Наташа Ермолаева, БКЛ152

Отчет по домашнему заданию по препроцессингу.

Baseline

Изначально в ходе решения задачи по определению тональности твитов были использованы `CountVectorizer` и `TfidfVectorizer`, которым был передан исходный текст (без каких-либо предварительных преобразований), а затем, в качестве классификатора, была применена логистическая регрессия, и получены следующие результаты:

	CountVectorizer	**TfidfVectorizer**
Макросредняя **F1** мера	0.463064212113	0.517616688856
Микросредняя **F1** мера	0.638753651412	0.671372930867

В последствии использовался `TfidfVectorizer`.

Затем, чтобы повысить результаты работы, была проведена нормализация, в том числе очистка текста (от ссылок, сочетаний *@username*, а также от знаков препинания), токенизация (как очищенного, так и исходного текста), лемматизация (а затем и стемминг), удаление стоп-слов (список со стоп-словами был взят **здесь**). Однако очистка данных и удаление стоп-слов не дало ощутимого прироста в результатах работы алгоритма (таблица ниже). Возможно, это связано с тем, что некоторые знаки препинания влияют непосредственно на тональность того или иного твита. То же можно сказать и относительно стоп-слов.

Затем был произведен перебор параметров `(C и penalty)` с помощью `GridSearchCV`. Лучшими параметрами оказались `{'C': 10, 'penalty': ‘l2’}`, которые и были использованы в дальнейшей работе.

До перебора (**`TfidfVectorizer + LogisticRegression`**):

	Очистка данных	**Стоп-слова**	**Очистка данных + стоп-слова**	**Без очистки данных и удаления стоп-слов**
Макро средн **F1** мера	0.55538623173	0.506772934394	0.503947955111	0.541085803258
Микро средн **F1** мера	0.69376825706	0.657740993184	0.649951314508	0.684031158715

После:

TfidfVectorizer + LogisticRegression(C=10, penalty='l2')

	Очистка данных + гиперпараметры	Стоп-слова + гиперпараметры	Очистка данных + стоп-слова + гиперпараметры	Без очистки данных и удаления стоп-слов
Макро средн **F1** мера	0.60066893699	0.580613304041	0.583832099676	0.603415121571
Микро средн **F1** мера	0.69571567673	0.675754625122	0.666017526777	0.704479065239

Таким образом, лучший результат достигается, если мы передаем в `TfidfVectorizer` исходный текст (без очистки от знаков и стоп-слов), а затем используем `LogisticRegression(C=10, penalty='l2')`.

Также неплохо работает замена лемматизации на стемминг:

TfidfVectorizer + SnowballStemmer('russian') + LogisticRegression(C=10, penalty='l2')

```
Макросредняя F1 мера -  0.602751632321
Микросредняя F1 мера -  0.70593962999
```

А параметр `ngram_range=(1, 5)` для `TfidfVectorizer` не улучшил результат работы:

```
Макросредняя F1 мера -  0.557303532116
Микросредняя F1 мера -  0.682570593963
```

Важные признаки:

```
Значимые слова для класса -  -1
['оштрафовать', 'говно', 'tele2', 'не', 'сбой', 'восстановление',
'гавный', 'заблокировать', 'прекратить', 'повысить']

Значимые слова для класса -  0
['восстановление', 'доллар', 'иа', 'гавный', 'оштрафовать',
'говно', 'ловить', 'даже', 'любить', 'beeline_rus']

Значимые слова для класса -  1
```

```
['любить', 'спасибо', 'защита', 'узбекистан', 'радовать',
'бесплатный', 'beeline_rus', 'благодарить', 'подарок', ‘приятно’]
```

Нетрудно заметить, что некоторые слова являются значимыми сразу для нескольких классов (пересекаются между собой -1 и 0, а также 0 и 1). Именно поэтому могут возникать проблемы в разграничении классов между собой (и, как следствие, в итоге получаем не очень высокий результат работы).