УДК 004.9 **doi: 10.26906/SUNZ.2022.2.079**

В. В. Прокопов, Є. В. Мелешко, М. С. Якименко, В. А. Резніченко, С. В. Шимко

Центральноукраїнський національний технічний університет, Кропивницький, Україна

РОЗРОБКА СИСТЕМИ ВИЯВЛЕННЯ КІБЕРЗАГРОЗ НА ОСНОВІ АНАЛІЗУ ДАНИХ З ВЕБ-РЕСУРСІВ НА МОВІ ПРОГРАМУВАННЯ PYTHON

Анотація. Метою даної роботи є створення системи виявлення кіберзагроз на основі аналізу даних мережевого трафіку веб-ресурсів засобами мови програмування Python та з використанням методів машинного навчання. Об'єктом дослідження є процес аналізу даних з веб-ресурсів у системах кібербезпеки. Предметом дослідження є методи та алгоритми машинного навчання для аналізу даних з веб-ресурсів. Для навчання розробленої моделі виявлення кібератак було обрано відкритий набір даних CSE-CIC-IDS2017, що містить найсучасніші поширені інформаційні атаки, що відповідають вигляду справжніх даних з реального світу, основні реалізовані атаки включають брутфорс FTP, брутфорс SSH, DoS, Heartbleed, веб-атаку, інфільтрацію, ботнет та DDoS. Розроблене програмне забезпечення виявлення кібератак на веб-сайти складається з декількох модулів, а саме: модуля попередньої обробки даних датасету, модуля дослідження ознакового простору мережевого трафіку та модуля використання алгоритмів машинного навчання для пошуку кібератак. Для вирішення завдання з відбору ознак у рамках реалізації даного програмного забезпечення було вирішено обрати стратегію відбір на основі моделі за допомогою одного з ансамблевих методів машинного навчання випадковий ліс. Відбір ознак на основі моделі використовує алгоритм машинного навчання з учителем, щоб обчислити важливість кожної ознаки, і залишає лише найважливіші з них. Для тренування моделі були вибрані наступні алгоритми машинного навчання: наївний баєсів класифікатор, k-найближчих сусідів, дерева рішень, метод опорних векторів (SVM) з використанням гауссівського ядра, адаптивний бустинг, дерева рішень з прискоренням (бустинг градієнта). Разом з тренуванням одразу виконувалася перехресна перевірка (з контролем) по семи блоках, для отримання більш точної оцінки узагальнюючої здатності моделі. Результат роботи – програмна реалізація методів машинного навчання для виявлення кібератак на веб-сайти за допомогою розпізнавання їх ознак у мережевому трафіку, а також проведення порівняння їх ефективності.

Ключові слова: кібербезпека, кібератака, кластеризація, аналіз даних, веб-ресурси, мережевий трафік.

Вступ

Забезпечення комп'ютерної безпеки у різних сферах людської життєдіяльності є задачею, яка має всі підстави розглядатися, як одна із найбільш важливих проблем сучасного суспільства. Адже суспільство стає все більш залежним від комп'ютерів та Інтернет-мережі у роботі та проведенні дозвілля. Зростає і значимість наявності вразливостей в комп'ютерних системах, що привертають увагу зловмисників, які використовують їх як можливість отримати гроші або заподіяти шкоду [1, 2]. Також вражливості комп'ютерних систем можуть використовуватися протиборчими сторонами для здобуття переваги під час інформаційних протиборств та війн [2]. Тому вкрай важливо захищати комп'ютерні системи та веб-ресурси від кібератак, а також розпізнавати такі атаки для їх своєчасного усунення, якщо методи превентивного захисту не спрацювали.

Одними з найефективніших методів виявлення кібератак на веб-ресурси є ті, що використовують методи машинного навчання [3-8]. Сучасні алгоритми аналізу даних дозволяють виявляти закономірності та ознаки в трафіку веб-сайтів [8-11], що дає додаткові можливості для виявлення кібератак методами штучного інтелекту.

Метою роботи є дослідження та програмна реалізація методів виявлення кібератак на веб-ресурси на основі машинного навчання та аналізу даних.

Основна частина

У даній роботі було розроблено програмне забезпечення для виявлення кібератак мережевого рівня моделі OSI у мережі Інтернет. Атаки виявлялися на основі їх ознак з використанням методів машинного навчання. Аналіз та обробка отриманих даних проводилася за допомогою використання мови програмування Python 3.10 та наступних бібліотек:

– pandas для обробки даних;

– scikit-learn для машинного навчання;

– numpy для виконання математичних операцій;

– matplotlib для візуалізації та виведення даних у вигляді графіків.

Для тренування розроблюваної моделі виявлення кібератак було обрано відкритий набір даних *CSE-CIC-IDS2017* [12]. Цей датасет був створений за результатами аналізу мережевого трафіку в ізольованому середовищі, в якому моделювалися дії звичайних користувачів, а також шкідливі дії порушників. Набір даних CSE-CIC-IDS2017 містить розмічений мережевий трафік з наявністю поширених кібератак та відповідає вигляду реального трафіку у форматі PCAP. У ньому представлені такі атаки [12]:

- DoS Hulk,
- PortScan,
- DDoS,
- DoS GoldenEye,
- FTP-Patator,
- SSHPatator,
- DoS slowloris,
- DoS Slowhttptest,
- Bot,
- Infiltration,
- Heartbleed,
- Web Attack – Brute Force,
- Web Attack – XSS,
- Web Attack – SQL Injection.

Датасет CSE-CIC-IDS2017 також включає результати аналізу мережевого трафіку за допомогою CICFlowMeter, інструменту генерації та аналізу мережевого трафіку, з позначеними потоками на основі відмітки часу, IP-адрес джерела і призначення, портів джерела та призначення, протоколів та атак (файли CSV). Створення реалістичного трафіку було головним пріоритетом у створенні цього набору даних. Було використано систему B-profile для профілювання абстрактної поведінки людських взаємодій і створення натуралістичного трафіку. Для цього набору даних було побудовано абстрактну поведінку 25 користувачів на основі протоколів HTTP, HTTPS, FTP, SSH та електронної пошти.

Сам набір даних в загальній сукупності містить у собі понад два мільйони чотириста тисяч зразків даних, кожен з яких розмічений як сутність, що належить до певного класу, який описує належність до нормального трафіку (benign) чи шкідливого (наприклад, PortScan, DDoS, Bot, і т.д.). Всього в датасеті виділяється близько п’ятнадцяти різних класів, та слід зазначити, що кількість зразків в кожному із них розподілена вкрай нерівномірно, та ті сильно відрізняється від класу до класу. Наприклад, кількість зразків у класі heartbleed складає усього одинадцять, тоді як клас goldeneye налічує близько десяти тисяч зразків. Якщо взяти усю сукупність даних то частка класу benign складає близько 84%, а решта 16% припадає на усі інші класи атак. Тож, беручи до уваги ці фактори, аби уникнути проблеми із збалансованістю даних необхідним є проведення наступних дій: об’єднання усіх класів атак в один єдиний (це також зводить проблему класифікації з мультикласової до бінарної); забезпечення міжкласового балансу.

Однією із методик вирішення проблеми дизбалансованості між класами є методика субдискретизації.

Сутність даного методу полягає у вибірці елементів із домінуючих класів із метою скорочення їх кількості. Стратегії субдискретизації можуть бути простими, як, наприклад, випадковий вибір групи елементів, але при цьому можливі втрати інформації у певних наборах даних. У таких випадках стратегія вибірки має передбачати в першу чергу видалення елементів, які дуже схожі на інші елементи, що залишаються в наборі даних. Для відкидання надлишкових даних було застосовано техніку субдискретизації мажоритарного (домінуючого) класу на основі центроїдів кластерів. Ця техніка полягає у створені за допомогою алгоритму кластеризації (у даному випадку метод k-середніх) кластеру домінуючого класу та у подальшому відкиданню зразків, керуючись відстанню від центроїда до положення зразку у просторі, розрахованою за евклідовою метрикою.

Увесь датасет представлений у вигляді кількох файлів у форматі .csv:

- Monday-WorkingHours.pcap_ISCX;
- Tuesday-WorkingHours.pcap_ISCX;
- Wednesday-workingHours.pcap_ISCX;
- Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX;
- Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX;
- Friday-WorkingHours-Morning.pcap_ISCX;
- Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX;
- Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.

Кожен файл частково чи повністю містить дані лише про певні види мережевих атак, тож у даній роботі їх було об’єднано у один єдиний файл задля забезпечення простоти у подальших маніпуляціях, змінах та перетвореннях даних.

Загальна кількість атрибутів, які описують кожний окремий зразок даних, становить близько 80-ти. Така велика кількість характеризуючих ознак, хоча і послугує для якнайбільш якіснішого відділення зразків між собою та класів, може виявитися надлишковою оскільки не кожна ознака може слугувати для виявлення унікальності, що буде виявляти відмінність одного класу від іншого; деякі ознаки взагалі можуть не нести ніякої корисної інформації, яка б описувала дані. Беручи до уваги велику кількість зразків даних та розмір ознакового простору буде доцільним провести відбір ознак. Тобто буде доречно створити таку підмножини ознак, яка буде значно меншою порівняно з наявною, але зведе до мінімуму втрату вагомої інформації. Зниження ознакового простору набору даних дозволить також отримати низку вагомих переваг, як наприклад: підвищення ступеню інтерпретації моделі; збільшення швидкості навчання; зменшення ймовірності приймання рішень моделлю на основі «шумів», що буде мати позитивний вплив на якість навчання.

Оскільки самих лише експертних знань (тобто прийняття рішень щодо формування чи відсіювання ознак оперуючись суто апріорними знаннями) може виявитися замало для прийняття рішення щодо відкидання того чи іншого атрибуту даних, то для виявлення важливості (та не важливості) тих чи інших атрибутів слід також прибігти до можливостей, які надають алгоритми машинного навчання.

Загалом виділяють три основні стратегії для відбору ознак: 1) одновимірні статистики, 2) відбір на основі моделі та 3) ітеративний відбір.

Для вирішення завдання з відбору ознак у рамках реалізації даного програмного забезпечення було вирішено обрати стратегію відбору ознак на основі моделі за допомогою одного з ансамблевих методів машинного навчання *випадковий ліс*. Відбір ознак на основі моделі використовує алгоритм машинного навчання з учителем, щоб обчислити важливість кожної ознаки, і залишає лише найважливіші з них. Модель машинного навчання з учителем, яка використовується для відбору ознак, не повинна використовуватись для побудови підсумкової моделі. Модель, що застосовується для відбору ознак, вимагає обчислення певного показника важливості для всіх ознак, щоб характеристики можна було ранжувати за цією метрикою.

Випадковий ліс (Random forest) – це алгоритм машинного навчання, що полягає у використанні сукупності дерев ухвалення рішень (таку сукупність

ще називають ансамблем) [13, 14]. Основними напрямками застосування алгоритму є задачі класифікації, регресії та кластеризації. Основна ідея полягає у використанні великого ансамблю дерев класифікації, кожне з яких саме собою дає дуже невисоку якість класифікації, але за рахунок того, що їх використовується велика кількості кінцевий результат виходить хорошим.

Спочатку для надання оцінки важливості ознак у тренувальному наборі проводиться навчання випадкового лісу на цьому наборі. В продовж процесу побудови моделі для кожного елемента тренувального набору записується так звана помилка невідібраних елементів (ПНЕ). Потім ця помилка усереднюється у всьому випадковому лісі для кожної із сутностей.

Задля того, аби винести оцінку важливості для *i*-го параметра після тренування, значення цього параметра випадковим чином перемішуються для всіх записів тренувального набору і виконується обчислення ПНЕ знову. Визначення важливості параметру відбувається шляхом усереднення по всіх деревах різниці показників ПНЕ до перемішування значень та після. Під час здійснення цього процесу проводиться нормалізація на обчислюється стандартне відхилення значення для всіх таких помилок.

Важливість параметру вибірки для тренувального набору визначається величиною його значення. Один із вагомих недоліків цього методу полягає в тому, що для категоріальних змінних із великою кількістю значень метод схильний вважати такі змінні важливішими. Часткове перемішування значень може знижувати вплив цього ефекту. З груп параметрів, важливість яких виявляється однаковою, вибираються менші за чисельністю групи [13].

Важливе місце у аналізі та досліджені даних відводиться їх попередній обробці. Етап попередньої обробки даних у даному випадку включає в себе перевірку кожного значення атрибуту, заповнення відсутніх значень, кодування даних у формат зрозумілий для моделі.

Так, зокрема, необхідно провести дослідження ознак і замінити значення Infinity на значення -1, замість значення inf поставити 0, теж саме із значеннями типу NaN [6]. Наостанок проводиться відбір та перетворення усіх атрибутів нечислового типу (категоріальні, строкові, змішані і т.д.).

Після проведення етапу попередньої обробки даних, зміни значень даних, заповнення пропущених місць, формування ознак, видалення ознак, форматування значень атрибутів – наступає етап розділення набору даних на дві різні підмножини: тренувальний та тестовий набори.

Для тренування моделі були вибрані наступні алгоритми машинного навчання [14]:

- *наївний баєсів класифікатор*. Цей класифікатор називається «наївним», тому що ґрунтується на вельми суворих статистичних вихідних передумовах, а саме: ознаки вибираються незалежно з деякого (невідомого заздалегідь) розподілу. Основна ідея, закладена в основу наївного байєсівського класифікатора, полягає в наступному: взявши елемент даних на основі набору ознак необхідно визначити ймовірність того, що йому треба присвоїти мітку якогось певного класу;

- *k-найближчих сусідів*. Даний алгоритм виділяється з понад інших алгоритмів машинного навчання своєю відносною простотою. В основі принципу роботи лежить наступний процес: запам’ятовування тренувального набору даних. Для проведення прогнозу для нового (невідомого) зразка даних, алгоритм знаходить найближчих сусідів – тобто найближчі до нового зразку точки з навчального набору. Цей тип методики машинного навчання відкладає більшу частину обчислень на час класифікації, замість того, щоб виконувати цю роботу під час навчання. Моделі лінивого навчання не навчаються узагальнення даних під час тренувальної стадії. Натомість вони фіксують усі передані їм точки тренувальних даних і використовують цю інформацію для створення локальних узагальнень на тестовій вибірці під час класифікації;

- *дерева рішень*. Дерева рішень являють собою досить універсальні та ґнучкі моделі навчання з учителем, які мають дуже важливу властивість – простоту інтерпретації. Використовують структуру даних у вигляді бінарного дерева для прийняття рішень. Дерева є цілком інтуїтивно зрозумілим способом візуального представлення та аналізу даних, тому дуже широко використовуються навіть поза межами області машинного навчання. Ці структури надають можливість прогнозування як категоріальних значень (дерева класифікації), так і значення у форматі дійсних чисел (дерева регресії), а також здатні містити і числові, і категоріальні дані без операцій нормалізації або створення фіктивних змінних;

- *метод опорних векторів (SVM)*. В найпростішому варіанті SVM представляє з себе лінійний класифікатор, тобто дозволяє створити гіперплощину у векторному просторі, щоб спробувати розділити два класи в заданому наборі даних. Метод опорних векторів використовує функцію залежних втрат, яка штрафує лише ті точки, які розташовані на неправильній стороні відносно гіперплощини або дуже близькі до гіперплощини, але знаходяться на правильній стороні. Більш точно, SVM класифікатор намагається знайти максимальну гіперплощину, що розділяє два класи, де «кордон» позначає відстань від площини, що розділяє векторний простір навпіл, до найближчих точок даних на кожній стороні. У тому випадку, коли дані розділені не прямою лінією, точки всередині цієї межі штрафуються пропорційно їх віддаленості від кордону;

- *Adaptive Boosting*. Алгоритм машинного навчання, що посилює класифікатори, поєднуючи їх у «комітет». AdaBoost є адаптивним тому, що кожен наступний комітет класифікаторів будується по об’єктах, неправильно класифікованих попередніми комітетами. AdaBoost чутливий до шуму даних і викидів. Однак він менш схильний до перенавчання в порівнянні з іншими алгоритмами машинного навчання;

- *дерева рішень з прискоренням* (градієнтний бустинг). Такий алгоритм застосовує витончені

комбінації прогнозів окремих дерев рішень на формування поліпшених узагальнених прогнозів. При використанні методики прискорення або Gradient Boosting декілька слабких об'єктів, що навчаються вибірково об'єднуються за допомогою виконання оптимізації градієнтного спуску у функції втрат, щоб отримати в результаті набагато більш потужну модель навчання. Основною методикою прискорення або Gradient Boosting є додавання окремих дерев до лісу по одному з використанням процедури градієнтного спуску для мінімізації втрат при додаванні дерев. Процедура додавання дерев у ліс зупиняється при досягненні встановленої граничної кількості, коли валідаційний набір втрат досягає прийнятного рівня або якщо подальше додавання дерев не може поліпшити (мінімізувати) рівень втрат.

Після проведення навчання моделі необхідним є надання оцінки її узагальнюючій здатності, тобто перевірка того наскільки вона на основі отриманих знань з навчального набору може їх ефективно використовувати для розпізнавання нових досі невідомих даних. Тобто необхідно визначити метрики якості (чи показники продуктивності) моделі.

Найпростішою метрикою якості моделі є оцінка правильності її роботи. Для її підрахунку необхідно знайти відсоткове відношення правильно класифікованих разків даних.

Більш ефективним показником може слугувати матриця помилок. Суть методу полягає у тому, щоб підрахувати, скільки разів зразки класу *A* були віднесені до класу *B*. При розглядані задачі класифікації мережевих даних як задачі бінарної класифікації необхідно буде з'ясувати скільки разів класифікатор плутав звичайні дані з атаками. Для розрахунку матриці помилок спочатку потрібно мати набір прогнозів, щоб їх можна було порівнювати з фактичними даними. Кожен рядок у матриці помилок представляє фактичний клас, а кожен стовпець – прогнозований клас.

У матриці помилок (табл. 1) підраховується кількість наступних подій:

– вірні класифікації даних як кібератак (TP);

– помилкові класифікації даних як кібератак (FP);

– вірні класифікації даних як звичайних (TN);

– помилкові класифікації даних як звичайних (FN).

Таблиця 1 – **Матриця помилок для визначення ефективності методу класифікації трафіку**

	Звичайний трафік	**Кібератака**
Розпізнано як звич. трафік	True-Positive (*tp*)	False-Negative (*fn*)
Розпізнано як кібератаку	False-Positive (*fp*)	True-Negative (*tn*)

На основі табл. 1 можна обчислювати різні показники якості роботи системи, зокрема, точність (1) та повноту (2) її роботи, а також F-міру (3):

$$Precision = \frac{tp}{tp + fp}, \quad (1)$$

$$Recall = \frac{tp}{tp + fn}, \quad (2)$$

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}. \quad (3)$$

Розроблюване програмне забезпечення аналізу даних для виявлення кібератак складається з декількох модулів:

– модуль попередньої обробки даних датасету;

– модуль дослідження ознакового простору мережевого трафіку;

– модуль використання алгоритмів машинного навчання для пошуку кібератак.

Модуль попередньої обробки даних об'єднує в один файл датасет, що складається із декількох окремих файлів, а також виконує низку важливих дій пов'язаних із підготуванням даних до подальшої роботи алгоритмами. Так із самого початку відбувається перевірка всіх файлів набору даних на наявність у них нульових значень, пропусків, типів даних, які не будуть сприйматися алгоритмами машинного навчання і т.д. Також проводиться простий статистичний аналіз кількості даних датасету з подальшою візуалізацією у вигляді графіків. Після проведення попередньої обробки даних відбувається злиття новоутворених даних в один файл із подальшим збереженням.

Збереження проміжних файлів необхідне для того аби забезпечити модульність функціонування системи, оскільки це дозволить виконувати окремі операції у довільному порядку. Таке рішення необхідне оскільки виконання деяких дій пов'язаних з аналізом та обробкою інформації може потребувати великої кількості часу, а збереження деяких проміжних даних та результатів виконання операцій дозволить уникнути необхідності у поступовому і безперервному виконанні системи і зарадить втраті більшої кількості вагомих даних у разі збою. Також таке зберігання даних на кожному етапі роботи системи має позитивний вплив на економію ресурсів персонального комп'ютера та часу користувача.

Наступним компонентом системи є модуль дослідження ознакового простору. В цьому модулі відбувається виконання операцій пов'язаних із дослідженням ознакового простору даних. З усіх ознак вибираються найбільш вагомі та значущі для зразків свого класу. При необхідності відбувається формування нових (наприклад, за допомогою об'єднання двох), видалення незначущих і т.д.

Модуль використання алгоритмів машинного навчання для пошуку кібератак відповідає за:

1) розділення всього датасету на дві підмножини навчальний та тестовий набори даних;

2) тренування моделей машинного навчання на навчальних даних;

3) перевірку їх узагальнюючої здатності на тестовому масиві даних за допомогою метрик якості роботи.

Функціональна схема програмного забезпечення для аналізу даних зображена на рис. 1.

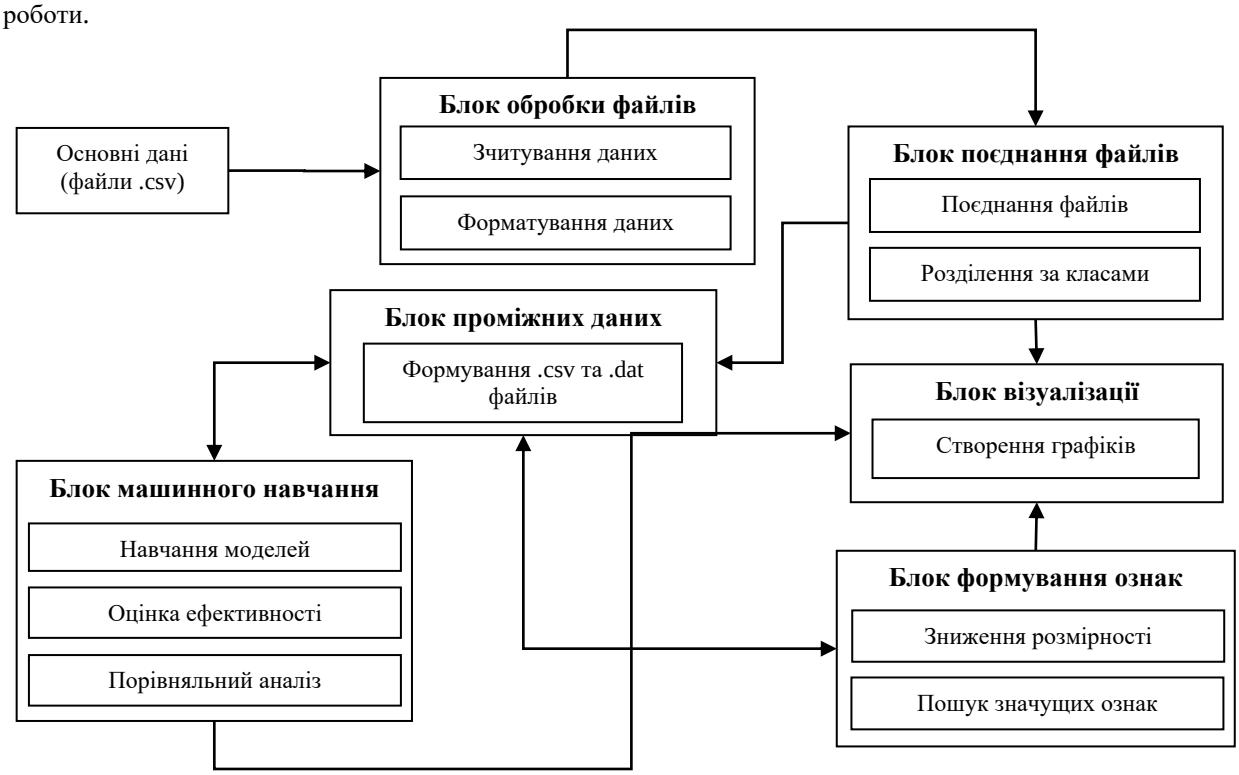

Рис. 1. Функціональна схема системи виявлення кіберзагроз на основі аналізу даних з веб-ресурсів

Програма починає роботу із пошуку необхідних файлів у каталозі та формування списку. Далі відбувається відкриття одного файлу з даними Інтернет-трафіку і починається його обробка. Кожен рядок файлу представляє собою опис певного виду трафіку (зразок даних), де у колонці вказано значення його ознак, сукупність яких, тим чи інакшим чином, відрізняє один зразок від іншого. Відкриття та обхід полів відбувається за допомогою бібліотеки для обробки даних pandas, оскільки вона допомагає зчитувати файли формату .csv та перетворювати їх на дані формату Serial та DataFrame – формат представлення даних, який значно полегшує проведення будь-яких операцій з даними (вилучення, сортування, видалення, зміна і т.д.). Проведення даної процедури необхідне для того, щоб здійснити форматування даних, привести їх до виду зрозумілого для мови програмування Python або навіть позбутися деяких надлишкових ознак, тих, які не несуть якої-небудь вагомої інформації. Також відбувається сортування ранніх видів сутностей за значенням поля label, яке вказує чи є сутність шкідливою чи ні (benign – звичайний трафік, а усе інше можна віднести до трафіку зловмісного характеру).

Після завершення операцій форматування та класифікацій відбувається статична обробка новостворених файлів, яка необхідна для того, щоб мати точне представлення про види та кількість полів, що відносяться до тієї чи іншої сутності.

Наступним етапом є виявлення таких ознак для кожного окремого набору сутностей, які несуть найбільшу про них інформацію. Для втілення цієї мети використовується алгоритм Random forest. Він в даному випадку використовується для обчислення значущості ваги об’єктів у наборі даних.

Потім у розроблюваній системі відбувається навчання моделі на наборі даних CSE-CIC-IDS2017, а також її тестування для визначення точності її роботи при виявленні кібератак у мережевому трафіку.

Ефективність моделей оцінювалася за показниками їх правильності, точності, повноти та F-мірою. Найбільш ефективні результати (за F-мірою) показали градієнтний бустинг (97,8%) та адаптивний бустинг (97,6%), потім усі інші: k-найближчих сусідів (96%), ядерний метод опорних векторів (95%), дерево рішень (95%) та баєсів класифікатор (77%).

Висновки

У статті наведено результати дослідження та розробки системи виявлення кіберзагроз на основі аналізу даних з веб-ресурсів на мові програмування Python. Було розроблено програмне забезпечення для виявлення кібератак мережевого рівня моделі OSI у мережі Інтернет. Атаки виявлялися на основі їх ознак з використанням методів машинного навчання. Для тренування моделі виявлення кібератак було обрано відкритий набір даних CSE-CIC-IDS2017, що містить змодельований розмічений мережевий трафік зі звичайними даними та кібератаками у форматі PCAP, основні реалізовані атаки включають брутфорс FTP, брутфорс SSH, DoS, Heartbleed, веб-атаку, інфільтрацію, ботнет та DDoS.

Програмне забезпечення обробки та аналізу даних складається з декількох модулів:

- модуль попередньої обробки даних датасету;
- модуль дослідження ознакового простору мережевого трафіку;
- модуль використання алгоритмів машинного навчання для пошуку кібератак.

Для тренування моделі були обрані наступні алгоритми машинного навчання: наївний баєсів класифікатор, k-найближчих сусідів, дерева рішень, метод опорних векторів (SVM) з використанням гауссівського ядра, адаптивний бустинг, дерева рішень з прискоренням (градієнтний бустинг).

Разом з тренуванням одразу виконувалася перехресна перевірка (з контролем) по семи блоках, для отримання більш точної оцінки узагальнюючої здатності моделі.

Найбільш ефективні результати (за f-мірою) показали градієнтний бустинг (97,8%) та адаптивний бустинг (97,6%), потім усі інші: k-найближчих сусідів (96%), ядерний метод опорних векторів (95%), дерево рішень (95%) та баєсів класифікатор (77%).

Розроблене програмне забезпечення призначене для аналізу мережевого трафіку і пошуку кібератак, володіє таким функціональними можливостями для роботи з даними: проведення статистичного аналізу даних, систематизація, виявлення закономірностей та тенденцій, графічне (візуальне) представлення.

СПИСОК ЛІТЕРАТУРИ

1. Chang J. (2021) “10 Cybersecurity Trends for 2022/2023: Latest Predictions You Should Know”, URL: https://financesonline.com/cybersecurity-trends/
2. Branch J. (2021). “What's in a Name? Metaphors and Cybersecurity”, International Organization, vol. 75, no. 1, pp. 39-70. doi:10.1017/S002081832000051X URL: https://www.cambridge.org/core/journals/international-organization/article/abs/
3. Ford V., Siraj A. (2014) “Applications of Machine Learning in Cyber Security”, ISCA 27th International Conference on Computer Applications in Industry and Engineering (CAINE-2014), held in New Orleans, LA, October 13-15, 2014.
4. Lewis M. (2017) “Rise of the machines: Machine Learning & its cyber security applications”, NCC Group Whitepape.
5. Sommer R., Paxson V. (2010) “Outside the Closed World: On Using Machine Learning for Network Intrusion Detection”, 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305-316, doi: 10.1109/SP.2010.25. URL: https://ieeexplore.ieee.org/document/5504793
6. Burkov A. (2019) The Hundred-Page Machine Learning Book. – pp. 160.
7. Чио К., Фримэн Д. Машинное обучение и безопасность / пер. с анг. А. В. Снастина. – М.: ДМК Пресс, 2020. – 388 с.
8. Kostas K. (2018) “Anomaly Detection in Networks Using Machine Learning”. Research Proposal, march 2018, pp. 1-64.
9. Bhattacharyya D. K. (2013) “Network Anomaly Detection: A Machine Learning Perspective 1st Edition”, Chapman and Hall/CRC. – pp. 366.
10. Flach P. (2012) “Machine Learning: The Art and Science of Algorithms that Make Sense of Data. 1st edition”, Cambridge University Press. – pp. 416.
11. Орельен Ж. (2018) “Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем”, Пер. с англ. – СпБ.: ООО "Альфа-книга”. – 688 с.
12. Canadian Institute for Cybersecurity (2017) “Intrusion Detection Evaluation Dataset (CSE-CIC-IDS2017)”, URL: https://www.unb.ca/cic/datasets/ids-2017.html
13. Breiman L. (2001) “Random Forests”, Machine Learning journal, Vol. 45, no. 1. – P. 5-32. – doi:10.1023/A:1010933404324, URL: https://link.springer.com/article/10.1023/A:1010933404324
14. Lutins E. “Ensemble Methods in Machine Learning: What are They and Why Use Them?”, Towards Data Science. URL: https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f
15. Sunil (2017) “Commonly used Machine Learning Algorithms (with Python and R Codes)”, Analytics Vidhya. URL: https://www.analyticsvidhya.com/blog /2017/09/common-machine-learning-algorithms/

Received (Надійшла) 14.02.2022
Accepted for publication (Прийнята до друку) 27.04.2022

The methods of data storing of a recommendation system based on linked lists

V. Prokopov, Ye. Meleshko, M. Yakymenko, V. Reznichenko, S. Shymko

Abstract. The goal of this work is to develop a system for detecting cyber threats based on the analysis of network traffic data of web resources using Python programming language and using machine learning methods. The object of research is the process of analyzing data from web resources in cybersecurity systems. The subject of research is the methods and algorithms of machine learning for the analysis of data from web resources. CSE-CIC-IDS2017 open data set was chosen to train the developed model of cyberattack detection, which contains the most modern common information attacks that correspond to the real world data, the main implemented attacks include FTP brute force, SSH brute force, DoS, Heartbleed, web attack , infiltration, botnet and DDoS. The developed software for detecting cyberattacks on websites consists of several modules, namely: a module for data processing pre-processing, a module for researching the feature space of network traffic and a module for using machine learning algorithms to search for cyberattacks. To solve the problem of selection of features in the implementation of this software, it was decided to choose a selection strategy based on the model using one of the ensemble methods of machine learning random forest. Model-based feature selection uses a machine learning algorithm with the teacher to calculate the importance of each feature, leaving only the most important ones. The following machine learning algorithms were chosen to train the model: naive Bayesian classifier, k-nearest neighbors, decision trees, support vector machine (SVM) with using Gaussian basis, and decision trees with acceleration (gradient boosting). Along with the training, a cross-check (with control) was performed on seven blocks at once, in order to obtain a more accurate assessment of the generalization ability of the model. The result of this work is the software implementation of machine learning methods to detect cyber-attacks on websites by identifying their features in network traffic, as well as comparing their effectiveness.

Keywords: cybersecurity, cyber-attack, clustering, data analysis, web resources, network traffic.