ارائـهٔ یـک روش فـازی-تکـاملی بـرای تشـخیص خطاهای نرم‌افزار

مهدی افتخاری*، مریم مجیدی مؤمن آبادی و مجتبی خمر
بخش مهندسی کامپیوتر، دانشکدهٔ فنی و مهندسی، دانشگاه شهید باهنر کرمان، کرمان، ایران

چکیده

تشخیص خطاهای نرم‌افزار، یکی از بزرگ‌ترین چالش‌های توسعهٔ نرم‌افزار است و بیش‌ترین بودجه را در فرآیند توسعهٔ نرم‌افزار به خود اختصاص می‌دهد. با توجه به اهمیت تشخیص خطاهای نرم‌افزار، در این مقاله روشی بر مبنای مجموعه‌های فازی و الگوریتم‌های تکاملی ارائه می‌شود. از آن‌جا که ماهیت مجموعه‌داده‌های تشخیص خطای نرم‌افزار نامتوازن است، از مزایای الگوریتم‌های خوشه‌بندی فازی به‌منظور نمونه‌برداری از داده‌ها و توجه بیشتر به طبقهٔ اقلیت استفاده شده است. روش پیشنهادی درواقع یک الگوریتم ترکیبی است که در ابتدا از روش خوشه‌بندی c میانگین فازی به‌منظور نمونه‌برداری بوت‌استراپ وزن‌دار استفاده می‌شود. وزن داده‌ها همان درجهٔ عضویت آنهاست و درجهٔ عضویت داده‌های طبقهٔ اقلیت افزایش می‌یابد. در گام بعدی، از الگوریتم خوشه‌بندی کاهشی برای ایجاد طبقه‌بند استفاده می‌شود که توسط داده‌های تولیدشده در مرحلهٔ قبل آموزش می‌بینند؛ همچنین از الگوریتم ژنتیک دودویی برای انتخاب ویژگی‌های مناسب استفاده می‌شود. نتایج به‌دست‌آمده و هم‌چنین مقایسهٔ آنها با چندین روش معروف در این زمینه، کارایی مناسب روش پیشنهادی را نشان می‌دهد. برای انجام آزمایش‌ها از ده پایگاه دادهٔ معروف با گسترهٔ وسیعی از اندازه و نرخ عدم توازن، استفاده شده است و برای تأیید نتایج از آزمون آماری تی بهره برده‌ایم.

واژگان کلیدی: الگوریتم های تکاملی، تشخیص خطای نرم‌افزار، طبقه‌بندی، مجموعه داده‌های نامتوازن، منطق فازی.

Proposing an evolutionary-fuzzy method for software defects detection

Mahdi Eftekhari, Maryam Majidi momenabadi & Mojtaba Khamar
Department computer engineering, Shahid Bahonar University of Kerman, Kerman, Iran.

Abstract

Software defects detection is one of the most important challenges of software development and it is the most prohibitive process in software development. The early detection of fault-prone modules helps software project managers to allocate the limited cost, time, and effort of developers for testing the defect-prone modules more intensively. In this paper, according to the importance of software defects detection, a method based on fuzzy sets and evolutionary algorithms is proposed. Due to the imbalanced nature of software defect detection datasets, benefits of fuzzy clustering algorithms were used to data sampling and more attention to the minority class. This method is a combined algorithm which, firstly has used fuzzy c-mean clustering as weighted bootstrap sampling. Weight of data (their membership's degrees) increases for minority class. In the next step, the subtractive clustering algorithm is applied to produce the classifier which was trained by produced data in the previous step. The binary genetic algorithm was utilized to select appropriate features. The results and also comparisons with eight popular methods in software defect detection literature, show an acceptable performance of the proposed method. The experiments were performed on ten real-world datasets with a wide range of data sizes and imbalance rates. Also T-test is used as the statistical significance test for pair wise comparison of our proposed method against the others. The final results of T-test are shown in tables

* Corresponding author
*نویسندهٔ عهده‌دار مکاتبات • تاریخ ارسال مقاله: ۱۳۹۶/۶/۲۴ • تاریخ آخرین بازنگری: ۱۳۹۷/۶/۱۴ • تاریخ پذیرش: ۱۳۹۷/۱۰/۱۹

[DOI: 10.29252/jsdp.15.4.3]

for three performance measures (G-mean, AUC and Balanced) over various datasets. (As the obtained results apparently show our proposed method has the ability to improve three aforementioned performance criteria simultaneously). Some methods just have improved the G-mean measure while the AUC and Balance criteria have lower values than the others. Securing a high level of three performance measures simultaneously illustrates the ability of our proposed algorithm for handling the imbalance problem of software defects detection datasets.

Keywords: **classification, evolutionary algorithm, fuzzy logic, imbalance datasets, software defect detection.**

۱- مقدّمه و پیشینه

در سال ۲۰۰۵ که منبع دادهٔ پرومیس[1] تولید شد، این امکان به پژوهش‌گران داده شد که بتوانند مطالعات و آزمایش‌های خود را با هم مقایسه کنند. در این منبع داده، مجموعه‌داده‌های خطای نرم‌افزار وجود دارد که بر اساس ویژگی‌های مختلف نرم‌افزار، خطاهای نرم‌افزار را در دسته‌های مختلف دسته‌بندی می‌کنند. پس از آن، پژوهش‌ها در زمینهٔ تشخیص خطای نرم‌افزار بسیار افزایش پیدا کرد [1].

با افزایش اندازه و پیچیدگی سامانه‌های نرم‌افزاری، صنعت نرم‌افزار با چالش تحویل سامانهٔ باکیفیت، قابل اعتماد، به‌موقع و با هزینهٔ تعیین‌شده مواجه شد. برای کیفیت، تعاریف زیادی وجود دارد؛ امّا یکی از تعاریف قابل قبول برای کیفیت بالا، خطای کم است [2]. خطاهای نرم‌افزاری به‌طورمعمول به‌عنوان انحراف از خصوصیات مورد انتظار است که منجر به شکست‌هایی درعمل در زمان آینده می‌شود.

برای کم‌کردن خطای سامانه‌های نرم‌افزاری به آزمایش نرم‌افزار نیاز است؛ امّا حدود ۵۰ تا ۶۰ درصد هزینهٔ توسعهٔ نرم‌افزار به آزمایش مربوط می‌شود [3]. به‌دلیل هزینه‌بربودن آزمایش کلیهٔ ماژول‌های یک سامانهٔ نرم‌افزاری بزرگ، لازم است، فقط ماژول‌هایی که احتمال خطای بیشتری دارند، آزمایش شوند؛ بنابراین احتیاج به تشخیص ماژول‌های مستعد خطا[2] پیدا شد. با تشخیص ماژول‌های مستعد خطا در ابتدای یک پروژهٔ نرم‌افزاری، مدیران پروژه‌های نرم‌افزاری می‌توانند زمان، هزینه و تلاش تیم توسعه را بیشتر صرف آزمایش ماژول‌های مستعد خطا کنند و دیگر نیاز به آزمایش کل سامانهٔ نرم‌افزاری نیست. آنها همچنین می‌توانند برای پیشبرد پروژهٔ نرم‌افزاری برنامه‌ریزی، کنترل و اجرای بهتر داشته باشند و با تحویل سامانهٔ با کیفیت، با هزینه تعیین‌شده و به‌موقع رضایت مشتری را بیشتر جلب می‌کنند [4]. نکتهٔ بسیار مهم این است که ماژول‌های مستعد خطا، بخش کوچکی از کل سامانهٔ نرم‌افزار را تشکیل می دهند. به‌عبارت دیگر، به‌دلیل ماهیت مسألهٔ تشخیص خطای نرم‌افزار، یک نمونهٔ خطا نسبت به یک نمونهٔ بدون خطا، به احتمال کم‌تری اتفاق می‌افتد؛ بنابراین داده‌های تشخیص خطای نرم‌افزار نامتوازن هستند؛ به این معنا که تعداد نمونه‌های طبقهٔ بدون خطا بیشتر از تعداد نمونه‌های طبقهٔ خطادار است؛ بنابراین تمرکز این مقاله بر روی مسئلهٔ طبقه‌بندی با توزیع نامتوازن طبقه است.

مدل‌های تشخیص خطای نرم‌افزار می‌توانند ماژول‌های نرم‌افزاری را به دو طبقهٔ ماژول‌های خطادار و بدون خطا اختصاص دهند؛ امّا استفاده از طبقه‌بندهای استاندارد برای مسائل نامتوازن مناسب نیست؛ زیرا اغلب الگوریتم‌های یادگیری ماشین فرض می‌کنند که تعداد نمونه‌های آموزشی در طبقه‌های متفاوت برابر است. درنتیجه طبقه‌بند یادگرفته‌شده اغلب از طبقهٔ اکثریت نتیجه می‌شود که این موضوع به‌دلیل آن‌که آموزش طبقهٔ اقلیت به‌درستی انجام نشده است به پیش‌بینی بسیار ضعیف از طبقهٔ اقلیت منجر می‌شود. بنابراین کنترل و حل مسأله دادهٔ نامتوازن برای بهبود کارایی پیش‌بینی‌کننده‌های خطای نرم‌افزار امری ضروری است. روش‌های عددی برای حل مسائل دادهٔ نامتوازن به دو دستهٔ اصلی تقسیم می‌شوند:

۱. نمونه‌گیری از داده که خود به سه دستهٔ زیادکردن طبقهٔ اقلیت، کم‌کردن طبقهٔ اکثریت و روش‌های ترکیبی تقسیم می‌شود.

۲. اصلاح الگوریتم‌های موجود یا ارائهٔ یک الگوریتم جدید که شامل سه روش یادگیری حساس به هزینه، یادگیری مبتنی بر هسته و یادگیری مبتنی بر تشخیص است [5].

درصورتی‌که یک نمونه، خطادار باشد، ولی به‌اشتباه آن را بدون خطا طبقه‌بندی کنیم، کیفیت نرم‌افزار به‌شدت پایین می‌آید. درحالی‌که طبقه‌بندی اشتباه یک نمونهٔ بدون خطا به‌عنوان خطادار، هزینهٔ آزمایشِ فقط یک ماژول را درپی خواهد داشت. بنابراین هزینهٔ طبقه‌بندی اشتباه طبقهٔ اقلیت، بسیار بیشتر از هزینهٔ طبقه‌بندی اشتباه طبقهٔ اکثریت است. با توجه به این مسأله، بسیار مهم است که بتوانیم طبقهٔ خطادار یا همان طبقهٔ اقلیت را به‌درستی پیش‌بینی کنیم [1].

[1] PROMISE
[2] Defect-prone modules

[DOI: 10.29252/jsdp.15.4.3]

مدل‌های پیش‌بینی خطا که رابطهٔ بین معیارهای نرم‌افزارو ماژول‌های مستعد خطا را پیدا می‌کنند، از روش‌های مختلف پیش‌بینی استفاده کرده‌اند. از جملهٔ مهم‌ترین و متداول‌ترین آنها می‌توان به روش‌های طبقه‌بندی فازی –[5] [7]، الگوریتم‌های تکاملی [10]–[8]، شبکه‌های عصبی مصنوعی [13]–[7] ,[5]–[2]، ماشین‌های بردار پشتیبان –[8] [19]–[12] ,[10] اشاره کرد.

وانگ و همکارش در سال ۲۰۱۲ در مورد این مسأله پژوهش کردند که چه‌طور روش‌های یادگیری طبقه‌های نامتوازن می‌توانند به حل مسألهٔ تشخیص خطای نرم‌افزارکمک کنند [1]. آنها انواع مختلف روش‌های یادگیری طبقه‌های نامتوازن را (در قالب هفت الگوریتم) بررسی کردند که شامل سه نوع روش نمونه‌گیری مجدد، روش‌های حرکت حد آستانه[1] و الگوریتم‌های جمعی[2] می‌شدند. برای به‌دست‌آوردن بیشینهٔ پتانسیل هر یک از روش‌ها، در ابتدا آنها بهترین مقادیر پارامترها را برای هر روش جستجو کردند. این جستجو برای به‌دست‌آوردن بهترین پارامترها مبتنی بر معیارهای تعادل، میانگین هندسی و سطح زیر نمودار[3] انجام شد؛ سپس بهترین مقادیر پارامترها برای هر روش تنظیم و الگوریتم‌ها با یکدیگر مقایسه شدند. نتیجهٔ مقایسهٔ این روش‌ها این بود که الگوریتم آدابوست ان سی[4] بهترین کارایی را در معیارهای میانگین هندسی، سطح زیر نمودار و تعادل به‌دست آورد. آنها برای ارتقای کارایی الگوریتم آدابوست ان سی، یک نسخهٔ پویا از آن پیشنهاد دادند و نام الگوریتم پیشنهادی خود را آدابوست ان سی پویا[5] گذاشتند. الگوریتم آنها قادر است، در حین یادگیری، پارامترهای خود را تنظیم کند؛ بدون آن‌که نیاز باشد هیچ پارامتری از قبل تعریف شود. در انتها آنها الگوریتم خود را با الگوریتم های آدابوست ان سی و بیز ساده[6] مقایسه کردند. آنها گزارش دادند که روش پیشنهادی‌شان کارآمدتر و مؤثرتر از آدابوست ان سی اصلی و بیز ساده است.

یان و همکارانش در سال ۲۰۱۰ یک روش جدید با استفاده از رگرسیون بردار پشتیبان فازی[7] برای تشخیص تعداد خطاهای نرم‌افزار پیشنهاد کردند [6]. ورودی فازی‌ساز رگرسور می‌تواند مجموعه‌داده‌های معیارهای نرم‌افزار نامتعادل را مدیریت کند. نتایج آزمایش‌های آنها روی مجموعه‌داده‌های MIS و RSDIMU نشان داد وقتی از ماژول‌هایی که شامل تعداد زیادی خطا هستند، استفاده می‌شود، در مقایسه با دیدگاه رگرسیون بردار پشتیبان، رگرسیون بردار پشتیبان فازی می‌تواند میانگین مربعات خطای کمتر و دقت بیش‌تری در تعداد کل خطاها به‌دست آورد. رّسلی و همکارانش در سال ۲۰۱۱ با استفاده از الگوریتم ژنتیک کاربردی برای پیش‌بینی خطای نرم‌افزار طراحی [10] و برای انجام آزمایش‌های خود، از معیارهای یک نرم‌افزار منبع باز به‌عنوان ورودی الگوریتم ژنتیک استفاده کردند. هدف آنها از طراحی پیشنهادشده، توسعهٔ یک ابزار خودکار برای توسعه‌دهندگان نرم‌افزار است که بتوانند تشخیص دهند کدام ماژول‌های نرم‌افزاری احتمال بیشتری دارد که در آینده باعث ایجاد مشکل شوند.

بخش‌های بعدی مقاله به‌شرح زیر ارائه خواهد شد: در بخش دو به توضیح در مورد روش خوشه‌بندی فازی استفاده‌شده در این مقاله پرداخته می‌شود. در بخش سه روش پیشنهادی توضیح داده شده است. در بخش چهار به تحیل و بحث در مورد نتایج پرداخته می‌شود و در بخش پنج با نتیجه‌گیری که حاصل بررسی روش پیشنهادی و دیگر روش‌ها می‌باشد مقاله پایان می‌یابد.

۲- خوشه‌بندی فازی

خوشه‌بندی یکی از شاخه‌های یادگیری بدون نظارت، و فرآیند خودکاری است که در طی آن، نمونه‌ها به دسته‌هایی که اعضای آن مشابه یکدیگرند، تقسیم می‌شوند که به این دسته‌ها *خوشه* گفته می‌شود. برای مشابه‌بودن می‌توان معیارهای مختلفی مثل معیار فاصله را درنظر گرفت که در آن اشیای نزدیک‌تر به هم به‌عنوان یک خوشه درنظر گرفته می‌شود. به این نوع خوشه‌بندی، *خوشه‌بندی مبتنی بر فاصله* نیز گفته می‌شود. در خوشه‌بندی کلاسیک، هر نمونهٔ ورودی متعلّق به یک و فقط یک خوشه است و نمی‌تواند عضو دو خوشه و یا بیش‌تر باشد و به زبان دیگر خوشه‌ها هم‌پوشانی ندارند. حال، حالتی را در نظر بگیرید که میزان تشابه یک نمونه با دو خوشه و یا بیش‌تر یکسان باشد. تفاوت اصلی خوشه‌بندی کلاسیک و خوشه‌بندی فازی در این است که در خوشه‌بندی فازی، یک نمونه می‌تواند متعلّق به بیش از یک خوشه باشد. چنان‌چه ایدهٔ روشنی درمورد تعداد خوشه‌ها وجود نداشته باشد، استفاده از خوشه‌بندی کاهشی روشی سریع برای حل این موضوع محسوب می شود. گاهی اوقات مراکزی که با این روش تخمین زده شده‌اند، به‌عنوان نقاط اولیه برای دیگر الگوریتم‌های خوشه‌بندی مورد استفاده قرار می‌گیرند. خوشه‌بندی کاهشی که خوشه‌بندی تفاضلی نیز

[1] Threshold-moving
[2] Ensemble learning
[3] Area Under Curve(AUC)
[4] AdaBoost.NC
[5] Dynamic version of AdaBoost.NC(DNC)
[6] Naive Bayes
[7] Fuzzy Support Vector Regression (FSVR)

[DOI: 10.29252/jsdp.15.4.3]

نامیده می‌شود، در اصل یک فرم تغییریافته از روش خوشه‌بندی کوهستان[1] است. در این الگوریتم هر نقطه به‌عنوان یک پتانسیل برای مرکز خوشه در نظر گرفته می‌شود که اندازه‌گیری پتانسیل طبق معادلهٔ زیر به‌دست می‌آید [6]:

$$P_i = \sum_{j=1}^{N} e^{\alpha \|x_i - x_j\|^2} \quad (۱)$$

$$\alpha = \frac{4}{(r_a)^2} \quad (۲)$$

که x_i بیان‌گر نمونهٔ i ام است. هم‌چنین مقدار α طبق رابطهٔ (۲) به‌دست می‌آید. $r_a > 0$ مقداری مثبت است که به‌عنوان شعاع همسایگی برای هر مرکز خوشه تعریف می‌شوند. بنابراین پتانسیل تخصیص داده‌شده به هر خوشه به فاصلهٔ آن از نقاط دیگر وابسته است و در مواردی با همسایهٔ متراکم، منجر به خوشه‌هایی با پتانسیل بالا می‌شود. بعد از محاسبهٔ پتانسیل برای هر نقطه، یک نقطه که دارای بالاترین پتانسیل است به‌عنوان مرکز خوشه انتخاب می‌شود. ابتدا x_1 به‌عنوان مرکز نخستین گروه با پتانسیل P_1 تعیین می‌شود. سپس پتانسیل P_i برای x_i مطابق معادلهٔ (۳) کاهش می‌یابد [6]:

$$P_i = P_i - p_1 e^{(-\beta \|x_i - x_j\|^2)} \quad (۳)$$

$$\beta = \frac{4}{{r_b}^2} \quad (۴)$$

که مقدار β از رابطهٔ (۴) به‌دست می‌آید و $r_b > 0$ عددی مثبت است که همسایه‌ای را با کاهش قابل توجه‌ای در چگالی تعیین می‌کند. بنابراین، مقدار چگالی نقطهٔ داده‌ای که نزدیک به نخستین خوشه است، به‌طور قابل توجهی کاهش می‌یابد. این رویّه (انتخاب مراکز و کاهش پتانسیل) به‌کرّار اجرا شده تا ضوابط و معیارهای توقف تأمین شود. خروجی خوشه‌بندی کاهشی، یک سامانهٔ استنتاج فازی سوگنو است. یک قانون سوگنو به‌صورت زیر تعریف می‌شود [6]:

$$\text{Rule1: If x is } A_1 \text{ and y is } B_1 \text{then } f_1 = p_1 x + q_1 y + r_1 \quad (۵)$$

که x, y متغیرهای ورودی، $A_1 ، B_1$ مجموعه‌های فازی و f_1 تابع خروجی را نشان می‌دهد. خروجی نهایی برای یک نمونه در یک سامانهٔ فازی سوگنو با استفاده از روش میانگین وزن‌دار که در رابطهٔ (۶) نشان داده شده است، به‌دست می‌آید.

$$z = \frac{\sum_{i=1}^{N} w_i f_i}{\sum_{i=1}^{N} w_i} \quad (۶)$$

که w_i قدرت شلیک[2] قانون i ام را بیان می‌کند.

۱-۲- خوشه بندی فازی c- میانگین

مشابه الگوریتم c-میانگین کلاسیک در این الگوریتم نیز تعداد خوشه‌ها (c) از قبل مشخّص شده است. تابع هدفی که برای این الگوریتم تعریف‌شده به‌صورت زیر است [20] ,[15]:

$$J = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik} {d_{ik}}^2 = \sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^m \| x_k - v_i \|^2 \quad (۷)$$

در فرمول بالا m یک عدد حقیقی بزرگ‌تر از یک است که در بیش‌تر موارد برای m عدد دو انتخاب می‌شود. x_k نمونه k ام و v_i نماینده یا مرکز خوشه i ام است. u_{ik} میزان تعلّق نمونه k ام در خوشه i ام را نشان می‌دهد. علامت $\|.\|$ میزان تشابه (فاصله) نمونه با (از) مرکز خوشه است که می‌توان از هر تابعی که بیان‌گر تشابه نمونه و مرکز خوشه باشد، استفاده کرد. از روی u_{ik} می‌توان یک ماتریس u تعریف کرد که دارای c سطر و n ستون است و مؤلفه‌های آن می‌توانند هر مقداری بین صفر تا یک را اختیار کنند. اگر تمامی مؤلفه‌های ماتریس u به‌صورت صفر و یا یک باشند، الگوریتم مشابه c-میانگین کلاسیک خواهد بود. با این‌که مؤلفه‌های ماتریس u می‌توانند هر مقداری بین صفر تا یک را اختیار کنند، امّا مجموع مؤلفه‌های هر یک از ستون‌ها باید برابر یک باشد و داریم [20] ,[6]:

$$\sum_{i=1}^{c} u_{ik} = 1, \forall\, k = 1, \dots, n \quad (۸)$$

معنای این شرط این است که مجموع تعلّق هر نمونه به c خوشه باید برابر یک باشد. با استفاده از شرط بالا و کمینه‌کردن تابع هدف خواهیم داشت [20] ,[6]:

$$u_{ik} = \left(\frac{1}{\sum_{j=1}^{c} \frac{d_{ik}}{d_{jk}}}\right)^{2/m-1} \quad (۹)$$

$$v_i - \frac{\sum_{k=1}^{n} u_{ik}^m x_k}{\sum_{k=1}^{n} u_{ik}^m} \quad (۱۰)$$

[1] Mountain

[2] Firing strength

[DOI: 10.29252/jsdp.15.4.3]

۳- روش پیشنهادی

بهینه‌سازی، یکی از مهم‌ترین موضوعات در زمینهٔ مهندسی محسوب می‌شود و هدف آن، تنظیم ورودی‌های مسأله و ایجاد نتایج بهینه است. در بسیاری از مسائل، انتخاب بهترین جواب از طریق جستجوی کل فضای مسأله عملی زمان‌بر است؛ بنابراین یک راه‌حل مناسب، استفاده از الگوریتم‌های تکاملی است. بر این اساس در روش پیشنهادی، ابتدا از روش‌های خوشه‌بندی فازی برای تولید یک مدل استنتاج فازی از نوع سوگنو جهت تشخیص خطای نرم‌افزاراستفاده و سپس با استفاده از الگوریتم‌های تکاملی مدل‌های به‌دست آمده بهینه می‌شود. این کار از طریق تنظیم پارامترهای مدل‌های فازی انجام خواهد شد. روندنمای روش پیشنهادی در شکل (۱) آمده است. در ادامه به توضیح مراحل مختلف روش پیشنهادی می‌پردازیم.

۱. داده‌ها به روشِ اعتبارسنجیِ متقابلِ ده‌تایی[1] تقسیم شده‌اند. بنابراین اجرای الگوریتم پیشنهادی با رویهٔ ده‌تایی، ده مرتبه تکرار شد. هم‌چنین برای به‌دست‌آوردن نتایج قابل اعتمادتر، رویهٔ پیشنهادی نیز ده مرتبه تکرار شده است.

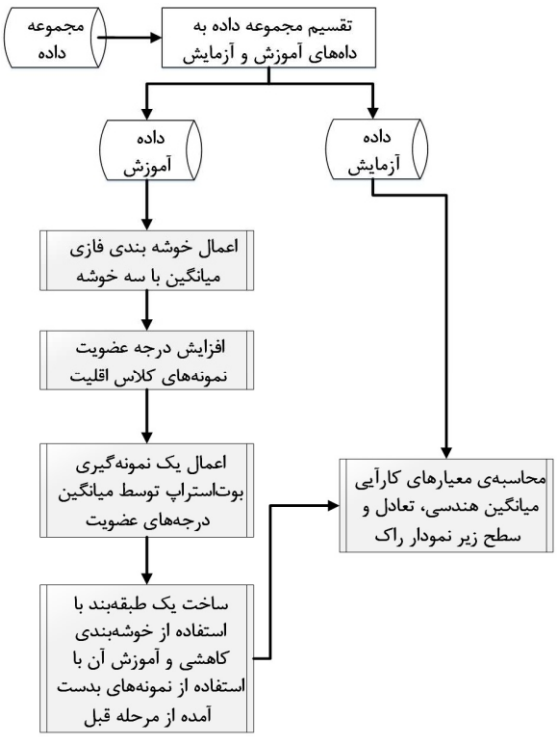
۲. خوشه‌بندی c میانگین فازی با تعداد خوشه‌های سه اجرا شد. البته در ابتدا با توجه به این‌که هدف اولیه، خوشه‌بندی مجموعه داده به دو دسته بود، یک دسته برای نمونه‌های طبقهٔ اکثریت و دیگری برای نمونه‌های طبقهٔ اقلیت، تعداد خوشه‌ها دو درنظر گرفته شد؛ امّا بعد از چندین سعی و خطا، بهترین نتایج با تعداد خوشه‌های برابر سه به‌دست آمد.

۳. درجه‌های عضویت نمونه‌ها در خوشه‌ها برای اجرای نمونه‌گیری بوت‌استراپ وزن‌دار در مرحلهٔ بعد اصلاح شدند. ازاین‌رو درجه‌های عضویت نمونه‌های اقلیت توسط نرخ عدم توازن افزایش یافتند. این افزایش درجهٔ عضویت بر اساس رابطهٔ زیر صورت می‌گیرد:

$$\mu(x_i) = \mu(x_i) + (IR \times Coef) \qquad (۱۱)$$

به‌طوری x_i نمونه‌ای است که به طبقهٔ اقلیت تعلّق دارد و IR نرخ عدم توازن که برابر است با نسبت تعداد نمونه‌های طبقهٔ اکثریت به تعداد نمونه‌های کلاس اقلیت. مسألهٔ تشخیص خطا جزو مسائل نامتوازن و تعداد نمونه‌های طبقهٔ اکثریت همواره بسیار بیشتر از تعداد نمونه‌های طبقهٔ اقلیت است؛ بنابراین نرخ عدم توازن مقداری بیشتر از یک خواهد بود. مقدار $Coef$ را در بیش‌تر مجموعه‌داده‌ها برابر با ۰/۵ در نظر گرفتیم.

[1] 10 fold cross validation



(شکل-۱): روندنمای روش پیشنهادی
(Figure-1): Flow chart of proposed method

۴. درجه عضویت‌های اصلاح‌شده، به‌عنوان پارامترهای ورودی به الگوریتم نمونه‌گیری بوت‌استراپ داده می‌شوند. اضافه‌کردن وزن ممکن است، باعث افزایش درجه‌های عضویت به مقداری بیش از یک شود؛ بنابراین در الگوریتم بوت‌استراپ، ابتدا درجه‌های عضویت اصلاح‌شده را نرمال می‌کنیم که به مفهوم نگاشت مقادیر بین صفر و یک است؛ سپس احتمال هر درجه عضویت اصلاح‌شده و نرمال‌شده را از طریق رابطهٔ (۱۲) به‌دست می‌آوریم.

$$pr = \frac{max_{un}}{sum(max_{un})} \qquad (۱۲)$$

که در آن max_{un}، درجهٔ عضویت اصلاح و نرمال‌شده است؛ سپس احتمال تجمعی را برای احتمالات درجه‌های عضویت، با استفاده از رابطهٔ (۱۳) محاسبه می‌کنیم.

$$Cpr(i) = Cpr(i-1) + pr(i) \qquad (۱۳)$$

سپس با انتخاب چرخ گردان، اندیس نمونه‌های انتخابی را به‌دست می‌آوریم.

۵. با استفاده از خوشه‌بندی کاهشی و داده‌های آموزشی، طبقه‌بند ساخته می‌شود. این طبقه‌بند درواقع یک سامانهٔ استنتاج سوگنوی فازی است که از تصویرکردن خوشه‌ها روی ابعاد مختلف به‌دست می‌آید. از آنجا که خروجی سامانه‌های سوگنو به‌طورمعمول مقدار پیوسته‌ای است،

[DOI: 10.29252/jsdp.15.4.3]

به‌منظور گسسته‌سازی خروجی از تابع جزء صحیح استفاده می‌کنیم. همچنین می‌توان از تابع سیگموید[1] جهت مقیاس‌کردن خروجی سامانهٔ فازی بین صفر و یک استفاده کرد و بدین ترتیب از سامانهٔ سوگنوی فازی به‌عنوان یک طبقه‌بند استفاده کرد.

تعداد قوانین در سامانهٔ سوگنوی فازی تولیدشده توسط خوشه‌بندی کاهشی به شعاع خوشه‌بندی وابسته است. مقادیر شعاع بزرگ باعث تولید قوانین کمتر و مقادیر کوچک، باعث تولید قوانین بیشتر می‌شود. در این مقاله برای هر قانون، دو خروجی به‌صورت زیر در نظر گرفته شده است. دلیل این کار، راحت‌بودن در استفاده از ماتریس گیجی بوده است.

If x_1 is A_1 and x_2 is B_1 then $y_1^{(1)} = 0.2x_1 - 0.3x_2 - 1$ and $y_2^{(1)} = 0.3x_1 + 0.4x_2 - 2$.
If x_1 is A_2 and x_2 is B_2 then $y_1^{(2)} = 5x_1 - 3x_2 + 4$ and $y_2^{(2)} = 4x_1 + 5x_2 - 5$.

$$W_1 = \mu_{A_1}(x_1).\mu_{B_1}(x_2) \quad W_2 = \mu_{A_2}(x_1).\mu_{B_2}(x_2),$$

$$Y_1^{Total} = \frac{W_1 y_1^{(1)} + W_2 y_1^{(2)}}{W_1 + W_2}$$

$$Y_2^{Total} = \frac{W_1 y_2^{(1)} + W_2 y_2^{(2)}}{W_1 + W_2}$$

درنهایت طبقهٔ خروجی با استفاده از دو مقدار Y_1^{Total} و Y_2^{Total} به‌صورت زیر تعیین می‌شود:

$$Out_1 = sigmoid(Y_1^{Total}) \quad (۱۴)$$

$$Out_2 = sigmoid(Y_2^{Total}) \quad (۱۵)$$

مقادیر $[Out_1, Out_2]$ در کنار یکدیگر طبقه پیشنهادی را تعیین می‌کنند. اگر این مقادیر $[1,0]$ باشند، طبقه نخست و اگر $[0,1]$ باشند، طبقه دوم را تعیین می‌کنند.

۶. در آخرین مرحله، مقادیر معیارهای کارایی برای روش پیشنهادی طبق روابط زیر محاسبه می‌شود[1]:

$$G - mean = \sqrt{Tpr.Tnr} \quad (۱۶)$$

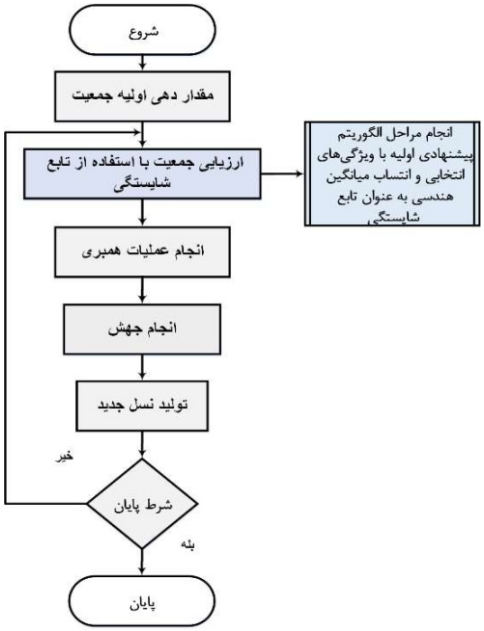
$$AUC = \frac{1 + Tpr - Fpr}{2} \quad (۱۷)$$

$$Balance = 1 - \frac{\sqrt{(1 - Tnr)^2 + (1 - Tpr^2)}}{\sqrt{2}} \quad (۱۸)$$

در الگوریتم پیشنهادی اولیه از تمام ویژگی‌های مجموعه‌داده‌ها، استفاده کردیم. برای بهبود این روش و درنتیجه به‌دست‌آوردن نتایج بهتر، به‌جای استفاده از تمام ویژگی‌های مجموعه‌داده‌ها، مهم‌ترین ویژگی‌ها را انتخاب کردیم. برای به‌دست‌آوردن بهترین انتخاب ویژگی در هر مجموعه‌داده، الگوریتم ژنتیک استفاده شده است. در شکل (۲)، روندنمای الگوریتم ژنتیک استفاده‌شده آمده و سپس توضیحات بیشتر در مورد آن بیان شده است.

این الگوریتم برای هر ده مجموعه‌داده به‌طور جداگانه انجام می‌شود. طول هر کروموزوم در الگوریتم ژنتیک، به تعداد ویژگی‌های هر مجموعه داده هست. به‌ازای هر ویژگی در مجموعه‌داده، صفر و یک در کروموزوم کد می‌شود. صفر به معنای انتخاب‌نشدن ویژگی و یک بدین معناست که ویژگی متناظر آن، انتخاب می‌شود. در ابتدا به‌طور تصادفی اعداد صفر و یک در کروموزوم کد می‌شود؛ سپس الگوریتم پیشنهادی اولیه با ویژگی‌های انتخاب‌شده در کروموزوم، اجرا می‌شود. در هر بار اجرا، شایستگی افراد جمعیت محاسبه می‌شود. همان‌طور که در شکل (۲) نشان داده شده است، برای محاسبهٔ شایستگی افراد جمعیت، الگوریتم پیشنهادی اولیه دومرتبه اجرا که در هر مرتبه از روش اعتبارسنجیِ متقابلِ ده‌تایی استفاده و سپس میانگین مقادیر میانگین هندسی برای محاسبهٔ تابع شایستگی برگردانده می‌شود. تابع شایستگی طبق رابطهٔ (۱۹) درنظرگرفته شده است که باید کمینه شود. این بدان معناست که مقدار میانگین هندسی باید بیشینه شود.

$$Fitness = 1 - Gmean \quad (۱۹)$$



(شکل-۲): روندنمای الگوریتم ژنتیک استفاده‌شده در این مقاله
(Figure -2): Flow chart of genetic algorithm used in this paper

در جدول (۱) پارامترهای الگوریتم ژنتیک استفاده‌شده آورده شده است.

[1] Sigmoid

[DOI: 10.29252/jsdp.15.4.3]

(جدول-۱): پارامترهای الگوریتم ژنتیک
(Table-1): Parameters of Genetic algorithm

پارامتر الگوریتم ژنتیک	مقدار
اندازهٔ جمعیت	۱۰۰
تعداد نسل(تعداد تکرار)	۱۰۰
طول ژن	۱
احتمال همبری	۰/۷
احتمال جهش	۰/۱

۴- نتایج و بحث

در این بخش، کارایی الگوریتم پیشنهادی خود را با هشت الگوریتم متداول در مسئله تشخیص خطا مقایسه می‌کنیم و سپس به تحلیل نتایج می‌پردازیم. هشت الگوریتم پرکاربرد یادگیری طبقه‌های نامتوازن عبارتند از: الگوریتم کاهش طبقهٔ اکثریت به‌طور تصادفی[1]، الگوریتم کاهش کلاس اکثریت به‌طور تصادفی و متعادل[2]، الگوریتم مبتنی بر حرکت حد آستانه[3]، الگوریتم اسموت ارتقایافته[4]، الگوریتم آدابوست ان سی[5]، الگوریتم بیز ساده با لاگ فیلتر[6]، الگوریتم جنگل تصادفی[7]، الگوریتم آدابوست ان سی پویا[8]. مشخصات مجموعه داده‌هایی که در ساخت مدل‌های پیشنهادی استفاده شده‌اند، در جدول (۲) فهرست شده است [1]. این مجموعه‌داده‌ها در انبار دادهٔ پرومیس در دسترس هستند و به‌ترتیب نرخ عدم توازن در جدول قرارگرفته‌اند. به‌عنوان‌مثال، مجموعه‌دادهٔ mc2 دارای ۱۶۱ نمونه و مجموعه دادهٔ jm1 دارای ۱۰۸۸۵ نمونه است. همچنین درصد نمونه‌های خطادار از ۳۲/۲۹ درصد تا ۶/۹۴ درصد متغیر است.

در این مقاله برای اطمینان از وجود تفاوتی معنی‌دار میان روش‌ها از آزمون‌های آماری استفاده شده است [21] منظور از تفاوت معنی‌دار میان روش‌ها این است که اختلاف میان روش‌ها به حدّ کافی بزرگ باشد تا بتوان اطمینان حاصل کرد که نتایج به‌طور تصادفی یا براثر شانس، بهتر نشده‌اند. انتخاب آزمون آماری مناسب، بستگی به تعداد متغیرهای پژوهش، تعداد گروه‌های مقایسه‌شده، مستقل یا وابسته‌بودن گروه‌ها، نرمال‌بودن یا نبودن توزیع داده‌ها و نوع داده‌ها (عددی، رتبه‌ای، اسمی) دارد. اگر نوع داده‌ها کمّی و توزیع آن‌ها نرمال باشد از مدل‌های آماری پارامتریک استفاده می‌کنیم. برای تحلیل نتایج گزارش‌شده در این مقاله، از آزمون تی[9] با دو نمونهٔ مستقل که جزو مدل‌های آماری پارامتریک است، استفاده و هم‌چنین برای ارزیابی کارایی روش‌ها و نشان‌دادن عملی‌بودن نتایج ارائه‌شده در این پژوهش، از سه معیار تعادل، سطح زیر نمودار و میانگین هندسی استفاده کرده‌ایم. این معیارها روی توزیع طبقهٔ داده‌ها حساس هستند و بنابراین برای مسئله تشخیص خطا مناسب هستند. هر چه مقدار این سه معیار بیشتر باشد، پیش‌بینی‌کننده عملکرد بهتری دارد.

(جدول-۲): مجموعه داده‌های پرومیس استفاده‌شده در ساخت مدل‌های پیش‌بینی
(Table-2):PROMISE dataset utilized in obtaining proposed models

مجموعه داده	زبان برنامه‌نویسی	تعداد نمونه	تعداد ویژگی	٪خطا
mc2	C++	۱۶۱	۳۹	۳۲/۲۹
kc2	C++	۵۲۲	۲۱	۲۰/۴۹
jm1	C	۱۰۸۸۵	۲۱	۱۹/۳۵
kc1	C++	۲۱۰۹	۲۱	۱۵/۴۵
pc4	C	۱۴۵۸	۳۷	۱۲/۲۰
pc3	C	۱۵۶۳	۳۷	۱۰/۲۳
cm1	C	۴۹۸	۲۱	۹/۸۳
kc3	Java	۴۵۸	۳۹	۹/۳۸
mw1	C	۴۰۳	۳۷	۷/۶۹
pc1	C	۱۱۰۹	۲۱	۶/۹۴

جدول‌های ۳، ۴ و ۵ نشان‌گر میانگین و انحراف معیار معیارهای میانگین هندسی، تعادل و سطح زیر نمودار است که توسط نُه روش یادگیری بر روی ده مجموعه داده به‌دست آمده‌اند. نتایج مربوط به هشت روش یادگیری غیر از روش پیشنهادی از مرجع [1] آورده شده است. هر عدد در جدول، میانگین صد اجرای مستقل است. مقادیر پررنگ‌شده در هر سطر بیان‌گر این مسئله هستند که از بقیهٔ موارد به‌طور معنادار بهتر هستند. تفاوت معناداری میان چند مقدار پررنگ‌شده در یک سطر، وجود ندارد. جداول ۶، ۷ و ۸ نشان‌دهندهٔ برد، باخت یا مساوی‌بودن نتایج روش پیشنهادی با هشت روش متداول دیگر به‌ترتیب در معیارهای میانگین هندسی، تعادل و سطح زیر نمودار هستند. در این جداول حروف W، L و T به‌ترتیب بیان‌گر برد، باخت و مساوی‌بودن نتیجه هستند. از مشاهدهٔ این جداول متوجه می‌شویم که روش پیشنهادی بهترین نتیجه را در معیارهای میانگین هندسی و تعادل، در شش مجموعه‌داده از ده مجموعه‌داده به‌دست آورد. برای داده‌هایی که نرخ عدم توازن زیادی دارند، مثل مجموعه‌داده‌های cm1، kc3، mw1 و pc1 ، روش پیشنهادی در معیارهای میانگین هندسی و تعادل عملکرد بسیار خوبی دارد.

[1] Random undersampling(RUS)
[2] Balanced random undersampling (RUS-bal)
[3] Threshold-moving(THM)
[4] SMOTEBoost (SMB)
[5] AdaBoost.NC (BNC)
[6] Naive Bayes with the log filter (NB)
[7] Random Forest (RF)
[8] Dynamic AdaBoost.NC(DNC)
[9] T-Test

[DOI: 10.29252/jsdp.15.4.3]

در مقایسهٔ روش پیشنهادی با هشت روش متداول دیگر، درمجموع، روی ده مجموعه‌داده، تعداد ۶۷ مرتبه در معیار میانگین هندسی، به‌طور معنادار نتیجه‌ای بهتر از نتایج روش‌های دیگر به‌دست آورده است. همچنین در هفت مورد، نتیجهٔ مساوی و تنها در شش مورد، نتیجهٔ بدتری داشته است. به‌عبارت‌دیگر الگوریتم پیشنهادی در تمامی مجموعه‌داده‌ها، در معیار میانگین هندسی، از روش‌های حرکت حد آستانه، اسموت ارتقایافته و جنگل تصادفی عملکرد بهتری دارد. در پنج مورد به‌طور معنادار نتیجه‌ای مساوی با روش‌های کاهش طبقهٔ اکثریت به‌طور تصادفی و کاهش طبقهٔ اکثریت به‌طور تصادفی و متعادل و بیز ساده به‌دست آمده است. در مجموعه داده‌های pc3 و pc4 از روش‌های آدابوست ان سی و آدابوست ان سی پویا مقدار میانگین هندسی کمتری به‌دست آورد. با توجه به نتایج به‌دست‌آمده درمی‌یابیم که روش پیشنهادشده در پیش‌بینی درست نمونه‌های طبقهٔ خطادار و بدون خطا، قابلیت زیادی دارد و دارای دقت بالا در هر دو طبقه است. چون مقدار معیار میانگین هندسی، نرخ تشخیص درست را بازتاب می‌کند.

معیار سطح زیر نمودار میزان تعادل در نرخ تشخیص درست و نرخ تشخیص اشتباه را نشان می‌دهد. از مجموع هشتاد مقایسهٔ انجام‌شده بر روی ده مجموعه داده در معیار سطح زیر نمودار راک، روش پیشنهادی در ۳۴ مرتبه نسبت به دیگر روش‌ها، به‌طور معناداری به نتایج بهتری دست پیدا کرده است. هم‌چنین با سیزده نتیجهٔ روش‌های متداول دیگر، بنابر آزمون تی، ازلحاظ مفهومی برابر است. بنابراین می‌توان گفت درمجموع از هشتاد مقایسهٔ انجام‌شده، در ۴۷ مورد نتیجه‌ای بهتر یا مساوی با دیگر روش‌های متداول تشخیص خطا به‌دست آورده است. تنها در ۳۳ مورد، عملکرد خوبی نداشته است. در معیار سطح زیر نمودار راک، روش پیشنهادی از روش‌های کاهش طبقهٔ اکثریت به‌طور تصادفی، کاهش طبقهٔ اکثریت به‌طور تصادفی و متعادل و حرکت حد آستانه، در نه مجموعه داده بهتر عمل کرده و نسبت به بقیه روش‌ها عملکرد خوبی نداشته است.

معیار تعادل، فاصلهٔ اقلیدسی بین ایده‌آل‌ترین نقطه را (آنجا که نرخ تشخیص خطا یک و نرخ عدم‌تشخیص خطا صفر است) در نمودار راک و موقعیت واقعی نشان می‌دهد. مقدار این معیار بین صفر و یک متغیر است. اگر مقدار آن برابر با یک باشد، به معنای این است که تمام خطاها بدون اشتباه تشخیص داده شده‌اند. از مشاهدهٔ جداول ۳، ۴، ۵ درمی‌یابیم که در شش مجموعه‌داده از ده مجموعه‌داده، روش پیشنهادی نسبت به هشت روش متداول دیگر، به نتایج بهتری در معیار تعادل دست پیداکرده است. روش آدابوست ان سی پویا در چهار مورد و روش آدابوست ان سی در دو مورد عملکردهای بهتری نسبت به روش پیشنهادشده داشتند. بقیه روش‌ها در هر ده مجموعه‌داده به‌طور معنادار، نسبت به روش پیشنهادی عملکرد ضعیف‌تری داشتند. بنابراین روش پیشنهادشده در معیار تعادل تعداد ۶۸ برد، ۴ مساوی و هشت باخت از مجموع هشتاد مقایسه انجام‌شده به‌دست آورد که این نتایج نشان‌دهندهٔ پیش‌بینی درست خطاها با دقت به‌نسبه خوب، توسط روش پیشنهادی است. به‌طورکلی از مشاهدهٔ جداول درمی‌یابیم که برای داده‌هایی با نرخ عدم توازن بالا، روش پیشنهادی انتخاب خوبی است. به‌طورکلی روش‌های آدابوست ان سی، آدابوست ان سی پویا و روش پیشنهادی کارایی بهتری نسبت به روش‌های کاهش کلاس اکثریت به‌طور تصادفی، کاهش طبقهٔ اکثریت به‌طور تصادفی و متعادل، حرکت حدّ آستانه، اسموت ارتقایافته، بیز ساده و جنگل تصادفی دارند.

شکل (۳) نمودار میله‌ای میانگین مقادیر معیارهای کارایی میانگین هندسی، سطح زیرنمودار و تعادل همراه با مقادیر انحراف‌معیار را بر روی ده مجموعه‌دادهٔ خطای نرم‌افزار نشان می‌دهد. این مقادیر توسط هشت روش متداول در زمینهٔ یادگیری مسائل نامتوازن و یک روش پیشنهادشده به‌دست آمده است که محور افقی نمودار را تشکیل می‌دهند. محور عمودی این نمودار نشان‌گر میانگین مقادیر سه معیار کارایی بر روی ده مجموعه‌داده است. همان‌طور که در نمودار میله‌ای مشاهده می‌شود، ارتفاع مقادیر معیارهای کارایی میانگین هندسی و تعادل در روش پیشنهادی از تمام روش‌های متداول دیگر بالاتر است. این نتیجه به معنای آن است که روش پیشنهادی نسبت به روش‌های دیگر، در پیش‌بینی نمونه‌های هر دو طبقه از دقت بالاتری برخوردار است. چون معیار میانگین هندسی نشان‌دهندهٔ تشخیص درست طبقه‌های مثبت و منفی و معیار تعادل، بیان‌گر کارایی پیش‌بینی‌کننده در ایجاد تعادل میان نرخ تشخیص درست و نرخ عدم‌تشخیص درست است. ارتفاع معیار سطح زیرنمودار در روش جنگل تصادفی از همهٔ روش‌ها بیشتر است، امّا میانگین مقادیر معیارهای تعادل و میانگین هندسی آن از همهٔ روش‌ها کمتر است. بنابراین، این روش دقت کافی برای پیش‌بینی طبقه‌های مثبت و منفی ندارد و روش مناسبی برای حل مسئلهٔ تشخیص خطای نرم‌افزار نیست. به‌طورکلی از مشاهدهٔ نمودار میله‌ای نتیجه می‌گیریم که روش پیشنهادی و روش‌های آدابوست ان سی و آدابوست ان سی پویا، کارایی بهتری نسبت به روش‌های کاهش طبقهٔ اکثریت به‌طور تصادفی، کاهش طبقهٔ اکثریت به‌طور تصادفی و متعادل، حرکت حد آستانه، اسموت ارتقایافته، بیز ساده و جنگل تصادفی دارند و برای مسئلهٔ خطای نرم‌افزار مناسب هستند.

[DOI: 10.29252/jsdp.15.4.3]

(جدول-۳): مقایسه میانگین و انحراف‌معیار روش پیشنهادی با سایر روش‌ها برای معیار میانگین هندسی
(Table-3): comparing the mean and standard deviation of proposed method with the others for G-mean measure

G-mean	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC	Proposed method
mc2	0.571±0.150	0.578±0.140	0.566±0.173	0.627±0.169	0.597±0.135	0.6070.137	0.5350.191	0.571±0.152	**0.682±0.120**
kc2	0.720±0.081	0.720±0.080	0.675±0.120	0.642±0.108	**0.762±0.083**	0.734±0.070	0.6470.105	0.777±0.701	0.759±0.081
jm1	0.649±0.022	0.645±0.018	0.647±0.019	0.541±0.025	0.679±0.016	0.594±0.020	0.494±0.025	0.672±0.017	**0.692±0.073**
kc1	0.670±0.071	0.686±0.050	0.643±0.064	0.582±0.072	**0.723±0.046**	0.694±0.031	0.562±0.072	0.734±0.040	0.7190.043
pc4	0.799±0.076	0.816±0.053	0.747±0.097	0.723±0.077	**0.865±0.048**	0.780±0.036	0.638±0.089	0.734±0.040	0.8050.037
pc3	0.669±0.095	0.690±0.057	0.588±0.110	0.504±0.099	**0.757±0.073**	0.680±0.039	0.423±0.122	0.859±0.047	0.689±0.072
cm1	0.485±0.254	0.584±0.143	0.452±0.250	0.301±0.257	0.597±0.216	0.681±0.084	0.152±0.225	0.745±0.061	**0.730±0.097**
kc3	0.563±0.247	0.682±0.123	0.430±0.268	0.433±0.262	0.665±0.175	0.724±0.082	0.225±0.265	0.659±0.133	**0.743±0.155**
mw1	0.540±0.253	0.631±0.160	0.336±0.315	0.388±0.304	0525±0.306	0.637±0.191	0.334±0.313	0.647±0.272	**0.729±0.154**
pc1	0.645±0.157	**0.711±0.089**	0.601±0.195	0.553±0.191	0.691±0.130	0.576±0.050	0.505±0.167	0.698±0.145	**0.730±0.102**

[DOI: 10.29252/jsdp.15.4.3]

(جدول-۴): مقایسهٔ میانگین و انحراف‌معیار روش پیشنهادی با سایر روش‌ها برای معیار تعادل.

(Table-4): comparing the mean and standard deviation of proposed method with the others for Balance

balance	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC	Proposed method
Mc2	0.571±0.125	0.574±0.122	0570±0.135	0.620±0.146	0.588±0.128	0.595±0.128	0.538±0.130	0.569±0.134	**0.665±0.113**
Kc2	0.705±0.086	0.709±0.078	0.660±0.117	0.68±0.0101	0.753±0.084	0.719±0.062	0.617±0.101	**0.777±0.071**	0.720±0.097
Jm1	0.646±0.023	0.642±0.0019	0.643±0.02020	0.518±0.021	0.678±0.017	0.584±0.020	0.474±0.019	0.672±0.027	**0.688±0.513**
Kc1	0.659±0.073	0.677±0.052	0.627±0.067	0.551±0.063	0.718±0.050	0.663±0.026	0.529±0.060	**0.733±0.042**	0.716±0.042
Pc4	0.784±0.086	0.808±0.050	0.721±0.104	0.682±0.080	**0.854±0.049**	0.753±0.031	0.593±0.083	**0.850±0.043**	0.789±0.032
Pc3	0.659±0.093	0.683±0.052	0.571±0.100	0.478±0.071	**0.749±0.078**	0.651±0.031	0.432±0.064	**0.739±0.064**	0.663±0.083
Cm1	0.526±0.146	0.577±0.112	0.502±0.145	0.407±0.122	0.607±0.118	0.663±0.284	0.345±0.225	0.653±0.117	**0.719±0.091**
Kc3	0.581±0.164	0.669±0.114	0.483±0.151	0.472±0.132	0.655±0.144	0.693±0.129	0.389±0.110	0.655±0.143	**0.733±0.123**
Mw1	0.566±0.161	0.619±0.126	0.450±0.165	0.470±0.159	0567±0.193	0.636±0.137	0.444±0.157	0.623±0.177	**0.717±0.136**
Pc1	0.636±0.085	0.688±0.078	0.596±0.149	0.541±0.128	0.665±0.126	0.553±0.037	0.496±0.108	0.682±0.133	**0.722±0.099**
pc1	0.645±0.157	**0.711±0.089**	0.601±0.195	0.553±0.191	0.691±0.130	0.576±0.050	0.505±0.167	0.698±0.145	**0.730±0.102**

[DOI: 10.29252/jsdp.15.4.3]

(جدول-۵): مقایسهٔ میانگین و انحراف‌معیار روش پیشنهادی با سایر روش‌ها
برای معیار سطح زیرنمودار.

(Table-5): comparing the mean and standard deviation of proposed method with the others for AUC

AUC	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC	Proposed method
mc2	0.615±0.133	0.623±0.135	0.639±0.116	**0.750±0.128**	0.690±0.130	0.700±0.150	0.722±0.118	0.649±0.142	0.702±0.107
kc2	0.730±0.087	0.726±0.092	0.687±0.092	0.742±0.092	0.803±0.085	**0.820±0.071**	**0.823±0.077**	**0.828±0.074**	0.768±0.072
jm1	0.665±0.023	0.658±0.020	0.661±0.024	0.691±0.017	0.733±0.017	0.679±0.019	0.748±0.019	**0.766±0.016**	0.701±0.027
kc1	0.710±0.063	0.713±0.059	0.670±0.058	0.755±0.046	0.802±0.033	0.758±0.038	0.802±0.033	**0.818±0.034**	0.721±0.043
pc4	0.823±0.069	0.827±0.063	0.791±0.068	0.923±0.023	**0.938±0.025**	0.863±0.035	**0.937±0.019**	0.917±0.031	0.810±0.038
pc3	0.689±0.087	0.694±0.073	0.664±0.071	0.813±0.052	**0.836±0.050**	0.777±0.054	**0.848±0.045**	0.816±0.056	0.717±0.057
cm1	0.622±0.126	0.622±0.129	0.595±0.139	0.704±0.112	**0.785±0.080**	0.748±0.091	0.742±0.105	**0.787±0.097**	0.738±0.090
kc3	0.643±0.183	0.686±0.157	0.590±0.151	0.740±0.133	**0.806±0.104**	**0.8150.082**	0.827±0.105	**0.797±0.102**	0.761±0.116
mw1	0.647±0.145	0.681±0.131	0.584±0.142	**0.758±0.145**	**0.777±0.137**	**0.780±0.142**	**0.756±0.138**	0.714±0.139	**0.749±0.125**
Pc1	0.726±0.129	0.739±0.098	0.735±0.116	0.851±0.80	**0.871±0.057**	0.700±0.096	0.847±0.067	**0.866±0.081**	0.740±0.089

[DOI: 10.29252/jsdp.15.4.3]

(جدول-۶): نتایج مقایسه بر اساس t-test به‌صورت برد، مساوی یا باخت برای معیار میانگین هندسی.

(Table-6): comparing results of T-test as Win, Tie or Lost for G-mean

G-mean	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC
mc2	W	W	W	W	W	W	W	W
kc2	W	W	W	W	T	W	W	L
jm1	W	W	W	W	W	W	W	W
kc1	W	W	W	W	T	W	W	L
pc4	T	T	W	W	L	W	W	L
pc3	W	T	W	W	L	T	W	L
cm1	W	W	W	W	W	W	W	W
kc3	W	W	W	W	W	W	W	W
mw1	W	W	W	W	W	W	W	W
pc1	W	T	W	W	W	W	W	W
Total Wins = 67 Total losses = 6 Total Ties = 7								

(جدول-۷): نتایج مقایسه بر اساس آزمون تی به‌صورت برد، مساوی یا باخت برای معیار تعادل.

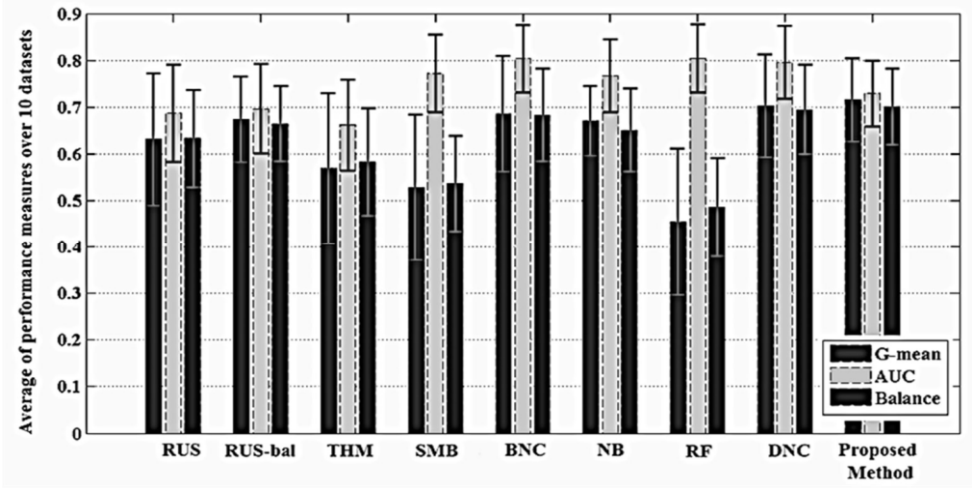
(Table-7): comparing results of T-test as Win, Tie or Lost for Balance

Balance	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC
mc2	W	W	W	W	W	W	W	W
kc2	W	W	W	W	L	T	W	L
jm1	W	W	W	W	W	W	W	W
kc1	W	W	W	W	T	W	W	L
pc4	T	L	W	W	L	W	W	L
pc3	W	T	W	W	L	W	W	L
cm1	W	W	W	W	W	W	W	W
kc3	W	W	W	W	W	W	W	W
mw1	W	W	W	W	W	W	W	W
pc1	W	W	W	W	W	W	W	W
Total Wins = 68 Total losses =8 Total Ties = 4								

(جدول-۸): نتایج مقایسه بر اساس آزمون تی به‌صورت برد، مساوی یا باخت برای معیار سطح زیرنمودار.

(Table-8): comparing results of T-test as Win, Tie or Lost for AUC

AUC	RUS	RUS-bal	THM	SMB	BNC	NB	RF	DNC
mc2	W	W	W	L	W	T	T	W
kc2	W	W	W	W	T	L	L	L
jm1	W	W	W	W	L	W	L	L
kc1	W	W	W	L	L	L	L	L
pc4	T	L	W	L	L	L	L	L
pc3	W	W	W	L	L	L	L	L
cm1	W	W	W	W	L	T	T	L
kc3	W	W	W	T	L	L	L	L
mw1	W	W	W	T	T	T	T	W
pc1	W	T	T	L	L	W	L	L
Total Wins = 34 Total losses =33 Total Ties = 13								



(شکل-۳): مقایسه کارایی روش پیشنهادی با بقیه روشها با نمودار میله‌ای: میانگین مقادیر معیارهای کارایی میانگین هندسی، سطح زیر نمودار و تعادل، همراه با مقادیر انحراف معیار آنها بر روی ده مجموعه دادهٔ خطای نرم‌افزار.

(Figure-3): performance comparison of our proposed method with the others by bar chart: the mean of G-mean, AUC and balance values as well as their standard deviation for ten datasets.

[DOI: 10.29252/jsdp.15.4.3]

۵- نتیجه‌گیری

تشخیص ماژول‌های مستعد خطا در ابتدای پروژه‌های نرم‌افزاری، باعث صرفه‌جویی در تلاش و هزینه‌های گروه توسعه می‌شود. این ماژول‌ها درصد کمی از سامانهٔ نرم‌افزاری را شامل می‌شوند؛ بنابراین مسئلهٔ تشخیص خطا جزو مسائل نامتوازن است و تشخیص ماژول‌های مستعد خطا نسبت به ماژول‌های بدون خطا اهمیت بیشتری دارد.

در این مقاله با استفاده از خوشه‌بندی کاهشی و خوشه‌بندی فازی c میانگین فازی یک مدل پیش‌بینی خطای نرم‌افزار تولید و بدین منظور در ابتدا از تمام ویژگی‌های نمونه‌ها در مجموعه داده‌ها استفاده کردیم؛ امّا با توجه به این‌که ممکن است در هر مجموعه‌داده بعضی از ویژگی‌ها از بقیه مهم‌تر باشند، برای بهبود نتایج به‌دست آمده از الگوریتم ژنتیک برای انتخاب ویژگی استفاده کردیم. برای کنترل و حل نامتوازن‌بودن طبقه‌ها، نمونه‌های طبقهٔ اقلیت را وزن‌دار کردیم تا طبقه‌بند به طبقهٔ اقلیت توجه بیشتری کند.

نتایج به‌دست‌آمده از روش پیشنهادی با هشت الگوریتم متداول مقایسه و ارزیابی شده‌اند. این الگوریتم‌ها شامل سه نوع روش یادگیری جمعی، نمونه‌گیری مجدد و حرکت حدّ آستانه می‌شوند. این ارزیابی بر روی ده مجموعه‌دادهٔ خطای نرم‌افزار که گسترهٔ وسیعی از اندازه و نرخ عدم توازن را در بردارند، انجام شد. برای نشان‌دادن این‌که نتایج به‌دست‌آمده درعمل مفید هستند، از سه معیار ارزیابی کارایی استفاده کردیم. این معیارها شامل میانگین هندسی، تعادل و معیار سطح زیرنمودار هستند. نتایج مقایسه‌ها نشان دادند که الگوریتم پیشنهادی بهترین نتایج را در معیارهای میانگین هندسی و تعادل به‌دست آورد. الگوریتم جنگل تصادفی مقدار معیار سطح زیر نمودار بالا و مقدار معیارهای تعادل و میانگین هندسی بسیار کمی به‌دست آورد. بنابراین این الگوریتم برای مسئلهٔ تشخیص خطای نرم‌افزار مناسب نیست. بالابودن مقدار سطح زیرنمودار در الگوریتم آدابوست ان سی پویا نسبت به روش پیشنهادی و کمتربودن مقادیر میانگین هندسی و تعادل نشان‌دهندهٔ این است که مقادیر نرخ تشخیص درست هر دو طبقه، نسبت به روش پیشنهادی اختلاف بیشتری دارند؛ درحالی‌که در روش پیشنهادی ما، این مقادیر هر دو بالا و به هم نزدیک هستند. علاوه‌بر معیارهای دقّت، از دیگر مزایای الگوریتم پیشنهادی می‌توان به مواردی چون وابسته‌نبودن زیاد به پارامترها، زمان محاسباتی به‌نسبه کم، پیچیده‌نبودن طبقه‌بندهای به‌دست‌آمده و ساخت طبقه‌بند قدرتمند با استفاده از منطق فازی و الگوریتم‌های تکاملی اشاره کرد. روش پیشنهادی باوجودآن‌که در معیارهای میانگین هندسی و تعادل نتایج بسیار خوبی به‌دست آورده است، امّا در معیار سطح زیرنمودار نتایج چندان رضایت‌بخش نیست. علّت این است که روش پیشنهادی روی داده‌های با نرخ عدم توازن بالا، خوب عمل می‌کند و روی داده‌های با نرخ عدم توازن پایین، به‌خوبی روش آدابوست ان سی پویا نیست.

۶-مراجع 6- References

[1] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Trans. Reliab.*, vol. 62, no. 2, pp. 434–443, 2013.

[2] B.-J. Park, S.-K. Oh, and W. Pedrycz, "The design of polynomial function-based neural network predictors for detection of software defects," *Inf. Sci. (Ny).*, vol. 229, pp. 40–57, 2013.

[3] P. C. Pendharkar, "Exhaustive and heuristic search approaches for learning a software defect prediction model," *Eng. Appl. Artif. Intell.*, vol. 23, no. 1, pp. 34–40, 2010.

[4] J. Zheng, "Cost-sensitive boosting neural networks for software defect prediction," *Expert Syst. Appl.*, vol. 37, no. 6, pp. 4537–4543, 2010.

[۵] مهدی زاده محبوبه، افتخاری مهدی. ارائهٔ روش جدید مبتنی‌بر برنامه‌نویسی ژنتیک برای وزن‌دهی قوانین فازی در طبقه‌بندی نامتوازن. پردازش علائم و داده‌ها. ۱۳۹۳; ۱۱ (۲): ۱۲۵-۱۱۱

[5] Mahdizadeh Mahboubeh, Eftekhari Mahdi. "A new fuzzy rules weighting approach based on Genetic Programming for imbalanced classification" . *JSDP.*, no. 11 (2), pp.111-125, 2015

[6] Z. Yan, X. Chen, and P. Guo, "Software defect prediction using fuzzy support vector regression," *Adv. Neural Networks-ISNN 2010*, pp. 17–24, 2010.

[7] A. K. Pandey and N. K. Goyal, "A fuzzy model for early software fault prediction using process maturity and software metrics," *Int. J. Electron. Eng.*, vol. 1, no. 2, pp. 239–245, 2009.

[8] S. Di Martino, F. Ferrucci, C. Gravino, and F. Sarro, "A genetic algorithm to configure support vector machines for predicting fault-prone components," in *International Conference on Product Focused Software Process Improvement*, 2011, pp. 247–261.

[9] P. S. Sandhu, S. Khullar, S. Singh, S. K. Bains, M. Kaur, and G. Singh, "A Study on Early Prediction of Fault Proneness in Software Modules using Genetic Algorithm," *World Acad. Sci. Eng. Technol.*, vol. 72, 2010.

[DOI: 10.29252/jsdp.15.4.3]

[10] M. M. Rosli, N. H. I. Teo, N. S. M. Yusop, and N. S. Mohammad, “The design of a software fault prone application using evolutionary algorithm,” in *Open Systems (ICOS), 2011 IEEE Conference on*, 2011, pp. 338–343.

[11] M.-Y. Chen, “A hybrid ANFIS model for business failure prediction utilizing particle swarm optimization and subtractive clustering,” *Inf. Sci. (Ny).*, vol. 220, pp. 180–195, 2013.

[12] M. E. R. Bezerra, A. L. I. Oliveira, and S. R. L. Meira, “A constructive rbf neural network for estimating the probability of defects in software modules,” in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, 2007, pp. 2869–2874.

[13] M. E. R. Bezerra, A. L. I. Oliveiray, and P. J. L. Adeodatoz, “Predicting software defects: A cost-sensitive approach,” in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, 2011, pp. 2515–2522.

[14] H. A. Al-Jamimi and L. Ghouti, “Efficient prediction of software fault proneness modules using support vector machines and probabilistic neural networks,” in *Software Engineering (MySEC), 2011 5th Malaysian Conference in*, 2011, pp. 251–256.

[15] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, “A possibilistic fuzzy c-means clustering algorithm,” *IEEE Trans. fuzzy Syst.*, vol. 13, no. 4, pp. 517–530, 2005.

[16] D. Gray, D. Bowes, N. Davey, Y. Sun, and B. Christianson, “Using the Support Vector Machine as a Classification Method for Software Defect Prediction with Static Code Metrics.,” in *EANN*, 2009, vol. 2009, pp. 223–234.

[17] K. O. Elish and M. O. Elish, “Predicting defect-prone software modules using support vector machines,” *J. Syst. Softw.*, vol. 81, no. 5, pp. 649–660, 2008.

[18] S. Wang, A. Mathew, Y. Chen, L. Xi, L. Ma, and J. Lee, “Empirical analysis of support vector machine ensemble classifiers,” *Expert Syst. Appl.*, vol. 36, no. 3, pp. 6466–6476, 2009.

[19] I. Gondra, “Applying machine learning to software fault-proneness prediction,” *J. Syst. Softw.*, vol. 81, no. 2, pp. 186–195, 2008.

[20] S. R. Kannan, S. Ramathilagam, and P. C. Chung, “Effective fuzzy c-means clustering algorithms for data clustering problems,” *Expert Syst. Appl.*, vol. 39, no. 7, pp. 6292–6300, 2012.

[21] O. T. Yıldız, O. Aslan, and E. Alpaydın, “Multivariate statistical tests for comparing classification algorithms,” *Lect Notes Comp Sci*, vol. 6683, pp. 1–15, 2011.

مهدی افتخاری مدرک کارشناسی خود را در سال ۱۳۷۹ در رشتهٔ مهندسی کامپیوتر گرایش سخت‌افزار از دانشگاه شیراز و مدارک کارشناسی ارشد و دکترای خود را به‌ترتیب در سال‌های ۱۳۸۲ و ۱۳۸۶ در رشتهٔ مهندسی کامپیوتر گرایش هوش مصنوعی از همان دانشگاه اخذ کرد. وی از سال ۱۳۸۶ تا کنون عضو هیأت علمی بخش مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه شهید باهنر کرمان است و در سال ۱۳۹۳ به مرتبه دانشیاری ارتقا پیدا کرده است. حوزه‌های تخصصی ایشان مجموعه‌ها و مدل‌های فازی، الگوریتم های تکاملی و یادگیری ماشین است. وی تاکنون بیش از ۱۲۰ مقالهٔ علمی در نشریات و کنفرانس‌های معتبر داخلی و خارجی به چاپ رسانیده است.

نشانی رایانامهٔ ایشان عبارت است از:

m.eftekhari@uk.ac.ir

مریم مجیدی مؤمن‌آبادی مدرک کارشناسی خود را در سال ۱۳۸۹ در رشته مهندسی کامپیوتر گرایش نرم‌افزاراز دانشگاه شهید باهنر کرمان و مدرک کارشناسی ارشد خود را در سال ۱۳۹۳ در رشتهٔ مهندسی فناوری اطلاعات گرایش طراحی و تولید نرم‌افزار دانشگاه آزاد اسلامی واحد کرمان اخذ کرد. حوزه‌های تخصّصی ایشان مجموعه‌ها و مدل‌های فازی، طراحی نرم‌افزارو توسعهٔ نرم‌افزاربرای تلفن‌های همراه است.

نشانی رایانامهٔ ایشان عبارت است از:

Maryam.majidi93@gmail.com

مجتبی خمر مدرک کارشناسی خود را در سال ۱۳۹۳ در رشتهٔ مهندسی کامپیوتر گرایش سخت‌افزار از دانشگاه سیستان و بلوچستان و مدرک کارشناسی ارشد خود را در سال ۱۳۹۶ در رشتهٔ مهندسی کامپیوتر گرایش هوش مصنوعی از دانشگاه شهید باهنر کرمان اخذ کرد. حوزه‌های تخصّصی ایشان مجموعه‌ها و مدل‌های فازی، و یادگیری ماشین است.

نشانی رایانامهٔ ایشان عبارت است از:

Mojtabakhammar69@gmail.com

[DOI: 10.29252/jsdp.15.4.3]