JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 27, No. 2, pp. 545-556, Feb. 2026

Check for updates

오픈소스 개발자 분석을 위한 다차원적 활동 지표 제안 및 시계열적 특성 규명

김 가 은[1] · 이 예 영[2] · 김 장 원[3*]
[1]국립군산대학교 소프트웨어학과 학부생
[2]국립군산대학교 소프트웨어학과 학사
[3]국립군산대학교 소프트웨어학과 교수

Multi-Dimensional Activity Metrics for Open-Source Developers: A Time-Series Characterization

Gaeun Kim[1] · Yeyeong Lee[2] · Jangwon Gim[3*]

[1]Undergraduate Student, Department of Software Science & Engineering, Kunsan National University, Gunsan 54150, Korea
[2]B.S., Department of Software Science & Engineering, Kunsan National University, Gunsan 54150, Korea
[3]Professor, Department of Software Science & Engineering, Kunsan National University, Gunsan 54150, Korea

[요 약]

최근 오픈소스 소프트웨어 플랫폼의 급격한 성장으로 개발자 활동 데이터를 활용한 연구가 증가하고 있다. 그러나 기존 연구의 대다수는 커밋이나 이슈의 단순 합산에 의존하여, 개발자의 생산성을 양적인 관점에서만 평가한다는 한계를 지닌다. 이를 해결하기 위해, 본 논문에서는 SW개발 프로세스와 협업 맥락을 다차원적으로 측정할 수 있는 4개 차원의 21가지 세분화된 지표를 제안한다. 앙상블 학습을 통해 숙련도별 핵심 변수를 식별하고, 복합 신뢰도 지수를 적용하여 6가지 핵심 시계열 지표의 최적 해상도를 도출하였다. 그 결과 활동량 기반 5계층 분류에서 92.4%의 정확도를 달성하였다. 본 논문에서 제안하는 맥락 중심의 프레임워크는 향후 역량 진단 및 맞춤형 추천 시스템 구축을 위한 실증적 기반을 제공할 것으로 기대된다.

[Abstract]

With the rapid growth of open-source software platforms, research utilizing developer activity data has increased substantially. However, most existing studies rely on the simple aggregation of commits or issues and evaluate developer productivity solely from a quantitative perspective. To address this limitation, this paper proposes 21 granular metrics, categorized into four dimensions, to measure software development processes and collaborative contexts in a multidimensional manner. Using ensemble learning, we identified key variables by proficiency level and applied the Composite Reliability Index to determine optimal temporal resolutions for six core time-series metrics. As a result, the classification model achieved 92.4% accuracy in predicting five activity-based proficiency tiers. The context-centered framework proposed in this study is expected to provide a robust empirical foundation for advanced competency diagnosis and personalized recommendation systems.

색인어 : 오픈소스 소프트웨어, 개발자 생산성, 다차원 지표, 앙상블 학습, 시계열 분석

Keyword : Open-Source Software, Developer Productivity, Multidimensional Metrics, Ensemble Learning, Time-Series Analysis

http://dx.doi.org/10.9728/dcs.2026.27.2.545

Received 22 December 2025; **Revised** 23 January 2026
Accepted 06 February 2026

***Corresponding Author; Jangwon Gim**

Tel: +82-63-469-8916
E-mail: jwgim@kunsan.ac.kr

Ⅰ. 서 론

최근 깃헙(GitHub)과 같은 오픈소스 소프트웨어 플랫폼이 글로벌 소프트웨어 생태계의 중심으로 자리 잡으면서, 축적된 개발자 활동 데이터는 소프트웨어 공학의 핵심 연구 자원이 되고 있다[1]. 과거의 제한적인 개발 환경과 달리, 오늘날 오픈소스 기여 내역은 개발자의 기술적 전문성과 협업 역량을 실증하는 투명한 '디지털 포트폴리오'로서 기능하고 있다[2]. 기업과 조직 관점에서 이러한 데이터는 단순히 인재를 채용하기 위한 참고 자료를 넘어, 프로젝트의 지속 가능성을 예측하고 조직 내 핵심 기여자를 식별하기 위한 필수적인 지표로 활용된다[3]. 특히 원격 근무와 비동기 커뮤니케이션이 보편화된 현대의 개발 환경에서, 코드 리뷰를 통한 지식 공유나 이슈 토론 과정에서 드러나는 문제 해결 능력은 개발자의 '소프트 스킬'까지 정량적으로 파악할 수 있는 유일한 근거가 되기에 그 분석의 중요성은 더욱 증대되고 있다[4],[5]. 그러나 기존의 대다수 소프트웨어 저장소 마이닝 연구는 커밋, 이슈, 풀 리퀘스트 등의 단순 발생 건수를 정적으로 합산하여 평가하는 1차원적인 방식에 머물러 있다[6]-[9]. 이러한 단순 빈도 기반 접근은 개발자가 '얼마나 많이' 활동했는지에 대한 물리적 양은 보여줄 수 있으나, 해당 기여가 프로젝트에 미치는 실질적 가치나 기술적 난이도를 반영하지 못한다는 결정적인 한계를 갖는다. 예를 들어, 코드의 가독성을 위한 단순 공백 수정과 시스템의 아키텍처를 개선하는 리팩토링이 동일한 '1회의 커밋'으로 간주될 경우, 개발자의 실제 생산성은 왜곡될 수 있다[10]. 또한, 이러한 정적 집계 방식은 개발자가 '어떻게' 일하는지에 대한 심층적인 프로세스 정보를 소실시킨다. 높은 커밋 수가 반드시 높은 업무 효율성을 담보하지 않으며, 오히려 빈번한 수정이나 반복적인 재작업은 불필요한 코드 작성을 야기하거나 심각한 개발 비효율성을 방증하는 부정적 신호일 수 있다[11],[12]. 협업 관점에서도 단순한 댓글 수보다는 코드 리뷰 텍스트에 내포된 피드백의 구체성이나 기술적 깊이가 훨씬 중요함에도 불구하고, 기존 지표는 이러한 질적 상호작용의 맥락을 탐지하지 못한다[13]. 무엇보다 가장 큰 문제는 시간의 흐름에 따라 변화하는 동적인 맥락 정보의 부재이다. 개발자의 활동은 고정된 것이 아니라 프로젝트의 생명주기나 개인의 역량 성숙도에 따라 그 역할과 기여 패턴이 끊임없이 진화한다[14]. 그러나 전체 기간에 대한 평균값이나 단순 합산은 이러한 시계열적 개발 활동의 변동성을 평탄화하여, 개발자의 성장 추세나 위기 징후와 같은 중요한 시그널을 축소하는 결과를 초래한다.

본 논문은 정적 횟수 중심의 평가를 다차원적 지표와 동적 맥락의 관점에서 재정립하는 새로운 분석 프레임워크를 제안한다. 구체적으로, 개발 효율성과 협업의 질적 특성을 포괄하는 21가지의 다차원적 활동 지표를 정의하고, 통계적 검증을 통해 각 지표에 최적화된 시계열 해상도를 규명한다[15]. 이러한 다각적 접근은 데이터의 노이즈를 제거하고 개발자의 고유한 행동 패턴을 입체적으로 포착함으로써, 개발자 분석의 패러다임을 단순 총량 중심에서 동적 프로세스 중심으로 전환하는 실증적 방법론을 제시한다. 본 논문의 구성은 다음과 같다. 제2장에서는 관련 연구를 검토하고 기존 정량적 지표의 한계를 논한다. 제3장에서는 개발 효율성과 협업 맥락을 포괄하는 21가지의 다차원적 활동 지표를 정의하고, 앙상블 학습 및 시계열 해상도 최적화를 위한 통계적 방법론을 기술한다. 이어지는 제4장에서는 지표 간 상관관계와 변수 중요도 분석, 그리고 시계열 패턴 검증 결과를 순차적으로 제시한다. 제5장에서는 도출된 통계적 수치를 바탕으로 개발자의 성장 경로와 생태계의 구조적 특성을 심층적으로 고찰하며, 마지막으로 제6장에서는 결론과 함께 연구의 한계 및 향후 연구 방향을 제시한다.

Ⅱ. 관련연구

기존 연구들은 개발자 활동을 정적 건수 또는 고수준 집계 지표에 초점을 맞추고 있다. 대표적인 예로, [6]은 프로젝트 추천 시스템을 위해 이슈 생성, 이슈 댓글, PR 생성, PR 병합 등 10개의 개별 개발 활동에 대한 누적 건수를 지표화하였다. 또한, 이들은 개별 활동 수치를 산술적으로 결합한 융합 지표와, 활동의 발생여부만을 0 또는 1로 식별하는 이진 융합지표를 추가로 정의하였다. 이와 같은 정량적 접근 방식은 소프트웨어 저장소 마이닝 분야 전반에서 널리 활용되고 있다. 예를 들어, 리뷰어 할당 문제에는 PR 수락률이나 파일 위치 정보가 활용되었으며[7], 소프트웨어 결함 예측에는 커밋 파일 수나 개발자 경험 데이터가 주요 변수로 사용되었다[8]. 또한, 코드 소유권을 정의하기 위해서는 수정된 라인 수가 핵심 지표로 제안된 바 있다[12].

소프트웨어 개발 공수 예측 분야에서도 유사한 접근 방식이 확인된다. [16]은 개발자 활동 기반 지표를 제안하며, 그 핵심 요소로 총 개발자 수, 총 SLOC 수정량, 총 개발 소요 시간 등을 사용하였다. 해당 연구는 시간 요소를 지표에 포함하고 있으나, 이를 이슈 해결 주기와 같은 프로세스 효율성을 분석하는 데는 충분히 활용하지 못했다. 단지 총 소요 시간이라는 거시적인 집계 값으로 취급하여, 최종 공수 산정을 위한 입력 변수로만 활용했다는 한계를 갖는다.

이러한 정량적 편중은 개발자 경험 정량화에 관한 문헌을 종합적으로 분석한 매핑 연구에서도 알 수 있다[17]. 해당 연구에 따르면, 대다수의 지표가 개발자의 코드 활동을 단순히 산술적으로 계산하는 방식에 국한되어 있음을 알 수 있다.

이처럼 횟수나 총량에 집중하는 방식은 활동은 양적 규모를 측정하는 데는 유용할 수 있으나, 개발 수행 과정에 대한 심층적인 통찰을 제공하지 못한다는 점에서 다음과 같은 명확한 한계를 갖는다. 첫째, 정적 지표는 효율성을 측정하지 못한다[18]. 생산성 프레임워크인 SPACE에서는 활동량과 효

율성이 다른 차원임을 강조하지만, 기존 연구들은 이를 충분히 고려하지 못하고 있다. 예를 들어, 10개의 이슈를 10일 만에 처리한 개발자와 100일 만에 처리한 개발자는 동일한 해결 건수로 집계되어, 생산성의 차이를 식별하기 어렵다. 둘째, 협업의 맥락을 반영하지 못한다. 단순한 커밋 건수 집계는 그것이 독립적인 작업인지, 아니면 동료와의 활발한 협업 결과물인지를 구분하지 못해 협업의 질적 측면을 분석하는 데 한계가 있다. 기존 연구들은 코드 리뷰 텍스트에 내포된 피드백의 구체성과 같은 질적 요소를 간과하는 경향이 있다[13]. 셋째, 데이터의 동적 패턴을 평탄화 시킨다. 기존의 누적 총합 방식은 시계열적 특성을 고려하지 않아, 연 단위나 월 단위로 변동하는 개발자의 민첩한 변화 패턴을 포착하지 못하고 잘못된 해석을 유발할 수 있다. 특히 딥러닝 프레임워크 커뮤니티와 같이 생명주기가 뚜렷한 프로젝트에서 개발자의 역할과 기여 패턴은 끊임없이 진화하므로, 이러한 동적 맥락의 부재는 한계가 있다[14]. 즉, 기존 연구들은 개발자의 활동을 동적인 과정이 아닌 정적인 결과의 관점에서만 해석했다는 공통적인 한계를 가진다. 따라서 본 논문은 이러한 한계를 극복하기 위해, 개발자 행동 패턴을 동적 프로세스 관점에서 진단할 수 있는 21가지 지표를 제안한다. 나아가 시계열 분석 방법론[15]에 기반하여 변동계수와 자기상관계수를 활용하여 시계열 특성을 규명함으로써, 단순 정적 집계에 머물렀던 기존 연구들과는 달리 활동의 맥락과 흐름을 입체적으로 분석한다는 점에서 명확한 차별성을 갖는다.

Ⅲ. 연구방법

3-1 데이터 수집 및 전처리

깃헙 생태계 내 개발자들의 활동 패턴 분석을 위해, 2024년 1년간의 깃헙 아카이브 데이터를 Google Cloud에서 활용하였다[19]. 2024년 한 해 동안 깃헙 내에서 활동한 고유 사용자는 약 2,544만 명(25,445,857명)으로 식별되었다. 이 중, 코드 커밋, 풀 리퀘스트, 이슈 생성 등 실질적인 기여 활동을 최소 5회 이상 수행한 '핵심 기여자' 그룹은 약 1,561만 명(15,616,980명)에 달하는 것으로 확인되었다. 이는 본 논문에서 사용하는 실험데이터가 통계적 분석을 수행하기에 충분한 규모와 다양성을 갖추고 있음을 시사한다. 그러나 모집단의 활동량 분포 분석 결과, 대다수의 인원이 하위 활동 그룹에 밀집하고 상위 소수가 전체 활동의 상당 부분을 점유하는 전형적인 파레토 분포를 따르는 것으로 확인되었다. 이러한 데이터 불균형 문제를 해소하고 각 숙련도별 특성을 공정하게 비교하기 위해, 본 논문은 전체 모집단에서 무작위로 200만 명의 후보군을 우선 선정한 후, 표 1과 같이 연간 총 활동량을 기준으로 개발자를 5개 계층(Very Low ~ Very High)으로 구분하는 층화 추출법을 적용하였다.

표 1. 개발자 기여도에 따른 계층적 표본 추출 범주 및 현황
Table 1. Stratified sampling categories based on developer contributions

category	range	counts
Very_Low	0 ~ 249	1000
Low	250 ~ 499	1000
Medium	500 ~ 749	1000
High	750 ~ 999	1000
Very_High	1000 ~	1000

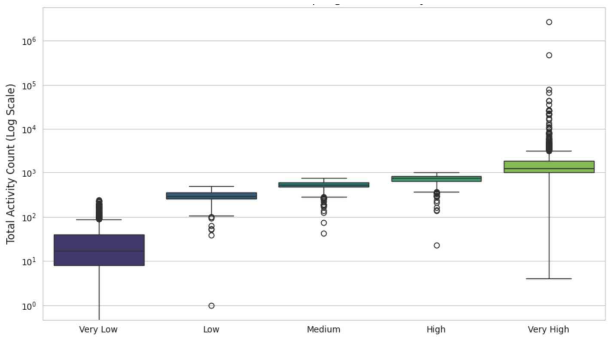

그림 1. 연간 총 활동량 기준의 개발자 5계층 분류 및 활동 분포
Fig. 1. Classification and activity distribution of five developer tiers based on total annual activity

구체적으로 활동량 0회부터 1,000회까지의 구간을 250 단위의 등간격으로 나누어 하위 4개 등급을 설정하였으며, 1,000회 이상의 활동을 수행한 그룹은 'Very High' 등급으로 통합하였다. 본 논문에서 등간격 분류를 채택한 이유는 활동량의 절대적 의미를 보존하기 위함이다. 분위수 기반 분류는 상대적 순위만을 반영하므로, 예컨대 연간 250회 활동과 750회 활동 간의 실질적 차이가 희석될 수 있다. 반면 등간격 분류는 동일한 활동량 증가분이 각 계층 간 일관된 의미를 갖도록 하여, 계층별 특성 비교의 해석 가능성을 높인다. 최종 실험 데이터세트는 각 계층별로 1,000명씩 균등하게 무작위 비복원 추출을 수행하여, 총 5,000명의 고유 사용자로 구축하였다. 이는 표본 내 중복 사용자를 방지하여 데이터셋의 무결성을 확보하고, 향후 10-Fold 교차 검증 시 각 Fold마다 500명의 균등한 표본 분포를 보장하기 위함이다.

그림 1은 층화 추출된 최종 데이터세트의 활동량 분포를 보여준다. Y축이 로그 스케일로 표현되었음에도, 5개의 계층이 서로 겹치지 않고 분리되어 있음을 확인할 수 있다. 이는 250 단위의 등간격 구간이 활동량 수준에 따른 계층 구분에 유효함을 보여준다. 'Very High' 등급의 경우 사분위 범위(IQR)와 이상치 분포가 다른 계층에 비해 넓게 나타나는데, 이는 해당 그룹 내 이질성이 존재함을 의미한다. 특히, 'Very High' 등급에서 상위 이상치가 연간 수만 건에 달하는 활동량을 보이는데, 이는 본 논문의 표본이 일반 개발자뿐만 아니라 상위 활동 기여자들까지 포함하고 있음을 나타낸다.

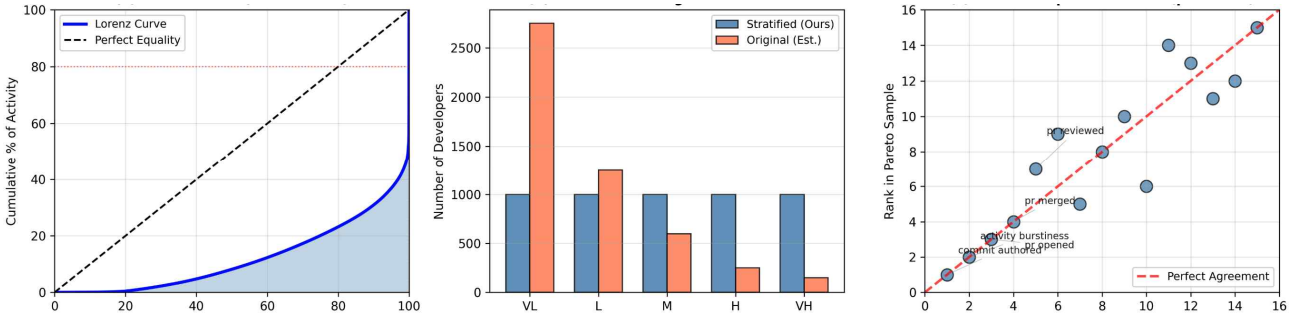

그림 2. 층화 추출 타당성 검증: (a) Lorenz 곡선(Gini = 0.758), (b) 층화 vs 원본 분포, (c) 지표 중요도 순위 비교(ρ = 0.907)
Fig. 2. Sampling strategy validation: (a) Lorenz Curve (Gini = 0.758), (b) Stratified vs original distribution, (c) Feature importance rank comparison (ρ = 0.907)

그림 2는 층화 추출의 타당성을 검증한 결과이다. 그림 2(a)의 Lorenz 곡선은 원본 데이터의 Gini 계수가 0.758로 불균형을 보임을 나타낸다. 그림 2(b)는 층화 추출 후 각 계층별 1,000명씩 균등 분포가 확보되었음을 보여준다. 그림 2(c)는 층화 표본과 원본 모집단 간 지표 중요도 순위의 Spearman 상관계수가 ρ = 0.907로 나타나, 균등 샘플링에도 불구하고 원본의 특성이 보존됨을 확인하였다. 표 2는 수집된 데이터의 범위와 주요 속성을 나타낸다. 커밋 데이터는 모든 브랜치를 대상으로 개발 활동을 누락 없이 수집하였다. 풀 리퀘스트 데이터는 생성부터 병합까지의 전체 수명 주기와 리뷰 이력을 포함하여, 코드 수용률과 병합 소요 시간 산출이 가능하도록 구성하였다. 이슈 데이터는 보고자와 해결자 정보를 모두 포함하고, 저장소 데이터는 개발자가 기여한 고유 저장소 수를 통해 활동 범위를 측정하는 데 활용된다.

표 2. 깃헙 데이터 수집 범위 및 주요 속성
Table 2. Scope of data collection and key attributes

Name	Scope & Definition
Commits (Activity)	Unrestricted PushEvent logs across all branches without numerical limits. Captures comprehensive development persistence beyond standard API constraints.
Pull Requests (Collaboration)	Full lifecycle data (creation to merge) including review interactions. Quantifies collaboration intensity and code acceptance rates using precise timestamps.
Issues (Problem Solving)	Complete event logs covering both reporter and solver roles. Enables precise calculation of resolution cycle times with full historical context.
Repositories (Context)	Count of all unique repositories contributed to. Measures the breadth of technical engagement across different projects.

3-2 개발자 활동 지표

본 논문은 기존 연구의 한계를 보완하기 위해 총 21개의 개발자 활동 지표를 제안한다. 선행 연구들은 정적 빈도 중심의 접근[6],[16],[17], 단순 활동량과 업무 효율성 간의 불일치[18], 협업 맥락 반영 미흡[13], 동적 패턴 고려 부족[14]이라는 공통된 한계를 지닌다. 특히 [18]은 개발자가 인식하는 생산성이 단순 활동량이 아닌 작업의 질과 효율성에 있음을 시사했으나, 기존 지표들은 이러한 실질적 프로세스 효율성을 충분히 반영하지 못하고 있다. 이에 본 논문은 다음과 같은 3단계 설계 제안한다. (1) 기초 활동량 측정을 위해 기존 연구[6],[8]에서 검증된 10개의 정적 지표를 선정하였다. (2) SW 개발 효율성 및 협업 패턴을 정량화 하기 위해 5개의 파생 지표를 추가하였다. (3) 동적 패턴 규명[14],[17]에서 언급된 시계열 정보의 부재를 보완하고자 6개의 동적 지표를 정의하였다. 표 3은 21개 지표를 4가지 카테고리로 분류하고, 각 지표의 산출 방법과 참고문헌을 제시한다. 첫째, 코드 활동(Code Activity)은 개발 활동의 기초 규모와 지속성을 측정한다. 커밋 수(commit_authored)를 통한 양적 평가와 더불어, 커밋 메시지 분석을 통해 자가 버그 수정(self_bugfix_count) 빈도를 산출하여 코드 유지보수 성향을 파악한다. 일일 커밋 시계열(commit_ts)은 개발 활동의 지속성 여부를 동적으로 분석한다. 둘째, 협업(Collaboration)은 9가지 지표를 통해 팀 내 상호작용의 질과 효율성을 측정한다. PR 생성(pr_opened)과 병합(pr_merged) 빈도, 그리고 코드 리뷰 수행 횟수(pr_reviewed)를 통해 협업 참여도를 측정한다. PR 수용률(pr_acceptance_rate)은 활동량 대비 코드 수용도를 정량화하며, 평균 병합 시간(avg_pr_merge_time)은 프로세스 효율성을 나타낸다. 특히 리뷰 기여 비율(review_contribution_ratio)은 기존 연구에서 지적된 협업 맥락의 부재를 보완하여[13], 개인 기여 중심 개발자와 협업 지향적 기여자를 구분하는 척도로 활용된다. 3개의 시계열 지표(pr_opened_ts, pr_merged_ts, review_ts)는 협업 패턴의 시간적 변화를 추적한다. 셋째, 문제 해결(Problem Solving)은 이슈 생성(issue_opened)과 종료(issue_closed) 건수를 통해 문제 발견 및 해결 능력을 측정하고, 이슈의 라이프사이클을 추적하여 과업 완수 능력을 평가한다. 이슈 해결률(issue_resolution_rate)은 문제 해결의 완결성을 나타내며, 평균 이슈 주기(avg_issue_cycle_time)는 이슈 처리 과정의 프로세스 효율성을 측정한다.

표 3. 21개 다중 모달 지표의 분류 및 정의
Table 3. Classification and definition of 21 multi-modal metrics

Category	Metric Name	Type	Computation Method	Ref.
1. Code Activity (Development)	commit_authored	Scalar	Count of all PushEvent occurrences for the developer over the observation period.	[6][16]
	self_bugfix_count	Scalar	Count of PushEvents where the first commit message matches the regex pattern combining action keywords and object keywords within 30 characters.	[8]
	commit_ts	Time-Series	Daily count of PushEvents aggregated for each day in the observation period, producing a 365-element vector.	[15]
2. Collaboration (Review & Merge)	pr_opened	Scalar	Count of PullRequestEvents where the action field equals 'opened'.	[6]
	pr_merged	Scalar	Count of PullRequestEvents where the merged field equals true.	[6]
	pr_reviewed	Scalar	Count of all PullRequestReviewEvent occurrences for the developer.	[7]
	pr_acceptance_rate	Ratio	Ratio of pr_merged to pr_opened. Returns 0 if pr_opened equals 0.	[18]
	avg_pr_merge_time	Scalar	Mean of time differences between merged_at and created_at timestamps for all merged PRs, measured in days.	[16]
	review_contribution_ratio	Ratio	Ratio of collaborative activities (reviews + comments) to total activities (commits + PRs opened + issues opened + reviews + comments). A small constant ($\varepsilon=10^{-9}$) is added to prevent division by zero.	[13]
	pr_opened_ts	Time-Series	Daily count of PullRequestEvents with action 'opened', aggregated for each day.	[15]
	pr_merged_ts	Time-Series	Daily count of PullRequestEvents with merged=true, aggregated for each day.	[15]
	review_ts	Time-Series	Daily count of PullRequestReviewEvents, aggregated for each day.	[15]
3. Problem Solving (Issue Tracking)	issue_opened	Scalar	Count of IssuesEvents where the action field equals 'opened'.	[3][6]
	issue_closed	Scalar	Count of IssuesEvents where the action field equals 'closed'.	[3][6]
	issue_resolution_rate	Ratio	Ratio of issue_closed to issue_opened. Returns 0 if issue_opened equals 0.	[18]
	avg_issue_cycle_time	Scalar	Mean of time differences between closed_at and created_at timestamps for all closed issues, measured in days.	[16]
	issue_opened_ts	Time-Series	Daily count of IssuesEvents with action 'opened', aggregated for each day.	[15]
	issue_closed_ts	Time-Series	Daily count of IssuesEvents with action 'closed', aggregated for each day.	[15]
4. Work Style & Context (Behavioral Profile)	activity_burstiness	Index	Coefficient of Variation (CV) of daily activity counts, calculated as the standard deviation divided by the mean. Higher values indicate irregular (bursty) work patterns.	[15]
	weekend_activity_ratio	Ratio	Ratio of total activities on weekends (Saturday and Sunday) to total activities across all days.	[18]
	unique_repository_count	Scalar	Count of distinct repository IDs across all events for the developer.	[6]

2개의 시계열 지표(issue_opened_ts, issue_closed_ts)는 문제 해결 활동의 시간적 분포 특성을 분석한다. 넷째, 작업 스타일 및 맥락(Work Style & Context)은 시계열 데이터에서 파생된 행동 특성을 규명한다. 이 카테고리는 누적 집계 방식이 동적 패턴을 평탄화한다는 기존 한계[14],[17]에 대응한다. 활동 불규칙성 지수(activity_burstiness)는 변동계수(CV)를 산출함으로써 개발자의 작업 리듬 안정성을 진단한다. 주말 활동 비율(weekend_activity_ratio)은 직업적 루틴과 개인적 열정을 구분하며, 고유 저장소 수(unique_repository_count)는 개발자의 활동 범위를 측정한다.

3-3 분석 방법론

본 논문에서 제안하는 21개 다중 지표의 유효성을 검증하고 최적의 분석 해상도를 도출하기 위해, 다음과 같이 3단계의 통계적 및 기계학습 기반 분석을 수행한다. 첫째, 지표 간 상관관계 분석을 통해 변수 간의 다중공선성을 진단하고 정적 특성을 파악한다. 특히 활동량 지표와 협업 및 효율성 지표 간의 상관관계 히트맵을 분석함으로써, 본 논문에서 제안하는 파생 변수들이 기존의 단순 빈도 지표와 구별되는 독립적인 차원을 형성하고 있는지 검증한다. 둘째, 변수 중요도 분석을 통해 개발자 계층을 구분 짓는 핵심 요소를 식별한다. 이를 위해 앙상블 학습 모델인 RF와 XGB를 활용한다. RF의 지니 불순도 감소량은 변수의 전반적인 분류 기여도를, XGB

의 정보 이득은 모델 예측 정확도 향상에 기여하는 결정적 요인을 측정한다. 이 두 모델의 상호 검증을 통해, 제안된 21개 지표 중 실제 개발자의 숙련도와 작업 패턴을 설명하는 데 있어 가장 영향력이 높은 지표를 선별한다. 셋째, 시계열 해상도 최적화를 수행한다. 임의의 시간 단위 집계로 인한 데이터의 평탄화 및 정보 손실을 방지하기 위해, 통계적 근거에 기반한 최적의 시계열 해상도를 도출한다. 이를 위해 코드 활동, 협업, 문제 해결 카테고리에 포함된 6가지 시계열 지표를 각각 일(Daily), 주(Weekly), 월(Monthly) 단위로 집계한 후, 다음의 세 가지 기준을 통해 비교 평가한다. 시계열 지표 분석을 위해서 변동계수(Coefficient of Variation, CV), 자기상관계수(Lag-1 Autocorrelation Coefficient, r_1), 복합 신뢰도 점수(Composite Reliability Index, CRI) 및 패턴 강도(F_x) 검증을 적용한다. 이때, 시계열의 안정성을 평가하기 위해 변동계수를 활용한다(수식 1). 여기에서 σ와 μ는 각각 시계열 데이터의 표준편차와 평균을 의미한다. 산출된 CV 값이 낮을수록 데이터의 변동성이 적어 해당 시간 해상도에서의 분석 신뢰도가 높음을 시사한다.

$$CV = \frac{\sigma}{\mu} \tag{1}$$

데이터의 민첩성을 측정하기 위해 시차(Lag)가 1일 때의 자기상관계수(r_1)를 사용한다(수식 2). 이때 y_t는 t시점의 관측값, n은 전체 데이터의 수를 나타낸다. r_1은 직전 시점$(t-1)$이 현재 시점(t)에 미치는 영향력을 의미하므로, 이 값이 0에 가까울수록 과거의 상태에 의존하지 않는 독립적이고 민첩한 행동 패턴을 보이는 것으로 판단한다.

$$r_1 = \frac{\sum_{t-2}^{n}(y_t - \bar{y})(y_{t-1} - \bar{y})}{\sum_{t-1}^{n}(y_t - \bar{y})^2} \tag{2}$$

안정성(CV)와 민첩성(r_1)을 동시에 고려하기 위해 제안된 복합 신뢰도 점수(Composite Reliability Index, CRI)를 산출한다(수식 3). 두 지표는 스케일이 다르므로 각각 Min-Max 정규화를 수행한 후, 동일한 가중치(0.5)를 적용하여 합산한다. 도출된 CRI 점수가 0에 수렴할수록 변동성과 자기상관성이 모두 낮은, 가장 이상적인 분석 해상도임을 의미한다.

$$CRI = 0.5 \times Norm(CV) + 0.5 \times Norm(r_1) \tag{3}$$

마지막으로, 선정된 해상도가 단순 노이즈 감소를 넘어 유의미한 규칙성을 포함하는지 검증을 위해 시계열 Y_t에 STL(Seasonal-Trend decomposition using LOESS) 분해를 적용하여 추세(T_t), 계절성(S_t), 잔차(R_t)로 분리한 후 패턴 강도를 분석한다(수식 4).

$$F_X = \max\left(0, 1 - \frac{Var(R_t)}{Var(X_t + R_t)}\right) \tag{4}$$

여기서 규칙적 성분($X_t = T_t + S_t$)은 설명 가능한 규칙적 성분을 의미하며, F_x가 1에 가까울수록 노이즈 대비 뚜렷한 패턴이 관측됨을 의미한다.

Ⅳ. 분석 결과

4-1 21개 지표의 탐색적 데이터 분석

표 4는 본 논문에서 제안한 총 21개 지표 중, 15가지 정적 지표의 기술 통계량을 보인다. 분석 결과, 각 카테고리 별로 뚜렷한 데이터 분포 특성이 관측되었다. 첫째는 데이터의 영과잉(Zero-Inflation) 현상이다. 'Code Activity'(commit_authored) 활동에 대한 중앙값이 372로 나타난 반면 'Collaboration'(pr_reviewed)과 'Problem Solving'(issue_closed)의 중앙값은 '0'을 보였다. 둘째, 분포의 비대칭성을 보인다. 'commit_authored', 'unique_repository_count'의 왜도가 각각 67.55, 69.94로 롱테일 분포를 보인다. 반면 'review_contribution_ratio'는 평균 0.13, 최대값 1.0의 분포를 보였다. 셋째, Work Style지표의 안정성이다. 'weekend_activity_ratio'는 왜도 1.23, 평균 0.21, 중앙값 0.20으로 정규분포에 근접한 형태를 띠고 있음이 확인되었다.

4-2 지표 간 상관관계 및 다중공선성 진단

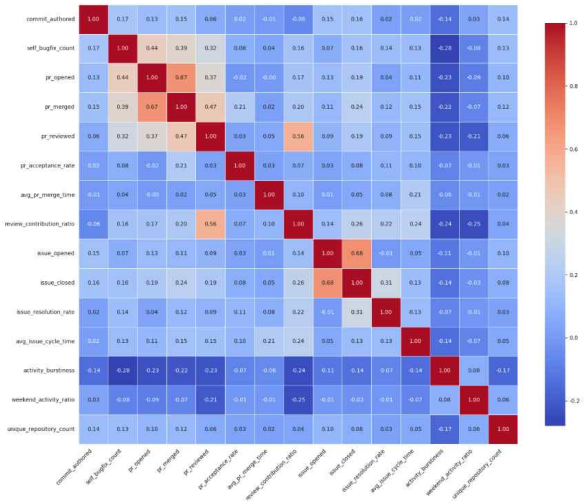

그림 3. 10-Fold 교차 검증을 통해 산출된 다중 모달 지표 간의 평균 피어슨 상관관계 히트맵

Fig. 3. 10-fold cross-validated mean pearson correlation matrix of multi-modal metrics

지표 간의 상호 의존성 및 독립성을 검증하기 위해 피어슨 상관관계 분석을 수행하였다. 그림 3은 산출된 평균 상관계수

행렬을 히트맵으로 시각화 보인다. 분석 결과 대다수의 지표 쌍이 0.3이하의 낮은 상관계수(파란색 영역)를 기록하여 지표 간의 높은 직교성이 확인되었다. 가장 기초적인 양적 지표인 'commit_authored'는 협업 시도 지표인 'pr_opened'와 0.13의 낮은 양의 상관관계를 보였으며, 질적 기여도를 나타내는 'pr_reviewed'(0.06) 및 'pr_acceptance_rate'(0.02)와는 상관계수가 0에 근접하여 선형적 연관성이 거의 나타나지 않았다. 반면, 작업의 시작과 종료를 나타내는 프로세스 지표 간에는 비교적 높은 상관관계가 관측되었다.

4-3 앙상블 학습 기반의 핵심 변수 중요도 분석

본 논문에서 앞서 도출한 15개의 정적 스칼라 지표가 개발자의 숙련도를 결정하는 데 있어 어떠한 변별력을 가지는지 규명하기 위해, 앙상블 학습 기반의 변수 중요도 분석을 수행하였다. 실험 설계 및 학습 방법은 다음과 같다. 첫째, 학습 데이터의 입력 변수로는 표 4에서 정의한 15개의 스칼라 지표를 사용하였으며, 타겟 변수는 개발자의 전체 활동량을 기준으로 5단계(Very Low ~ Very High)계층으로 구분된 숙련도 등급을 정답 레이블로 설정하였다. 둘째, 분석 모델로는 Random Forest(RF)와 XGBoost(XGB)를 채택하였다[20],[21]. 이는 변수 간의 다중공선성을 완화하고 비선형적 상호작용을 효과적으로 포착하기 위함이다. 셋째, 특정 데이터 분포에 의한 과적합을 방지하고 결과의 일반화 성능을 확보하기 위해 10-Fold Cross-Validation을 적용하였다. 그

표 4. 제안된 다중 모달 지표의 기술 통계량(N=5,000)
Table 4. Descriptive statistics of proposed multi-modal metrics

Category	Metric Name	Mean	Median	Std	Min	Max	Skewness
1. Code Activity	commit_authored	1,248.22	372	38,252.83	0	2,659,937	67.55
	self_bugfix_count	58.84	19	173	0	9,505	34.34
2. Collaboration	pr_opened	50.08	10	203.19	0	11,056	39.4
	pr_merged	54.83	6	164.73	0	6,653	17.89
	pr_reviewed	67.31	0	192.76	0	3,611	6.81
	pr_acceptance_rate	1.78	0.58	18.06	0	714	30.13
	avg_pr_merge_time	5.39	0	38.17	0	1,635	24.29
	review_contribution_ratio	0.13	0.02	0.2	0	1	1.68
3. Problem Solving	issue_opened	17.79	0	109.2	0	5,803	34.94
	issue_closed	20.36	0	112.97	0	5,775	31.29
	issue_resolution_rate	1.47	0	10.69	0	448	23.38
	avg_issue_cycle_time	44.14	0.01	138.53	0	2,957	8.39
4. Work Style & Context	activity_burstiness	4.28	2.68	3.93	0.07	19.13	2.04
	weekend_activity_ratio	0.21	0.2	0.18	0	1	1.23
	unique_repository_count	53.38	19	985.8	1	69,496	69.94

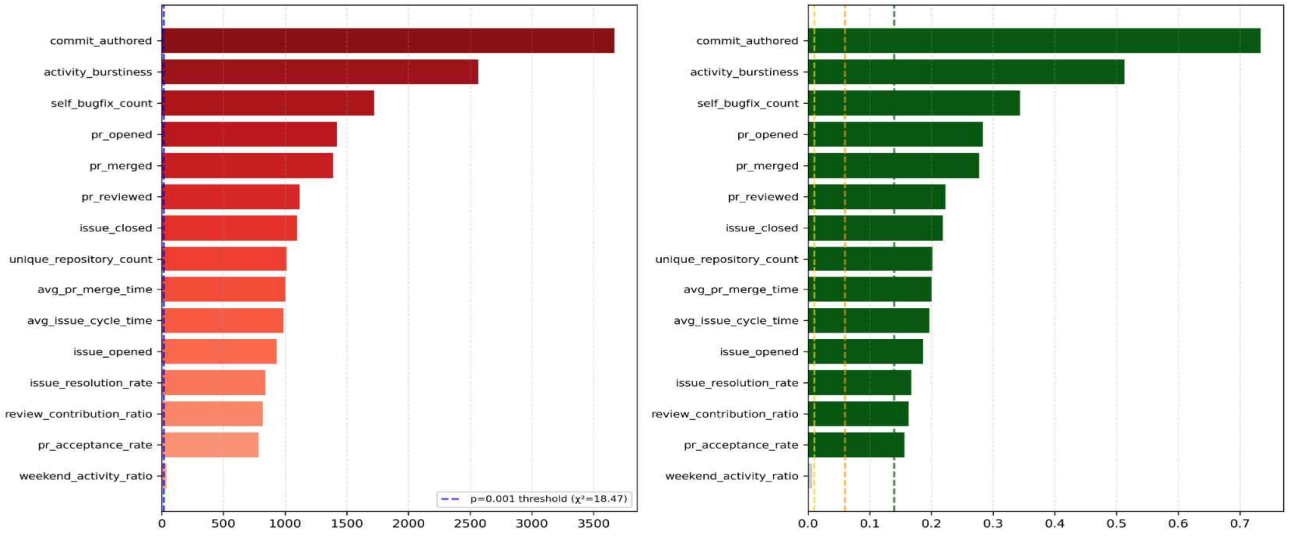

그림 4. Kruskal-Wallis 검정 결과: (a) H-통계량, (b) 효과 크기(η^2)
Fig. 4. Kruskal-Wallis test results: (a) H-statistic, (b) Effect size (η^2)

결과, RF는 정확도 90.34%, F1-Score(Macro) 0.904를 기록하였으며, XGB는 정확도 92.44%, F1-Score(Macro) 0.925를 기록하여, 15개 스칼라 지표만으로 연간 활동량에 따른 5단계 분류를 효과적으로 수행 할 수 있음을 확인하였다. 그림 4는 Kruskal-Wallis 검정을 통해 15개 지표의 계층 간 차이를 검증한 결과이다. Kruskal-Wallis 검정은 세 개 이상의 독립 집단 간 중앙값 차이를 검정하는 비모수 방법이다.

그림 4(a)에서 모든 지표가 $p < 0.001$ 수준에서 유의한 차이를 보였으며, commit_authored($H = 3,500$)가 가장 큰 집단 간 차이를 나타냈다. activity_burstiness($H \approx 2,500$)와 self_bugfix_count($H \approx 1,800$) 순으로 나타났다. 그림 4(b)는 Cohen's 기준에 따른 효과 크기(η^2)를 제시한다. 14개 지표가 Large 효과($\eta^2 > 0.14$)를 보였으며, commit_authored ($\eta^2 = 0.72$)와 activity_burstiness($\eta^2 = 0.52$)가 가장 큰 효과 크기를 보인다. 반면 weekend_activity_ratio($\eta^2 = 0.007$)만이 Negligible 수준으로, 숙련도와의 연관성이 낮음을 검증하였다.

표 5는 SHAP 분석을 통해 각 지표의 전역 중요도와 최대 기여 계층(Peak Tier)을 제시한다. 계층별 주요 기여 지표를 분석하면, Very High 계층은 commit_authored(4.73), issue_opened(0.47), unique_repo_count(0.28)가 주요 기여 지표로 나타나, 최상위 숙련도가 절대적 활동량과 다양한 프로젝트 참여로 구분됨을 보여준다. High 계층은 review_contrib_ratio(1.91), pr_opened(1.02), activity_burstiness(0.33)가 높은 기여를 보여, 중상위 경계에서 협업 참여와 규칙적 작업 패턴이 핵심 변별 요인임을 나타낸다.

표 5. SHAP 기반 지표별 전역 중요도 및 계층별 기여 특성
Table 5. SHAP-based global feature importance and tier-specific contribution patterns

Rank	Metric	SHAP (Global / Peak)	Rank	Metric	SHAP (Global / Peak)
1	commit_authored	3.18 (VH / 4.73)	9	unique_repo_count	0.17 (VH / 0.28)
2	review_contrib_ratio	0.98 (H / 1.91)	10	pr_acceptance_rate	0.10 (VL / 0.22)
3	pr_opened	0.74 (H / 1.02)	11	weekend_act_ratio	0.09 (H / 0.17)
4	pr_merged	0.51 (H / 0.61)	12	avg_issue_cycle	0.09 (VL / 0.19)
5	issue_opened	0.34 (VH / 0.47)	13	self_bugfix_count	0.08 (L / 0.14)
6	issue_closed	0.33 (H / 0.41)	14	avg_pr_merge_time	0.06 (M / 0.09)
7	pr_reviewed	0.22 (L / 0.26)	15	issue_resol_rate	0.03 (M / 0.08)
8	activity_burstiness	0.19 (H / 0.33)	–	–	–

표 6. Random Forest(Gini)와 XGBoost(Gain) 기반 상위 10개 변수 중요도
Table 6. Top 10 feature importances: Random Forest (Gini) vs XGBoost (Gain)

Rank	Random Forest (RF)	Importance	XGBoost (XGB)	Importance
1	commit_authored	0.406	commit_authored	0.318
2	activity_burstiness	0.112	pr_reviewed	0.137
3	pr_opened	0.062	pr_opened	0.126
4	pr_merged	0.058	issue_closed	0.098
5	self_bugfix_count	0.057	review_contribution_ratio	0.091
6	pr_reviewed	0.043	pr_merged	0.068
7	review_contribution_ratio	0.041	issue_opened	0.038
8	unique_repository_count	0.038	activity_burstiness	0.029
9	issue_closed	0.034	self_bugfix_count	0.017
10	weekend_activity_ratio	0.032	avg_issue_cycle_time	0.015

표 7. CRI 기반의 시계열 해상도 최적화 및 지표 유형 분류
Table 7. Optimization of time-series resolution and metric typing based on CRI

Category	Metric	Type	Daily CRI	Weekly CRI	Monthly CRI	Optimal Resolution	Pattern Strength (Fx)
Code Activity	commit_ts	Input	0.938	0.697	0.000	Monthly	0.640
Collaboration	pr_opened_ts	Input	0.983	0.677	0.000	Monthly	0.566
	pr_merged_ts	Output	0.500	0.599	0.500	Daily	0.607
	review_ts	Input	0.500	0.606	0.163	Monthly	0.679
Problem Solving	issue_opened_ts	Input	0.804	0.635	0.000	Monthly	0.427
	issue_closed_ts	Output	0.500	0.586	0.500	Daily	0.341

Low 계층은 pr_reviewed(0.26)와 self_bugfix_count(0.14)가 주요 기여 지표로, 코드 리뷰 참여가 하위 계층 탈출의 신호로 작용함을 시사한다. Very Low 계층은 pr_acceptance_rate(0.22)와 avg_issue_cycle(0.19)가 높은 기여를 보여, 효율성 지표가 최하위 계층 식별에 활용됨을 확인하였다. 한편, weekend_activity_ratio는 전역 중요도 0.09(11위)로 숙련도 예측에 대한 기여도가 낮음을 확인하였다.

표 6은 RF, XGB 두 모델의 변수 중요도를 비교한 결과이다. RF는 불순도 감소량(Gini Importance)를, XGB는 정보 이득(Information Gain)을 기준으로 각 지표의 중요도를 산출한다. 두 모델 모두에서 'commit_authored'가 1위로 평가하였으나, 2위 이하에서 관점 차이가 나타났다. RF는 activity_burstiness(2위, 0.112)를 높게 평가하여 규칙적 작업 패턴을 중시하는 반면, XGB는 pr_reviewed(2위, 0.137)와 pr_opened(3위, 0.126)를 높게 평가하여 협업 참여를 중시한다. 이는 불순도 감소량이 분할 빈도가 높은 지표에, 정보 이득이 단일 분할에서 큰 정보 이득을 제공하는 지표에 높은 중요도를 부여하기 때문이다. 또한, 단순 횟수 지표인 pr_merged보다 파생 지표인 review_contribution_ratio (RF 7위, XGB 5위)와 issue_closed(RF 9위, XGB 4위)가 상위권에 나타나, 본 논문에서 제안한 다중 모달 지표 체계의 유효성을 확인하였다.

4-4 최적 시계열 해상도 도출 및 패턴 분석

표 7은 6가지 주요 시계열 지표에 대한 최적 해상도와 패턴 검증 결과를 제시한다. 분석 결과, 지표의 특성에 따라 최적 해상도가 '월'과 '일'으로 구분되었다. 먼저, 개발자가 주도적으로 수행하는 commit_ts, pr_opened_ts, review_ts, issue_opened_ts는 월간 해상도에서 CRI가 0.000~0.163으로 낮게 나타나 최적 해상도로 선정되었다. 이는 일 단위의 변동이 월 단위로 집계되면서 완화되고, 전반적인 활동 흐름이 주된 패턴으로 나타남을 의미한다. 시차 상관분석에서 review_ts는 월간 해상도에서 Lag 1 상관계수가 0.328로 가장 높았으며, commit_ts도 0.285를 기록하였다. 이는 상위 기여자들이 월 단위로 지속적인 활동 패턴을 유지함을 나타낸다. 반면, pr_merged_ts와 issue_closed_ts는 일간 해상도가 최적으로 나타났다. pr_merged_ts는 Lag 1이 0.314로 높았으나, Lag 2(-0.042)와 Lag 3(-0.113)에서 급격히 감소하였다. 이는 Output 지표가 직전 시점과의 연관성은 높으나, 장기적 추세로 확장되지 않는 단기 집중 특성을 보임을 나타낸다. 숙련도 단계별 시계열 특성을 분석하면, 하위 숙련도 그룹(Very Low, Low)은 commit_ts가 0.640의 높은 패턴 강도를 보여, 월 단위의 꾸준한 코드 작성이 주요 식별 요인으로 작용한다. 상위 숙련도 그룹(High, Very High)은 commit_ts 외에도 review_ts와 pr_merged_ts가 높은 패턴 강도를 보인다. 특히 pr_merged_ts의 패턴 강도(0.607)는 상위 그룹이 활동 투입뿐 아니라 성과 산출에서도 일관된 패턴을 보임을 나타낸다.

V. 고 찰

본 논문의 분석 결과는 깃헙 기반 오픈소스 활동의 구조적 특성부터 개발자의 성장 경로에 이르기까지 다양한 시사점을 제공한다. 먼저, 기술 통계 및 상관관계 분석 결과는 깃헙 오픈소스 활동이 계층적 구조임을 보인다. 협업 및 문제 해결 지표에서 확인된 영과잉 현상은 대다수의 개발자가 코딩에는 참여하지만, 코드 리뷰나 이슈 관리와 같은 유지보수 활동에는 참여하지 못하고 있음을 나타낸다. 이는 소수의 '슈퍼 기여자'가 생태계를 주도한다는 파레토 법칙을 뒷받침한다. 상위 숙련자를 판별하기 위해서는 단순 활동량이 아닌 협업 참여 여부가 핵심 변별 요인임을 알 수 있다. 이러한 맥락에서 관측된 또 하나의 중요한 현상은 양적 활동과 질적 기여의 탈동조화이다. 커밋 수와 코드 수용률 간의 낮은 상관관계는 '코드 생산량'이 곧 '코드 품질'이나 '실질적 기여도'를 담보하지 않음을 보여주며, 커밋 횟수에 의존하는 기존의 단일 지표 중심 평가 방식이 상위 기여자의 다면적인 영향력을 설명하는 데 한계가 있음을 드러낸다. 이는 개발자의 역량을 평가하기 위해서는 다양한 지표를 결합한 비선형 모델링이 필요함을 뜻한다. 한편, 주말 활동 비율의 정규 분포 특성은 개발자들이 직업적 루틴과 개인적 열정 사이에서 균형을 유지하려는 성향을 보여주는 행동 지표로서의 가치를 확인시켜 주었다. 다만, 그림 5(a)에서 보듯이 weekend_activity_ratio는 숙련도 계층 간 분포 차이가 미미하며(η^2 = 0.007), 변수 중요도 분석에서도 Random Forest 기준 10위로 나타나 숙련도 예측에 대한 직접적 기여도는 낮았다. 그러나 그림 5(b)는 동일 숙련도 내에서 주중 집중형(≤Q1) 개발자가 주말 활동형(≥Q3) 대비 2~3배 높은 협업 비율을 보임을 확인하였다. 이는 해당 지표가 역량 예측 변수로서는 제한적이나, "얼마나 잘하는가"가 아닌 "어떻게 일하는가"를 포착하는 행동 프로파일링 도구로서 활용 가능함을 뜻한다. 나아가 앙상블 모델 중요도 분석과 시계열 분석 결과는 고숙련자를 정의하는 기준을 '성실성'과 '협업 역량'이라는 두 축으로 구체화하였다. 특히, 상위 기여자들이 간헐적인 활동 폭발보다는 'activity_burstiness'로 대변되는 꾸준하고 규칙적인 작업 리듬을 유지하며, 동시에 단순한 '개발자'를 넘어 '검토자'로서의 역할을 수행하고 있음을 확인하였다. 또한, 시계열 분석에서 나타난 '노력의 지속'과 '성과의 민첩성'이라는 시간적 이중성은 개발자의 작업 패턴과 성과 창출의 특성을 동시에 보인다. 이러한 결과를 종합하면, 개발자의 성장은 '양적 축적'에서 '질적 확장'으로의 진화 과정으로 해석할 수 있다. 초급 단계에서는 'commit_ts'로 대변되는 절대적인 활동량을 월 단위로 안정화하여 기초적인 성실성을 확보하는 것이 성장의 전제 조건이 된다. 그러나 상위 단계로 진입하기 위해서는 단순 코딩을 넘어, 'review_ts'와 'issue_closed_ts'를 통해 협업 능력과 'pr_merged_ts'에서

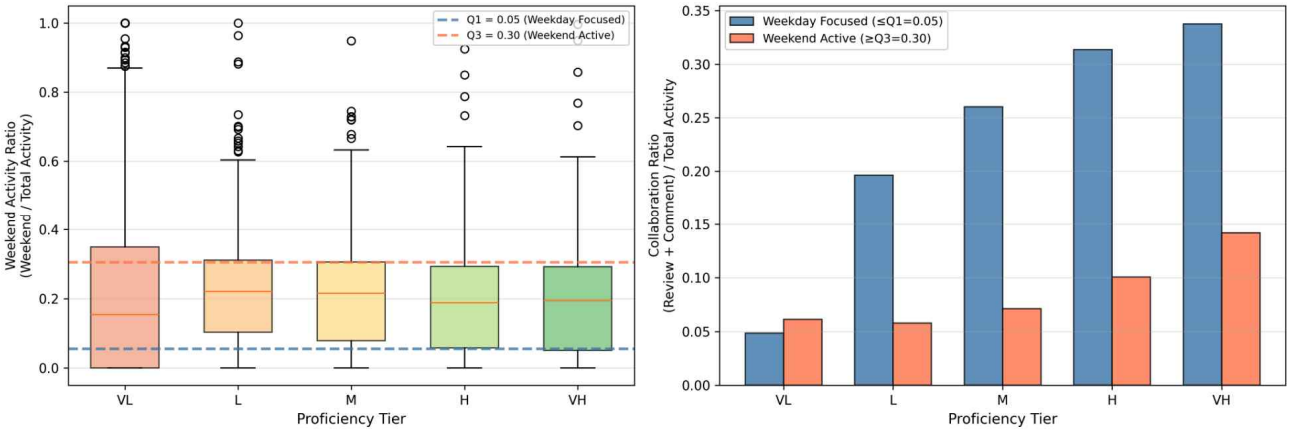

그림 5. 'Weekend_Activity_Ratio ' 분석 (a) 숙련도별 분포 , (b) 개발자 유형별 협업 비율
Fig. 5. 'Weekend_ Activity_ Ratio' analysis: (a) Distribution by tier, (b) Collaboration ratio by developer type

나타나는 문제 해결의 민첩성을 갖추는 질적 전환이 요구된다.

특히, 각 계층별로 주요 기여 지표와 시계열 특성이 차별적으로 나타난다는 점은 개발자를 단일 척도로 서열화하는 기존 방식을 넘어, 고유한 행동 패턴에 기반한 페르소나 유형 분류의 가능성을 시사한다. 본 논문은 다음 같은 한계점을 지닌다. 첫째, 분석 데이터가 깃헙 아카이브에 한정되어 있어, GitLab, Bitbucket 등 다른 플랫폼이나 기업 내부 저장소의 개발자 활동 패턴과는 차이가 있을 수 있다. 특히 기업 환경에서는 보안 정책, 조직 문화, 업무 프로세스 등이 활동 패턴에 영향을 미칠 수 있으므로, 본 논문의 결과를 일반화하는 데 주의가 필요하다. 둘째, 파레토 분포를 보이는 원본 데이터에서 계층별 균등 샘플링을 적용하였다. 이는 극소수인 상위 계층의 특성을 충분히 학습하기 위한 의도적 설계이며, 층화 표본과 원본 간 지표 중요도 순위의 Spearman 상관계수가 $\rho = 0.907$로 나타나 원본 특성이 보존됨을 확인하였으나, 향후 연구에서는 대규모 데이터 기반의 추가 검증이 필요하다. 셋째, 2024년 단일 연도 데이터를 분석하여 시간에 따른 개발자 성장 궤적의 변화를 추적하지 못하였다.

Ⅵ. 결 론

본 논문은 기존의 소프트웨어 저장소 마이닝 연구가 정적 횟수 집계에 의존하여 개발자의 역량을 1차원적인 양적 관점에서만 평가해 온 한계를 극복하고, 동적 프로세스와 시계열 특성을 포괄하는 새로운 분석 프레임워크를 정립하고자 하였다. 이를 위해 개발 활동을 코드 활동, 협업, 문제 해결, 작업 맥락의 4가지 차원으로 세분화한 21가지 다중 모달 지표를 제안하고, 앙상블 학습 및 시계열 해상도 최적화를 통해 그 유효성을 실증적으로 규명하였다. 본 논문을 통해 도출된 핵심 기여는 다음과 같다. 첫째, 상관관계 분석을 통해 활동량 지표와 질적 성과 지표 간의 통계적 독립성을 검증함으로써, 단일 지표 중심 평가의 구조적 한계를 정량적으로 밝혀냈다. 향후 성과 관리 시스템이 양적 생산성과 협업 효율성을 분리하여 접근해야 한다는 당위성을 뒷받침한다. 둘째, 복합 신뢰도 점수 및 패턴 강도 분석을 통해, 주도적 투입 활동은 월간 해상도에서, 결과적 성과는 일간 해상도에서 최적의 패턴을 보이는 이원화된 시계열 분석 전략을 수립하였다. 이는 지표의 성격에 따라 최적화된 관측 주기를 적용해야 함을 증명한 것이다. 셋째, 변수 중요도와 시계열 패턴의 통합 분석을 통해, 개발자 역량이 활동량의 선형적 증가가 아닌 역할의 질적 전환에 의해 결정됨을 실증하였다. 이러한 계층별 차별적 행동 특성은 개발자 페르소나를 정량적으로 유형화하기 위한 실증적 기반을 제공한다. 다만, 본 논문은 깃헙의 공개 저장소 데이터에 국한되어 있어, 기업 내부의 폐쇄적인 개발 환경이나 슬랙 등 비공식적 커뮤니케이션 채널의 데이터를 반영하지 못했다는 한계를 갖는다. 향후 연구에서는 다중 플랫폼 데이터 통합 및 종단 연구 설계를 통해 일반화 가능성을 확장하고,본 논문에서 규명한 계층별 지표 특성과 시계열 패턴을 토대로 개발자 페르소나를 유형화하여 맞춤형 역량 진단 및 추천 시스템을 구축하는 후속 연구를 진행할 계획이다.

참고문헌

[1] G. Gousios and D. Spinellis, “GHTorrent: GitHub’s Data from a Firehose,” in *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*, Zurich: Switzerland, pp. 12-21, June 2012. https://doi.org/10.1109/MSR.2012.6224294

[2] J. Marlow, L. Dabbish, and J. Herbsleb, “Impression Formation in Online Peer Production: Activity Traces and Personal Profiles in GitHub,” in *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, San Antonio: TX, pp. 117-128, February 2013. https://doi.org/10.1145/2441776.2441792

[3] J. Coelho and M. T. Valente, “Why Modern Open Source Projects Fail,” in *Proceedings of the 2017 11th Joint*

Meeting on Foundations of Software Engineering, Paderborn: Germany, pp. 186-196, September 2017. https://doi.org/10.1145/3106237.3106246

[4] J. Tsay, L. Dabbish, and J. Herbsleb, "Let's Talk About It: Evaluating Contributions through Discussion in GitHub," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, Hong Kong: China, pp. 144-154, November 2014. https://doi.org/10.1145/2635868.2635882

[5] M.-A. Storey, A. Zagalsky, F. F. Filho, L. Singer, and D. M. German, "How Social and Communication Channels Shape and Challenge a Participatory Culture in Software Development," *IEEE Transactions on Software Engineering*, Vol. 43, No. 2, pp. 185-204, February 2017. https://doi.org/10.1109/TSE.2016.2584053

[6] A. Şeker, B. Diri, and H. Arslan, "New Developer Metrics for Open Source Software Development Challenges: An Empirical Study of Project Recommendation Systems," *Applied Sciences*, Vol. 11, No. 3, 920, 2021. https://doi.org/10.3390/app11030920

[7] P. Thongtanunam, C. Tantithamthavorn, R. G. Kula, N. Yoshida, H. Iida, and K. Matsumoto, "Who Should Review My Code? A File Location-Based Code-Reviewer Recommendation Approach for Modern Code Review," in *Proceedings of the 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, Montreal: Canada, pp. 141-150, March 2015. https://doi.org/10.1109/SANER.2015.7081824

[8] M. D'Ambros, M. Lanza, and R. Robbes, "An Extensive Comparison of Bug Prediction Approaches," in *Proceedings of the 7th IEEE Working Conference on Mining Software Repositories*, Cape Town: South Africa, pp. 31-41, May 2010. https://doi.org/10.1109/MSR.2010.5463279

[9] F. P. Brooks, The Mythical Man-Month, *Datamation*, Vol. 20, No. 12, pp. 44-52, 1974.

[10] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, "The Promises and Perils of Mining GitHub," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, Hyderabad: India, pp. 92-101, May 2014. https://doi.org/10.1145/2597073.2597074

[11] M. Greiler, M.-A. Storey, and A. Noda, "An Actionable Framework for Understanding and Improving Developer Experience," *IEEE Transactions on Software Engineering*, Vol. 49, No. 4, pp. 1411-1425, April 2022. https://doi.org/10.1109/TSE.2022.3175660

[12] C. Bird, N. Nagappan, B. Murphy, H. Gall, and P. Devanbu, "Don't Touch My Code! Examining the Effects of Ownership on Software Quality," in *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, Szeged: Hungary, pp. 4-14, September 2011. https://doi.org/10.1145/2025113.2025119

[13] O. Kononenko, O. Baysal, L. Guerrouj, Y. Cao, and M. W. Godfrey, "Investigating Code Review Quality: Do People and Participation matter?," in *Proceedings of the 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 111-120, 2015. https://doi.org/10.1109/ICSM.2015.7332457

[14] Y. Chen, Z. Wan, Y. Zhuang, N. Liu, D. Lo, and X. Yang, "Understanding the OSS Communities of Deep Learning Frameworks: A Comparative Case Study of PyTorch and TensorFlow," *ACM Transactions on Software Engineering and Methodology*, Vol. 34, No. 3, February 2025. https://doi.org/10.1145/3705303

[15] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, Hoboken, NJ: John Wiley & Sons, 2015.

[16] R. Kapur and B. Sodhi, "OSS Effort Estimation Using Software Features Similarity and Developer Activity-Based Metrics," *ACM Transactions on Software Engineering and Methodology*, Vol. 31, No. 2, pp. 1-35, 2022. https://doi.org/10.1145/3485819

[17] R. Brasil-Silva and F. L. Siqueira, "Metrics to Quantify Software Developer Experience: A Systematic Mapping," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, Brno: Czech Republic, pp. 1562-1569, 2022. https://dl.acm.org/doi/10.1145/3477314.3507304

[18] A. N. Meyer, T. Fritz, G. C. Murphy, and T. Zimmermann, "Software Developers' Perceptions of Productivity," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, Hong Kong: China, pp. 19-29, November 2014. https://doi.org/10.1145/2635868.2635892

[19] GitHub Archive. GH Repository Archive [Internet]. Available: https://www.gharchive.org/.

[20] L. Breiman, "Random Forests," *The Journal of Machine Learning*, Vol. 45, No. 1, pp. 5-32, October 2001. https://doi.org/10.1023/A:1010933404324

[21] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco: CA, pp. 785-794, August 2016. https://doi.org/10.1145/2939672.2939785

김가은(Gaeun Kim)

2025년 : 국립군산대학교 (학부생)

2022년~현 재: 국립군산대학교 소프트웨어학과 학사과정
※관심분야 : 빅데이터 분석, 자연어처리, 지식그래프 등

이예영(Yeyeong Lee)

2025년 2월 : 국립군산대학교 (학사)

2021년~2025년: 국립군산대학교 소프트웨어학과 학사
※관심분야 : 데이터 분석, OSS, 자연어처리 등

김장원(Jangwon Gim)

2012년 8월 : 고려대학교 (공학박사)

2012년: 고려대학교 컴퓨터·전파·통신공학과(공학박사)
2013년~2017년 3월: 한국과학기술정보연구원 선임연구원
2017년 4월~현 재: 국립군산대학교 소프트웨어학과 교수
※관심분야 : 인공지능, 지식그래프, 지식그래프 임베딩 등