Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет Инфокоммуникационных технологий (ИКТ)

Образовательная программа Мобильные и сетевые технологии

О Т Ч Е Т

по Лабораторной работе

Дисциплина: Методы сетевого анализа.

Специальность: 09.03.03 Прикладная информатика.

Выполнил:

Балцат К. И.,

студент группы К33401

Санкт-Петербург
2023

ЗАДАНИЕ

На практике анализ текстовых данных в социальных сетях может быть полезен для целого ряда задач. К примеру, можно распознавать токсичные посты для их модерации или анализировать тональность текста, чтобы оценить отношение группы людей к тому или иному явлению. В этой лабораторной работе мы будем исследовать пользовательские посты, посвящённые COVID-19

Ход работы

1. Скачайте набор данных Covid 19 Indian Sentiments on covid19 and lockdown по ссылке

Covid 19 Indian Sentiments on covid19 and lockdown

С его помощью мы будем обучать модель распознавать настроение автора поста.

В датафрейме оставьте только те строки, у которых в столбце sentiment написано sad и joy

Для простоты выполнения оставляем только 2 возможных настроения, строки с постами, не соответствующими им, можно удалить.

Примените на этом наборе данных шаги предварительной обработки

К шагам предварительной обработки относятся:

приведение к нижнему регистру,

токенизация,

удаление стоп-слов (обратите внимание на то, что стоп-слова должны соответствовать языку, с которым ведётся работа, в данном случае это английский),

уберите заведомо неинформативные данные (хэштеги и отметки пользователей).

Также необходимо произвести векторизацию.

Пример выполнения этих шагов на другом наборе можно подсмотреть здесь:

Чтобы улучшить качество обучения можно попробовать перед векторизацией также применить лемматизацию (приведение слов к исходной форме). Для этого подойдёт, к примеру, nltk.WordNetLemmatizer (pymorphy2).

Плавное введение в Natural Language Processing (NLP)

Обучите классификатор

Можно использовать реализацию из библиотеки Scikit-learn

Оцените качество классификации. Обучите еще одну или несколько классический моделей классификаторов и сравните качество. Подберите наиболее оптимальные гиперпараметры векторизатора и классификатора, при которых метрики будут выше.

Примените шаги предобработки на наборе данных COVID19 Tweets

С помощью обученного классификатора найдите посты из набора данных COVID19 Tweets, которые можно отнести к классу sad. Какой процент от общего количества постов они составляют?

Проанализируйте тональность постов из набора данных COVID19 Tweets с помощью библиотеки TextBlob (Dostoevsky). Какое процентное соотношение постов разных настроений наблюдается?*

Выгрузка датасета

```
df = pd.read_csv("/content/finalSentimentdata2.csv")
df = df.rename(columns = {'Unnamed: 0':' ID'})
df
```

	ID	sentiment	text
0	3204	sad	agree the poor in india are treated badly thei...
1	1431	joy	if only i could have spent the with this cutie...
2	654	joy	will nature conservation remain a priority in ...
3	2530	sad	coronavirus disappearing in italy show this to...
4	2296	sad	uk records lowest daily virus death toll since...
...	...	...	...
3085	2579	sad	today at 02 30pm a 54 year old bangladeshi mal...
3086	3579	anger	corona virus i implore that you cease activity...
3087	221	joy	issa date once lockdown ends inshaallah (and c...
3088	2705	sad	the death toll due to covid 19 rose to 31 in j...
3089	2962	sad	the rates are become barrier for poor people t...

3090 rows × 3 columns

Разделение данных на тестовые и тренировочные

```
lemma_list = data.lemmatization   # Создание списка лемматизированных токенов
no_list = []
for i in lemma_list:
    no_list.append(", ".join(i))
data['lemma_no_list'] = no_list   # Создание нового столбца для лемматизированных токенов, которые выведены строкой, а не списком
data
```

	ID	sentiment	text	lower	token	no_stop_words	lemmatization	lemma_no_list
0	3204	sad	agree the poor in india are treated badly thei...	agree the poor in india are treated badly thei...	[agree, the, poor, in, india, are, treated, ba...	[agree, poor, india, treated, badly, poors, se...	[agree, poor, india, treated, badly, poor, see...	agree, poor, india, treated, badly, poor, seek...
1	1431	joy	if only i could have spent the with this cutie...	if only i could have spent the with this cutie...	[if, only, i, could, have, spent, the, with, t...	[could, spent, cutie, vc, sakshi__s, n, g, h, ...	[could, spent, cutie, vc, sakshi__s, n, g, h, ...	could, spent, cutie, vc, sakshi__s, n, g, h, c...
2	654	joy	will nature conservation remain a priority in ...	will nature conservation remain a priority in ...	[will, nature, conservation, remain, a, priori...	[nature, conservation, remain, priority, post,...	[nature, conservation, remain, priority, post,...	nature, conservation, remain, priority, post, ...
3	2530	sad	coronavirus disappearing in italy show this to...	coronavirus disappearing in italy show this to...	[coronavirus, disappearing, in, italy, show, t...	[coronavirus, disappearing, italy, show, ``, i...	[coronavirus, disappearing, italy, show, ``, i...	coronavirus, disappearing, italy, show, ``, in...
4	2296	sad	uk records lowest daily virus death toll since...	uk records lowest daily virus death toll since...	[uk, records, lowest, daily, virus, death, tol...	[uk, records, lowest, daily, virus, death, tol...	[uk, record, lowest, daily, virus, death, toll...	uk, record, lowest, daily, virus, death, toll,...
...	...	...	...	...	...	...	...	...
3083	2194	joy	it was tough to see you go brother excellent 6...	it was tough to see you go brother excellent 6...	[it, was, tough, to, see, you, go, brother, ex...	[tough, see, go, brother, excellent, 60, days,...	[tough, see, go, brother, excellent, 60, day, ...	tough, see, go, brother, excellent, 60, day, t...
3085	2579	sad	today at 02 30pm a 54 year old bangladeshi mal...	today at 02 30pm a 54 year old bangladeshi mal...	[today, at, 02, 30pm, a, 54, year, old, bangla...	[today, 02, 30pm, 54, year, old, bangladeshi, ...	[today, 02, 30pm, 54, year, old, bangladeshi, ...	today, 02, 30pm, 54, year, old, bangladeshi, m...
3087	221	joy	issa date once lockdown ends inshaallah (and c...	issa date once lockdown ends inshaallah (and c...	[issa, date, once, lockdown, ends, inshaallah,...	[issa, date, lockdown, ends, inshaallah, (, co...	[issa, date, lockdown, end, inshaallah, (, cor...	issa, date, lockdown, end, inshaallah, (, coro...
3088	2705	sad	the death toll due to covid 19 rose to 31 in j...	the death toll due to covid 19 rose to 31 in j...	[the, death, toll, due, to, covid, 19, rose, t...	[death, toll, due, covid, 19, rose, 31, jammu,...	[death, toll, due, covid, 19, rose, 31, jammu,...	death, toll, due, covid, 19, rose, 31, jammu, ...

Обучение многоклассового Баейсовского наивного классификатора

https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html

```
[17] clf = MultinomialNB(force_alpha=True, alpha=0)                    # Импорт классификатора
     clf.fit(vectorized_x_train, y_train)                              # Обучение классификатора на векторизованной тренировочной выборке
     pred = clf.predict(vectorized_x_test)                             # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred))
     print(classification_report(y_test, pred))
```

```
Точность предсказаний равна 0.6819672131147541
              precision    recall  f1-score   support

         joy       0.62      0.90      0.74       150
         sad       0.83      0.47      0.60       155

    accuracy                           0.68       305
   macro avg       0.73      0.69      0.67       305
weighted avg       0.73      0.68      0.67       305
```

Попытка улучшения результатов

+ Code + Text

```
[18] clf = MultinomialNB(force_alpha=False, alpha=1)                   # Импорт нового классификатора с включекнным сглаживанием Лапласса
     clf.fit(vectorized_x_train, y_train)                              # Обучение классификатора на векторизованной тренировочной выборке
     pred = clf.predict(vectorized_x_test)                             # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred))
     print(classification_report(y_test, pred))
```

```
Точность предсказаний равна 0.8360655737704918
              precision    recall  f1-score   support

         joy       0.90      0.75      0.82       150
         sad       0.79      0.92      0.85       155

    accuracy                           0.84       305
   macro avg       0.85      0.83      0.83       305
weighted avg       0.85      0.84      0.83       305
```

Обучение классификатора алгоритма случайного леса

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

```
clf1 = ensemble.RandomForestClassifier(n_estimators=1522, random_state=0, criterion="gini")  # Импорт классификатора с количеством деревьев, равным количеству строк в дата
clf1.fit(vectorized_x_train, y_train)                                                        # Обучение классификатора на векторизованной тренировочной выборке
pred1 = clf1.predict(vectorized_x_test)                                                      # Получение предсказаний для векторизованной тестовый выборки
print("Точность предсказаний равна", accuracy_score(y_test, pred1))
print(classification_report(y_test, pred1))
```

```
Точность предсказаний равна 0.819672131147541
              precision    recall  f1-score   support

         joy       0.75      0.95      0.84       150
         sad       0.93      0.70      0.80       155

    accuracy                           0.82       305
   macro avg       0.84      0.82      0.82       305
weighted avg       0.84      0.82      0.82       305
```

Попытка улучшения результатов

```
[20] clf1 = ensemble.RandomForestClassifier(n_estimators=100, random_state=0, criterion="gini")  # Импорт нового классификатора с количеством деревьев, равным 100
     clf1.fit(vectorized_x_train, y_train)                                                       # Обучение классификатора на векторизованной тренировочной выборке
     pred1 = clf1.predict(vectorized_x_test)                                                     # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred1))
     print(classification_report(y_test, pred1))
```

```
Точность предсказаний равна 0.8065573770491803
              precision    recall  f1-score   support

         joy       0.74      0.94      0.83       150
         sad       0.92      0.68      0.78       155

    accuracy                           0.81       305
   macro avg       0.83      0.81      0.80       305
weighted avg       0.83      0.81      0.80       305
```

Обучение классификатора метода опорных векторов

https://scikit-learn.org/stable/modules/svm.html

```
[23] clf2 = svm.SVC()                                                                        # Импорт классификатора метода опорных векторов
     clf2.fit(vectorized_x_train, y_train)                                                   # Обучение классификатора на векторизованной тренировочной выборке
     pred2 = clf2.predict(vectorized_x_test)                                                 # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred2))
     print(classification_report(y_test, pred2))
```

```
Точность предсказаний равна 0.7540983606557377
              precision    recall  f1-score   support

         joy       0.67      0.99      0.80       150
         sad       0.98      0.53      0.69       155

    accuracy                           0.75       305
   macro avg       0.82      0.76      0.74       305
weighted avg       0.83      0.75      0.74       305
```

Попытка улучшения результатов

```
LinearSVC: clf2
sklearn.svm._classes.LinearSVC instance
```

```
clf2.fit(vectorized_x_train, y_train)                                                        # Импорт нового классификатора линейного метода опорных векторов
pred2 = clf2.predict(vectorized_x_test)                                                      # Обучение классификатора на векторизованной тренировочной выборке
print("Точность предсказаний равна", accuracy_score(y_test, pred2))                         # Получение предсказаний для векторизованной тестовый выборки
print(classification_report(y_test, pred2))
```

```
Точность предсказаний равна 0.8229508196721311
              precision    recall  f1-score   support

         joy       0.78      0.89      0.83       150
         sad       0.88      0.75      0.81       155

    accuracy                           0.82       305
   macro avg       0.83      0.82      0.82       305
weighted avg       0.83      0.82      0.82       305
```

Обучение классификатора метода ближайших соседей

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

```
[25] clf3 = KNeighborsClassifier(n_neighbors=5)                          # Импорт классификатора с количеством соседей, равным 5
     clf3.fit(vectorized_x_train, y_train)                               # Обучение классификатора на векторизованной тренировочной выборке
     pred3 = clf3.predict(vectorized_x_test)                             # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred3))
     print(classification_report(y_test, pred3))
```

```
Точность предсказаний равна 0.49836065573770494
              precision    recall  f1-score   support

         joy       0.49      0.98      0.66       150
         sad       0.62      0.03      0.06       155

    accuracy                           0.50       305
   macro avg       0.56      0.51      0.36       305
weighted avg       0.56      0.50      0.35       305
```

Попытка улучшения результатов

```
[26] clf3 = KNeighborsClassifier(n_neighbors=3)                          # Импорт классификатора с количеством соседей, равным 3
     clf3.fit(vectorized_x_train, y_train)                               # Обучение классификатора на векторизованной тренировочной выборке
     pred3 = clf3.predict(vectorized_x_test)                             # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred3))
     print(classification_report(y_test, pred3))
```

```
Точность предсказаний равна 0.5245901639344263
              precision    recall  f1-score   support

         joy       0.51      0.89      0.65       150
         sad       0.61      0.17      0.27       155

    accuracy                           0.52       305
   macro avg       0.56      0.53      0.46       305
weighted avg       0.56      0.52      0.46       305
```

Обучение классификатора метода градиентного бустинга

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

```
[29] clf4 = ensemble.GradientBoostingClassifier()                        # Импорт классификатора
     clf4.fit(vectorized_x_train, y_train)                               # Обучение классификатора на векторизованной тренировочной выборке
     pred4 = clf4.predict(vectorized_x_test)                             # Получение предсказаний для векторизованной тестовый выборки
     print("Точность предсказаний равна", accuracy_score(y_test, pred4))
     print(classification_report(y_test, pred4))
```

```
Точность предсказаний равна 0.819672131147541
              precision    recall  f1-score   support

         joy       0.77      0.91      0.83       150
         sad       0.90      0.73      0.80       155

    accuracy                           0.82       305
   macro avg       0.83      0.82      0.82       305
weighted avg       0.83      0.82      0.82       305
```

Попытка улучшения результатов

```
[30] accuracy = 0
     for i in np.arange(0.0, 1.05, 0.05):
       clf4 = ensemble.GradientBoostingClassifier(learning_rate=i, random_state=0)
       clf4.fit(vectorized_x_train, y_train)
       pred4 = clf4.predict(vectorized_x_test)
       new_accuracy = accuracy_score(y_test, pred4)
       if new_accuracy > accuracy:
         accuracy = new_accuracy
         state = i

     print(accuracy)
     print(state)
```

```
0.8459016393442623
0.3000000000000004
```

Результат предобработки второго текста

```
final = covid[covid['full_clear'].str.contains(',')]  # Оставляем только те тексты, в которых имеется как минимум два слова (то есть строка содержит запятую)
final
```

	text	full_clear
0	If I smelled the scent of hand sanitizers toda...	smelled, scent, hand, sanitizers, today, someo...
1	Hey @Yankees @YankeesPR and @MLB - wouldn't it...	hey, would, made, sense, player, pay, respect
2	@diane3443 @wdunlap @realDonaldTrump Trump nev...	trump, never, claimed, hoax, claim, effort
3	@brookbanktv The one gift #COVID19 has give me...	one, gift, give, appreciation, simple, thing, ...
4	25 July : Media Bulletin on Novel #CoronaVirus...	25, july, medium, bulletin, novel
...	...	...
2759	Where are people going to on holidays? Eimer H...	people, going, holiday, eimer, hannon, talk, e...
2760	#MadhyaPradesh CM #ChouhanShivraj tests positi...	cm, test, positive
2761	Maha: COVID-19 hospital overcharges patients; ...	maha, covid19, hospital, overcharge, patient, ...
2762	Over 160k jobs stand to be lost in @WesternCap...	160k, job, stand, lost, due
2763	You need to hear this story! I believe masks d...	need, hear, story, believe, mask, help, u, sta...

2699 rows × 2 columns

```
[54] covid["full_clear"] = cleaned_texts
```

```
sentiments = []                                  # Создаём пустой список для добавления настроений
messages = covid.full_clear[0:100]               # Отбираем первые сто записей (так как долго выполняется)
for mes in tqdm(messages):                       # Создаём цикл, который проходит через каждый текст в списке
  blob = TextBlob(mes, analyzer=NaiveBayesAnalyzer())  # Анализируем настроение
  sentiments.append(blob.sentiment.classification)     # Добавляем результат в список
```

```
100% 100/100 [05:14<00:00, 2.97s/it]
```

```
[56] a = 0                                                                  # Счётчик для 'позитивных' текстов
     b = 0                                                                  # Счётчик для 'негативных' текстов
     for i in range(len(sentiments)):                                       # Создаём цикл, который проходит по каждому настроению из списка, полученного выше
       if sentiments[i] == "pos":                                           # Если настроение позитивное, то
         a += 1                                                             # добавляем единицу к счётчику 'позитивных' текстов
       elif sentiments[i] == "neg":                                         # Если настроение негативное, то
         b += 1                                                             # добавляем единицу к счётчику 'позитивных' текстов
     print(f"Доля постов, отнесенных к классу pos, среди 100 постов, равна {a}%")  # Вывод доли текстов с положительным настроением
     print(f"Доля постов, отнесенных к классу neg, среди 100 постов, равна {b}%")  # Вывод доли текстов с негативным настроением
```

```
Доля постов, отнесенных к классу pos, среди 100 постов, равна 67%
Доля постов, отнесенных к классу neg, среди 100 постов, равна 33%
```

```
[59] sentiments = covid.class_dost                                                  # Создаём список с метками настроений
     neutral = 0                                                                    # Счётчик для 'нейтральных' текстов
     skip = 0                                                                       # Счётчик для пропущенных текстов
     positive = 0                                                                   # Счётчик для 'позитивных' текстов
     speech = 0                                                                     # Счётчик для 'разговорных' текстов без эмоций
     negative = 0                                                                   # Счётчик для 'негативных' текстов
     for i in range(len(sentiments)):                                               # Создаём цикл, который проходит по каждому настроению из списка, п
       if sentiments[i] == "neutral":                                               # Если настроение нейтральное, то
         neutral += 1                                                               # добавляем единицу к счётчику 'нейтральных' текстов
       elif sentiments[i] == "skip":                                                # Если нейросеть пропустила данный текст, то
         skip += 1                                                                  # добавляем единицу к счётчику пропущенных текстов
       elif sentiments[i] == "positive":                                            # Если настроение позитивное, то
         positive += 1                                                              # добавляем единицу к счётчику 'позитивных' текстов
       elif sentiments[i] == "speech":                                              # Если настроение беззэмоциональное, то
         speech += 1                                                                # добавляем единицу к счётчику 'разговорных' текстов без эмоций
       elif sentiments[i] == "negative":                                            # Если настроение негативное, то
         negative += 1                                                              # добавляем единицу к счётчику 'негативных' текстов
     print(f"Доля текстов с нейтральным настроением равна {round(neutral / covid.shape[0], 5)} %")
     print(f"Доля пропущенных текстов равна {round(skip / covid.shape[0], 5)} %")
     print(f"Доля текстов с позитивным настроением равна {round(positive / covid.shape[0], 5)} %")
     print(f"Доля текстов без эмоций равна {round(Decimal(speech / covid.shape[0]), 5)} %")
     print(f"Доля текстов с негативным настроением равна {round(Decimal(negative / covid.shape[0]), 5)} %")
```

```
Доля текстов с нейтральным настроением равна 0.99863 %
Доля пропущенных текстов равна 0.00113 %
Доля текстов с позитивным настроением равна 0.00017 %
Доля текстов без эмоций равна 0.00005 %
Доля текстов с негативным настроением равна 0.00002 %
```

ВЫВОД

Я выполнил лабораторную работу и на практике освоил методы анализа сетей.