**DOI: 10.58254/viti.7.2025.19.213**

УДК 004.8

Фомкін Д. В. ORCID: 0000-0003-0128-9355 (ВІТІ ім. Героїв Крут)
Шемендюк О. В. ORCID: 0000-0002-5594-2973 (ВІТІ ім. Героїв Крут)
Ткач В. О. ORCID: 0000-0003-0013-7368 (ВІТІ ім. Героїв Крут)
Балан Д. Д. ORCID: 0000-0002-6714-8718 (ВІТІ ім. Героїв Крут)

# АНАЛІЗ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ ДЛЯ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО КОДУ В ФАЙЛАХ PDF: ВИБІР ОПТИМАЛЬНОЇ МОДЕЛІ

*В останні роки спостерігається різке зростання складних кібератак із використанням зловмисно закодованих документів, оскільки збільшуються об'єми інформації, яка передається. Файли, які виконуються, прикріплені до електронних листів або вебсторінок, можуть бути небезпечними, про що відомо більшості користувачів Інтернету. Тим не менш, документи є корисним інструментом для розповсюдження шкідливого програмного забезпечення, оскільки люди не знають про них. Поширення зловмисно закодованих документів зі збільшенням кількості передачі інформації призвело до зростання кількості складних атак. Файли формату портативних документів (Portable Document Format, далі – PDF) стали основним вектором атаки шкідливого програмного коду завдяки їх адаптивності та широкому використанню. Виявлення шкідливого програмного коду у файлах PDF є складним завданням через його здатність включати різні шкідливі елементи, такі як вбудовані сценарії, експлойти та шкідливі URL-адреси. У цій статті проведено аналіз у підходах машинного навчання та представлено порівняльний аналіз методів машинного навчання (далі – ML), включаючи Naive Bayes (далі – NB), K-Near Neighbor (далі – KNN), Average One Dependency Estimator (далі – A1DE), Random Forest (далі – RF) та Support Vector Machine (далі – SVM) для виявлення шкідливих PDF-файлів. Це дослідження базується на різних типах критеріїв тестування, з відсотковим поділом та K-кратною перехресною перевіркою, для отримання більш об'єктивних результатів. Результати дослідження підкреслюють ефективність моделей машинного навчання у точному виявленні шкідливого програмного коду в PDF-файлах, визначать найбільш ефективні методи та надають уявлення про розробку надійних систем для захисту від зловмисних дій. Подальші напрямки досліджень в області виявлення шкідливих PDF-файлів повинні бути зосереджені на дослідженні методів глибокого навчання, що може підвищити продуктивність і точність моделей.*

***Ключові слова:*** *кібербезпека, шкідливий програмний код, PDF-файл, машинне навчання, JavaScript.*

***D. Fomkin, O. Shemendyk, V. Tkach, D. Balan. Comparative analysis of machine learning models for malware detection in PDF: evaluation of training and testing criteria***

*In recent years, there has been a sharp increase in sophisticated cyberattacks using maliciously encrypted documents as the amount of information transmitted increases. Executable files attached to emails or web pages can be dangerous, as most internet users are aware of. Nevertheless, documents are a useful tool for spreading malware because people are unaware of them. The spread of maliciously encrypted documents with an increase in the number of information transfers has led to an increase in the number of sophisticated attacks. Portable Document Format (PDF) files have become the main attack vector of malicious code due to their adaptability and widespread use. Detecting malicious code in PDF files is challenging due to its ability to include various malicious elements, such as embedded scripts, exploits and malicious URLs. This article analyzes machine learning approaches and provides a comparative analysis of machine learning (ML) methods, including Naive Bayes (NB), K-Near Neighbor (KNN), Average One Dependency Estimator (A1DE), Random Forest (RF), and Support Vector Machine (SVM) to detect malicious PDF files. This study is based on different types of testing criteria, with percentage division and K-fold cross-validation, to obtain more objective results. The results of the study highlight the effectiveness of machine learning models in accurately detecting malicious code in PDF files, identify the most effective methods, and provide insight into the development of robust systems to protect against malicious activities. Further research in the field of detecting malicious PDF files should focus on investigating deep learning methods that can improve the performance and accuracy of models.*

***Keywords:*** *cybersecurity, malware, PDF, machine learning, JavaScript.*

**Постановка задачі.** В останні роки спостерігається різке зростання складних кібератак з використанням зловмисно закодованих документів, оскільки збільшуються об'єми передачі інформації. Основним вектором кібератак з використанням програмного коду, які були виявлені, є файли формату PDF, який легко адаптується порівняно з іншими форматами

документів. Шкідливі PDF-файли часто містять JavaScript або бінарні скрипти, які використовують слабкі сторони безпеки для виконання зловмисних дій [1]. Adobe Acrobat Reader виявив величезну кількість вразливостей в 2017 році. Кожне програмне забезпечення для читання має певні недоліки, і шкідливий PDF-файл може ними скористатися [2]. Поява більш сучасних технологій, що не виконуються, і методів атак на основі файлів зробила захист PDF більш складним, оскільки зловмисні PDF-файли зазвичай досліджують вектори зараження у ворожих обставинах [3; 4].

Вкрай важливе виявлення шкідливого програмного коду в PDF з кількох причин. По-перше, це допомагає захистити користувачів від несанкціонованого доступу або запуску небезпечних файлів, захищаючи їх від потенційної шкоди. По-друге, вразливості в програмному забезпеченні для читання PDF та інших програмах можуть бути використані за допомогою шкідливих PDF-файлів, що робить критично важливим виявлення та запобігання таких кібератак. По-третє, PDF-файли часто містять конфіденційну інформацію, яка може бути викрадена, що призводить до фінансових втрат, витоку даних або крадіжки особистих даних.

Отже, існує нагальна потреба у розробці ефективних методів захисту від складних атак з використанням шкідливого програмного коду у PDF-файлах.

**Аналіз останніх досліджень і публікацій.** Використовуючи різні моделі машинного навчання, було проведено кілька різновидів досліджень з метою виявлення шкідливого програмного забезпечення в PDF.

Ах Реум Канг та інші науковці описали використання PDF у 2019 році [5]. Вони дали ретельний аналіз структури та вмісту JavaScript у PDF-файлі з вбудованим XML. Потім були створені різноманітні функції, такі як методи кодування конфігурації матеріалу та змінних, таких як розмір файлу, ключові слова, версії та рядки, які можуть прочитати JavaScript.

Підходи до навчання стійких класифікаторів шкідливих програм PDF, що використовують спостережувальні функції стійкості, були описані Ю. Чен та іншими науковцями у 2019 році [6]. У роботі досліджено, що класифікатор повинен ідентифікувати шкідливий програмний код в PDF-файлі як шкідливий, вони демонструють, як точно оцінити найгіршу поведінку класифікатора шкідливого програмного коду щодо певних властивостей стійкості.

Наукові роботи Wepawet [7], Laskov та інші [8] були зосереджені на небезпечному JavaScript, який був представлений у шкідливому програмному забезпеченні формату Portable Document Format. Тут підходи машинного навчання були використані для розробки класифікаторів шкідливих програмних кодів у PDF.

Ес Джей Хітан та інші [9] запропонували атрибути, виходячи з лексичних особливостей скриптів JavaScript, а також функцій, констант, об'єктів, методів і ключових слів. Джей Занг та інші [10] об'єднали кількість об'єктів JavaScript, кількість сторінок та дані фільтрації потоку зі структурою PDF, характеристиками сутностей, інформацією метаданих та статистикою вмісту.

Після виявлення того, що шкідливий JavaScript функціонує інакше, ніж законний код JavaScript, Д. Лю, Х. Ван і А. Ставру [12] запропонували контекстно-залежний підхід. Цей підхід передбачає використання оригінального коду JavaScript як вхідних даних для функції "eval" для відкриття PDF-файлу, уважно відстежуючи будь-яку незвичайну поведінку на основі наведених інструкцій.

За даними Еррера-Сілва Я. А. й Ернандес-Альварес М. [13], кібератаки з використанням програм-вимагачів зросли за останні десять років, викликавши велике занепокоєння серед організацій.

Дхаларія М. і Гандотра Е. представили гібридний метод виявлення і класифікації шкідливих програм Android [14]. Запропонований метод поєднує методи статичного та

динамічного аналізу для ефективного виявлення шкідливих додатків та класифікації їх у різні сімейства шкідливих програм. Автори навчають моделі машинного навчання для виявлення шкідливих програм і класифікації сімейства, використовуючи функції, взяті як зі статичної, так і з динамічної поведінки додатків Android. Експериментальні результати демонструють ефективність гібридного підходу в точному виявленні та класифікації шкідливих програм Android, тим самим сприяючи сфері мобільної безпеки та допомагаючи запобігати шкідливим діям на пристроях Android.

Всі розглянуті вище дослідження мають один спільний недолік – обмеження сфери використання за типом загроз.

Таким чином, **основною метою** цього дослідження є розробка універсальної моделі виявлення шкідливого програмного коду, здатної захистити системи від шкідливих дій, спричинених шкідливими PDF-файлами, що необхідно для оперативного попередження та реагування на кіберінциденти в мережі, шляхом аналізу методів машинного навчання для виявлення шкідливого програмного коду в PDF-файлах.

**Виклад основного матеріалу.** У цьому дослідженні представлено аналіз деяких моделей машинного навчання, а саме: Average One Dependency Estimator (A1DE), K-Nearest Neighbor (KNN), Support Vector Machine (SVM) [15], Naive Bayes (NB) та Random Forest (RF) [16]. Це дослідження зосереджено на порівнянні різних моделей машинного навчання та критеріїв навчання моделей для пошуку кращого рішення для виявлення шкідливого програмного коду в PDF-файлах. Моделі машинного навчання включають A1DE, NB, KNN, RF та SVM, тоді як критерії навчання включають процентне розділення з 70 % та 30 % для навчання та тестування відповідно, а також 10-кратну перехресну перевірку. Загальна схема дослідження представлена на рисунку 1.

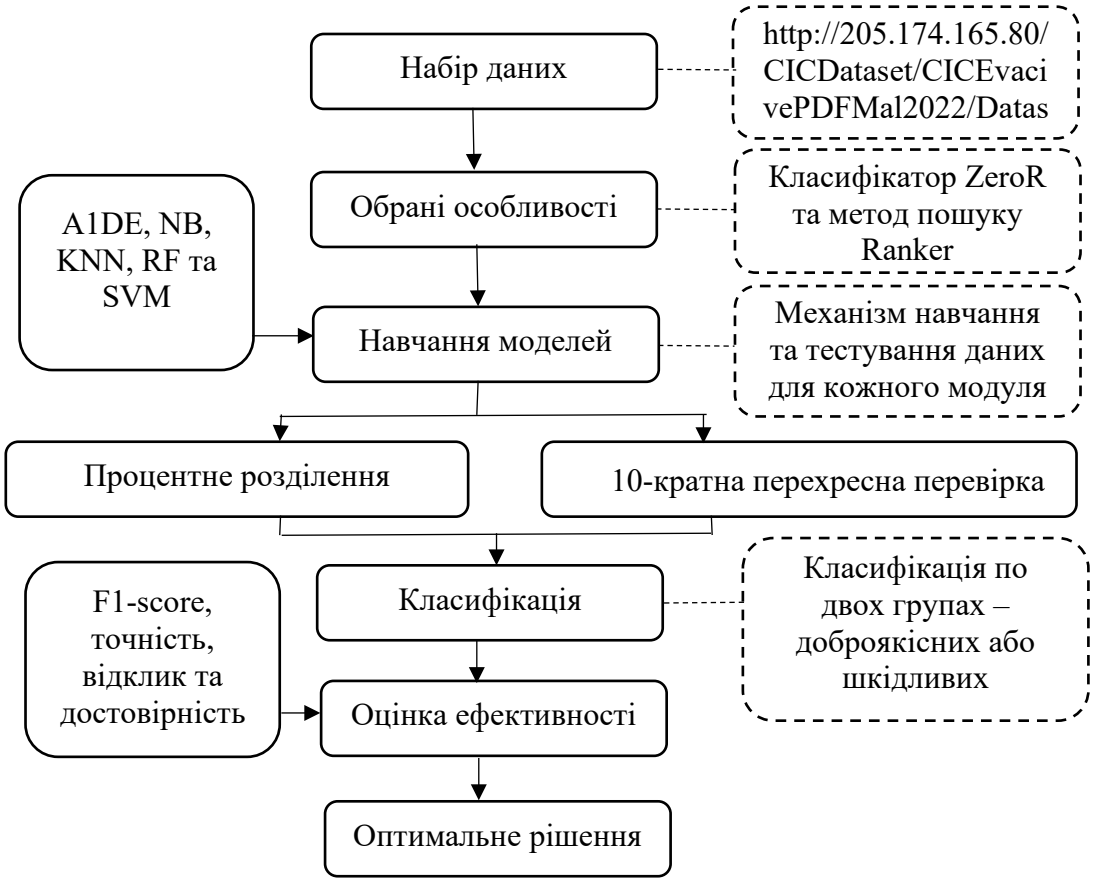



Рис. 1. Узагальнена схема дослідження

**Пояснення набору даних та попередня обробка**

Було взято набір даних про виявлення шкідливого програмного коду у файлах формату PDF від Канадського інституту кібербезпеки (https://www.unb.ca/cic/datasets/pdfmal-2022.html). Набір даних має 33 характеристики, 32 з яких є незалежними, а 1 з них – залежна. Перші 11 характеристик були усунені, оскільки вони не мали ефекту на етапі аналізу. Ці характеристики в сукупності називаються загальними ознаками, і вони містять наступну інформацію: шифрування, розмір метаданих, номер сторінки, заголовок, номер зображення, текст, номер об'єкта, шрифтові об'єкти, кількість вбудованих файлів та середній розмір усіх вбудованих носіїв – це всі фактори, які слід враховувати. Дані очищаються і не потребують подальшої попередньої обробки.

Для подальшого аналізу виникає необхідність вибрати деякі особливості, які найкращим чином підходять для аналізу. З цією метою ми вибираємо деякі особливості зі структурних особливостей, які визначають PDF-файл, з точки зору його структури, що вимагає більш ретельної обробки та розкриває інформацію про загальні рамки PDF-файлу.

Було використано методи Classifier Attribute Evaluator, що включають в себе класифікатор ZeroR та метод пошуку Ranker для отримання таких функцій. Для оцінки точності кількість складок, що використовуються, дорівнює 5. Вибрані функції ранжуються як вибрані атрибути: "21, 7, 8, 10, 6, 5, 4, 3, 2, 9, 11, 20, 18, 19, 12, 17, 16, 15, 14, 13, 1:21".

У такому порядку присутні такі атрибути: кольори, шифрування, JS, XFA startxref, трейлер, xref, кінцевий потік, потік, endobj, запуск, OpenAction, AA, EmbeddedFile, JBIG2Decode, Acroform, pageno, ObjStm, JavaScript, RichMedia та obj.

У таблиці 1 представлено обрані ознаки з їх описом.

*Таблиця 1*

| № | Ознака | Опис |
|---|---|---|
| 1 | Xref | Розмір потоку, оскільки шкідливий код може бути прихований у потоках |
| 2 | Trailer | Number of trailers inside the PDF |
| 3 | Pageno | Кількість трейлерів у PDF-файлі. Шкідливі PDF-файли часто містять менше сторінок – часто лише одну порожню сторінку – тому, що їм байдуже, як представлено їхній матеріал |
| 4 | Stream | Це показує, скільки послідовностей двійкових даних міститься в PDF-файлі |
| 5 | Encrypt | Ця функція вказує на те, чи захищений PDF-файл паролем |
| 6 | Objstm | Потоки, які містять додаткові об'єкти |
| 7 | Endstream | Ключові слова, що означають припинення трансляції |
| 8 | JS | Декілька об'єктів, що охоплюють код JavaScript |
| 9 | Obj | Це може бути ознакою спроби заплутатися |
| 10 | JavaScript | Це показує, як багато речей включає код JavaScript, найпоширеніший тип характеристики |
| 11 | AA | Визначає конкретну дію при інциденті |
| 12 | OpenAction | При відкритті PDF-файлу ця властивість визначає, яку дію слід виконати. Більшість шкідливих PDF-файлів, які часто зустрічаються, використовують цю функцію в поєднанні з JavaScript |
| 13 | endobj | PDF-файли дозволяють використовувати широкий спектр методів обфускації, включаючи обфускацію рядків у шістнадцятковій, вісімковій тощо, які часто використовуються в стратегіях ухилення |
| 14 | Acroform | Поля форм у PDF-файлах, створених за допомогою Acrobat, містять сценарії, які хакери можуть використати проти вас |
| 15 | Startxref | Численні ключові слова, які включають "startxref", вказують на місце початку таблиці Xref |
| 16 | JBIG2Decode | JBig2Decode – популярний фільтр для кодування небезпечних даних. Які елементи мають найбільшу кількість вкладених фільтрів? Вкладені фільтри |

| | | |
|---|---|---|
| | | можуть перешкоджати декодуванню та пропонувати ухилення |
| 17 | Richmeddia | Кількість мультимедійних ключових слів показує кількість флеш-пам'яті та вбудованих медіафайлів |
| 18 | Launch | Слово "launch" стосується акту виконання команди або програми |
| 19 | EmbeddedFile | Файли PDF можуть вкладати або вбудовувати інші речі, такі як документи Word, фотографії тощо, які можна використовувати зловмисно |
| 20 | XFA | XFA, архітектура XML-форм, що дозволяє використовувати технології сценаріїв, що можуть використовувати зловмисники, можна знайти в деяких PDF-файлах |
| 21 | Color | У PDF-файлі використовується багато кольорових схем |
| 22 | Class | Віднесіть до категорії доброякісних або шкідливих |

Завдяки своїй зручності, PDF став форматом документів, що найбільш часто використовується протягом усього часу. На жаль, розповсюдженість PDF-файлів та їх складні можливості дозволили зловмисникам використовувати їх різними способами. Зловмисник може скористатися кількома важливими властивостями PDF для поширення шкідливого корисного навантаження. Цей набір даних, який включає 10 019 записів, в яких 5551 шкідливий і 4468 є доброякісними, які схильні ухилятися від взаємно значущих ознак, виявлених у кожному класі, збирає ці шкідливі дані та інформацію.

Це дослідження зосереджено на двох різних типах аналізу тестування. Один заснований на відсотковому розщепленні набору даних, де ми використовували 70 % даних для навчання, а решта 30 % даних для тестування. Другий метод – K-кратна перехресна перевірка, в якій ми вибрали значення K як 10. У цьому дослідженні представлено порівняння цих методів тестування. Методи машинного навчання порівнюються за допомогою стандартних показників оцінки, таких як F1-score, точність, відгук та достовірність. Ці показники можна розрахувати за формулами (1)–(4):

$$\text{Точність} = \frac{TP}{TP+FP}; \tag{1}$$

$$\text{Чутливість} = \frac{TP}{TP+FN}; \tag{2}$$

$$F1 = \frac{2*\text{Точність}*\text{Чутливість}}{\text{Точність}+\text{Чутливість}}; \tag{3}$$

$$\text{Достовірність} = \frac{TP+TN}{TP+TN+FP+FN}, \tag{4}$$

де $F1$ – $F1$-score (ефективність прогнозування);

$TP$ – істинні позитивні значення розрахунків (true positive);

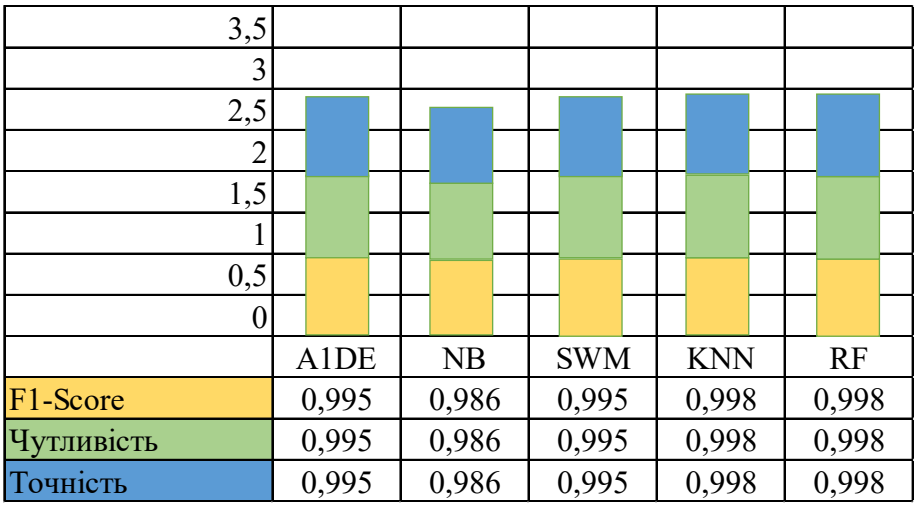
$FP$ – помилкові позитивні значення розрахунків (false positive);

$TN$ – істинно негативні розрахунків (true negative);

$FN$ – помилково негативні значення розрахунків (false negative).

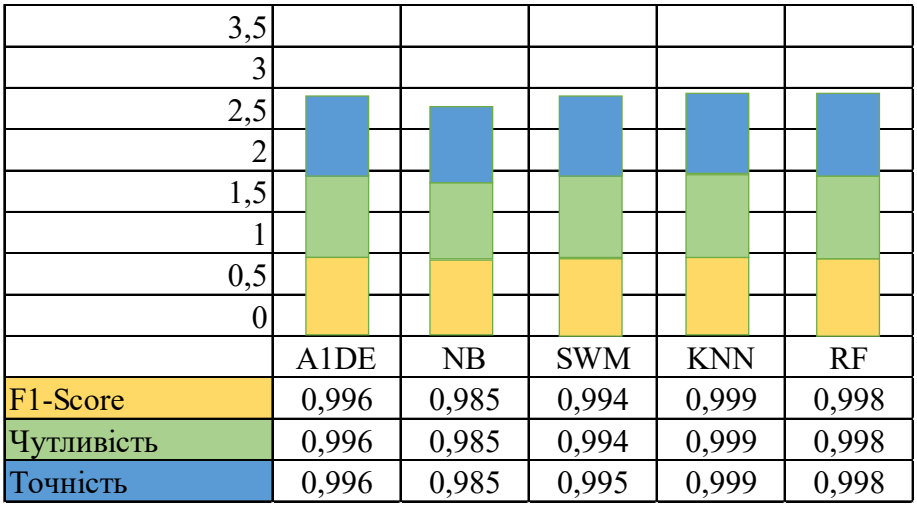
**Аналіз результатів.** Проведемо аналіз за допомогою вищезгаданих моделей машинного навчання, включаючи A1DE, NB, SVM, KNN та RF. Ці моделі оцінюються за допомогою F1-score (ефективність прогнозування), точності, чутливості та достовірності. Для відсоткового поділу було використано 70 % даних для навчання, а 30 % даних для тестування. У моделі K-кратної перевірки було обрано значення K як 10. Рисунок 2 представляє точність, чутливість та F1-score кожної техніки з використанням першого

критерію тестування, який становить 70 % та 30 % даних для навчання та тестування відповідно. На рисунку 3 представлено результати для 10-кратної перехресної перевірки.




| | A1DE | NB | SWM | KNN | RF |
|---|---|---|---|---|---|
| F1-Score | 0,995 | 0,986 | 0,995 | 0,998 | 0,998 |
| Чутливість | 0,995 | 0,986 | 0,995 | 0,998 | 0,998 |
| Точність | 0,995 | 0,986 | 0,995 | 0,998 | 0,998 |

Рис. 2. Аналіз точності, чутливості та F1-Score з використанням критерію розподілу відсотків




| | A1DE | NB | SWM | KNN | RF |
|---|---|---|---|---|---|
| F1-Score | 0,996 | 0,985 | 0,994 | 0,999 | 0,998 |
| Чутливість | 0,996 | 0,985 | 0,994 | 0,999 | 0,998 |
| Точність | 0,996 | 0,985 | 0,995 | 0,999 | 0,998 |

Рис. 3. Аналіз точності, чутливості та F1-Score за допомогою 10-кратної перехресної перевірки

Враховуючи критерії тестування, цей аналіз показує, що використовувати краще 10-кратну перехресну перевірку, а не поділ даних 70 % і 30 % для навчання та тестування. Крім того, KNN перевершує інші моделі, що використовуються за 10-кратною перехресною перевіркою. Використовуючи критерій відсоткового поділу, KNN і RF показують однакову продуктивність.

На рисунках 4 та 5 окремо представлений аналіз достовірності кожної моделі, що використовується з використанням обох критеріїв тестування, а саме з відсотковим поділом і K-кратною перехресною перевіркою.

В обох випадках KNN перевершує інші моделі з кращою точністю 99,8499 % за критеріями тестування відсоткового поділу та 99,8599 % за критеріями K-кратної перехресної перевірки. Цей аналіз показує, що в поточному сценарії K-кратна перехресна перевірка є кращими критеріями навчання та тестування для моделі навчання.

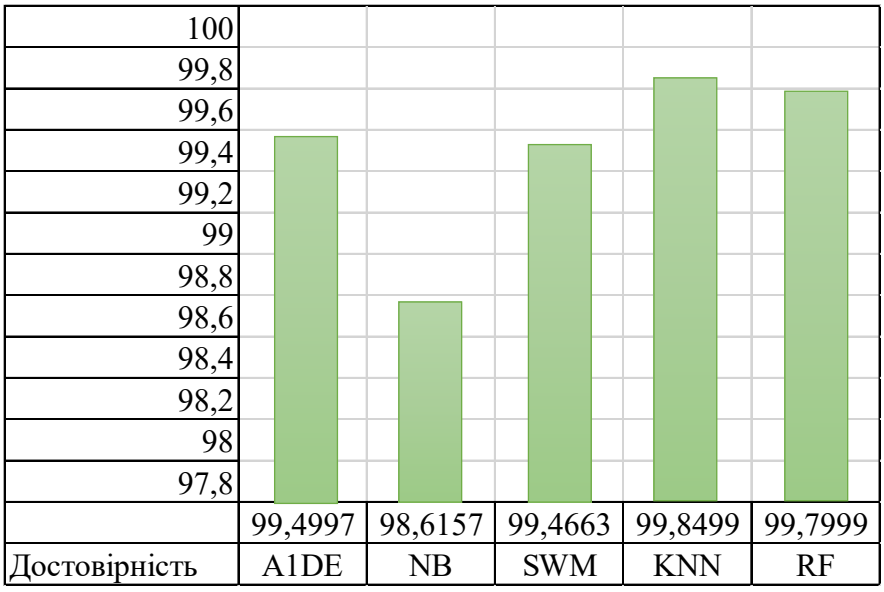



Рис. 4. Аналіз достовірності з використанням критеріїв тестування відсоткового поділу

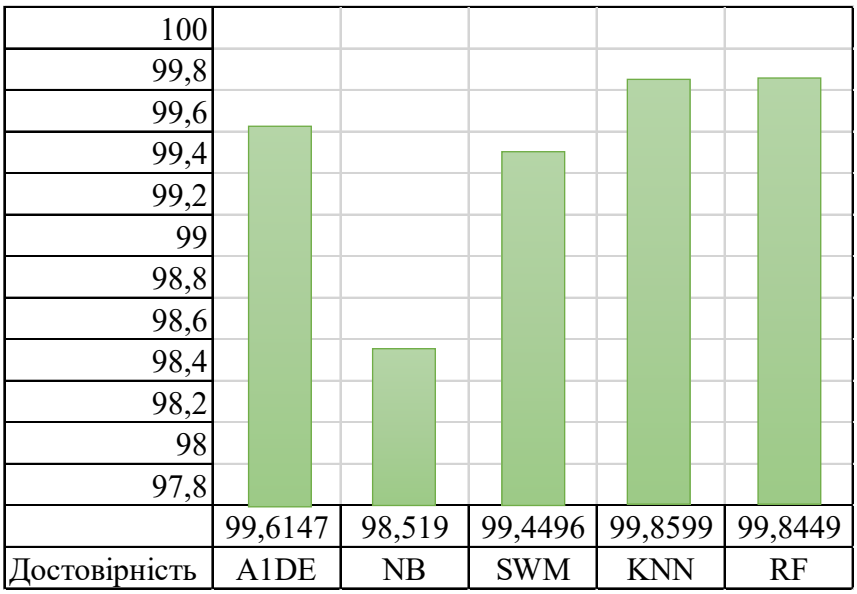



Рис. 5. 10-точковий аналіз з використанням 10-кратних критеріїв перехресної перевірки

На основі різноманітних вхідних значень, моделі машинного навчання прогнозують вихідні значення. Одним із найпростіших алгоритмів машинного навчання є KNN, який зазвичай використовується для класифікації. Він класифікує точку даних на основі того, як класифікуються її сусіди. Lazy Learner (навчання на основі екземплярів) – це інша назва KNN. Він нічого не вчиться протягом усього періоду навчання. Жодна дискримінаційна функція не генерується за допомогою навчальних даних. Іншими словами, ніякої підготовки не потрібно. Він лише черпає навчання зі збереженого набору навчальних даних для прогнозування в режимі реального часу. Техніка KNN набагато швидша, ніж інші розглянуті техніки, оскільки інші моделі вимагають навчання, тоді як техніка KNN не вимагає навчання раніше створення прогнозів, і свіжі дані можуть бути включені без зусиль, не впливаючи на точність методів.

В цілому, отримані результати показують, що моделі машинного навчання перевершують обидва методи оцінки, демонструючи їх ефективність у правильному

прогнозуванні цільової змінної. Оскільки розбиття даних на розділи та навчання моделі відрізняються в усіх двох методах, значення точності, отримані відсотковим поділом та 10-кратною перехресною перевіркою, різняться не суттєво. Однак відмінні рейтинги точності моделей демонструють їх потенціал для точних прогнозів у нинішній ситуації.

Результати роботи надають уявлення про ефективність моделей машинного навчання для виявлення шкідливих програм у PDF-файлах. Порівняльний аналіз та оцінка ефективності сприяють розробці надійних систем захисту від шкідливих дій, пов'язаних із шкідливим програмним забезпеченням PDF. Дослідження демонструє ефективність моделей машинного навчання у точному виявленні шкідливого програмного забезпечення PDF та надає уявлення про розробку надійних систем для захисту від шкідливих дій. Отримані дані свідчать про те, що KNN є перспективною моделлю для виявлення шкідливих програм PDF, але для перевірки та покращення результатів можуть знадобитися подальші дослідження та експерименти.

**Висновки.** У цій роботі було проведено порівняльний аналіз моделей машинного навчання для виявлення шкідливого коду в PDF-файлах, зосередивши увагу на моделях A1DE, NB, SVM, KNN та RF. При цьому використовувався набір даних, отриманий від Канадського інституту кібербезпеки та два критерії тестування: процентне ділення та 10-кратна перехресна перевірка. Оцінка базувалася на показниках точності, запам'ятовування, оцінки F1 та точності. Результати показали, що KNN перевершив інші моделі, досягнувши точності 99,8599 % за допомогою 10-кратної перехресної перевірки. Ці результати підкреслюють ефективність моделей машинного навчання у точному виявленні шкідливого програмного забезпечення PDF та надають уявлення про розробку надійних систем для захисту від шкідливих дій у файлах PDF. Дослідження сприяє посиленню заходів кібербезпеки, надаючи надійну модель для виявлення шкідливого програмного коду в PDF-файлах, яка може допомогти запобігти розповсюдженню витончених атак за допомогою зловмисно закодованих документів.

**Подальші напрямки досліджень** в області виявлення шкідливого програмного коду в PDF-файлах повинні бути зосереджені на декількох ключових областях, а саме дослідження методів глибокого навчання, таких як згорткові нейронні мережі (convolutional neural networks) та рекурентні нейронні мережі (recurrent neural networks) та можливості підвищення продуктивності і точності моделей. Включення методів обробки природної мови (natural language processing) для аналізу текстового вмісту в PDF-файлах, що може надати цінну інформацію для виявлення шкідливого програмного коду. Крім того, розробка систем, які можуть аналізувати PDF-файли в режимі реального часу та своєчасно виявляти нові загрози, була б дуже корисною.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Y. S. Jeong, J. Woo and A. R. Kang. Malware detection on byte streams of pdf files using convolutional neural networks // Security and Communication Networks. 2019. Vol. 2019. Pp. 144–152. DOI: 10.1155/2019/8485365.

2. A. Falah, S. R. Pokhrel, L. Pan and A. de Souza-Daw. Towards enhanced PDF maldocs detection with feature engineering: Design challenges // Multimedia Tools and Applications. 2022. Vol. 81, no. 28. Pp. 41103–41130. DOI: 10.1007/s11042-022-11960-x.

3. W. Xu, Y. Qi and D. Evans. "Automatically evading classifiers", in Proc. of the 23rd Annual Network and Distributed System Security Symp., San Diego, California, vol. 2016, no. February, pp. 21–24, 2016.

4. S. Sibi Chakkaravarthy, D. Sangeetha and V. Vaidehi. A survey on malware analysis and mitigation techniques. // Computer Science Review. 2019. Vol. 32. Pp. 1–23. DOI: 10.1016/j.cosrev.2019.01.002.

5. A. R. Kang, Y. S. Jeong, S. L. Kim and J. Woo. Malicious PDF detection model against adversarial attack built from benign PDF containing JavaScript // Applied Sciences. 2019. Vol. 9, no. 22. Pp. 4764. DOI: 10.3390/app9224764.

6. Y. Chen, S. Wang, D. She and S. Jana. "On training robust {PDF} malware classifiers," in 29th USENIX Security Symp. (USENIX Security 20), Boston, USA, pp. 2343–2360, 2020.

7. M. Cova, C. Kruegel and G. Vigna. "Detection and analysis of drive-by-download attacks and malicious JavaScript code", in Proc. of the 19th Int. Conf. on World Wide Web, North Carolina, USA, pp. 281–290, 2010.

8. P. Laskov and N. Srndic. "Static detection of malicious JavaScript-bearing PDF documents", in Proc. of the 27th Annual Computer Security Applications Conf., Orlando, Florida, USA, pp. 373–382, 2011.

9. S. J. Khitan, A. Hadi and J. Atoum. PDF forensic analysis system using YARA // International Journal of Computer Science and Network Security. 2017. Vol. 17, no. 5. Pp. 77–85.

10. J. Zhang. MLPdf: An effective machine learning based approach for PDF malware detection. 2018. Pp. 1–6.

11. S. S. Khan, M. Khan, S. Technology and R. Naseem. Challenges in opinion mining, comprehensive // A Science and Technology Journal, 2018. Vol. 33, no. 11. Pp. 123–135.

12. D. Liu, H. Wang and A. Stavrou. "Detecting malicious JavaScript in pdf through document instrumentation", in 2014 44th Annual IEEE/IFIP Int. Conf. on Dependable Systems and Networks, Atlanta, USA, pp. 100–111, 2014.

13. J. A. Herrera-Silva and M. Hernández-Álvarez. Dynamic feature dataset for ransomware detection using machine learning algorithms // Sensors. 2023. Vol. 23, no. 3. Pp. 1053. DOI: 10.3390/s23031053.

14. M. Dhalaria and E. Gandotra, "A hybrid approach for android malware detection and family classification," International Journal of Interactive Multimedia and Artificial Intelligence, vol. 6, no. 6, pp. 174–188, 2020. DOI:10.9781/ijimai.2020.09.001

15. M. Deore and U. Kulkarni. Mdfrcnn: Malware detection using faster region proposals convolution neural network // International Journal of Interactive Multimedia and Artificial Intelligence. 2022. Vol. 7, no. 4. Pp. 146–162.

16. T. Tsafrir, A. Cohen, E. Nir and N. Nissimю. Efficient feature extraction methodologies for unknown MP4-malware detection using machine learning algorithms // Expert Systems with Applications. 2023. Vol. 219. Pp. 119615. DOI: 10.1016/j.eswa.2023.119615.