МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных систем и технологий

Кафедра: «Измерительно-вычислительные комплексы»

Дисциплина: «Методы искусственного интеллекта»

Отчет

по лабораторной работе № 5

по теме: **«Исследование инструментов классификации библиотеки Scikit-learn»**

Выполнил:

студент гр. ИСТбд-42

Тагашев И. И.

Проверил:

к.т.н., доцент

Шишкин В.В.

Ульяновск 2022 г.

Выполнение лабораторной работы по теме:
«Исследование инструментов классификации библиотеки Scikit-learn»

Классификаторы: классификация с помощью стохастического градиентного спуска, с помощью опорных векторов и пассивно-агрессивный классификатор

Датасет: «Эмоции» https://www.kaggle.com/datasets/ishantjuyal/emotions-in-text

Для данного датасета для всех классификаторов в качестве целевого столбца выберем столбец-метку «emotion» (значения: «happy», «sadness», «fear», «anger», «love», «surprise»), а в качестве определяющего признака – столбец «текста».

Перед обучением на сырых данных подготовим данные к обучению:), разделим на данные и целевой столбец, с помощью train_test_split разделим данные на обучающую и тестовую выборку, а затем проведем векторизацию данных с помощью TfIdfVectorizer.

```
#загрузка датасета
dataset = pd.read_csv(r"emotions.csv")
#выделение целевого столбца
train_labels = dataset['Emotion']
#разделение датасета на обучающую и тестовую выборки
x_train,  x_test,  y_train,  y_test  =  train_test_split(dataset['Text'],
train_labels, test_size=0.1, random_state=0)
#векторизация данных
tfidf = TfidfVectorizer(stop_words = "english")
tfidf_train = tfidf.fit_transform(x_train)
tfidf_test = tfidf.transform(x_test)
```

```
                    Обучающая выборка
17777    i plan to do so by obtaining an mba and from t...
1593     i proclaim to have lost a bit of my sanity and...
17804              i am feeling depressed cursing my luck
12384                i feel agitated and simply irritated
20600    The women wait anxiously and when the boat ret...
                             ...
13123    i feel like i have a headcold and im groggy an...
19648        i feel its been very successful in doing that
9845     i just feel more resentful and tell myself it ...
10799    i had it in my head as it relates to the workp...
2732     i feel like garbage i cant think about being t...
         Name: Text, Length: 19313, dtype: object
                    Тестовая выборка
9933     i dont really have any details to share but i ...
6698                            i feeling almost defeated
11156    i also feel embarrassed because i can consciou...
19332    i just don t understand the betrayal the lying...
12314    im feeling selfish right now because i want th...
                             ...
9685     i am and feeling total love and acceptance for...
7274     i feel more happy inside on a scale i would say a
8226     i feel been accepted and although sip complian...
20059    Her face lit up with a delighted smile as she ...
20738    Her astonished eyes were taking in his costume...
         Name: Text, Length: 2146, dtype: object
```

Стохастический градиентный спуск

Проведем обучение и оценку модели, используя метод стохастического градиентного спуска:

```
sgd = SGDClassifier (loss='hinge', penalty='l2', alpha=1e-3, random_state=42, max_iter=5, tol=None)
sgd.fit(tfidf_train, y_train)
y_pred = sgd.predict(tfidf_test)
score = accuracy_score(y_test, y_pred)
print(f'Точность SGD-классификатора: {round(score * 100, 2)}%')
```

```
Точность SGD-классификатора: 88.12%
```

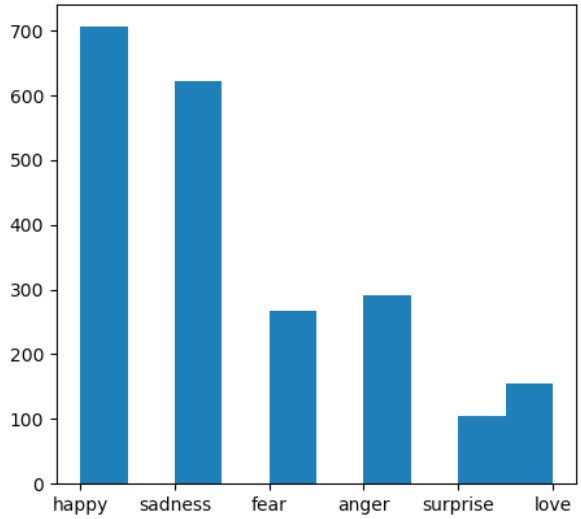
Затем произведем очистку данных (все знаки препинания и ненужные символы ничего не значат для данных, а также проведем лемматизацию):

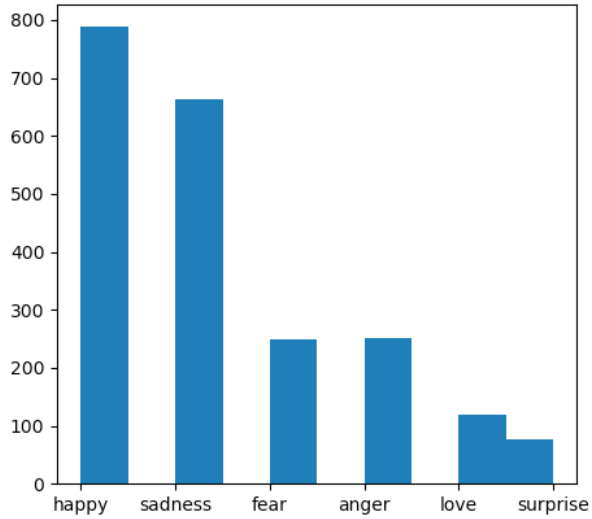
```
def remove_un(data):

    data = re.sub(r'\W', ' ', str(data))
    data = re.sub(r'\s+[a-zA-Z]\s+', ' ', data)
    data = re.sub(r'\^[a-zA-Z]\s+', ' ', data)
    data = re.sub(r'\s+', ' ', data, flags=re.I)
    data = re.sub(r'^b\s+', '', data)
    data = data.lower()
    lemmatizer = WordNetLemmatizer()
    lemmatizer.lemmatize(data)

    return data

#dataset['Text'] = dataset['Text'].apply(remove_un)
```

После очистки данных снова произведем обучение о оценку точности модели:

```
Точность SGD-классификатора: 88.12%
```

Далее произведем визуализацию данных. В данном случае результат удобно показать через гистограммы (первая гистограмма – результаты из тестовой выборки, вторая гистограмма – предсказанный результат):





	Значения тестовой выборки	**Предсказанные значения**
9933	Happy	Happy
6698	Sadness	Sadness
11156	Sadness	Sadness
…	…	…
8226	Happy	Happy
20059	Happy	Happy
20738	Surprise	Happy

Можно заметить, что данный классификатор путает эмоции счастья и удивления, возможно при качественной очистке данных и подготовке к обучению, результат точности классификатора будет выше.

Метод опорных векторов

Проведем обучение и оценку модели, используя логистическую регрессию:

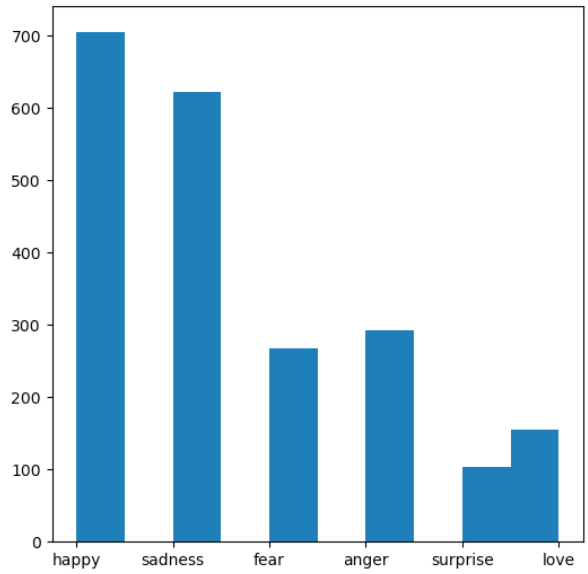
```
svc = LinearSVC()
svc.fit(tfidf_train, y_train)
y_pred = svc.predict(tfidf_test)
score = accuracy_score(y_test, y_pred)
print(f'Точность SVC-классификатора: {round(score * 100, 2)}%')
```

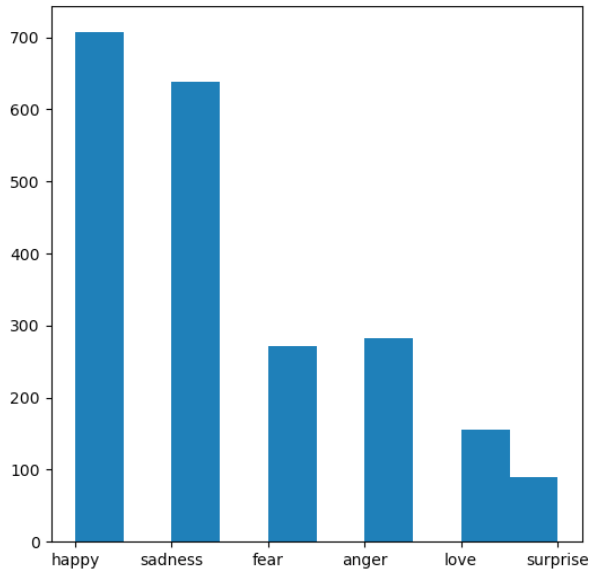
```
Точность SVC-классификатора: 89.7%
```

Затем сделаем очистку данных и снова произведем обучение о оценку точности модели:

```
Точность SVC-классификатора: 89.7%
```

Далее произведем визуализацию данных.





	Значения тестовой выборки	**Предсказанные значения**
9933	Happy	Happy
6698	Sadness	Sadness
11156	Sadness	Sadness
…	…	…
8226	Happy	Happy
20059	Happy	Happy
20738	Surprise	Happy

Данный классификатор показал более высокую точность, чем предыдущий.

Пассивно-агрессивный классификатор

Проведем обучение и оценку модели, используя пассивно-агрессивный классификатор:

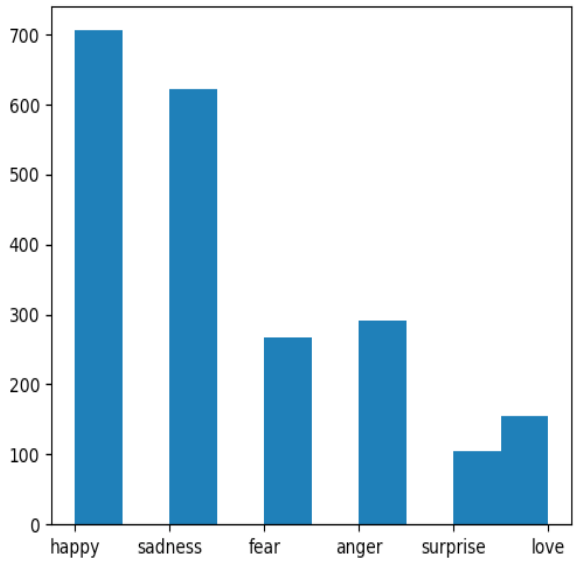
```
pac = PassiveAggressiveClassifier(max_iter = 50)
pac.fit(tfidf_train, y_train)
y_pred = pac.predict(tfidf_test)
score = accuracy_score(y_test, y_pred)
print(f'Точность пассивно-агрессивного классификатора: {round(score * 100,
2)}%')
```

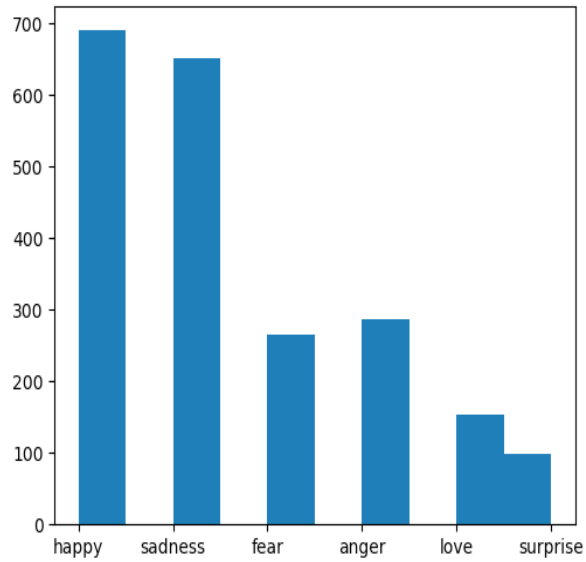
```
Точность пассивно-агрессивного классификатора: 86.53%
```

Затем сделаем очистку данных и снова произведем обучение о оценку точности модели:

```
Точность пассивно-агрессивного классификатора: 86.35%
```

Далее произведем визуализацию данных.





	Значения тестовой выборки	**Предсказанные значения**
9933	Happy	Happy
6698	Sadness	Sadness
11156	Sadness	Sadness
…	…	…
8226	Happy	Happy
20059	Happy	Happy
20738	Surprise	Fear

Можно заметить, что данный классификатор путает эмоции удивления и страха, возможно при качественной очистке данных и подготовке к обучению, результат точности классификатора будет выше.

Вывод

Таким образом, в результате выполнения лабораторной работы мы исследовали классификаторы библиотеки Sklearn. В результате исследования для данного датасета и при проведенной обработке данных наиболее точным оказался метод опорных векторов.