УДК 004.412:519:237.5
DOI https://doi.org/10.32782/2663-5941/2022.6/18

Макарова Л.М.
Національний університет кораблебудування імені адмірала Макарова

Латанська Л.О.
Національний університет кораблебудування імені адмірала Макарова

Давлатова Д.Х.
Національний університет кораблебудування імені адмірала Макарова

Кольцов А.В.
Національний університет кораблебудування імені адмірала Макарова

ДВОФАКТОРНА НЕЛІНІЙНА РЕГРЕСІЙНА МОДЕЛЬ ДЛЯ ОЦІНЮВАННЯ РОЗМІРУ ВЕБ-ЗАСТОСУНКІВ, ЩО СТВОРЮЮТЬСЯ МОВОЮ PYTHON З ВИКОРИСТАННЯМ DJANGO REST FRAMEWORK

В умовах нескінченного технологічного розвитку та прагнень до швидкого й зручного отримання інформації є постійна необхідність у створенні нових програмних продуктів, мобільних додатків та веб-застосунків. Кожен розробник ПЗ має за мету якомога швидше та якісніше виконати поставлені задачі, а також створити щось нове раніше за інших. Тобто компанії прагнуть максимально оптимізувати та автоматизувати як бізнес-процеси, так і роботу своїх співробітників. Нині існує багато мов програмування, кожна з яких має своє спрямування, переваги та недоліки, а також розроблена для вирішення конкретних питань та досягнення певних цілей. Для проведення аналізу було обрано мову програмування Python, оскільки вона поширена серед багатьох відомих компаній і є багатофункціональною мовою, що швидко розвивається, легко масштабується та має дуже зручний, логічний синтаксис, широку та всебічну підтримку від ком'юніті, велику базу фреймворків та готових бібліотек для вирішення найрізноманітніших задач. Проаналізувавши існуючі фреймворки Python, було обрано високорівневий та безкоштовний веб-фреймворк Django, що має доступ до вихідного коду, дозволяє легко та швидко створювати веб-сайти, що є безпечними і підтримуваними. У статті розглянуто проблему оцінювання розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework. Для досягнення мети роботи, а саме підвищення достовірності оцінювання розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework, була побудована двофакторна нелінійна регресійна модель. Було відібрано дані з 71 проекту з відкритим вихідним кодом, проведено аналіз на викиди та використано десятковий логарифм в якості нормалізуючого перетворення для побудови лінійної регресійної моделі. На основі нормалізованих даних та зворотного нормалізуючого перетворення побудовано нелінійну регресійну модель для вихідного негаусівського набору даних (кількості рядків коду, кількості класів та кількості методів). Проведені перевірка якості та порівняння з лінійною двофакторною моделлю показали, що за всіма параметрами якості побудована двофакторна нелінійна регресійна модель є кращою.

Ключові слова: *нелінійна регресійна модель, логарифмічне перетворення, веб-застосунки, Python, Django Rest Framework.*

Постановка проблеми. Одним з найвагоміших факторів в управлінні процесом розробки ПЗ є розмір програми. Від ефективності оцінювання розміру програм залежить успіх або невдача проекту на ранньому етапі розробки, тому проблема отримання ефективної системи оцінювання кількості строк коду – це важливе завдання, що вимагає удосконалення існуючих методів [1].

Важливим етапом аналізу складної системи є побудова математичної моделі, що здійснюється на підставі змістовної моделі і складається з вихідних, внутрішніх та результуючих параметрів. Однією з умов побудови якісної адекватної моделі є надійність вхідних даних, тому потрібна первинна обробка експериментальних даних із метою виявлення ненадійних вимірів [2].

Виявлення аномальних значень відноситься до проблеми пошуку зразків у даних, які не відповідають очікуваній поведінці.

Під час проведення регресійного аналізу дослідники, зазвичай, віддають перевагу побудові лінійних моделей, які легко оцінюються, і результати яких чітко інтерпретуються. Однак дані рідко носять абсолютно лінійний характер, тому для їх адекватного опису доводиться вдаватися до побудови нелінійних залежностей [3].

TIOBE Software представила рейтинг найпопулярніших мов програмування на серпень 2022 року [4]. Порівняно з минулим роком Python додав до популярності 3,56%, перемістившись з другого на перше місце з показником 15,42%. Це найвищий показник популярності цієї мови програмування за час існування рейтингу. Найнижчий був зафіксований у 2003 році (0,97%), коли Python посідав 13 місце в рейтингу [5].

Таким чином, побудова нелінійних регресійних моделей із використанням нормалізуючих перетворень для підвищення достовірності оцінювання розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework, і створення на їх основі інформаційної технології переробки інформації є актуальною та має практичну цінність.

Аналіз останніх досліджень і публікацій. Дослідження щодо розробки ПЗ та оцінювання вартості були численними та різноманітними. Тема все ще дуже актуальна, про що свідчать численні роботи, наявні в літературі [6–8]. Дослідники ретельно досліджували цю тему як з точки зору підходів до оцінювання (регресія, аналогія, експертне судження, функціональні точки, моделювання тощо), так і дослідницького підходу (теорія, опитування, експеримент, тематичне дослідження, моделювання тощо). Дослідження проводяться як у промисловому, так і в академічному контексті. Найбільш часто використовується підхід до оцінювання, заснований на регресії [9].

Оцінювання розміру ПЗ розглядається як фундаментальна діяльність з точки зору завдань управління розробкою ПЗ. Планування робіт і подальше оцінювання часу та зусиль прогнозуються на основі розміру ПЗ. Кількість рядків коду (LOC) і функціональність ПЗ є двома показниками, які часто використовуються для визначення розміру програми. LOC – це прямий показник, який легко підрахувати і яким легко маніпулювати. Існує багато дискусій щодо використання LOC в оцінюванні розміру ПЗ. Критики базуються на аргументах, що кількість рядків коду залежить від мови програмування та вимагає деталей, які можливо буде складно оцінити до завершення роботи над проектом .

Що стосується валідації методів оцінювання, переважна більшість дослідницьких підходів базується на використанні історичних даних. Крім того, більшість досліджень стосується промислового контексту.

Зібрати інформацію про реальні промислові проекти дуже складно. Більшість компаній не передають дані про свої проекти. Технології змінюються настільки швидко, що стало важко створити стандартну модель для оцінювання. Різні компанії використовують різні підходи, характер і розмір проектів різний.

Звужуючи тему до веб-додатків, одним із перших дослідників, які ввели метрику розміру для вимірювання веб-додатків шляхом статистичного аналізу основних характеристик веб-сайтів, був Тім Брей [10]. Спочатку моделі, що використовувалися для оцінювання веб-додатків були такими ж, як ті, що використовувалися для загальних програмних рішень.

Одним із перших науковців, які представили метод, спеціально розроблений для веб-додатків, був Рейфер через метрику WO та модель WEBMO [11]. Ряд дослідницьких робіт з оцінювання як розміру, так і трудомісткості веб-додатків також були проведені Мендесом та його співробітниками [12]. Однак наразі не існує єдиної моделі, здатної адекватно оцінити розмір веб-додатку.

Розробка веб-додатків стає дуже популярною на основі концептуальних моделей, і ці моделі можна коригувати та переглядати в будь-який час у процесі розробки додатків. Прогнозування розміру веб-додатків з високою достовірністю все ще є проблемою.

Постановка завдання. Метою роботи є підвищення достовірності оцінювання розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework. Щоб досягнути поставленої мети потрібно вирішити такі завдання: виконати аналіз та порівняння моделей оцінювання розміру ПЗ, що вже існують; зібрати проекти з відкритим вихідним кодом зі створення веб-застосунків мовою Python з використанням Django Rest Framework та обрати ті з них, що можуть бути використані для створення регресійної моделі; визначити необхідні метрики з кожного проекту та перевірити отримані дані на викиди; виконати нормалізацію даних, побудувати лінійне рівняння регресії та довірчий інтервал для нього; на основі нормалізованих даних

виконати побудову нелінійного рівняння регресії та довірчого інтервалу.

Виклад основного матеріалу дослідження. Нормальний (або гаусівський) розподіл заклав основу для багатьох статистичних методів, не останнім з яких є класична лінійна регресія.

Практика перетворення даних, щоб вони відповідали нормальному закону за допомогою певного нормалізуючого перетворення, широко використовується через її простоту, часто ця техніка працює добре як швидке та прагматичне рішення. Спотвореність даних впливає на їх оцінювання, тому дані зі значним відхиленням від нормальності можуть бути більш придатними для аналізу після відповідного перетворення.

Тож було обрано десятковий логарифм в якості нормалізуючого перетворення для побудови регресійної моделі для оцінювання кількості рядків коду веб-застосунків, що створюються мовою Python з використанням Django Rest Framework. У логарифмічному перетворенні кожна змінна x замінюється на log(x) з основою 10:

$$z = \lg(x), \quad (1)$$

і зворотним до нього є наступне перетворення:

$$x = 10^z. \quad (2)$$

Багатофакторна лінійна регресійна модель в загальному випадку має вигляд:

$$Z_Y = \hat{Z}_Y + \varepsilon = b_0 + b_1 Z_{X_1} + b_2 Z_{X_2} + \cdots + b_k Z_{X_k} + \varepsilon, \quad (3)$$

де ε – це випадкова похибка, розподіл якої підпорядковується нормальному закону та має математичне очікування рівне нулю.

Нелінійна регресійна модель будується за зворотним перетворенням (2) та лінійною регресійною моделлю (3):

$$Y = 10^{\varepsilon + b_0} \mathrm{X}_1^{b_1} \mathrm{X}_2^{b_2} \cdot \ldots \cdot \mathrm{X}_k^{b_k}. \quad (4)$$

Для побудови двофакторної нелінійної регресійної моделі для оцінювання розміру вебзастосунків, що створюються мовою Python з використанням Django Rest Framework з репозиторію GitHub [13], було зібрано дані 71 проекту з відкритим кодом.

Процес збору метрик проектів, написаних мовою Python, є досить не простою задачею, оскільки фактично немає такого ПЗ, що надає повну інформацію про проект. Завдяки розширенню до PyCharm Statistic [14] було зібрано такі метрики як: кількість файлів з розширенням .py; кількість рядків коду без коментарів та порожніх рядків; кількість порожніх рядків; кількість рядків з коментарями та загальна кількість рядків. Також мовою Python було додатково написано скрипт, що підраховує кількість класів та методів в кожному з обраних проектів. З цих даних було прийнято рішення обрати наступні фактори:

– загальна кількість рядків коду проекту в тисячах (kilolines of code, KLOC), в якості значення залежної змінної Y;

– кількість класів проекту в якості першої незалежної змінної X_1;

– кількість методів проекту в якості другої незалежної змінної X_2.

Перш за все необхідно виконати первинний аналіз даних, а саме: важливо переконатися, що вибірка не містить викидів. Також важливо виконати перевірку предикторів на наявність мультиколінеарності. У регресійній моделі лінійна залежність між двома або більше незалежними змінними демонструє присутність мультиколінеарності, що безперечно є негативним явищем і не дозволяє здійснити оцінювання окремого впливу кожного фактору на залежну змінну, оскільки маємо пов'язаність факторів між собою або високий ступінь кореляції.

Серед майбутніх предикторів в моделі множинної лінійної регресії наявність мультиколінеарності визначимо за коефіцієнтами впливу дисперсії (VIFs). Для лінійної моделі множинної регресії з k-предикторами X_i, i=1,...,k, VIFs – це діагональні елементи оберненої кореляційної матриці kxk k-предикторів. Якщо значення VIFs більше за 10 то дані мають проблеми з мультиколінеарністю, а у разі, коли значення VIFs знаходяться у межах від 1 до 5, то мультиколінеарності немає [15]. Виконавши перевірку впевнились, що обрані метрики, а саме: кількість класів X_1 та кількість методів X_2 проекту не мультиколінеарні (відповідні діагональні елементи мають значення 3,1), тому можемо використовувати їх в якості двох незалежних змінних.

Для перевірки даних на викиди було використано квадрат відстані Махаланобіса і для його розрахунку побудовано коваріаційну матрицю згідно з [16, 17]. При m = 3, α = 0,05 та кількості значень n = 71 отримали квантілі розподілів Пірсона та Фішера відповідно χ^2 = 7,81 та F = 2,74.

В табл. 1 наведені вихідні дані, нормалізовані дані, а також значення квадрату відстані Махаланобіса, та додатковий *TS* (Test Statictic) для розрахунку критерію Фішера (фрагмент даних).

Отримані викиди у кількості 9 видаляємо із первинного набору даних і повторно виконуємо перевірку на викиди для набору даних із 62 проектів, в результаті якої викидів виявлено не було.

Таблиця 1

Фрагмент обробки та дослідження вихідних емпіричних даних

№	Y	X_1	X_2	Z_y	Z_{x1}	Z_{x2}	d_i^2	*TS* для d_i^2
…								
10	0,295	2	7	-0,5302	0,3010	0,8451	**14,42**	**4,60**
11	0,355	5	15	-0,4498	0,6990	1,1761	**8,24**	2,63
12	0,573	25	24	-0,2418	1,3979	1,3802	6,43	2,05
13	28,757	473	1987	1,4587	2,6749	3,2982	**8,05**	2,57
14	13,389	344	465	1,1267	2,5366	2,6675	1,67	0,53
15	4,977	90	170	0,6970	1,9542	2,2304	0,21	0,07
16	0,926	16	96	-0,0334	1,2041	1,9823	1,38	0,44
17	3,393	67	146	0,5306	1,8261	2,1644	0,12	0,04
18	4,503	313	236	0,6535	2,4955	2,3729	1,38	0,44
19	4,037	87	182	0,6061	1,9395	2,2601	0,08	0,02
20	0,529	12	26	-0,2765	1,0792	1,4150	5,08	1,62
21	6,181	131	216	0,7911	2,1173	2,3345	0,34	0,11
22	0,888	4	31	-0,0516	0,6021	1,4914	4,76	1,52
23	27,301	723	592	1,4362	2,8591	2,7723	3,45	1,10
24	1,227	74	68	0,0888	1,8692	1,8325	2,28	0,73
25	306,187	1974	17285	2,4860	3,2953	4,2377	**29,23**	**9,33**
…								

За допомогою методу найменших квадратів було знайдено коефіцієнти лінійного рівняння регресії: b_0=–1,6796, b_1=0,0534, b_2=0,9571 та з використанням формули (3) побудовано лінійну регресійну модель для нормалізованих даних:

$$Z_Y = -1{,}6796 + 0{,}0534Z_{X_1} + 0{,}9571Z_{X_2} + \varepsilon.$$

Перевіривши залишки ε на відповідність нормальному закону розподілу за допомогою критерію згоди χ^2 Пірсона, підтвердили нормальний закон розподілу ($\chi^2_{кр} = 9{,}49$, $\chi^2 = 0{,}22$), тобто лінійна регресійна модель була побудована обґрунтовано. Далі за рівнянням (4) можемо перейти до нелінійної регресійної моделі:

$$Y = 10^{\varepsilon - 1{,}6796} X_1^{0{,}0534} X_2^{0{,}9571}.$$

Для нелінійної регресійної моделі були отримані значення коефіцієнту детермінації R^2 = 0,8936, середньої величини відносної похибки *MMRE* = 0,2342 і рівню прогнозування *PRED*(0,25) = 0,6290.

Було проведено порівняння з лінійною двофакторною моделлю (R^2 = 0,8555, *MMRE* = 0,3352, *PRED*(0,25) = 0,4776), яке показало, що за всіма параметрами якості двофакторна нелінійна регресійна модель є кращою: за параметром R^2 на 4,45%, за параметром *MMRE* на 30,13%, за параметром *PRED*(0,25) на 31,70%.

Висновки. У статті було розглянуто побудову двофакторної нелінійної регресійної моделі для оцінювання розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework на основі логарифмічного перетворення, завдяки чому було підвищено достовірність оцінювання розміру веб-застосунків. Отриману модель, що розроблена на основі незалежних змінних, а саме значення кількості класів X_1 та кількості методів X_2 проекту, можна використовувати для виконання прогнозу значень результативного показника *Y*.

В подальшому планується побудова довірчого інтервалу та інтервалу прогнозування нелінійної регресії, а також розробка програмного забезпечення для прогнозування розміру веб-застосунків, що створюються мовою Python з використанням Django Rest Framework.

Список літератури:

1. Разработка программного обеспечения – Краткое руководство. URL: https://coderlessons.com/tutorials/akademicheskii/programmnaia-inzheneriia/razrabotka-programmnogo-obespecheniia-kratkoe-rukovodstvo (дата звернення: 21.09.2022).

2. Павленко П.М. Основи математичного моделювання систем і процесів : навч. посіб. Київ : Книжкове вид-во НАУ, 2013. 201 с.

3. Adam Hayes. Nonlinearity. URL: https://www.investopedia.com/terms/n/nonlinearity.asp (дата звернення: 21.09.2022).

4. The software quality company TIOBE. TIOBE Index for September 2022. URL: https://www.tiobe.com/tiobe-index/ (дата звернення: 21.09.2022).

5. The software quality company TIOBE. The Python Programming Language. URL: https://www.tiobe.com/tiobe-index/python/ (дата звернення: 21.09.2022).

6. Boehm B., Abts C., Chulani S. Software development cost estimation approaches – A survey. *Annals of Software Engineering*. 2000. Vol. 10. P. 177-205.

7. Kemerer C.F. An empirical validation of software cost estimation models. *Communications of the ACM*. 1987. Vol. 30, № 5. P. 416-429.

8. Reifer D.J. Estimating Web Development Costs : There Are Differences. *CROSSTALK, The Journal of Defense Software Engineering*. 2002. P. 13-17.

9. Boehm B.W. Software engineering economicsю. New Jersey : Prentice Hall, 1981. 767 p.

10. Bray T. Measuring the Web. *Computer Networks and ISDN Systems*. 1996. Vol. 28, № 7-11. P. 993-1005.

11. Reifer D.J. Web development: estimating quick-to-market software. *IEEE Software*. 2000. Vol. 17, № 6. P. 57-64.

12. Kitchenham B., Mendes E. Why Comparative Effort Prediction Studies may be Invalid. *International Conference on Predictor Models in Software Engineering (PROMISE)*. New York, 2009. P. 1-5.

13. A code hosting platform for version control and collaboration GitHub. URL: https://github.com/ (дата звернення: 10.09.2022).

14. Plugin Statistic for PyCharm. URL: https://plugins.jetbrains.com/plugin/4509-statistic (дата звернення: 15.09.2022).

15. Chatterjee S., Hadi A. Regression Analysis by Example. Fifth edition. New York: A John Wiley & Sons, Inc., Publication, 2012. 421 p.

16. Prykhodko, S., Prykhodko, N., Makarova, L., Pugachenko, K., Detecting outliers in multivariate non-Gaussian data on the basis of normalizing transformations, in: *Proceedings of the 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, Kyiv, Ukraine, 2017, 846-849. https://doi.org/10.1109/UKRCON.2017.8100366

17. Prykhodko S., Prykhodko N., Makarova L., Pukhalevych A. Outlier Detection in Non-Linear Regression Analysis Based on the Normalizing Transformations. *Proceedings of the 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), IEEE.* (Lviv-Slavske, 2020). P. 407-410.

Makarova L.M., Latanska L.O., Davlatova D.H., Koltsov A.V. A TWO-FACTOR NONLINEAR REGRESSION MODEL FOR ESTIMATING THE SIZE OF WEB APPLICATIONS DEVELOPED IN PYTHON USING DJANGO REST FRAMEWORK

In the conditions of endless technological development and aspirations for quick and convenient access to information, there is a constant need to create new software, mobile applications and web applications. Every software developer has the goal of completing tasks as quickly and qualitatively as possible, as well as creating something new before others. That is, companies strive to optimize and automate both business processes and the work of their employees as much as possible. Today, there are many programming languages, each of which has its own direction, advantages and disadvantages, and is also designed to solve specific problems and achieve certain goals. The Python programming language was chosen for the analysis because it is common among many well-known companies and is a multi-functional language that develops quickly, is easily scalable and has a very convenient, logical syntax, wide and comprehensive support from the community, a large base of frameworks and ready-made libraries to solve a wide variety of problems. After analyzing the existing frameworks for the Python, the high-level and free Django web framework was chosen, which has access to the source code and allows you to easily and quickly create secure and maintainable websites. The article discusses the problem of estimating the size of web applications created in Python using the Django Rest Framework. To achieve the goal of the work, namely to increase the reliability of the estimation of the size of web applications created in the Python language using the Django Rest Framework, a two-factor nonlinear regression model was built. Data from 71 open-source projects were selected, outlier analysis was performed, and the decimal logarithm was used as a normalization transformation to construct a linear regression model. Based on the normalized data and the inverse normalizing transformation, a nonlinear regression model was built for the original non-Gaussian data set (number of lines of code, number of classes, and number of methods). The conducted quality check and comparison with the linear two-factor model showed that the constructed two-factor nonlinear regression model is better for all quality parameters.

Key words: *nonlinear regression model, logarithmic transformation, web applications, Python, Django Rest Framework.*