Инструкция к сдаче

1. Настоятельно рекомендуем сдавать практическое задание в виде ссылки на личный репозиторий на github.

2. Рекомендуемый способ организации данных в репозитории: создать отдельные папки по темам и помещать в них отдельные файлы для каждой задачи с правильным расширением.

Ссылка на инструкцию по работе с git и сдачу практики:

https://docs.google.com/document/d/1RAT_ukE39iOfbz1xa39QXae2hBUEZ4U6Fko_wFDdrsM/edit

Ссылка на видеокурс по Git:

https://geekbrains.ru/courses/66

Если остались сложности с системой git, то обратитесь к преподавателю или наставнику.

Тема “Предобработка текста с помощью Python”

Осуществим предобработку данных с Твиттера, чтобы очищенные данные в дальнейшем использовать для задачи классификации. Данный датасет содержит негативные (label = 1) и нейтральные (label = 0) высказывания. Для работы объединим train_df и test_df.

Задания:

1. Удалим @user из всех твитов с помощью паттерна "@[\w]*". Для этого создадим функцию:
    - для того, чтобы найти все вхождения паттерна в тексте, необходимо использовать re.findall(pattern, input_txt)
    - для для замены @user на пробел, необходимо использовать re.sub()
2. Изменим регистр твитов на нижний с помощью .lower().
3. Заменим сокращения с апострофами (пример: ain't, can't) на пробел, используя apostrophe_dict. Для этого необходимо сделать функцию: для каждого слова в тексте проверить (for word in text.split()), если слово есть в словаре apostrophe_dict в качестве ключа (сокращенного слова), то заменить ключ на значение (полную версию слова).
4. Заменим сокращения на их полные формы, используя short_word_dict. Для этого воспользуемся функцией, используемой в предыдущем пункте.
5. Заменим эмотиконы (пример: ":)" = "happy") на пробелы, используя emoticon_dict. Для этого воспользуемся функцией, используемой в предыдущем пункте.
6. Заменим пунктуацию на пробелы, используя re.sub() и паттерн r'[^\w\s]'.
7. Заменим спец. символы на пробелы, используя re.sub() и паттерн r'[^a-zA-Z0-9]'.
8. Заменим числа на пробелы, используя re.sub() и паттерн r'[^a-zA-Z]'.
9. Удалим из текста слова длиной в 1 символ, используя ' '.join([w for w in x.split() if len(w)>1]).
10. Поделим твиты на токены с помощью nltk.tokenize.word_tokenize, создав новый столбец 'tweet_token'.
11. Удалим стоп-слова из токенов, используя nltk.corpus.stopwords. Создадим столбец 'tweet_token_filtered' без стоп-слов.

12. Применим стемминг к токенам с помощью nltk.stem.PorterStemmer. Создадим столбец 'tweet_stemmed' после применения стемминга.
13. Применим лемматизацию к токенам с помощью nltk.stem.wordnet.WordNetLemmatizer. Создадим столбец 'tweet_lemmatized' после применения лемматизации.
14. Сохраним результат предобработки в pickle-файл.