**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»**

Выполнила:

студентка гр. ИСТбд-42

Довлетмурадова А.

Проверил:

к.т.н., доцент

*Шишкин В.В.*

Ульяновск 2022 г.

**Выполнение лабораторной работы по теме:**

**«Исследование инструментов классификации библиотеки Scikit-learn»**

Классификаторы:пассивно-агрессивный классификатор**,** логистическая регрессия и наивный байесовский классификатор.

Датасет: «Fake news» <https://www.kaggle.com/c/fake-news/data?select=test.csv>

Для данного датасета для всех классификаторов в качестве целевого столбца выберем столбец-метку «label», где «0» - новость реальная, «1» - фейковая, а в качестве определяющего признака – столбец «текста».

Перед обучением на сырых данных подготовим данные к обучению: удалим значения Nan (будут мешать преобразованиям), 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('train.csv')

#удаление значений Nan

dataset.dropna(axis=0, how="any", thresh=None, subset=['text'], inplace=True)

#выделение целевого столбца

train\_labels = dataset['label']

#разделение данных на обучающую и тестовую выборку

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', max\_df = 0.7)

tfidf\_train = tfidf.fit\_transform(x\_train)

tfidf\_test = tfidf.transform(x\_test)

Обучающая выборка

8266 NEWARK — Prosecutors opened the case agains...

20720 18 mins ago 2 Views 0 Comments 0 Likes New Z...

1090 Suspect captured in ‘ambush-style’ killings of...

838 The moment Joanna Acevedo first set foot in a ...

17457 Every year, the United States spends $67. 5 bi...

...

13153 Share on Facebook I don't know about you, but ...

19687 A Berkeley student argues that free speech put...

9866 With Justice Neil Gorsuch now seated on the Su...

10821 Leave a reply \nBill Still – Good morning, I’m...

2738 posted by Eddie Toys “R” Us stores in the UK a...

Name: text, Length: 18684, dtype: object

Тестовая выборка

9401 JERUSALEM — “Mr. President!” a woman called...

1474 MILO’s upcoming book DANGEROUS has jumped to t...

4491 NEW DELHI — India’s rapidly worsening air p...

17577 There is nothing presidential (in a dignified ...

777 On Thursday, Republican Senators Ted Cruz ( ) ...

...

16259 BRUSSELS — The United States and the Europe...

17784 WASHINGTON — Over 14 years in American cust...

15910 The House Select Intelligence Committee Friday...

14633 WASHINGTON — Senate Democrats on Tuesday bl...

5159 Bill White November 23, 2016 Are We Seeing The...

Name: text, Length: 2077, dtype: object

**Пассивно-агрессивный классификатор**

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

#обучение на сырых данных

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'Accuracy: {round(score \* 100, 2)}%')

Accuracy: 95.76%

Затем произведем очистку данных (все знаки препинания и ненужные символы, включая ссылки на веб-сайты, ничего не значат для данных):

#очистка данных

def remove\_un(data):

data = data.lower()

data = re.sub('\[.\*?\]','',data)

data = re.sub('\\W',' ',data)

data = re.sub('https?://\S+|www.\S+','',data)

data = re.sub('<.\*?>+','',data)

data = re.sub('[%s]'%re.escape(string.punctuation),'',data)

data = re.sub('\n','',data)

data = re.sub('\w\*\d\w','',data)

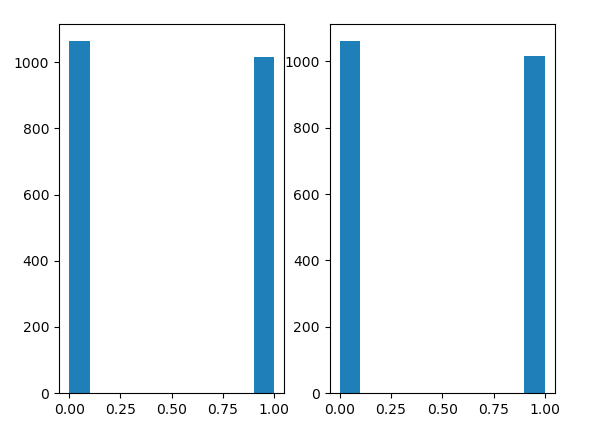
return data

dataset['text']= dataset['text'].apply(remove\_un)

После очистки данных снова произведем обучение о оценку точности модели:

Accuracy: 95.43%

Далее произведем визуализацию данных. В данном случае результат удобно показать через гистограммы (первая гистограмма – распределение реальных и фейковых новостей в тестовой выборке, вторая гистограмма – предсказанный результат для тестовой выборки):



Также представим результаты исследования в табличном виде:

|  |  |  |
| --- | --- | --- |
|  | **Значения тестовой выборки** | **Предсказанные значения** |
| 9401 | 0 | 0 |
| 1474 | 0 | 0 |
| 4491 | 0 | 0 |
| … | … | … |
| 15910 | 0 | 0 |
| 14633 | 0 | 0 |
| 5159 | 1 | 1 |

Таким образом, можно сделать вывод, что данный классификатор обладает высокой точностью прогнозирования результата на данном датасете.

**Логистическая регрессия**

Проведем обучение и оценку модели, используя логистическую регрессию:

lg = LogisticRegression()

lg.fit(tfidf\_train, y\_train)

y\_pred = lg.predict(tfidf\_test)

score = accuracy\_score(y\_test, y\_pred)

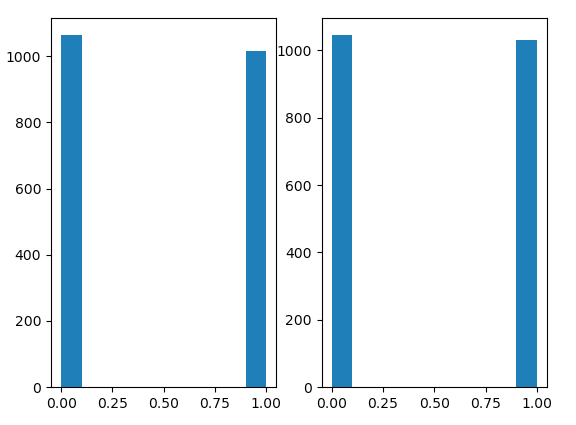
print(f'Accuracy: {round(score \* 100, 2)}%')

Accuracy: 94.99%

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

Accuracy: 94.7%

Далее произведем визуализацию данных. Первая гистограмма – распределение реальных и фейковых новостей в тестовой выборке, вторая гистограмма – предсказанный результат для тестовой выборки:



Также представим результаты исследования в табличном виде:

|  |  |  |
| --- | --- | --- |
|  | **Значения тестовой выборки** | **Предсказанные значения** |
| 9401 | 0 | 0 |
| 1474 | 0 | 0 |
| 4491 | 0 | 0 |
| … | … | … |
| 15910 | 0 | 0 |
| 14633 | 0 | 0 |
| 5159 | 1 | 1 |

Данный классификатор так же показал высокую точность прогнозирования, как и пассивно-агрессивный классификатор.

**Наивный байесовский классификатор**

Проведем обучение и оценку модели, используя наивный байесовский классификатор:

nbc = MultinomialNB()

nbc.fit(tfidf\_train, y\_train)

y\_pred = nbc.predict(tfidf\_test)

score = accuracy\_score(y\_test, y\_pred)

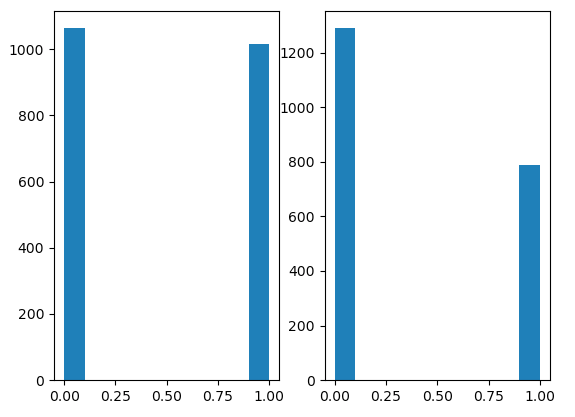
print(f'Accuracy: {round(score \* 100, 2)}%')print(f'Accuracy: {round(score \* 100, 2)}%')

Accuracy: 87.58%

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

Accuracy: 87.67%

Далее произведем визуализацию данных. Первая гистограмма – распределение реальных и фейковых новостей в тестовой выборке, вторая гистограмма – предсказанный результат для тестовой выборки:



Также представим результаты исследования в табличном виде:

|  |  |  |
| --- | --- | --- |
|  | **Значения тестовой выборки** | **Предсказанные значения** |
| 9401 | 0 | 0 |
| 1474 | 0 | 0 |
| 4491 | 0 | 0 |
| … | … | … |
| 15910 | 0 | 0 |
| 14633 | 0 | 0 |
| 5159 | 1 | 1 |

Данный классификатор по сравнению с остальными показал точность ниже.

**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 пассивно-агрессивный классификатор, в то время как менее точным стал многомерный наивный байесовский классификатор. Вероятнее всего такие результаты были получены из-за недостаточной обработки текстовых данных (а именно не была проведена леммография и тд.).