اختلال داده‌ها دارد.

نقاط ضعف

- پیچیدگی بیشتر از درخت تکی دارد.
- زمان اجرا و منابع محاسباتی بیشتری نسبت به درخت تکی مصرف می‌کند.
- نتایج آن قابل تفسیر نیستند مانند درخت تکی.

- Support vector machines (SVM)

الگوریتم ماشین بردار پشتیبان [۵]یک روش پیشبینی است که در مسائل دسته‌بندی و رگرسیون استفاده می‌شود، از جمله پیش‌بینی خطا. این الگوریتم بر پایه ایجاد یک صفحه جداکننده بین داده‌های دو دسته مختلف عمل می‌کند. هدف الگوریتم ماشین بردار پشتیبان، یافتن یک صفحه جداکننده بهینه است که بین دو دسته داده، حاشیه بیشینه را داشته باشد. حاشیه بیشینه، فاصله‌ای است که بین داده‌های هر دسته و صفحه جداکننده وجود دارد. با افزایش حاشیه، احتمال دسته‌بندی اشتباه داده‌های جدید کاهش می‌یابد.

---

[۴] Random Forest

[۵] Support Vector Machines

نقاط قوت

- قدرت بالا در شناسایی الگوها با وجود ویژگی‌های بسیار.
- درصورت تنظیم مناسب پارامترها دقت بسیار بالایی دارد.
- می‌تواند با ابعاد بالای داده مقابله کند.

نقاط ضعف

- پیکربندی پارامترهای آموزشی دشوار است.
- اجرا و محاسبه‌اش زمان‌برتر از سایر الگوریتم‌هاست.
- نتایج آموخته‌شده توسط آن قابل تفسیر نیستند.

۳-۲ جمع‌بندی

با توجه به تحلیل مقایسه‌ای الگوریتم‌های مختلف پیش‌بینی خطاهای نرم‌افزاری که در بالا آمده است، می‌توان نتیجه گرفت که هیچ الگوریتمی بدون نقاط ضعف نیست و عملکرد آنها به نوع مسئله و مشخصات داده‌ها بستگی دارد. الگوریتم Naive Bayes به دلیل سادگی بالا مزایایی همچون سرعت و سادگی پیاده‌سازی دارد اما فرض استقلال ویژگی‌های آن محدودیت‌هایی ایجاد می‌کند. الگوریتم‌هایی مانند درخت تصمیم و جنگل تصادفی قادر به مدل‌سازی روابط پیچیده‌ترند اما پیچیدگی بیشتری دارند. بنابراین هیچ الگوریتمی به طور مطلق برتری ندارد و باید مناسب ماهیت مسئله باشد. علاوه بر ماهیت مسئله، حجم و ویژگی‌های داده نیز در انتخاب الگوریتم مؤثر است. برخی الگوریتم‌ها مانند ماشین بردار پشتیبان برای داده‌های با ابعاد بالا مناسب‌ترند در حالی که دیگران مانند درخت تصمیم محدودیت‌هایی در این زمینه دارند. بنابراین باید همه این موارد را در نظر گرفت تا الگوریتم کارآمدتری انتخاب شود.

فصل سوم:روش پیاده‌سازی شده و نتیجه‌گیری

۳-۱ مقدمه

پروژه حاضر به بررسی روشی برای بهبود عملکرد الگوریتم K-Means با استفاده از الگوریتم پیاده‌سازی شده مرکز نقطه می‌پردازد. این الگوریتم، مشکل انتخاب تصادفی نقاط مرکزی در الگوریتم K-Means را برطرف می‌کند و سپس آن را برای پیش‌بینی خطاهای ماژول‌های نرم‌افزاری استفاده می‌کند. در این پروژه، از دسته‌بندی خوشه‌ای به عنوان یک روش یادگیری ماشین بدون ناظر برای پیش‌بینی عیب نرم‌افزار استفاده می‌شود. این روش مفید برای تمرین‌کنندگان نرم‌افزار است زیرا نیاز به داده‌های آموزش برچسب دار را کاهش می‌دهد. در این پروژه، روشی برای بهبود کیفیت نرم‌افزار با استفاده از یادگیری ماشین پیشنهاد شده است. برای ارزیابی روش پیاده‌سازی شده ، ده مجموعه داده استفاده شده است. نه مجموعه داده برای پیش‌بینی عیب نرم‌افزار استفاده شده و یک مجموعه داده برای آزمایش الگوریتم خوشه‌بندی استفاده شده است. نتایج نشان می‌دهد که الگوریتم مرکز نقطه پیاده‌سازی شده با خطاهای کمتری مواجه است و عملکرد الگوریتم K-Means را بهبود می‌بخشد. بر اساس نتایج به دست آمده، این پروژه به توسعه یک مدل خوشه‌بندی برای کار با داده‌ها، مانند پیش‌بینی ماژول‌های عیب نرم‌افزاری با دقت بیشتر کمک می‌کند. با توجه به اهمیت موضوع، تلاش برای بهبود روش‌های پیش‌بینی عیب نرم‌افزار و توسعه مدل‌های خوشه‌بندی مناسب، این پروژه می‌تواند به عنوان یک منبع مفید برای پژوهشگران و افراد مشتاق در زمینه نرم‌افزار و یادگیری ماشین باشد.

۳-۲ ساختار روش پیاده‌سازی شده

روش پیاده‌سازی شده در پروژه دارای مراحل زیر است:

۱. پیش پردازش داده: داده‌های از دست رفته را بررسی کنید و مقادیر از دست رفته را جای گزین کنید.

۲. محاسبه الگوریتم مرکز نقطه: مقدار k (تعداد خوشه‌ها) و مقادیر مرکز اولیه را با استفاده از الگوریتم مرکز نقطه تعیین کنید.

۳. انجام خوشه‌بندی K-Means: داده‌ها را با استفاده از الگوریتم K-Means با مقادیر مرکز اولیه به دست آمده از مرحله ۲ خوشه‌بندی کنید.

۴. محاسبه میزان خطا و شاخص رند: نتایج خوشه‌بندی را با مقایسه خوشه‌های به دست آمده با خوشه‌های واقعی با استفاده از یک ماتریس سردرگمی ارزیابی کنید. میزان خطا و شاخص رند را محاسبه کنید.

۵. مقایسه با K-Means: با انتخاب تصادفی مرکزهای اولیه، خوشه‌بندی ساده K-Means را انجام دهید. مقایسه نتایج با روش پیاده‌سازی شده برای نشان دادن عملکرد بهبود یافته در تعیین مرکز اولیه با استفاده از الگوریتم مرکز نقطه.

بنابراین به طور خلاصه، روش پیاده سازی ابتدا الگوریتم مرکز نقطه را برای تعیین تعداد خوشه‌های k و مقادیر مرکز اولیه برای خوشه‌بندی K-Means به روشی بهینه اعمال می‌کند. سپس خوشه‌بندی K-Means را انجام می‌دهد و نتایج را ارزیابی می‌کند و عملکرد بهتری را نسبت به K-Means ساده با مرکزهای اولیه تصادفی نشان می‌دهد.

۳-۳ پیاده‌سازی روش

این پروژه از زبان پایتون[۶] بر روی بستر گوگل کولب[۷] برای پیاده‌سازی استفاده کرده است. همچنین برای ذخیره پروژه بر روی مخزن از گیتهاب[۸] استفاده شده است. برای عملی شدن پیاده سازی ابتدا بطور کلی پروژه خوانده شده است و سپس مرحله به مرحله با استفاده از پایتون پیاده سازی گشته است.

---

[۶] Python

[۷] Google Colab

[۸] Github

۳-۴ روش ارزیابی

این مطالعه از مجموعه دادههای MDP ناسا مخزنPROMISE استفاده می کند. هر مجموعه داده MDP ناسا از چندین ماژول نرم افزاری و ویژگی های ویژه تشکیل شده است. ماژول هایی که دارای عیوب هستند به عنوان خطاهای مستعد و ماژول های غیر معیوب به عنوان مستعد خطا طبقه بندی می شوند.

آزمایش ها با استفاده از رایانه برای انجام فرآیند محاسبه روش پیاده‌سازی شده بود.رایانه مدل ASUS R542UN است و سیستم عامل Windows10 Pro 64 است.

۳-۴-۱ مجوزها

دیتاست این پروژه در مخزن PROMISS ناسا در دسترس عموم قرار دارد و ما از آن استفاده کردیم.

۳-۵ نتایج

در این پروژه ، یک الگوریتم بهبود یافته برای خوشه‌بندی k-means با نام الگوریتم مرکز نقطه پیشنهاد شده است. این الگوریتم با رفع مشکل انتخاب تصادفی مرکزهای خوشه در k-means، عملکرد این الگوریتم را بهبود می‌بخشد. سپس این الگوریتم مرکز نقطه برای پیش‌بینی خطاهای ماژول‌های نرم‌افزاری استفاده شده است. نتایج نشان داد که الگوریتم مرکز نقطه، خطاهای کوچکتری را نسبت به مقدار مرکز به‌دست آمده به‌صورت تصادفی در الگوریتم ساده k-means نسبت به نرم‌افزار اصلاح کرده است. این یافته‌ها مفید و مؤثر در توسعه مدل خوشه‌بندی برای پردازش داده‌ها می‌باشد و بهبود دقت پیش‌بینی خطاهای نرم‌افزار را ایجاد می‌کند.

جدول ۱: خروجی الگوریتم خوشه بندی K-means ساده

Simple K-Means Table

Data	True Negative	False Positive	False Negative	True Positive	Error	rand index
CM1	300	2	42	0	0.12790697674418605	0.872093023255814
KC1	1739	31	283	42	0.14988066825775656	0.8501193317422434
KC3	158	6	31	5	0.185	0.815
MC2	80	1	38	6	0.312	0.688
MW1	230	6	25	2	0.11787072243346007	0.8821292775665399
PC1	673	1	60	1	0.08299319727891157	0.9170068027210885
PC2	1465	12	15	1	0.01808439383791025	0.9819156061620897
PC3	960	1	138	0	0.1264786169244768	0.8735213830755232
PC4	1188	13	171	7	0.13343002175489485	0.8665699782451052

جدول ۲: خروجی الگوریتم خوشه بندی K-means ترکیبی با Point center

Proposed K-Means Table

Data	True Negative	False Positive	False Negative	True Positive	Error	Rand index
CM1	300	2	42	0	0.12790697674418605	0.872093023255814
KC1	1739	31	283	42	0.14988066825775656	0.8501193317422434
KC3	163	1	35	1	0.18	0.82
MC2	80	1	38	6	0.312	0.688
MW1	230	6	25	2	0.11787072243346007	0.8821292775665399
PC1	673	1	60	1	0.08299319727891157	0.9170068027210885
PC2	1470	7	15	1	0.014735432016075016	0.985264567983925
PC3	960	1	138	0	0.1264786169244768	0.8735213830755232
PC4	1201	0	177	1	0.12835387962291517	0.8716461203770849

۳-۶ جمع‌بندی

در این پروژه یک روش بهبود یافته برای خوشه بندی K-Means با استفاده از الگوریتم مرکز نقطه پیشنهاد شده است. این روش که شامل پنج مرحله است، مشکل انتخاب تصادفی مراکز در الگوریتم K-Means را برطرف می کند. سپس از این روش برای پیش بینی خطاهای ماژولهای نرم افزاری استفاده شده است. نتایج نشان داد که این روش مرکز نقطه با خطاهای کمتری نسبت به الگوریتم ساده K-Means با مراکز تصادفی عمل می کند. این روش می تواند برای توسعه مدلهای دقیق تر خوشه بندی و پیش بینی خطاهای نرم افزاری مفید باشد. پیاده سازی روش با استفاده از زبان پایتون روی گوگل کولب انجام شده و از مجموعه داده های PROMISS ناسا استفاده شده است.

منابع

۱. Chat GPT 4
۲. PROMISE DATASETS PAGE (uottawa.ca)
۳. klainfo/NASADefectDataset: NASA Cleaned Defect Datasets (github.com)
٤. Naive theAn improved software defect prediction model based on grey incidence analysis and Bayes algorithm
٥. Software Bug Prediction using Machine Learning Approach
٦. Novel XGBoost Tuned Machine Learning Model for Software Bug Prediction

واژه‌نامه

مخزن – Repository

مرکز نقطه - Point center

مراکز تصادفی- random center

پیوست

لینک کد بر روی Google Colab:

https://colab.research.google.com/drive/۱TI۰PXblOJgAUqYmx۱evTWnAjj_t_mwPk?usp=sharing

Abstract

Error prediction is an estimate that tries to predict future events or errors based on available data and information. In cases where previous data or similar patterns exist, error prediction can be used as a valuable tool in risk analysis and error management. Error prediction in software is an important approach to increase quality and reduce costs. By using different methods of data analysis and machine learning algorithms, errors can be detected before they occur and react in time. The review of various articles shows that methods such as clustering algorithms, neural networks, classification models, and data analysis have been helpful for this purpose. Similar algorithms such as XGBoost, Supervised, Regression, Decision tree, Random Forest, and K-means are used for error prediction. Clustering algorithms, especially K-Means, as an unsupervised learning method, can be helpful for software module error prediction. In this method, by

measuring the characteristics of the modules and dividing them into different clusters, it is possible to identify fault patterns. However, the K-Means algorithm sometimes does not perform well due to the random selection of cluster centers. The present project presents a method to improve the performance of K-Means. In this method, the point center algorithm is used to initially select the centers of the clusters to solve this problem. This improved algorithm is then used to predict software module errors. The test results on datasets show that the proposed method predicts the errors better. Overall, this project provides a suitable solution to improve the performance of the clustering algorithm and more accurately predict errors in the software, which can help increase quality and reduce costs.