تأثیر اندازه‌های طراحی نسبت به اندازه‌های کد در بهبود کارایی سامانه‌های آزمون خودکار

زهرا اسلامی مشکنانی[۱]، دانشجوی کارشناسی ارشد، اشکان سامی[۲]، استادیار

۱- دانشکده فنی و مهندسی- دانشگاه پیام نور تهران شمیرانات- تهران- ایران Zhr.eslami@gmail.com

۲- دانشکده مهندسی برق و کامپیوتر- دانشگاه شیراز- شیراز- ایران asami@ieee.org

چکیده: پیشگویی خودکار خطا بر اساس اندازه‌های در دسترس، یک موضوع مورد توجه محققان برای افزایش کیفیت نرم افزار است. علی‌رغم استفاده زیاد از اندازه‌های کد در مطالعات گذشته، تنها سه مطالعه است که تأثیر اندازه‌های کد و اندازه‌های طراحی را در پیشگویی خطای نرم‌افزار مقایسه می‌کند و نتایج متناقضی هم دارند. بنابراین برای فهم واقع‌بینانه‌تر و قابل اطمینان تر از تأثیر اندازه‌های طراحی نسبت به اندازه‌های کد، آزمایشات متنوع‌تری روی طیف وسیع‌تری از مجموعه‌های داده با به‌کارگیری تعداد بیشتری از دسته‌بندی کننده‌ها انجام شده است. مدل‌های پیشگویی خطا، مبتنی بر اندازه کد و اندازه طراحی و ترکیب آن دو ساخته شده است. معیارهای ارزیابی کارایی مدل، AUC و معیار F هستند. بر اساس آزمون‌های آماری مختلف، در کارایی مدل‌هایی که از ترکیب اندازه‌های کد و طراحی استفاده می‌کنند نسبت به آن‌هایی که از اندازه‌های کد استفاده می‌کنند. پیشرفت قابل توجهی وجود ندارد. همچنین اندازه‌های طراحی به تنهایی، نسبت به اندازه‌های کد به تنهایی و یا ترکیب اندازه‌های کد و طراحی، تأثیر کمتری روی شناسایی پیمانه‌های مستعد خطا دارد.

واژه‌های کلیدی : اندازه طراحی، اندازه کد، پیشگویی خطا، داده کاوی

Effect of design metrics on improving accuracy of automatic test systems based on software code

Z. Eslami moshknani[1], A. Sami[2]

1- Faculty of Engineering, University of Payam Noor Teharan, Branch of Shemiranat

2- Faculty of Electrical and Computer Engineering, Shiraz University

Abstract: **Automatic defect prediction based on available metrics is an interesting research topic to increase software quality. In spite of high usage of code metrics in previous researches, there are only three researches that have been conducted on comparing the effect of code metrics and design metrics in software defect prediction which have contradictory results. Thus, to have a better and more reliable understanding of the effect of design metrics over code metrics the investigation has been extended to an extensively larger dataset with many more classifiers. Defect prediction models have been built based on code metrics, design metrics and both. Due to imbalance datasets accuracy was not used. In contrast, AUC and F-measure are the performance evaluation measures of classifiers. Based on different statistical tests, no significant improvement in performance of the models based on combination of design and code metrics compared to the models which have been implemented with code metrics exist. Generally, design metrics alone have less discriminating effect on detecting fault-prone modules in comparison with code metrics alone or combination of code and design metrics.**

Keywords: Design metric, Code metric, Defect prediction, Data mining

تاریخ ارسال مقاله: ۹۱/۲/۱۳

تاریخ اصلاح مقاله: ۹۱/۹/۲۰

تاریخ پذیرش مقاله: ۹۲/۱/۲۸

نام نویسنده مسئول: زهرا اسلامی مشکنانی

نشانی نویسنده مسئول: ایران- تهران- دانشگاه پیام نور تهران مرکز شمیرانات

۱- مقدمه

خطاها و نقص‌ها در نرم‌افزار، هزینه بر هستند [1, 2] چرا که، شناسایی و تصحیح آن‌ها اغلب ۳۰٪ تا ۵۰٪ از بودجه پروژه را به خود اختصاص می‌دهند [3]. این هزینه نمی‌تواند به طور کامل حذف شود اما اگر خطای نرم‌افزار به طور خودکار، پیشگویی شود بعداً هزینه‌ای صرف شناسایی و تصحیح خطا نخواهد شد و همچنین کیفیت نرم‌افزار به دلیل نداشتن خطا، بالا می‌رود و باعث رضایتمندی بیشتر مشتری هم می‌شود. همچنین در مقایسه با نرخ ۶۰٪ شناسایی اتوماتیک خطا نسبت به مرور دستی نرم‌افزار (بر اساس یک پانل اندازه‌ها[1] در ۲۰۰۲ IEEE) [4] و در نظر گرفتن این که یک مرورگر خبره فقط می‌تواند ۸ تا ۲۰ خط از کاراکتر را در یک دقیقه[2] بازرسی کند واضح است که چرا پیشگویی‌های خودکار خطا[3] بر اساس اندازه‌های موجود در پروژه، این چنین مورد توجه تولید کنندگان نرم‌افزار است. از این رو مهندسین نرم‌افزار به ساخت سامانه‌هایی رو آوردند که بر اساس روش‌های داده کاوی[4]، فایل‌های مشکوک به داشتن خطا را شناسایی می‌کنند و در نتیجه باعث بالا رفتن کیفیت نرم‌افزار، رضایت مشتری و کاهش هزینه می‌شود. در داده کاوی، فایل‌ها به صورت یک رکورد که اندازه‌های آن استخراج شده‌اند بیان شده است.

اولین کار داده کاوی توسط سلبی و پورتر [5] بر اساس اندازه‌های کد[5] بوده است و پس از آن نیز به اندازه‌های کد توجه زیادی شده است ولی از اندازه‌های طراحی[6] استفاده چندانی نشده است. در برنامه‌های نرم‌افزار، اندازه‌های طراحی دقت اندازه‌های کد را ندارند چرا که مثل اندازه‌های کد ممکن است به روز نباشند. تنها در سه مطالعه کارایی[7] اندازه‌های کد و اندازه‌های طراحی و ترکیب اندازه‌های کدو طراحی در پیشگویی خطای نرم‌افزار مقایسه شده و نتایج به دست آمده از آن‌ها هم متناقض است [6, 7, 8]. در نتایج اولین مطالعه مشاهده شده که مدل پیشگویی که از اندازه‌های طراحی استفاده می‌کند به خوبی مدلی است که از اندازه‌های کد استفاده می‌کند و پیشرفت مهمی وجود ندارد موقعی که اندازه‌های طراحی و اندازه‌های کد ترکیب می‌شوند و با هم به عنوان پیشگوها استفاده می‌شوند [6]. در نتایج دومین مطالعه دیده می‌شود از بین سه مدل پیشگویی که از اندازه‌های کد یا اندازه‌های طراحی و یا ترکیب اندازه‌های کد و طراحی استفاده می‌کنند مدل پیشگویی که از ترکیب اندازه‌های کد و طراحی ساخته می‌شود بالاترین کارایی و مدل پیشگویی که از اندازه‌های طراحی ساخته می‌شوند پایین‌ترین کارایی را دارند [7]. در سومین مطالعه اظهار شده است مدل پیشگویی که از ترکیب اندازه‌های طراحی UML و اندازه‌های کد استفاده می‌کند بهترین کارایی را دارد و استفاده از اندازه‌های UML به عنوان پیشگو نسبت به اندازه‌های کد، عملکرد بهتری دارد [8]. از آنجائیکه در همه‌ی مطالعات فوق هم تعداد پروژه‌ها در آزمایشات کم بوده و هم از تعداد محدودی از دسته‌بندی کننده‌ها استفاده شده، همچنین کارایی مدل‌های ساخته شده بر اساس فقط یکی دو معیار آماری بررسی شده است، در این مقاله ارزیابی چنین مدل‌هایی روی طیف وسیع‌تری از مجموعه‌های داده، انجام شده ضمن این که از دسته‌بندی کننده‌های خیلی بیشتر و معیار های آماری متفاوتی برای بررسی کارایی این مدل‌ها استفاده می‌شود تا به نتایج واقع‌بینانه‌تری برای رفع تناقض در مطالعات مشابه قبلی دست یافت. هر پیمانه[8] به وسیله یک مجموعه از ویژگی‌های سطح کد و سطح طراحی توصیف می‌شود. این مقاله روی ۱۷ مجموعه داده (۱۲ مجموعه داده NASA [9, 10] و همچنین ۵ مجموعه داده AR [10, 11]) که در آن اندازه‌های طراحی و کد ایستا در دسترس هستند، برای ساخت مدل‌های پیشگویی خطا، پیاده شده است که در آن پس از توصیف مجموعه‌های داده، کارایی مدل‌ها روی ۱- فقط اندازه‌های کد ایستا ۲- فقط اندازه‌های طراحی ۳- ترکیبی از اندازه‌های کد ایستا و اندازه‌های سطح طراحی، مقایسه شده است.

موارد یافت شده شامل: ۱) مدل‌های ساخته شده با استفاده از اندازه‌های طراحی و اندازه‌های کد ایستا و ترکیب اندازه‌های کد و طراحی مفیدند؛ ۲) با توجه به نمودارهای[9] جعبه‌ای[10] و آزمون‌های[11] آماری، کارایی اندازه‌های طراحی در پیشگویی خطای نرم‌افزار نسبت به اندازه‌های کد و ترکیب اندازه‌های کد و طراحی ضعیف‌تر است؛ ۳) مدل پیشگویی خطای نرم‌افزار مبتنی بر معیار F[12] نسبت به مدل پیشگویی خطای نرم افزار مبتنی بر AUC[13] ، کارایی بهتری را روی سه گروه از اندازه‌های کد و اندازه‌های طراحی و ترکیب اندازه‌های کد و طراحی ارائه کرده است.

ساختار مقاله به صورت زیر می‌باشد: در بخش دوم اندازه‌های نرم-افزار بررسی شده است. در بخش سوم طراحی آزمایشی برای ساخت مدل پیشگوی خطای نرم‌افزار با انتخاب روش‌های مدل‌سازی انجام می‌شود و روش‌های ارزیابی عملکرد مدل‌ها بررسی می‌شود. بخش چهارم شامل ارائه نتایج تجربی و بحث روی پیاده‌سازی مدل‌های پیشگویی خطا مبتنی بر AUC و مدل‌های مبتنی بر معیار F است و کارایی آن‌ها مقایسه شده است. بخش پنجم، کارهای مرتبط پیشین مطرح می‌گردد. بخش ششم به جمع‌بندی و نتیجه‌گیری در مورد مطالب مطرح شده و کار آینده اختصاص یافته است.

۲- توصیف اندازه‌ها

مجموعه‌های داده استفاده شده در این مطالعه شامل دو دسته NASA و AR هستند و البته اندازه‌هایی که در این مقاله استفاده می‌شود در هر دو مشترک می‌باشد. دوازده مجموعه داده استفاده شده در این مطالعه که از انباره داده مجموعه فضایی آمریکا NASA است در جدول ۱ نشان داده شده‌اند و هر یک با اندازه‌های مختلفی توصیف می‌شوند. این مجموعه‌های داده هم با پسوند ARFF در گوگل قابل جستجو می‌باشد و هم از انباره داده پرامیس [10, 11] قابل دسترس هستند. مثلاً پروژه‌های CM1, KC1, PC1 و KC2 هر کدام ۲۲ اندازه کلی دارند و پروژه‌های KC3 و MC2 هر کدام ۴۰ مشخصه دارند که می‌تواند به عنوان متغیرهای پیشگو استفاده شوند. پنج مجموعه داده دیگر استفاده شده در این مقاله AR است که در دسترس می‌باشد و به منظور آزمایشات بیشتر و اطمینان از نتایج مطالعه، از انباره داده پرامیس، استخراج شده است. این پنج پروژه که در جدول ۲ نشان داده شده است هر کدام شامل ۳۰ مشخصه کلی هستند.

جدول (۱) : مجموعه‌های داده NASA

مجموعه‌های داده	مثال‌ها	درصد دارای خطا	اندازه‌ها		
			همه	طراحی	کد
CM1	۵۰۵	٪ ۹/۸۳	۲۲	۵	۱۷
KC1	۲۴۰۷	٪ ۱۵/۴۶	۲۲	۵	۱۷
KC3	۴۵۸	٪ ۹/۳۹	۴۰	۱۶	۲۰
PC1	۱۱۰۷	٪ ۶/۹۴	۲۲	۵	۱۷
PC3	۱۵۶۳	٪ ۱۰/۲۳	۳۸	۱۶	۲۰
PC4	۱۴۵۸	٪ ۱۲/۲	۳۸	۱۶	۲۰
MW1	۴۳۳	٪ ۷/۶۹	۳۸	۱۶	۲۰
MC2	۱۶۱	٪ ۳۲/۳	۴۰	۱۶	۲۰
MC1	۹۴۶۶	٪ ۰/۷۲	۳۹	۱۵	۲۰
PC2	۵۵۸۹	٪ ۰/۴۱	۳۷	۱۶	۱۹
PC5	۱۷۱۸۶	٪ ۳	۳۹	۱۵	۲۰
KC2	۵۲۲	٪ ۲۰/۵	۲۲	۵	۱۷

برای یک پروژه اندازه‌های گوناگونی از قبیل اندازه‌های تحلیل، اندازه‌های طراحی، اندازه‌های کد، اندازه‌های آزمون و اندازه‌های نگهداری وجود دارد ولی در این مطالعه طبق معیار تقسیم بندی جیان و همکاران[7] سه گروه اندازه مطابق جدول ۳ برای هر پیمانه در نظر گرفته شده و شامل اندازه‌های طراحی و اندازه‌های کد و بقیه اندازه‌ها می‌باشد. بقیه اندازه‌ها، هم با طراحی و هم با کد مربوط می‌شوند. بعضی مجموعه‌های داده مثل PC5، ۴ اندازه دارند که در ستون بقیه اندازه‌ها دسته‌بندی شده است. بعضی از مجموعه‌های داده مثل CM1 هیچ مشخصه‌ای جزء ستون بقیه اندازه‌ها ندارند. به علاوه یک گروهی که همه نامیده می‌شود تعریف شده است که شامل مجموعه کاملی از اندازه‌های پیمانه است.

جدول (۲) : مجموعه‌های داده AR

مجموعه های داده	مثال‌ها	درصد دارای خطا	اندازه‌ها		
			همه	طراحی	کد
AR1	۱۲۱	٪ ۷/۴۳	۳۰	۱۰	۱۸
AR3	۶۳	٪ ۱۲/۷	۳۰	۱۰	۱۸
AR4	۱۰۷	٪ ۱۸/۶۹	۳۰	۱۰	۱۸
AR5	۳۶	٪۲۲/۲۲	۳۰	۱۰	۱۸
AR6	۱۰۱	٪ ۱۴/۸۵	۳۰	۱۰	۱۸

۳- طراحی آزمایشی

۱۷ مجموعه داده (۱۲ مجموعه داده NASA و ۵ مجموعه داده AR) فراخوانی شده‌اند که هر یک سه گروه از اندازه‌ها را دارند (همه، کد، طراحی) و متغیر پیشگویی شده، خطا است.

از ۲۰ الگوریتم دسته‌بندی[14] درWEKA [12] برای ساخت مدل پیشگویی خطا روی ۱۷ مجموعه داده استفاده شده است و کارایی مدل‌ها یک بار بر مبنای منحنی ROC[15] و منطقه زیر منحنی (AUC) و یک بار بر مبنای معیار F ارزیابی می‌شوند. از نمودارهای جعبه‌ای جهت نمایش گرافیکی کارایی مدل‌ها استفاده می‌شود. کارایی مدل‌های ساخته شده از اندازه‌ها در گروه طراحی و گروه کد و گروه اندازه‌های همه با هم مقایسه می‌شوند. از آن جائیکه ۳ گروه اندازه و ۱۷ مجموعه وجود دارد در نتیجه ۵۱=۳×۱۷ نمودار جعبه‌ای برای AUC و همین تعداد نمودار را برای معیار F خواهیم داشت.

سپس از آزمون‌های آماری غیر پارامتری دمسار [13] برای مقایسه کارایی مدل‌های به وجود آمده از سه گروه اندازه کد و طراحی و ترکیب اندازه‌های کد و طراحی در مدل‌های مبتنی بر AUC و همچنین به طور جداگانه در مدل‌های مبتنی بر معیار F استفاده می‌شود. این آزمون در دو مرحله انجام می‌گیرد، ابتدا از آزمون فریدمن[16] برای مقایسه سه گروه اندازه‌ها با هم در هر

مجموعه داده استفاده می‌شود و در صورتی که اختلاف قابل توجهی بین کارایی این سه گروه وجود داشته باشد از آزمون ویلکاکسون[۱۷] برای مقایسه دو به دو از مدل‌های به وجود آمده از سه گروه اندازه‌ها در هر مجموعه داده استفاده می‌شود. در نتیجه به سه آزمون: ۱) اندازه‌های همه و کد ۲) اندازه‌های همه و طراحی ۳) اندازه‌های کد و طراحی ویلکاکسون برای هر مجموعه داده نیاز خواهیم داشت.

ترسیم نمودارهای جعبه‌ای و انجام آزمون‌های آماری به وسیله نرم‌افزار SPSS انجام شده است[14].

در ادامه این بخش ضمن معرفی ۲۰ نمونه خوب از دسته‌بندی کننده‌ها[۱۸]، نمودارهای جعبه‌ای توصیف می‌شود و از آن برای مقایسه کارایی گروه‌های مختلف اندازه استفاده خواهد شد. سرانجام آزمون‌های آماری و فرضیه‌های استفاده شده در آن نشان داده می‌شود.

جدول (۳) : اندازه‌های استفاده شده در این مقاله

گروه	کد	طراحی	بقیه اندازه‌ها
اندازه‌ها	PARAMETER_COUNT FORMAL PARAMETER NUM_OPERATORS:N1 NUM_OPERANDS:N2 NUM_UNIQUE_OPERATORS:$\mu 1$ NUM_UNIQUE_OPERATORS: $\mu 2$ HALSTEAD_CONTENT: μ HALSTEAD_LENGTH:N HALSTEAD_LEVEL:L HALSTEAD_DIFFICULTY:D HALSTEAD_VOLUME:V HALSTEAD_EFFORT:E HALSTEAD_PROG_TIME:T HALSTEAD_ERROR_EST:B NUMBER_OF_LINES LOC_BLANK LOC_CODE_AND_COMMENT:NCSLOC LOC_COMMENTS LOC_EXECUTABLE PERCENT_COMMENTS LOC_TOTAL	EDGE_COUNT:e NODE_COUNT:n BRANCH_COUNT CALL_PAIRS CONDITION_COUNT CYCLOMATIC_COMPLEXITY:v(G) DECISION_COUNT DECISION_DENSITY DESIGN_ COMPLEXITY:iv(G) DESIGN_DENSITY ESSENTIAL_ COMPLEXITY:ev(G) ESSENTIAL_DENSITY MAINTENANCE_SEVERITY MODIFIED_CONDITION_COUNT MULTIPLE_ CONDITION_COUNT PATHOLOGICAL_ COMPLEXITY	NORMALIZED_ CYCLOMATIC_COMPLEXITY GLOBAL_DATA_COMPLEXY:gdv(G) GLOBAL_DATA_DENSITY CYCLOMATIC_DENSITY

۳-۱- دسته‌بندی کننده‌ها

مدل‌های پیشگو با استفاده از دسته‌بندی کننده‌های موجود در WEKA که در جدول ۴ نشان داده شده، ساخته می‌شوند. WEKA پیاده‌سازی الگوریتم‌های دسته‌بندی را روی مجموعه داده فراهم می‌سازد.

هر مجموعه داده دارای یک اندازه دودویی است که خطا[۱۹] نامیده می‌شود به طوری که هر گاه تعداد خطا در پیمانه، بزرگ‌تر یا مساوی ۱ باشد، این اندازه برابر با درست[۲۰] و در غیر این صورت نادرست[۲۱] می‌شود. خطا، متغیر پیشگویی شده است و به همراه دیگر اندازه‌های پروژه برای ساخت مدل‌های پیشگو در WEKA استفاده می‌شود.

نوزده الگوریتم دسته‌بندی با پارامترهای پیش فرضشان استفاده شده‌اند ولی جنگل‌های تصادفی[۲۲] با ۵۰۰ درخت رشد یافته است (پیش فرض آن در WEKA، ۱۰ درخت است که کافی نیست). تصادفاً بریمن [15] مخترع جنگل‌های تصادفی از ۵۰۰ درخت جنگل به عنوان پیش فرض استفاده کرده است.

در مورد همه دسته‌بندی‌های فوق در ویتن و فرانک [16] توضیح بیشتری موجود است.

جدول (۴) : دسته‌بندی کننده‌ها

	learners		learners
۱	BayesNet	۱۱	MultiBoostAB
۲	NaïveBayes	۱۲	ThresholdSelector
۳	Bagging	۱۳	HyperPipes
۴	Logistic	۱۴	DecisionTable
۵	SMO	۱۵	JRip
۶	RBFNetwork	۱۶	PART
۷	VotedPerceptron	۱۷	ADTree
۸	IBK	۱۸	J48
۹	KStar	۱۹	REPTree
۱۰	RandomCommittee	۲۰	RandomForest

۳-۲- ارزیابی مدل‌های پیشگو

در این مطالعه، از دو روش زیر برای ارزیابی کارایی مدل پیشگویی استفاده شده است:

۱- AUC: منحنی ROC، گیرنده مشخصه عامل است و بین دو محور TP و FP ترسیم می‌شود.

منحنی ROC، موازنه‌ای را بین حساسیت[۲۳] و ویژگی[۲۴] بین نقاط برش نشان می‌دهد یعنی افزایش حساسیت، کاهش ویژگی را به دنبال خواهد داشت. اگر در گراف، حساسیت را در مقابل (ویژگی- ۱) رسم کنیم، پیشگویی مدل با صحت کامل، ۱۰۰٪ منطقه زیر منحنی ROC را در بر خواهد گرفت.

منطقه زیر منحنی ROC که AUC نام دارد اندازه‌گیری عددی کارایی دسته‌بندی‌های مختلف است. با AUC، کارایی دسته‌بندی‌های مختلف برای سه گروه اندازه روی ۱۷ پروژه مذکور مقایسه و ارزیابی می‌شوند.

۲- معیارF: برای اجرای خوب، یک مدل باید هم به دقت[۲۵] بالا و هم به فراخوانی[۲۶] بالا برسد و یک موازنه‌ای بین این دو وجود دارد. برای مثال اگر یک مدل، همه پیمانه‌ها را به عنوان معیوب پیش‌گویی کند فراخوانی ۱ خواهد بود اما دقت پایین است بدیهی است که در این مورد ما نمی‌توان گفت مدل خوب اجرا شده است. به عبارت دیگر اگر یک مدل تنها یک پیمانه را به عنوان معیوب پیش‌گویی کند و معلوم شود که پیش‌گویی صحیح بوده است دقت مدل ۱ خواهد بود با این وجود دوباره نمی‌توان عملکرد این مدل را خوب در نظر گرفت. معیار F (رابطه ۱) به دقت و فراخوانی به وسیله در نظر گرفتن معنای هارمونیک و همساز آن‌ها، اهمیت برابر می‌دهد و شبیه دقت و فراخوانی، مقداری بین ۰ و ۱ دارد.

$$\text{معیار F} = \frac{\text{۲*دقت*فراخوانی}}{\text{دقت+فراخوانی}} \qquad (۱)$$

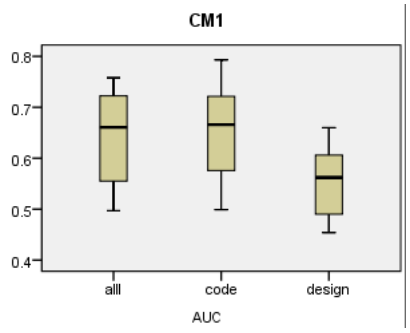
مقدار بالاتر چه برای AUC و چه برای معیار F نشان دهنده کارایی بهتر پیشگویی است. قابل ذکر است که از معیار F در هیچ یک از مطالعات مشابه قبلی برای بررسی کارایی مدل‌ها استفاده نشده است.

وارسی اعتبار[۲۷]، مهارت آماری از تقسیم یک نمونه از داده به دو زیر مجموعه زیر است : زیر مجموعه آموزش[۲۸] و آزمون. در آزمایشات ۹۰٪ از داده‌ها به طور تصادفی به زیر مجموعه آموزش اختصاص داده می‌شوند و ۱۰٪ باقی‌مانده برای آزمون استفاده می‌شوند.

۳-۳- نمودار جعبه‌ای

یک نمودار جعبه‌ای شامل یک جعبه و یک نمودار سوزنی متحرک است و به طور گرافیکی، توزیع داده‌های عددی را به صورت پنج ترتیب آماری شامل کوچک‌ترین مشاهده (حداقل نمونه)، چارک پایین‌تر (Q1)، میانه (Q2)، چارک بالاتر (Q3) و بزرگ‌ترین مشاهده (حداکثر نمونه)، به تصویر می‌کشد.

شکل ۱، یک نمودار جعبه‌ای از مجموعه داده CM1 مبتنی بر AUC را نشان می‌دهد. از سمت راست، نمودار جعبه‌ای اول، کارایی مدل‌هایی را نشان می‌دهد که از اندازه‌های طراحی استفاده کرده‌اند و نمودار بعدی، متعلق به اندازه‌های کد و نمودار آخر، از ترکیب اندازه‌های کد و طراحی (همه) به وجود آمده است. با توجه به شکل مشاهده می‌شود عملکرد مدل‌هایی که از اندازه‌های کد و همه استفاده کرده‌اند نسبت به مدل‌هایی که از اندازه‌های طراحی استفاده کرده‌اند بهتر است.



شکل (۱) : نمودار جعبه‌ای مجموعه داده CM1 مبتنی بر AUC

۳-۴- آزمون‌های آماری

دمسار، نگاهی به کارهای تئوری در آزمون‌های آماری برای مقایسه دسته‌بندی‌های متعدد، روی چند مجموعه داده داشته است. او آزمون رتبه علامت گذاری شده ویلکاکسون را برای مقایسه دو دسته‌بندی و آزمون فریدمن را برای مقایسه بیش از دو دسته‌بندی پیشنهاد کرد. در آزمون رتبه علامت‌گذاری شده ویلکاکسون و آزمون فریدمن، همتایان غیر پارامتری برای آزمون‌های پارامتری آنوا[۲۹] و آزمون تی[۳۰] هستند. از آنجائیکه در توزیع نرمال از آزمون‌های پارامتری و در توزیع غیر نرمال از آزمون‌های غیر پارامتری استفاده می‌شود؛ به علت اینکه در گروه همه، توزیع غیر نرمال وجود دارد هر چند در کد و طراحی توزیع نرمال است؛ ولی با این وجود مجبور هستیم از آزمون‌های غیر پارامتری استفاده کنیم. بر پایه پیشنهاد دمسار، ابتدا از آزمون فریدمن برای مقایسه کارایی مدل‌های برای مقایسه کارایی مدل‌های توسعه یافته از ۳ گروه اندازه‌ها روی ۱۷ مجموعه داده استفاده می‌شود و به شرط آن که آزمون‌های فریدمن، اختلاف قابل توجه آماری را نشان بدهد در آن صورت به مقایسه دو به دو از مدل‌های اتخاذ شده با آزمون ویلکاکسون نیاز خواهیم داشت، تا اینکه بهترین مدل برای هر مجموعه داده اجرا شود. فرض می‌شود که دو مدل از اندازه‌های مختلف A و B با ۲۰ روش مدل-سازی در یک مجموعه داده وجود دارد و می‌خواهیم بهترین مدل پیشگویی را از بین دو مدل استفاده کنیم.

فرضیه‌های آزمون آماری مناسب عبارتند از :

H0: اختلاف قابل توجه آماری در کارایی مدل‌هایی که از اندازه‌های گروه A یا B استفاده می‌کنند وجود ندارد، بنابراین مساوی در نظر گرفته می‌شوند.

H1: کارایی مدل گروه A از کارایی مدل گروه B بهتر است.

H2: کارایی مدل گروه A از کارایی مدل گروه B بدتر است.

با اجرای آزمون ویلکاکسون، مقدار p[۳۱]بزرگ‌تر از ۰/۰۵ نشان دهنده این است که هیچ اختلاف قابل توجهی در کارایی دو مدل از اندازه‌های A و B وجود ندارد، در چنین موردی H0 پذیرفته می‌شود و اگر مقدار p کوچک‌تر از ۰/۰۵ باشد بنا به ترتیب رتبه‌ها، H1 یا H2 پذیرفته می‌شود. پس از انجام این آزمون، رابطه کارایی بین مدل‌های گروه A و B به صورت زیر است :

اگر H0 آنگاه A=B و اگر H1 آنگاه A>B و اگر H2 آنگاه A<B

۴- نتایج تجربی

در این بخش، نتایج آزمایشات انجام شده که شامل نمودارهای جعبه‌ای و آزمون‌های آماری است، یک بار بر مبنای AUC و یک بار بر مبنای معیار F ارائه می‌شوند و سپس مقایسه‌ای بین مدل‌های پیشگویی خطای نرم‌افزار مبتنی بر AUC و معیار F با توجه به نمودارهای جعبه‌ای آن‌ها روی مجموعه‌های داده یکسان انجام می‌شود.

۴ -۱- نتایج تجربی مبتنی بر AUC

۴-۱-۱- نمودارهای جعبه‌ای مبتنی بر AUC

شکل ۱ نمودار جعبه‌ای برای AUC از مجموعه داده CM1 و شکل ۲ نمودارهای جعبه‌ای برای AUC از ۱۶ مجموعه داده دیگر را نشان می‌دهد. از این نمودارهای جعبه‌ای می‌توان روند زیر را مشاهده کرد.

- در کارایی مدل‌هایی که از ترکیب اندازه‌های کد و طراحی استفاده می‌کنند، نسبت به مدل‌هایی که از اندازه‌های کد استفاده می‌کنند، پیشرفت قابل توجهی وجود ندارد.
- کارایی مدل‌هایی که از اندازه‌های طراحی استفاده می‌کنند در اکثر موارد پایین‌ترین است. به صورت بصری در نمودارهای مربوط به پروژه‌های CM1، MC2، MC1، PC4، PC3، PC2، PC1، KC1، KC3 و AR4 می‌توان پایین بودن حالت طراحی نسبت به کد یا ترکیب آن دو را مشاهده کرد.

۴-۱-۲- آزمون‌های آماری مبتنی بر AUC

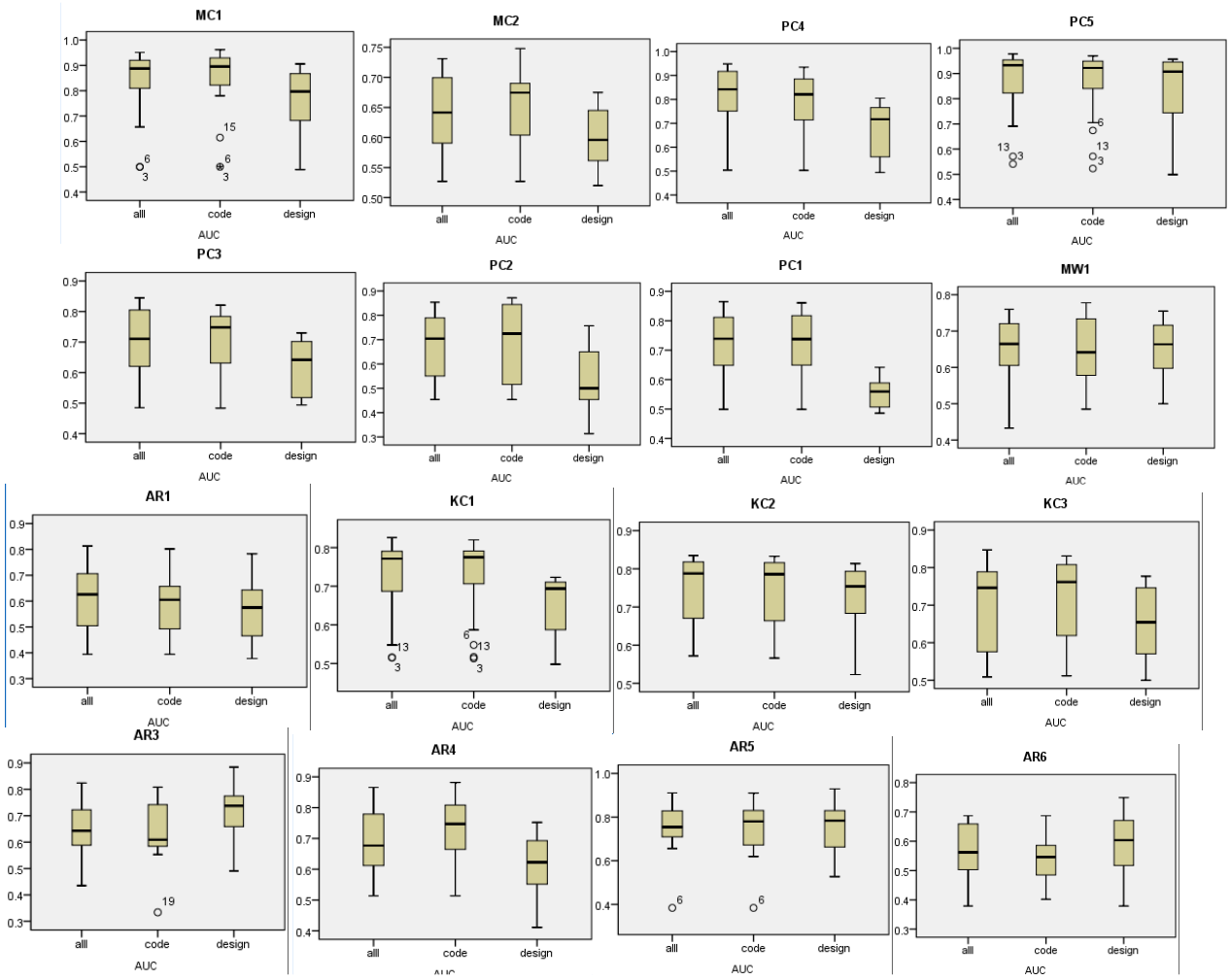
همان طوری که بیان شد بر پایه پیشنهاد دمسار، ابتدا از آزمون فریدمن برای مقایسه کارایی ۳ گروه اندازه‌ها روی ۱۷ مجموعه داده استفاده می‌شود. فرضیه این آزمون به صورت زیر است:

اگر مقدار P کمتر از ۰/۰۵ باشد به معنی آن است که اختلاف قابل توجهی در کارایی مدل‌های ساخته شده از سه گروه اندازه‌ها وجود دارد. جدول ۵ نتایج آزمون فریدمن را روی مجموعه‌های داده NASA و جدول ۶ نتایج آزمون فریدمن را روی مجموعه‌های داده AR نشان می‌دهد. در جدول ۵ در پروژه‌های KC3, PC1, PC2, KC1, PC3, PC4, PC5, MC1, MC2, KC2 و CM1 مقدار P کوچک‌تر از ۰/۰۵ است و در جدول ۶ در پروژه‌ی AR3 و AR4، مقدار P کوچک‌تر از ۰/۰۵ است. با توجه به فرضیه پروژه‌ها، اختلاف آماری قابل توجهی در کارایی مدل‌های پیشگویی نرم‌افزار وجود دارد. در آزمون فریدمن می‌توان ثابت کرد اختلاف آماری قابل توجه در کارایی مدل‌ها وجود دارد ولی نمی‌توان گفت که کارایی

کدام مدل بیشتر و یا کمتر است پس لازم است که به سراغ آزمون ویلکاکسون و مقایسه دو به دو روی مدل‌های توسعه یافته از گروه اندازه‌های گوناگون در هر مجموعه داده رفت. جدول ۷ نتایج آزمون ویلکاکسون را روی مجموعه‌های داده AR و جدول ۸ نتایج آزمون ویلکاکسون را روی مجموعه‌های داده NASA نشان می‌دهد. ستون دوم مدل‌های ساخته شده از اندازه‌های همه و کد را مقایسه می‌کند و ستون سوم مقایسه‌ای بین مدل‌های ساخته شده از گروه اندازه‌های همه و طراحی است در حالی که ستون چهارم مقایسه بین مدل‌های ساخته شده از گروه اندازه‌های کد و طراحی را فراهم می‌کند. ستون آخر نتایج مقایسه را بیان می‌کند.

با تجزیه و تحلیل نمودارهای جعبه‌ای به صورت آماری نتایج آزمون‌ها به صورت زیر بیان می‌شود:

- در ۴ مورد، کارایی مدل‌های ساخته شده از گروه اندازه‌های همه و کد و طراحی برابر هستند.
- در ۱۳ مورد، کارایی مدل‌های ساخته شده از گروه اندازه‌های همه و کد برابرند
- در ۲ مورد، کارایی مدل‌های ساخته شده از گروه اندازه همه بهتر از کارایی مدل‌های ساخته شده از گروه اندازه کد می‌باشد و در دو مورد دیگر بالعکس است.
- در ۱۰ مورد، کارایی مدل‌های ساخته شده از گروه اندازه طراحی کم‌ترین است.



شکل (۲) : نمودارهای جعبه‌ای مبتنی بر AUC از ۱۶ مجموعه داده

جدول (۵): نتایج آزمون فریدمن روی مجموعه داده‌های NASA

مجموعه‌های داده		KC1	KC3	PC1	PC2	PC3	PC4	PC5	MC1	MC2	MW1	CM1	KC2
آزمون فریدمن	مقدارP	۰	۰	۰	۰	۰	۰	۰/۰۱۷	۰/۰۰۱	۰	۰/۸۵۰	۰	۰/۰۲۰

جدول (۶) : نتایج آزمون فریدمن روی مجموعه داده‌های AR

مجموعه‌های داده		AR1	AR3	AR4	AR5	AR6
آزمون فریدمن	مقدارP	۰/۹۰۷	۰/۰۰۷	۰	۰/۱۳۵	۰/۱۲۶

جدول (۷) : نتایج آزمون ویلکاکسون روی مجموعه داده‌های AR

نتایج	فرضیه‌ها و مقدار p			مجموعه های داده
	طراحی- کد	طراحی - همه	کد - همه	
طراحی = کد = همه	۰/۸۴۵, H0	۰/۳۰۵, H0	۰/۱۹۶, H0	AR1
کد = همه < طراحی	۰/۰۲۳, H2	۰/۰۳۳, H2	۰/۱۸۴, H0	AR3
طراحی < همه < کد	۰/۰۰۱, H1	۰/۰۰۲, H1	۰/۰۲۰, H2	AR4
طراحی = کد = همه	۰/۶۶۸, H0	۰/۶۱۴, H0	۰/۵۴۲, H0	AR5
کد < طراحی = همه	۰/۰۱۳, H2	۰/۳۴۹, H0	۰/۰۲۶, H1	AR6

با توجه به آمار بدست آمده مشاهده می‌شود در عملکرد مدل پیشگو، اندازه‌های کد و ترکیب اندازه‌های کد و طراحی، در اکثر موارد (۱۳ مورد) تأثیر یکسانی دارند در حالی که این تأثیر برای اندازه‌های طراحی خیلی ضعیف‌تر است.

جدول (۸) : نتایج آزمون ویلکاکسون روی مجموعه داده‌های NASA

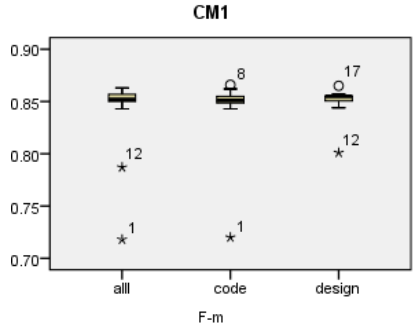
نتایج	فرضیه‌ها و مقدار p			مجموعه های داده
	طراحی - کد	طراحی- همه	کد - همه	
طراحی < کد = همه	۰, H1	۰, H1	۰/۶۲۹, H0	KC1
طراحی < کد = همه	۰/۰۰۲, H1	۰/۰۰۷, H1	۰/۳۸۷, H0	KC3
طراحی < کد = همه	۰, H1	۰, H1	۰/۰۹۶, H0	PC1
طراحی < کد = همه	۰/۰۰۱, H1	۰, H1	۰/۴۳۵, H0	PC2
طراحی < کد = همه	۰, H1	۰, H1	۰/۴۵۶, H0	PC3
طراحی < کد < همه	۰, H1	۰, H1	۰/۰۰۲, H1	PC4
طراحی = کد = همه	۰/۰۱۰, H1	۰/۰۵۲, H1	۰/۵۵۶, H0	PC5
طراحی < کد = همه	۰, H1	۰, H1	۰/۱۱۷, H0	MC1
طراحی < همه < کد	۰/۰۰۸, H1	۰/۰۰۲, H1	۰/۰۳۵, H2	MC2
طراحی = کد = همه	۰/۸۷۲, H0	۰/۴۶۹, H0	۰/۷۵۸, H0	MW1
طراحی < کد = همه	۰, H1	۰, H1	۰/۰۶۴, H0	CM1
کد = همه طراحی = همه طراحی <کد	۰/۰۲۸, H1	۰/۰۸۶, H0	۰/۷۳۲, H0	KC2

۴-۲- نتایج تجربی مبتنی بر معیار F

۴-۲-۱- نمودارهای جعبه‌ای مبتنی بر معیار F

شکل‌های ۳ و ۴ نمودارهای جعبه‌ای برای معیار F از ۱۷ مجموعه داده را نشان می‌دهد. از این نمودارهای جعبه‌ای نتایج زیر مشهود است:

- کارایی مدل‌هایی که از ترکیب اندازه‌های کد و طراحی استفاده می‌کنند نسبت به مدل‌هایی که از اندازه‌های کد استفاده می‌کنند تقریباً برابر است.
- کارایی مدل‌هایی که از اندازه‌های طراحی استفاده می‌کنند نسبت به مدل‌هایی که از اندازه‌های کد یا ترکیب اندازه‌های کد و طراحی استفاده می‌کنند در پروژه‌های KC1, KC3, PC1, PC3 و PC4 پایین‌تر است.



شکل (۳) : نمودارهای جعبه‌ای مبتنی بر معیار F از مجموعه داده CM1

۴-۲-۲- آزمون‌های آماری مبتنی بر معیار F

همان طوری که بیان شد بر پایه پیشنهاد دمسار، ابتدا از آزمون فریدمن برای مقایسه کارایی ۳ گروه اندازه‌ها روی ۱۷ مجموعه داده استفاده می‌شود. جدول ۹ نتایج آزمون فریدمن را روی مجموعه‌های داده NASA و جدول ۱۰ نتایج آزمون فریدمن را روی مجموعه‌های داده AR نشان می‌دهد. در پروژه‌های KC1, KC3, PC1, PC3, PC4, PC5, MW1, AR1, AR3 و AR4، مقدار P کوچک‌تر از ۰/۰۵ بوده و به این معناست که اختلاف آماری قابل توجهی در کارایی مدل‌های پیشگویی نرم‌افزار در این پروژه‌ها وجود دارد. پس لازم است که به سراغ آزمون ویلکاکسون و مقایسه دو به دو روی مدل‌های توسعه یافته از گروه اندازه‌های گوناگون در هر مجموعه داده رفت جدول ۱۱ نتایج آزمون ویلکاکسون را روی مجموعه‌های داده AR و جدول ۱۲ نتایج آزمون ویلکاکسون را روی مجموعه‌های داده NASA و نشان می‌دهد.

نتایج این آزمون‌ها به صورت زیر است :

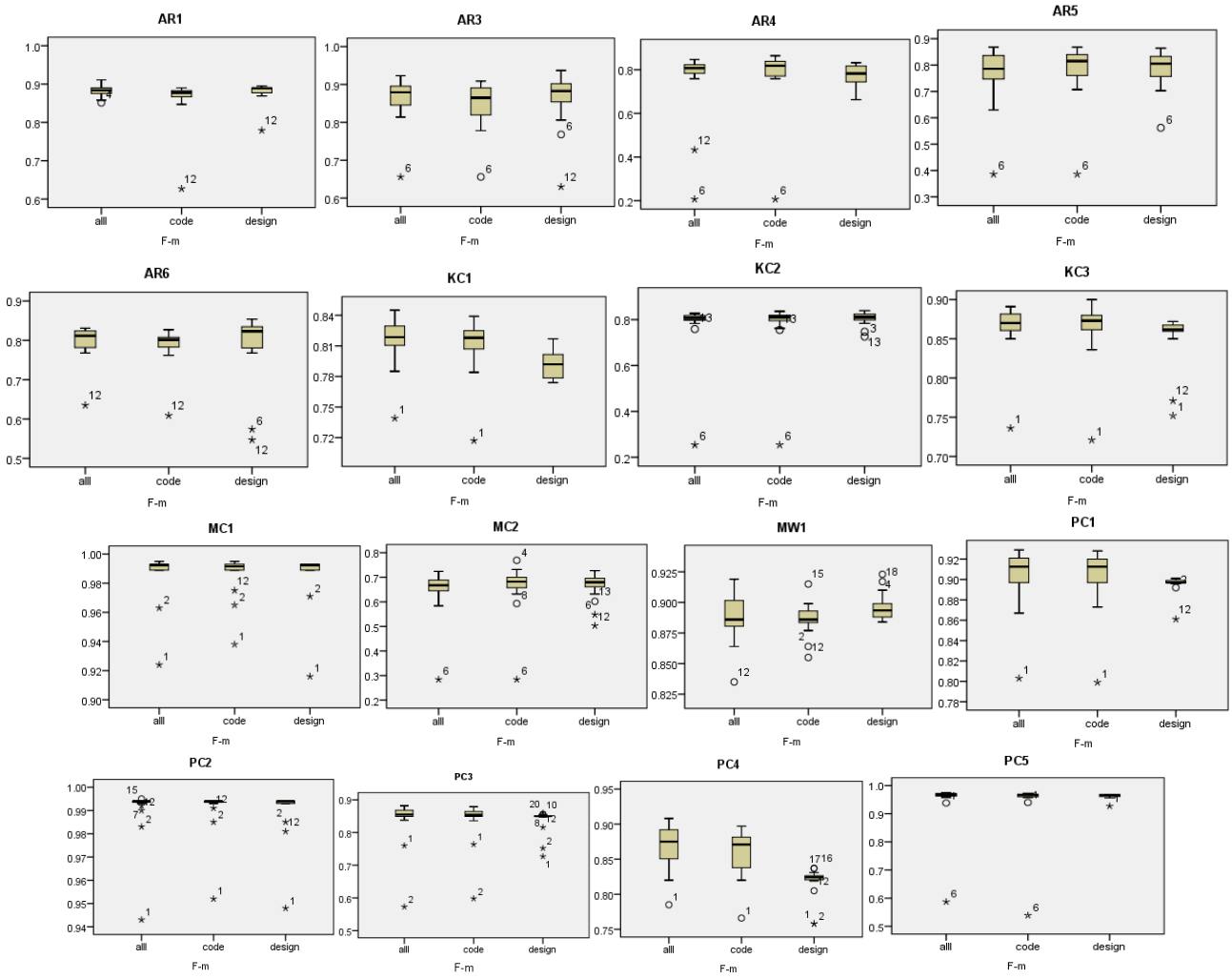
- در ۶ مورد، کارایی مدل‌های ساخته شده از گروه اندازه‌های همه و کد و طراحی برابر هستند.
- در ۱۳ مورد، کارایی مدل‌های ساخته شده از گروه اندازه‌های همه و کد برابر هستند.
- در ۳ مورد، کارایی مدل‌های ساخته شده از گروه اندازه همه بهتر از کارایی مدل‌های ساخته شده از گروه اندازه کد می‌باشد.
- در ۶ مورد، کارایی مدل‌های ساخته شده از گروه اندازه طراحی کم‌ترین است.

از آمار بالا باز هم مشاهده می‌شود که اندازه‌های کد و ترکیب اندازه‌های کد و طراحی در اکثر موارد (۱۳ مورد)، تأثیر یکسانی در کارایی مدل پیشگو دارند در حالی که این تأثیر برای اندازه‌های طراحی ضعیف‌تر است هرچند اندازه‌های طراحی نسبت به حالت AUC قوی‌تر عمل کرده است.

۴-۳- مقایسه مدل‌ها

مقایسه بین مدل‌های پیشگویی خطای نرم‌افزار مبتنی بر AUC و مبتنی بر معیار F با مقایسه نمودارهای جعبه‌ای آن‌ها روی مجموعه‌های داده یکسان امکان پذیر است.

یک نمونه از این مقایسه در شکل‌های ۵ و ۶ آمده است. همان‌طور که ملاحظه می‌شود محور عمودی در شکل ۵ معرف معیار F و در شکل ۶ معرف AUC است از آنجائیکه معیار F و AUC کاملاً متفاوت هستند نتایج مشاهده شده در هر آزمایش، به صورت مستقل مهم است نه در مقایسه با هم از مقدار حقیقی. با توجه به جایگاه نمودارهای جعبه‌ای نسبت به محور عمودی، در مدل پیشگویی خطای نرم‌افزار مبتنی بر معیار AUC، کارایی مدل‌هایی که از اندازه طراحی استفاده می‌کنند نسبت به کد و ترکیب اندازه‌های کد و طراحی پایین‌تر است در حالی که در مدل پیشگویی خطای نرم‌افزار مبتنی بر معیار F، کارایی مدل‌های به وجود آمده با اندازه طراحی و کد و ترکیب اندازه‌های کد و طراحی یکسان است. در این مقایسه استفاده از اندازه طراحی، عملکرد بهتری را در معیار F نشان می‌دهد و این بهبودی در مقایسه نمودار مبتنی بر AUC و معیار F برخی پروژه‌ها به چشم می‌خورد.



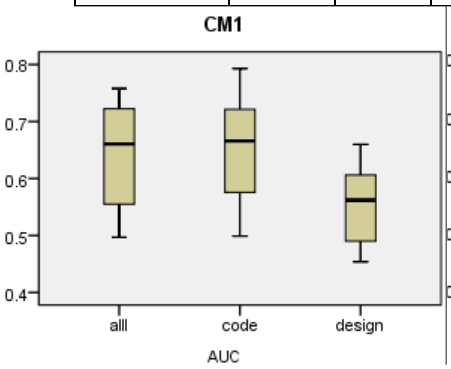
شکل (۴) : نمودارهای جعبه‌ای مبتنی بر معیار F از ۱۶ مجموعه داده

جدول (۹): نتایج آزمون فریدمن روی مجموعه داده‌های NASA

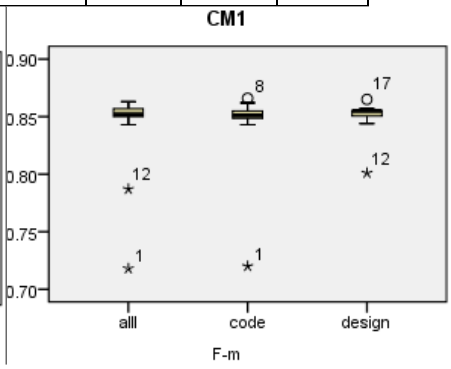
مجموعه‌های داده		KC1	KC3	PC1	PC2	PC3	PC4	PC5	MC1	MC2	MW1	CM1	KC2
آزمون فریدمن	مقدارp	۰	۰/۰۱	۰/۰۰۵	۰/۲۷۵	۰/۰۲۵	.	۰	۰/۲۹۴	۰/۷۸۴	۰/۰۰۲	۰/۴۰۱	۰/۸۴۵

جدول (۱۰) : .نتایج آزمون فریدمن روی مجموعه داده‌های AR

مجموعه‌های داده		AR1	AR3	AR4	AR5	AR6
آزمون فریدمن	مقدارp	۰/۰۱۲	۰/۰۱۰	۰/۰۰۴	۰/۴۳۴	۰/۳۲۵



شکل (۶) : نمودارهای جعبه‌ای CM1 مبتنی بر AUC



شکل (۵): نمودارهای جعبه‌ای CM1 مبتنی بر معیار F

جدول (۱۱) : نتایج آزمون ویلکاکسون روی مجموعه داده‌های AR

نتایج	فرضیه‌ها و مقدارp			مجموعه های داده
	طراحی - کد	طراحی - همه	کد - همه	
طراحی < کد= همه	۰/۰۰۱, H2	۰/۹۷۹, H0	۰/۰۵۰, H1	AR1
کد < همه طراحی=همه طراحی=کد	۰/۰۹۱, H0	۰/۶۰۱, H0	۰/۰۰۱, H1	AR3
کد = همه طراحی=همه طراحی<کد	۰/۰۳۵, H1	۰/۲۰۴, H0	۰/۰۸۸, H0	AR4
کد > همه طراحی=همه طراحی=کد	۱, H0	۰/۳۰۱, H0	۰/۰۴۲, H2	AR5
طراحی= کد = همه	۰/۲۱۲, H0	۰/۲۳۱, H0	۰/۲۲۴, H0	AR6

جدول (۱۲) : نتایج آزمون ویلکاکسون روی مجموعه داده‌های NASA

نتایج	فرضیه‌ها و مقدارp			مجموعه های داده
	طراحی- کد	طراحی -همه	کد -همه	
طراحی < کد =همه	۰/۰۰۲, H1	۰/۰۰۱, H1	۰/۰۸۳, H0	KC1
طراحی< کد =همه	۰/۰۱۶, H1	۰/۰۰۷, H1	۰/۸۸۸, H0	KC3
طراحی< کد =همه	۰/۰۰۵, H1	۰/۰۰۷, H1	۰/۴۱۹, H0	PC1
طراحی = کد = همه	۰/۰۶۶, H0	۰/۹۳۱, H0	۰/۱۶۷, H0	PC2
طراحی< کد =همه	۰/۰۲۴, H1	۰/۰۱۹, H1	۰/۸۸۷, H0	PC3
طراحی < کد <همه	۰, H1	۰, H1	۰/۰۰۷, H1	PC4
طراحی = کد <همه	۰/۰۷۲, H0	۰/۰۰۷, H1	۰/۰۰۴, H1	PC5
طراحی = کد =همه	۰/۸۱۲, H0	۰/۱۳۴, H0	۰/۴۴۵, H0	MC1
طراحی = کد = همه	۰/۶۴۱, H0	۱, H0	۰/۱۹۱, H0	MC2
کد = همه <طراحی	۰/۰۰۲, H2	۰/۰۳۷, H2	۰/۵۷۰, H0	MW1
طراحی= کد =همه	۰/۳۸۲, H0	۰/۲۱۹, H0	۰/۸۳۳, H0	CM1
طراحی= کد =همه	۰/۵۰۱, H0	۰/۲۶۳, H0	۰/۹۲۵, H0	KC2

۴- کارهای مرتبط پیشین

اولین کار داده کاوی توسط سلبی و پورتر [5] انجام شده است. آن‌ها از ۱۶ پروژه ناسا استفاده کردند و یک روش خودکار برای تولید مدل‌های تجربی مبتنی بر اشیا برای شناسایی خطاهای نرم افزار ارائه کردند و نشان دادند که به طور متوسط درختان دسته بندی کننده، ۷۹/۳ ٪ از پیمانه‌های نرم افزار را که دارای خطا بوده‌اند شناسایی کردند. بعد از آن نیز بسیاری از محققان روی موضوع پیشگویی خطای نرم افزار کار کرده‌اند و برای به دست آوردن مدل‌های پیشگویی صحیح تلاش شده است[17] .

در سیستم‌های شی گرا، از مجموعه اندازه‌های شی گرا (اندازه‌های CK) به عنوان پیشگو استفاده شده است [18]. در این مطالعات، اندازه‌های طراحی شی گرا از کد منبع استخراج شده‌اند و بر سودمندی این اندازه‌ها اشاره شده است. اما در مقاله حاضر، اندازه‌های طراحی از نمودارهای طراحی مثل گراف‌های جریان و نمودارهای UML استخراج شده‌اند و در این زمینه مطالعات کمی وجود دارد که اندازه‌های طراحی از ابزار طراحی جمع‌آوری شده باشد.

در یکی از مطالعات اولیه، آلبرگ و آلسن [19] در مخابرات اریکسون AB پیمانه‌های دارای خطا را پیش‌بینی کردند. آن‌ها اندازه‌های طراحی خود را از گراف‌های FDL استخراج کرده و یک مجموعه‌ای از اندازه‌های مستقیم مثل تعداد شاخه‌ها، تعداد گراف در پیمانه، تعداد اتصالات در گراف و اندازه‌های غیر مستقیم مثل پیچیدگی تولید می‌کنند.

در تحقیقاتی هم اندازه‌های طراحی از نمودارهای UML استخراج شده‌اند. جنرو و همکارانش اندازه‌های طراحی UML را که می‌تواند در پیشگویی مورد استفاده قرار گیرد بررسی کردند[20]. آن‌ها نشان دادند که ماکزیمم DIT[۳۲] از یک کلاس که نشان دهنده‌ی پیچیدگی ساختار است با قابلیت فهم و تغییرپذیری نمودارهای کلاس مرتبط است. مطالعه قبلی همچنین اندازه‌های سطح جزئیات[۳۳] در مدل‌های UML و ارتباط آن با تراکم خطا در پیاده‌سازی را وارسی کرده است و نشان داده‌اند که سطح جزئیات نمودارهای ترتیبی به طور قابل توجهی با تراکم خطای کلاس‌ها مرتبط است[21].

سیستا [22] نمودارهای UML را از کد جاوا با استفاده از آنالیز کد ایستا و پویا بازیابی کرد. تنلا و پتریش[23] نمودارهای ترتیبی را از کد شی گرا با آنالیزهای ایستا روی جریان داده استخراج کردند. برایان و همکارانش [24] بازیابی نمودارهای ترتیبی، شرایط و جریان داده را از کد جاوا به وسیله استفاده از روش‌های تبدیل نشان دادند، ولی تنها سه مطالعه است که کارایی اندازه‌های کد و طراحی را در پیشگویی خطای نرم‌افزار مقایسه می‌کند.

اولین مطالعه به وسیله ژئو و همکارانش در [6] است. آن‌ها اندازه‌های طراحی را از گراف FDL از یک سیستم بلادرنگ جمع‌آوری کردند و برای ساخت مدل‌های رگرسیون، دو مشخصه پیشگو از گروه اندازه‌های طراحی و فقط یک مشخصه از گروه اندازه‌های کد و دو مشخصه نیز برای ترکیب اندازه‌های کد و طراحی به عنوان بهترین پیشگوها انتخاب کردند، سپس با محاسبه تعداد خطا در هر مدل و تفسیر آن اظهار کردند که مدل پیشگویی که از اندازه‌های طراحی استفاده می‌کند به خوبی مدلی است که از اندازه‌های کد استفاده می‌کند. آن‌ها همچنین بیان کردند که موقعی که اندازه‌های طراحی و اندازه‌های کد ترکیب می‌شوند و با هم به عنوان پیشگوها استفاده می‌شوند (طراحی = کد = ترکیب) پیشرفت مهمی وجود ندارد.

دومین مطالعه به وسیله جیان و همکارانش [7] صورت گرفته است که مدل‌های پیشگویی ساخته شده با استفاده از اندازه‌های کد و طراحی روی ۱۳ مجموعه داده NASA را مقایسه کرده‌اند. آن‌ها اندازه‌های طراحی را از گراف‌های FDL و نمودارهای UML جمع‌آوری کرده و از آن‌ها به عنوان پیشگوها استفاده کرده‌اند و از AUC هر مدل، برای بررسی و مقایسه عملکرد مدل‌ها استفاده کردند. آن‌ها اظهار کردند از بین سه مدل پیشگویی که از اندازه‌های کد یا اندازه‌های طراحی و یا ترکیب اندازه‌های کد و طراحی استفاده می‌کنند مدل پیشگویی که از ترکیب اندازه‌های کد و طراحی ساخته می‌شود بالاترین کارایی و مدل پیشگویی که از اندازه‌های طراحی ساخته می‌شوند پایین‌ترین کارایی را دارند (طراحی < کد < ترکیب).

سومین مطالعه به وسیله نوگراهو و همکارانش [8] روی داده‌های یک صنعت مهم با سیستم جاوا صورت گرفته است. آن‌ها اندازه‌ها را از نمودارهای ترتیبی مدل UML (انتخاب دو مشخصه[۳۴] از گروه طراحی) و از کد منبع (انتخاب یک مشخصه[۳۵] از گروه کد) استخراج کردند و سه مدل پیشگویی بر اساس اندازه‌های UML و بر اساس اندازه‌های کد و بر اساس ترکیب اندازه‌های کد و UML ساختند و نشان دادند که دو اندازه طراحی UML که از نمودارهای ترتیبی به دست آورده بودند پیشگوهای مهمی از کلاس هستند. همچنین آن‌ها یافتند که مدل پیشگویی که از ترکیب اندازه‌های طراحی UML و اندازه‌های کد استفاده می‌کند بهترین کارایی را دارد و استفاده از اندازه‌های UML به عنوان پیشگو نسبت به اندازه‌های کد، عملکرد بهتری دارد (کد<طراحی<ترکیب).

به هر حال، یافته‌های آنان هر کدام بر اساس تحلیل‌هایی از مجموعه‌های داده متفاوت است. در این مقاله برای ساخت مدل‌های پیشگو، از طیف وسیع‌تری از مجموعه‌های داده و ۲۰ دسته‌بندی کننده استفاده شده در حالی که در مطالعات مرتبط قبلی، حداکثر ۵ دسته-بندی کننده به کار برده شده بود. همچنین برای ارزیابی کارایی مدل‌های پیشگو، ضمن استفاده از AUC، از معیار F که در هیچ یک از مطالعات مشابه قبلی استفاده نشده بود، کمک گرفته شده است و نتایج قابل توجهی مشاهده می‌شود (طراحی< کد = ترکیب).

۵- نتیجه گیری

هدف این مقاله بررسی مقایسه تأثیر اندازه‌های طراحی نسبت به اندازه‌های کد در بهبود عملکرد سامانه‌های آزمون خودکار بوده است و برای ساخت مدل‌های پیشگوی خطای نرم‌افزار، از ۱۷ مجموعه داده‌ی در دسترس، از انباره داده پرامیس و ۲۰ دسته‌بندی کننده استفاده شده است و کارایی این مدل‌ها با AUC و معیار F ارزیابی شده است.

با توجه به نمودارهای جعبه‌ای و آزمون‌های آماری دیده می‌شود در عملکرد مدل‌هایی که از ترکیب اندازه‌های کد و طراحی استفاده می‌کنند نسبت به آن‌هایی که از اندازه‌های کد استفاده می‌کنند بهبودی قابل توجهی مشاهده نمی‌شود. همچنین اندازه‌های طراحی نسبت به اندازه‌های کد و یا ترکیب اندازه‌های کد و طراحی، تأثیر کمتری بر افزایش کارایی سامانه‌های آزمون خودکار دارد. ضمناً مدل پیشگویی خطای نرم‌افزار مبتنی بر معیار F نسبت به مدل پیشگویی خطای نرم‌افزار مبتنی بر AUC، کارایی بهتری را در هر سه گروه از اندازه‌های کد و طراحی و ترکیب اندازه‌های کد و طراحی ارائه کرده است که البته اختلاف بین عملکرد مدل‌هایی که از اندازه‌های کد و یا ترکیب اندازه‌های کد و طراحی استفاده می‌کنند با مدل‌هایی که از اندازه‌های طراحی استفاده می‌کنند در مدل‌های مبتنی بر معیار F نسبت به AUC کمتر است..

این نتیجه‌ها با توجه به مقایسه نمودارهای جعبه‌ای مبتنی بر AUC در شکل‌های ۱ و ۲ و نمودارهای جعبه‌ای مبتنی بر معیار F در شکل‌های ۳ و ۴ مشهود است. همچنین با توجه به مقایسه نتایج آزمون‌های آماری مبتنی بر AUC در جدول‌های ۷ و ۸ و نتایج آزمون‌های آماری مبتنی بر معیار F در جدول‌های ۱۱ و ۱۲ کاملاً محسوس است.

مراجع

[1] N. Ohlsson, M. Helander, C. Wohlin, "Quality improvement by identification of fault-prone modul using software design metrics", Proc, 6th International Conference on Software Quality, pp. 1–13, Ottawa, Canada, 1996.

[2] W. Gibbs, Software's chronic crisis, Scientific American, 86–94, September (1994).

[3] I. Sommerville, Software Engineering, Addison-Wesley, Reading, MA, 1996.

[4] F. Shull, V. B. ad B, Boehm, A. Brown, P. Costa, M. Lindvall, D. Port, I. Rus, R. Tesoriero, M. Zelkowitz, "What we have learned about fighting defects", in Proceedings of 8th International Software Metrics Symposium, pp. 249–258, Ottawa, Canada, 2002. available from http://fcmd.umd.edu/fcmd/Papers/shull defects.ps.

[5] R.W. Selby, A.A. Porter, "Software metric classification trees help guide the maintenance of large-scale system", Software Maintenance, 1989., Proceedings., Conference on, pp. 116–123, Miami, FL, USA, 1989.

[6] M. Zhao, C. Wohlin, N. Ohlsson, and M. Xie. "A comparison between software design and code metrics for the prediction of software fault content" Information and Software Technology, 40(14) :801–809, 1998.

[7] Y. Jiang, B. Cuki, T. Menzies, and N. Bartlow. "Comparing design and code metrics for software quality prediction", In Proceedings of the 4th international workshop on Predictor models in software engineering, pages 11–18. ACM NewYork, NY, USA, 2008.

[8] A. Nugroho, M.r.v. Chaudron, E. Arisholm, "Assessing UML Design Metrics for Predicting Fault-prone Classes in a Java System", IEEE, PP. 21-30, 2010.

[9] Metric data program. NASA Independent Verification and Validation facility, Available from http://MDP.ivv.nasa.gov.

[10]. Available http://tunedit.org/repo/PROMISE/DefectPrediction

[11] Promise data repository. Available http://promisedata.org/repository

[12] U. of Waikato. Weka software package. The University of Waikato, available http://www.cs.waikato.ac.nz/ml/weka/.

[13] J. Demsar, "Statistical comparisons of classifiers over multiple data sets", Journal of Machine Learning Research, 7, pp. 1-30, 2006.

[14] SPSS Statistics 19.0.0

[15] L. Breiman. Random forests. Machine Learning, 45:5–32, 2001.

[16] I. H. Witten and E. Frank, Data Mining, Practical machine learning tools and techniques, Morgan Kaufmann, Los Altos, US, 2005.

[17] R. Moser, W. Pedrycz, and G. Succi, "Comparative analysis of the efficiency of change metrics and static code attributes for defect prediction", Proceedings of the 30th International Conference on Software Engineering, pages 181–190, 2008.

[18] L. Briand, W. Melo, and J. W¨ust, "Assessing the Applicability of Fault-Proneness Models Across Object-Oriented Software Projects", IEEE Transactions on Software Engineering, pages706–720, 2002.

[19] N. Ohlsson and H. Alberg, "Predicting fault-prone software modules in telephone switches", IEEE Transactions on Software Engineering, 22(12) , 886–894, 1996**..**

[20] M. Genero, E. Manso, A. Visaggio, G. Canfora, and M. Piattini, "Building measure-based prediction models for UML class diagram maintainability", Empirical Software Engineering, 12(5) , 517–549, 10 2007.

[21] A. Nugroho, B. Flaton, and M. R. V. Chaudron, " Empirical analysis of the relation between level of detail in UML models and defect density", In Czarnecki, editor, Proceedings of the 11th International Conference on Model Driven Engineering Languages and Systems (MODELS), volume 5301 of LNCS, pages 600–614. Springer-Verlag, 2008.

[22] T. Systa, "Static and dynamic reverse engineering techniques for Java software systems", PhD thesis, 2000.

[23] P. Tonella and A. Potrich, "Reverse engineering of object oriented code", Springer-Verlag, Berlin, Heidelberg, New York, 2005.

[24] L. Briand, Y. Labiche, and J. Leduc, "Toward the reverse engineering of uml sequence diagrams for distributed java software", IEEE Transactions on Software Engineering, 32(9) , 642–663, 2006.

زیرنویس‌ها

[1] metric
[2] 8 to 20 LOC/minute
[3] Automatic defect prediction
[4] Data mining
[5] Code metrics
[6] Design metrics
[7] performance
[8] module
[9] diagram
[10] boxplot
[11] test
[12] F-measure
[13] Area under the curve
[14] Machine learning
[15] Receiver operating characteristic
[16] Friedman
[17] Wilcoxon
[18] Machine learners
[19] DEFECT
[20] TRUE
[21] FALSE
[22] Random Forest
[23] Sensitivity(recall)
[24] specificity
[25] precision
[26] recall
[27] Cross-validation
[28] train
[29] Anova
[30] t- test
[31] P-value
[32] Depth of Inheritance Tree
[33] Level of detail(LoD)
[34] Message detailedness and import coupling
[35] KSLOC