Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Аналіз та отримання інформації з Telegram-чатів за допомогою технологій машинного навчання

Виконав:
студент 2 курсу, групи КІТм-21-2
Лебідь В.М.

Спеціальність 123 Комп’ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп’ютерні інтелектуальні технології
Керівник доц. Сердюк Н.М

Допускається до захисту _____
(підпис)

Зав. кафедри _____ О.Г. Руденко
(підпис)

2022 р.

Харківський національний університет радіоелектроніки

Факультет	Комп’ютерної інженерії та управління
Кафедра	Комп’ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність	123 Комп’ютерна інженерія
Тип програми	освітньо-професійна
Освітня програма	Комп’ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«_____»_____2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Лебідю Вадиму Миколайовичу

1. Тема роботи (проекту) Аналіз та отримання інформації з Telegram-чатів за допомогою технологій машинного навчання
затверджена наказом університету від "07" листопада 2022 р. № 1455_Ст
2. Термін подання студентом роботи до екзаменаційної комісії 19 грудня 2022 р.
3. Вихідні дані до роботи (проекту)
1) Документація мови програмування Python
2) Середа розробки PyCharm
3) Документація sqlite3
4) DataSet Russian Toxic Comments
4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної області
2) Аналіз існуючих рішень
3) Аналіз та обрання методу або методів для обробки та аналізу тексту
4) Програмно реалізувати Telegram-бота
5) Відлагодження Telegram-бота
6) Створити інструкцію користувача
7) Висновки
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням кафедри)
_____
_____
6. Консультанти розділів роботи (п.6 включається до завдання за наявністю консультантів згідно до наказу, зазначеному у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Видача та узгодження теми проекту	08.11.2022	виконано
2.	Аналіз предметної області	9.11.2022 -13.11.2022	виконано
3.	Аналіз існуючих рішень	14.11.2022	виконано
4.	Аналіз та обрання методу або методів для обробки та аналізу тексту	15.11.2022 - 17.11.2022	виконано
5.	Програмна реалізація Telegram-бота	18.11.2022 - 26.11.2022	виконано
6.	Відлагодження Telegram-бота	27.11.2022 - 29.11.2022	виконано
7.	Створення інструкції для користувача	30.11.2022	виконано
8.	Оформлення матеріалів атестаційної роботи	01.12.2022 - 4.12.2022	виконано
9.	Перевірка виконаного проекту керівником	5.12.2022	виконано
10.	Попередній захист	17.12.2022	виконано
11.	Захист проекту	20.12.2022	виконано

Дата видачі завдання 08 __листопада__2022 р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Сердюк Н.М._____
(підпис) (посада, ініціали, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 60 сторінок, 17 рисунків, 3 таблиць, 1 додаток, 19 джерел.

Темою кваліфікаційної роботи є аналіз та отримання інформації з Telegram-чатів за допомогою технологій машинного навчання.

Метою кваліфікаційної роботи є розробка багатофункціонального автоматизованого акаунту отримання та аналізу інформації повідомлень користувачів за запланованим сценарієм.

Об'єктом дослідження цієї роботи є реалізація методів обробки тексту та аналізу зібраних даних з використанням методу логістичної регресії та Term frequency - Inverse document frequency (TF-IDF).

Предметом дослідження є аналіз повідомлень за допомогою оцінки важливості слова в контексті категорії, документа або колекції документів.

Відповідно до мети кваліфікаційної роботи виконано поставлені задачі. Виконано аналіз предметної області для виявлення актуальності розробки Telegram-боту. Проаналізовано існуючі рішення вже створених розробок, виявлено недоліки та переваги. Проаналізовано та обрано методи для обробки та аналізу тексту з метою виявлення токсичних коментарів, а саме логістичну регресію та TF-IDF. Програмно реалізовано Telegram-бота на мові програмування Python з використанням середовища програмування PyCharm який може отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів месенджеру Telegram. Спроектовано базу даних, та розроблено структурну схему телеграм боту. Виконано відлагодження Telegram-бота для запобігання появі помилок в його роботі. Створено інструкцію для користувача, в якому зібрані відомості щодо правильного та ефективного використання Telegram-боту.

АНАЛІЗ, ІНФОРМАЦІЯ, ДОСЛІДЖЕННЯ, ПОВІДОМЛЕННЯ, МАШИННЕ НАВЧАННЯ, PYTHON

ABSTRACT

Explanatory note of the qualification work: 60 pages, 17 figures, 3 tables, 1 appendix, 19 sources.

The topic of the qualification work is the analysis and obtaining of information from Telegram chats using machine learning technologies.

The method of qualification work is the development of multifunctional automated account creation and information analysis based on user messages according to the planned scenario.

The object of research of this work is the implementation of methods of text processing and analysis of collected data using the method of logistic regression and term frequency - inverse document frequency (TF-IDF).

The subject of research is the analysis of messages by evaluating the importance of a word in the context of a category, document or collection of documents.

the tasks set for the purpose of the qualification work have also been completed. An analysis of the subject area was carried out for the relevance of Telegram bot development. Existing solutions of already created developments were analyzed, shortcomings and advantages were revealed. Text processing and analysis methods for detecting toxic comments, as well as logistic regression and TF-IDF, were analyzed and selected. The Telegram bot was implemented in the Python programming language using the PyCharm software environment, which can process, analyze and respond accordingly to the messages of users of the Telegram messenger. The database was designed, and the structural diagram of the telegram bot was developed. Debugging of the Telegram bot was performed to prevent errors in its operation. A user manual has been created, which contains information on the correct and effective use of the Telegram bot.

ANALYSIS, INFORMATION, RESEARCH, NOTIFICATION, MACHINE LEARNING, PYTHON

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління
Кафедра Комп’ютерних інтелектуальних технологій та систем

АНОТАЦІЯ

КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Аналіз та отримання інформації з Telegram-чатів за допомогою технологій машинного навчання

(тема)

Виконав:
студент 2 курсу, групи КІТм-21-2
Лебідь В.М.
(прізвище, ініціали)

Спеціальність 123 Комп’ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп’ютерні інтелектуальні технології
Керівник доц. Сердюк Н.М

2022 р.

АНОТАЦІЯ

Лебідь В.М. Аналіз та отримання інформації з Telegram-чатів за допомогою технологій машинного навчання – Магістерська кваліфікаційна робота.

Побудова систем машинного навчання є на сьогоднішній день однією з найпопулярніших, актуальних та сучасних галузей людської діяльності на стику інформаційних технологій, математичного аналізу та статистики. Машинне навчання все глибше проникає в наше життя за допомогою користувацьких продуктів, створених за допомогою методів штучного інтелекту. Очевидно, що ці технології розвиватимуться й надалі, поступово стаючи частиною повсякденної рутини у багатьох галузях людської професійної діяльності. Проте з часів своєї появи, машинне навчання встигло мати численні проблеми, головна з яких – досить висока трудомісткість. Побудова систем машинного навчання потребує величезної кількості часу високопрофесійних фахівців як у сфері штучного інтелекту, так і в тій предметній галузі, до якої ця технологія застосовується [1].

Метою кваліфікаційної роботи є розробка багатофункціонального автоматизованого акаунту отримання та аналізу інформації повідомлень користувачів за запланованим сценарієм.

Об'єктом дослідження цієї роботи є реалізація методів обробки тексту та аналізу зібраних даних з використанням методу логістичної регресії та Term Frequency-Inverse Document Frequency (TF-IDF).

Предметом дослідження є аналіз повідомлень за допомогою оцінки важливості слова в контексті категорії, документа або колекції документів.

Відповідно до мети кваліфікаційної роботи виконано поставлені задачі, а саме:

– виконано аналіз предметної області для виявлення актуальності розробки Telegram-боту для аналізу та отримання коментарів месенджеру Telegram;

– проаналізовано існуючі рішення для виявлення вже створених розробок, їх недоліків та переваг;

– проаналізовано та обрано методи для виявлення токсичних коментарів;

– програмно реалізовано Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– відлагоджено Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– створено інструкцію для користувачів, в якій зібрані відомості щодо правильного та ефективного використання Telegram-боту.

За допомогою пошуку в інтернеті було знайдено та виконано аналіз існуючих рішень, а саме:

– AntiToxic.

– Bot TMBotsbot.

– Twitter Toxic.

Виявилося що таких Telegram-ботів які мають функціонал для розпізнавання токсичних повідомлень досить мало, виходячи з цього розробка Telegram-боту буде актуальною.

Розробку Telegram-боту за допомогою мови програмування Python з використанням IDE PyCharm було розділено на дві частини. Модулю для класифікації токсичних повідомлень, та модулю для роботи Telegram-боту.

Python є однією з найпопулярніших мов програмування для науки про дані, і завдяки його дуже активним розробникам і спільноті з відкритим кодом було розроблено велику кількість корисних бібліотек для наукових обчислень і машинного навчання.

PyCharm - це спеціалізоване інтегроване середовище розробки Python (IDE), яке надає широкий спектр необхідних інструментів для розробників

Python, тісно інтегрованих для створення зручного середовища для продуктивної розробки Python, веб-розробки та науки про дані.

Для створення Telegram-бота який виконує функції отримання та аналізу інформації повідомлень користувачів за запланованим сценарієм необхідно було обрати метод який буде аналізувати коментарі та надавати їм оцінку токсичності.

Токсичний коментар можливо визначити як грубий, неповажний або необґрунтований коментар, який, імовірно, змусить інших користувачів залишити обговорення.

Перед початком вилучення ознак необхідно зменшити розмірність та усунути нерелевантні ознаки, щоб ефективність та продуктивність алгоритмів класифікації була покращена. Попередня обробка допомагає покращити вхідні дані при виконанні машинного навчання або інших статистичних методів, таким чином, попередня обробка спрямована на перетворення пропозицій огляду вільного тексту на набір слів і водночас збагачення їх семантичного значення.

Розглянуто три методи класифікації тексту: мішок слів, логістична регресія та TF-IDF, з яких було обрано та реалізовано два метода які доповнюють один одного, а саме логістичної регресії та TF-IDF.

Логістичні регресійні моделі математично менш складні, ніж інші методи машинного навчання, можуть обробляти великі обсяги даних з високою швидкістю, оскільки вони вимагають меншої обчислювальної потужності, такої як пам'ять і здатність до обробки, які можна використовувати для пошуку відповідей на питання, які мають два або більше кінцевих результатів.

Term frequency, inverse document frequency (TF-IDF) - статистичний показник, який використовується для оцінки важливості слова в контексті категорії, документа або колекції документів. Використовується для аналізу текстових даних.

Як правило, TF-IDF визначається для кожного слова. Чим вище значення цього показника, тим важливіше слово у тих категорії, документа, колекції. При

цьому даний показник також дозволяє врахувати і слова, що широко вживаються, знизивши їх значимість у контексті об'єкта для аналізу.

Визначено що для класифікації коментарів необхідно виконати такі дії:

– завантажити датасет;
– токенізувати, де кожне слово у датасеті розбивається на токени;
– видалити стоп-слова, розділові знаки або небажані маркери;
– виконати лематизацію - скорочення слів до їх кореня;
– векторне уявлення ознак;
– алгоритм навчання.

Для навчання використовується Scikit-learn - один з найпоширеніших і популярних пакетів Python для науки про дані та машинного навчання. Дозволяє виконувати безліч операцій і надає широкий спектр алгоритмів. На додаток до цього, Scikit-Learn пропонує чудову документацію та методи навчання [2].

Для підбиття підсумку роботи методів навчання використовується матриця плутанини. За допомогою якої можна побачити точність класифікатора, порівнюючи фактичні та прогнозовані класи. Точність яка була отримана під час навчання моделі TF-IDF дорівнює 90%. За допомогою логістичної регресії вдалося підвищити точність до 97%.

Для функціонування Telegram-боту розроблено базу даних для збереження стану роботи боту, налаштувань для груп та каналів, а також даних користувачів.

Для інтеграції бази даних використовується модуль Python SQLite3. Це стандартизований Python DBI API 2.0, який забезпечує зрозумілий і простий у використанні інтерфейс для взаємодії з базами даних SQLite.

Під час розробки Telegram-боту виникали помилки :

- неправильна вказівка параметрів для блоку виключення;
- перетин імен із іменами модулями стандартної бібліотеки Python;
- невірно вказаний шлях до розміщеного датасету.

Такі помилки важко виправити, якщо не видно, що відбувається всередині. Відладчик PyCharm дозволяє встановлювати точки зупинки всередині коду, для усунення проблем. За допомогою відладчика PyCharm вдалося виявити помилки та виправити їх.

Практичне використання Telegram-боту в тому що велику кількість повідомлень, які публікують користувачі окремо взятої соціальної мережі, необхідно аналізувати в режимі реального часу з метою модерації, щоб не допустити поширення різної протизаконної чи образливої інформації, загроз та інших токсичних коментарів. Зрозуміло, такий великий обсяг інформації може бути оброблений досить швидко тільки автоматично. Виникає необхідність навчити комп'ютер «розуміти» текст, написаний людиною, що є нетривіальним завданням, навіть під «розумінням» тексту мається на увазі лише його класифікація.

Поставлені в роботі задачі виконані в повному обсязі. Подальший розвиток проекту передбачає реалізацію збереження з класифікацією повідомлень для збільшення датасету для підвищення точності класифікації та підвищення надійності роботи Telegram-боту для опрацювання коментарів 24/7.

АНАЛІЗ, ІНФОРМАЦІЯ, ДОСЛІДЖЕННЯ, ПОВІДОМЛЕННЯ, МАШИННЕ НАВЧАННЯ, PYTHON

Публікації здобувача за темою роботи:

1. Лебідь В.М. Основні проблеми пошуку необхідного контенту при виборі сайтів: Цифровізація науки та сучасні тренди її розвитку: матеріали II міжнародної студентської конференції., м. Миргород 5 листопада 2022. Миргород. С. 90–91.

2. Лебідь В.М. Класифікація токсичних коментарів з використанням онтологій Наука сьогодення: від досліджень до стратегічних рішень: матеріали IV міжнародної студентської конференції., м. Івано-Франківськ 17 червня 2022. Івано-Франківськ. С. 183-185.

3. Classification of objects based on a tree-shaped artificial immune network model. Mykola Korablyov, Oleksandr Fomichov, Natalia Axak / Zbarazh, Ukraine / Editors: Shakhovska, Natalya, Medykovskyy, Mykola O. Springer, 2021. – pp. 160-172 (Scopus)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

СКБД - система керування базами даних

TF-IDF - Term Frequency Inverse Document Frequency

IDLE - Integrated Development and Learning Environment

ID – Identifier

LR – Logistic Regression

MaxEnt - Maximum entropy

ВСТУП

Популярність месенджерів в Інтернеті зростає із розвитком сучасних технологій. На даний час існує досить багато месенджерів WhatsApp, Viber, Discord, Skype, Facebook Messenger, Signal, та інші в яких користувачі пишуть свої повідомлення.

Telegram - це популярна міжплатформна програма обміну повідомленнями, яка широко використовується, оскільки пропонує деякі покращені функції конфіденційності та шифрування, а також підтримує функції великого групового чату. Він також не пов'язаний з іншими платформами соціальних медіа (наприклад, Facebook Messenger і WhatsApp належать Facebook), що робить сервіс більш привабливим для деяких. Програма є багатоплатформною, з версіями програми, доступними для iOS, Android, Windows, Mac і Linux. Також доступ до Telegram можливо отримати з веб-браузера.

Більшість коментарів дійсно приносять користь людям, але деякі високотоксичні коментарі впливають на зростання онлайн-переслідувань та навіть особистих атак. В даний час дуже популярна тема визначення токсичних коментарів.

Токсичний коментар можливо визначити як грубий, неповажний або необґрунтований коментар, який, імовірно, змусить інших користувачів залишити обговорення.

В інтернеті на онлайн каналах класифікація токсичних коментарів зазвичай здійснюється модератором або за допомогою інструментів класифікації тексту. При цьому різним платформам та спільнотам складно підтримувати шановне спілкування і часто доводиться обмежувати коментарі користувачів або згортати проект через нездатність платформи обробляти коментарі користувачів.

Метою кваліфікаційної роботи є розробка багатофункціонального автоматизованого аккаунту отримання та аналізу інформації повідомлень

користувачів за запланованим сценарієм.

Багатофункціональним автоматизованим аккаунтом є Telegram-бот - автоматизований багатофункціональний помічник, який може збирати, аналізувати, та надавати інформацію за запитом згідно зі заздалегідь підготовленим сценарієм.

Об'єктом дослідження цієї роботи є реалізація методів обробки тексту та аналізу зібраних даних з використанням методів логістичної регресії та TF-IDF.

Предметом дослідження є аналіз повідомлень за допомогою оцінки важливості слова в контексті категорії, документа або колекції документів.

Відповідно до мети кваліфікаційної роботи необхідно виконати поставлені задачі, а саме:

– виконати аналіз предметної області для виявлення актуальності розробки Telegram-боту для аналізу та отримання коментарів месенджеру Telegram;

– проаналізувати існуючі рішення для виявлення вже створених розробок, їх недоліків та переваг;

– проаналізувати та обрати метод або методи для обробки та аналізу тексту для виявлення токсичних коментарів;

– програмно реалізувати Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– відлагодити Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– створити інструкцію для користувача - довідково-інформаційний документ, в якому зібрані відомості щодо правильного та ефективного використання Telegram-боту.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Зростання популярності онлайн-платформ, що дозволяють користувачам спілкуватися один з одним, ділитися думками щодо різних подій, залишати коментарі, підштовхнуло до розвитку алгоритмів обробки природної мови. Десятки мільйонів повідомлень на день, які публікують користувачі окремо взятої соціальної мережі, необхідно аналізувати в режимі реального часу або близько до того з метою модерації, щоб не допустити поширення різної протизаконної чи образливої інформації, загроз та інших видів токсичних коментарів. Зрозуміло, такий великий обсяг інформації може бути оброблений досить швидко тільки автоматично. Виникає необхідність навчити комп'ютер «розуміти» текст, написаний людиною, що є нетривіальним завданням, навіть під «розумінням» тексту мається на увазі лише його класифікація. Бурхливий розвиток технологій машинного навчання зумовило повсюдне впровадження нових алгоритмів. Багато завдань, у тому числі й задачі обробки природної мови, які довгі роки вважалося практично неможливо вирішити, зараз цілком успішно вирішуються з використанням технологій машинного навчання.

Машинне навчання - це підгалузь штучного інтелекту, яка широко визначається як здатність машини імітувати інтелектуальну поведінку людини.

Машинне навчання, безсумнівно, є однією з найбільш захоплюючих підмножин штучного інтелекту. Важливо розуміти, що змушує машинне навчання працювати і таким чином, як його можна використовувати в майбутньому.

Процес машинного навчання починається з введення навчальних даних в обраний алгоритм. Нові вхідні дані подаються в алгоритм машинного навчання для перевірки правильності роботи алгоритму. Прогноз і результати потім перевіряються один проти одного.

Якщо прогноз і результати не збігаються, алгоритм багаторазово перенавчається, поки з даних не отримано бажаний результат. Це дає

можливість алгоритму машинного навчання постійно вчитися самостійно і виробляти оптимальну відповідь, поступово збільшуючись в точності з плином часу [19].

В Інтернеті компанією Jigsaw, яка займається проблемами безпеки, спільно з Google було проведено конкурс, метою якого було створення алгоритму, що вирішує завдання детектування токсичних коментарів [3]. Це говорить про актуальність, а також низький рівень досліджень цієї проблеми.

Метою кваліфікаційної роботи є розробка багатофункціонального автоматизованого аккаунту отримання та аналізу інформації повідомлень користувачів за запланованим сценарієм.

Об'єктом дослідження цієї роботи є реалізація методів обробки тексту та аналізу зібраних даних з використанням методів логістичної регресії та TF-IDF.

Предметом дослідження є аналіз повідомлень за допомогою оцінки важливості слова в контексті категорії, документа або колекції документів.

Відповідно до мети кваліфікаційної роботи необхідно виконати поставлені задачі, а саме:

– виконати аналіз предметної області для виявлення актуальності розробки Telegram-боту для аналізу та отримання коментарів месенджеру Telegram;

– проаналізувати існуючі рішення для виявлення вже створених розробок, їх недоліків та переваг;

– проаналізувати та обрати метод або методи для обробки та аналізу тексту для виявлення токсичних коментарів;

– програмно реалізувати Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– відлагодити Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– створити інструкцію для користувача - довідково-інформаційний документ, в якому зібрані відомості щодо правильного та ефективного використання Telegram-боту.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

За допомогою пошуку від компанії Google вдалось виявити декілька розробок.

AntiToxic Bot - бот, що розпізнає токсичність користувачів у телеграм-чатах.

Розроблено на Python з використанням бібліотек pytorch та pyTelegramBotApi. Дозволяє збирати та виводити статистику токсичних користувачів, якщо їх рейтинг перевищує встановлений, використовується датасет з 14000 повідомленнями. Використовується архітектура нейромережі – CNN + GRU.

Як вказує розробник: «Бот більш-менш справляється зі своїм завданням, але іноді нормальне повідомлення визначає як токсичне. Ця проблема вирішується, знаходженням більшої датасету і ускладненням архітектури нейромережі, але при ускладненні архітектури ресурсів Google Colab`а може не вистачити.»

TMBotsbot, даний бот виявився з дещо більшим функціоналом та працює як в групах та каналах, але функціонал для каналів не має ніякого відношення до виявлення токсичних коментарів, а бо ж спаму.

Функціонал для груп:

– при додаванні бота @TMBotsbot до групи з правами адміністратора, він захищатиме її від спаму та мату;

– антиспам - видаляє повідомлення, що містять посилання, символи @ і #, ієрогліфи, якщо у користувача було менше 5 повідомлень у вашій груп;

– антимат – видаляє повідомлення, що містять нецензурні слова та блокує користувача на деякий час. Щоразу час блокування подвоюється;

– статистика - виводить статистику за кількістю повідомлень для вашої групи;

– виклик статистики доступний лише адміністраторам за командою /tmstats у вашій групі.

Функціонал для каналів:

– при додаванні бота @TMBotsbot до каналу з правами адміністратора, він буде додавати кнопки лайків до всіх постів.

Twitter Toxic, як заявляє розробник: «Бот намагається оцінити профіль твітера і надати приблизний емоційний колір.». Під час проведення аналізу існуючих рішень, виявився непрацездатним.

При виконанні аналізу існуючих рішень виявилося що таких Telegram-ботів які мають функціонал для розпізнавання токсичних повідомлень досить мало, виходячи з цього розробка Telegram-боту буде актуальною.

3 АЛГОРИТМИ ТА МЕТОДИ КЛАСИФІКАЦІЇ ТЕКСТУ

Розробку Telegram-боту за допомогою мови програмування Python з використанням IDE PyCharm можливо розділити на дві частини. Модулю для класифікації токсичних повідомлень, та модулю для роботи Telegram-боту.

Для створення Telegram-бота який буде виконувати отримання та аналіз інформації повідомлень користувачів за запланованим сценарієм необхідно обрати метод який буде аналізувати коментарі та надавати їм оцінку токсичності.

В роботі [4] наголошується на тому що найбільш ранні відповідні роботи з класифікації токсичних коментарів були опубліковані в 2018 році, а їх кількість значно зросла в 2019 році. Більшість первинних досліджень використовували один або два набори даних з позначеними токсичними коментарями для машинного навчання під наглядом. Таблиця 3.1 показує в скількох первинних дослідженнях був використаний конкретний метод машинного навчання.

Таблиця 3.1 Методи машинного навчання, що використовуються в первинних дослідженнях

Machine learning method	Метод машинного навчання	Кількість робіт
Convolutional neural network	Згортувальна нейронна мережа	12
Logistic regression classifier	Класифікатор логістичної регресії	9
Bidirectional long short-term memory	Двонаправлена довга короткочасна пам'ять	8
Bidirectional Gated Recurrent Unit Networks	Двонаправлені закриті рекурентні мережі блоків	6
Long Short-Term Memory	Довгострокова короткочасна пам'ять	5
Support Vector Machine	Підтримка векторної машини	5

Продовження таблиці 3.1 - Методи машинного навчання, що використовуються в первинних дослідженнях

Machine learning method	Метод машинного навчання	Кількість робіт
Bidirectional Encoder Representations from Transformers	Двонаправлені подання кодувальника від трансформерів	4
Naive Bayes classifier	Наївний байєсівський класифікатор	4
Capsule Network	Капсульна мережа	3
Random Forest	Випадковий ліс	2
Decision tree	Дерево рішень	2
KNN classification	Класифікація KNN	2
Gated Recurrent Unit	Керований рекурентний блок	2
Extreme Gradient Boosting	Екстремальне підвищення градієнта	2
Recurrent Neural Network	Рекурентна нейронна мережа	2
Bi-directional Gated Recurrent Unit Long Short Term Memory	Двонаправлена закрита рекурентна одиниця довготривалої короткочасної пам'яті	1
Gaussian Naive Bayes	Гауссовий Наївний Байєс	1
Genetic Algorithms	Генетичні алгоритми	1
Partial Classifier Chains	Ланцюги часткового класифікатора	1

Проаналізувавши методи машинного навчання вказані в [4], в первинних дослідженнях найбільш використовуваними є глибокі нейронні мережі, але найкраще підходять більш прості і швидкі методи, такі як логістична регресія та TF-IDF.

При виконанні кваліфікаційної роботи розглянуто три методи класифікації тексту, до них відносяться:

– мішок слів;

– логістична регресія;

– TF-IDF.

Мішок слів - це спрощене уявлення тексту, яке показує, які слова зустрілися у тексті, але не враховує їх порядок. Таке уявлення легко запрограмувати, зручне для використання в завданнях автоматичної обробки тексту. Незважаючи на свою простоту, виявляється досить корисним і дозволяє успішно вирішувати такі завдання як класифікація тексту, тобто віднесення тексту до певної групи/категорії.

Подання мішка слів - це таблиця з числами, де стовпці таблиці - унікальні слова, а рядки - документи колекції. У осередках таблиці перебуває число входжень слова до документа. Це означає що, у кожному рядку вийде набір чисел, що характеризує склад документа.

Нехай є два тексти: «Це були найкращі часи» та «Це був найгірший час». В обох пропозиціях зустрічається сумарно 5 різних слів, якщо привести до початкових форм: «Це», «Бути», «Найкращий», «Найгірший», «Час». Це буде наш словник.

Виділимо зустрінуті слова зі словника у текстах: у першому зустрілися [«Це», «Бути», «Кращий», «Час»], а в другому - [«Це», «Бути», «Гірший», «Час»].

Векторне уявлення мішка слів для першого тексту буде [1 1 1 0 1], де нуль стоїть на місці елемента «Гірший», оскільки воно не зустрілося в ньому, а для другого - [1 1 0 1 1], де нуль на місці слова "Кращий". Так було виконано перехід до спрощеного машино читаного представлення двох текстів.

У реальних завданнях все дедалі складніше. Щоб не смітити в таблиці, з тексту видаляють службові слова. Слова призводять не обов'язково до початкової форми, але іноді обрізають, залишаючи лише граматичну основу. Тому правильніше називати їх не словами, а «токенами». Іноді стовпці позначають не окремі слова, а пари слів, що йдуть (біграми) або трійки (триграми).

Найчастіше в осередках пишуть не абсолютний показник «слово зустрілося 15 разів», а відносний показник зі статистики: він називається TF-IDF та описує важливість слова для класифікації тексту.

Мішок слів – корисний інструмент (або модель), який використовується для різних завдань, наприклад, для класифікації текстів на спам/не спам, визначення схожості текстів та як спрощений спосіб подання текстів для різних завдань машинного навчання [5].

Логістична регресія, незважаючи на свою назву, є швидше лінійною моделлю класифікації, ніж регресією. Логістична регресія також відома у літературі як логітрегресія, класифікація з максимальною ентропією або логарифмічний лінійний класифікатор. У цій моделі ймовірності, що описують можливі наслідки одного випробування, моделюються за допомогою логістичної функції.

Логістична регресія - це метод аналізу даних, який використовує математику для пошуку зв'язків між двома факторами даних. Цей зв'язок потім використовується для прогнозування значення одного з цих факторів на основі іншого. Прогноз зазвичай має кінцеве число результатів, таких як "так" або "ні".

Логістична регресія є важливим методом у сфері штучного інтелекту та машинного навчання. Моделі машинного навчання - це програми, які можна навчити виконувати складні завдання обробки даних без участі людини.

Логістичні регресійні моделі математично менш складні, ніж інші методи машинного навчання, можуть обробляти великі обсяги даних з високою швидкістю, оскільки вони вимагають меншої обчислювальної потужності, такої як пам'ять і здатність до обробки, які можна використовувати для пошуку відповідей на питання, які мають два або більше кінцевих результатів [18].

Term frequency, inverse document frequency (TF-IDF) - статистичний показник, який використовується для оцінки важливості слова в контексті категорії, документа або колекції документів. Використовується для аналізу текстових даних.

Як правило, TF-IDF визначається для кожного слова. Чим вище значення цього показника, тим важливіше слово у тих категорії, документа, колекції. При цьому даний показник також дозволяє врахувати і слова, що широко вживаються, знизивши їх значимість у контексті об'єкта для аналізу.

Формула для визначення показника має такий вигляд:

$$TF - IDF = TF * IDF, \quad (3.1)$$

де:

- TF – частота слова у конкретній категорії/документі/колекції (залежно від того, які дані аналізуються);
- IDF – зворотна частота документа (популярність слова).

Частота слова в категорії визначається за такою формулою:

$$TF = \frac{n_t}{\sum_{i=1}^{k} n_i}, \quad (3.2)$$

де:

- n_t кількість окремих слів у категорії/документі/колекції;
- $\sum_{i=1}^{k} n_i$ загальна кількість усіх слів у категорії/документі/колекції.

Зворотна частота документа (також часто називають інверсією частоти) визначається за такою формулою:

$$IDF = \ln\left(\frac{n_c}{\sum_{j=1}^{m} n_j}\right), \quad (3.3)$$

де:

- n_c — кількість категорій/документів/колекцій всього;
- $\sum_{j=1}^{m} n_j$ кількість категорій/документів/колекцій у яких містить слово, що цікавить.

Перший компонент формули 3.1 для обчислення TF-IDF практично завжди не змінюється. Метод розрахунку частоти може різнитися залежно від специфіки завдання, обсягу даних для аналізу, кількості категорій. При цьому основний зміст показника залишається без змін і він дозволяє знизити «вагу» слів, що широко вживаються.

При аналізі текстових даних метрику TF-IDF краще розраховувати після проведення процесів токенізації, а також лематизації [6].

Токенізація - це процес поділу фрагмента тексту на менші одиниці, які називаються токенами.

Лематизація - процес приведення словоформи до леми - її нормальної (словарної) форми. Алгоритми лематизації на мові python реалізовані, в бібліотеці NLTK.

Для створення Telegram-боту було обрано метод Term Frequency-Inverse Document Frequency (TF-IDF), та логістичної регресії (LR).

Мета вилучення ознак полягає в тому, щоб зменшити розмірність та усунути нерелевантні ознаки, щоб ефективність та продуктивність алгоритмів класифікації була покращена. Попередня обробка допомагає покращити вхідні дані при виконанні машинного навчання або інших статистичних методів, таким чином, попередня обробка спрямована на перетворення пропозицій огляду вільного тексту на набір слів і водночас збагачення їх семантичного значення. У цьому процесі беруть участь три під задачі. Вони є частиною мови, тегування та осмисленим вибором слів.

Отже, для класифікації коментарів необхідно виконати такі дії:

– завантажити датасет;
– токенізувати, де кожне слово у датасеті розбивається на токени;
– видалити стоп-слова, розділові знаки або небажані маркери;
– виконати лематизацію - скорочення слів до їх кореня;
– векторне уявлення ознак;
– алгоритм навчання.

Схематичне зображення роботи модулю для класифікації токсичних повідомлень представлена на рисунку 3.1.

Рисунок 3.1 - Класифікації токсичних повідомлень

Для підбиття підсумку роботи методів використовується матриця плутанини. Обчислення матриці плутанини може дати краще уявлення у тому,

що робить класифікаційна модель, яких типів помилок вона допускає. Кроки для обчислення матриці плутанини такі:

– отримати тестовий набір даних або перевірочний набір даних з очікуваними значеннями результатів;

– зробити прогноз для кожного рядка у тестовому наборі даних;

– виходячи з очікуваних результатів і прогнозів, підрахувати кількість правильних прогнозів для кожного класу.

За допомогою матриці плутанини можна побачити точність класифікатора, порівнюючи фактичні та прогнозовані класи. Матриця двійкової плутанини складається із квадратів зображених на рисунку 3.2.

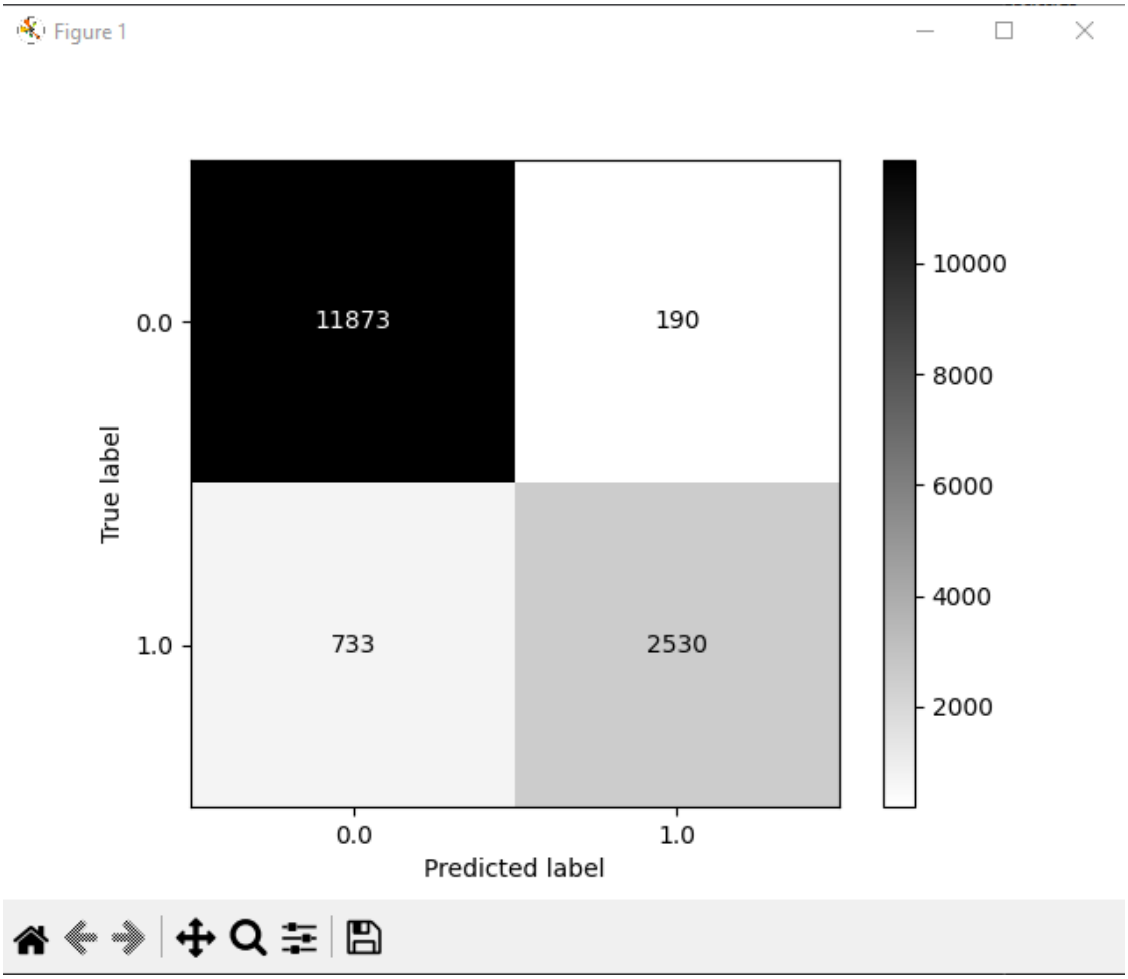

Рисунок 3.2 – Матриця плутанини

Матриця плутанини - це не що інше, як таблиця з двома вимірами, а саме. «Фактичні» та «Предсказані», і, крім того, обидва виміри мають «Справжні

позитиви (TP)», «Справжні негативи (TN)», «Помилкові позитиви (FP)», «Помилкові негативи (FN)», як показано в таблиці 3.2.

Таблиця 3.2 – матриця плутанини

		Актуальне значення	
		1	0
Передбачуване значення	1	True Negative	False Positive
	0	False Negative	True Positive

Розрахувати тест точності з матриці плутанини можливо за формулою:

$$\text{Точність} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.4)$$

де:

– TP: True Positive: прогнозовані значення, правильно прогнозовані як фактичні позитивні;

– FP: Передбачені значення неправильно передбачають фактичний позитивний результат тобто негативні значення прогнозуються як позитивні;

– FN: False Negative: позитивні значення прогнозуються як негативні;

– TN: True Negative: прогнозовані значення, правильно прогнозовані як фактичні негативні.

Підставивши отримані значення (Рис. 3.2) з матриці плутанини у формулу отримаємо точність 90%.

Другою частиною є модуль для безпосередньої роботи Telegram-боту який забезпечує підключення до Telegram мережі, керування ботом, отримання повідомлень, видачі результатів за сценаріями та роботи з базою даних. Структуру боту вказано в розділі 4.5, лістинги боту розміщені в додатку А.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ TELEGRAM-БОТУ

4.1 Мова програмування Python

Python (найчастіше вживане прочитання — «Па́йтон», запозичено назву з британського шоу Монті Пайтон) — інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднування наявних компонентів. Python підтримує модулі та пакети модулів, що сприяє модульності та повторному використанню коду. Інтерпретатор Python та стандартні бібліотеки доступні як у скомпільованій, так і у вихідній формі на всіх основних платформах. В мові програмування Python підтримується кілька парадигм програмування, зокрема: об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована [7].

Серед основних її переваг можна назвати такі:

– чистий синтаксис (для виділення блоків слід використовувати відступи);

– переносність програм (що властиве більшості інтерпретованих мов);

– стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);

– можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);

– стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане мовою Python;

– зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної

величини, у діалоговому режимі може використовуватися як потужний калькулятор);

– відкритий код (можливість редагувати його іншими користувачами).

Python має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Python, динамічна обробка типів, а також те, що це інтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Python є однією з найпопулярніших мов програмування для науки про дані, і завдяки його дуже активним розробникам і спільноті з відкритим кодом було розроблено велику кількість корисних бібліотек для наукових обчислень і машинного навчання.

Незважаючи на те, що продуктивність інтерпретованих мов, таких як Python, для інтенсивних обчислювальних завдань поступається мовам програмування нижчого рівня, були розроблені бібліотеки розширень, такі як NumPy і SciPy, які базуються на реалізаціях нижнього рівня Fortran і C для швидких векторизованих операцій на багатовимірні масиви. Для програмування завдань машинного навчання ми буде використано scikit-learn, яка наразі є однією з найпопулярніших і доступних бібліотек машинного навчання з відкритим кодом [8].

4.2 Середовище розробки PyCharm

PyCharm - це спеціалізоване інтегроване середовище розробки Python (IDE), яке надає широкий спектр необхідних інструментів для розробників Python, тісно інтегрованих для створення зручного середовища для продуктивної розробки Python, веб-розробки та науки про дані.

PyCharm доступний в трьох версіях:

– community (безкоштовна та з відкритим вихідним кодом): Для розумної та інтелектуальної розробки Python, включаючи допомогу в написанні коду, рефакторингу, візуальному налагодженні та інтеграції контролю версій;

– professional (платний): для професійної розробки на Python, веб-розробки та науки про дані, включаючи допомогу в написанні коду, рефакторінг, візуальне налагодження, інтеграцію контролю версій, віддалене налаштування, розгортання, підтримку популярних веб-фреймворків, таких як Django та Flask, підтримку баз даних, дослідницькі інструменти (включаючи підтримку Jupyter) та інструменти великих даних;

– edu (безкоштовний та з відкритим вихідним кодом): Вивчати мови програмування та пов'язані з ними технології за допомогою інтегрованих освітніх інструментів;

– щоб почати розробку Python за допомогою PyCharm, потрібно завантажити та встановити Python з сайту python.org, залежно від платформи.

PyCharm підтримує наступні версії Python:

- Python 2: версія 2.7;
- Python 3: Від версії 3.6 до версії 3.11.

Крім того, в версії Professional можна розробляти додатки Django, Flask і Pyramid. Він повністю підтримує HTML (включаючи HTML5), CSS, JavaScript і XML: ці мови інтегровані в IDE через плагіни і включені за замовчуванням. Підтримка інших мов і фреймворків також може бути додана за допомогою плагінів (зайдіть в Налаштування | "Плагіни" або "PyCharm" | «Налаштування» | "Плагіни" для користувачів macOS, щоб дізнатися більше або налаштувати їх під час першого запуску IDE).

PyCharm — це кросплатформна IDE, яка працює на Windows, macOS та Linux [9].

4.3 Використані бібліотеки

Pandas - програмна бібліотека, написана для мови програмування Python для маніпулювання даними та їхнього аналізу. Вона, зокрема, пропонує структури даних та операції для маніпулювання чисельними таблицями та часовими рядами. pandas є вільним програмним забезпеченням, що випускається за трипунктовою ліцензією BSD. Ця назва походить від терміну «панельні дані[en]» (англ. panel data), який в економетрії позначає багатовимірні структуровані набори даних.

Matplotlib - бібліотека мови програмування Python для візуалізації даних з двовимірною та тривимірною графікою. Отримані зображення можна використовувати як ілюстрації в публікаціях. Matplotlib був написаний і підтриманий в першу чергу Джоном Хантером і поширюється на умовах ліцензії, подібної до BSD. Зображення, згенеровані в різних форматах, можуть використовуватися в інтерактивній графіці, в наукових публікаціях, графічному інтерфейсі користувача, веб-додатках, де потрібно побудова графіків. У документації автор зізнається, що Matplotlib починав з імітації графічних команд MATLAB, але є від нього самостійним проектом. Версія 2.1.1 - остання стабільна - вимагає Python версії 2.7 або 3.4 і вище і NumPy версії 1.7.1 і вище. Бібліотека Matplotlib побудована на принципах ООР, але має процедурний інтерфейс, який забезпечує аналоги команд MATLAB.

Scikit-learn (також відома як sklearn або scikits.learn) — це безкоштовна програмна бібліотека машинного навчання для мови програмування Python, яка надає функціональність для створення та тренування різноманітних алгоритмів класифікації, регресії та кластеризації, таких як лінійна регресія, random forest, градієнтний бустинг[en], і працює у зв'язці з бібліотеками NumPy та SciPy. Scikit-learn є однією з найбільш популярних бібліотек машинного навчання.

Модуль re - забезпечує операції узгодження шаблонів регулярних виразів, подібних до тих, що знаходяться в Perl.

Шаблони регулярних виразів і рядків пошуку можуть бути як рядками Юнікоду, так і 8-бітовими рядками. однак рядки Юнікоду та 8-бітні рядки не

можна змішувати. Тобто ви не можете зіставити рядок Юнікоду з байтовим шаблоном регулярних виразів або навпаки. Аналогічно, якщо ви замінюєте на основі регулярного виразу, рядок заміни повинна бути однотипною. що і регулярний вираз, і рядок пошуку.

Регулярні вирази використовують символ зворотної риски '\' для вказівки спеціальних форм або для дозволу використання спеціальних символів, що використовуються в шаблонах пошуку, не називаючи їх особливого значення. Це суперечить тому, що Python використовує один і той же символ з тією ж метою в рядкових літералах. Наприклад, щоб зіставити зворотну скісну риску літерального '\', вам може знадобитися написати '\\\\' як рядок шаблону пошуку, оскільки регулярний вираз повинен бути \\, і кожна зворотна скісна риска повинна бути виражена як \\ всередині звичайного рядка Python буквально.

Рішення полягає у використанні необроблених позначень рядків Python для шаблонів регулярних виразів. Зворотна скісна риска не обробляється особливим чином у рядку, буквальному з префіксом 'r'. Отже, r"\n" є двосимвольним рядком, що містить '\' і 'n', тоді як '\n' - це однозначний рядок, що містить новий рядок. Як правило, шаблони пошуку будуть виражені в коді Python за допомогою цього необробленого запису рядка.

Важливо зазначити, що більшість операцій регулярних виразів доступні у вигляді функцій модульного рівня та методів для скомпільованих регулярних виразів. Функції модуля re не вимагають попередньої компіляції об'єкта регулярного виразу, але не дозволяють деяким параметрам тонкої настройки шаблону знайти регулярний вираз [10].

Pymorphy2 – морфологічний аналізатор, написаний мовою Python і використовує словники з OpenCorpora. Він працює під Python і розповсюджується за ліцензією MIT.

Морфологічний аналіз – це визначення характеристик слова на основі того, як це слово пишеться. При морфологічному аналізі немає інформації про сусідніх словах [11].

Модуль tqdm призначений для швидкого та розширюваного впровадження індикаторів виконання (progressbar) у зовнішні інтерфейси програм на Python, надаючи кінцевим користувачам візуальну індикацію ходу обчислень чи передачі. Він також буде корисний з метою налагодження як інструменту профілювання, так і як спосіб відображення інформації журналу ітеративної задачі. Завдяки простоті використання бібліотека є ідеальним кандидатом для включення в освітні курси Python [12].

NLTK є провідною платформою для створення програм Python для роботи з даними людської мови. Він надає прості у використанні інтерфейси для більш ніж 50 корпусів і лексичних ресурсів, таких як WordNet, а також набір бібліотек для обробки тексту для класифікації, токенізації, сформування основи, тегування, синтаксичного аналізу та семантичного міркування, оболонки для індустріальних бібліотек NLP, і активний дискусійний форум.

Завдяки практичному посібнику, який представляє основи програмування разом із темами з комп'ютерної лінгвістики, а також вичерпну документацію API, NLTK підходить як для лінгвістів, інженерів, студентів, викладачів, дослідників, так і для промислових користувачів. NLTK доступний для Windows, Mac OS X і Linux. Найкраще те, що NLTK є безкоштовним проектом із відкритим кодом, керованим спільнотою.

NLTK називають «чудовим інструментом для навчання та роботи в комп'ютерній лінгвістиці з використанням Python» і «дивовижною бібліотекою для гри з природною мовою» [13].

Обробка природної мови за допомогою Python забезпечує практичний вступ до програмування для обробки мови. Написана творцями NLTK, вона веде читача через основи написання програм на Python, роботи з корпусами, категоризації тексту, аналізу лінгвістичної структури тощо.

Бібліотека pyTelegramBotAPI (telebot) на Python. Необхідна для взаємодії з Telegram Bot Api (являє собою HTTP-інтерфейс для роботи з ботами в Telegram.

Модуль Python SQLite3 використовується для інтеграції бази даних SQLite з Python. Це стандартизований Python DBI API 2.0, який забезпечує

зрозумілий і простий у використанні інтерфейс для взаємодії з базами даних SQLite. Немає необхідності встановлювати цей модуль окремо, оскільки він постачається разом із Python після версії 2.5x [14].

Використаний датасет можливо знайти за адресою: https://www.kaggle.com/code/teserakt/russian-toxic-comments-preprocessing-baseline/data

4.4 Проектування бази даних

Для функціонування Telegram-боту необхідно розробити базу даних в якій буде збережена інформація про користувачів та групи які використовують даний бот.

При розробці бази даних потрібно дотримуватись вимог проектування:

– функціональної повноти;
– мінімальної надмірності;
– цілісності бази даних;
– цілісністю таблиці;
– несуперечністю;
– безпекою;
– відновлюваністю;
– узгодженістю;
– ефективністю;
– логічної і фізичної незалежності;
– розширюваність або відкритість [15].

Кожна група має свій унікальний ідентифікатор, так само як і користувача месенджеру-Telegram. Користувачі можуть належати до декількох груп одночасно.

Сутностями є:

– ID-групи;
– стан боту;

- рівень токсичності;
- ID-користувача;
- логін користувача.

На рисунку 4.1 показана кратність відношень і властивості (атрибути) сутностей.

Рисунок 4.1 - ER-діаграма бази даних

Структура даних зображена в таблиці 4.1. Тут представлені сутності з їх атрибутами і типами полів. У таблиці детально описано, які з атрибутів є ключовими полями цієї сутності.

Таблиця 4.1 Таблиця сутностей

Найменування сутності	Опис	Найменування атрибуту	Тип даних	Ключове поле
Групи	Зберігає інформацію про групи	ID-групи	BIGINT	Так
		Стан боту	BOOLEAN	Ні
		Рівень токсичності	DOUBLE	Ні
Користувачі-Групи	Зв’язуюча таблиця	ID-користувача	BIGINT	Так
		ID-групи	BIGINT	Так

Користувачі	Зберігає інформацію про користувачів	ID-користувача	BIGINT	Так
		Логін користувача	CHAR	Ні

4.5 Структурна схема Telegram-боту

В розробленому проекті Telegram-боту є 7 елементів (Рис. 4.2):

– папка venv. Модуль venv підтримує створення полегшених «віртуальних середовищ», кожне з яких має власний незалежний набір пакетів Python, встановлених у своїх site каталогах. Віртуальне середовище створюється поверх існуючої установки Python, відомої як «базовий» Python віртуального середовища, і за бажання може бути ізольована від пакетів у базовому середовищі, тому доступні лише ті, які явно встановлені у віртуальному середовищі. При використанні віртуального середовища звичайні інструменти установки, такі як pip, встановлюватимуть пакети Python у віртуальне середовище без необхідності робити це явно [16];

– файл analizator.py містить в собі всю логіку яка необхідна для аналізу повідомлень на токсичність;

– файл auth_data.py зберігає в собі токен доступу до Telegram-боту;

– файл database.py приймає на себе роль управління базою даних, такі яка внесення даних, обробка;

– файл db.db зберігає основну інформацію про роботу Telegram-бота, користувачів, та груп;

– файл labeled.csv основний датасет Telegram-боту, зберігає інформацію з розміченими токсичними коментарями, даний датасет можливо редагувати та доповнювати;

– файл pyvenv.cfg зберігає конфігурацію для конкретного середовища на основі існуючої версії Python у системі;

– файл main.py, головний файл запуску Telegram-боту, зберігає код

для обробки даних Telegram-боту.

Лістинги файлів Telegram-боту розміщені в додатку А.

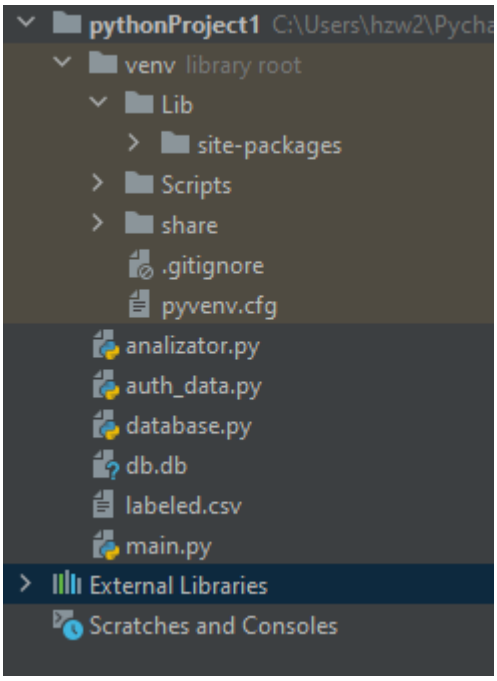

Рисунок 4.2 – Структура папки з проектом

5 ВІДЛАГОДЖЕННЯ КОДУ TELEGRAM-БОТУ

Відладчик PyCharm - дуже ефективний засіб. Перш ніж приступати до його використання, слід ознайомитися з базовими термінами, такими як відладчик, налагодження і режим відладки.

Термін налагодження може мати різні значення, але в першу чергу він означає усунення помилок в коді. Робиться це по-різному. Наприклад, налагодження може виконуватися шляхом перевірки коду на наявність помилок або за допомогою аналізатора коду. Код можна налагоджувати за допомогою профілювальника продуктивності. Крім того, налагодження може здійснюватися за допомогою відладчика.

Відладчик - це вузькоспеціалізоване засіб розробки, яке приєднується до працюючого Telegram-боту і дозволяє перевіряти код.

Деякі розробники досі налагоджують програми, використовуючи оператори print, тому що концепція складна, а PDB виглядає страшно. Графічний налагоджувач PyCharm робить процес максимально простим, наочно візуалюючи налагодження у зручному форматі. З PyCharm ви легко освоїтеся і зможете швидко почати користуватися основними функціями налагодження.

Звичайно, PyCharm вміє налагоджувати код, який виконується на локальному комп'ютері у системному оточенні, virtualenv, Anaconda чи Conda. З PyCharm Professional Edition ви також зможете налагоджувати код, який запускається в Docker-контейнері, на віртуальній машині або віддаленому хості через SSH.

При роботі з шаблонами іноді в них може виникнути помилка. Такі помилки важко виправити, якщо не видно, що відбувається всередині. Відладчик PyCharm дозволяє встановлювати точки зупинки всередині шаблонів Django та Jinja2, щоб ви могли легко позбутися проблем.

Розробка через тестування (TDD) передбачає дослідження коду під час написання тестів. Використовуйте для цього наладчик, встановлюючи точки зупинки у контексті, який ви досліджуєте.

PDB - чудовий інструмент, але необхідність змінювати код може призвести до випадкового запису викликів pdb.set_trace() у ваш Git-репозиторій.

Точки зупинки є у всіх налагоджувачах, але тільки деякі з них можуть запропонувати гнучкі точки зупинки. Вам, напевно, доводилося багато разів натискати Continue, поки ви нарешті не дісталися до ітерації циклу, в якій виникає помилка. З умовними точками зупинки PyCharm у цьому немає потреби.

Найчастіше все, що потрібно зробити, це подивитися, яке значення має певна змінна під час виконання коду. Ви можете налаштувати точки зупинки PyCharm таким чином, що вони не зупинятимуть виконання коду, а лише підготують для вас повідомлення.

Винятки можуть зіпсувати ваш день, тому відладчик PyCharm вміє зупинятися на винятках, навіть якщо ви не зовсім впевнені, звідки вони взялися.

Щоб допомогти контролювати процес налагодження, у PyCharm передбачено спеціальне вікно, в якому ви можете побачити всі точки зупинки та відключити деякі з них, використовуючи прапорці. Крім того, можна тимчасово відключити всі точки зупинки, поки вони знову вам не знадобляться.

Як тільки PyCharm досягне точки зупинки, ви побачите всі значення змінних прямо у коді. Щоб було простіше зрозуміти, які значення змінилися з моменту останнього спрацьовування точки зупинки, значення, що змінилися, підсвічуються.

Ви можете налаштувати відображення змінних, додавши watches (відстеження значень). Незалежно від їхньої складності, PyCharm покаже саме те, що вам потрібно.

Якщо ви хочете знати, як працює ваш код, не обов'язково скрізь ставити точки зупинки. Ви можете стежити за всім, що відбувається, крокуючи кодом за допомогою відладчика.

У деяких випадках найпростіший спосіб відтворення – примусово встановити певне значення для змінної. PyCharm пропонує вирахувати вираз

для швидких змін, а також використовувати консоль, якщо ви хочете краще контролювати процес. Консоль може використовувати оболонку ipython, якщо вона встановлена.

Для налагодження Python 3.6 відладчик PyCharm - найшвидший з існуючих, навіть швидше, ніж PDB. Це означає, що ви завжди можете виконувати свій код у налагоджувачі під час розробки і легко додавати точки зупинки, коли потрібно. Головне переконатися, що ви натиснули на install, коли PyCharm запитує, чи потрібно встановлювати прискорення Cython [17].

6 ІНСТРУКЦІЯ КОРИСТУВАЧА

Під час виконання підготовки Telegram-боту до роботи, адміністратор боту буде проінформований про його стан роботи.

На рисунку 6.1 зображено розподіл завантаженого датасету з класифікацією коментарів, що дозволяє візуально отримати інформацію яку кількість процентів складають токсичні та нетоксичні коментарі, де 0 – не токсичні коментарі, 1 – токсичні коментарі.

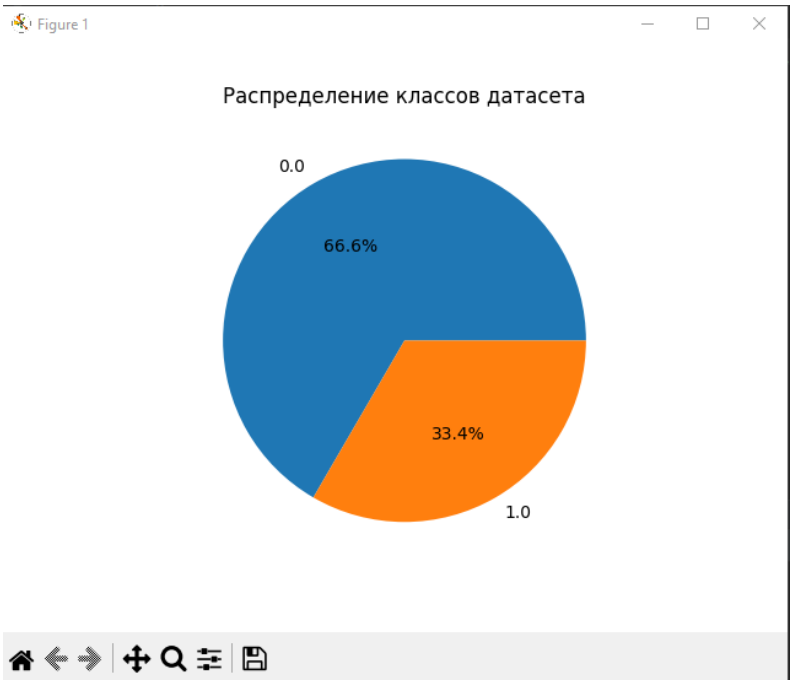

Рисунок 6.1 – Класифікація датасету

Хід машинного навчання можливо побачити на рисунку 6.2.

```
C:\Users\hzw2\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\hzw2\PycharmProjects\pythonProject1\main.py
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hzw2\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14415 entries, 0 to 14414
Data columns (total 2 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   comment  14415 non-null  object
 1   toxic    14415 non-null  float64
dtypes: float64(1), object(1)
memory usage: 225.4+ KB
 81%|████████  | 11631/14415 [02:50<00:47, 58.36it/s]
```

Рисунок 6.2 – Робота Telegram-боту

Як тільки навчання буде завершено виведеться матриця плутанини (Рис. 6.3) для узагальнення продуктивності алгоритму. Обчислення матриці плутанини може дати краще уявлення про те, що модель правильно класифікує завантажений датасет.

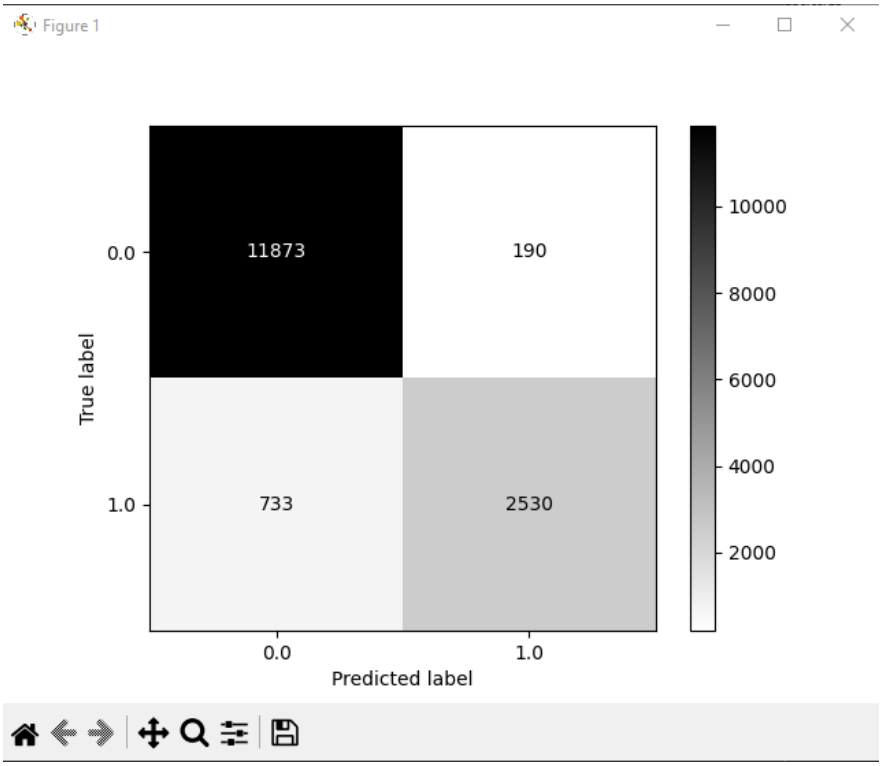

Рисунок 6.3 – Матриця плутанини

Результат роботи функції логістичної регресії зображено на рисунку 6.4.

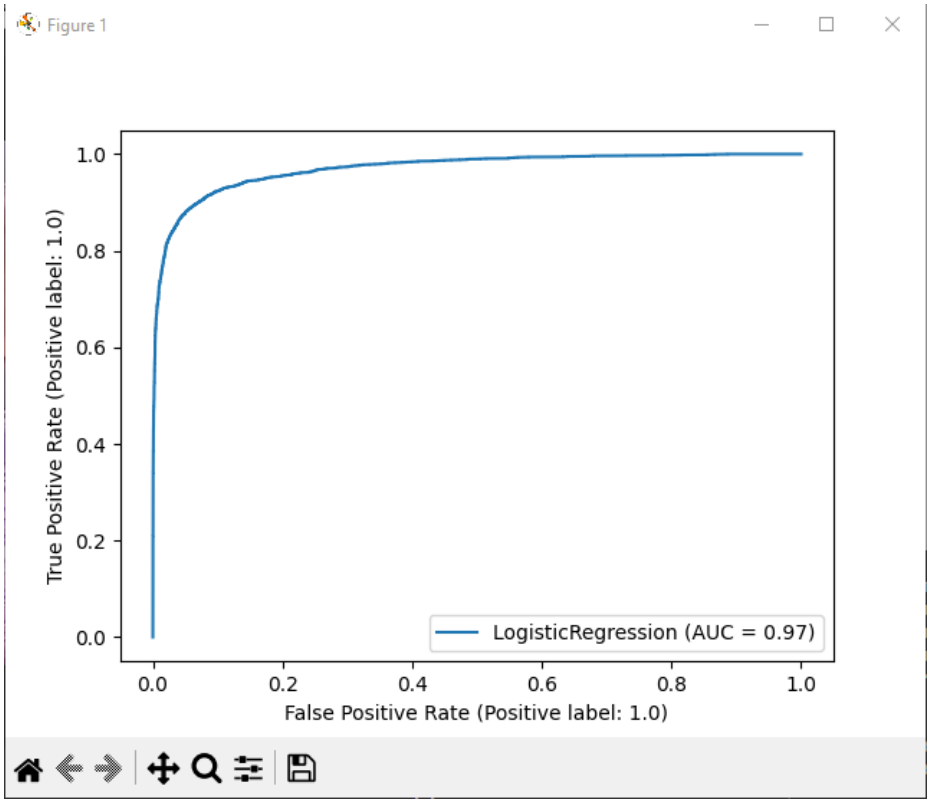

Рисунок 6.4 – Логістична регресія

Area Under Curve (AUC) представляє площу під кривою. Площа під кривою обчислюється для характеристики ефективності моделі класифікації. Чим вище значення AUC, тим краще модель прогнозує 0 як 0 і 1 як 1. Тобто токсичний коментар який в датасеті вказано 1 як 1, нетоксичний коментар 0 як 0. Ефективність моделі дорівнює 97%.

Як тільки навчання буде завершено буде виведено наступний текст (Рис. 6.5), а також повідомленя про успішне підключення до бази даних. Telegram-бот готовий до роботи, та обслуговування користувачів мережі.

```
Data base connected OK
Good
```

Рисунок 6.5 – Робота Telegram-боту

Використовуючи пошук в месенджері користувачам мережі необхідно ввести @TestQToc_bot що дозволить їм знайти його, та виконати його запуск написнути кнопку «Запустити».

Бот може опрацьовувати як безпосередньо повідомлення від користувачів, так і групах в які вони його додадуть.

Після запуску Telegram-бот вивиде наступне повідомдення (Рис. 6.6).

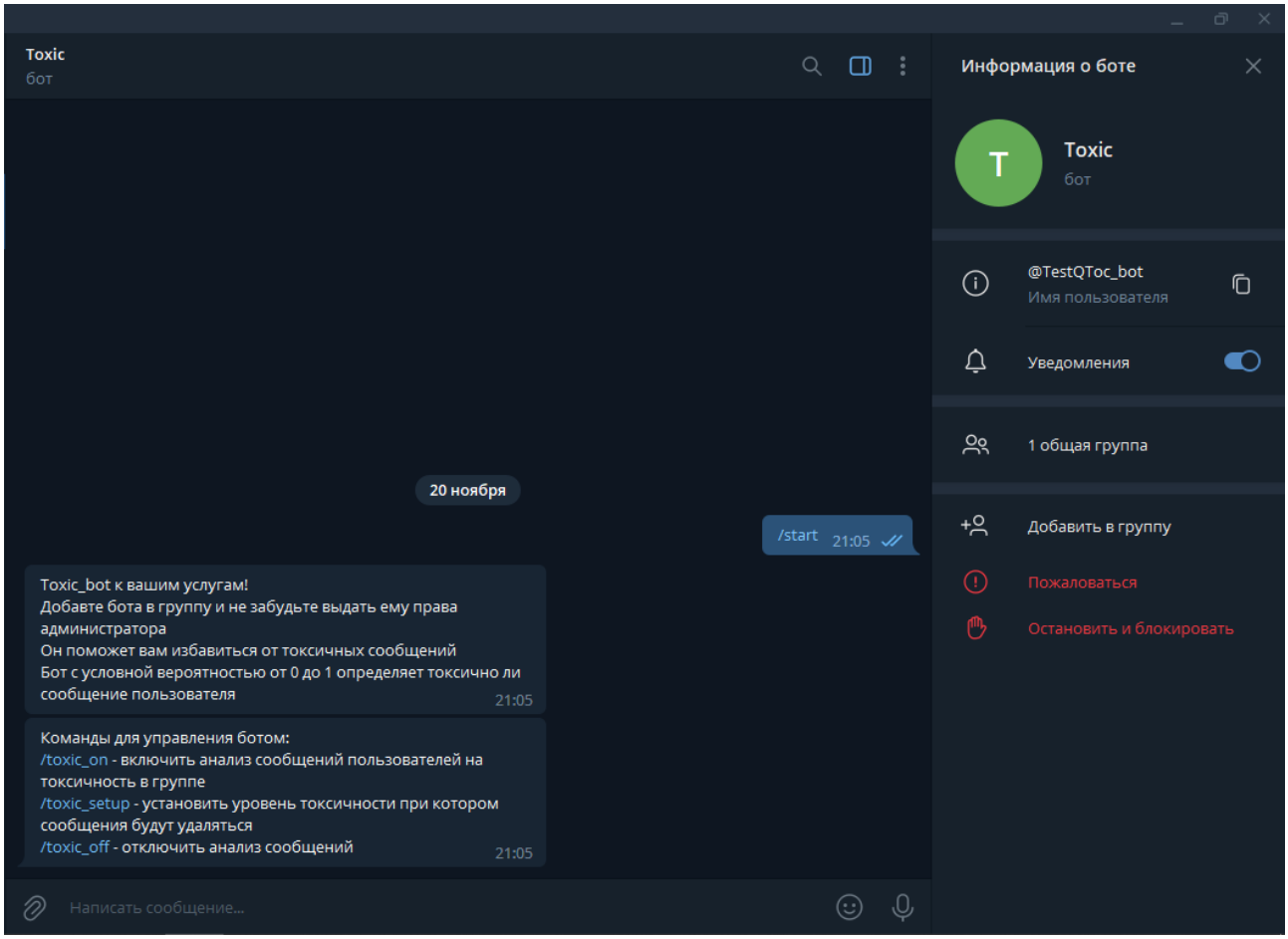

Рисунок 6.6 – Робота Telegram-боту

Для створення групи в месенджері Telegram необхідно обрати пункт «Створити групу» (Рис. 6.7), вказати назву групи (Рис. 6.8), та додати бота до групи (Рис. 6.9).

Для роботи боту в группі необхідні права адміністратора! Надання прав адміністратора можна побачити на рисунках 6.10-6.11.

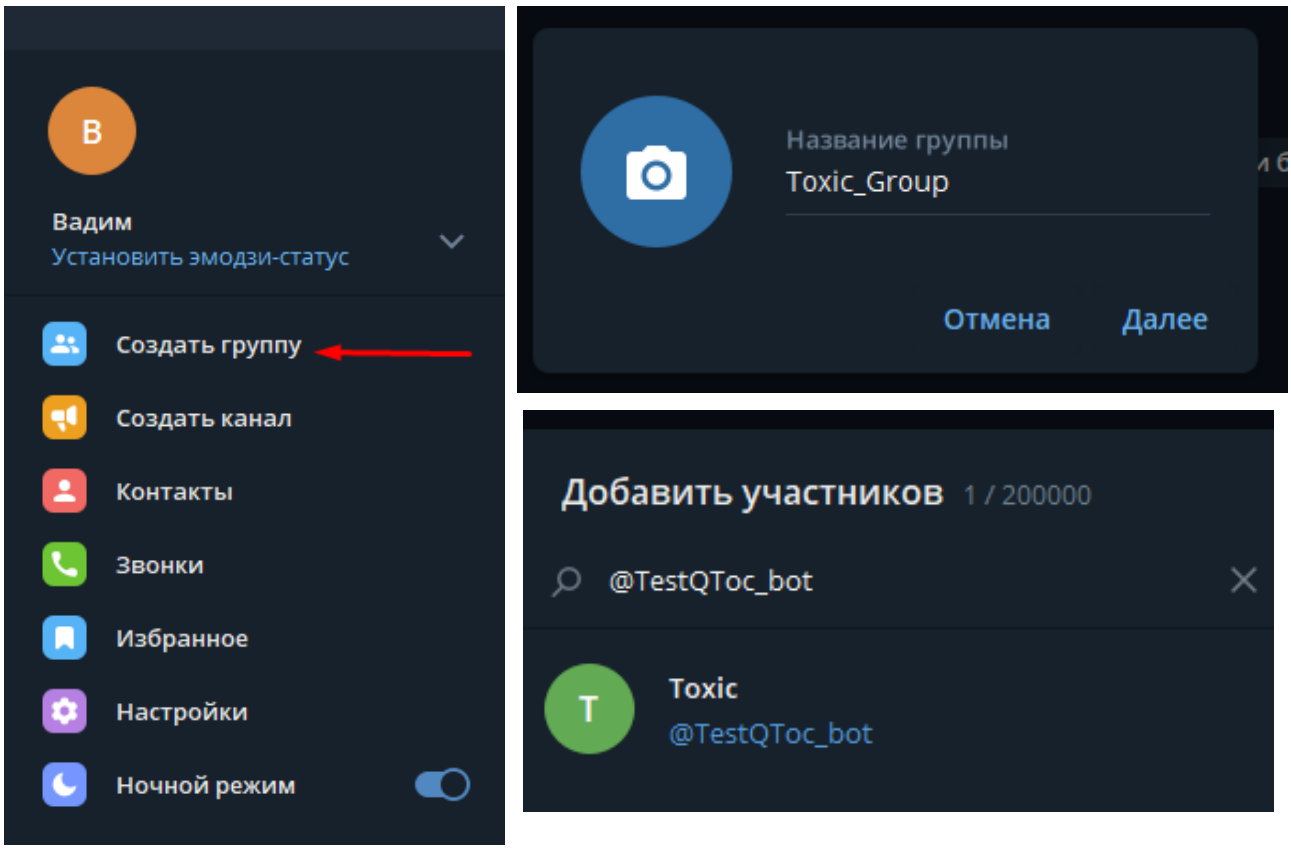

Рисунок 6.7-6.9 – Створення групи

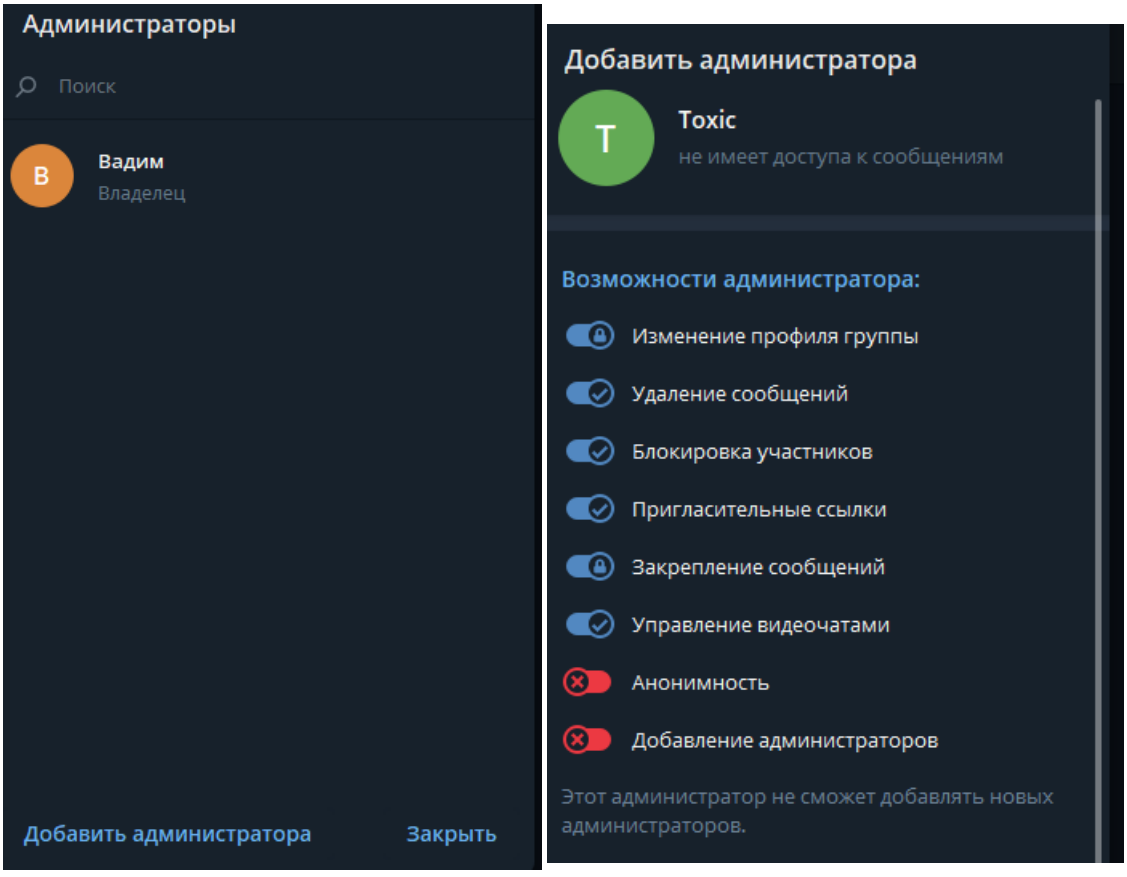

Рисунок 6.10-6.11 – Адміністрування групи

Список користувачів групи та надані права зображені на рисунку 6.12.

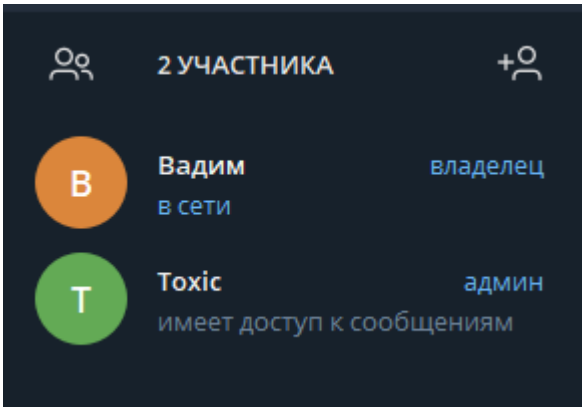

Рисунок 6.12 – Список учасників групи

Керувати ботом може лише адміністратор групи. Для керуванням передбачено декілька команд:

– /start, запускає бот;

– /toxic_on, вмикає функцію перевірки повідомлень на токсичність;

– /toxic_off, вимикає функцію перевірки повідомлень на токсичність;

– /toxic_setup, за допомогою даної команди встановлюється рівень реагування на повідомлення. Якщо токсичність повідомлення виявиться вище встановленого рівня воно негайно буде видалено (Рис. 6.13).

Реагування боту на токсичне повідомлення можна побачити на рисунку 6.14.

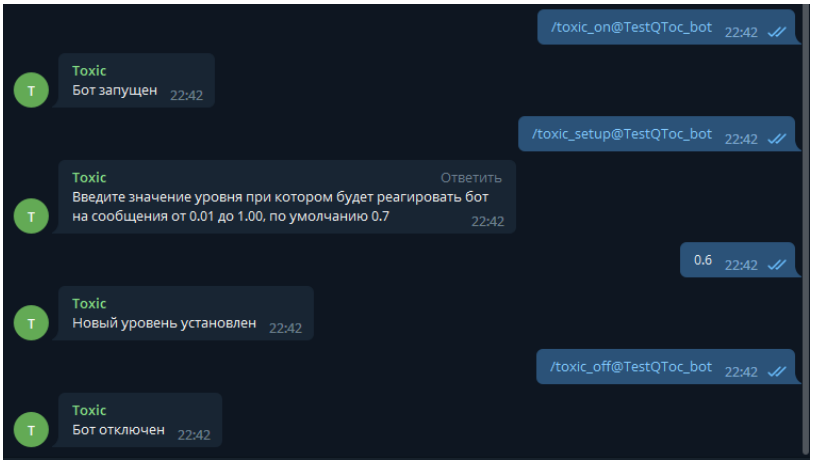

Рисунок 6.13 – Робота Telegram-боту

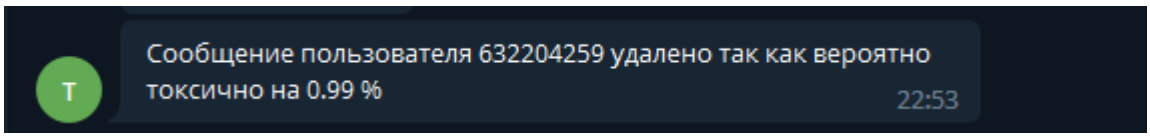

Рисунок 6.14 – Реагування боту на токсичне повідомлення

ВИСНОВКИ

В ході виконання кваліфікаційної роботи розроблено Telegram-бот для аналізу та отримання інформації з телеграм чатів для виявлення токсичних коментарів, який може опрацьовувати коментарі користувачів месенджеру Telegram реальному часі.

Відповідно до мети кваліфікаційної роботи виконано поставлені задачі, а саме:

– виконано аналіз предметної області для виявлення актуальності розробки Telegram-боту для аналізу та отримання коментарів месенджеру Telegram;

– проаналізовано існуючі рішення для виявлення вже створених розробок, їх недоліків та переваг;

– проаналізовано та обрано методи для виявлення токсичних коментарів;

– програмно реалізовано Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– відлагоджено Telegram-бота який зможе отримувати, аналізувати, та відповідно реагувати на повідомлення користувачів;

– створено інструкцію для користувачів, в якій зібрані відомості щодо правильного та ефективного використання Telegram-боту.

Обрані методи Term Frequency-Inverse Document Frequency та логістичної регресії для розробки Telegram-боту мають точність класифікування повідомлень 97%, що є добрим результатом.

Даний бот значно полегшить модерацію коментарів груп Telegram, що зменшить навантаження на адміністраторів, та дасть їм більше часу на створення унікального контенту та підвищить його якість, а також сприятиме шановному спілкуванню користувачів по відношенню один до одного.

Реалізація поставлених цілей виконана за допомогою розробки Telegram-боту на мові програмування Python з використанням IDE PyCharm.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Коротеев, Михаил Викторович. "Обзор некоторых современных тенденций в технологии машинного обучения." E-Management 1.1 (2018): 26-35.
2. What is scikit learn — a beginner guide to popular machine learning Python library [Електронний ресурс] // Radek Fabisiak. – 2712. – Режим доступу до ресурсу: https://www.blog.duomly.com/what-is-scikit-learn-introduction-to-popular-machine-learning-and-data-science-python-library/.
3. Моржов, Сергей Владимирович. "Современные методы детектирования и классификации токсичных комментариев с использованием нейронных сетей." Моделирование и анализ информационных систем 27.1 (2020): 48-61.
4. Androcec, Darko. "Machine learning methods for toxic comment classification: a systematic review." Acta Universitatis Sapientiae, Informatica 12.2 (2020): 205-216.
5. Что такое мешок слов? [Електронний ресурс] // СИСТЕМНЫЙ БЛОКЪ. – 906. – Режим доступу до ресурсу: https://sysblok.ru/glossary/chto-takoe-meshok-slov/.
6. Метрика TF-IDF (Term frequency–inverse document frequency) [Електронний ресурс] – Режим доступу до ресурсу: https://wiki.loginom.ru/articles/tf-idf.html.
7. Вікіпедія Вільна енциклопедія: [Веб-сайт]. URL: https://uk.wikipedia.org/wiki/Python.
8. Sebastian R. Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2 / R. Sebastian, M. Vahid., 2019. – 772 с. – (3).

9. Get started | PyCharm [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://www.jetbrains.com/help/pycharm/quick-start-guide.html#meet.
10. Использование регулярных выражений в Python. [Електронний ресурс] // DOCS-Python.ru. – 2022. – Режим доступу до ресурсу: https://docs-python.ru/standart-library/modul-re-python/.
11. Руководство - Морфологический анализатор pymorphy2 [Електронний ресурс] – Режим доступу до ресурсу: https://pymorphy2.readthedocs.io/en/0.2/user/index.html.
12. Обновляемый progressbar для программ на Python. [Електронний ресурс] // DOCS-Python.ru. – 2022. – Режим доступу до ресурсу: https://docs-python.ru/packages/tqdm-progressbar/.
13. Natural Language Toolkit [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://www.nltk.org/.
14. Python SQLite [Електронний ресурс] // GeeksforGeeks. – 2021. – Режим доступу до ресурсу: https://www.geeksforgeeks.org/python-sqlite/.
15. БАЗИ ДАНИХ ТА СУБД [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://eim.snau.edu.ua/wp-content/uploads/2022/02/%D0%91%D0%94%D1%82%D0%B0-%D0%A1%D0%A3%D0%91%D0%94-%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B2%D0%BA%D0%B0%D0%B7%D1%96%D0%B2%D0%BA%D0%B8.pdf.
16. venv — Creation of virtual environments [Електронний ресурс] // Python Software Foundation. – 2022. – Режим доступу до ресурсу: https://docs.python.org/3/library/venv.html.
17. Отладка Python-кода с PyCharm [Електронний ресурс] – Режим доступу до ресурсу: https://www.jetbrains.com/ru-ru/pycharm/features/debugger.html.
18. Что такое логистическая регрессия? [Електронний ресурс] – Режим доступу до ресурсу: https://aws.amazon.com/ru/what-is/logistic-regression/.

19. How Does Machine Learning Work? [Електронний ресурс] // Simplilearn - Online Certification Training Course Provider. – 2022. – Режим доступу до ресурсу: https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-machine-learning.