Визначення та опис проблеми.

***Проблема:***

Аналіз та обробка текстової інфомації за допомогою API та алгоритмів *Natural Language Processing* для виявленння тональності тексту(Sentimental analysis) та пошуку схожої за змістом інформації для можливої побудови системи рекомендацій продуктів.

Згідно з IBM Marketing Cloud study, 90% інформації, яка сьогодні знаходиться в Інтернеті, було створено лише за останні 2 роки. Щодня людство генерує неймовірні обсяги інформації, зокрема текстової, створюючи величезну кількість даних про те, що відбувається у світі, що думають люди та як вони реагують на різні події. Усі ці текстові дані є неоціненним ресурсом. Аналізуючи їх, можна вирішувати значну кількість бізнес-проблем. Проте, аналіз такої кількості контенту нелегкий, оскільки перетворення тексту, створеного людьми в структуровану інформацію є складним завданням. З цього постає проблема аналізу тексту та обробки природної мови.

***Аналіз тексту (Text Analysis/Text Mining)*** - це процес вивчення великих колекцій письмових ресурсів для створення нової інформації та перетворення неструктурованого тексту в структуровані дані для подальшого аналізу та використання алгоритмів машинного навчання(Machine Learning).

***Обробка природної мови(Natural Language Processing)*** - це галузь штучного інтелекту(Artificial Intelligance), яка допомагає комп'ютерам розуміти, інтерпретувати людську мову, керувати нею, аналізувати тональність та виділяти важливі частини тексту.

*Основні завдання ОПМ:*

* категоріювання вмісту(content categorization);
* визначення теми та моделювання(topic discovery and modeling);
* контекстне вилучення(contextual extraction);
* аналіз тональності(sentiment analysis).
* конвертація мовлення в текст та перетворення тексту в мовлення(speech-to-text and text-to-speech conversion);
* підсумовування документів(document summarization);
* машинний переклад(machine translation).

*Приклади використання ОПМ:*

* особисті помічники (Siri, Alexa);
* автоматичний перекладач (Skype Translator, Google Translate);
* пошукові системи (Google Search);
* системи рекомендацій продуктів(Amazon, YouTube, Netflix, Twitter).

***Аналіз тональності тексту(Sentiment analysis)***- це процес виявлення емоційної забарвленості тексту, оцінки автора відносно об'єктів, мова про які йде в тексті та чи є контент позитивним, негативним або нейтральним.

Розробка додатків, пов’язаних з аналізом тексту та обробкою природної мови є складною. Людська мова не є високоструктурованою, вона не завжди точна, часто неоднозначна. ЇЇ структура може залежати від багатьох факторів, зокрема граматичних правил, термінів, скорочень, сленгу, діалектів, соціального контексту та синтаксичних помилок.

Опис функціональних можливостей API.

**Musixmatch API**

*Musixmatch API* - сервіс, за допомогою якого, можна отримати доступ до найбільшої в світі бази даних з текстами більш ніж 14 мільйонів пісень на більш ніж 50 різних мовах.

***Функціональні можливості:***

За допомогою методів Musixmatch API можна отримати:

* список кращих виконавців даної країни;
* список найпопулярніших пісень даної країни;
* інформацію про трек з бази даних: назва, виконавець, інструментальний прапорець та обкладинка;
* текст пісні;
* фрагмент певної пісні;
* субтитри композиції;
* синхронізацію для композиції;
* перекладені пісні для певної мови;
* текст композиції на основі назви та виконавця;
* субтитри для пісні, вказавши її назву, виконавця та тривалість;
* інформацію про виконавця;
* дискографію виконавця;
* список виконавців, якимось чином пов'язаних з даним;
* інформацію про альбом: назва, дата релізу, обкладинка;
* список пісень альбому;
* список останніх пісень з оновленою лірикою.

**AYLIEN Text Analysis API**

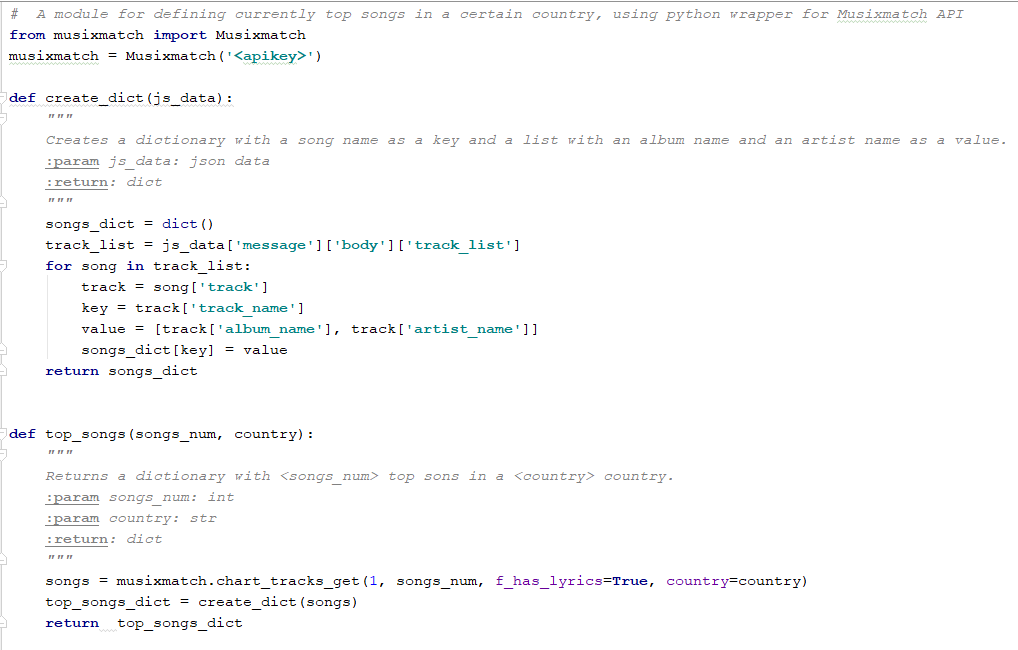
*AYLIEN Text Analysis API* - пакет інформаційного пошуку, машинного навчання та API для природних мов, який дозволяє аналізувати текст та вилучати із нього структуровані дані.

***Функціональні можливості:***

* класифікація тексту;
* аналіз тональності тексту;
* пошук ключових слів;
* визначення концепції;
* підсумовування тексту;
* тегування зображень;
* генерування хештегів;
* визначення мови;
* визначення пов'язаних фраз;
* семантичне маркування.

Приклад використання Musixmatch API за допомогою програми-оболонки.

Для роботи з Musixmatch API доцільно використовувати Python-обгортку pymusixmatch. Наступний приклад - модуль, який використовує цю оболонку для визначення заданої кількості найпопулярніших пісень у певній країні.



1. Встановлюємо обгортку*(pip install pymusixmatch*);
2. Реєструємо новий додаток на <https://developer.musixmatch.com/> , де буде згенеровано унікальний ключ доступу;
3. Імпортуємо musixmatch, замість <apikey> прописуємо отриманий ключ;
4. Функція create\_dict() приймає дані у форматі json та повертає словник, де ключем є назва пісні, а значенням - словник, який містить назву альбому та ім’я виконавця;
5. Функція top\_songs() приймає число - кількість пісень та рядок - назву країни і поветає словник - результат функції create\_dict(). musixmatch.chart\_tracks\_get - метод Python-обгортки для визначення найпопуляніших пісень у країні(повертає дані у форматі json).

Вимоги на систему.

|  |  |  |
| --- | --- | --- |
|  | **Складова частина** | **Опис** |
| 1 | Спонсор проекту (Project Sponsor) | Ініціатор проекту - Жировецька Софія, студентка 1 курсу програми з комп’ютерних наук Українського Католицького Університету. |
| 2 | Бізнес потреба (Business Need) | Покращити системи рекомендацій пісень та виконавців.  Забезпечити користувачам легкий дотуп до інформації про аналіз композицій та дискогафій виконавців. |
| 3 | Бізнес вимоги (Business Requirements) | Надає користувачу інфомацію про емоційну забарвленість пісні  Визначає ключові слова та тематику комозиції  Будує статистику емоційної забавленості дискогафії певного виконавця(позитивна, негативна, нейтральна) та найчастіше вживаних слів  Надає інформацію про виконавця  Шукає схожих виконавців |
| 4 | Питання та обмеження (Special Issues or Constraints) | Граничний термін розробки – 25 травня.  Щотижня повинен бути написаний звіт про виконану роботу.  Гіт-репозиторій проекту: https://github.com/SophiaZhyrovetska/Course\_work.git |